hlmm Documentation

Release 1.2.0a1

Alexander Thomas Ian Strudwick Young

Dec 14, 2017

Contents:

CHAPTER 1

Introduction

block_sparse is a python library for performing computations with

CHAPTER 2

Tutorial

CHAPTER 3

Documentation for 'block_sparse' module

Documentation for the regrnd model class.

class `block_sparse.`**`block_sparse`**(*blocks*, *nonzero*, *submatrices*, *dtype=<type 'numpy.float32'>*, *row_names=None*, *col_names=None*)

Define a block-sparse matrix

Parameters **blocks** : `list`

list of [row_block_boundaries,col_block_boundaries], where each of row_block_boundaries and col_block_boundaries is a 1D `array` of integers, beginning with 0, followed by the end boundaries of each block in increasing order

nonzero : `array`

boolean numpy array with number of rows equal to number of row blocks, and number of columns equal to number of col blocks. If entry [i,j] of nonzero is True, then the corresponding block is non-zero; if it is False, then the corresponding block is zero.

submatrices : `list`

list of submatrices for non-zero blocks in row-major order; e.g., block (1,1), (1,2), (2,1), (2,2),... Each submatrix can be an `array`, a *`block_sparse`* matrix, or a *`symmetric_block_sparse`* matrix.

dtype : numpy data type object

Set the default data type for the submatrices. Default `float32`

row_names : `array`

numpy array with names of the row-blocks. Default `None`

col_names : `array`

numpy array with names of the col-blocks. Default `None`

Returns **matrix** : *`block_sparse`*

block-sparse matrix

Methods

add(A)	Matrix addition of a matrix A to current matrix
dot(A)	Right multiply the current matrix with another block_sparse matrix, symmetric_block_sparse matrix, or array, A.
frobenius(A)	Compute the frobenius inner product between the current matrix and matrix A
get_submatrix(block)	Retrieve a particular block of the matrix
get_type(block)	Retrieve the type of a particular block of the matrix
norm()	Compute the frobenius norm of the current matrix
qform(y[, z])	Computes quadratic form defined by current matrix and input vectors.
to_dense()	Return the current matrix as a standard (dense) numpy array
transpose()	Return the transpose of the block-sparse matrix

add (*A*)

Matrix addition of a matrix A to current matrix

Parameters A : matrix

matrix A with same dimensions as current matrix. The matrix A can be an `array`, `block_sparse` matrix, or `symmetric_block_sparse` matrix. It must have the same block structure as the current matrix if the matrix is a `block_sparse` matrix or `symmetric_block_sparse` matrix.

Returns `block_sparse`

the block-sparse matrix formed by matrix addition of the current matrix to A

dot (*A*)

Right multiply the current matrix with another `block_sparse` matrix, `symmetric_block_sparse` matrix, or `array`, A.

Parameters A : matrix

matrix A with compatible dimensions and block structure: i.e. the row blocks of A must match the column blocks of the current matrix, unless A is an `array`.

Returns `block_sparse`

the block-sparse matrix formed by right multiplication of the current matrix by A

frobenius (*A*)

Compute the frobenius inner product between the current matrix and matrix A

Parameters A : matrix

matrix A with same dimensions as current matrix. The matrix A can be an `array`, `block_sparse` matrix, or `symmetric_block_sparse` matrix. It must have the same block structure as the current matrix if the matrix is a `block_sparse` matrix or `symmetric_block_sparse` matrix.

Returns `float`

the frobenius inner product between the current matrix and matrix A

get_submatrix(*block*)
Retrieve a particular block of the matrix

Parameters **block** : `tuple`

tuple (i,j) giving the index of the block

Returns block

either an `array`, a *block_sparse* matrix, or a *symmetric_block_sparse* matrix.

get_type(*block*)
Retrieve the type of a particular block of the matrix

Parameters **block** : `tuple`

tuple (i,j) giving the index of the block

Returns block type

either `array`, *block_sparse*, or *symmetric_block_sparse*.

norm()
Compute the frobenius norm of the current matrix

Returns `float`

the frobenius norm of the current matrix

qform(*y*, *z=None*)
Computes quadratic form defined by current matrix and input vectors. Let X be the current *block_sparse* matrix, and y and z column vectors. When it is defined, this computes the quadratic form y'Xz. If only y is provided, this computes the quadratic form y'Xy.

Parameters **y** : `array`

1D numpy array of same length as number of rows of current matrix

z [`array`] 1D numpy array of same length as number of rows of current matrix. Default `None`.

Returns `float`

the value of the quadratic form y'Xz

to_dense()
Return the current matrix as a standard (dense) numpy array

Returns `array`

transpose()
Return the transpose of the block-sparse matrix

Returns *block_sparse*

class block_sparse.**symmetric_block_sparse**(*blocks*, *nonzero*, *submatrices*, *dtype=<type 'numpy.float32'>*, *row_names=None*, *col_names=None*)
Define a symmetric block-sparse matrix. Inherits some methods from *block_sparse*.

Parameters **blocks** : `array`

1D numpy integer array, starting at zero, followed by block boundaries, which are the same for both rows and columns

nonzero : array

symmetric boolean numpy array with number of rows equal to number of row blocks, which is equal to the number of col blocks. If entry [i,j] of nonzero is True, then the corresponding block is non-zero; if it is False, then the corresponding block is zero.

submatrices : list

list of submatrices for non-zero blocks in row-major order, ignoring lower-triangular blocks; e.g., block (1,1), (1,2), (2,2),... Each submatrix can be a array, a *block_sparse* matrix, or a *symmetric_block_sparse* matrix.

dtype : numpy data type object

Set the default data type for the submatrices. Default float32

row_names : array

numpy array with names of the row-blocks. Default None

col_names : array

numpy array with names of the col-blocks. Default None

Returns *symmetric_block_sparse*

block-sparse matrix

Methods

add(A)	Matrix addition of a matrix A to current matrix.
dot(A)	Right multiply the current matrix with another *block_sparse* matrix, *symmetric_block_sparse* matrix, or array, A.
frobenius(A)	Compute the frobenius inner product between the current matrix and matrix A
get_submatrix(block)	Retrieve a particular block of the matrix
get_type(block)	Retrieve the type of a particular block of the matrix
norm()	Compute the frobenius norm of the current matrix
qform(y[, z])	Let X be the current *symmetric_block_sparse* matrix, and y and z column vectors.
to_dense()	Return the current matrix as a standard (dense) numpy array
transpose()	Return the transpose of the symmetric block-sparse matrix

add(*A*)

Matrix addition of a matrix A to current matrix.

Parameters A : matrix

matrix A with same dimensions as current matrix. The matrix A can be a array, *block_sparse* matrix, or *symmetric_block_sparse* matrix. It must have the same block structure as the current matrix if the matrix is a *block_sparse* matrix or *symmetric_block_sparse* matrix.

Returns matrix

If A is *symmetric_block_sparse*, returns a *symmetric_block_sparse* matrix. Otherwise, returns a *block_sparse* matrix.

get_submatrix (*block*)
Retrieve a particular block of the matrix

Parameters **block** : tuple

tuple (i,j) giving the index of the block

Returns block

either a array, a *block_sparse* matrix, or a *symmetric_block_sparse* matrix.

get_type (*block*)
Retrieve the type of a particular block of the matrix

Parameters **block** : tuple

tuple (i,j) giving the index of the block

Returns block type

either array, *block_sparse* matrix, or *symmetric_block_sparse* matrix.

qform (*y*, *z=None*)
Let X be the current *symmetric_block_sparse* matrix, and y and z column vectors. When it is defined, this computes the quadratic form y'Xz. If only y is provided, this computes the quadratic form y'Xy.

Parameters **y** : array

1D numpy array of same length as number of rows of current matrix

z [array] 1D numpy array of same length as number of rows of current matrix. Default None.

Returns float

the value of the quadratic form y'Xz

to_dense ()
Return the current matrix as a standard (dense) numpy array

Returns array

transpose ()
Return the transpose of the symmetric block-sparse matrix

Returns *symmetric_block_sparse*

the current matrix, as it is symmetric

block_sparse.**matmul** (*X*, *A*)
Matrix multiplication between *block_sparse* and *symmetric_block_sparse* matrices, as well as matrix multiplication between a *block_sparse* or *symmetric_block_sparse* matrix and an array.

Parameters **X** : matrix

The matrix X can be a *block_sparse* matrix, a *symmetric_block_sparse* matrix, or a array.

A : matrix

The matrix A can be a *block_sparse* matrix, a *symmetric_block_sparse* matrix, or a array. Note that the number of rows of A must match the number of columns of X. Furthermore, if X and A are both *block_sparse* or *symmetric_block_sparse*, then the column blocks of X must match the row blocks of A.

Returns *block_sparse*

the block-sparse matrix formed by matrix multiplication XA

block_sparse.**dense_to_block_sparse**(*dense*, *blocks*, *symmetric*, *dtype=<type 'numpy.float64'>*)

Convert a standard (dense) numpy array into a *block_sparse* or a *symmetric_block_sparse* matrix. Note this simply imposes a block structure onto the matrix so that it can interact with other block matrices. It does not take advantage of any sparsity in the input matrix.

Parameters **dense** : array

input matrix

blocks [list] list of [row_block_boundaries,col_block_boundaries], where each of row_block_boundaries and col_block_boundaries is a 1D array of integers, beginning with 0, followed by the end boundaries of each block in increasing order

symmetric [bool] if True, returns a *symmetric_block_sparse* matrix; if False, returns a *block_sparse* matrix

dtype [numpy data type] the default data type of the returned matrix

Returns matrix

the current matrix as a *block_sparse* or a *symmetric_block_sparse* matrix

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

b

Index