List of orthogonal (or orthonormal) basis vectors.

Examples

```
>>> from sympy import I, Matrix
>>> v = [Matrix([1, I]), Matrix([1, -I])]
>>> Matrix.orthogonalize(*v)
[Matrix([
[1],
[I]]), Matrix([
[ 1],
[-I]])]
```

See also:

MatrixBase.QRdecomposition (page 1287)

References

[R570]

rowspace(*simplify=False*)

Returns a list of vectors that span the row space of M.

Examples

```
>>> from sympy import Matrix
>>> M = Matrix(3, 3, [1, 3, 0, -2, -6, 0, 3, 9, 6])
>>> M
Matrix([
[ 1,  3, 0],
[-2, -6, 0],
[ 3,  9, 6]])
>>> M.rowspace()
[Matrix([[1, 3, 0]]), Matrix([[0, 0, 6]])]
```

MatrixEigen Class Reference

class sympy.matrices.matrices.**MatrixEigen**

Provides basic matrix eigenvalue/vector operations. Should not be instantiated directly. See `eigen.py` for their implementations.

bidiagonal_decomposition(*upper=True*)

Returns $(U, B, V.H)$ for

$$A = UBV^H$$

where A is the input matrix, and B is its Bidiagonalized form

Note: Bidiagonal Computation can hang for symbolic matrices.

Parameters

upper : bool. Whether to do upper bidiagnalization or lower.

True for upper and False for lower.

References

[R571], [R572]

bidiagonalize(*upper=True*)

Returns B, the Bidiagonalized form of the input matrix.

Note: Bidiagonal Computation can hang for symbolic matrices.

Parameters

upper : bool. Whether to do upper bidiagnalization or lower.

True for upper and False for lower.

References

[R573], [R574]

diagonalize(*reals_only=False*, *sort=False*, *normalize=False*)

Return (P, D), where D is diagonal and

D = P^-1 * M * P

where M is current matrix.

Parameters

reals_only : bool. Whether to throw an error if complex numbers are need

to diagonalize. (Default: False)

sort : bool. Sort the eigenvalues along the diagonal. (Default: False)

normalize : bool. If True, normalize the columns of P. (Default: False)

Examples

```
>>> from sympy import Matrix
>>> M = Matrix(3, 3, [1, 2, 0, 0, 3, 0, 2, -4, 2])
>>> M
Matrix([
[1,  2, 0],
[0,  3, 0],
[2, -4, 2]])
>>> (P, D) = M.diagonalize()
>>> D
Matrix([
[1, 0, 0],
[0, 2, 0],
```

(continues on next page)

(continued from previous page)

```
[0, 0, 3]])
>>> P
Matrix([
[-1, 0, -1],
[ 0, 0, -1],
[ 2, 1,  2]])
>>> P.inv() * M * P
Matrix([
[1, 0, 0],
[0, 2, 0],
[0, 0, 3]])
```

See also:

is_diagonal (page 1337), *is_diagonalizable* (page 1241)

eigenvals(*error_when_incomplete=True*, ***flags*)

Compute eigenvalues of the matrix.

Parameters

error_when_incomplete : bool, optional

If it is set to `True`, it will raise an error if not all eigenvalues are computed. This is caused by `roots` not returning a full list of eigenvalues.

simplify : bool or function, optional

If it is set to `True`, it attempts to return the most simplified form of expressions returned by applying default simplification method in every routine.

If it is set to `False`, it will skip simplification in this particular routine to save computation resources.

If a function is passed to, it will attempt to apply the particular function as simplification method.

rational : bool, optional

If it is set to `True`, every floating point numbers would be replaced with rationals before computation. It can solve some issues of `roots` routine not working well with floats.

multiple : bool, optional

If it is set to `True`, the result will be in the form of a list.

If it is set to `False`, the result will be in the form of a dictionary.

Returns

eigs : list or dict

Eigenvalues of a matrix. The return format would be specified by the key `multiple`.

Raises

MatrixError

If not enough roots had got computed.

NonSquareMatrixError

If attempted to compute eigenvalues from a non-square matrix.

Examples

```
>>> from sympy import Matrix
>>> M = Matrix(3, 3, [0, 1, 1, 1, 0, 0, 1, 1, 1])
>>> M.eigenvals()
{-1: 1, 0: 1, 2: 1}
```

Notes

Eigenvalues of a matrix A can be computed by solving a matrix equation $\det(A - \lambda I) = 0$

It's not always possible to return radical solutions for eigenvalues for matrices larger than $4, 4$ shape due to Abel-Ruffini theorem.

If there is no radical solution is found for the eigenvalue, it may return eigenvalues in the form of *sympy.polys.rootoftools.ComplexRootOf* (page 2431).

See also:

MatrixDeterminant.charpoly (page 1228), *eigenvects* (page 1240)

eigenvects(*error_when_incomplete=True, iszerofunc=<function _iszero>, **flags*)

Compute eigenvectors of the matrix.

Parameters

error_when_incomplete : bool, optional

Raise an error when not all eigenvalues are computed. This is caused by `roots` not returning a full list of eigenvalues.

iszerofunc : function, optional

Specifies a zero testing function to be used in `rref`.

Default value is `_iszero`, which uses SymPy's naive and fast default assumption handler.

It can also accept any user-specified zero testing function, if it is formatted as a function which accepts a single symbolic argument and returns `True` if it is tested as zero and `False` if it is tested as non-zero, and `None` if it is undecidable.

simplify : bool or function, optional

If `True`, `as_content_primitive()` will be used to tidy up normalization artifacts.

It will also be used by the `nullspace` routine.

chop : bool or positive number, optional

If the matrix contains any Floats, they will be changed to Rationals for computation purposes, but the answers will be returned after being evaluated with evalf. The `chop` flag is passed to `evalf`. When `chop=True` a default precision will be used; a number will be interpreted as the desired level of precision.

Returns

ret : [(eigenval, multiplicity, eigenspace), ...]

A ragged list containing tuples of data obtained by `eigenvals` and `nullspace`.

`eigenspace` is a list containing the `eigenvector` for each eigenvalue.

`eigenvector` is a vector in the form of a `Matrix`. e.g. a vector of length 3 is returned as `Matrix([a_1, a_2, a_3])`.

Raises

NotImplementedError

If failed to compute nullspace.

Examples

```
>>> from sympy import Matrix
>>> M = Matrix(3, 3, [0, 1, 1, 1, 0, 0, 1, 1, 1])
>>> M.eigenvects()
[(-1, 1, [Matrix([
[-1],
[ 1],
[ 0]])]), (0, 1, [Matrix([
[ 0],
[-1],
[ 1]])]), (2, 1, [Matrix([
[2/3],
[1/3],
[  1]])])]
```


See also:

eigenvals (page 1239), *MatrixSubspaces.nullspace* (page 1236)

is_diagonalizable(*reals_only=False*, ***kwargs*)

Returns `True` if a matrix is diagonalizable.

Parameters

reals_only : bool, optional

If `True`, it tests whether the matrix can be diagonalized to contain only real numbers on the diagonal.

If `False`, it tests whether the matrix can be diagonalized at all, even with numbers that may not be real.

Examples

Example of a diagonalizable matrix:

```
>>> from sympy import Matrix
>>> M = Matrix([[1, 2, 0], [0, 3, 0], [2, -4, 2]])
>>> M.is_diagonalizable()
True
```

Example of a non-diagonalizable matrix:

```
>>> M = Matrix([[0, 1], [0, 0]])
>>> M.is_diagonalizable()
False
```

Example of a matrix that is diagonalized in terms of non-real entries:

```
>>> M = Matrix([[0, 1], [-1, 0]])
>>> M.is_diagonalizable(reals_only=False)
True
>>> M.is_diagonalizable(reals_only=True)
False
```

See also:

is_diagonal (page 1337), *diagonalize* (page 1238)

property is_indefinite

Finds out the definiteness of a matrix.

Explanation

A square real matrix A is:

- A positive definite matrix if $x^T Ax > 0$ for all non-zero real vectors x.
- A positive semidefinite matrix if $x^T Ax \geq 0$ for all non-zero real vectors x.
- A negative definite matrix if $x^T Ax < 0$ for all non-zero real vectors x.
- A negative semidefinite matrix if $x^T Ax \leq 0$ for all non-zero real vectors x.
- An indefinite matrix if there exists non-zero real vectors x, y with $x^T Ax > 0 > y^T Ay$.

A square complex matrix A is:

- A positive definite matrix if $\text{re}(x^H Ax) > 0$ for all non-zero complex vectors x.
- A positive semidefinite matrix if $\text{re}(x^H Ax) \geq 0$ for all non-zero complex vectors x.
- A negative definite matrix if $\text{re}(x^H Ax) < 0$ for all non-zero complex vectors x.
- A negative semidefinite matrix if $\text{re}(x^H Ax) \leq 0$ for all non-zero complex vectors x.
- An indefinite matrix if there exists non-zero complex vectors x, y with $\text{re}(x^H Ax) > 0 > \text{re}(y^H Ay)$.

A matrix need not be symmetric or hermitian to be positive definite.

- A real non-symmetric matrix is positive definite if and only if $\frac{A+A^T}{2}$ is positive definite.
- A complex non-hermitian matrix is positive definite if and only if $\frac{A+A^H}{2}$ is positive definite.

And this extension can apply for all the definitions above.

However, for complex cases, you can restrict the definition of $\text{re}(x^H A x) > 0$ to $x^H A x > 0$ and require the matrix to be hermitian. But we do not present this restriction for computation because you can check `M.is_hermitian` independently with this and use the same procedure.

Examples

An example of symmetric positive definite matrix:

```
>>> from sympy import Matrix, symbols
>>> from sympy.plotting import plot3d
>>> a, b = symbols('a b')
>>> x = Matrix([a, b])
```

```
>>> A = Matrix([[1, 0], [0, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric positive semidefinite matrix:

```
>>> A = Matrix([[1, -1], [-1, 1]])
>>> A.is_positive_definite
False
>>> A.is_positive_semidefinite
True
```

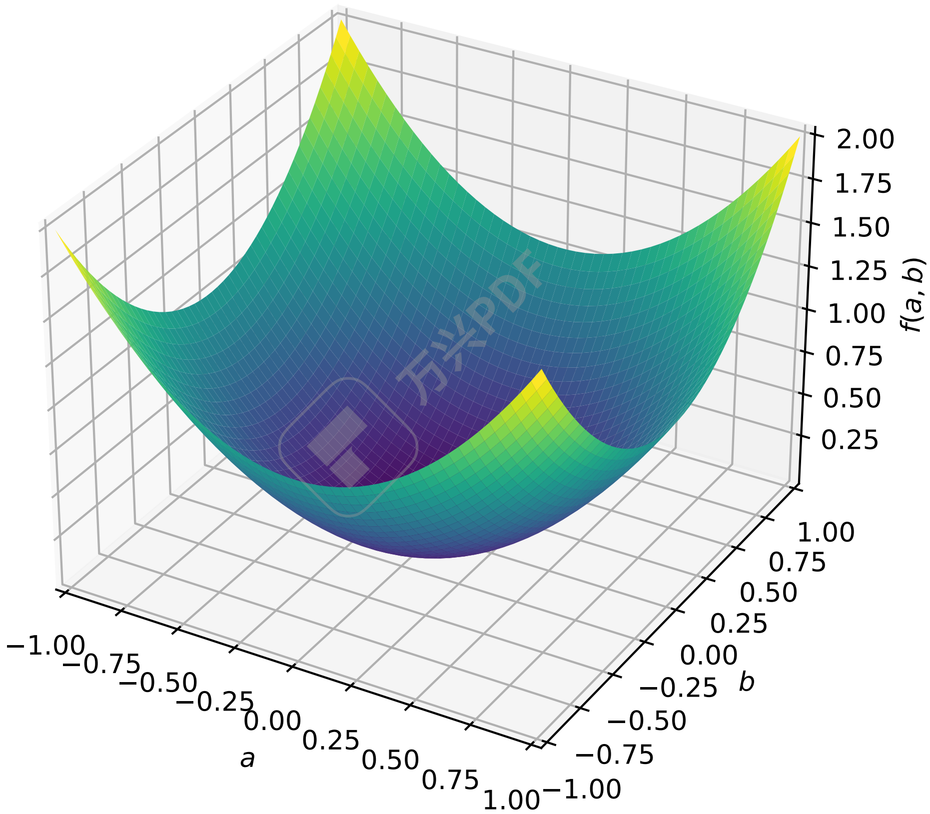
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric negative definite matrix:

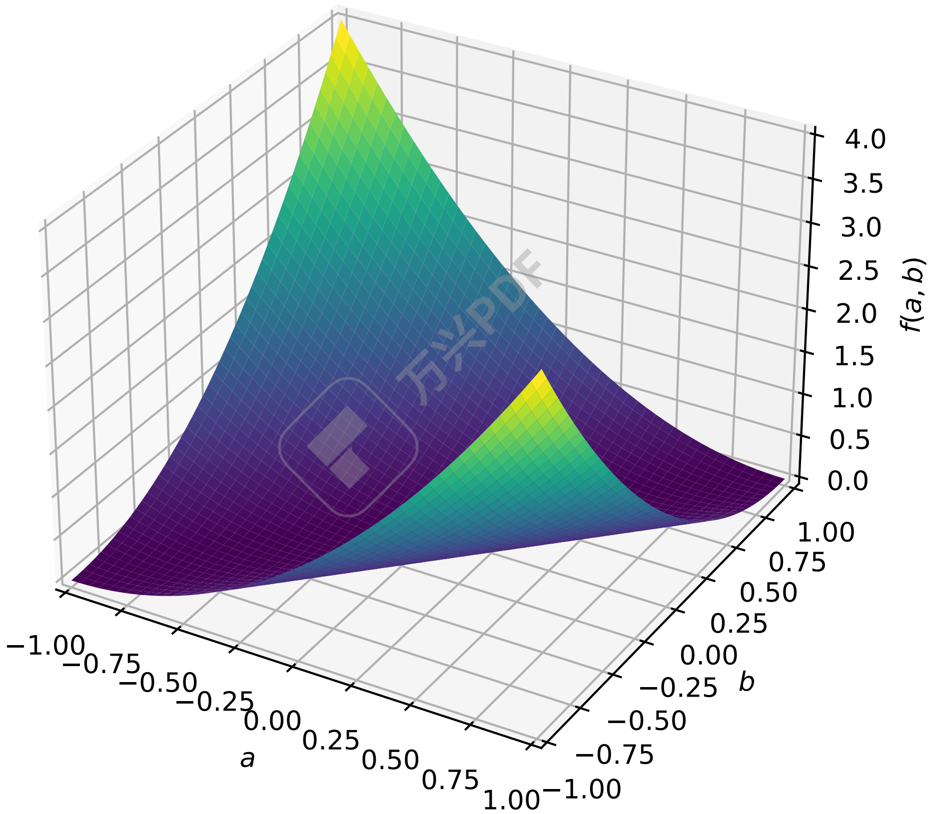
```
>>> A = Matrix([[-1, 0], [0, -1]])
>>> A.is_negative_definite
True
>>> A.is_negative_semidefinite
True
>>> A.is_indefinite
False
```

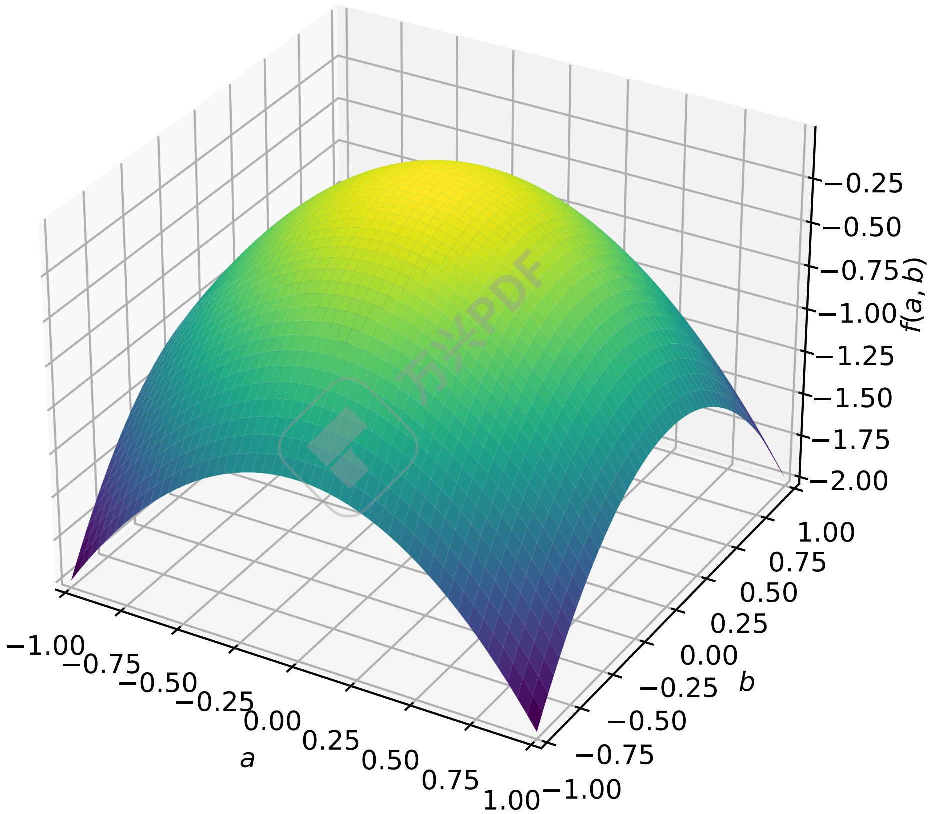
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric indefinite matrix:



−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
b
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
f(a, b)

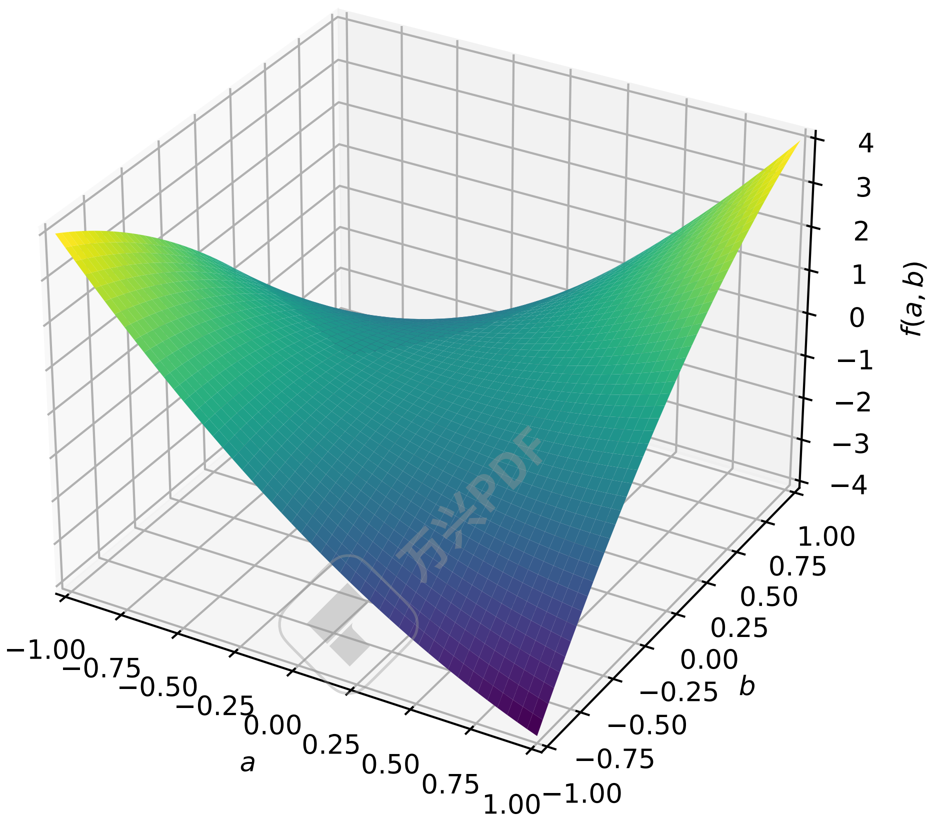

4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b



−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

```
>>> A = Matrix([[1, 2], [2, -1]])
>>> A.is_indefinite
True
```

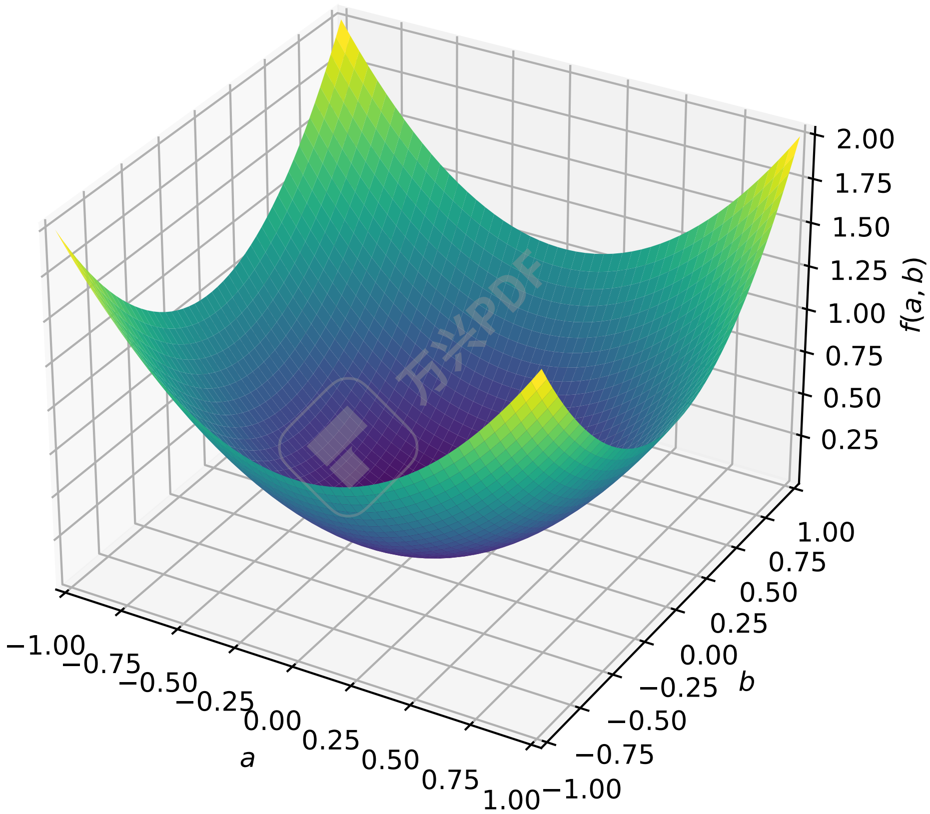
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



An example of non-symmetric positive definite matrix.

```
>>> A = Matrix([[1, 2], [-2, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

Notes

Although some people trivialize the definition of positive definite matrices only for symmetric or hermitian matrices, this restriction is not correct because it does not classify all instances of positive definite matrices from the definition $x^T A x > 0$ or $\text{re}(x^H A x) > 0$.

For instance, `Matrix([[1, 2], [-2, 1]])` presented in the example above is an example of real positive definite matrix that is not symmetric.

However, since the following formula holds true;

$$\text{re}(x^H A x) > 0 \iff \text{re}(x^H \frac{A + A^H}{2} x) > 0$$

We can classify all positive definite matrices that may or may not be symmetric or hermitian by transforming the matrix to $\frac{A+A^T}{2}$ or $\frac{A+A^H}{2}$ (which is guaranteed to be always real symmetric or complex hermitian) and we can defer most of the studies to symmetric or hermitian positive definite matrices.

But it is a different problem for the existance of Cholesky decomposition. Because even though a non symmetric or a non hermitian matrix can be positive definite, Cholesky or LDL decomposition does not exist because the decompositions require the matrix to be symmetric or hermitian.

References

[R575], [R576], [R577]

property is_negative_definite

Finds out the definiteness of a matrix.

Explanation

A square real matrix A is:

- A positive definite matrix if $x^T A x > 0$ for all non-zero real vectors x.
- A positive semidefinite matrix if $x^T A x \geq 0$ for all non-zero real vectors x.
- A negative definite matrix if $x^T A x < 0$ for all non-zero real vectors x.
- A negative semidefinite matrix if $x^T A x \leq 0$ for all non-zero real vectors x.
- An indefinite matrix if there exists non-zero real vectors x, y with $x^T A x > 0 > y^T A y$.

A square complex matrix A is:

- A positive definite matrix if $\text{re}(x^H A x) > 0$ for all non-zero complex vectors x.
- A positive semidefinite matrix if $\text{re}(x^H A x) \geq 0$ for all non-zero complex vectors x.
- A negative definite matrix if $\text{re}(x^H A x) < 0$ for all non-zero complex vectors x.
- A negative semidefinite matrix if $\text{re}(x^H A x) \leq 0$ for all non-zero complex vectors x.
- An indefinite matrix if there exists non-zero complex vectors x, y with $\text{re}(x^H A x) > 0 > \text{re}(y^H A y)$.

A matrix need not be symmetric or hermitian to be positive definite.

- A real non-symmetric matrix is positive definite if and only if $\frac{A+A^T}{2}$ is positive definite.
- A complex non-hermitian matrix is positive definite if and only if $\frac{A+A^H}{2}$ is positive definite.

And this extension can apply for all the definitions above.

However, for complex cases, you can restrict the definition of $\text{re}(x^H Ax) > 0$ to $x^H Ax > 0$ and require the matrix to be hermitian. But we do not present this restriction for computation because you can check `M.is_hermitian` independently with this and use the same procedure.

Examples

An example of symmetric positive definite matrix:

```
>>> from sympy import Matrix, symbols
>>> from sympy.plotting import plot3d
>>> a, b = symbols('a b')
>>> x = Matrix([a, b])
```

```
>>> A = Matrix([[1, 0], [0, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric positive semidefinite matrix:

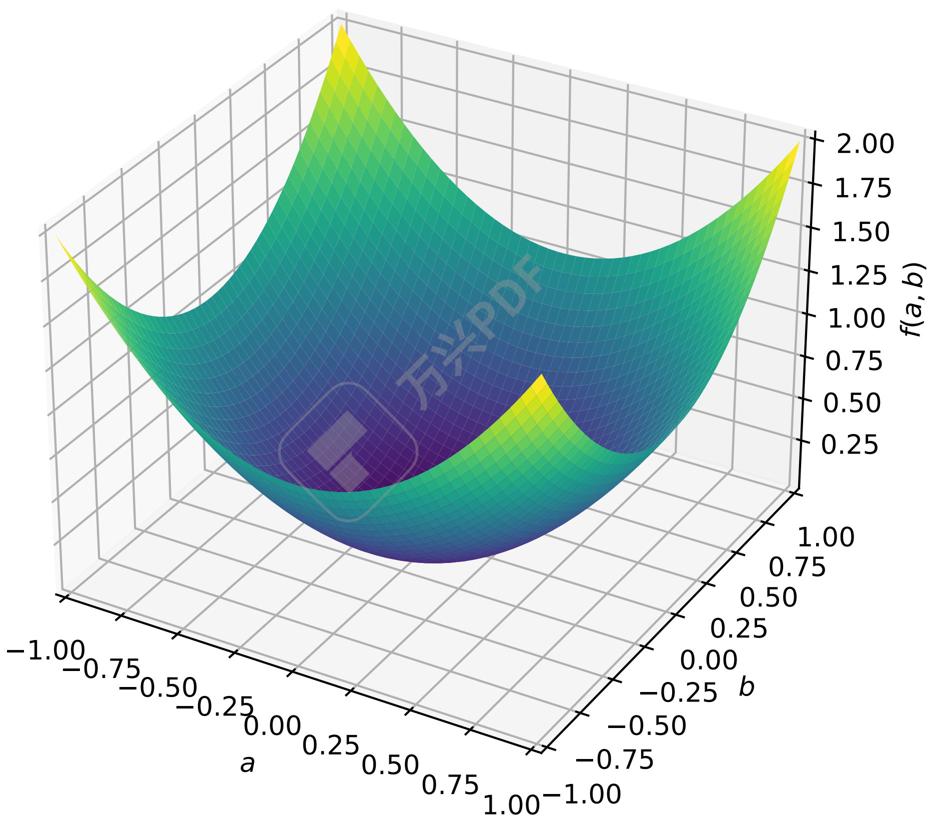
```
>>> A = Matrix([[1, -1], [-1, 1]])
>>> A.is_positive_definite
False
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

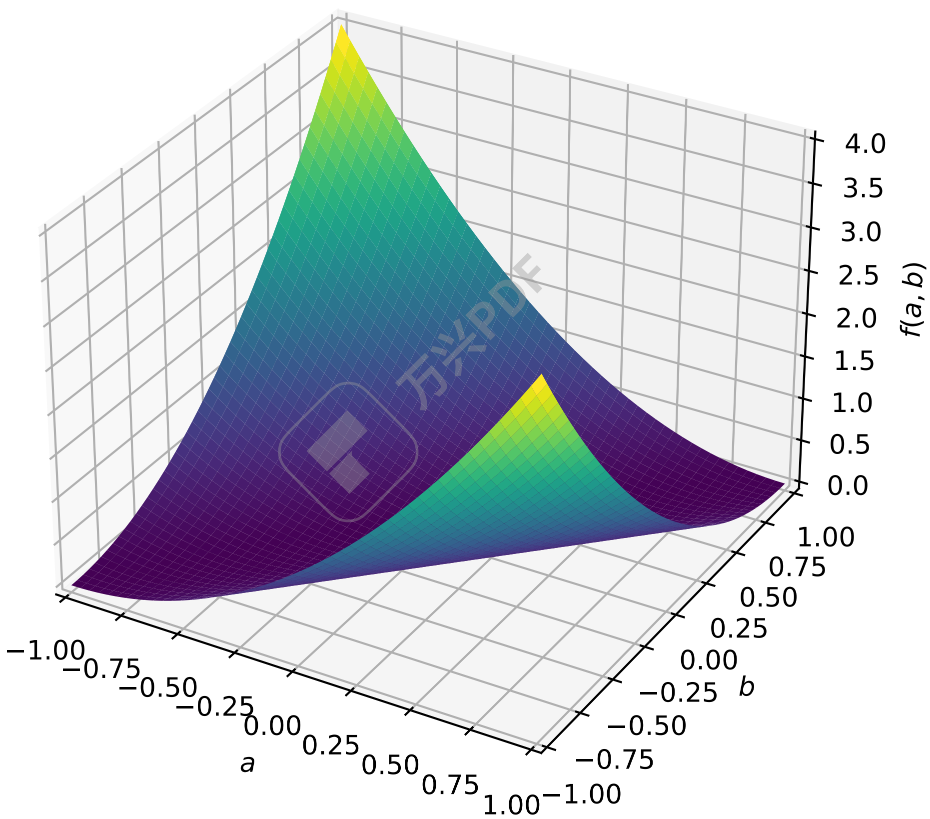
An example of symmetric negative definite matrix:

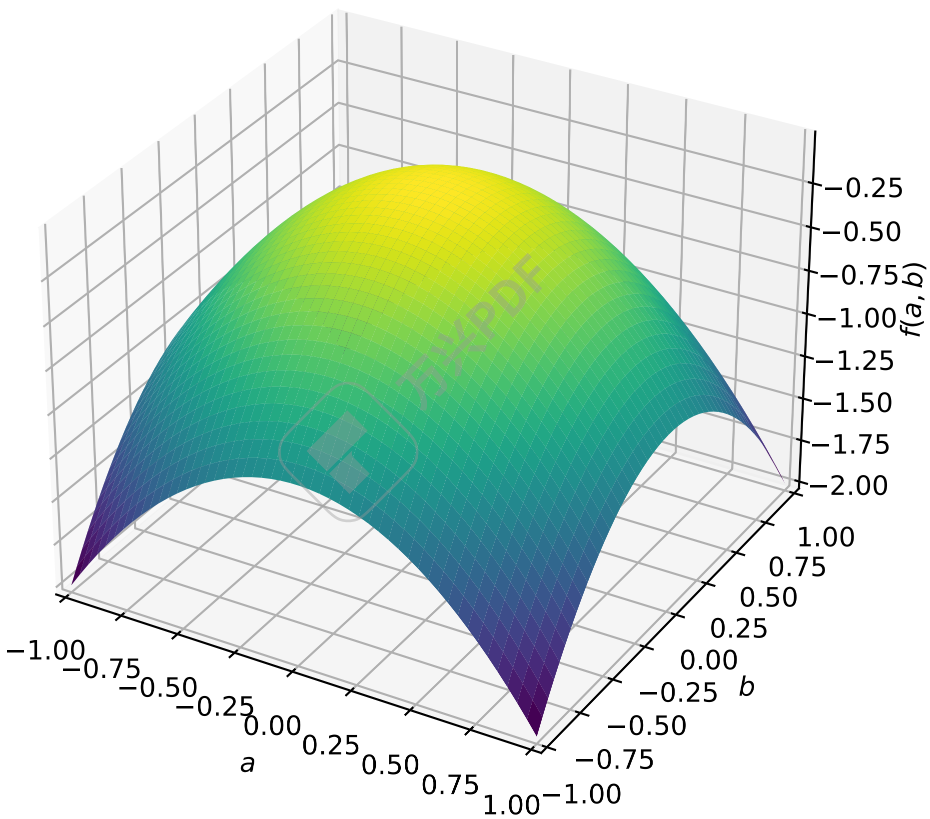
```
>>> A = Matrix([[-1, 0], [0, -1]])
>>> A.is_negative_definite
True
>>> A.is_negative_semidefinite
True
>>> A.is_indefinite
False
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
b


4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

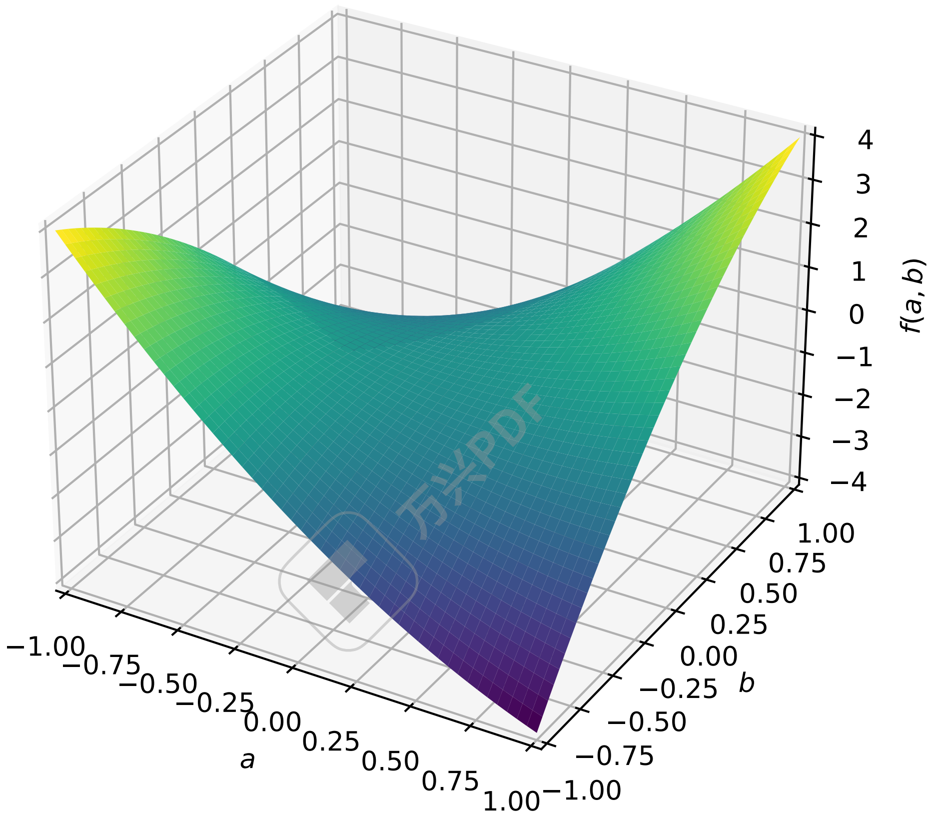


−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
f(a, b)

An example of symmetric indefinite matrix:

```
>>> A = Matrix([[1, 2], [2, -1]])
>>> A.is_indefinite
True
```

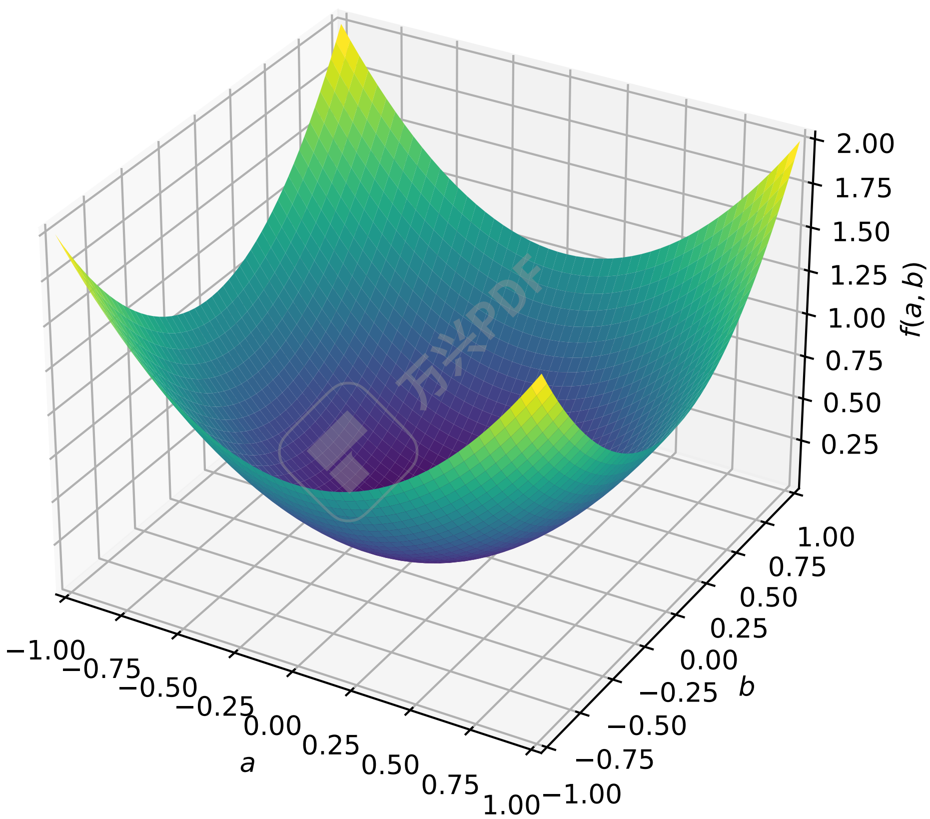
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



An example of non-symmetric positive definite matrix.

```
>>> A = Matrix([[1, 2], [-2, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

Notes

Although some people trivialize the definition of positive definite matrices only for symmetric or hermitian matrices, this restriction is not correct because it does not classify all instances of positive definite matrices from the definition $x^T Ax > 0$ or $\text{re}(x^H Ax) > 0$.

For instance, `Matrix([[1, 2], [-2, 1]])` presented in the example above is an example of real positive definite matrix that is not symmetric.

However, since the following formula holds true;

$$\text{re}(x^H Ax) > 0 \iff \text{re}(x^H \frac{A + A^H}{2} x) > 0$$

We can classify all positive definite matrices that may or may not be symmetric or hermitian by transforming the matrix to $\frac{A+A^T}{2}$ or $\frac{A+A^H}{2}$ (which is guaranteed to be always real symmetric or complex hermitian) and we can defer most of the studies to symmetric or hermitian positive definite matrices.

But it is a different problem for the existance of Cholesky decomposition. Because even though a non symmetric or a non hermitian matrix can be positive definite, Cholesky or LDL decomposition does not exist because the decompositions require the matrix to be symmetric or hermitian.

References

[R578], [R579], [R580]

property is_negative_semidefinite

Finds out the definiteness of a matrix.

Explanation

A square real matrix A is:

- A positive definite matrix if $x^T Ax > 0$ for all non-zero real vectors x.
- A positive semidefinite matrix if $x^T Ax \geq 0$ for all non-zero real vectors x.
- A negative definite matrix if $x^T Ax < 0$ for all non-zero real vectors x.
- A negative semidefinite matrix if $x^T Ax \leq 0$ for all non-zero real vectors x.
- An indefinite matrix if there exists non-zero real vectors x, y with $x^T Ax > 0 > y^T Ay$.

A square complex matrix A is:

- A positive definite matrix if $\text{re}(x^H Ax) > 0$ for all non-zero complex vectors x.
- A positive semidefinite matrix if $\text{re}(x^H Ax) \geq 0$ for all non-zero complex vectors x.
- A negative definite matrix if $\text{re}(x^H Ax) < 0$ for all non-zero complex vectors x.
- A negative semidefinite matrix if $\text{re}(x^H Ax) \leq 0$ for all non-zero complex vectors x.
- An indefinite matrix if there exists non-zero complex vectors x, y with $\text{re}(x^H Ax) > 0 > \text{re}(y^H Ay)$.

A matrix need not be symmetric or hermitian to be positive definite.

- A real non-symmetric matrix is positive definite if and only if $\frac{A+A^T}{2}$ is positive definite.
- A complex non-hermitian matrix is positive definite if and only if $\frac{A+A^H}{2}$ is positive definite.

And this extension can apply for all the definitions above.

However, for complex cases, you can restrict the definition of $\text{re}(x^H Ax) > 0$ to $x^H Ax > 0$ and require the matrix to be hermitian. But we do not present this restriction for computation because you can check `M.is_hermitian` independently with this and use the same procedure.

Examples

An example of symmetric positive definite matrix:

```
>>> from sympy import Matrix, symbols
>>> from sympy.plotting import plot3d
>>> a, b = symbols('a b')
>>> x = Matrix([a, b])
```

```
>>> A = Matrix([[1, 0], [0, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```



```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric positive semidefinite matrix:

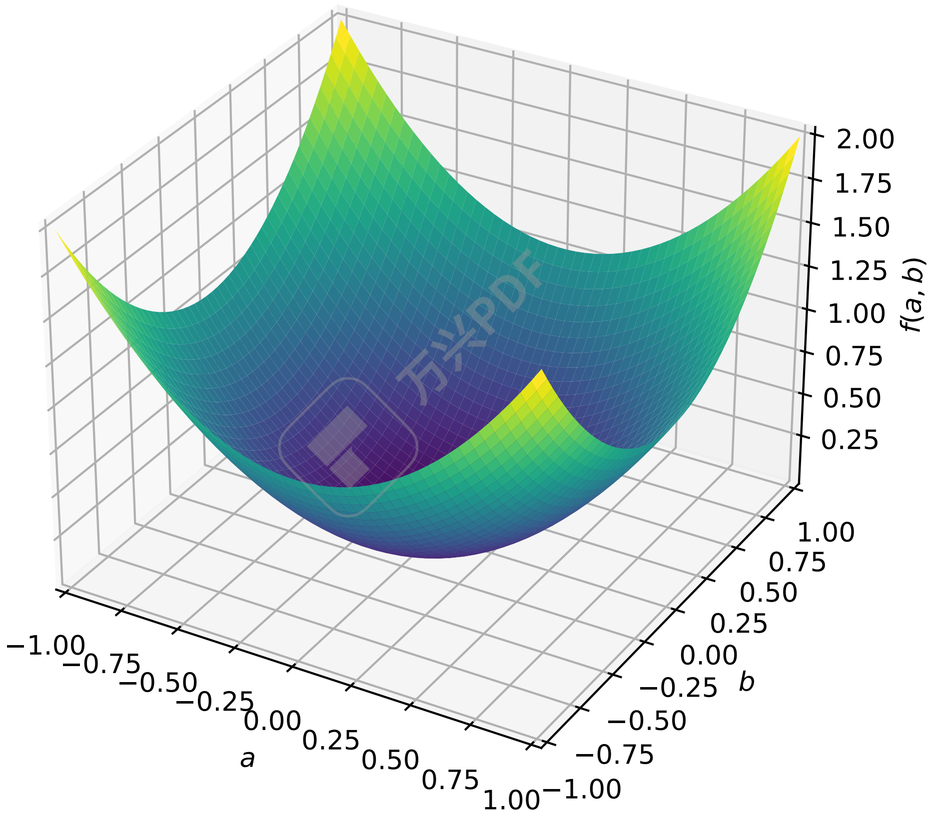
```
>>> A = Matrix([[1, -1], [-1, 1]])
>>> A.is_positive_definite
False
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

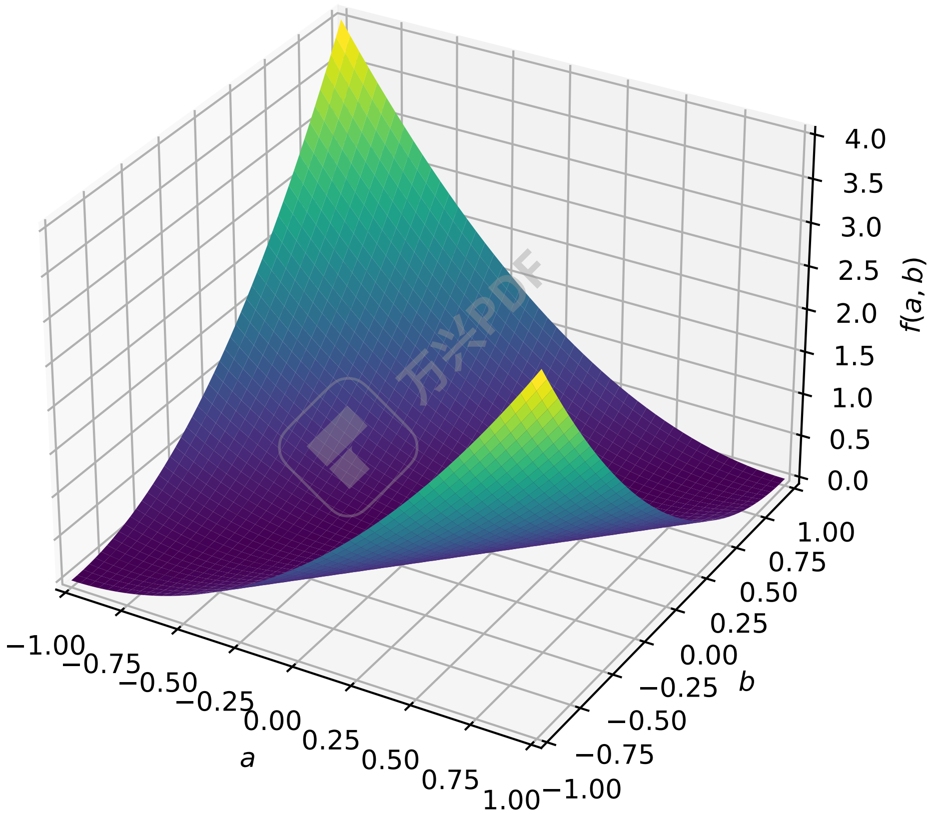
An example of symmetric negative definite matrix:

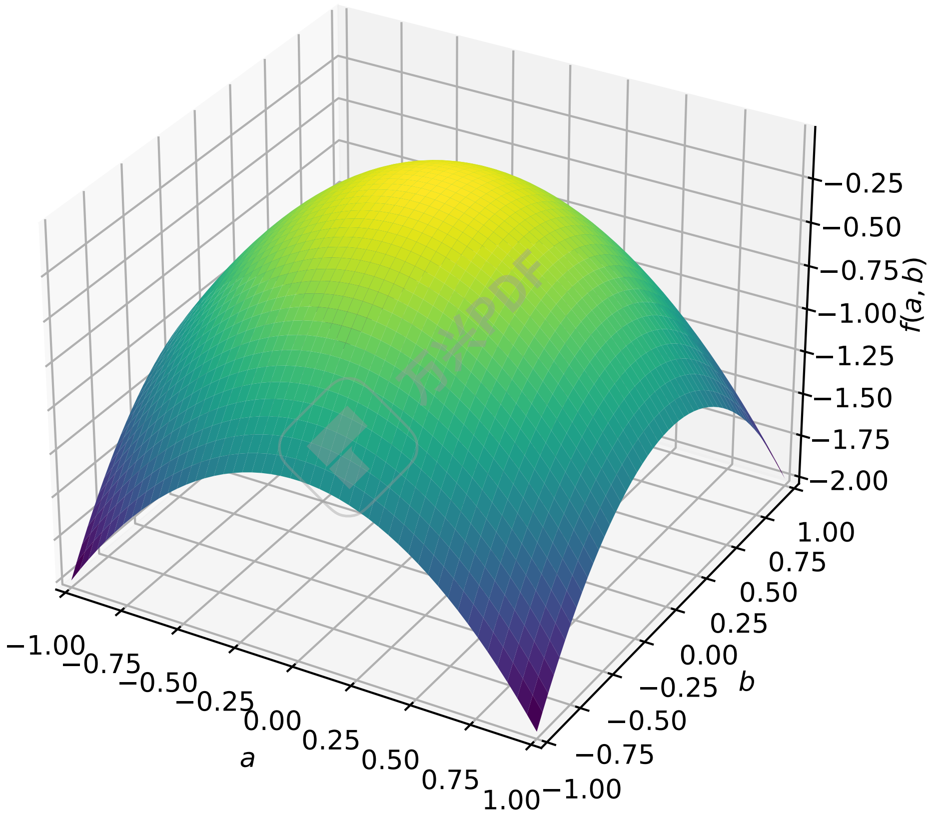
```
>>> A = Matrix([[-1, 0], [0, -1]])
>>> A.is_negative_definite
True
>>> A.is_negative_semidefinite
True
>>> A.is_indefinite
False
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b


f(a, b)
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
a
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
b
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

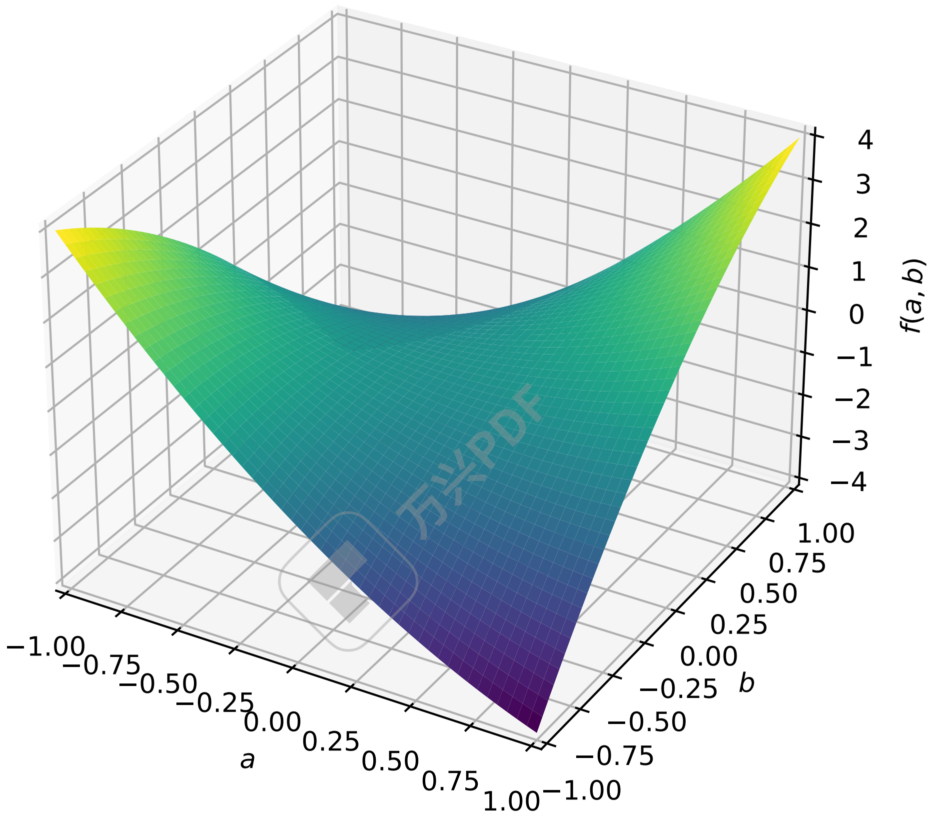


f(a, b)
−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
a
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
b
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00

An example of symmetric indefinite matrix:

```
>>> A = Matrix([[1, 2], [2, -1]])
>>> A.is_indefinite
True
```

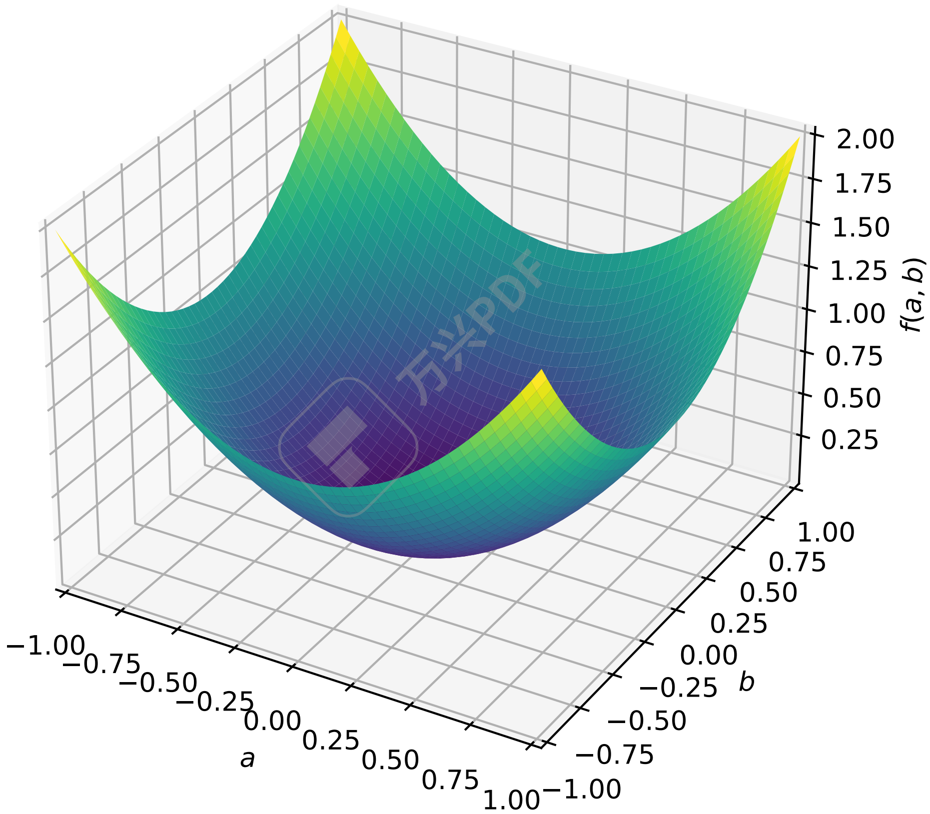
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



An example of non-symmetric positive definite matrix.

```
>>> A = Matrix([[1, 2], [-2, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a

Notes

Although some people trivialize the definition of positive definite matrices only for symmetric or hermitian matrices, this restriction is not correct because it does not classify all instances of positive definite matrices from the definition $x^T Ax > 0$ or $\text{re}(x^H Ax) > 0$.

For instance, `Matrix([[1, 2], [-2, 1]])` presented in the example above is an example of real positive definite matrix that is not symmetric.

However, since the following formula holds true;

$$\text{re}(x^H Ax) > 0 \iff \text{re}(x^H \frac{A + A^H}{2} x) > 0$$

We can classify all positive definite matrices that may or may not be symmetric or hermitian by transforming the matrix to $\frac{A+A^T}{2}$ or $\frac{A+A^H}{2}$ (which is guaranteed to be always real symmetric or complex hermitian) and we can defer most of the studies to symmetric or hermitian positive definite matrices.

But it is a different problem for the existance of Cholesky decomposition. Because even though a non symmetric or a non hermitian matrix can be positive definite, Cholesky or LDL decomposition does not exist because the decompositions require the matrix to be symmetric or hermitian.

References

[R581], [R582], [R583]

property is_positive_definite

Finds out the definiteness of a matrix.

Explanation

A square real matrix A is:

- A positive definite matrix if $x^T Ax > 0$ for all non-zero real vectors x.
- A positive semidefinite matrix if $x^T Ax \geq 0$ for all non-zero real vectors x.
- A negative definite matrix if $x^T Ax < 0$ for all non-zero real vectors x.
- A negative semidefinite matrix if $x^T Ax \leq 0$ for all non-zero real vectors x.
- An indefinite matrix if there exists non-zero real vectors x, y with $x^T Ax > 0 > y^T Ay$.

A square complex matrix A is:

- A positive definite matrix if $\text{re}(x^H Ax) > 0$ for all non-zero complex vectors x.
- A positive semidefinite matrix if $\text{re}(x^H Ax) \geq 0$ for all non-zero complex vectors x.
- A negative definite matrix if $\text{re}(x^H Ax) < 0$ for all non-zero complex vectors x.
- A negative semidefinite matrix if $\text{re}(x^H Ax) \leq 0$ for all non-zero complex vectors x.
- An indefinite matrix if there exists non-zero complex vectors x, y with $\text{re}(x^H Ax) > 0 > \text{re}(y^H Ay)$.

A matrix need not be symmetric or hermitian to be positive definite.

- A real non-symmetric matrix is positive definite if and only if $\frac{A+A^T}{2}$ is positive definite.
- A complex non-hermitian matrix is positive definite if and only if $\frac{A+A^H}{2}$ is positive definite.

And this extension can apply for all the definitions above.

However, for complex cases, you can restrict the definition of $\text{re}(x^H Ax) > 0$ to $x^H Ax > 0$ and require the matrix to be hermitian. But we do not present this restriction for computation because you can check `M.is_hermitian` independently with this and use the same procedure.

Examples

An example of symmetric positive definite matrix:

```
>>> from sympy import Matrix, symbols
>>> from sympy.plotting import plot3d
>>> a, b = symbols('a b')
>>> x = Matrix([a, b])
```

```
>>> A = Matrix([[1, 0], [0, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```



```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric positive semidefinite matrix:

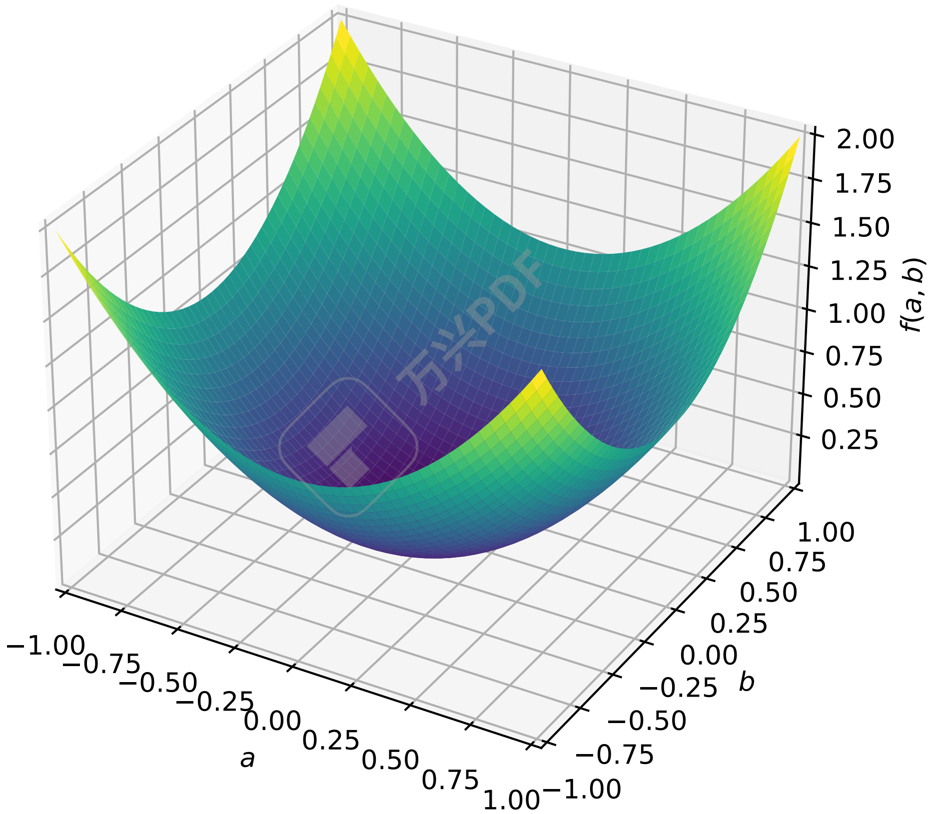
```
>>> A = Matrix([[1, -1], [-1, 1]])
>>> A.is_positive_definite
False
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

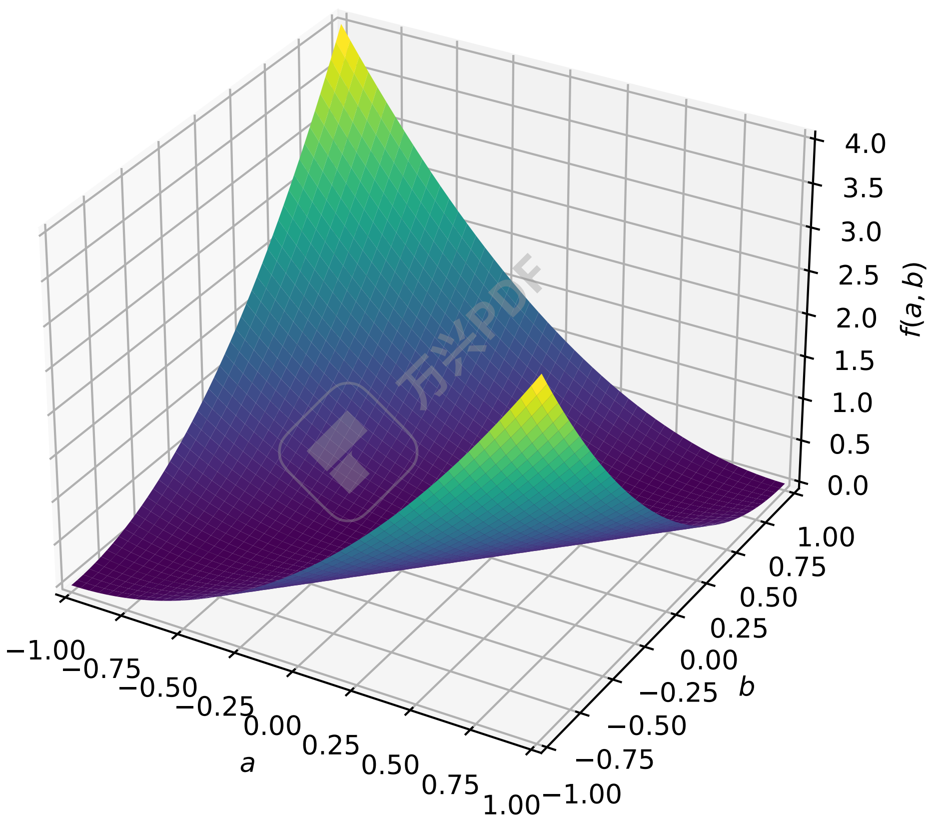
An example of symmetric negative definite matrix:

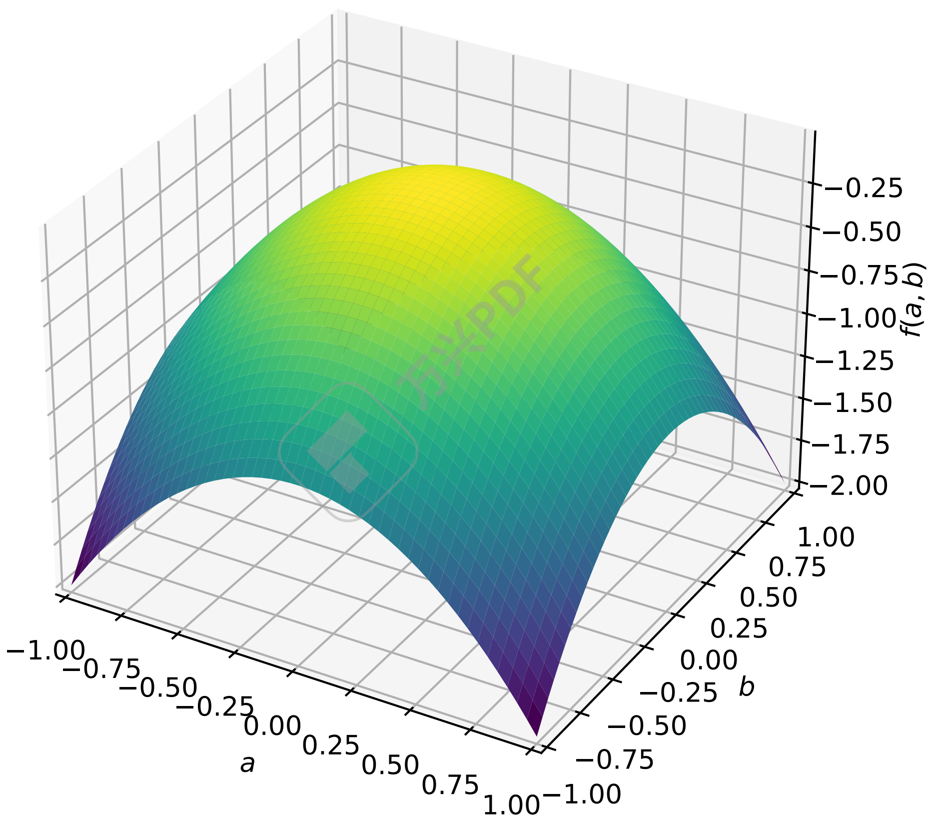
```
>>> A = Matrix([[-1, 0], [0, -1]])
>>> A.is_negative_definite
True
>>> A.is_negative_semidefinite
True
>>> A.is_indefinite
False
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b


4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

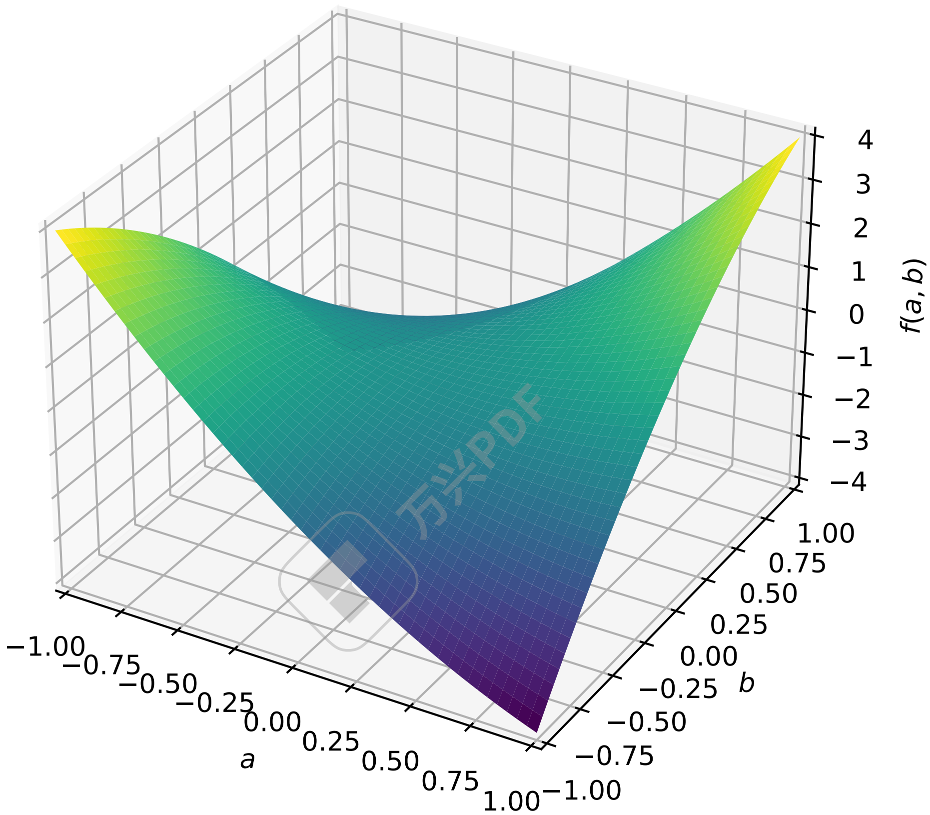


−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

An example of symmetric indefinite matrix:

```
>>> A = Matrix([[1, 2], [2, -1]])
>>> A.is_indefinite
True
```

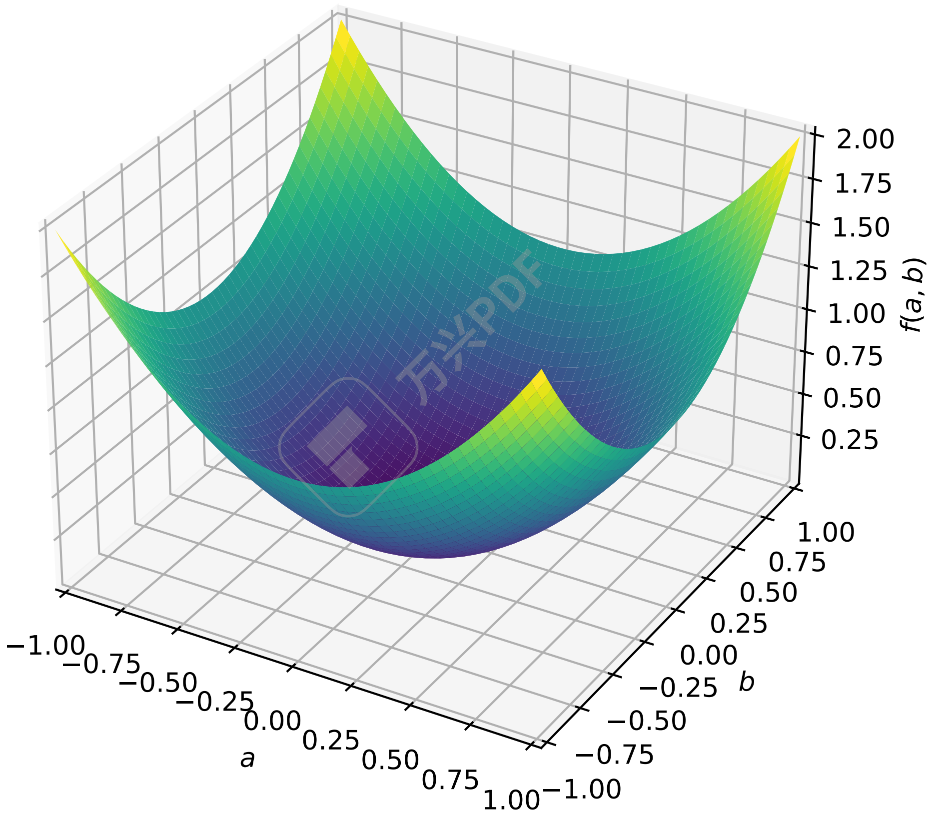
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



An example of non-symmetric positive definite matrix.

```
>>> A = Matrix([[1, 2], [-2, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b

Notes

Although some people trivialize the definition of positive definite matrices only for symmetric or hermitian matrices, this restriction is not correct because it does not classify all instances of positive definite matrices from the definition $x^T A x > 0$ or $\text{re}(x^H A x) > 0$.

For instance, `Matrix([[1, 2], [-2, 1]])` presented in the example above is an example of real positive definite matrix that is not symmetric.

However, since the following formula holds true;

$$\text{re}(x^H A x) > 0 \iff \text{re}(x^H \frac{A + A^H}{2} x) > 0$$

We can classify all positive definite matrices that may or may not be symmetric or hermitian by transforming the matrix to $\frac{A+A^T}{2}$ or $\frac{A+A^H}{2}$ (which is guaranteed to be always real symmetric or complex hermitian) and we can defer most of the studies to symmetric or hermitian positive definite matrices.

But it is a different problem for the existance of Cholesky decomposition. Because even though a non symmetric or a non hermitian matrix can be positive definite, Cholesky or LDL decomposition does not exist because the decompositions require the matrix to be symmetric or hermitian.

References

[R584], [R585], [R586]

property is_positive_semidefinite

Finds out the definiteness of a matrix.

Explanation

A square real matrix A is:

- A positive definite matrix if $x^T A x > 0$ for all non-zero real vectors x.
- A positive semidefinite matrix if $x^T A x \geq 0$ for all non-zero real vectors x.
- A negative definite matrix if $x^T A x < 0$ for all non-zero real vectors x.
- A negative semidefinite matrix if $x^T A x \leq 0$ for all non-zero real vectors x.
- An indefinite matrix if there exists non-zero real vectors x, y with $x^T A x > 0 > y^T A y$.

A square complex matrix A is:

- A positive definite matrix if $\text{re}(x^H A x) > 0$ for all non-zero complex vectors x.
- A positive semidefinite matrix if $\text{re}(x^H A x) \geq 0$ for all non-zero complex vectors x.
- A negative definite matrix if $\text{re}(x^H A x) < 0$ for all non-zero complex vectors x.
- A negative semidefinite matrix if $\text{re}(x^H A x) \leq 0$ for all non-zero complex vectors x.
- An indefinite matrix if there exists non-zero complex vectors x, y with $\text{re}(x^H A x) > 0 > \text{re}(y^H A y)$.

A matrix need not be symmetric or hermitian to be positive definite.

- A real non-symmetric matrix is positive definite if and only if $\frac{A+A^T}{2}$ is positive definite.
- A complex non-hermitian matrix is positive definite if and only if $\frac{A+A^H}{2}$ is positive definite.

And this extension can apply for all the definitions above.

However, for complex cases, you can restrict the definition of $\text{re}(x^H Ax) > 0$ to $x^H Ax > 0$ and require the matrix to be hermitian. But we do not present this restriction for computation because you can check `M.is_hermitian` independently with this and use the same procedure.

Examples

An example of symmetric positive definite matrix:

```
>>> from sympy import Matrix, symbols
>>> from sympy.plotting import plot3d
>>> a, b = symbols('a b')
>>> x = Matrix([a, b])
```

```
>>> A = Matrix([[1, 0], [0, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

An example of symmetric positive semidefinite matrix:

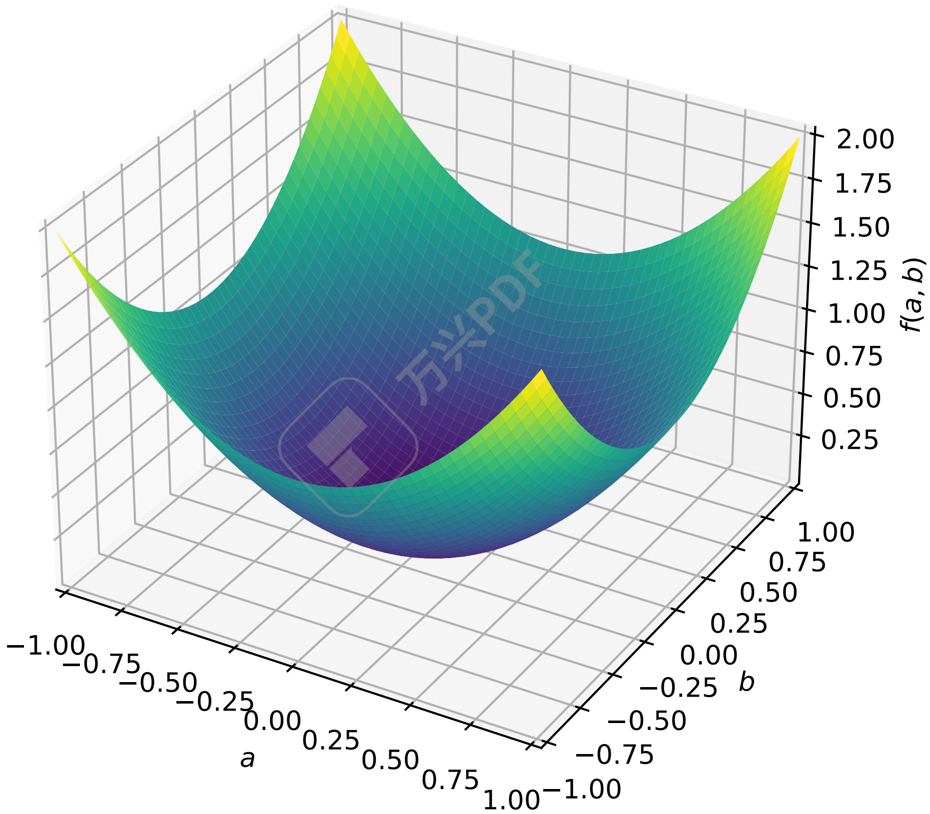
```
>>> A = Matrix([[1, -1], [-1, 1]])
>>> A.is_positive_definite
False
>>> A.is_positive_semidefinite
True
```

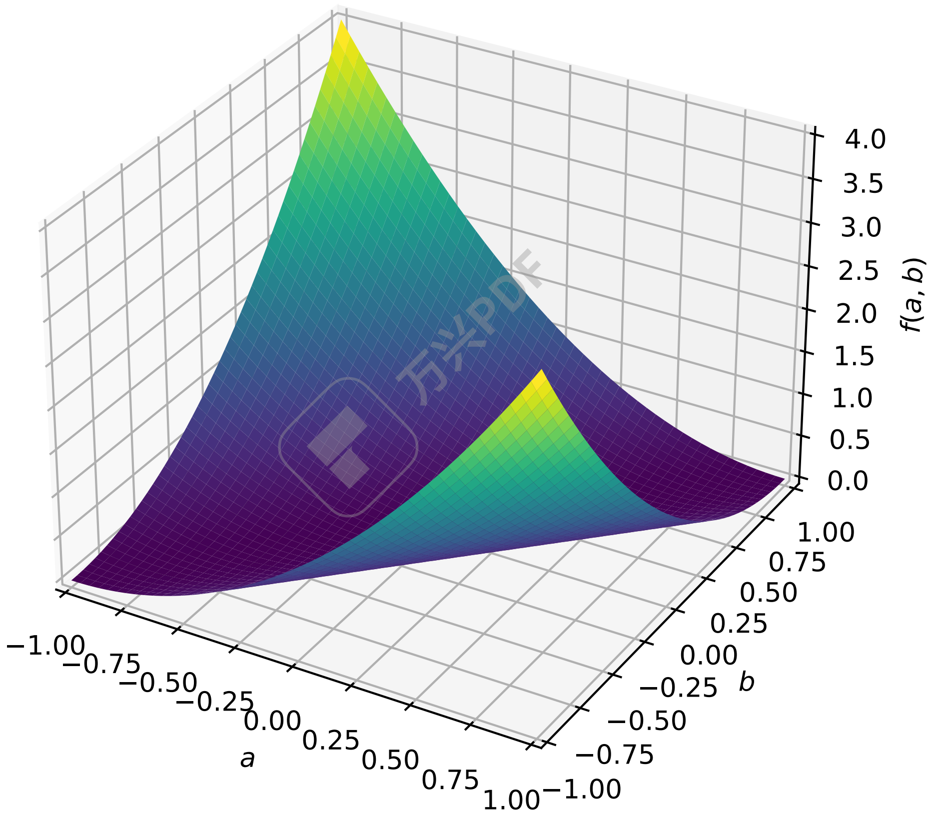
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```

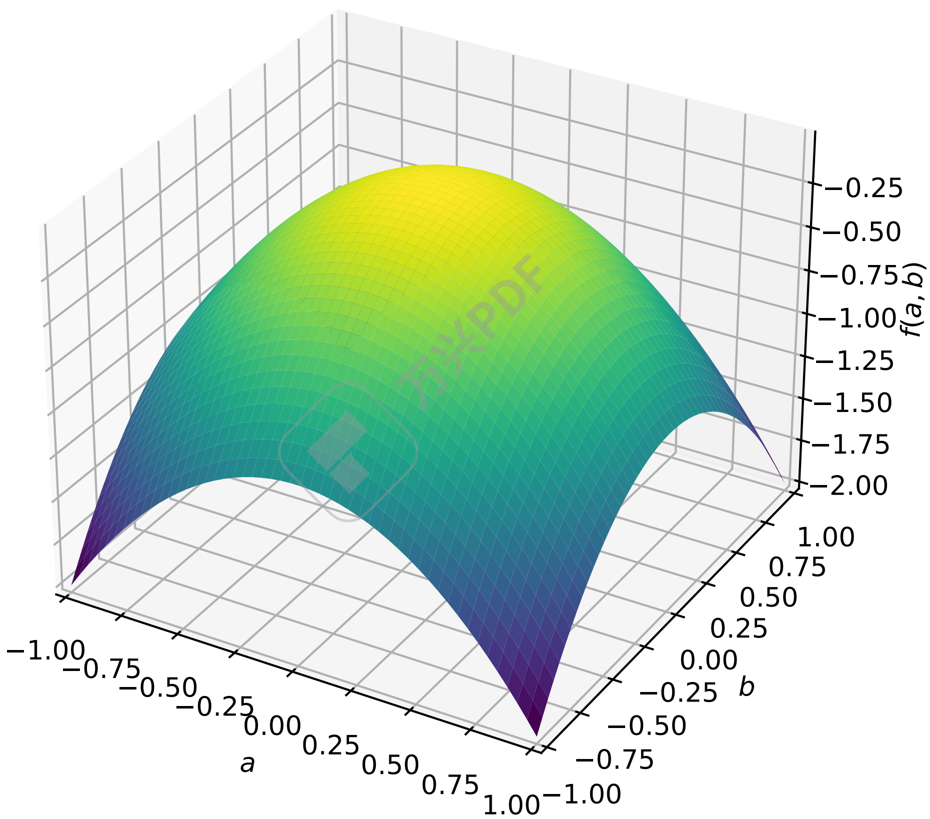
An example of symmetric negative definite matrix:

```
>>> A = Matrix([[-1, 0], [0, -1]])
>>> A.is_negative_definite
True
>>> A.is_negative_semidefinite
True
>>> A.is_indefinite
False
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```


2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
b

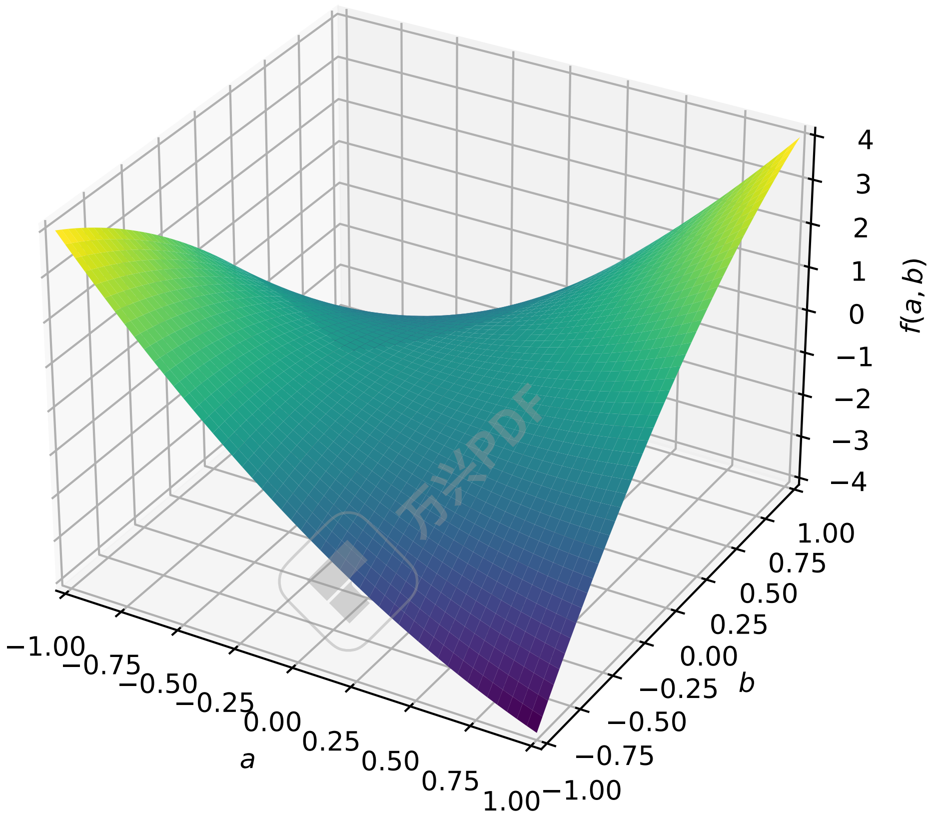

4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b


−0.25
−0.50
−0.75
−1.00
−1.25
−1.50
−1.75
−2.00
f(a, b)
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a

An example of symmetric indefinite matrix:

```
>>> A = Matrix([[1, 2], [2, -1]])
>>> A.is_indefinite
True
```

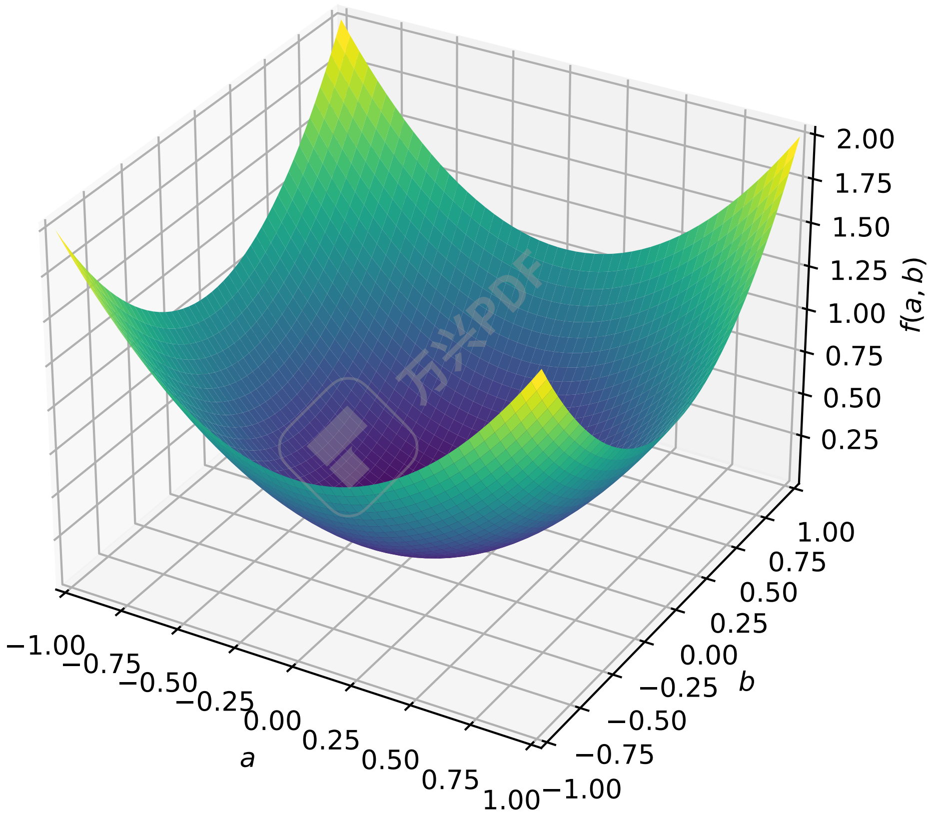
```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



An example of non-symmetric positive definite matrix.

```
>>> A = Matrix([[1, 2], [-2, 1]])
>>> A.is_positive_definite
True
>>> A.is_positive_semidefinite
True
```

```
>>> p = plot3d((x.T*A*x)[0, 0], (a, -1, 1), (b, -1, 1))
```



2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
f(a, b)
−1.00
−0.75
−0.50
−0.25
0.00
0.25
0.50
0.75
1.00
a
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
b