SANDIA REPORT

SAND2004-3360
Unlimited Release
Printed July 2004

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.

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

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 July 2004

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.

@tensor/and.m


```
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 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


```
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 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 size ',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)

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

@tensor/display.m


```
function display(t)
%TENSOR/DISPLAY 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 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


```
function a = double(t)
%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.


a = t.data;
```

@tensor/ge.m


```
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, 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(@ge,A,B);
```

@tensor/gt.m


```
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, 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(@gt,A,B);
```

@tensor/issamesize.m


```
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, 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
```

@tensor/ldivide.m


```
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, 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

% if (istensor(A,'CP') & istensor(B,'CP'))
%     C = tensor(A);
%     C.lambda = [B.lambda; B.lambda];
%     M = order(A);
%     for m = 1 : M
%         C.data{m} = [A.data{m} B.data{m}];
%     end
%     return;
% end

C = multiarrayop(@ldivide,A,B);
```

@tensor/le.m


```
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, 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


```
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, 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/matricize.m


```
function A = matricize(T,idx,version)
%TENSOR/MATRICIZE
%
%   MATRICIZE(T,I) or MATRICIZE(T,I,'DDV') matricizes (or "unfolds") a
%   tensor according to Definition 1 in L. De Lathauwer, B. De Moor
%   and J. Vandewalle, SIMAX 21(4):1253-1278.
%
%   MATRICIZE(T,I,'Kiers') matricizes (or "unfolds") a tensor
%   according to H.A.L. Kiers, J. Chemometrics 2000; 14"105-122.
%
%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 that the numerical version specification is essentially a
% "hidden" option. Options 1-3 are versions of DDV, in increasingly
% level of nifty-ness. Option 4 is Kiers.

if (idx > order(T))
  error('Invalid index');
end

if ~exist('version','var')
   version = 3;
end

if isa(version,'char')
   if strcmp(version,'DDV')
            version = 3;
   elseif strcmp(version,'Kiers')
            version = 4;
   else
            error('Invalid 3rd argument');
   end
end

if (version == 1)
  A = matricize_version1(T,idx);
elseif (version == 2)
  A = matricize_version2(T,idx);
elseif (version == 3)
  A = matricize_version3(T,idx);
elseif (version == 4)
  A = matricize_kiers(T,idx);
else
  error('Invalid version');
end
```

@tensor/matricize_kiers.m


```
function A = matricize_kiers(T,idx)
%TENSOR/MATRICIZE_KIERS
%
%  MATRICIZE(T,I) matricizes (or "unfolds") a Tensor according to
%  the definition in Kiers, J. Chemometrics, 2000(14):105-122.
%
%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 (idx > order(T))
  error('Invalid index');
end

A = double(T);
I = size(T);
M = ndims(T);

A = shiftdim(A, idx - 1);

m = I(idx);
n = prod(I)/m;

A = reshape(A,m,n);
```

@tensor/matricize_version1.m


```
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.
%
%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 (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
```

@tensor/matricize_version1.m


```
  for m = 1 : M
    if m == 1
      idxstr = int2str(i(1));
    else
      idxstr = [idxstr ',' int2str(i(m))];
    end
  end
  A(newi, newj) = eval(['T.data(', idxstr, ')']);

end
```

@tensor/matricize_version2.m


```
function A = matricize_version2(T,idx)
%TENSOR/MATRICIZE_VERSION2
%
%   See MATRICIZE: This version first reorders the 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 (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));
    end
    newj = newj + (i(M) - 1);

    for m = 1 : M
        if m == 1
            idxstr = int2str(i(1));
        else
            idxstr = [idxstr ',' int2str(i(m))];
        end
    end
    A(newi, newj) = eval(['T.data(', idxstr, ')']);

end
```

@tensor/matricize_version3.m


```
function A = matricize_version3(T,idx)
%TENSOR/MATRICIZE_VERSION3
%
%   See MATRICIZE: This version is the simplest.
%
%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 (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(idx);
n = prod(I)/m;

A = reshape(A,m,n);
```

@tensor/minus.m


```
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, 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


```
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, 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

dimA = size(A);
dimB = size(B);

if ( dimA(length(dimA)) ~= dimB(1) )
  error('tensor dimensions must agree.');
end

C = product(A,B,order(A),1);

%error('TENSOR/MTIMES not fully implemented');
```

@tensor/multiarrayop.m


```
function C = multiarrayop(fname,A,B)
%TENSOR/MULTIARRAYOP Generic functions for tensors.
%
%   MULTIARRAYOP(F,A,B) applies the 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, 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 prod(size(A)) == 1
    sz = size(B);
else
    sz = size(A);
end

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

@tensor/ndims.m


```
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);
```

@tensor/norm.m


```
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));
```

@tensor/not.m


```
function B = not(A)
%TENSOR/NOT Logical NOT.
%
%   ~A is a tensor whose elements are 1's where A has zero
%   elements, and 0's where A has non-zero elements.
%
%   B = NOT(A) is called for the syntax '~A' when A is a tensor.
%
%Brett W. Bader and Tamara G. Kolda, 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


```
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);
```

@tensor/order.m


```
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);
```

@tensor/permute.m


```
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 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.


if length(Idx) == 1

    if (Idx == 1)
        return;
    else
        error('Invalid Order');
    end

end

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

@tensor/plus.m


```
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 1-by-1 matrix).
%   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, 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(@plus,A,B);
```

@tensor/power.m


```
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, 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(@power,A,B);
```

@tensor/product.m


```
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 the 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 matrix
%   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,I], then A must have
%   size(A,M) = I. The result will be the same order and size as A
%   except that size(C,M) = J.
%
%   C = PRODUCT(A,B,N,'vec') computes the product of a tensor A with a
%   vector B; i.e., A x_N B. The integer M specifies the dimension in
%   A along which B should be multiplied. If size(B) = [I,1], then A
%   must have size(A,N) = I. 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 specified dimensions.
%
%   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
%   Y.  Here A is a cell array of order at least 4, and X and Y are
%   appropriately sized matrices.
%
%   Case 2: If DIMS containts negative entries, then we compute the
%   product of A and the cell array U except in the dimensions
%   specified in DIMS.
%
%   Example 2: B = product(A, U, -3) computes B = A x_1 U{1} x_2 U{2}
%   x_4 U{4}. Here A is a 4th-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
```

@tensor/product.m


```
%   DIMS(i). The result is a tensor of reduced order. Note that the
%   flag 'vec' must be specified to indicate that U contains
%   vectors rather than matrices.
%
%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 (nargin < 2)
    error('product requires at least two arguments');
end

if ~isa(varargin{1},'tensor')
    error('First argument must be a tensor.');
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Cell Array Multiplication
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if isa(varargin{2},'cell')

    A = varargin{1};
    U = varargin{2};

    if (nargin >= 3) & ~isa(varargin{3},'char')
        dims = varargin{3};
    else
        dims = [1 : order(A)];
    end

    % Determine str argument to be passed to next product call
    if (((nargin == 3) | (nargin == 4)) & ...
        (isa(varargin{nargin},'char')) & ...
        (strcmp(varargin(nargin),'vec')))
        str = 'vec';
    else
        str = 'mat';
    end

    % Check validity of DIMS and then check for "minus" case
    if (max(abs(dims)) > order(A))
        error('An entry in DIMS exceeds order of A.');
    elseif (max(dims) < 0)
        dims = setdiff([1:order(A)], -dims);
    end

    % Check validity of parameters passed to product
    N = length(dims);
    if (N > order(A)) | (N > length(U))
        error('DIMS is too long.');
    elseif (N < length(U)) & (length(U) < order(A))
        error('If length(DIMS) < length(U), then length(U) must equal order(A).'
```

@tensor/product.m


```
);
    elseif (length(U) > order(A))
        error('Length of U greater than order of A.');
    end

    % Reorder dims from least to greatest
    [sdims,sidx] = sort(dims);

    % Check sizes of U and DIMS to determine version to use.
    if (N == length(U))
        uidx = sidx;   % index U by sorted order
    else
        uidx = sdims;  % index U by (sorted) dimension
    end

    % Calculate individual products
    C = product(A, U{uidx(N)}, sdims(N), str);
    for n = (N - 1): -1 : 1
        C = product(C, U{uidx(n)}, sdims(n), str);
    end

    return;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Otherwise, the 1st two entries are both tensors
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

A = tensor(varargin{1});
B = tensor(varargin{2});
sizeA = size(A);
sizeB = size(B);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Scalar Inner Product
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if (nargin == 2)
    if ~issamesize(A,B)
        error('A and B are not the same size for product(A,B)');
    end
    n = prod(sizeA);
    A = reshape(A.data,n,1);
    B = reshape(B.data,n,1);
    C = sum(A.*B);
    return;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% More Complex Cases
% Case 1 : Inner Product
% Case 2 : x_m Multiplication with Vector
% Case 3 : x_m Multiplication with Matrix
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

if (nargin == 4) & ~isa(varargin{4},'char')
    casetype = 1;
    Adims = varargin{3};
    Bdims = varargin{4};
elseif (nargin == 4) & isa(varargin{4},'char') & strcmp(varargin{4},'vec')
```

@tensor/product.m


```
    casetype = 2;
    Adims = varargin{3};
    Bdims = 1;
else
    casetype = 3;
    Adims = varargin{3};
    Bdims = 2;
end

if (casetype == 2) & ((length(sizeB) ~= 2) | (sizeB(2) ~= 1))
    error('B is not a vector');
end

if (casetype == 3) & (length(sizeB) ~= 2)
    error('B is not a matrix');
end

if (casetype == 2) | (casetype == 3)
    if prod(size(Adims)) ~= 1
        error('M must be a scalar');
    end
end

if ~issamesize(tensor(Adims),tensor(Bdims))
    error('ADIMS and BDIMS are not the same size');
end

for i = 1 : length(Adims)
    if (sizeA(Adims(i)) ~= sizeB(Bdims(i)))
        error(['The ' int2str(i) '-th specified dimension does not match.']);
    end
end

[A_New, Adims_New] = product_preprocess(A, Adims);
[B_New, Bdims_New] = product_preprocess(B, Bdims);

C = A_New' * B_New;

Cdims = [Adims_New, Bdims_New];

if isempty(Cdims)
    Cdims = 1;
    MatDims = [1 1];
elseif length(Cdims) == 1
    MatDims = [Cdims 1];
else
    MatDims = Cdims;
end
C = reshape(C, MatDims);

if (casetype == 3)
    % Calculating A x_n B. The result produced thus far has the
    % in a different shape. We need to permute it.
    N = ndims(A);
    n = Adims;
    Cdims_New = [1:n-1, N, n:N-1];
    C = tensor(C, [Cdims 1]);
    C = permute(C, Cdims_New);
elseif (casetype == 2)
```

@tensor/product.m


```
    % In this case, we can assume that Bdims = 1 and so B_Dims_New =
    % 1. Want to drop the last dimension, so it's not included.
    C = tensor(C, Adims_New);
else
    C = tensor(C, Cdims);
end


% ------------------------------------------------------------
% -- Support Functions ---------------------------------------
% ------------------------------------------------------------

function [A_New, I_New] = product_preprocess(A, A_Dims)
% Process A so that the dimensions specified by A_Dims are first
% and the tensor A is reshaped into a matrix with those dimensions
% in A_Dims corresponding to the rows of the matrix, and the
% remainder corresponding to the columns.

M = ndims(A);
I = size(A);

% if sum(~ismember(A_Dims, [1:M])) > 0
%     error('Indices exceed number of dimensions in tensor');
% end

M_Comp = setdiff([1:M], A_Dims);
I_Match = I(A_Dims);
I_New = I(M_Comp);

A_Permuted = permute(A, [A_Dims, M_Comp]);
A_New = reshape(A_Permuted.data, prod(I_Match), prod(I_New));
```

@tensor/rdivide.m


```
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, 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(@rdivide,A,B);
```

@tensor/shiftdim.m


```
function B = shiftdim(varargin)
%SHIFTDIM Shift dimensions.
%
%   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};
n = varargin{2};

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


% The following functionality has not been implemented yet

%   [B,NSHIFTS] = SHIFTDIM(X) returns the array B with the same
%   number of elements as X but with any leading singleton
%   dimensions removed. NSHIFTS returns the number of dimensions
%   that are removed. If X is a scalar, SHIFTDIM has no effect.
%
%   SHIFTDIM is handy for creating functions that, like SUM
%   or DIFF, work along the first non-singleton dimension.
%
%   Examples:
%      a = rand(1,1,3,1,2);
%      [b,n]  = shiftdim(a); % b is 3-by-1-by-2 and n is 2.
%      c = shiftdim(b,-n);   % c == a.
%      d = shiftdim(a,3);    % d is 1-by-2-by-1-by-1-by-3.
%
```

@tensor/size.m


```
function m = size(t,idx)
% TENSOR/SIZE Size of tensor.
%
%   D = SIZE(T) returns the size of the tensor.  Trailing singleton
%   dimensions may or may not be ignored, depending on the type of
%   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 = t.size(idx);

else

    m = t.size;

end
```

@tensor/squeeze.m


```
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.  2-D tensors are
%   unaffected by squeeze so that row vectors remain rows.
%
%   For example,
%       squeeze( 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));
```

@tensor/subsasgn.m


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


```
function a = subsref(t,s)
%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 '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


```
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.
%
%   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 is 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, 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

% COPY CONSTRUCTOR
```

@tensor/tensor.m


```
if isa(varargin{1}, 'cp_tensor') | isa(varargin{1}, 'tucker_tensor')

    t = full(varargin{1});
    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');
        end

        % -- Error Check --
        % First, check that the matching dimensions do indeed math
        sz = size(t.data);
        j = min( [length(sz),length(t.size)] );
        for i = 1 : j
            if (sz(i) ~= t.size(i))
                error('Specified size is incorrect');
            end
        end

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

        for i = j + 1 : length(t.size)
            if (t.size(i) ~= 1)
                error('Specified size is incorrect');
            end
        end

    end

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

% REVERSE MATRICIZE
if nargin == 4

    if (strcmp(varargin{4},'Kiers'))
```

@tensor/tensor.m


```
        t = tensorize_kiers(varargin{1}, varargin{2}, varargin{3});
    elseif (strcmp(varargin{4},'DDV'))
        t = tensorize_ddv(varargin{1}, varargin{2}, varargin{3});
    else
        error('Invalid type in argument 4 of tensor constructor');
    end

end

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


% ------------------------------------------------------------------
% -- Support Functions -----------------------------------------
% ------------------------------------------------------------------

function [t] = tensorize_ddv(A, i, dims)

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:-1:2)];
T = reshape(A, dims(indx2));
T = permute(T, indx2);

t.data = T;
t.size = dims;

%%%%%

function [t] = tensorize_kiers(A, i, dims)

if (prod(dims) ~= prod(size(A)))
  error('Invalid dimensions');
end

M = length(dims);
indx = circshift([1:M], -[0 i-1]);
T = reshape(A, dims(indx));
T = shiftdim(T, mod(M-i+1, M));

t.data = T;
t.size = dims;
```

@tensor/times.m


```
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, 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));
```

@tensor/uminus.m


```
function t = uminus(t)
%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.data = -t.data;
```

@tensor/uplus.m


```
function t = uplus(t)
%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.


% This function does nothing!
```

@tensor/xor.m


```
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 ~( issamesize(A,B)  |  (prod(size(A)) == 1)  |  (prod(size(B)) == 1) )
    error('Tensor size mismatch.')
end

C = multiarrayop(@xor,A,B);
```

@cp_tensor/and.m


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

@cp_tensor/cp_tensor.m


```
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 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
```

@cp_tensor/cp_tensor.m


```
end

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

@cp_tensor/disp.m


```
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,'\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


```
function display(t)
%CP_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,'\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


```
function a = double(t)
%CP_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))');
```

@cp_tensor/full.m


```
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, 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


```
function C = 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))');
```

@cp_tensor/gt.m


```
function C = 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))');
```

@cp_tensor/issamesize.m


```
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, 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
```

@cp_tensor/ldivide.m


```
function C = 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))');
```

@cp_tensor/le.m


```
function C = le(A,B)
%CP_TENSOR/LT 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))');
```

@cp_tensor/lt.m


```
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))');
```

@cp_tensor/matricize.m


```
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)');
```

@cp_tensor/minus.m


```
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 m = 1 : M
        u{m} = [A.u{m} B.u{m}];
    end
    C = cp_tensor(lambda,u);
    return;
end

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

@cp_tensor/mtimes.m


```
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
```

@cp_tensor/ndims.m


```
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, 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);
```

@cp_tensor/norm.m


```
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))');
```

@cp_tensor/not.m


```
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))');
```

@cp_tensor/or.m


```
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))');
```

@cp_tensor/order.m


```
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, 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


```
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);
```

@cp_tensor/plus.m


```
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))');
```

@cp_tensor/power.m


```
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))');
```

@cp_tensor/product.m


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

@cp_tensor/rdivide.m


```
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))');
```

@cp_tensor/size.m


```
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
```

@cp_tensor/subsasgn.m


```
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
```

@cp_tensor/subsref.m


```
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
```

@cp_tensor/times.m


```
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. (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


```
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;
```

@cp_tensor/uplus.m


```
function t = uplus(t)
%CP_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.

% This function does nothing!
```

@cp_tensor/xor.m


```
function C = xor(A,B)
%CP_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.

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

@tucker_tensor/and.m


```
function C = and(A,B)
%TUCKER_TENSOR/AND Logical AND.
%
%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))');
```

@tucker_tensor/disp.m


```
function disp(t)
%TUCKER_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,'\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


```
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,'\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


```
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))');
```

@tucker_tensor/full.m


```
function t = full(t,version)
%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, 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 m = 2 : M
        tmp = product(tmp, t.u{m}, m);
    end
    t = tensor(tmp, size(t));
    return;
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
            tmpi = prod(I(m+1:M));
            k(m) = floor (tmpk / tmpi) + 1;
            tmpk = tmpk - (k(m) - 1) * tmpi;
        end
        k(M) = tmpk + 1;
        % Create string containing indices
        for m = 1 : M
            if m == 1
                idxstr = int2str(k(1));
            else
                idxstr = [idxstr ',' int2str(k(m))];
            end
        end
        % Add in rank-1 matrix corresponding to
        % lambda(k1,k2,...,kM)
        tmp = 1;
        for m = 1 : M
            tmp = tmp * t.u{m}(:,k(m))';
            tmp = reshape(tmp, prod(I(1:m)), 1);
        end
        tmp = reshape(tmp, I);
        tmpstr = ['t.lambda.u(' idxstr ')'];
```

@tucker_tensor/full.m


```
        tmplambda = eval(tmpstr);
        if kmult == 1
            a = tmplambda * tmp;
        else
            a = a + tmplambda * tmp;
        end
    end
    t = tensor(a, size(t));
    return;
end


return;
```

@tucker_tensor/ge.m


```
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))');
```

@tucker_tensor/gt.m


```
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))');
```

@tucker_tensor/issamesize.m


```
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
```

@tucker_tensor/ldivide.m


```
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 and(ldivide(A),full(B))');
```

@tucker_tensor/le.m


```
function C = le(A,B)
%TUCKER_TENSOR/LT 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))');
```

@tucker_tensor/lt.m


```
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))');
```

@tucker_tensor/matricize.m


```
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)');
```

@tucker_tensor/minus.m


```
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))');
```

@tucker_tensor/mtimes.m


```
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
```

@tucker_tensor/ndims.m


```
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, 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);
```

@tucker_tensor/norm.m


```
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))');
```

@tucker_tensor/not.m


```
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))');
```

@tucker_tensor/or.m


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

@tucker_tensor/order.m


```
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, 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


```
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, 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


```
function C = plus(A,B)
%TUCKER_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.

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

@tucker_tensor/power.m


```
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))');
```

@tucker_tensor/product.m


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

@tucker_tensor/rdivide.m


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

@tucker_tensor/size.m


```
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, 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
```

@tucker_tensor/subsasgn.m


```
function t = subsasgn(t,s,b)
%TUCKER_TENSOR/SUBASGN Subscripted reference 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 '.'
        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')
    case '{}'
        u = t.u;
        u{s.subs{:}} = b;
        t = tucker_tensor(t.lambda, u);
    otherwise
        error('Invalid subsasgn');
end
```

@tucker_tensor/subsref.m


```
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 '.'
        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');
    case '{}'
        a = t.u{s.subs{:}};
    otherwise
        error('Invalid subsref');
end
```

@tucker_tensor/times.m


```
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))');
```

@tucker_tensor/tucker_tensor.m


```
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, 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}, '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');
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 = size(t.lambda);

if length(k) ~= length(t.u)
    error(['LAMBDA has order ', int2str(length(k)), ...
           ' but there are ', int2str(length(t.u)), ' matrices']);
```

@tucker_tensor/tucker_tensor.m


```
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


```
function t = 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.

t.lambda = -t.lambda;
```

@tucker_tensor/uplus.m


```
function t = 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.


% This function does nothing!
```

@tucker_tensor/xor.m


```
function C = 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.

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

DISTRIBUTION:

1	MS 9519	Tammy Kolda, 8962
1	MS 1110	Brett Bader, 9233
3	MS 9018	Central Technical Files, 8945-1
1	MS 0899	Technical Library, 9616
1	MS 9021	Classified Office, 8511/Technical Library, MS 0899, 9616
		DOE/OSTI via URL