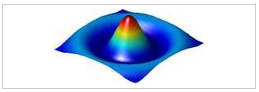
Download Submission
Code covered by the BSD License

Highlights from
MTIMESX - Fast Matrix Multiply with Multi-Dimensional Support

mtimesx.m
mtimesx_build(x) mtimesx_build compiles mtimesx.c with BLAS libraries
mtimesx_sparse(a,transa,b...mtimesx_sparse does sparse matrix multiply of two inputs
mtimesx_test_ddequalTest routine for mtimesx, op(double) * op(double) equality vs MATLAB
mtimesx_test_ddspeed(nn,d...Test routine for mtimesx, op(double) * op(double) speed vs MATLAB
mtimesx_test_dsequal Test routine for mtimesx, op(double) * op(single) equality vs MATLAB
mtimesx_test_dsspeed(nn,d...Test routine for mtimesx, op(double) * op(single) speed vs MATLAB
mtimesx_test_nd(n) Test routine for mtimesx, multi-dimensional speed and equality to MATLAB
mtimesx_test_sdequal Test routine for mtimesx, op(single) * op(double) equality vs MATLAB
mtimesx_test_sdspeed(nn,d...Test routine for mtimesx, op(single) * op(double) speed vs MATLAB
mtimesx_test_ssequal Test routine for mtimesx, op(single) * op(single) equality vs MATLAB
mtimesx_test_ssspeed(nn,d...Test routine for mtimesx, op(single) * op(single) speed vs MATLAB
View all files



MTIMESX - Fast Matrix Multiply with Multi-Dimensional Support

by James Tursa
30 Nov 2009 (Updated 23 Feb 2011)

Beats MATLAB 300% - 400% in some cases ... really!

Watch this File

5.0 | 29 ratings
Rate this file

142 Downloads (last 30 days)
File Size: 256 KB
File ID: #25977

File Information

Description MTIMESX is a fast general purpose matrix and scalar multiply routine that has the following features:

- Supports multi-dimensional (nD, n>2) arrays directly
- Supports Transpose, Conjugate Transpose, and Conjugate pre-operations
- Supports singleton expansion
- Utilizes BLAS calls, custom C loop code, or OpenMP multi-threaded C loop code
- Can match MATLAB results exactly or approximately as desired
- Can meet or beat MATLAB for speed in most cases

MTIMESX has six basic operating modes:

- BLAS: Always uses BLAS library calls
- LOOPS: Always uses C loops if available
- LOOPSOMP: Always uses OpenMP multi-threaded C loops if available
- MATLAB: Fastest BLAS or LOOPS method that matches MATLAB exactly (default)
- SPEED: Fastest BLAS or LOOPS method even if it doesn't match MATLAB exactly
- SPEEDOMP: Fastest BLAS, LOOPS, or LOOPOMP method even if it doesn't match MATLAB exactly

MTIMESX inputs can be:

single
double
double sparse

The general syntax is (arguments in brackets [ ] are optional):

```
mtimesx( [directive] )
mtimesx( A [,transa] ,B [,transb] [,directive] )
```

Where transa, transb, and directive are the optional inputs:

```
transa = A character indicating a pre-operation on A:
transb = A character indicating a pre-operation on B:
      The pre-operation can be any of:
      'N' or 'n' = No pre-operation (the default if trans_ is missing)
      'T' or 't' = Transpose
```

'C' or 'c' = Conjugate Transpose
'G' or 'g' = Conjugate (no transpose)
directive = One of the modes listed above, or other directives

Examples:

```
C = mtimesx(A,B) % performs the calculation C = A * B
C = mtimesx(A,'T',B) % performs the calculation C = A.' * B
C = mtimesx(A,B,'g') % performs the calculation C = A * conj(B)
C = mtimesx(A,'c',B,'C') % performs the calculation C = A' * B'
mtimesx('SPEEDOMP','OMP_SET_NUM_THREADS(4)') % sets SPEEDOMP mode with number of threads = 4
```

For nD cases, the first two dimensions specify the matrix multiply involved. The remaining dimensions are duplicated and specify the number of individual matrix multiplies to perform for the result. i.e., MTIMESX treats these cases as arrays of 2D matrices and performs the operation on the associated parings. For example:

If A is (2,3,4,5) and B is (3,6,4,5), then

mtimesx(A,B) would result in C(2,6,4,5), where C(:,:,i,j) = A(:,:,i,j) * B(:,:,i,j), i=1:4, j=1:5

which would be equivalent to the MATLAB m-code:

```
C = zeros(2,6,4,5);
for m=1:4
   for n=1:5
      C(:,:,m,n) = A(:,:,m,n) * B(:,:,m,n);
   end
end
```

The first two dimensions must conform using the standard matrix multiply rules taking the transa and transb pre-operations into account, and dimensions 3:end must match exactly or be singleton (equal to 1). If a dimension is singleton then it is virtually expanded to the required size (i.e., equivalent to a repmat operation to get it to a conforming size but without the actual data copy). This is equivalent to a bsxfun capability for matrix multiplication.

Acknowledgements *This file* inspired Mmx Multithreaded Matrix Operations On N D Matrices, Small Size Linear Solver, Frontal, and Merton Structural Credit Model (Matrixwise Solver).

MATLAB release MATLAB 7.4 (R2007a)

Other requirements A C compiler, such as the built-in lcc compiler. (You don't have to know anything about C to use mtimesx ... it is self-building)

Tags for This File Add Tags

4
5 for m14 for n15 c
6
as the code is pretty big
blas
bsxfun
but where is this for loop in your code
is there a simple way to call openmp for example
m
mtimes
multiply
n a
n b
n end end we could use openmp to send the multiplications on dif
speed
when doing c zeros2

Save Cancel

Rate this Submission (Rating not required)

Comment on this Submission

Submit

Please remember the following when providing feedback on a submission:

Be considerate of the file's author and other community members.
Provide specific information on what you like and dislike about the submission.

Comments and Ratings (128)

Comments and Ratings (128)

04 Jul 2014	Julien	From my understanding, the file "mex_C_win64.xml" replaces "mexopts.bat". "Error using mtimesx_build (line 169) A C/C++ compiler has not been selected with mex -setup". I replaced line 169 by : mexopts = [prefdir '\mex_C_win64.xml'];. It did work with a few warnings...
18 Jun 2014	Matt J	Thanks, James. This thread describes where the information has moved to http://www.mathworks.com/matlabcentral/answers/67521#answer_138814 I can post those xml files somewhere, if you like. However, is there no way to get mtimesx to compile simply with the "mex" command, maybe with a small sacrifice in performance?
07 Jun 2014	James Tursa	Hmmm ... Well, I don't have access to R2014 so I will have to code in the blind on this. My understanding is that MATLAB will in real time make a guess as to the correct compiler to use based on source code extension. So the information that was in mexopts.bat (user had already selected a compiler) is no longer available for my automatic build stuff. I will put out an update shortly, but may have to ask for info direct from the user for compiler stuff. I will try to get it out there sometime next week.
03 Jun 2014	Matt J	Thought I'd ping again. It doesn't look like mtimesx_build is compatible with R2014, since it looks for mexopts.bat, which is now gone. Can we hope to get a version that supports R2014? James?
29 May 2014	Matt J	I have the same problem as Safdar in R2014a. Even after running mex -setup, it appears as though mtimesx_build cannot find mexopts.bat. Possibly, the locations of relevant files/directories may have shifted?
22 May 2014	Stefan	
23 Apr 2014	Safdar	Sorry for the multiple posts. It seems something went wrong when submitting the post. Sorry
23 Apr 2014	Safdar	I got the following message ... Build routine for mtimesx ... Checking for PC ... Finding path of mtimesx C source code files ... Found file mtimesx.c in D:\Documents\Projects\mtimesx_20110223\mtimesx.c ... Found file mtimesx_RealTimesReal.c in D:\Documents\Projects\mtimesx_20110223 \mtimesx_RealTimesReal.c Error using mtimesx_build (line 169) A C/C++ compiler has not been selected with mex -setup Error in mtimesx (line 271) mtimesx_build;
23 Apr 2014	Safdar	Hi James, I unzipped the file in my working folder and tries to use mtimesx, I got the following message ... Build routine for mtimesx ... Checking for PC ... Finding path of mtimesx C source code files ... Found file mtimesx.c in D:\Documents\Projects\mtimesx_20110223\mtimesx.c ... Found file mtimesx_RealTimesReal.c in D:\Documents\Projects\mtimesx_20110223 \mtimesx_RealTimesReal.c Error using mtimesx_build (line 169) A C/C++ compiler has not been selected with mex -setup Error in mtimesx (line 271) mtimesx_build;
23 Apr 2014	Safdar	Hi James, I unzipped the file in my working folder and tries to use mtimesx, I got the following message ... Build routine for mtimesx ... Checking for PC ... Finding path of mtimesx C source code files ... Found file mtimesx.c in D:\Documents\Projects\mtimesx_20110223\mtimesx.c ... Found file mtimesx_RealTimesReal.c in D:\Documents\Projects\mtimesx_20110223 \mtimesx_RealTimesReal.c Error using mtimesx_build (line 169) A C/C++ compiler has not been selected with mex -setup Error in mtimesx (line 271) mtimesx_build;
23 Apr 2014	Safdar	Hi James, I unzipped the folder in my working path and then tried to used mtimesx, I got the following error ... Build routine for mtimesx ... Checking for PC ... Finding path of mtimesx C source code files ... Found file mtimesx.c in D:\Documents\Projects\mtimesx_20110223\mtimesx.c ... Found file mtimesx_RealTimesReal.c in D:\Documents\Projects\mtimesx_20110223

Comments and Ratings (128)

		\mtimesx_RealTimesReal.c Error using mtimesx_build (line 169) A C/C++ compiler has not been selected with mex -setup Error in mtimesx (line 271) mtimesx_build;
26 Mar 2014	Alireza	@Evan: have you tired blas_lib = 'lmwblas'?
09 Mar 2014	Evan	I would like to second Matthieu's question. For blas_lib, I've tried both blas_lib = '/Applications/MATLAB_R2013b.app/bin/maci64/lapack.spec'; and blas_lib = '/Applications/MATLAB_R2013b.app/bin/maci64/blas.spec'; With both, I get: ld: warning: ignoring file /Applications/MATLAB_R2013b.app/bin/maci64/blas.spec, file was built for unsupported file format Anyone have any ideas?
05 Dec 2013	Marc Crapeau	Works well for me on Linux. If it can helps, I have installed the library libblas.so in my home following the version "shared library" of this page: http://gcc.gnu.org/wiki/GfortranBuild Then I have compiled the mex file with the command return by the mtimesx routine when the automatic installation failed: blas_lib = 'the_actual_path_and_name_of_your_systems_BLAS_library'; mex('-DDEFINEUNIX','-largeArrayDims','mtimesx.c',blas_lib) Gained a factor 2 in speed for my 3D matrices multiplications, and a clearer matlab code :)
22 Aug 2013	Hannes	Hi all, I tried to locate compliers via mex -setup, but got the answer => No supported SDK or compiler was found on this computer. I installed Microsoft Windows SDK 7.1 but still not able to work. Does somebody know the path and install directory or what the problem is? thanks for help
18 Jul 2013	WK	I don't know why, I did the test again in the linux cluster and the windows one, it works in the parallel computation now. So strange!
18 Jul 2013	WK	James, by the way, what version of matlab last time you test this package? I place it in my labtop (windows 64) and linux cluster running matlab 2013, only few operations that mtimesx will beat matlab and it only beat is for less than 10%. I saw that this package was uploaded in 2011 so just wonder if matlab 2013 made lots of change to boost the performance.
17 Jul 2013	James Tursa	@WK: I don't have the parallel toolbox, so am unable to give much advice here. My understanding is that each worker in the parallel environment is a separate process, which I would expect would mean that MTIMESX could be used. But since I don't have the toolbox I can't verify this. Have you actually run some tests and gotten wrong results?
17 Jul 2013	WK	I just get it compiled and installed in my 64-bit matlab with windows SDK7 as well as the unix cluster. I find that it doesn't work in spmd, doesn't it?
15 Jul 2013	WK	Thanks for the great library. I have two questions. Do I have to compile it to get library? So it means the library is really depending on the machine, right? Also, our lab has a multi-node cluster, each node are equipped with 8 cores. I think this library is based on multi-threading, if I run the matlab code in spmd and in each core, I run your library so will it conflict? Sorry, I didn't have any experience in compiling mex before, I tried but I don't have MSVC installed, I only have mingw.
25 Jun 2013	JiaDa	How about announce another function for fast matrix inverse with multi-dimensional support? I wrote a version but I think your version will far fast than mine. :P
25 Jun 2013	JiaDa	
04 May 2013	Matthieu	Hello, I'm trying to use mtimesx on mac OS 10.8 with matlab R2011b but I don't understand how to compile it. I tryed several solutions described in comments but none worked. blas_lib = '/Applications/MATLAB_R2011b.app/bin/maci64' blas_lib = /Applications/MATLAB_R2011b.app/bin/maci64 >> mex('-DDEFINEUNIX','mtimesx.c',blas_lib) mtimesx.c: In function 'mexFunction':

Comments and Ratings (128)

mtimesx.c:592: warning: assignment discards qualifiers from pointer target type
ld: can't map file, errno=22 for architecture x86_64
collect2: ld returned 1 exit status

mex: link of ' "mtimesx.mexmaci64"' failed.

Error using mex (line 206)
Unable to complete successfully.

Can you help me ?

15 Mar 2013 James Tursa

@Agustin: It looks like you put the files in the lcc compiler directory. I would advise that you put the files in a work directory instead (not any of the "official" sys or bin etc directories that are installed with MATLAB) and try again. Make sure this work directory is on the MATLAB path. As a side note, in general it is *not* a good idea to put your own files into any of the "official" directories that are installed with MATLAB ... doing so risks breaking built-in MATLAB functions and capabilities.

15 Mar 2013 agustin darrosa

thanks james but i have this problem

Build routine for mtimesx
... Checking for PC
... Finding path of mtimesx C source code files
... Found file mtimesx.c in C:\Program Files\MATLAB\R2008a\sys\lcc\mtimesx\mtimesx.c
... Found file mtimesx_RealTimesReal.c in C:\Program Files\MATLAB\R2008a\sys\lcc\mtimesx
\mtimesx_RealTimesReal.c
... Opened the mexopts.bat file in C:\Users\Gusa\AppData\Roaming\MathWorks\MATLAB
\R2008a\mexopts.bat
... Reading the mexopts.bat file to find the compiler and options used.
... LCC is the selected compiler
... OpenMP compiler not detected ... you may want to check this website:
http://openmp.org/wp/openmp-compilers/
... Using BLAS library lib_blas = 'C:\Program Files\MATLAB\R2008a\extern\lib\win32
\lcc\libmwblas.lib'
... Now attempting to compile ...
mex('C:\Program Files\MATLAB\R2008a\sys\lcc\mtimesx\mtimesx.c',lib_blas,'-
DCOMPILER=LCC')
it can´t find C:\Program Files\MATLAB\R2008a\sys\lcc\mtimesx\mtimesx.exp
it cant´find C:\Program Files\MATLAB\R2008a\sys\lcc\mtimesx\mtimesx.lib
... mex mtimesx.c build completed ... you may now use mtimesx.

then i introduce:

C = mtimesx(a,b)
??? Undefined function or method 'mtimesx' for input arguments of type 'double'.

what can i do?

12 Mar 2013 James Tursa

@Agustin: For Windows, generally you just put all the files in a directory on the MATLAB path and then type mtimesx at the prompt. For other systems, see other posts below.

12 Mar 2013 agustin darrosa

Hi

How can i introduce this "function" in matlab??

I tried it but i have errors

Sorry for my ignorance

08 Mar 2013 James Tursa

@Tonio: No, MTIMESX does not do this. But from your description it sounds like something like this might work for you:

a = cell array of matrices
b = cell array of vectors
c = cellfun(@mtimes,a,b,'UniformOutput',false)

06 Mar 2013 Tonio

Can I used that for multiplying an (m,n,k) * (n,k) = (m,k) where each k 'slice' needs to be multiplied together and they may contain variables number of elements, i.e. 'm' can vary in each slice? In my code each slice is stored in a cell in matlab.

04 Feb 2013 James Tursa

@Robert: Yes, MTIMESX is CPU based, not GPU.

04 Feb 2013 Robert

Hello James,

nice work. A basically like to multiply n matrices B (400,400) with one Matrix A (400,400). I used arrayfun before, but your code is about 2.5 times faster. Soon I will get a Tesla NVidia graphic card and I would like to ask, is there any possibility to run the calculation on the GPU instead on the CPU?
I tried to call mtimesx with tww GPU Arrays but as my computation times is still fast (on my slow GPU) I guess he was not using my GPU right now.
Perhaps I also missed this, but to my understanding mtimesx is using the CPU only, or??

Thanks

Robert

Comments and Ratings (128)

23 Jan 2013	Charles	@Michael Thanks, that did it.
22 Jan 2013	Michael Völker	Charles, for my own record and to make it machine-searchable, I wrote earlier how I was able to compile mtimesx. Taking the liberties to quote myself: > On a 64Bit Debian based Linux I managed to compile it with > mex -DDEFINEUNIX CFLAGS="\$CFLAGS -march=native" -largeArrayDims -lmwblas -lmwlapack -lgomp mtimesx.c > using gcc-4.5. Can you try that, or did you already try? Michael
22 Jan 2013	Charles	@Sebastian Thanks for the response. I think I wasn't linking to the right file. Now, I seem to be linking to the blas library, but I'm getting a new error. I am using the code "mex -DEFINEUNIX 'mtimesx.c' -lmwblas" but I get an error that reads "mtimesx.c:592: warning: assignment discards qualifiers from pointer target type ". I saw that someone else had gotten this error before and the code was fixed, so I downloaded the files again but I still am getting the error. When I tried to use mtimesx in my code I get an error and matlab crashes. I haven't had any problems with other mex files, so it seems to be specific to mtimesx. Any help is appreciated. Thanks in advance.
21 Jan 2013	Sebastiaan	@Charles: under Unix, the BLAS library linked to is normally mwblas , not blas_lib.
20 Jan 2013	Charles	This program is great. Really useful. I am having great success with it on my PC. However, I cannot seem to create the mex file on a UNIX computer. I am defining the BLAS library as blas_lib. Then I enter the code "mex('-DDEFINEUNIX','mtimesx.c',blas_lib)". However, I get a string of errors that are all similar to "mtimesx.c;(.text+0x76a1)undefined reference to `saxpy_'". It says this for saxpy, sger, sdot, daxpy, dger, ddot, sgemv,ssyrk, ssyr2k, sgemm, dgemv, dsyrk and dgemm. Can anyone please help me with creating the .mexa64 file? Thanks.
30 Oct 2012	James Tursa	@KSIDHU: Why can't you use .* and ./ directly? What are the exact dimensions of your two matrices? If the matrices need singleton expansion, then look at the function bsxfun. What is the overall operation you are trying to do?
29 Oct 2012	KSIDHU	Hi How can I perform elementwise multiplication and division of two matrices? I have looked through the manual but cannot find any information on it. Is there a way to check if BLAS multi-threading is available? Is this better than OpenMP multi-thread I need to perform .*, ./ on matrices with numel ~3000x100,000 so any tips would be great, and also is there a multi threaded sum function for large matrices (along dim 1 or 2). MATLAB 2010a Thanks in advance KS
13 Sep 2012	James Tursa	I like the reasoning you both have spelled out. I will make logical*other conform to the other, and make logical*logical output a double by default but have an option to do logical AND/OR operations and output a logical instead. Thanks for your inputs.
13 Sep 2012	John	I like Michael's reasoning and I think he is right about the 'double' result. So in general I think that the double should be the default. However, given that i enjoy being efficient with data, I would appreciate a 'logical' option.
12 Sep 2012	Michael Völker	Damn, the E-Mail notification stopped to work, so I didn't know you replied to my message. My spontaneous thought was that (logical matrix) * (logical matrix) should *of course* result in a double output, since matrix multiplication involves a sum, so we should expect to get results different than 0 or 1. So, as you already guessed, I would like to have an auto-adjust-precision feature. This is what I would expect: [inputA] * [inputB] ==> [output] --- logical | logical | double logical | single | single logical | double | double single | logical | single double | logical | double (I hope it does not get formatted too ugly...) I also like John's idea to make the behaviour more optional.
11 Sep 2012	John	Personally, I do not understand why it should be a 'double' result. So, my vote goes towards a logical result. However, would it be possible to add this as an option? (e.g.: mtimesx(a,b,'logical') will produce

Comments and Ratings (128)

a logical result, if a and b are logical). This way the advanced user can be efficient with data, whereas the less experienced user will still get results that they are familiar with....

28 Aug 2012 James Tursa

@John: OK, that's two votes. Automatic logical conversion goes into the next version.

Q: Should a (logical matrix) * (logical matrix) give a double matrix result (ala MATLAB), or a logical matrix result (i.e., essentially treating the * - operations as AND OR operations)?

28 Aug 2012 John

Excellent code!

I experienced the same 'error' as Michael Völker, so I think automatically converting the logical/sparse data to double might be a good idea as an enhancement!

Nevertheless, great work! Saved me a lot of time/coding.

16 Aug 2012 James Tursa

@Michael Völker: The error message you are getting is not a bug. As stated in the doc, using arguments that are not double or single will cause MTIMESX to invoke the built-in mtimes function to do the matrix multiply. If MATLAB will do it then great ... you get that result. But if MATLAB will not do it then you will get whatever error message MATLAB puts out. This is the intended behavior. You can convert the logical to double first, of course, to avoid the error message. If you would like MTIMESX to do this automatically for logical inputs then I can take that as an enhancement request (seems reasonable to me).

16 Aug 2012 Michael Völker

James,

I found a bug with logical input.
Try:

foo = mtimesx(true(2,2), randn(1,1,5));

And you probably get this error message:

Error using *
Inputs must be 2-D, or at least one input must be scalar.
To compute elementwise TIMES, use TIMES (.*) instead.

Error in mtimesx_sparse (line 47)
result = a * b;

Michael

28 Jul 2012 Sam T

James:

I would like to thank you for coming up with such a brilliant piece of work. An excellent and well written code!!
Lately, I have been trying to use MTIMESX code to multiply two big n-dimensional matrices (A = 400x400x400 and B = 400x200). But for simplistic purposes, consider the following example:

A = 3 x 3 x 3
B = 3 x 2

and C = A*B;
such that dimension of C = 3 x 3 x 2

where C(i, j, l) = A(i, j, k) * B(k, l);
i.e.
C(:, :, 1) = A(:, :, k) * B(k, 1);
C(:, :, 2) = A(:, :, k) * B(k, 2);

Now, if I use mtimesx(A, B), then resultant matrix is (3 x 2 x 3) as compared (3 x 3 x 2).

This is because first two dimensions specify the matrix multiply involved. In this case it becomes 1st and 2nd dimension of matrix A while it should ideally be 2nd and 3rd dimension of matrix A to get the desired result.

I am wondering if there is a way around it? I look forward to hearing from you.

Thanks for your help.

25 Jul 2012 James Tursa

@Sam T: I don't think I get it yet. In your example you have:

C(:, :, 1) = A(:, :, k) * B(k, 1);

A(:,:,k) is 3x3
B(k,1) is 1x1

Are you trying to do an nD scalar*matrix multiply here? MTIMESX can do that (with some modifications of the inputs), but I am not quite sure that is really what you want. Can you clarify? Maybe write out an explicit for loop showing how C is to be filled in its entirety?

24 Jul 2012 Sam T

James:

I would like to thank you for coming up with such a brilliant piece of work. An excellent and well written code!!
Lately, I have been trying to use MTIMESX code to multiply two big n-dimensional matrices (A = 400x400x400 and B = 400x200). But for simplistic purposes, consider the following example:

A = 3 x 3 x 3
B = 3 x 2

Comments and Ratings (128)

and C = A*B;
such that dimension of C = 3 x 3 x 2

where C(i, j, l) = A(i, j, k) * B(k, l);
i.e.
C(:, :, 1) = A(:, :, k) * B(k, 1);
C(:, :, 2) = A(:, :, k) * B(k, 2);

Now, if I use mtimesx(A, B), then resultant matrix is (3 x 2 x 3) as compared (3 x 3 x 2).

This is because first two dimensions specify the matrix multiply involved. In this case it becomes 1st and 2nd dimension of matrix A while it should ideally be 2nd and 3rd dimension of matrix A to get the desired result.

I am wondering if there is a way around it? I look forward to hearing from you.

Thanks for your help.

24 Jul 2012 Sam T

James:

I would like to thank you for coming up with such a brilliant piece of work. An excellent and well written code!!
Lately, I have been trying to use MTIMESX code to multiply two big n-dimensional matrices (A = 400x400x400 and B = 400x200). But for simplistic purposes, consider the following example:

A = 3 x 3 x 3
B = 3 x 2

and C = A*B;
such that dimension of C = 3 x 3 x 2

where C(i, j, l) = A(i, j, k) * B(k, l);
i.e.
C(:, :, 1) = A(:, :, k) * B(k, 1);
C(:, :, 2) = A(:, :, k) * B(k, 2);

Now, if I use mtimesx(A, B), then resultant matrix is (3 x 2 x 3) as compared (3 x 3 x 2).

This is because first two dimensions specify the matrix multiply involved. In this case it becomes 1st and 2nd dimension of matrix A while it should ideally be 2nd and 3rd dimension of matrix A to get the desired result.

I am wondering if there is a way around it? I look forward to hearing from you.

Thanks for your help.

24 Jul 2012 Sam T

James:

I would like to thank you for coming up with such a brilliant piece of work. An excellent and well written code!!
Lately, I have been trying to use MTIMESX code to multiply two big n-dimensional matrices (A = 400x400x400 and B = 400x200). But for simplistic purposes, consider the following example:

A = 3 x 3 x 3
B = 3 x 2

and C = A*B;
such that dimension of C = 3 x 3 x 2

where C(i, j, l) = A(i, j, k) * B(k, l);
i.e.
C(:, :, 1) = A(:, :, k) * B(k, 1);
C(:, :, 2) = A(:, :, k) * B(k, 2);

Now, if I use mtimesx(A, B), then resultant matrix is (3 x 2 x 3) as compared (3 x 3 x 2).

This is because first two dimensions specify the matrix multiply involved. In this case it becomes 1st and 2nd dimension of matrix A while it should ideally be 2nd and 3rd dimension of matrix A to get the desired result.

I am wondering if there is a way around it? I look forward to hearing from you.

Thanks for your help.

02 Jul 2012 Boaz Schwartz

01 Jun 2012 Clay Fulcher

mtimesx is an awesome code. Besides speeding up my structural dynamics and signal processing applications, it is much easier to apply than using loops. I'm also using a multi-D matrix inverter called multinv that can be downloaded from this forum. The two codes together allow me to operate on matrices of functions as easily as matrices of single values.

30 May 2012 Clay Fulcher

@ James Tursa: Thanks James! I greatly appreciate it!

14 May 2012 James Tursa

@ Clay Fulcher: Some options to possibly get more speed:

1) NDFUN by Peter Boettcher. E.g., see this thread:

http://www.mathworks.com/matlabcentral/newsreader/view_thread/294669#861239

Comments and Ratings (128)

2) FEX submissions. E.g., see this submission:

http://www.mathworks.com/matlabcentral/fileexchange/31222

3) Wait a couple of months for the next MTIMESX submission, when this capability will be included (along with in-place operations and nD matrix multiply versions of PROD and CUMPROD).

13 May 2012 Clay Fulcher

James, I need to invert a 3D array of function values. I have to step through each slice of the array in a loop, inverting each slice. Do you know of any way to do this more efficiently?

08 Feb 2012 Jonathan Sullivan

@ James
The array sizes I'm working with are much larger what I posted. The size of A is 100x33x3. The size of B is 33x1x3x5000x5. That might be why you had trouble recreating it. I'm running on Windows 7, 64 bit, MATLAB 2011b, compiler is cl. If it makes a difference, some of the values in the first matrix are -inf.

08 Feb 2012 James Tursa

@ Jonathan: Yes, weird indeed. For slices 100x100 in size, MTIMESX will always use BLAS dgemm calls regardless of the method setting (you can verify this by using the 'DEBUG' setting). That is, the exact same code path gets executed ragardless of whether 'MATLAB', 'SPEED', or 'SPEEDOMP' is set. That setting should have had no effect on this particular outcome. Also, I have been unable to reproduce this error on a 32-bit WinXP system. Can you provide me some specifics on your MATLAB version, machine OS, compiler used, etc? Does it crash if you restart MATLAB and immediatly try the example?

08 Feb 2012 Jonathan Sullivan

@ James
So the problem, at least in part, is caused by using the 'SPEEDOMP' flag. I removed the flag, and now it works. Weird, huh?

08 Feb 2012 James Tursa

@ Jonathan Sullivan: Your first example without repmat should have worked. Looks like I introduced a bug in the latest release. I will look into this right away ...

08 Feb 2012 Jonathan Sullivan

James,

Fantastic code. It has come in handy. It is well documented, incredibly fast, and extremely useful, especially for N-D arrays.

For the N-D case, this code would be even more powerful if the singleton dimension capability were expanded. Currently, all the dimensions from 3:end must be either the same size (A to B), or must all be singleton for one of the variables.

For example
A = rand(100,100,3);
B = rand(100,100,3,5);
C = mtimesx(A,B); % Does not work. Crashes MATLAB.

You can get around this by using repmat:
A = rand(100,100,3);
B = rand(100,100,3,5);
C = mtimesx(repmat(A,[1 1 1 5]),B); % Works, but slow

Unfortunately, explicitly expanding out large arrays has overhead which is more than I'd like.

Overall code. A+.

30 Oct 2011 Michael Völker

James, thank you for your hint to using 'lapack' from the FEX.
Indeed it produces the desired result, by calling

lapack('D=DSDOT(i,s,i,s,i)', length(A), A, 1, A, 1)

with 'A' from my example above. Unfortunately, this is ~5 times slower than calling mtimesx() with double input.

On double input, calling the equivalent 'DDOT' routine from the lapack-package takes more than 10 times longer than your mtimesx (wow, by the way...) to compute the very same result.
The difference appears to be even worse for complex input.

So I am still hoping that you might somehow get this feature in your code and maintain the speed...

19 Oct 2011 James Tursa

@ Michael Völker: FYI, another way to use the DSDOT routine is to use this submission:

http://www.mathworks.com/matlabcentral/fileexchange/16777-lapack

Not sure if it works on 64-bit systems, but you might give it a try.

19 Oct 2011 James Tursa

@ Michael Völker: Thanks for the comments. I will look into your request. There is a BLAS routine called DSDOT that does what you request. I will look into how easily I can incorporate this into MTIMESX for BLAS specific results. For the SPEED modes MTIMESX sometimes uses custom code to calculate the dot product (e.g., see function RealKindDotProduct), and in fact the accumulation is done in double precision regardless of input type (calculations themselves are single for single inputs), so it might not be too difficult to adapt this to return the double precision output if requested instead of always converting back to single for single inputs. Thanks for the suggestion.

Comments and Ratings (128)

19 Oct 2011 Michael Völker

Incredibly great work. Thank you for the elaborate code, the very useful features, the intuitive syntax and behaviour and the docu.

On a 64Bit Debian based Linux I managed to compile it with
mex -DDEFINEUNIX CFLAGS="\$CFLAGS -march=native" -largeArrayDims -lmwblas -lmwlapack -lgomp mtimesx.c
using gcc-4.5.

After using mtimesx for a while I currently miss only one feature, which is to define the precision of the output independently from the input variables.
This would be useful when calculating the sum of squares of a huge single-precision vector where the correct result exceeds realmax('single') and prior casting takes a lot of time.

Example:
>> A = randn(1e8, 1,'single'); % plenty of sane values
>> A(1) = 1e20; % only few outliers
>> sos = A' * A % or sos = mtimesx(A, 'C', A)

sos =

Inf

Doing A = double(A) beforehand solves the problem:
>> sos = A' * A

sos =

1.000000040081755e+40

but
>> tic, A = double(A); toc
Elapsed time is 0.472007 seconds.

In fact, that casting to double takes much more time than the actual computation in mtimesx. Since I do this repeatedly in an iteration, the overhead sums up to minutes.

So my wish would be like this:
C = mtimesx(A, B, 'double'); % C is double even if A or B are single

Nevertheless, thank you so much for this code.

21 Jun 2011 James Tursa

@Gary: Follow-up, ndfun as I recall only works for real variables. If you have complex variables then the only efficient option I could suggest would be a custom mex routine.

21 Jun 2011 James Tursa

@Gary: For this type of matrix product you should be using something like ndfun('mprod',a) by Peter Boettcher. You can find the source code here:

http://www.mit.edu/~pwb/matlab/

If you need a 64-bit version of this you can see the hack I posted on this thread:

http://www.mathworks.com/matlabcentral/newsreader/view_thread/294669#791467

Using ndfun will avoid all the array slice copies you are doing in the above example code.

21 Jun 2011 Gary

Thanks for the answer, but I think you misunderstood my problem or me your answer.
I have eg 10000 matrices and want to calculate the product. By dividing into 2*5000 I can use the advantage of your nD multiply. Here is the code:

```
a=rand(n,n,L)

tic
b=a(:,:,1);
for i=2:L
b=a(:,:,i)*b;
end
toc

c=a;
tic
while(L~=1)
L2=floor(L/2)*2;
d=c(:,:,end);
c = mtimesx(c(:,:,2:2:L2),c(:,:,1:2:L2-1));
if(L2~=L)
c(:,:,end+1) = d;
end
L=size(c,3);
end
toc
```

So will this never be faster than normal multiplies?

20 Jun 2011 James Tursa

@Gary: MTIMESX is generally not going to be any faster at generic matrix multiplies than MATLAB since it is calling exactly the same BLAS routines as MATLAB. The exceptions are small (4x4 or less) multiplies, dot products, outer products, and some matrix-vector multiplies since they can sometimes be done faster with inline code which MTIMESX uses in the SPEED or LOOP modes. Also, the nD multiply case is almost always faster with MTIMESX than the equivalent loop(s) in MATLAB for any size since array slice copies are avoided.

20 Jun 2011 Gary

Hi

Updates

04 Dec 2009	Fixed bug for (scalar) * (sparse) when the scalar contains an inf or NaN (in which case the zeros do not stay zero). Slight update to pdf doc.
07 Dec 2009	Fixed bug in scalar multiply code that was causing incorrect results & crashes.
10 Dec 2009	Added singleton expansion capability for multi-dimensional matrix multiplies.
11 Dec 2009	Fixed a bug for empty transa or transb inputs. Now treats these the same as 'N'. Also simplified the nD singleton expansion code a bit.
07 Jan 2010	Added the multi-dimensional test routine mtimesx_test_nd.m for speed and equality tests. Added its description to the pdf file.
16 Feb 2010	Fixed a typo in the build routine for 64-bit systems, changed -largearraydims to -largeArrayDims
23 Feb 2010	Fixed a bug for some of the (row vector) * (matrix) and (matrix transposed) * (column vector) operations in MATLAB mode that involved incorrect dimensions in the dgemv and sgemv calls.
10 Aug 2010	Added capability for nD scalar multiplies (i.e., 1x1xN * MxKxN). Replaced the buggy mxRealloc API routine with custom code. Updated mtimesx_test_nd.m file.
07 Oct 2010	Added OpenMP support for custom code Expanded sparse * single and sparse * nD support Fixed (nD complex scalar)C * (nD array) bug
23 Feb 2011	Fixed typos in the dsyrk, dsyr2k, ssyrk, and ssyr2k BLAS function prototypes. (These typo fixes should not change any results)