MMTK: MATLAB Wrappers for the Mimetic Methods Toolkit

Generated by Doxygen 1.8.6

Thu Sep 10 2015 15:44:26

Contents

Chapter 1

Introduction

We define numerical methods that are based on discretizations preserving the properties of their continuum counterparts to be **mimetic**.

The **Mimetic Methods Toolkit (MTK)** is a C++ library for mimetic numerical methods. It is a set of classes for **mimetic quadratures**, **mimetic interpolation**, and **mimetic discretization methods** for the numerical solution of ordinary and partial differential equations.

1.1 MTK Concerns

Since collaborative development efforts are definitely important in achieving the level of generality we intend the library to possess, we have divided the library's source code according to the designated purpose the classes possess within the library. These divisions (or concerns) are grouped by layers, and are hierarchically related by the dependence they have among them.

One concern is said to depend on another one, if the classes the first concern includes, rely on the classes the second concern includes.

In order of dependence these are:

1. Roots.
2. Enumerations.
3. Tools.
4. Data Structures.
5. Numerical Methods.
6. Grids.
7. Mimetic Operators.

1.2 MTK Flavors

The MTK collection of wrappers is:

1. MMTK: MATLAB wrappers collection for MTK; intended for sequential computations.

Others are being designed and developed.

1.3 Contact, Support and Credits

The MTK is developed by researchers and adjuncts to the `Computational Science Research Center (CSRC)` at `San Diego State University (SDSU)`.

Developers are members of:

1. Mimetic Numerical Methods Research and Development Group.
2. Computational Geoscience Research and Development Group.
3. Ocean Modeling Research and Development Group.

Currently the developers are:

1. **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu** - ejspeiro
2. Jose E. Castillo, Ph.D. - jcastillo at mail dot sdsu dot edu
3. Guillermo F. Miranda, Ph.D. - unigrav at hotmail dot com
4. Christopher P. Paolini, Ph.D. - paolini at engineering dot sdsu dot edu
5. Angel Boada.
6. Johnny Corbino.
7. Raul Vargas-Navarro.

1.4 Acknowledgements and Contributions

The authors would like to acknowledge valuable advising, contributions and feedback, from research personnel at the Computational Science Research Center at San Diego State University, which were vital to the fruition of this work. Specifically, our thanks go to (alphabetical order):

1. Mohammad Abouali, Ph.D.
2. Dany De Cecchis, Ph.D.
3. Julia Rossi.

Chapter 2

Programming Tools

The development of MTK has been made possible through the use of the following applications:

1. Editor: Kate - KDE Advanced Text Editor. Version 3.13.3. Using KDE Development Platform 4.13.3 (C) 2000-2005 The Kate Authors.
2. Compiler: gcc version 4.4.5 (Ubuntu/Linaro 4.4.4-14ubuntu5). Copyright (C) 2013 Free Software Foundation, Inc.
3. Debugger: GNU gdb (Ubuntu 7.7.1-0ubuntu5∼14.04.2) 7.7.1. Copyright (C) 2014 Free Software Foundation, Inc.

Chapter 3

Licensing and Modifications

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Modifications to source code should be reported to: esanchez@mail.sdsu.edu and a copy of the modified files should be reported once modifications are completed. Documentation related to said modifications should be included.
2. Redistributions of source code must be done through direct downloads from the project's GitHub page: http://www.csrc.sdsu.edu/mtk
3. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
4. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
5. Usage of the binary form on proprietary applications shall require explicit prior written permission from the the copyright holders.
6. Neither the name of the copyright holder nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

The copyright holders provide no reassurances that the source code provided does not infringe any patent, copyright, or any other intellectual property rights of third parties. The copyright holders disclaim any liability to any recipient for claims brought against recipient by any third party for infringement of that parties intellectual property rights.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Chapter 4

Read Me File and Installation Instructions

README File for the Mimetic Methods Toolkit (MTK)

By: **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu**

1. Description

We define numerical methods that are based on discretizations preserving the properties of their continuum counterparts to be **mimetic.**

The **Mimetic Methods Toolkit (MTK)** is a C++ library for mimetic numerical methods. It is arranged as a set of classes for **mimetic quadratures,** **mimetic interpolation**, and **mimetic discretization** methods for the numerical solution of ordinary and partial differential equations.

This collection of **MATLAB Wrappers for the MTK (MMTK)** allows developers to invoke the MTK from a MATLAB environment.

---

2. Dependencies

This README assumes all of these dependencies are installed in the following folder:

```
/home/ejspeiro/Libraries/
```

In this version, the MTK optionally uses ATLAS-optimized BLAS and LAPACK routines for the internal computation on some of the layers. However, ATLAS requires both BLAS and LAPACK in order to create their optimized distributions. Therefore, the following dependencies tree arises:

For Linux and OS X:

1. MATLAB R2014a or greater - Available from: http://www.mathworks.com/

1. MTK - Available from: http://csrc.sdsu.edu/mtk/

3. Installation

The following instructions assume MATLAB R2014a or greater.

You have two options, either follow the instructions given, **expert install**, or use the provided **patch file**, **naive install**.

EXPERT PART 1. CONFIGURATION OF THE MEX COMPILER.

From your MATLAB prompt, type:

```
  mex -setup C++
```

```
MEX configured to use 'g++' for C++ language compilation.
Warning: The MATLAB C and Fortran API has changed to support MATLAB
     variables with more than 2^32-1 elements. In the near future
     you will be required to update your code to utilize the
     new API. You can find more information about this at:
```

http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-64-bit-api.html.

~~~
```
~~~

EXPERT PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.

The previous step creates the following file:

~~~
```
/home/ejspeiro/.matlab/R2014a/mex_C++_glnxa64.xml
```
~~~

The purpose of this section is to configure the MATLAB R2014a Mex compiler so that it can work with the latest C++ standard (C++11).

Please execute the following changes on the aforementioned file (line numbers may differ):

Lines 26 and 27:

~~~
```
CMDLINE1="$CXX -std=c++11 -c $DEFINES $INCLUDE $CXXFLAGS $OPTIM $SRC -o $OBJ"
CMDLINE2="$LDXX -std=c++11 $LDFLAGS $LDTYPE $LINKOPTIM $LINKEXPORT $OBJS $CXXLIBS $LINKLIBS -o $EXE"
```
~~~

Line 33:

~~~
```
CXXFLAGS="-std=c++11 -ansi -fexceptions -fPIC -fno-omit-frame-pointer -pthread"
```
~~~

Line 35 and 26:

~~~
```
CXXOPTIMFLAGS="-std=c++11 -O -DNDEBUG"
CXXDEBUGFLAGS="-std=c++11 -g"
```
~~~

Line 38:

~~~
```
LDXX="gfortran"
```
~~~

Line 59:

~~~
```
CXXFLAGS="-std=c++11 -ansi -pthread"
```
~~~

NAIVE PART 1. CONFIGURATION OF THE MEX COMPILER.

From your MATLAB prompt, type:

~~~
```

  mex -setup C++

MEX configured to use ’g++’ for C++ language compilation.
Warning: The MATLAB C and Fortran API has changed to support MATLAB
     variables with more than 2^32-1 elements. In the near future
     you will be required to update your code to utilize the
     new API. You can find more information about this at:

 http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-
64-bit-api.html.

```
~~~

NAIVE PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.

```
cd $HOME/.matlab/R2014a
chmod +w mex_C++_glnxa64.xml
patch < mex_C++_glnxa64.patch
chmod -w mex_C++_glnxa64.xml
```
Exit terminal, and restart MATLAB. You can use C++11 to create MEX files now!

PART 3: CONFIGURATION OF THE MAKEFILE.

The following steps are required the build and test the MTK. Please use the accompanying Makefile.inc file, which should provide a solid template to start with. The following command provides help on the options for make:

```

## $ make help

Makefile for the MMTK.

Options are:

- all: builds he library, the tests, and examples.

gendoc: generates the documentation for the library.

- **- clean: cleans ALL the generated files.**

```

PART 4. BUILD THE MMTK.

From your shell, at the base folder of the MMTK, just type:

```
make
```

If successful you'll read:

```
----- Library created! Check in /home/ejspeiro/Dropbox/MTK/lib
```

4. Frequently Asked Questions

Q: Why haven't you guys implemented GBS to build the library?
A: I'm on it as we speak! ;)

Q: When will the other flavors be ready?
A: Soon! I'm working on getting help on developing those.

Q: Is there any main reference when it comes to the theory on Mimetic Methods?
A: Yes! Check: http://www.csrc.sdsu.edu/mimetic-book

Q: Do I need to generate the documentation myself?
A: You can if you want to... but if you DO NOT want to, just go to our website.

5. Contact, Support, and Credits

The MTK is developed by researchers and adjuncts to the
Computational Science Research Center (CSRC)
at San Diego State University (SDSU).

Developers are members of:

1. Mimetic Numerical Methods Research and Development Group.
2. Computational Geoscience Research and Development Group.
3. Ocean Modeling Research and Development Group.

Currently the developers are:

Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu -

- Jose E. Castillo, Ph.D. - jcastillo at mail dot sdsu dot edu

- Guillermo F. Miranda, Ph.D. - unigrav at hotmail dot com
- Christopher P. Paolini, Ph.D. - paolini at engineering dot sdsu dot edu
- Angel Boada.
- Johnny Corbino.
- Raul Vargas-Navarro.

Finally, please feel free to contact me with suggestions or corrections:

Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu -

Thanks and happy coding!

Chapter 5

Tests and Test Architectures

Tests are given in the `files list` section. They are provided in the /tests/ folder within the distributed software.

In this page we intend to make a summary of all of the architectures in where the MTK has been tested. The MTK is intended to be as portable as possible throughout architectures. The following architectures have provided flawless installations of the API and correct execution of the examples:

```
1. Linux 3.2.0-23-generic-pae #36-Ubuntu SMP i386 GNU/Linux
   Intel(R) Pentium(R) M processor 1.73GHz 2048 KB of cache and stepping of 8
   gcc version 4.6.3 (Ubuntu/Linaro 4.6.3-1ubuntu5)
```

Further architectures will be tested!

Chapter 6

Examples

Examples are given in the `files list` section. They are provided in the /examples/ folder within the distributed software.

Chapter 7

User Manual, References and Theory

The main source of references for this work can be found in:

http://www.csrc.sdsu.edu/mimetic-book/

However, a .PDF copy of this manual can be found here.

Chapter 8

File Index

8.1 File List

Here is a list of all files with brief descriptions:

Chapter 9

File Documentation

9.1 examples/Div1D.m File Reference

Variables

- format shortG
- close all
- clc

9.1.1 Variable Documentation

9.1.1.1 clear all

Definition at line 22 of file Div1D.m.

9.1.1.2 clc

Definition at line 24 of file Div1D.m.

9.1.1.3 format shortG

Definition at line 21 of file Div1D.m.

9.2 Div1D.m

```
00001 % FUNCTION_NAME - Div1D
00002 % Creation of different high-order mimetic divergence operators with the MMTK.
00003 %
00004 % Other m-files required: None.
00005 % Subfunctions: The MMTK.
00006 % MAT-files required: None.
00007 %
00008 % See also: MMTKDiv1D.
00009 %
00010 % Author: Eduardo Sanchez, Ph.D.
00011 % Computational Science Research Center.
00012 % 5500 Campanile Drive, San Diego State University.
00013 % Email: esanchez at mail dot sdsu dot edu
```

```
% Website: http://www-rohan.sdsu.edu/~sanche94/
% 2015-09-10 14:18:35

%------------- BEGIN CODE --------------

addpath('../mexsrc');

format shortG;
close all;
clear all;
clc;

MMTKDiv1D(2)

%------------- END OF CODE --------------
```

9.3 include/mmtk.h File Reference

Includes the entire API.

9.3.1 Detailed Description

This file contains the related documentation

Author

: Eduardo J. Sanchez: esanchez at mail dot sdsu dot edu

Warning

This file contains no logic for the MMTK. It has been created just for documentation purposes.

Definition in file mmtk.h.

9.4 mmtk.h

```

/*
Copyright (C) 2015, Computational Science Research Center, San Diego State
University. All rights reserved.

Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:

1. Modifications to source code should be reported to: esanchez@mail.sdsu.edu
and a copy of the modified files should be reported once modifications are
completed. Documentation related to said modifications should be included.

2. Redistributions of source code must be done through direct
downloads from the project's GitHub page: http://www.csrc.sdsu.edu/mtk

3. Redistributions of source code must retain the above copyright notice, this
list of conditions and the following disclaimer.

4. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation and/or
other materials provided with the distribution.

5. Usage of the binary form on proprietary applications shall require explicit
prior written permission from the the copyright holders.

6. Neither the name of the copyright holder nor the names of its contributors
may be used to endorse or promote products derived from this software without
specific prior written permission.
```

```

The copyright holders provide no reassurances that the source code provided does
not infringe any patent, copyright, or any other intellectual property rights of
third parties. The copyright holders disclaim any liability to any recipient for
claims brought against recipient by any third party for infringement of that
parties intellectual property rights.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/
```

9.5 Makefile.inc File Reference

9.6 Makefile.inc

```
# Makefile setup file for MMTK.

SHELL := /bin/bash

# Please set the following variables up:

#   2. Absolute path to base directory of the MTK... where is the MTK?
#   __________________________________________________________________

BASE = /home/ejspeiro/Dropbox/MTK

#   2. The machine (platform) identifier and required precision.
#   __________________________________________________________________

# Options are:
# - LINUX: A LINUX box installation.
# - OSX: Uses OS X optimized solvers.

PLAT = LINUX

# Options are:
# - SINGLE: Use 4 B floating point numbers.
# - DOUBLE: Use 8 B floating point numbers.

PRECISION = DOUBLE

#   3. Optimized solvers and operations by means of ATLAS in Linux?
#   __________________________________________________________________

# If you have selected OSX in step 1, then you don’t need to worry about this.

# Options are ON xor OFF:

ATL_OPT = OFF

#   4. Paths to dependencies (header files for compiling).
#   __________________________________________________________________

# GLPK include path (soon to go):

GLPK_INC = $(HOME)/Libraries/glpk-4.55/include

# Linux: If ATLAS optimization is ON, users should only provide the path to
# ATLAS:

ATLAS_INC = $(HOME)/Libraries/ATLAS_3.8.4-CORE/include

# OS X: Do nothing.

#   5. Paths to dependencies (archive files for (static) linking).
#   __________________________________________________________________

# GLPK linking path (soon to go):

GLPK_LIB = $(HOME)/Libraries/glpk-4.55/lib/libglpk.a

# If optimization is OFF, then provide the paths for:

BLAS_LIB = $(HOME)/Libraries/BLAS/libblas.a
LAPACK_LIB = $(HOME)/Libraries/lapack-3.4.1/liblapack.a

# WARNING: Vendor libraries should be used whenever they are available.

# However, if optimization is ON, please provide the path the ATLAS’ archive:

ATLAS_LIB   = $(HOME)/Libraries/ATLAS_3.8.4-CORE/ATLAS_3.8.4-BUILD-Citadel/lib

#   6. Compiler and its flags.
#   __________________________________________________________________

# Debug Level. Options are:
# 0. NO debug at all NOR any run-time checks... be cautious!
# 1. Verbose (execution messages) AND run-time checks.
# 2. Level 1 plus intermediate scalar-valued results.
# 3. Level 2 plus intermediate array-valued results.

DEBUG_LEVEL = 3

# Flags recommended for release code:

CCFLAGS = -O2

# Flags recommended for debugging code:

CCFLAGS =

#   7. Archiver, its flags, and ranlib:
#   __________________________________________________________________

ARCH      = ar
ARCHFLAGS = cr

# If your system does not have "ranlib" then set: "RANLIB = echo":

RANLIB = echo

# But, if possible:

RANLIB = ranlib

#   8. Valgrind’s memcheck options:
#   __________________________________________________________________

MEMCHECK_OPTS = -v --tool=memcheck --leak-check=full --show-leak-kinds=all \
--track-origins=yes --freelist-vol=20000000

# Done!

#   __________________________________________________________________
#   __________________________________________________________________
#   __________________________________________________________________

#   MMTK-related.
#   __________________________________________________________________

SRC       = $(BASE)/src
INCLUDE   = $(BASE)/include
LIB       = $(BASE)/lib
MTK_LIB   = $(LIB)/libmtk.a
TESTS     = $(BASE)/tests
EXAMPLES  = $(BASE)/examples

#   Compiling-related.
#   __________________________________________________________________

CC = mex

CCFLAGS += -largeArrayDims -g

ifeq ($(PRECISION),DOUBLE)
  CCFLAGS += -DMTK_PRECISION_DOUBLE
else
  CCFLAGS += -DMTK_PRECISION_SINGLE
```

```
endif

CCFLAGS += -DMTK_DEBUG_LEVEL=$(DEBUG_LEVEL) -I$(INCLUDE)

# Only the GLPK is included because the other dependencies are coded in Fortran.

ifeq ($(ATL_OPT),ON)
  CCFLAGS  += -I$(GLPK_INC) $(ATLAS_INC)
else
  CCFLAGS  += -I$(GLPK_INC)
endif

#   Linking-related.
#   ___________________________________________________________________

NOOPT_LIBS  = $(LAPACK_LIB) $(GLPK_LIB) $(BLAS_LIB) -lstdc++ -lgfortran

OPT_LIBS = -L$(ATLAS_LIB) -latlas -llapack -lblas -lm -latlas \
-lstdc++ -lgfortran

LIBS = $(MTK_LIB)

ifeq ($(PLAT),OSX)
  LIBS  += -framework Accelerate $(GLPK_LIB)
else
  ifeq ($(ATL_OPT),ON)
    LIBS += $(OPT_LIBS)
  else
    LIBS += $(NOOPT_LIBS)
  endif
endif

#   Documentation-related.
#   ___________________________________________________________________

DOCGEN      = doxygen
DOCFILENAME = doc_config.dxcf
DOC         = $(HOME)/Dropbox/MMTK/doc
DOCFILE     = $(HOME)/Dropbox/MMTK/$(DOCFILENAME)
```

9.7 mexsrc/MMTKDiv1D.cc File Reference

MEX file for the Div1D class.

9.7.1 Detailed Description

Author

: Eduardo J. Sanchez (ejspeiro) - esanchez at mail dot sdsu dot edu

Definition in file MMTKDiv1D.cc.

9.8 MMTKDiv1D.cc

```
/*
Copyright (C) 2015, Computational Science Research Center, San Diego State
University. All rights reserved.

Redistribution and use in source and binary forms, with or without modification,
are permitted provided that the following conditions are met:

1. Modifications to source code should be reported to: esanchez@mail.sdsu.edu
and a copy of the modified files should be reported once modifications are
completed. Documentation related to said modifications should be included.

2. Redistributions of source code must be done through direct
downloads from the project's GitHub page: http://www.csrc.sdsu.edu/mtk
```

```

3. Redistributions of source code must retain the above copyright notice, this
list of conditions and the following disclaimer.

4. Redistributions in binary form must reproduce the above copyright notice,
this list of conditions and the following disclaimer in the documentation and/or
other materials provided with the distribution.

5. Usage of the binary form on proprietary applications shall require explicit
prior written permission from the the copyright holders.

6. Neither the name of the copyright holder nor the names of its contributors
may be used to endorse or promote products derived from this software without
specific prior written permission.

The copyright holders provide no reassurances that the source code provided does
not infringe any patent, copyright, or any other intellectual property rights of
third parties. The copyright holders disclaim any liability to any recipient for
claims brought against recipient by any third party for infringement of that
parties intellectual property rights.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
(INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
(INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
*/

#if __cplusplus == 201103L

#include <iostream>
#include <iomanip>
#include <cmath>
#include <cstring>

#include "mex.h"

#include "mtk.h"

using namespace std;

static int MTKDiv1D(int order_accuracy,
                    mtk::Real tau,
                    mtk::Real *outS,
                    mtk::Real *outM,
                    mtk::Real *outQ);

static int MTKDiv1DReturnAsDenseMatrix(int order_accuracy,
                                       mtk::Real west_bndy_x,
                                       mtk::Real east_bndy_x,
                                       int num_cells,
                                       mtk::Real tau,
                                       mtk::Real *outMatrix);

void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[]) {

  int order_accuracy;  // kk inside MTK.
  int num_cells;       // scaler to matrix size.

  mtk::Real *outMatrix;  // Where to place our output.
  mtk::Real *outS;       // output Stencil.
  mtk::Real *outQ;       // output Weights.
  mtk::Real *outM;       // output Mimetic Coefficients.

  mtk::Real west_bndy_x; // Start of the Interval.
  mtk::Real east_bndy_x; // End of the Interval.
  mtk::Real tau;         // Mimetic tolerance (optional).

//   if (nrhs >= 1 && nrhs <= 2) {
//
//     order_accuracy = mxGetScalar(prhs[0]);
//
//     if (nrhs == 1 && order_accuracy >= 8) {
//       std::cout << "Default mimetic threshold set as 1e-6."<<std::endl;
//       tau = 1e-6;
//     }
//     if (nrhs == 2 && order_accuracy < 8) {
//       std::cout << "Order of accuracy does not need mimetic threshold. If provided it will be ignored."<<std::endl;
//       tau = mxGetScalar(prhs[1]);
//     }
//     if(nrhs == 2 && order_accuracy >= 8) {
//       tau = mxGetScalar(prhs[1]);
//     }
//     if (nlhs != 3) {
//       mexErrMsgIdAndTxt("MyToolbox:arrayProduct:nlhs","3 outputs are needed: S, M, Q.");
//     }
//     if (nlhs == 3) {
//
//       // Stencil
//       plhs[0] = mxCreateDoubleMatrix((mwSize) 1,(mwSize) order_accuracy,mxREAL);
//
//       //Mimetic coefficients Rows
//       plhs[1] = mxCreateDoubleMatrix( (mwSize) (order_accuracy/2-1), (mwSize) (3*order_accuracy/ 2), mxREAL);
//
//       // Weights
//       plhs[2] = mxCreateDoubleMatrix((mwSize) 1,(mwSize) order_accuracy, mxREAL);
//
//       outS = mxGetPr(plhs[0]);
//       outM =mxGetPr(plhs[1]);
//       outQ = mxGetPr(plhs[2]);
//
//       mtk_1d_div(order_accuracy, tau, outS ,outM,outQ);
//     }
//   } else {
//
//     if (nrhs > 5) {
//       mexErrMsgIdAndTxt("MyToolbox:arrayProduct:nrhs", "At most 5 inputs are permited: (order, a, b, cells, tau).");
//     }
//
//     if (nrhs < 4) {
//       mexErrMsgIdAndTxt("MyToolbox:arrayProduct:nrhs", "At leats 4 inputs are required: (order, a, b, cells).");
//     }
//
//     if (nlhs != 1) {
//       mexErrMsgIdAndTxt("MyToolbox:arrayProduct:nlhs","One output is needed.");
//     }
//
//     order_accuracy = mxGetScalar(prhs[0]);
//
//     west_bndy_x = mxGetScalar(prhs[1]);
//
//     east_bndy_x = mxGetScalar(prhs[2]);
//
//     num_cells = mxGetScalar(prhs[3]);
//
//     if (num_cells < 3*order_accuracy - 1) {
//       mexErrMsgIdAndTxt("MyToolbox:arrayProduct:nrhs", "Number of cells too small for required order.");
//     }
//
//     if (order_accuracy >= 8) {
//       std::cout << "Order of accuracy too high. Mimetic threshold will be applied." << std::endl;
//     }
//
//     if (nrhs == 5 && order_accuracy >= 8) {
//       tau = mxGetScalar(prhs[4]);
//     }
//
//     if(nrhs == 5 && order_accuracy <8 ) {
//       std::cout << "Order of accuracy does not need mimetic threshold. If provided it will be ignored."<<std::endl;
//       tau = mxGetScalar(prhs[4]);
//     }
//
//     if(nrhs == 4 && order_accuracy >= 8) {
//       //Default threshold, used only if order >=8
//       std::cout << "Default mimetic threshold set as 1e-6."<<std::endl;
//       tau = 1.0e-6;
//     }
//
//     plhs[0] = mxCreateDoubleMatrix((mwSize) (num_cells + 2),
//                                    (mwSize) (num_cells + 1),
//                                    mxREAL);
//
```

```
//     outMatrix =  mxGetPr(plhs[0]);
//
//     MTKDiv1DReturnAsDenseMatrix(order_accuracy, west_bndy_x, east_bndy_x, num_cells, tau, outMatrix);
//   }
}

static int MTKDiv1D(int order_accuracy,
                    mtk::Real tau,
                    mtk::Real *outS,
                    mtk::Real *outM,
                    mtk::Real *outQ) {

//   mtk::MTK_1DDiv *div = new  mtk::MTK_1DDiv(order_accuracy,tau);
//
//   int Number_of_Extra_Rows=order_accuracy/2-1;
//
//   div = div->Construct1DDiv();
//
//   if (div == nullptr) {
//     return EXIT_FAILURE;
//   }
//
//   mtk::Real *ww;
//   mtk::Real *ss;
//   mtk::MTK_DenseMatrix *ee;
//   mtk::MTK_DenseMatrix *eeT;
//
//   ss = div->ReturnStencil();
//
//   ww = div->ReturnWeights();
//
//   for(auto ii = 0; ii < order_accuracy; ii++) {
//     outQ[ii]= ww[ii];
//     outS[ii]= ss[ii];
//   }
//
//   ee = div->ReturnMimeticCoefficients();
//   eeT= new mtk::MTK_DenseMatrix(3*order_accuracy/2,Number_of_Extra_Rows);
//   eeT->Transpose(*ee,*eeT);
//
//   for(auto ii=0;ii<3*order_accuracy/2*Number_of_Extra_Rows;ii++) {
//   out M[ii]=eeT->DenseMatrix_To_Real(*eeT)[ii];
//   }
//
//   delete [] ww;
//   delete [] ss;
//   delete [] ee;
}

static int MTKDiv1DReturnAsDenseMatrix(int order_accuracy,
                                       mtk::Real west_bndy_x,
                                       mtk::Real east_bndy_x,
                                       int num_cells,
                                       mtk::Real tau,
                                       mtk::Real *outMatrix) {

//   int nn = num_cells + 2;
//   int mm = num_cells + 1;
//
//   mtk::MTK_DenseMatrix *dd;
//   mtk::MTK_DenseMatrix *ddT;
//
//   mtk::MTK_1DDiv *div = new  mtk::MTK_1DDiv(order_accuracy,tau);
//
//   div = div->Construct1DDiv();
//
//   ddT = new mtk::MTK_DenseMatrix(mm,nn);
//
//   if (div == nullptr) {
//     std::cout << "Problem constructing the matrix... " << std::endl;
//     return EXIT_FAILURE;
//   }
//
//   dd = div->ReturnAsMatrix(num_cells, west_bndy_x, east_bndy_x);
//
//   dd->Transpose(*dd,*ddT);
//
//   for(auto ii = 0; ii < mm*nn; ii++) {
//     outMatrix[ii]=ddT->DenseMatrix_To_Real(*ddT)[ii];
//   }
//
```

```
//   delete [] ddT;
}

#endif
```

9.9 mexsrc/MMTKDiv1D.m File Reference

9.10 MMTKDiv1D.m

```
% MMTKDiv1D Create a mimetic one-dimensional divergence operator.
%
%   div = MMTKDiv1D(k, w, e, n, tau) Returns the divergence order k as a n + 2
%                                    by n + 1 matrix, in the interval (w,e) with
%                                    discretized using n cells with the given
%                                    mimetic threshold tau (optional).
%
%   [s, m, q] = MMTKDiv1D(k, tau)    Returns the divergence order k as the
%                                    interior stencil s, mimetic coefficients m,
%                                    mimetic weights q and mimetic threshold tau
%                                    (optional).
%
% For more information, see
% <a href="http://www.csrc.sdsu.edu/mimetic-book/">
% Mimetic Discretization Methods</a>.
%
```

9.11 README.md File Reference

9.12 README.md

```
# README File for the Mimetic Methods Toolkit (MTK)

By: **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu**
    __________________________________________________________________

## 1. Description

We define numerical methods that are based on discretizations preserving the
properties of their continuum counterparts to be **mimetic**.

The **Mimetic Methods Toolkit (MTK)** is a C++ library for mimetic numerical
methods. It is arranged as a set of classes for **mimetic quadratures**,
**mimetic interpolation**, and **mimetic discretization** methods for the
numerical solution of ordinary and partial differential equations.

This collection of **MATLAB Wrappers for the MTK (MMTK)** allows developers to
invoke the MTK from a MATLAB environment.

    __________________________________________________________________

## 2. Dependencies

This README assumes all of these dependencies are installed in the following
folder:

```
$(HOME)/Libraries/
```

In this version, the MTK optionally uses ATLAS-optimized BLAS and LAPACK
routines for the internal computation on some of the layers. However, ATLAS
requires both BLAS and LAPACK in order to create their optimized distributions.
Therefore, the following dependencies tree arises:

### For Linux and OS X:

1. MATLAB R2014a or greater - Available from: http://www.mathworks.com/

```

~~~
1. MTK - Available from: http://csrc.sdsu.edu/mtk/
    ___________________________________________________________________

## 3. Installation

The following instructions assume MATLAB R2014a or greater.

You have two options, either follow the instructions given, **expert install**,
or use the provided **patch file**, **naive install**.

### EXPERT PART 1. CONFIGURATION OF THE MEX COMPILER.

From your MATLAB prompt, type:

```
>> mex -setup C++
MEX configured to use 'g++' for C++ language compilation.
Warning: The MATLAB C and Fortran API has changed to support MATLAB
     variables with more than 2^32-1 elements. In the near future
     you will be required to update your code to utilize the
     new API. You can find more information about this at:

http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-
64-bit-api.html.
>>
```

### EXPERT PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.

The previous step creates the following file:

```
$(HOME)/.matlab/R2014a/mex_C++_glnxa64.xml
```

The purpose of this section is to configure the MATLAB R2014a Mex compiler so
that it can work with the latest C++ standard (C++11).

Please execute the following changes on the aforementioned file (line numbers
may differ):

Lines 26 and 27:

```
CMDLINE1="$CXX -std=c++11 -c $DEFINES $INCLUDE $CXXFLAGS $OPTIM $SRC -o $OBJ"
CMDLINE2="$LDXX -std=c++11 $LDFLAGS $LDTYPE $LINKOPTIM $LINKEXPORT $OBJS $CXXLIBS $LINKLIBS -o $EXE"
```

Line 33:

```
CXXFLAGS="-std=c++11 -ansi -fexceptions -fPIC -fno-omit-frame-pointer -pthread"
```

Line 35 and 26:

```
CXXOPTIMFLAGS="-std=c++11 -O -DNDEBUG"
CXXDEBUGFLAGS="-std=c++11 -g"
```

Line 38:

```
LDXX="gfortran"
```

Line 59:

```
CXXFLAGS="-std=c++11 -ansi -pthread"
```

NAIVE PART 1. CONFIGURATION OF THE MEX COMPILER.

From your MATLAB prompt, type:

```
>> mex -setup C++
MEX configured to use 'g++' for C++ language compilation.
Warning: The MATLAB C and Fortran API has changed to support MATLAB
~~~

```
     variables with more than 2^32-1 elements. In the near future
     you will be required to update your code to utilize the
     new API. You can find more information about this at:

http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-
64-bit-api.html.
>>
```

NAIVE PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.

```
cd $HOME/.matlab/R2014a
chmod +w mex_C++_glnxa64.xml
patch < mex_C++_glnxa64.patch
chmod -w mex_C++_glnxa64.xml
```
Exit terminal, and restart MATLAB. You can use C++11 to create MEX files now!

PART 3: CONFIGURATION OF THE MAKEFILE.

The following steps are required the build and test the MTK. Please use the
accompanying `Makefile.inc` file, which should provide a solid template to
start with. The following command provides help on the options for make:

```
$ make help
-----
Makefile for the MMTK.

Options are:
- all: builds he library, the tests, and examples.

- gendoc: generates the documentation for the library.

- clean: cleans ALL the generated files.
-----
```

PART 4. BUILD THE MMTK.

From your shell, at the base folder of the MMTK, just type:

```
make
```

If successful you'll read:

```
----- Library created! Check in /home/ejspeiro/Dropbox/MTK/lib
```
    ____

4. Frequently Asked Questions

Q: Why haven't you guys implemented GBS to build the library?
A: I'm on it as we speak! ;)

Q: When will the other flavors be ready?
A: Soon! I'm working on getting help on developing those.

Q: Is there any main reference when it comes to the theory on Mimetic Methods?
A: Yes! Check: http://www.csrc.sdsu.edu/mimetic-book

Q: Do I need to generate the documentation myself?
A: You can if you want to... but if you DO NOT want to, just go to our website.
    ____

5. Contact, Support, and Credits

The MTK is developed by researchers and adjuncts to the
[Computational Science Research Center (CSRC)](http://www.csrc.sdsu.edu/)
at [San Diego State University (SDSU)](http://www.sdsu.edu/).

Developers are members of:

1. Mimetic Numerical Methods Research and Development Group.
2. Computational Geoscience Research and Development Group.
3. Ocean Modeling Research and Development Group.

```
Currently the developers are:

- **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu** - @ejspeiro
- Jose E. Castillo, Ph.D. - jcastillo at mail dot sdsu dot edu
- Guillermo F. Miranda, Ph.D. - unigrav at hotmail dot com
- Christopher P. Paolini, Ph.D. - paolini at engineering dot sdsu dot edu
- Angel Boada.
- Johnny Corbino.
- Raul Vargas-Navarro.

Finally, please feel free to contact me with suggestions or corrections:

**Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu** - @ejspeiro

Thanks and happy coding!
```

Index