Preliminary Versions of the MATLAB Tensor Classes for Fast Algorithm Prototyping

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.
Preliminary Versions of the MATLAB Tensor Classes for Fast Algorithm Prototyping

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.
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, 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
%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);

function disp(t,name)
%TENSOR/DISP Command window display of a tensor.
%
%Brett W. Bader and Tamara G. Kolda, 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
%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 = '';
else
    namedot = [name ' . '];
end

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

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

fprintf(1,'%s of size ',namedot);
printsize(t.size);
fprintf(1,'
');

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

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

function printsize(sz)
for i = 1 : length(sz) − 1
    fprintf(1,'%d x ',sz(i));
end
fprintf(1,'%d', sz(length(sz)));
Function $C = \ge(A,B)$

\% $A \ge B$ 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 $'A \ge B'$ when $A$ or $B$ is a
\% tensor.

function $C = GE(A,B)$

\% 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(@ge,A,B);

function $C = gt(A,B)$

\% $A > B$ 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.

function $C = GT(A,B)$

\% 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(@gt,A,B);

function $b = issamesize(A,B)$

\% ISSAMESIZE returns true if tensors $A$ and $B$ are the same size.
\% Brett W. Bader and Tamara G. Kolda, 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.

m = 1 : M
for %
    M = order(A);
    C.lambda = [B.lambda; B.lambda];
    end
end

function $C = ldivide(A,B)$

\% TERMS/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 $'A \A B'$ when $A$ or $B$ is
\% a tensor.

function $C = LDIVIDE(A,B)$

\% 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);

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

2/28 Thursday July 08, 2004
if strcmp(version,'DDV')
    if isa(version,'char')
        version = 3;
    end
    if ~exist('version','var')
        error('Invalid index');
    end
    if (idx > order(T))
        % level of nifty-ness. Option 4 is Kiers.
        % others the license terms of this work. (5) Any of these conditions
        % can be waived if you get permission from the authors.
        error('Invalid index');
    end
    elseif strcmp(version,'Kiers')
        A = matricize_kiers(T,idx);
    elseif strcmp(version,'DDV')
        A = matricize_version3(T,idx);
    elseif (version == 3)
        A = matricize_version2(T,idx);
    elseif (version == 2)
        A = matricize_version1(T,idx);
    end
    else
        A = reshape(A,m,n);
        n = prod(I)/m;
    end
    A = shiftdim(A, idx − 1);
    A = double(T);
    end
error('Invalid index');
else
    elseif strcmp(version,'char')
        version = 4;
    version = 3;
    end
    elseif strcmp(version,'Kiers')
        A = matricize_kiers(T,idx);
    elseif strcmp(version,'DDV')
        A = matricize_version3(T,idx);
    elseif (version == 3)
        A = matricize_version2(T,idx);
    elseif (version == 2)
        A = matricize_version1(T,idx);
    else
        error('Invalid version');
    end
    if (idx > order(T))
        % level of nifty-ness. Option 4 is Kiers.
        % others the license terms of this work. (5) Any of these conditions
        % can be waived if you get permission from the authors.
        error('Invalid index');
    end
    elseif strcmp(version,'Kiers')
        A = matricize_kiers(T,idx);
    elseif strcmp(version,'DDV')
        A = matricize_version3(T,idx);
    elseif (version == 3)
        A = matricize_version2(T,idx);
    elseif (version == 2)
        A = matricize_version1(T,idx);
    else
        error('Invalid index');
    end
    end
A = tensor(A);
B = tensor(B);
if ~ issame(size(A,B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )
    error('Tensor size mismatch');
end
C = multiarrayop(@(a,b)A <= B);
C = multiarrayop(@(a,b)A < B);

% 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.
% Brett W. Bader and Tamara G. Kolda, 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.
% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories,
% 2004. Please address questions or comments to: tkgolda@sandia.gov.
function A = matricize_version1(T,idx)
    %TENSOR/MATRICIZE_VERSION1
    %
    % See MATRICIZE: This version uses the exact formula from Definition 1
    % in L. De Lathauwer, B. De Moor and J. Vandewalle, SIMAX
    % 21(4):1253-1278.
    % 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 (idx > order(T))
        error('Invalid index');
    end

    % Method 1
    I = size(T);
    M = ndims(T);
    m = I(idx);
    n = prod(I) / I(idx);
    A = zeros(m,n);
    for imult = 1 : prod(I)
        tmpi = imult - 1;
        for m = 1 : M - 1
            tmpdiv = prod(I(m+1:M));
            i(m) = floor (tmpi / tmpdiv) + 1;
            tmpi = tmpi - (i(m) - 1) * tmpdiv;
        end
        i(M) = tmpi + 1;
        newi = i(idx);
        newj = 1;
        if (idx == 1)
            for m = idx + 1 : M - 1
                newj = newj + (i(m) - 1) * prod(I([m+1:M,1:idx-1]));
            end
            newj = newj + (i(M) - 1);
        else
            for m = idx + 1 : M
                newj = newj + (i(m) - 1) * prod(I([m+1:M,1:idx-1]));
            end
            for m = 1 : idx - 2
                newj = newj + (i(m) - 1) * prod(I(m+1:idx-1));
            end
            newj = newj + (i(idx-1) - 1);
        end
    end

    for imult = 1 : prod(I)
        tmpi = imult - 1;
        for m = 1 : M - 1
            tmpdiv = prod(I(m+1:M));
            i(m) = floor (tmpi / tmpdiv) + 1;
            tmpi = tmpi - (i(m) - 1) * tmpdiv;
        end
        i(M) = tmpi + 1;
        newi = i(idx);
        newj = 1;
        for m = 2 : M - 1
            newj = newj + (i(m) - 1) * prod(I([m+1:M,1:idx-1]));
        end
        newj = newj + (i(M) - 1);
        for m = 1 : M
            if m == 1
                idxstr = int2str(i(1));
            else
                idxstr = [idxstr ',', int2str(i(m))];
            end
            A(newi, newj) = eval(['T.data(', idxstr, ')']);
        end
    end
end

function A = matricize_version2(T,idx)
    %TENSOR/MATRICIZE_VERSION2
    %
    % See MATRICIZE: This version first reorders the tensor.
    % 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 (idx > order(T))
        error('Invalid index');
    end

    % Method 2
    T.data = shiftdim(T.data, idx - 1);
    I = size(T);
    M = ndims(T);
    m = I(1);
    n = prod(I(2:M));
    A = zeros(m,n);
    for imult = 1 : prod(I)
        tmpi = imult - 1;
        for m = 1 : M - 1
            tmpdiv = prod(I(m+1:M));
            i(m) = floor (tmpi / tmpdiv) + 1;
            tmpi = tmpi - (i(m) - 1) * tmpdiv;
        end
        i(M) = tmpi + 1;
        newi = i(1);
        newj = 1;
        for m = 2 : M - 1
            newj = newj + (i(m) - 1) * prod(I([m+1:M,1:idx-1]));
        end
        newj = newj + (i(M) - 1);
        for m = 1 : M
            if m == 1
                idxstr = int2str(i(1));
            else
                idxstr = [idxstr ',', int2str(i(m))];
            end
            A(newi, newj) = eval(['T.data(', idxstr, ')']);
        end
    end
end

function A = matricize_version3(T,idx)
    %TENSOR/MATRICIZE_VERSION3
    %
    % See MATRICIZE: This version is the simplest.
    % 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 (idx > order(T))
        error('Invalid index');
    end

    if (order(T) == 1)
        return;
    end
    A = double(T);
    I = size(T);
    M = ndims(A);
    A = shiftdim(A, idx - 1);
    A = permute(A, [1,M:−1:2]);
    m = I(1);
    n = prod(I)/m;
    A = reshape(A,m,n);
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, 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 for commercial purposes. (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);

function C = mtimes(A,B)
% TENSOR/MTIMES Implement A*B for tensors.
% MTIMES(A,B) is the product of A and B. Any scalar 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, 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(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
if prod(size(A)) == 1
    sz = size(B);
else
    sz = size(A);
end
C = feval('mtimes', A.data, B.data);

function n = ndims(t)
% TENSOR/NDIMS Return the number of dimensions
% NDIMS(T) returns the number of dimensions of tensor T.

Brett W. Bader and Tamara G. Kolda, 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 = order(t);
function n = norm(T)
% Frobenius norm of a tensor.
% NORM(T) returns the Frobenius norm of a tensor.
% Brett W. Bader and Tamara G. Kolda, 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));
end

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, 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));
end

function C = or(A,B)
% TENSOR/OR Logical OR.
% A | B is a matrix 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, 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);
end

function n = order(t)
% TENSOR/ORDER Return the number of dimensions
% ORDER(T) returns the number of dimensions of tensor T.
% Brett W. Bader and Tamara G. Kolda, 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);
end
C = multiarrayop(@power, A, B);

error('Tensor size mismatch.')

~( issamesize(A, B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )

if

B = tensor(B);
A = tensor(A);

% can be waived if you get permission from the authors.

if

A = tensor(A); B = tensor(B);

if | issamesize(A, B) | (prod(size(A)) == 1) | (prod(size(B)) == 1) )

error('Tensor size mismatch.')

else

error('Invalid Order');
return;

end

T.data = permute(T.data, Idx);
T.size = T.size(Idx);

end

T = permute(T, Idx);

function

C = power(A, B)

TENSOR/POWER

Powers of tensors.

Z = X.^Y denotes element-by-element powers. X and Y
must have the same dimensions unless one is a scalar.
A scalar can be operated into anything.

C = POWER(A, B) is called for the syntax 'A .^ B' when A or B is a
scalar.

B. W. Bader and Tamara G. Kolda, 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 length(Idx) == 1
if | Idx == 1 |
else
error('Invalid Order');
end

end

T.data = permute(T.data, Idx);
T.size = T.size(Idx);

end

T = permute(T, Idx);

function

C = product(varargin)

TENSOR/PRODUCT

Tensor Multiplication.

PRODUCT(A, B) computes the scalar product of tensors A and B; A and
B must have the same size.

PRODUCT(A, B, ADIMS, BDIMS) computes the product of tensors A and B
in dimensions specified by the row vectors ADIMS and BDIMS.
The result is a tensor of size equal to [size(A) size(B)] minus
the respective dimensions in ADIMS and BDIMS.

C = PRODUCT(A, B, N) computes the product of tensor A with a
vector B; i.e., A x_N B. The integer N specifies the dimension in A along
which B should be multiplied. If size(B) == [J,1], then A must have
size(A,N) == J. The result will be of order one less than A
because the N-th dimension removed. Note that the flag 'vec'
must be specified to indicate that B is an N-vector.

PRODUCT(A, U) computes the product of a tensor A and a cell
array U; i.e., A x_1 U(1) x_2 U(2) ... x_N U(N). If the tensor A
is of size I1 x I2 x ... x IN, then the n-th cell of U is a
matrix of size Jn x In. The result is a tensor of size J1 x J2
x ... x JN.

PRODUCT(A, U, 'vec') computes the product of a tensor A and a cell
array U; i.e., A x_1 U(1) x_2 U(2) ... x_N U(N). If the tensor A
is of size I1 x I2 x ... x IN, then the n-th cell of U is a vector
of size In x 1. The result is a tensor of order 1 and size 1. Note
that the flag 'vec' must be specified when the order of the result
is to be reduced.

PRODUCT(A, U, DIMS) computes the product of a tensor A and a
cell array U along the dimensions specified in DIMS.

Case 1: if DIMS contains positive entries, the i-th cell in array
U is multiplied by the dimension specified by DIMS(i). In this
case, it is assumed that length(U) == length(DIMS).

Example 1: B = product(A, [X Y, '3 4']) computes B = A x_3 X x_4.
T. Here A is a cell array of order at least 4, and X and Y are
appropriately sized matrices.

Example 2: B = product(A, U, -3) computes B = A x_3 U(1) x_2 U(2)
x_4 U(4). Here A is a 4-th order tensor, and U is a cell array with
4 entries.

PRODUCT(A, U, DIMS, 'vec') computes the product of a tensor A and a
cell array U along the specified dimensions. In other words, the
i-th cell in array U is multiplied by the dimension specified by

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for

for
if prod(size(Adims)) ~= 1
if (casetype == 2) | (casetype == 3)
end
A = varargin{1};
if (casetype == 3) & (length(sizeB) ~= 2)
if (casetype == 2) & ((length(sizeB) ~= 2) | (sizeB(2) ~= 1))
if isa(varargin{2},'cell')
end
end
B = varargin{2};
C = varargin{3};

% Case 1 : Inner Product

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

% Cell Array Multiplication

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

Bdims = 2;

% Scalar Inner Product

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

end
end

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

for i = 1 : length(Adims)
if (ismember(Adims(i), 1:M)) == 0
error('M must be a scalar');
end
C = tensor(C, [Cdims(i)]);
end
C = reshape(C, Cdims);
end


% Check validity of parameters passed to product

N = length(A);  
if (N > order(A)) | (N > length(U))
error('DIMS is too long.');
elseif (N < length(U)) & (length(U) < order(A))
end

% Determine str argument to be passed to next product call

str = 'mat';
if (strcmp(varargin(nargin),'vec'))
str = 'vec';
elseif (isa(varargin(nargin), 'char')) & ...
error('ADIMS and BDIMS are not the same size');
end
else

% Case 3 : x_m Multiplication with Matrix


end

elseif

% More Complex Cases

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

C = tensor(C, Cdims);
sizeB = size(B);

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

C = permute(C, Cdims_New);
Cdims_New = [1:n−1, N, n:N−1];

n = prod(sizeA);
if
elseif

% In this case, we can assume that Bdims = 1 and so B_Dims_New =
Adims = [Adims(2), Bdims];
Cdims = [Adims(2), Bdims];

end


% Process A so that the dimensions specified by Adims are first
% in A_Dims corresponding to the rows of the matrix, and the
% remainder corresponding to the columns.

N = ndims(A);
I = size(A);
if

MATLAB Tensor Classes by B. W. Bader and T. G. Kolda
Function C = rdivide(A,B)

- 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.

@tensor/rdivide.m

function B = shiftdim(varargin)

- B = SHIFTDIM(X,N) shifts the dimensions of 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, 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};
B = varargin{2};

if ~[isempty(size(A)) | (prod(size(A)) == 1) | (prod(size(B)) == 1)]
  error('Tensor size mismatch.')
end

B = tensor(B, size(B));

The following functionality has not been implemented yet

B = tensor(squeeze(A.data));

B = tensor(shiftdim(A.data, n));

end

Function B = size(t,idx)

- TENSOR/SIZE Size of tensor.
- T = size(T,DIM) returns the size of the dimension specified by DIM.
- T = size(T,DIM) returns the size of the dimension specified by DIM.
- See also ORDER, NDIMS.

Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.

Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

m = t.size idx;

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

Function B = squeeze(A)

- TENSOR/SIZE Size of tensor.
- B = SQUEEZE(A) returns a tensor B with the same elements as A but with all the singleton dimensions removed. A singleton dimension is a dimension such that size(A,dim)==1. 2-D tensors are unaffected by squeeze so that row vectors remain rows.

For example,

squeezes( tensor(rand(2,1,3)) )

is a 2-by-3 tensor.

Brett W. Bader and Tamara G. Kolda, 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 = tensor(squeeze(A.data));
% Support Functions

function [t] = tensorize_ddv(A, i, dims)
    % TENSOR/DDV
    if prod(dims) ~= prod(size(A))
        error('Invalid dimensions');
    end
    M = length(dims);
    indx = circshift([1:M], [-0 i-1]);
    indx2 = [indx(1) indx(M:1-i:2)];
    T = reshape(A, dims(indx2));
    T = permute(T, indx2);
    t.data = T;
    t.size = dims;
end

function t = tensorize_kiers(A, i, dims)
    % TENSOR/KIERS
    if prod(dims) ~= prod(size(A))
        error('Invalid dimensions');
    end
    M = length(dims);
    indx = circshift([1:M], [-0 i-1]);
    indx = [indx(1) indx(M:1-i:2)];
    T = reshape(A, dims(indx));
    T = shiftdim(T, mod(M-i+1, M));
    t.data = T;
    t.size = dims;
end

function t = uminus(t)
    % TENSOR/UMINUS
    t.data = -t.data;
end

function t = uplus(t)
    % TENSOR/UPLUS
    % This function does nothing!
end

function C = times(A, B)
    % TENSOR/TIMES
    % 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.

    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.');
    end
    C = A.data .* B.data;
    return;
end

% Brett W. Bader and Tamara G. Kolda, 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.');
end
C = A.data .* B.data;
C = tensor(C, size(A));
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, 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 ~( issame size(A,B) | prod(size(A)) == 1 | prod(size(B)) == 1 )
   error('Tensor size mismatch.' )
end
C = multiarrayop(@xor,A,B);

function C = and(A,B)
%CP_TENSOR/AND Logical AND.
%   AND(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, 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 and(full(A),full(B));

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, 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.
% 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 isvarargin(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('Array 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('Array U' int2str(i) ' does not have int2str(k) columns');
    end
end
function disp(t)
%CP_TENSOR/DISP Command window display.

%Brett W. Bader and Tamara G. Kolda, 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.

fprintf(1,'
');
fprintf(1,'%s is a CP tensor of size ', inputname(1));
printsize(size(t));
fprintf(1,'
');
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)));
Function $C = \text{ge}(A,B)$

CP_TENSOR/GE Greater than or equal.

Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.

Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.

error('Use ge(full(A),full(B))');

function $C = \text{gt}(A,B)$

CP_TENSOR/GT Greater than.

Brett W. Bader and Tamara G. Kolda, 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))');

Function $b = \text{issamesize}(A,B)$

CP_TENSOR/ISSAMESIZE

ISSAMESIZE(A,B) returns true if tensors A and B are the same size.

Brett W. Bader and Tamara G. Kolda, 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 = true;
else
    b = false;
end

function $b = \text{ldivide}(A,B)$

CP_TENSOR/LDIVIDE Left array divide.

Brett W. Bader and Tamara G. Kolda, 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 and(ldivide(A),full(B))');
Function C = le(A,B)
CP_TENSOR/LE Less than or equal.

Brett W. Bader and Tamara G. Kolda, 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 and(le(A),full(B))');

Function C = lt(A,B)
CP_TENSOR/LT Less than.

Brett W. Bader and Tamara G. Kolda, 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 and(lt(A),full(B))');

Function A = matricize(T,idx) version
CP_TENSOR/MATRICIZE

Brett W. Bader and Tamara G. Kolda, 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 matricize(full(A),idx,type)');

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, 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.')
    end
    lambda = [A.lambda; −B.lambda];
    M = order(A);
    for n = 1 : M
        u(n) = [A.u(n) B.u(n)];
    end
    C = cp_tensor(lambda,u);
    return;
end
error('Use minus(full(A),full(B))');
function C = mtimes(A,B)
%CP_TENSOR/MTIMES Implement A*B.
%Brett W. Bader and Tamara G. Kolda, 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.

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

function n = ndims(t)
%CP_TENSOR/NDIMS Return the number of dimensions
%Brett W. Bader and Tamara G. Kolda, 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 = order(t);

function B = not(A)
%CP_TENSOR/NOT Logical NOT.
%Brett W. Bader and Tamara G. Kolda, 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))');

function n = norm(T)
%CP_TENSOR/NORM Frobenius norm.
%Brett W. Bader and Tamara G. Kolda, 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 norm(full(A))');
function C = or(A,B)
%CP_TENSOR/OR Logical OR.
% Brett W. Bader and Tamara G. Kolda, 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))');

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, 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);

function n = order(t)
%CP_TENSOR/ORDER Return the number of dimensions
% Brett W. Bader and Tamara G. Kolda, 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);

function C = plus(A,B)
%CP_TENSOR/PLUS Binary addition.
% Brett W. Bader and Tamara G. Kolda, 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.");
end
lambda = [A.lambda; B.lambda];
M = order(A);
for m = 1 : M
u{m} = [A.u{m} B.u{m}];
end
C = cp_tensor(lambda, u);
return;
end
error('Use plus(full(A),full(B))');
function C = power(A,B)
%
% TENSOR/POWER

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

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

function C = product(varargin)
%
% CP_TENSOR/PRODUCT Tensor Multiplication.

% Brett W. Bader and Tamara G. Kolda, 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 product(full(C),...)');

function C = rdivide(A,B)
%
% TENSOR/RDIVIDE Right array divide.

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

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

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

if exist('idx','var')
    m = size(t.u(idx), 1);
else
    for i = 1 : order(t)
        m(i) = size(t.u{i}, 1);
    end
end
Function t = subsasgn(t,s,b)
%CP_TENSOR/SUBASGN Subscripted reference.
%Brett W. Bader and Tamara G. Kolda, 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
switch s.type
  case '.'
    switch s.subs
      case 'lambda'
        a = t.lambda;
      case 'u'
        a = t.u;
      otherwise
        error(['No such field: ', s.subs]);
    end
  case '()'
    a = 0;
    for k = 1 : length(t.lambda)
      b = 1;
      for i = 1 : length(s.subs)
        b = b * t.u{i}(s.subs{i},k);
      end
      a  = a + t.lambda(k) * b;
    end
  case '{}'
    a = t.u{s.subs{:}};
  otherwise
    error('Invalid subsref');
end
function C = times(A,B)
%CP_TENSOR/TIMES Element-wise multiplication.
%Brett W. Bader and Tamara G. Kolda, 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. (5) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.
error('Use times(full(A),full(B))');
function a = subsref(t,s)
%CP_TENSOR/SUBSREF Subscripted reference.
%Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.
%Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.
switch s.type
  case '.'
    switch s.subs
      case 'lambda'
        a = t.lambda;
      case 'u'
        a = t.u;
      otherwise
        error(['No such field: ', s.subs]);
    end
  case '()'
    a = 0;
    for k = 1 : length(t.lambda)
      b = 1;
      for i = 1 : length(s.subs)
        b = b * t.u{i}(s.subs{i},k);
      end
      a  = a + t.lambda(k) * b;
    end
  case '{}'
    a = t.u{s.subs{:}};
  otherwise
    error('Invalid subsref');
end
function t = uminus(t)
%CP_TENSOR/UMINUS Unary minus for tensors.
%Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.
%Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions. (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.
t.lambda = -t.lambda;
Function \( t = \text{uplus}(t) \)

\text{UPPLUS Unary plus for tensors.}

\text{For tensors.}

\text{Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.}

\text{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.}

\text{This function does nothing!}

\function{C = \text{xor}(A,B)}

\text{XOR Logical EXCLUSIVE OR.}

\text{Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.}

\text{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))};

\function{C = \text{and}(A,B)}

\text{AND Logical AND.}

\text{Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.}

\text{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))};

\function{\text{disp}(t)}

\text{Disp Command window display.}

\text{Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 2004. Please address questions or comments to: tgkolda@sandia.gov.}

\text{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 disp(full(t))};
function display(t)
    % TUCKER_TENSOR/DISPLAY Command window display.
    Brett W. Bader and Tamara G. Kolda, 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.
    fprintf(1,'
');
    fprintf(1,'%s is a Tucker tensor of size ', inputname(1));
    printsize(size(t));
    fprintf(1,'
');
    disp(' ');
    disp([inputname(1), '.lambda = ']);
    disp(t.lambda);
    for j = 1 : order(t)
        disp([inputname(1), '.U{', int2str(j), '} = ']);
        disp(t.u{j});
    end
    fprintf(1,'
');
end

function disp(t)
    % TUCKER_TENSOR/DISPLAY Command window display.
    Brett W. Bader and Tamara G. Kolda, 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.
    fprintf(1,'
');
    fprintf(1,'%s is a Tucker tensor of size ', inputname(1));
    printsize(size(t));
    fprintf(1,'
');
    disp(' ');
    disp([inputname(1), '.lambda = ']);
    disp(t.lambda);
    for j = 1 : order(t)
        disp([inputname(1), '.U{', int2str(j), '} = ']);
        disp(t.u{j});
    end
    fprintf(1,'
');
end

function printsize(sz)
    for i = 1 : length(sz) − 1
        fprintf(1, '%d x ', sz(i));
    end
    fprintf(1, '%d', sz(length(sz)));
end

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

function t = full(t)
    % TUCKER_TENSOR/FULL Convert to a dense tensor.
    % Brett W. Bader and Tamara G. Kolda, 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('version','var')
        version = 1;
    end
    if version == 1
        M = order(t);
        tmp = product(t.lambda, t.u{1}, 1);
        for n = 2 : M
            tmp = product(tmp, t.u{n}, n);
        end
        t = tensor(tmp, size(t));
    else
        K = size(t.lambda);
        M = order(t);
        I = size(t);
        % Loop through all combinations of indices (using one loop % instead of M)
        for kmult = 1 : prod(K)
            % Extract indices
            tmpk = kmult − 1;
            for m = 1 : M − 1
                tmp = prod([tmp, m+1:M]);
            end
            k(m) = floor (tmpk / tmp) + 1;
            tmpk = tmpk − (k(m) − 1) * tmp;
            if kmult == 1
                idstr = int2str(k(1));
            else
                idstr = [idstr , ',', int2str(k(m))];
            end
        end
        % Add in rank-1 matrix corresponding to % lambda(k1,k2,...,KM)
        tmp = 1;
        for n = 1 : M
            tmp = tmp * t.u{n}(k(m));
        end
        tmp = reshape(tmp, prod([1:M]), 1);
        tmplambda = eval(tmpstr);
        if kmult == 1
            a = tmplambda * tmp;
        else
            a = a + tmplambda * tmp;
        end
    end
    t = tensor(a, size(t));
end
function C = ge(A,B)
%TUCKER_TENSOR/GE Greater than or equal.
%Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories,
%2004. Please address questions or comments to: tgkolda@sandia.gov.
%Terms of use: You are free to copy, distribute, display, and use this
%work, under the following conditions. (1) You must give the original
%authors credit. (2) You may not use or redistribute this work for
%commercial purposes. (3) You may not alter, transform, or build upon
%this work. (4) For any reuse or distribution, you must make clear to
%others the license terms of this work. (5) Any of these conditions
%can be waived if you get permission from the authors.
error('Use ge(full(A),full(B))');

function C = gt(A,B)
%TUCKER_TENSOR/GT Greater than.
%Brett W. Bader and Tamara G. Kolda, 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))');

function b = issamesize(A,B)
%TUCKER_TENSOR/ISSAMESIZE
%   ISSAMESIZE(A,B) returns true if A and B are the same size.
%Brett W. Bader and Tamara G. Kolda, 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 = true;
else
  b = false;
end

function C = ldivide(A,B)
%TUCKER_TENSOR/LDIVIDE Left array divide.
%Brett W. Bader and Tamara G. Kolda, 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 ldivide(A,B)');
Function C = le(A,B)
%TUCKER_TENSOR/LE Less than or equal.
%Brett W. Bader and Tamara G. Kolda, 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 and(le(A),full(B))');

MATLAB Tensor Classes by B. W. Bader and T. G. Kolda
Thursday July 08, 2004 23/28

Function C = lt(A,B)
%TUCKER_TENSOR/LT Less than.
%Brett W. Bader and Tamara G. Kolda, 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 and(lt(A),full(B))');

Function A = matricize(T,idx,version)
%TUCKER_TENSOR/MATRICIZE Convert tensor to a matrix.
%Brett W. Bader and Tamara G. Kolda, 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 matricize(full(A),idx,type)');

Function C = minus(A,B)
%TUCKER_TENSOR/MINUS Binary subtraction.
%Brett W. Bader and Tamara G. Kolda, 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 minus(full(A),full(B))');
Function C = mtimes(A,B)
TUCKER_TENSOR/MTIMES Implement A*B.

Brett W. Bader and Tamara G. Kolda, 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.

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

Function n = ndims(t)
TUCKER_TENSOR/NDIMS Return the number of dimensions.

Brett W. Bader and Tamara G. Kolda, 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 = order(t);

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

Brett W. Bader and Tamara G. Kolda, 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))');

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

Brett W. Bader and Tamara G. Kolda, 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 norm(full(A))');
Function C = or(A,B)

TUCKER_TENSOR/OR Logical OR.

% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 12/2004. 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))');

Function n = order(t)

TUCKER_TENSOR/ORDER Return the number of dimensions

% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 12/2004. 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);

Function b = permute(a,order)

TUCKER_TENSOR/PERMUTE Permute dimensions.

% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 12/2004. 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);

Function C = plus(A,B)

TUCKER_TENSOR/PLUS Binary addition.

% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories, 12/2004. Please address questions or comments to: tgkolda@sandia.gov.
% Terms of use: You are free to copy, distribute, display, and use this work, under the following conditions: (1) You must give the original authors credit. (2) You may not use or redistribute this work for commercial purposes. (3) You may not alter, transform, or build upon this work. (4) For any reuse or distribution, you must make clear to others the license terms of this work. (5) Any of these conditions can be waived if you get permission from the authors.
error('Use plus(full(A),full(B))');
Function

C = power(A,B)

TUCKER_TENSOR/POWER

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

function

C = product(varargin)

TUCKER_TENSOR/PRODUCT Tensor Multiplication.

% Brett W. Bader and Tamara G. Kolda, 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 product(full(C),...)');

function

m = size(t,idx)

TUCKER_TENSOR/SIZE Size of tensor.

% Brett W. Bader and Tamara G. Kolda, Sandia National Laboratories,
% 2004. Please address questions or comments to: tgkolda@sandia.gov.
% Terms of use: You are free to copy, distribute, display, and use this
% work, under the following conditions. (1) You must give the original
% authors credit. (2) You may not use or redistribute this work for
% commercial purposes. (3) You may not alter, transform, or build upon
% this work. (4) For any reuse or distribution, you must make clear to
% others the license terms of this work. (5) Any of these conditions
% can be waived if you get permission from the authors.
% if exist('idx','var')
% m = size(t.u(idx), 1);
% else
% for i = 1 : order(t)
% m(i) = size(t.u{i}, 1);
% end
% end
function t = subsasgn(t,s,b)
%TUCKER_TENSOR/SUBASGN Subscripted assignment for tensor.
% Brett W. Bader and Tamara G. Kolda, 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 'lambda'
        t = tucker_tensor(t.lambda, u);
    otherwise
        error('TUCKER_TENSOR assignment with non-lambda subsref does not work');
    end

switch s.subs
    case 't.u'
        u{s.subs{:}} = b;
    case 't.lambda'
        t = tucker_tensor(t.lambda, u);
    otherwise
        error('Invalid subscript assignment');
end

function a = subsref(t,s)
%TUCKER_TENSOR/SUBSREF Subscripted reference.
% Brett W. Bader and Tamara G. Kolda, 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 'lambda'
        a = t.lambda;
    case 't.u'
        a = t.u;
    otherwise
        error('No such field: ', s.subs);
end

switch s.subs
    case '()'
        error('Subsref with () not supported for Tucker tensor');
    case '{}'
        a = 1:u{s.subs{:}};
    otherwise
        error('Invalid subsref');
end

function C = times(A,B)
%TUCKER_TENSOR/TIMES Element-wise multiplication.
% Brett W. Bader and Tamara G. Kolda, 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))']);
for \( i = 1 : \text{length}(t.u) \)
    if \( \text{size}(t.u(i),2) \neq k(i) \)
        error(['Matrix U' int2str(i) ' does not have ' int2str(k(i)) 'columns'])
    end
end

\( t = \text{class}(t, 'tucker_tensor'); \)
return;

function \( t = \text{uminus}(t) \)
\%TUCKER_TENSOR/UMINUS Unary minus for tensors.
\%Brett W. Bader and Tamara G. Kolda, 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\( t.\lambda = -t.\lambda; \)

function \( t = \text{uplus}(t) \)
\%TUCKER_TENSOR/UPLUS Unary plus for tensors.
\%Brett W. Bader and Tamara G. Kolda, 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\% This function does nothing!

function \( C = \text{xor}(A,B) \)
\%TUCKER_TENSOR/XOR Logical EXCLUSIVE OR.
\%Brett W. Bader and Tamara G. Kolda, 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.
\nerror(['Use xor(full(A),full(B))']);
<table>
<thead>
<tr>
<th>Quantity</th>
<th>Location</th>
<th>Recipient</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MS 9519</td>
<td>Tammy Kolda</td>
<td>8962</td>
</tr>
<tr>
<td>1</td>
<td>MS 1110</td>
<td>Brett Bader</td>
<td>9233</td>
</tr>
<tr>
<td>3</td>
<td>MS 9018</td>
<td>Central Technical Files</td>
<td>8945-1</td>
</tr>
<tr>
<td>1</td>
<td>MS 0899</td>
<td>Technical Library</td>
<td>9616</td>
</tr>
<tr>
<td>1</td>
<td>MS 9021</td>
<td>Classified Office</td>
<td>8511/Technical Library, MS 0899, 9616</td>
</tr>
</tbody>
</table>

DOE/OSTI via URL