MATLAB Tensor Classes for Fast Algorithm Prototyping: Source Code

Brett W. Bader and Tamara G. Kolda

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy's National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.
MATLAB Tensor Classes for Fast Algorithm Prototyping: Source Code

Brett W. Bader  
Computational Sciences Department  
Sandia National Laboratories  
Albuquerque, NM 87185–0316  
bwbader@sandia.gov

Tamara G. Kolda  
Computational Sciences and Mathematics Research Department  
Sandia National Laboratories  
Livermore, CA 94551–9217  
tgkolda@sandia.gov

ABSTRACT
We present the source code for three MATLAB classes for manipulating tensors in order to allow fast algorithm prototyping. A tensor is a multidimensional or $N$-way array. This is a supplementary report; details on using this code are provided separately in SAND-XXXX.

Keywords: higher-order tensors, $n$-way arrays, multidimensional arrays, MATLAB
Installation instructions:
1. Unpack the tar/zip file into the directory of your choice. Note the directory for addition to your MATLAB path in step 3.
2. Start MATLAB.
3. Add the Tensor directory to your path with the "addpath" or "path" commands. For example:
   ```matlab
   addpath ~/Tensor
   or
   path('~/Tensor', path);
   ```
The directory provided must have access to the class directories (@tensor, @cp_tensor, @tucker_tensor, and @tensor_as_matrix) for MATLAB to recognize them. The command "addpath" prepends the specified directory to the current matlabpath whereas "path" has more flexibility.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

### `@tensor/and.m`

```matlab
function C = and(A,B)
% TENSOR/AND Logical AND.
%
%   A & B is a tensor whose elements are 1's where both A and B
%   have non-zero elements, and 0's where either has a zero element.
%   A and B must have the same dimensions unless one is a scalar.
%   C = AND(A,B) is called for the syntax 'A & B' when A or B is a
%   tensor.

%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B)  |  (prod(size(A)) == 1)  |  (prod(size(B)) == 1) )
    error('Tensor size mismatch.');
end
C = multiarrayop(@and,A,B);
```

### `@tensor/disp.m`

```matlab
function disp(t,name)
% TENSOR/DISP Command window display of a tensor.

% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

if ~exist('name','var')
    namedot = '';
    name = 'Tensor';
else
    namedot = [name '.'];
    name = [name ' is a tensor'];
end
if strcmp(get(0,'FormatSpacing'),'compact')
    skipspaces = 1;
else
    skipspaces = 0;
end
if skipspaces == 1
    fprintf('
');
end
fprintf('disp,t,name);%n
fprintf(',"td", sz(length(sz)));
```
Function: display(t)
% TENSOR/DISPLAY Command window display of a tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

disp(t,inputname(1));

Function: a = double(t)
% TENSOR/DATETIME Convert tensor to double array.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

a = t.data;

Function: C = ge(A,B)
% TENSOR/GE Greater than or equal.
% A >= B does element by element comparisons between A and B and
% returns a tensor of the same size with elements set to one where
% the relation is true and elements set to zero where it is not. A
% and B must have the same dimensions unless one is a scalar. A
% scalar can be compared with anything.
% C = GE(A,B) is called
% for the syntax 'A >= B' when A or B is a
% tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )
    error('Tensor size mismatch.');
end
C = multiarrayop('ge',A,B);

Function: C = gt(A,B)
% TENSOR/GT Greater than.
% A > B does element by element comparisons between A and B
% and returns a tensor of the same size with elements set to one
% where the relation is true and elements set to zero where it is
% not. A and B must have the same dimensions unless one is a
% scalar. A scalar can be compared with anything.
% C = GT(A,B) is called
% for the syntax 'A > B' when A or B is a
% tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )
    error('Tensor size mismatch.');
end
C = multiarrayop('gt',A,B);
function b = isissamesize(A,B)

\% ISSAMESIZE(A,B) returns true if tensors A and B are the same size.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% distribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not use or redistribute this work for commercial purposes. (3) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.

if (ndims(A) == ndims(B)) && (size(A) == size(B))
    b = (1==1); \% true
else
    b = (0==1); \% false
end

function C = ldivide(A,B)

\% LDIVIDE Left array divide.
\% A .\ B denotes element-by-element division. A and B
\% must have the same dimensions unless one is a scalar.
\% A scalar can be divided with anything.
\% C = LDIVIDE(A,B) is called for the syntax 'A .\ B' when A or B is
\% a tensor.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% distribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not use or redistribute this work for commercial purposes. (3) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) || (prod(size(A)) == 1) || (prod(size(B)) == 1) )
    error('Tensor size mismatch.' )
end
C = multiarrayop(@ldivide,A,B);

function C = le(A,B)

\% LT Less than or equal.
\% A <= B does element by element comparisons between A and B
\% and returns a tensor of the same size with elements set to one
\% where the relation is true and elements set to zero where it is
\% not. A and B must have the same dimensions unless one is a
\% scalar. A scalar can be compared with anything.
\% C = LE(A,B) is called for the syntax 'A <= B' when A or B is a
\% tensor.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% distribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not use or redistribute this work for commercial purposes. (3) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) || (prod(size(A)) == 1) || (prod(size(B)) == 1) )
    error('Tensor size mismatch.' )
end
C = multiarrayop(@lt,A,B);

function C = lt(A,B)

\% LT Less than.
\% A < B does element by element comparisons between A and B
\% and returns a tensor of the same size with elements set to one
\% where the relation is true and elements set to zero where it is
\% not. A and B must have the same dimensions unless one is a
\% scalar. A scalar can be compared with anything.
\% C = LT(A,B) is called for the syntax 'A < B' when A or B is a
\% tensor.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% distribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not use or redistribute this work for commercial purposes. (3) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) || (prod(size(A)) == 1) || (prod(size(B)) == 1) )
    error('Tensor size mismatch.' )
end
C = multiarrayop(@lt,A,B);
Function C = minus(A,B)\n\n\% TENSOR/MINUS Binary subtraction for tensors.
\% minus(A,B) subtracts tensor B from A. A and B must have the same
\% dimensions unless one is a scalar. A scalar can be subtracted
\% from anything.
\% C = minus(A,B) is called for the syntax 'A - B' when A or B is a
\% tensor.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% redistribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not alter, transform, or build upon this work. (3) You may
\% not use or redistribute this work for commercial purposes. (4) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.
\% A = tensor(A);
\% B = tensor(B);
\% if (~(issamesize(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1))
\%   error('Tensor size mismatch.');
\% end
\% C = multiarrayop(minus,A,B);
\n\nFunction C = mtimes(A,B)\n\n\% TENSOR/MTIMES Implement A*B for tensors.
\% MTIMES(A,B) is the product of A and B. Any scalar may multiply
\% a tensor. Otherwise, the last dimension of A must equal the
\% first dimension of B.
\% C = MTIMES(A,B) is called for the syntax 'A * B' when A or B is a
\% tensor.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% redistribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not alter, transform, or build upon this work. (3) You may
\% not use or redistribute this work for commercial purposes. (4) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.
\% A = tensor(A);
\% B = tensor(B);
\% if (prod(size(B)) == 1)
\%   C = tensor(A.data * B.data, size(A));
\%   return;
\% elseif (prod(size(A)) == 1)
\%   C = tensor(A.data * B.data, size(B));
\%   return;
\% end
\% dimA = size(A);
\% dimB = size(B);
\% if (dimA(length(dimA)) == dimB(1))
\%   error('Tensor dimensions must agree.');
\% end
\% C = ttt(A,B,ndims(A),1);
\n\nFunction C = multiarrayop(fname,A,B)
\% TENSOR/MULTIARRAYOP Generic multidimensional array functions
\% for tensors.
\% multiarrayop(fname,A,B) applies the multidimensional array function
\% specified by a function handle or function name, F, to the given
\% tensor arguments, A and B. For example, if F = @plus, then
\% multiarrayop(F,A,B) adds the multidimensional array data of A and
\% B.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% redistribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not alter, transform, or build upon this work. (3) You may
\% not use or redistribute this work for commercial purposes. (4) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.
\% A = tensor(A);
\% B = tensor(B);
\% if (prod(size(B)) == 1)
\%   C = tensor(A.data * B.data, size(A));
\%   return;
\% elseif (prod(size(A)) == 1)
\%   C = tensor(A.data * B.data, size(B));
\%   return;
\% end
\% n = ndims(t);
\% for
\%   C = feval(fname, A.data, B.data);
\% end
\% C = tensor(C, sz);
\nFunction n = ndims(t)
\% TENSOR/NDIMS Return the number of dimensions.
\% NDIMS(T) returns the number of dimensions of tensor T. NDIMS is
\% useful for determining the minimum number of subscripts capable of
\% accessing all of the elements of T. In particular, NDIMS of a
\% zero-order tensor (i.e., a scalar) is one; all others cases are
\% identical to ORDER.
\% See also ORDER.
\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
\% Sandia National Laboratories, 2004. Please address questions or
\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
\% redistribute, display, and use this work, under the following
\% conditions. (1) You must give the original authors credit. (2) You may
\% not alter, transform, or build upon this work. (3) You may
\% not use or redistribute this work for commercial purposes. (4) You may
\% not alter, transform, or build upon this work. (4) For any reuse or
\% redistribution, you must make clear to others the license terms of this
\% work. (5) Any of these conditions can be waived if you get permission
\% from the authors.
\% n = max(length(t.size), 1);

MATLAB Tensor Classes by B. W. Bader and T. G. Kolda

Page 1/1
Page 1/1
Page 1/1
Page 1/1
@tensor/norm.m

**Function**: n = norm(T)

**TENSOR/NORM** Frobenius norm of a tensor.

% n = norm(T) returns the Frobenius norm of a tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

if T == T' &&
  T.data = reshape(T.data,1,prod(size(T)));
else
  n = sqrt(sum(T.data));
  v = reshape(T.data, prod(size(T.data)), 1);
  n = norm(v);

@tensor/not.m

**Function**: B = not(A)

**TENSOR/NOT** Logical NOT.

% ~A is a tensor whose elements are 1's where A has zero
% elements, and 0's where A has non-zero elements.
% B = NOT(A) is called for the syntax '~A' when A is a tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

B = feval(@not, A.data);
B = tensor(B, size(A));

@tensor/or.m

**Function**: C = or(A,B)

**TENSOR/OR** Logical OR.

% A | B is a tensor whose elements are 1's where either A or B
% has a non-zero element, and 0's where both have zero elements.
% A and B must have the same dimensions unless one is a scalar.
% C = OR(A,B) is called for the syntax 'A | B' when A or B is a
% tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

A = tensor(A);
B = tensor(B);
if ~( isequal(size(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1))
    error('Tensor size mismatch.');
end
C = multiarrayop(8or,A,B);

@tensor/order.m

**Function**: n = order(t)

**TENSOR/ORDER** Return the order of a tensor

% ORDER(T) returns the mathematical order of tensor T.  In most
% cases, ORDER is equal to NDIMS, the number of dimensions of a
% tensor.  The single difference is that the ORDER of a scalar is 0
% and not 1.
% See also NDIMS.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

n = length(t.size);
function T = permute(T,Idx)

% TENSOR/PERMUTE Permute tensor dimensions
% T = PERMUTE(T,ORDER) rearranges the dimensions of T so that they
% are in the order specified by the vector ORDER. The result has the
% same values of T, but the order of the subscripts needed to access
% any particular element are rearranged as specified by ORDER.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not alter, transform, or build upon this work. (3) You may
% not use or redistribute this work
% for commercial purposes. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived
% if you get permission
% from the authors.

if length(Idx) == 1
    if (Idx == 1)
        return;
    else
        error('Invalid Order.);
    end
end
T.data = permute(T.data,Idx);
T.size = T.size(Idx);

function C = plus(A,B)

% TENSOR/PLUS Binary addition for tensors.
% C = PLUS(A,B) adds tensors A and B. A and B must have the same
% dimensions unless one is a scalar. A scalar can be added to
% anything.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not alter, transform, or build upon this work. (3) You may
% not use or redistribute this work
% for commercial purposes. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived
% if you get permission
% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )
    error('Tensor size mismatch.'
end
C = multiarrayop(@plus,A,B);

function C = rdivide(A,B)

% TENSOR/RDIVIDE Right array divide.
% C = Rdivide(A,B) is called for the syntax 'A ./ B' when A or B is a
% tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not alter, transform, or build upon this work. (3) You may
% not use or redistribute this work
% for commercial purposes. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived
% if you get permission
% from the authors.

A = tensor(A);
B = tensor(B);
if ~( issamesize(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )
    error('Tensor size mismatch.'
end
C = multiarrayop(@rdivide,A,B);
Function B = shiftdim(varargin)
%SHIFTDIM Shift dimensions.

%   B = SHIFTDIM(X,N) shifts the dimensions of tensor X by N. When N is
%   positive, SHIFTDIM shifts the dimensions to the left and wraps
%   the N leading dimensions to the end. When N is negative, SHIFTDIM
%   shifts the dimensions to the right and pads with singletons.

% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

A = varargin{1};
N = varargin{2};
B = feval(@shiftdim, A.data, N);
B = tensor(B, size(B));

M = size(t, idx);
if exist('idx', 'var')
    M = t.size(idx);
else
    M = t.size;
end

switch s.type
    case '.'
        error(['Cannot change field ', s.subs, ' directly.']);
    case '()'
        data = t.data;
        data(s.subs{:}) = b;
        t = tensor(data, t.size);
    case '{}'
        error('Subscript cell reference not supported for tensor.');
    otherwise
        error('Incorrect indexing into tensor.');
end
TENSOR/SUBSREF Subscripted reference.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

switch s.type
    case '{}'
        case '()''
            end
            else
                error('Incorrect indexing into tensor.');
        otherwise
            error('[No such field: ', s.subs, '].');
        end
    case '[]'
        a = t.data(s.subs{:});
        if prod(size(a)) > 1
            a = tensor(a);
        end
        if isempty(a)
            sz = a.tsize;
            if isa(a, 'cp_tensor') | isa(a, 'tucker_tensor')
                error('Specified size is incorrect.');
            end
            a = tensor(a);
            return;
        end
        end
    otherwise
        error('Incorrect indexing into tensor.');
end

TENSOR Tensor class constructor.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

function t = tensor(varargin)
    % TENSOR creates an empty dense tensor object.
    % TENSOR(T) creates a tensor by copying the tensor T or
    % converting a CP or Tucker tensor T.
    % TENSOR(Z,DIMS) creates a tensor from the multidimensional array Z.
    % The DIMS argument is used to specify any trailing singleton
    % dimensions. If DIMS is empty, then the result is a zero order
    % tensor, i.e., a scalar.
    % TENSOR(A) create a tensor from a tensor_as_matrix object.
    % TENSOR(A,1,DIMS,TYPE) creates a tensor by reshaping a matrix A
    % stored as an I-mode matricization. The dimensions of the
    % resulting tensor are specified by DIMS. The TYPE specifies which
    % type of matricization is used; the choices for TYPE are:
    % - 'ODV': Definition 1 in L. De Lathauwer, B. De Moor and
    %   J. Vandewalle, SIMAX 21(4):1253-1278 (this is the default)
    % - 'Kiers': Definition from J.A.L. Kiers, J. Chemometrics
    %   2000(14):105-122
    % See also TENSOR/MATRICIZE, CP_TENSOR, TUCKER_TENSOR.

    nargin = length(varargin);
    if nargin == 0
        t = class(t, 'tensor');
        t.size = varargin{1}.size;
        t.data = varargin{1}.data;
        if isa(varargin{1}, 'tensor')
            t = class(t, 'tensor');
            return;
        end
        t = class(t, 'tensor');
        return;
    end
    % EMPTY/DEFAULT CONSTRUCTOR
    if size(varargin{1}) == 0
        t.data = [];
        t.size = 0;
        t = class(t, 'tensor');
        return;
    end
    % COPY CONSTRUCTOR
    if isa(varargin{1}, 'tensor')
        t.data = varargin{1}.data;
        t.size = varargin{1}.size;
        t = class(t, 'tensor');
        return;
    end
    % Convert CP_TENSOR OR TUCKER_TENSOR
    if isa(varargin{1}, 'cp_tensor') | isa(varargin{1}, 'tucker_tensor')
        t = class(t, 'tensor');
        return;
    end
    % Convert Tensor_as_matrix
    if isa(varargin{1}, 'tensor_as_matrix')
        a = varargin{1};
        idx = {a.rindices a.cindices};
        if isempty(idx)
            idx = 1;
        end
        sz = a.tsize;
        if isa(a, 'tensor')
            sz = sz(idx);
        end
        t.data = reshape(a.data, sz 1 1);
        t.size = sz;
        t = class(t, 'tensor');
        sz = size(t.data);
        t = permute(t, idx);
        return;
    end
    % Convert A multidimensional array
    if nargin == 1 || (nargin == 2)
        if isa(varargin{1}, 'numeric') && isa(varargin{1}, 'logical')
            error('Z must be a multidimensional array.');
        end
        t.data = varargin{1};
        t.size = [];
        if nargin == 1
            t.size = size(t.data);
        else
            if length(t.size) == 2 && (size(t.size,2) == 1)
                error('DIMS must be a row vector.');
            end
            % Error Check --
            % First, check that the matching dimensions do indeed match
            sz = size(t.data);
            j = min([length(sz),length(t.size)]);
            for i = 1 : j
                if sz(i) ~= t.size(i)
                    error('Specified size is incorrect.');
                end
            end
            t.data = varargin{2};
            t.size = varargin{2};
        end
        end
        t.data = varargin{1};
        t.size = varargin{1}.size;
        if isa(varargin{1}, 'tensor')
            t.data = varargin{1};
            t.size = varargin{1}.size;
            t = class(t, 'tensor');
            return;
        end
        t.data = varargin{1};
        t.size = varargin{1}.size;
        t = class(t, 'tensor');
        return;
    end
    % Convert the multidimensional array Z.
    if nargin == 1
        t = tensor(varargin{1});
        return;
    end
    % End function
    end

end

end

end

end

end

end
function C = ttm(a,v,dims)

% TENSOR/TTM Tensor times matrix
% B = TTM(A,V,N) computes the n-mode product of tensor A with a
% matrix V; i.e., A x_N V. The integer N specifies the dimension
% (or mode) of A along which V should be multiplied. If size(V) =
% [J,1], then A must have size(A,N) = 1. The result will be the
% same order and size as A except that size(B,N) = J.
% B = TTM(A,U) computes the n-mode product of tensor A with a
% sequence of matrices in the cell array U. The n-mode products
% are computed sequentially along all dimensions (or modes) of A.
% The cell array U contains ndims(A) matrices.
% B = TTM(A,U,DIMS) computes the sequence tensor-matrix products
% along the dimensions specified by DIMS.

Consider the following examples. Let A be a 4th-order tensor of
size I x J x K x L. Let V be a matrix with I columns, \( A \times_1 V \times_2 X \times_4 Z \). Let U be a matrix with J columns, \( A \times_1 V \times_2 X \times_4 Z \), and Z be a
matrix with K columns. Let U be \( (V,Y,Z) \) be a cell array
containing the four matrices.

%%
% Ex: B = ttm(A, V, 1) computes B = A x_1 V.
% Ex: B = ttm(A, U, 1) computes B = A x_1 V.
% Ex: B = ttm(A, V, 2) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, 2) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, V, [3 4]) computes B = A x_3 Y x_4 Z.
% Ex: B = ttm(A, U, [3 4]) computes B = A x_3 Y x_4 Z.
% Ex: B = ttm(A, V, [1 2 4]) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, [1 2 4]) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, [1 2]) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, 0) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, 0) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, V, [1 2 4]) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, U, [1 2 4]) computes B = A x_1 V x_2 X x_4 Z.
% Ex: B = ttm(A, V, [3 4]) computes B = A x_3 Y x_4 Z.
% Ex: B = ttm(A, U, [3 4]) computes B = A x_3 Y x_4 Z.
% Ex: B = ttm(A, V, [1 2 3 4]) computes
% B = A x_1 V x_2 X x_3 Y x_4 Z.
% Ex: B = ttm(A, U, [1 2 3 4]) computes B = A x_1 V x_2 X x_3 Y x_4 Z.
% Ex: B = ttm(A, U, 0) computes B = A x_1 V x_2 X x_3 Y x_4 Z.
% Ex: B = ttm(A, U, 0) computes B = A x_1 V x_2 X x_3 Y x_4 Z.
% Ex: B = ttm(A, U, −3) computes B = A x_1 V x_2 X x_3 Y x_4 Z.
% See also TTT, TVT.

function c = ttm(a,v,dims)

% Convert v to a tensor
v = tensor(v);
% Compute product
C = ttm(a,v,dims);
% Permute rows to proper order in c
dims = setdiff([1:ndims(a)],n);
dims = sortd(dims);
c = permute(c,dims);

function C = ttimes(a,b)
% TIMES Element-wise multiplication for tensors.
% TIMES(A,B) denotes element-by-element multiplication. A and B
% must have the same dimensions unless one is a scalar. A scalar
can be multiplied into anything.
% C = TIMES(A,B) is called for the syntax 'A .* B' when A or B is
% a tensor.

error('Nth dimension of A is not equal to number of columns in V.');
Function c = ttt(varargin)

% TENSOR/TIT Tensor Times Tensor

% TTT(A,B) computes the outer product of tensors A and B.
% TTT(A,B,ADIMS,BDIMS) computes the contracted product of tensors
% A and B in the dimensions specified by the row vectors ADIMS and
% BDIMS.  The sizes of the dimensions specified by ADIMS and BDIMS
% must match; that is, size(A,ADIMS) must equal size(B,BDIMS).  The
% result is a tensor of size equal to
% [size(A,1:size(ADIMS)-1,ADIMS), size(B,1:size(BDIMS)-1,BDIMS)].
% TTT(A,B,ADIMS) computes the inner product of tensors A and B in the
% dimensions specified by the vector ADIMS.  The sizes of the
% dimensions specified by ADIMS must match; that is, size(A,ADIMS) must
% equal size(B,BDIMS).  If BDIMS is 1:size(BDIMS), the result is a scalar.

% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% redistribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

%%%%%%%%%%%%%%%%%%%%%%
%%% ERROR CHECKING %%%
%%%%%%%%%%%%%%%%%%%%%%

% Check the number of arguments
if nargin < 2
    error('TTT requires at least two arguments.');
end

% Check the first argument
if ~isvarargin(1), 'tensor'
    error('First argument must be a tensor.');
else
    a = varargin{1};
end

% Check the second argument
if ~isvarargin(2), 'tensor'
    error('Second argument must be a tensor.');
else
    b = varargin{2};
end

% Optional 3rd argument
if nargin > 3
    adims = varargin{3};
else
    adims = [];
end

% Optional 4th argument
end

function c = ttv(a,v,n)

% TENSOR/TTV Tensor times vector

% B = TTV(A,v,N) computes the product of tensor A with a (column)
% vector V.  The integer N specifies the dimension in A along which
% V is multiplied.  If size(V) = [I,1], then A must have size(A,N) = I.
% Note that ndims(B) = ndims(A) - 1 because the N-th dimension
% is removed.

B = TTV(A,v,N)

% B = TTV(A,U,DIMS) computes the sequence tensor−vector products
% along the dimensions specified by DIMS.

Consider the following examples. Let A be a 4th-order tensor of
size I x J x K x L. Let V be a (column) vector of size I, X be a
vector of size J, Y be a column vector of size K, and Z be a
column vector of size L. Let U = (V,X,Y,Z) be a cell array
containing the four vectors. The symbol _i means to compute
the product of a tensor and a vector along dimension i.

---
Ex: B = ttv(A, v, 1) computes B = A _1 V.
Ex: B = ttv(A, u, 1) computes B = A _1 U.

---
Ex: B = ttv(A, [V,X,Y,Z], [1 2 3 4]) computes
B = A _1 V _2 X _3 Y _4 Z.
Ex: B = ttv(A, U, [1 2 3 4]) computes
B = A _1 _2 V _3 _4 X _5 Y _6 Z.
Ex: B = ttv(A, U, [2 1 3 4]) computes
B = A _1 _2 V _3 _4 X _5 Y _6 Z.

---
Ex: B = ttv(A, [y,z], [3 4]) computes B = A _3 Y _4 Z.
Ex: B = ttv(A, U, [3 4]) computes B = A _3 Y _4 Z.

---
Ex: B = ttv(A, [v,x,z], [1 2 4]) computes B = A _1 V _2 X _4 Z.
Ex: B = ttv(A, U, [1 2 4]) computes B = A _1 _2 V _3 _4 X _5 Z.
Ex: B = ttv(A, [v,x,z], -3) computes B = A _1 _2 _3 V _4 X _5 Z.
Ex: B = ttv(A, U, -3) computes B = A _1 _2 _3 _4 V _5 X _6 Z.

---
See also TIT, TTM.
 function c = ttv(a, v, dims)
    % Check validity of DIMS and then check for "minimum case''
    if (max(abs(dims)) > ndims(a))
        error('An entry in DIMS exceeds order of A.');
    elseif (max(dims) < 0)
        dims = setdiff([1:ndims(a)], -dims);
    end
    % Check validity of parameters passed to TTVS
    n = length(dims);
    if (n > ndims(a)) | (n > length(v))
        error('DIMS is too long.');
    elseif (n < length(v)) & (length(v) < ndims(a))
        error('If length(DIMS) < length(V), then length(V) must equal ndims(A).');
    elseif (length(v) > ndims(a))
        error('Length of V greater than order of A.);
    end
    % Reorder dims from largest to smallest
    [sdims, sidx] = sort(-dims);
    sdims = -sdims;
    % Check sizes of U and DIMS to determine version to use.
    if (n == length(v))
        % index V by sorted order
        vidx = sidx;
    else
        % index V by (sorted) dimension
        vidx = sdims;
    end
    % Calculate individual products
    c = ttv(a, v{vidx(1)}, sdims(1));
    for i = 2 : n
        c = ttv(c, v{vidx(i)}, sdims(i));
    end
end

MATLAB Tensor Classes by B. W. Bader and T. G. Kolda

function t = uminus(t)
% TENSOR/UMINUS Unary minus for tensors.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
distribution, you must make clear to others the license terms of this
work. (5) Any of these conditions can be waived if you get permission
from the authors.

t.data = -t.data;

function t = uplus(t)
% TENSOR/UPLUS Unary plus for tensors.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% distribution, you must make clear to others the license terms of this
work. (5) Any of these conditions can be waived if you get permission
from the authors.

% This function does nothing!

function C = xor(A, B)
% TENSOR/XOR Logical EXCLUSIVE OR.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004.  Please address questions or
% comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
distribution, you must make clear to others the license terms of this
work. (5) Any of these conditions can be waived if you get permission
from the authors.

A = tensor(A);
B = tensor(B);
if ~(issamesize(A, B) | (prod(size(A)) == 1) | (prod(size(B)) == 1))
    error('Tensor size mismatch.');
end
C = multiarrayop(@xor, A, B);
% C = CTRANSPOSE(A) swaps the row and column indices of A.

l % TENSOR_AS_MATRIX/CTRANSPOSE Complex conjugate transpose.

l Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
l Sandia National Laboratories, 2004. Please address questions or
l comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
l redistribute, display, and use this work, under the following
l conditions. (1) You must give the original authors credit. (2) You may
l not use or redistribute this work for commercial purposes. (3) You may
l not alter, transform, or build upon this work. (4) For any reuse or
l redistribution, you must make clear to others the license terms of this
l work. (5) Any of these conditions can be waived if you get permission
l from the authors.

l tmp = a.rindices;
l a.rindices = a.cindices;
l a.cindices = tmp;

l a.data = a.data';

function disp(t,name)
%TENSOR_AS_MATRIX/DISP Command window display of a tensor stored as
%a matrix.

l Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
l Sandia National Laboratories, 2004. Please address questions or
l comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% redistribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

l if ~exist('name','var')
    namedot = '';
    name = 'Tensor_as_Matrix';
    name = [name ' is a matrix'];
l end

l if strcmp(get(0,'FormatSpacing'),'compact')
    skipspaces = 0;
else l
    skipspaces = 1;
end

l if skipspaces ~= 1
    fprintf(1,'
');
end

l if isempty(t.data)
    fprintf(1,'data = 
');
else l
    fprintf(1,'data = 
');
disp(t.data);
end

function printsize(v)
%printsize prints a size list.

l n = length(v);

l for i = 1:n-1
    fprintf(1,'%d x ',v(i));
end

l if (n > 0)
    fprintf(1,'%d', v(n));
else l
    fprintf(1,'0');
end

function printvec(v)
%printvec prints a value list.

l n = length(v);

l for i = 1:n
    fprintf(1,'%d, ',v(i));
end

l if (n > 0)
    fprintf(1,'%d', v(n));
end

fprintf(1,'
]');

function display(t)
%TENSOR_AS_MATRIX/DISPLAY Command window display of a tensor.

l Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
l Sandia National Laboratories, 2004. Please address questions or
l comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% redistribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

l disp(t,inputname(1));
\texttt{function a = double(t)}
\texttt{\% TENSOR\_AS\_MATRIX/D double Convert tensor to double array.}
\texttt{\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,}
\texttt{\% Sandia National Laboratories, 2004. Please address questions or}
\texttt{\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,}
\texttt{\% distribute, display, and use this work, under the following}
\texttt{\% conditions. (1) You must give the original authors credit. (2) You may}
\texttt{\% not use or redistribute this work for commercial purposes. (3) You may}
\texttt{\% not alter, transform, or build upon this work. (4) For any reuse or}
\texttt{\% redistribution, you must make clear to others the license terms of this}
\texttt{\% work. (5) Any of these conditions can be waived if you get permission}
\texttt{\% from the authors.}
\texttt{a = t.data;}

\texttt{function c = mtimes(a,b)}
\texttt{\% TENSOR\_AS\_MATRIX/mtimes Multiplies two tensor\_as\_matrix objects.}
\texttt{\% C = MTIMES(A,B) computes the product of A and B. The result is a}
\texttt{\% TENSOR\_AS\_MATRIX object and can be transformed into a tensor.}
\texttt{\% C = MTIMES(A,B) is called for the syntax 'A * B' when A or B is a}
\texttt{\% TENSOR\_AS\_MATRIX object. Both A and B must be TENSOR\_AS\_MATRIX}
\texttt{\% objects.}
\texttt{\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,}
\texttt{\% Sandia National Laboratories, 2004. Please address questions or}
\texttt{\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,}
\texttt{\% distribute, display, and use this work, under the following}
\texttt{\% conditions. (1) You must give the original authors credit. (2) You may}
\texttt{\% not use or redistribute this work for commercial purposes. (3) You may}
\texttt{\% not alter, transform, or build upon this work. (4) For any reuse or}
\texttt{\% redistribution, you must make clear to others the license terms of this}
\texttt{\% work. (5) Any of these conditions can be waived if you get permission}
\texttt{\% from the authors.}
\texttt{if \texttt{\textasciitilde isa(a,\textquoteleft tensor\_as\_matrix\textquoteleft)}}
\texttt{\texttt{c = mtimes(tensor\_as\_matrix(a,1),b);}}
\texttt{\texttt{return;}}
\texttt{\texttt{end}}
\texttt{if \texttt{\textasciitilde isa(b,\textquoteleft tensor\_as\_matrix\textquoteleft)}}
\texttt{\texttt{c = mtimes(a,tensor\_as\_matrix(b,1));}}
\texttt{\texttt{return;}}
\texttt{end}
\texttt{if size(a,2) \texttt{\textasciitilde=} size(b,1)}
\texttt{\texttt{error([\textquoteleft Size mismatch: Number of columns in A is not equal to ' ...}
\texttt{\texttt{\textquoteright the number of rows in B'	extquoteright]);}}
\texttt{end}
\texttt{c = a; c.tsize = [a.tsize(a.rindices) b.tsize(b.cindices)];}
\texttt{c.rindices = [1:length(a.rindices)]; c.cindices = [1:length(b.cindices)] + length(a.rindices);}
\texttt{c.data = a.data \texttt{* b.data;}}

\texttt{function ss = size(a, idx)}
\texttt{\% TENSOR\_AS\_MATRIX/size Size of matrix.}
\texttt{\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,}
\texttt{\% Sandia National Laboratories, 2004. Please address questions or}
\texttt{\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,}
\texttt{\% distribute, display, and use this work, under the following}
\texttt{\% conditions. (1) You must give the original authors credit. (2) You may}
\texttt{\% not use or redistribute this work for commercial purposes. (3) You may}
\texttt{\% not alter, transform, or build upon this work. (4) For any reuse or}
\texttt{\% redistribution, you must make clear to others the license terms of this}
\texttt{\% work. (5) Any of these conditions can be waived if you get permission}
\texttt{\% from the authors.}
\texttt{switch s.type}
\texttt{case \textquoteleft .\textquoteright\texttt{}}
\texttt{switch s.subs}
\texttt{case \textquoteleft data\textquoteright\texttt{}}
\texttt{a = t.data;}
\texttt{case \textquoteleft tsize\textquoteright\texttt{}}
\texttt{a = t.tsize;}
\texttt{case \textquoteleft rindices\textquoteright\texttt{}}
\texttt{a = t.rindices;}
\texttt{case \textquoteleft cindices\textquoteright\texttt{}}
\texttt{a = c.cindices;}
\texttt{otherwise}
\texttt{error([\textquoteleft No such field: ', s.subs]);}
\texttt{end}
\texttt{otherwise}
\texttt{error('Invalid subsref into tensor\_as\_matrix.\textquoteright\texttt{)}}
\texttt{end}

\texttt{function s = subsref(t, s)}
\texttt{\% TENSOR\_AS\_MATRIX/subsref Subscripted reference.}
\texttt{\% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,}
\texttt{\% Sandia National Laboratories, 2004. Please address questions or}
\texttt{\% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,}
\texttt{\% distribute, display, and use this work, under the following}
\texttt{\% conditions. (1) You must give the original authors credit. (2) You may}
\texttt{\% not use or redistribute this work for commercial purposes. (3) You may}
\texttt{\% not alter, transform, or build upon this work. (4) For any reuse or}
\texttt{\% redistribution, you must make clear to others the license terms of this}
\texttt{\% work. (5) Any of these conditions can be waived if you get permission}
\texttt{\% from the authors.}
\texttt{switch s.type}
\texttt{case \textquoteleft .\textquoteright\texttt{}}
\texttt{switch s.subs}
\texttt{case \textquoteleft data\textquoteright\texttt{}}
\texttt{a = t.data;}
\texttt{case \textquoteleft tsize\textquoteright\texttt{}}
\texttt{a = t.tsize;}
\texttt{case \textquoteleft rindices\textquoteright\texttt{}}
\texttt{a = t.rindices;}
\texttt{case \textquoteleft cindices\textquoteright\texttt{}}
\texttt{a = c.cindices;}
\texttt{otherwise}
\texttt{error([\textquoteleft No such field: ', s.subs]);}
\texttt{end}
\texttt{otherwise}
\texttt{error('Invalid subsref into tensor\_as\_matrix.\textquoteright\texttt{)}}
\texttt{end}
function t = cp_tensor(varargin)
    % CP_TENSOR Tensor stored in CANDECOMP/PARAFAC form.
    % CP_TENSOR(T) creates a CP tensor by copying an existing CP tensor.
    % CP_TENSOR(lambda,U1,U2,...,UM) creates a CP tensor from its constituent parts. Here lambda is a k-vector and each Um is a matrix with k columns.
    % CP_TENSOR(lambda, U) is the same as above except that U is a cell array containing matrix Um in cell m.
    % See also TENSOR and TUCKER_TENSOR.
    if nargin == 1 & isa(varargin{1}, 'cp_tensor')
        t.lambda = varargin{1}.lambda;
        t.u = varargin{1}.
        t = class(t,'cp_tensor');
    end
    t.lambda = varargin{1}.
    if isa(varargin{2}, 'cell')
        t.u = varargin{2};
    else
        error('One or more of the specified dimensions is out of range');
    end
    % Copy CONSTRUCTOR
    if nargin == 1 & isa(varargin{1}, 'cp_tensor')
        t.lambda = varargin{1}.lambda;
        t.u = varargin{1}.
        t = class(t,'cp_tensor');
    end
    t.lambda = varargin{1}.
    if isa(varargin{2}, 'cell')
        t.u = varargin{2};
    else
        error('One or more of the specified dimensions is out of range');
    end
    % Check that each Um is indeed a matrix
    for i = 1 : length(t.lambda)
        if ndims(t.u{i}) ~= 2 | size(t.u{i},2) ~= 1
            error('Matrix U#{i} is not a matrix!');
        end
    end
    % Size error checking
    k = length(t.lambda);
    for i = 1 : length(t.u)
        if size(t.u{i},2) ~= k
            error('Matrix U#{i} is not a matrix!');
        end
    end
end

% TENSOR Constructor for matrix representation of a tensor
T = tensor(A, rdims, arg3) ;
TENSOR Constructor
T = tensor(T, RDIMS) ;
TENSOR Constructor
T = tensor(T); % Default construction
TENSOR Constructor
T = tensor(T, RDIMS, CDIMS) ;
TENSOR Constructor
T = tensor(T, RDIMS, STR) ;
TENSOR Constructor
T = tensor(T, RDIMS, CDIMS, STR) ;
TENSOR Constructor
T = tensor(T, RDIMS, CDIMS, STR, ORDER) ;
TENSOR Constructor
```matlab
function disp(t)
%CP_TENSOR/DISP Command window display.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tpgolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

disp(' CP tensor of size ');
disp(size(t));
for j = 1 : order(t)
    disp(['U{', int2str(j), '} = ']);
disp(t.u{j});
end
%−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
function printsize(sz)
    for i = 1 : length(sz) - 1
        fprintf(1,'%d x ',sz(i));
    end
    fprintf(1,'%d', sz(length(sz)));
```

```matlab
function display(t)
%CP_TENSOR/DISPLAY Command window display.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tpgolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

fprintf(1,'
');
fprintf(1,'%s is a CP tensor of size ', inputname(1));
printsize(size(t));
fprintf(1,'
');
disp(['lambda = ']);
disp(t.lambda);
for j = 1 : order(t)
    disp([inputname(1), '.U{', int2str(j), '} = ']);
disp(t.u{j});
end
%−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
function printsize(sz)
    for i = 1 : length(sz) - 1
        fprintf(1,'%d x ',sz(i));
    end
    fprintf(1,'%d', sz(length(sz)));
```

```matlab
function a = double(t)
%CP_TENSOR/DOWNCONV Convert tensor to double array.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tpgolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

error('Use double(full(t)).');
```

```matlab
function disp(t)
%CP_TENSOR/DISP Command window display.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tpgolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

disp(' CP tensor of size ');
disp(size(t));
disp(['lambda = ']);
disp(t.lambda);
for j = 1 : order(t)
    disp(['U{', int2str(j), '} = ']);
disp(t.u{j});
end
%−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−−
function printsize(sz)
    for i = 1 : length(sz) - 1
        fprintf(1,'%d x ',sz(i));
    end
    fprintf(1,'%d', sz(length(sz)));```
function t = full(t)
% CP_TENSOR/FULL Convert CP tensor to a dense tensor.
% t = full(t) converts CP tensor to a dense tensor.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

I = length(t.lambda);
M = order(t);
I = size(t);
for k = 1 : K
    % Add in rank-1 matrix corresponding to
    % lambda(k)
    tmp = 1;
    for m = 1 : M
        tmp = tmp * t.u{m}(:,k)';
        tmp = reshape(tmp, prod(I(1:m)), 1);
    end
    if length(I) == 1
        tmpI = [I 1];
    else
        tmpI = I;
    end
    tmp = reshape(tmp, tmpI);
    if k == 1
        a = t.lambda(k) * tmp;
    else
        a = a + t.lambda(k) * tmp;
    end
end
return;

function C = ge(A,B)
% CP_TENSOR/GE Greater than or equal.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.
error('Use ge(full(A),full(B)).');
function C = ldivide(A,B)
%CP_TENSOR/LDIVIDE Left array divide.

% Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use and(ldivide(A),full(B)).');
Function C = mtimes(A,B)
CP_TENSOR/MTIMES Implement A*B.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

% Note: We can do scalar times a tensor, but anything more complex is an error.
if isa(B,'numeric') & size(B) == [1 1]
    c = cp_tensor(B * A.lambda, A.u);
elseif isa(A,'numeric') & size(A) == [1 1]
    c = cp_tensor(A * B.lambda, B.u);
else
    error('Use mtimes(full(A),full(B)).');
end

Function n = ndims(T)
CP_TENSOR/NDIM Return the number of dimensions

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

n = order(t);

Function n = norm(T)
CP_TENSOR/NORM Frobenius norm.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use norm(full(A)).');

Function B = not(A)
CP_TENSOR/NOT Logical NOT.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use not(full(A)).');
Function \( C = \text{or}(A,B) \)

\[ \text{CP_TENSOR/OR Logical OR.} \]

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

```matlab
error('Use or(full(A),full(B)).');
```

Function \( n = \text{order}(t) \)

\[ \text{CP_TENSOR/ORDER Return the number of dimensions} \]

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

```matlab
n = length(t.u);
```

Function \( b = \text{permute}(a,\text{order}) \)

\[ \text{CP_TENSOR/PERMUTE Permute dimensions.} \]

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

```matlab
lambda = a.lambda(order);
for i = 1 : length(order)
    u{i} = a.u(order(i));
end
b = cp_tensor(lambda, u);
```

Function \( C = \text{plus}(A,B) \)

\[ \text{CP_TENSOR/PLUS Binary addition.} \]

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

```matlab
if (isa(A,'cp_tensor') & isa(B,'cp_tensor'))
    if ~( isequal(A,B) )
        error('Tensor size mismatch.');
    end
    lambda = [A.lambda; B.lambda];
    N = order(A);
    for m = 1 : N
        u(m) = [A.u(m) B.u(m)];
    end
    C = cp_tensor(lambda, u);
    return;
end
error('Use plus(full(A),full(B)).');
```
function C = power(A,B)
%TENSOR/POWER
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.
error('Use power(full(A),full(B)).');

function m = size(t,idx)
%CP_TENSOR/SIZE Size of tensor.
%
%   D = SIZE(T) returns the size of the tensor.
%   I = size(T,DIM) returns the size of the dimension specified by
%   the scalar DIM.
%
% See also ORDER, NDIMS.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.
if exist('idx','var')
m = size(t.u(idx), 1);
else
  for i = 1 : order(t)
    m(i) = size(t.u{i}, 1);
  end
end

function t = subsasgn(t,s,b)
%CP_TENSOR/SUBASGN Subscripted reference.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.
switch s.type
  case '.
    switch s.subs
      case 'lambda'
        t = cp_tensor(b, t.u);
      otherwise
        error(['Cannot change field ', s.subs, ' directly']);
    end
  case '()' error('Cannot change individual entries in CP tensor.');
  otherwise error('Invalid subsasgn.');
end
function a = subsref(t,s)
    %CP_TENSOR/SUBSREF Subscripted reference.
    %
    %Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

    switch s.type
    case '.
        switch s.subs
        case 'lambda'
            a = t.lambda;
        case 'u'
            a = t.u;
        otherwise
            error(’No such field: ’, s.subs);
        end
    case '()'
        a = 0;
        for k = 1 : length(t.lambda)
            b = 1;
            for i = 1 : length(s.subs)
                b = b * t.u{i}(s.subs{i},k);
            end
            a  = a + t.lambda(k) * b;
        end
    case '{}'
        a = t.u{s.subs{:}};
    otherwise
        error(’Invalid subsref.’);
    end

function C = times(A,B)
    %CP_TENSOR/TIMES Element-wise multiplication.
    %
    %Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

    error(’Use times(full(A),full(B)).’);

function t = uminus(t)
    %CP_TENSOR/UMINUS Unary minus for tensors.
    %
    %Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

    t.lambda = -t.lambda;

function t = uplus(t)
    %CP_TENSOR/UPLUS Unary plus for tensors.
    %
    %Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

    % This function does nothing!
Function C = xor(A,B)

CP_TENSOR/XOR Logical EXCLUSIVE OR.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use xor(full(A),full(B)).');

function C = and(A,B)

TUCKER_TENSOR/AND Logical AND.

Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use and(full(A),full(B)).');
Function: `a = double(t)`  
TUCKER_TENSOR/DIST Convert tensor to double array.  
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,  
Sandia National Laboratories, 2004. Please address questions or  
comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,  
distribute, display, and use this work, under the following  
conditions. (1) You must give the original authors credit. (2) You may  
not use or redistribute this work for commercial purposes. (3) You may  
not alter, transform, or build upon this work. (4) For any reuse or  
distribution, you must make clear to others the license terms of this  
work. (5) Any of these conditions can be waived if you get permission  
from the authors.  
error('Use double(full(t)).');

Function: `A = full(B)`  
TUCKER_TENSOR/FULL Convert to a dense tensor.  
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,  
Sandia National Laboratories, 2004. Please address questions or  
comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,  
distribute, display, and use this work, under the following  
conditions. (1) You must give the original authors credit. (2) You may  
not use or redistribute this work for commercial purposes. (3) You may  
not alter, transform, or build upon this work. (4) For any reuse or  
distribution, you must make clear to others the license terms of this  
work. (5) Any of these conditions can be waived if you get permission  
from the authors.  
M = order(B);  
tmp = ttm(B.lambda, B.u{1}, 1);  
for m = 2 : M  
tmp = ttm(tmp, B.u{m}, m);  
end  
A = tensor(tmp, size(B));  
return;

Function: `C = ge(A,B)`  
TUCKER_TENSOR/GE Greater than or equal.  
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,  
Sandia National Laboratories, 2004. Please address questions or  
comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,  
distribute, display, and use this work, under the following  
conditions. (1) You must give the original authors credit. (2) You may  
not use or redistribute this work for commercial purposes. (3) You may  
not alter, transform, or build upon this work. (4) For any reuse or  
distribution, you must make clear to others the license terms of this  
work. (5) Any of these conditions can be waived if you get permission  
from the authors.  
error('Use ge(full(A),full(B)).');

Function: `C = gt(A,B)`  
TUCKER_TENSOR/GT Greater than.  
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,  
Sandia National Laboratories, 2004. Please address questions or  
comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,  
distribute, display, and use this work, under the following  
conditions. (1) You must give the original authors credit. (2) You may  
not use or redistribute this work for commercial purposes. (3) You may  
not alter, transform, or build upon this work. (4) For any reuse or  
distribution, you must make clear to others the license terms of this  
work. (5) Any of these conditions can be waived if you get permission  
from the authors.  
error('Use gt(full(A),full(B)).');
```matlab
function b = issamesize(A,B)
% TUCKER_TENSOR/ISSAMESIZE

% ISSAMESIZE(A,B) returns true if tensors A and B are the same size.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

if ((ndims(A) == ndims(B)) & (size(A) == size(B)))
  b = (1==1);   % true
else
  b = (0==1);   % false
end
```

```matlab
function C = ldivide(A,B)
% TUCKER_TENSOR/LDIVIDE Left array divide.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

error('Use ldivide(full(A),full(B)).');
```

```matlab
function C = le(A,B)
% TUCKER_TENSOR/LE Less than or equal.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

error('Use le(full(A),full(B)).');
```

```matlab
function C = lt(A,B)
% TUCKER_TENSOR/LT Less than.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions. (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

error('Use lt(full(A),full(B)).');
```
function C = minus(A,B)

%TUCKER_TENSOR/MINUS Binary subtraction.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions: (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

error('Use minus(full(A),full(B)).');

function C = mtimes(A,B)

%TUCKER_TENSOR/MTIMES Implement A*B.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions: (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

% Note: We can do scalar times a tensor, but anything more complex is
% an error.

if (prod(size(B)) == 1)
    C = tucker_tensor(B * A.lambda, A.u);
elseif (prod(size(A)) == 1)
    C = tucker_tensor(A * B.lambda, B.u);
else
    error('Use mtimes(full(A),full(B)).');
end

function n = ndims(t)

%TUCKER_TENSOR/NDIMS Return the number of dimensions.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions: (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

n = order(t);

function n = norm(T)

%TUCKER_TENSOR/NORM Frobenius norm.
% Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
% Sandia National Laboratories, 2004. Please address questions or
% comments to: tgkolda@sandia.gov. Terms of use: You are free to copy,
% distribute, display, and use this work, under the following
% conditions: (1) You must give the original authors credit. (2) You may
% not use or redistribute this work for commercial purposes. (3) You may
% not alter, transform, or build upon this work. (4) For any reuse or
% redistribution, you must make clear to others the license terms of this
% work. (5) Any of these conditions can be waived if you get permission
% from the authors.

error('Use norm(full(A)).');
Function: B = not(A)
TUCKER_TENSOR/NOT Logical NOT.

Function: C = or(A,B)
TUCKER_TENSOR/OR Logical OR.

Function: n = order(t)
TUCKER_TENSOR/ORDER Return the number of dimensions

Function: b = permute(a,order)
TUCKER_TENSOR/PERMUTE Permute dimensions.
Function:

```matlab
function C = plus(A,B)
%TUCKER_TENSOR/PLUS Binary addition.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004.  Please address questions or
%comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

error('Use plus(full(A),full(B)).');
```

Function:

```matlab
function C = power(A,B)
%TUCKER_TENSOR/POWER
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004.  Please address questions or
%comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

error('Use power(full(A),full(B)).');
```

Function:

```matlab
function C = rdivide(A,B)
%TUCKER_TENSOR/RDIVIDE Right array divide.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004.  Please address questions or
%comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

error('Use rdivide(full(A),full(B)).');
```

Function:

```matlab
function m = size(t,idx)
%TUCKER_TENSOR/SIZE Size of tensor.
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004−3360,
%Sandia National Laboratories, 2004.  Please address questions or
%comments to: tgkolda@sandia.gov.  Terms of use: You are free to copy,
%distribute, display, and use this work, under the following
%conditions. (1) You must give the original authors credit. (2) You may
%not use or redistribute this work for commercial purposes. (3) You may
%not alter, transform, or build upon this work. (4) For any reuse or
%distribution, you must make clear to others the license terms of this
%work. (5) Any of these conditions can be waived if you get permission
%from the authors.

if exist('idx','var')
    m = size(t.u(idx), 1);
else
    for i = 1 : order(t)
        m(i) = size(t.u{i}, 1);
    end
end
```

MATLAB Tensor Classes by B. W. Bader and T. G. Kolda

Thursday October 07, 2004
t = tucker_tensor(varargin)

function a = subsref(t,s)

function a = subsasgn(t,s,b)

function t = times(u,B)

function a = times(full(A),full(B))
function t = uminus(t)
%TUCKER_TENSOR/UMINUS Unary minus for tensors.
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tpgolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

t.lambda = -t.lambda;

function t = uplus(t)
%TUCKER_TENSOR/UPLUS Unary plus for tensors.
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tpgolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

% This function does nothing!

function C = xor(A,B)
%TUCKER_TENSOR/XOR Logical EXCLUSIVE OR.
Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360, Sandia National Laboratories, 2004. Please address questions or comments to: tpgolda@sandia.gov. Terms of use: You are free to copy, redistribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or redistribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use xor(full(A),full(B)).');