

SANDIA REPORT

SAND2004-XXXX
Unlimited Release
Printed October 2004

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.



Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

NOTICE: This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government, nor any agency thereof, nor any of their employees, nor any of their contractors, subcontractors, or their employees, make any warranty, express or implied, or assume any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represent that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government, any agency thereof, or any of their contractors or subcontractors. The views and opinions expressed herein do not necessarily state or reflect those of the United States Government, any agency thereof, or any of their contractors.

Printed in the United States of America. This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from

U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865)576-8401
Facsimile: (865)576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: <http://www.doe.gov/bridge>

Available to the public from

U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd
Springfield, VA 22161

Telephone: (800)553-6847
Facsimile: (703)605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: <http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online>



SAND2004-XXXX
Unlimited Release
Printed October 2004

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

This page intentionally left blank.

README.txt

Page 1/1

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:

```
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: tkgolda@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.

@tensor/and.m

Page 1/1

```
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: tkgolda@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

Page 1/2

```
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: tkgolda@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(1,'\n');
end

fprintf(1,'%s of ',name);
printsize(t.size);
fprintf(1,'\n');

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

fprintf(1,'%s',namedot);
if isempty(t.data)
    fprintf(1,'data = []\n');
else
    fprintf(1,'data = \n');
    disp(t.data);
end

function printsize(sz)

if length(sz) == 0
    fprintf(1,'order 0 (i.e., a scalar)');
    return;
end

fprintf(1,'size ');
for i = 1 : length(sz) - 1
    fprintf(1,'%d x ',sz(i));
end
```

@tensor/disp.m

Page 2/2

```
fprintf(1,'%d', sz(length(sz)));
```

@tensor/display.m

Page 1/1

```
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: tkgolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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(t,inputname(1));
```

@tensor/double.m

Page 1/1

```
function a = double(t)
%TENSOR/DOUBLE 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: tkgolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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 = t.data;
```

@tensor/ge.m

Page 1/1

```
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: tkgolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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(@ge,A,B);
```

@tensor/gt.m

Page 1/1

```
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: tkgolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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(@gt,A,B);
```

@tensor/issamesize.m

Page 1/1

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

@tensor/ldivide.m

Page 1/1

```
function C = ldivide(A,B)
%TENSOR/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
%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(@ldivide,A,B);
```

@tensor/le.m

Page 1/1

```
function C = le(A,B)
%TENSOR/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
%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(@le,A,B);
```

@tensor/lt.m

Page 1/1

```
function C = lt(A,B)
%TENSOR/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
%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(@lt,A,B);
```

@tensor/minus.m

Page 1/1

```
function C = minus(A,B)
%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: tkgolda@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(@minus,A,B);
```

@tensor/mtimes.m

Page 1/1

```
function C = mtimes(A,B)
%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: tkgolda@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 (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);
```

@tensor/multiarrayop.m

Page 1/1

```
function C = multiarrayop(fname,A,B)
%TENSOR/MULTIARRAYOP Generic multidimensional array functions for tensors.
%
% MULTIARRAYOP(F,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: tkgolda@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 prod(size(A.data)) == 1
    sz = size(B);
else
    sz = size(A);
end

C = feval(fname, A.data, B.data);
C = tensor(C, sz);
```

@tensor/ndims.m

Page 1/1

```
function 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: tkgolda@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.
```

```
n = max(length(t.size), 1);
```

@tensor/norm.m

Page 1/1

```
function n = norm(T)
%TENSOR/NORM Frobenius norm of a tensor.
%
%   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
%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 = T.^2;
%T.data = reshape(T.data,1,prod(size(T)));
%n = sqrt(sum(T.data));

v = reshape(T.data, prod(size(T.data)), 1);
n = norm(v);
```

@tensor/not.m

Page 1/1

```
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
%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.

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

@tensor/or.m

Page 1/1

```
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
%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(@or,A,B);
```

@tensor/order.m

Page 1/1

```
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
%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.

n = length(t.size);
```

@tensor/permute.m

Page 1/1

```
function T = permute(T,Idx)
%TENSOR/PERMUTE Permute tensor dimensions
%
% B = PERMUTE(A,ORDER) rearranges the dimensions of A so that they
% are in the order specified by the vector ORDER. The result has the
% same values of A, 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: tkgolda@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 length(Idx) == 1
    if (Idx == 1)
        return;
    else
        error('Invalid Order.');
```

@tensor/plus.m

Page 1/1

```
function C = plus(A,B)
%TENSOR/PLUS Binary addition for tensors.
%
% 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.
%
% C = PLUS(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: tkgolda@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.')
```

@tensor/power.m

Page 1/1

```
function C = power(A,B)
%TENSOR/POWER
%
% Z = X.^Y denotes element-by-element powers. X and Y must have the
% same dimensions unless one is a scalar. A scalar can operate into
% anything.
%
% C = POWER(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: tkgolda@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.')
```

@tensor/rdivide.m

Page 1/1

```
function C = rdivide(A,B)
% TENSOR/RDIVIDE Right 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 = 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: tkgolda@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.')
```

@tensor/shiftdim.m

Page 1/1

```
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
%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 = varargin{1};
n = varargin{2};

B = feval(@shiftdim, A,data, n);
B = tensor(B, size(B));
```

@tensor/size.m

Page 1/1

```
function m = size(t,idx)
% TENSOR/SIZE Size of tensor.
%
% D = SIZE(T) returns the size of the tensor. Trailing singleton
% dimensions will be reported if they are included explicitly in 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 = t.size(idx);

else

    m = t.size;

end
```

@tensor/squeeze.m

Page 1/1

```
function B = squeeze(A)
%SQUEEZE Remove singleton dimensions.
%
% B = SQUEEZE(A) returns a tensor B with the same elements as
% A but with all the singleton dimensions removed. A singleton
% is a dimension such that size(A,dim)=1.
%
% For example,
% squeeze( tensor(rand(2,1,3)) )
% is a 2-by-3 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.

sizeA = size(A);
B = squeeze(A,data);
B = tensor(B, sizeA(sizeA > 1));
```

@tensor/subsasgn.m

Page 1/1

```
function t = subsasgn(t,s,b)
%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 '.'
    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.m

Page 1/1

```
function a = subsref(t,s)
%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 'data'
        a = t.data;
    case 'size'
        a = t.size;
    otherwise
        error(['No such field: ', s.subs]);
    end
case '{}'
    a = t.data(s.subs{:});
    if prod(size(a)) > 1
        a = tensor(a);
    end
case '{}'
    error(['Subscript cell reference cannot be used for dense tensors.'])
otherwise
    error('Incorrect indexing into tensor.')
end
```

@tensor/tensor.m

Page 1/3

```
function t = tensor(varargin)
% TENSOR Tensor class constructor.
%
% 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) creates a tensor from the multidimensional array Z.
%
% TENSOR(Z,DIMS) creates a tensor from the multidimensional array
% Z. The DIMS argument is used to specify any trialing 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,I,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:
%
% - 'DDV': 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.
%
%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.

% EMPTY/DEFAULT CONSTRUCTOR
if size(varargin) == 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
```

@tensor/tensor.m

Page 2/3

```
end

% CONVERT CP_TENSOR OR TUCKER_TENSOR
if isa(varargin{1}, 'cp_tensor') | isa(varargin{1}, 'tucker_tensor')
    t = full(varargin{1});
    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 ~isempty(sz)
        sz = sz(idx);
    end
    t.data = reshape(a.data, [sz 1 1]);
    t.size = sz;
    t = class(t, 'tensor');
    [sidx indx] = sort(idx);
    t = permute(t,indx);
    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
        t.size = varargin{2};
        if (length(t.size) > 2) & (size(t.size,1) ~= 1)
            error('DIMS must be a row vector.');
```

@tensor/tensor.m

Page 3/3

```
end

% Second, check that the remaining dimensions are okay
for i = j + 1 : length(sz)
    if (sz(i) ~= 1)
        error('Specified size is incorrect.');
```

```
end

t = class(t, 'tensor');
return;

error('Unsupported use of function TENSOR.');
```

```

@tensor/times.m Page 1/1
function C = times(A,B)
% TENSOR/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.
%
%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 (prod(size(A)) == 1) | (prod(size(B)) == 1)
    C = tensor(A.data * B.data);
    return;
end

if ~issamesize(A,B)
    error('Tensor order and size must agree.');
```

```

@tensor/ttm.m Page 1/3
function c = ttm(a,v,n)
% 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,I}, then A must have size(A,N) = I. 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, X be a
% matrix with J columns, Y be a matrix with K columns, and Z be a
% matrix with L columns. Let U = {V,X,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,X,Y,Z}, [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) computes B = A x_1 V x_2 X x_3 Y x_4 Z.
%
% ----
%
% Ex: B = ttm(A, {Y,Z}, [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,X,Z}, [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,X,Z}, -3) computes B = A x_1 V x_2 X x_4 Z.
%
% Ex: B = ttm(A, U, -3) computes B = A x_1 V x_2 X x_4 Z.
%
% ----
%
% See also TTT, TTV.
```

```

@tensor/ttm.m Page 2/3
%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 CHECKING ***
%*****

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

```

@tensor/ttm.m Page 3/3
%*****
% Convert v to a tensor
v = tensor(v);

% Compute product
c = ttt(a, v, n, 2);

% Permute rows to proper order in c
dims = [setdiff(1:ndims(a),n) n];
[sdims sidx] = sort(dims);
c = permute(c,sidx);
return;

%*****
function c = ttms(a,v,dims)

% Check validity of DIMS and then check for "minus" case
if (max(abs(dims)) > ndims(a))
    error('An entry in DIMS exceeds order of A.');
```

@tensor/ttt.m

Page 1/2

```
function c = ttt(varargin)
% TENSOR/TTT 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,setminus(1:ndims(A),ADIMS)) size(B,setminus(1:ndims(B),BDIMS))].
%
% TTT(A,B,DIMS) computes the inner product of tensors A and B in the
% dimensions specified by the vector DIMS. The sizes of the
% dimensions specified by DIMS must match; that is, size(A,DIMS) must
% equal size(B,DIMS). If DIMS = 1:N, 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,
% 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 CHECKING **
%*****
% Check the number of arguments
if (nargin < 2)
    error('TTT requires at least two arguments.');
```

@tensor/ttt.m

Page 2/2

```
if nargin >= 4
    bdims = varargin{4};
else
    bdims = adims;
end
if ~isequal(size(adims),size(bdims))
    error('Dimensions specified by ADIMS and BDIMS do not match.');
```

```
c = tensor(cmatrix);
```

@tensor/ttv.m

Page 1/3

```
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) = [1,1], then A must have size(A,N) =
% 1. Note that ndims(B) = ndims(A) - 1 because the N-th dimension
% is removed.
%
% B = TTV(A,U) computes the product of tensor A with a sequence of
% vectors in the cell array U. The products are computed
% sequentially along all dimensions (or modes) of A. The cell array
% U contains ndims(A) vectors.
%
% 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
% columns 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 V.
%
% ----
% 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 V *2 X *3 Y *4 Z.
% Ex: B = ttv(A, U) computes B = A *1 V *2 X *3 Y *4 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 V *2 X *4 Z.
% Ex: B = ttv(A, {V,X,Z}, -3) computes B = A *1 V *2 X *4 Z.
% Ex: B = ttv(A, U, -3) computes B = A *1 V *2 X *4 Z.
%
% ----
% See also TTT, TTM.
```

@tensor/ttv.m

Page 2/3

```
% 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 CHECKING **
%*****
% Check the number of arguments
if (nargin < 2)
    error('TTV requires at least two arguments.');
```

```
if isa(v,'cell')
```

```
if (nargin < 3)
```

```
n = [1:ndims(a)];
```

```
end
```

```
c = ttvs(a,v,n);
```

```
return;
```

```
end
```

```
%** ERROR CHECKING **
```

@tensor/ttv.m

Page 3/3

```

%%% COMPUTE THE PRODUCT %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Convert v to a tensor
v = tensor(v, length(v));

% Compute product
c = ttv(a,v,n,1);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function c = ttvs(a,v,dims)

% Check validity of DIMS and then check for "minus" case
if (max(abs(dims)) > ndims(a))
    error('An entry in DIMS exceeds order of A.');
```

@tensor/uminus.m

Page 1/1

```

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;
```

@tensor/uplus.m

Page 1/1

```

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!
```

@tensor/xor.m

Page 1/1

```

function C = xor(A,B)
%TENSOR/XOR Logical EXCLUSIVE OR.
%
% XOR(A,B) is the logical symmetric difference of elements A and B.
% The result is one where either A or B, but not both, is nonzero.
% The result is zero where A and B are both zero or nonzero. A and
% B must have the same dimensions (or one can be 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,
%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.')
```

@tensor_as_matrix/ctranspose.m Page 1/1

```
function a = ctranspose(a)
% TENSOR_AS_MATRIX/CTRANSPOSE Complex conjugate transpose.
%
% C = CTRANSPOSE(A) swaps the row and column indices of A.
%
%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.

tmp = a.rindices;
a.rindices = a.cindices;
a.cindices = tmp;
a.data = a.data';
```

@tensor_as_matrix/disp.m Page 1/2

```
function disp(t,name)
%TENSOR_AS_MATRIX/DISP Command window display of a tensor stored as
%a matrix.
%
%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_as_Matrix';
else
    namedot = [name '.'];
    name = [name ' is a matrix'];
end

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

if skipspace ~= 1
    fprintf(1,'\n');
end

fprintf(1,'%s corresponding to a tensor of size ',name);
printsize(t,tsize);
fprintf(1,'\n');

fprintf(1,'Row index spans tensor mode(s) ');
printvec(t.rindices);
fprintf(1,'\n');
fprintf(1,'Column index spans tensor mode(s) ');
printvec(t.cindices);
fprintf(1,'\n');

if skipspace ~= 1
    fprintf(1,'\n');
end

fprintf(1,'%s',namedot);
if isempty(t.data)
    fprintf(1,'data = []\n');
else
    fprintf(1,'data = \n');
    disp(t.data);
end

function printsize(v)
n = length(v);
```

@tensor_as_matrix/disp.m Page 2/2

```
for i = 1 : n - 1
    fprintf(1,'%d x ',v(i));
end
if (n > 0)
    fprintf(1,'%d', v(n));
else
    fprintf(1,'0');
end

function printvec(v)
n = length(v);
fprintf(1,'[');
for i = 1 : n - 1
    fprintf(1,'%d, ',v(i));
end
if (n > 0)
    fprintf(1,'%d', v(n));
end
fprintf(1,']');
```

@tensor_as_matrix/display.m Page 1/1

```
function display(t)
%TENSOR_AS_MATRIX/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
%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(t,inputname(1));
```

@tensor_as_matrix/double.m

Page 1/1

```
function a = double(t)
%TENSOR_AS_MATRIX/DOUBLE 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: tgcolda@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 = t.data;
```

@tensor_as_matrix/mtimes.m

Page 1/1

```
function c = mtimes(a,b)
%TENSOR_AS_MATRIX/MTIMES Multiplies two tensor_as_matrix objects.
%
% C = MTIMES(A,B) computes the product of A and B. The result is a
% TENSOR_AS_MATRIX object and can be transformed into a tensor.
%
% C = MTIMES(A,B) is called for the syntax 'A * B' when A or B is a
% TENSOR_AS_MATRIX object. Both A and B must be TENSOR_AS_MATRIX
% objects.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgcolda@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 ~isa(a,'tensor_as_matrix')
    c = mtimes(tensor_as_matrix(a,1),b);
end

if ~isa(b,'tensor_as_matrix')
    c = mtimes(a,tensor_as_matrix(b,1));
end

if size(a,2) ~= size(b,1)
    error(['Size mismatch: Number of columns in A is not equal to' ...
          ' the number of rows in B']);
end

c = a;
c.tsize = [a.tsize(a.rindices) b.tsize(b.cindices)];
c.rindices = 1:length(a.rindices);
c.cindices = [1:length(b.cindices)] + length(a.rindices);
c.data = a.data * b.data;
```

@tensor_as_matrix/size.m

Page 1/1

```
function sz = size(a,idx)
%TENSOR_AS_MATRIX/SIZE Size of matrix.
%
% D = SIZE(X) returns the two-element row vector D = [M, N]
% containing the number of rows and columns in the matrix.
%
% M = SIZE(X,DIM) returns the length of the dimension specified by
% the scalar DIM. For example, SIZE(X,1) returns the number of
% rows.
%
% See also TSIZE.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgcolda@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')
    sz = size(a.data, idx);
else
    sz = size(a.data);
end
```

@tensor_as_matrix/subsref.m

Page 1/1

```
function a = subsref(t,s)
%TENSOR_AS_MATRIX/SUBSREF Subscripted reference.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tgcolda@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 'data'
        a = t.data;
    case 'tsize'
        a = t.tsize;
    case 'rindices'
        a = t.rindices;
    case 'cindices'
        a = t.cindices;
    otherwise
        error(['No such field: ', s.subs]);
    end
otherwise
    error('Invalid subsref into tensor_as_matrix.')
```

@tensor_as_matrix/tensor_as_matrix.m Page 1/2

```

function A = tensor_as_matrix(T, rdims, arg3)
% TENSOR_AS_MATRIX Constructor for matrix representation of a tensor
%
% TENSOR_AS_MATRIX(T, RDIMS) creates a matrix representation of a
% tensor T by rearranging and reshaping T. The dimensions (or modes)
% specified in RDIMS map to the rows of the matrix, and the
% remaining dimensions (in ascending order) map to the columns.
%
% TENSOR_AS_MATRIX(T, RDIMS, CDIMS) creates a matrix representation
% of tensor T. The dimensions specified in RDIMS map to the rows of
% the matrix, and the dimensions specified in CDIMS map to the
% columns, in the order given.
%
% TENSOR_AS_MATRIX(T, RDIM, STR) creates the same matrix
% representation as above, except only one dimension in RDIM maps to
% the rows of the matrix, and the remaining dimensions span the
% columns in an order specified by the string argument STR as
% follows:
%
% 'fc' - Forward cyclic. Order the remaining dimensions in the
% columns by [RDIM+1:ndims(T), 1:RDIM-1]. This is the
% ordering defined by Kiers.
%
% 'bc' - Backward cyclic. Order the remaining dimensions in the
% columns by [RDIM-1:-1:1, ndims(T):-1:RDIM+1]. This is the
% ordering defined by De Lathauwer, De Moor, and Vandewalle.
%
% See also TENSOR.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@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 isa(T,'double')
    A = tensor_as_matrix(tensor(T),1);
    return;
end

% Save the size of T and the number of dimensions
tsize = size(T);
n = ndims(T);

% Compute the default set of column indices (cdims might get changed later)
cdims = setdiff(1:n, rdims);

% Process optional third argument
if exist('arg3','var')
    if isa(arg3,'char')
        if (numel(rdims) ~= 1)
            error('Only one row dimension when third argument is a string.');
        end
        if strcmp(arg3,'fc') % Forward cyclic.

```

@tensor_as_matrix/tensor_as_matrix.m Page 2/2

```

        cdims = [rdims+1:n, 1:rdims-1];
    elseif strcmp(arg3,'bc') % Backward cyclic.
        cdims = [rdims-1:-1:1, n:-1:rdims+1];
    else
        error('Unrecognized option');
    end
elseif isa(arg3,'numeric')
    cdims = arg3;
end

% Error check
if (max(rdims) > n) | (min(rdims) < 1)
    error('One or more of the specified dimensions is out of range');
elseif (max(cdims) > n) | (min(cdims) < 1)
    error('One or more of the specified dimensions is out of range');
elseif (length(rdims) ~= n) | setdiff(1:n, [rdims cdims])
    error('Incorrect specification of dimensions');
end

% Permute T so that the dimensions specified by RDIMS come first
T = permute(T,[rdims cdims]);

% Convert T to a matrix
data = reshape(T.data, prod(tsize(rdims)), prod(tsize(cdims)));

% Save class variables
A.tsize = tsize;
A.rindices = rdims;
A.cindices = cdims;
A.data = data;
A = class(A, 'tensor_as_matrix');
```

@cp_tensor/and.m Page 1/1

```

function C = and(A,B)
%CP_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: tskolda@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 and(full(A),full(B)).');
```

@cp_tensor/cp_tensor.m Page 1/2

```

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
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@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.
%
% Copy CONSTRUCTOR
if (nargin == 1) & isa(varargin{1}, 'cp_tensor')
    t.lambda = varargin{1}.lambda;
    t.u = varargin{1}.u;
    t = class(t, 'cp_tensor');
    return;
end

t.lambda = varargin{1};
if ~isa(t.lambda,'numeric') | ndims(t.lambda) ~= 2 | size(t.lambda,2) ~= 1
    error('LAMBDA must be a column vector.');
end

if isa(varargin{2},'cell')
    t.u = varargin{2};
else
    for i = 2 : nargin
        t.u[i-1] = varargin{i};
    end
end

% Check that each Um is indeed a matrix
for i = 1 : length(t.u)
    if ndims(t.u{i}) ~= 2
        error(['Matrix U' int2str(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' int2str(i) ' does not have ' int2str(k) ' columns.']);
    end
end

```

@cp_tensor/cp_tensor.m

Page 2/2

```

end
end
t = class(t, 'cp_tensor');
return;

```

@cp_tensor/disp.m

Page 1/1

```

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: tkgolda@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, '\n');
fprintf(1, 'CP tensor of size ');
printsize(size(t))
fprintf(1, '\n');

disp(' ');
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)));

```

@cp_tensor/display.m

Page 1/1

```

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: tkgolda@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, '\n');
fprintf(1, '%s is a CP tensor of size ', inputname(1));
printsize(size(t));
fprintf(1, '\n');

disp(' ');
disp([inputname(1), '.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)));

```

@cp_tensor/double.m

Page 1/1

```

function a = double(t)
%CP_TENSOR/DOUBLE 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: tkgolda@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)).');

```

@cp_tensor/full.m

Page 1/1

```

function t = full(t)
%CP_TENSOR/FULL Convert CP tensor to a dense tensor.
%
% 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,
%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.

K = 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

t = tensor(a, I);
return;

```

@cp_tensor/ge.m

Page 1/1

```

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,
%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)).');

```

@cp_tensor/gt.m

Page 1/1

```

function C = gt(A,B)
%CP_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)).');

```

@cp_tensor/issamesize.m

Page 1/1

```

function b = issamesize(A,B)
%CP_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
%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 ((ndims(A) == ndims(B)) & (size(A) == size(B)))
    b = (1==1); % true
else
    b = (0==1); % false
end

```

@cp_tensor/ldivide.m

Page 1/1

```
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: tskolda@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 and(ldivide(A),full(B)).');
```

@cp_tensor/le.m

Page 1/1

```
function C = le(A,B)
%CP_TENSOR/LT 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: tskolda@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 and(le(A),full(B)).');
```

@cp_tensor/lt.m

Page 1/1

```
function C = lt(A,B)
%CP_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: tskolda@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 and(lt(A),full(B)).');
```

@cp_tensor/minus.m

Page 1/1

```
function C = minus(A,B)
%CP_TENSOR/MINUS Binary subtraction.
%
% MINUS(A,B) subtracts tensor B from A. A and B must have the same
% dimensions.
%
% C = MINUS(A,B) is called for the syntax 'A - B' when A or B is a
% CP tensor.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@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 (isa(A,'cp_tensor') & isa(B,'cp_tensor'))
    if ~(issamesize(A,B))
        error('Tensor size mismatch.')
```

@cp_tensor/mtimes.m

Page 1/1

```
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: tkgolda@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.

% 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
```

@cp_tensor/ndims.m

Page 1/1

```
function n = ndims(t)
%CP_TENSOR/NDIMS Return the number of dimensions
%
% NDIMS(T) returns the number of dimensions of tensor T.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tkgolda@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.

n = order(t);
```

@cp_tensor/norm.m

Page 1/1

```
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: tkgolda@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 norm(full(A)).');
```

@cp_tensor/not.m

Page 1/1

```
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: tkgolda@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 not(full(A)).');
```

@cp_tensor/or.m

Page 1/1

```
function C = or(A,B)
%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,
%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 or(full(A),full(B)).');
```

@cp_tensor/order.m

Page 1/1

```
function n = order(t)
%CP_TENSOR/ORDER Return the number of dimensions
%
% ORDER(T) returns the number of dimensions of tensor T.
%
%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.

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

@cp_tensor/permute.m

Page 1/1

```
function b = permute(a,order)
%CP_TENSOR/PERMUTE Permute dimensions.
%
% B = PERMUTE(A,ORDER) rearranges the dimensions of A so that they
% are in the order specified by the vector ORDER. The tensor
% produced has the same values of A 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 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.

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

@cp_tensor/plus.m

Page 1/1

```
function C = plus(A,B)
%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,
%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 (isa(A,'cp_tensor') & isa(B,'cp_tensor'))

    if ~(issamesize(A,B))
        error('Tensor size mismatch.')
```

@cp_tensor/power.m

Page 1/1

```
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)).');
```

@cp_tensor/rdivide.m

Page 1/1

```
function C = rdivide(A,B)
%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)).');
```

@cp_tensor/size.m

Page 1/1

```
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
```

@cp_tensor/subsasgn.m

Page 1/1

```
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.')
```

```
case '{}'
    u = t.u;
    u{s.subs{:}} = b;
    t = cp_tensor(t.lambda, u);
otherwise
    error('Invalid subsasgn.');
```

```
end
```

@cp_tensor/subsref.m

Page 1/1

```
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.');
```

@cp_tensor/times.m

Page 1/1

```
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)).');
```

@cp_tensor/uminus.m

Page 1/1

```
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;
```

@cp_tensor/uplus.m

Page 1/1

```
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!
```

@cp_tensor/xor.m

Page 1/1

```
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: tskolda@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 xor(full(A),full(B)).');
```

@tucker_tensor/and.m

Page 1/1

```
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: tskolda@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 and(full(A),full(B)).');
```

@tucker_tensor/disp.m

Page 1/1

```
function disp(t)
%TUCKER_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: tskolda@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,'\n');
fprintf(1,'Tucker tensor of size ');
printsize(size(t))
fprintf(1,'\n');

disp(' ');
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)));
```

@tucker_tensor/display.m

Page 1/1

```
function display(t)
%TUCKER_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: tskolda@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,'\n');
fprintf(1,'%s is a Tucker tensor of size ', inputname(1));
printsize(size(t));
fprintf(1,'\n');

disp(' ');
disp([inputname(1), '.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)));
```

@tucker_tensor/double.m

Page 1/1

```
function a = double(t)
%TUCKER_TENSOR/DOUBLE 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: tskolda@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)).');
```

@tucker_tensor/full.m

Page 1/1

```
function A = full(B)
%TUCKER_TENSOR/FULL Convert to a dense tensor.
%
% A = FULL(B) converts Tucker tensor B to dense tensor A.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@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;
```

@tucker_tensor/ge.m

Page 1/1

```
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: tskolda@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)).');
```

@tucker_tensor/gt.m

Page 1/1

```
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: tskolda@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)).');
```

@tucker_tensor/issamesize.m

Page 1/1

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

@tucker_tensor/ldivide.m

Page 1/1

```
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: tskolda@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 ldivide(full(A),full(B)).');
```

@tucker_tensor/le.m

Page 1/1

```
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: tskolda@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 le(full(A),full(B)).');
```

@tucker_tensor/lt.m

Page 1/1

```
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: tskolda@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 lt(full(A),full(B)).');
```

@tucker_tensor/minus.m

Page 1/1

```
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: tkgolda@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 minus(full(A),full(B)).');
```

@tucker_tensor/mtimes.m

Page 1/1

```
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: tkgolda@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.

% 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
```

@tucker_tensor/ndims.m

Page 1/1

```
function n = ndims(t)
%TUCKER_TENSOR/NDIMS Return the number of dimensions.
%
% NDIMS(T) returns the number of dimensions of tensor T.
%
% See also ORDER.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tkgolda@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.

n = order(t);
```

@tucker_tensor/norm.m

Page 1/1

```
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: tkgolda@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 norm(full(A)).');
```

@tucker_tensor/not.m

Page 1/1

```
function B = not(A)
%TUCKER_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,
%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 not(full(A)).');
```

@tucker_tensor/or.m

Page 1/1

```
function C = or(A,B)
%TUCKER_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,
%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 or(full(A),full(B)).');
```

@tucker_tensor/order.m

Page 1/1

```
function n = order(t)
%TUCKER_TENSOR/ORDER Return the number of dimensions
%
% ORDER(T) returns the number of dimensions of tensor T.
%
% 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
%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.

n = length(t,u);
```

@tucker_tensor/permute.m

Page 1/1

```
function b = permute(a,order)
%TUCKER_TENSOR/PERMUTE Permute dimensions.
%
% B = PERMUTE(A,ORDER) rearranges the dimensions of A so that they
% are in the order specified by the vector ORDER. The tensor
% produced has the same values of A 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 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.

lambda = permute(a.lambda,order);

for i = 1 : length(order)
    u{i} = a.u{order(i)};
end

b = tucker_tensor(lambda, u);
```

@tucker_tensor/plus.m

Page 1/1

```
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: tskolda@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)).');
```

@tucker_tensor/power.m

Page 1/1

```
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: tskolda@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)).');
```

@tucker_tensor/rdivide.m

Page 1/1

```
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: tskolda@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)).');
```

@tucker_tensor/size.m

Page 1/1

```
function m = size(t,idx)
%TUCKER_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: tskolda@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
```

@tucker_tensor/subsasgn.m

Page 1/1

```
function t = subsasgn(t,s,b)
%TUCKER_TENSOR/SUBASGN Subscripted reference for tensor.
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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 = tucker_tensor(b, t.u);
    otherwise
        error(['Cannot change field ', s.subs, ' directly.']);
    end
case '()'
    error('Cannot change individual entries in CP tensor.')
```

@tucker_tensor/subsref.m

Page 1/1

```
function a = subsref(t,s)
%TUCKER_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: tskolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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 '()'
    error('Subsref with () not supported for Tucker tensor.')
```

@tucker_tensor/times.m

Page 1/1

```
function C = times(A,B)
%TUCKER_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: tskolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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)).');
```

@tucker_tensor/tucker_tensor.m

Page 1/2

```
function t = tucker_tensor(varargin)
%TUCKER_TENSOR Tensor stored in Tucker form.
%
% TUCKER_TENSOR(T) creates a TUCKER tensor by copying an existing
% TUCKER tensor.
%
% TUCKER_TENSOR(lambda,U1,U2,...,UM) creates a TUCKER tensor from
% its constituent parts. Here lambda is a dense tensor of size
% K1 x K2 x ... x KM and each Um is a matrix with Km columns.
%
% TUCKER_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 CP_TENSOR
%
%Brett W. Bader and Tamara G. Kolda, Released under SAND2004-3360,
%Sandia National Laboratories, 2004. Please address questions or
%comments to: tskolda@sandia.gov. Terms of use: You are free to copy,
%istribute, 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.

% Copy CONSTRUCTOR
if (nargin == 1) & isa(varargin{1}, 'tucker_tensor')
    t.lambda = varargin{1}.lambda;
    t.u = varargin{1}.u;
    t = class(t, 'tucker_tensor');
    return;
end

t.lambda = varargin{1};
if ~isa(t.lambda, 'tensor')
    error('LAMBDA must be a tensor.');
```

@tucker_tensor/tucker_tensor.m Page 2/2

```

    ' but there are ', int2str(length(t.u)), ' matrices.']);
end
for i = 1 : length(t.u)
    if size(t.u{i},2) ~= k(i)
        error(['Matrix U' int2str(i) ' does not have ' int2str(k(i)) 'columns.'])
    ];
end
end
t = class(t, 'tucker_tensor');
return;

```

@tucker_tensor/uminus.m Page 1/1

```

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: tskolda@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;

```

@tucker_tensor/uplus.m Page 1/1

```

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: tskolda@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!

```

@tucker_tensor/xor.m Page 1/1

```

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: tskolda@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 xor(full(A),full(B)).');

```