MMTK: MATLAB Wrappers for the Mimetic Methods Toolkit

Generated by Doxygen 1.8.6

Mon Sep 14 2015 16:52:43

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.

Finally, a collection of grid visualization routines that is compatible with these wrappers can be found in:

MATLAB Visualizers for Uniform Staggered Grids

---

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

- close all
- clc
- west = 0.0
- east = 1.0
- kk
- ss num_cells = 3∗kk - 1
- div = MMTKDiv1D(kk, west, east, num_cells)

9.1.1 Variable Documentation

9.1.1.1 clear all

Definition at line 21 of file Div1D.m.

9.1.1.2 clc

Definition at line 23 of file Div1D.m.

9.1.1.3 div = MMTKDiv1D(kk, west, east, num_cells)

Definition at line 33 of file Div1D.m.

9.1.1.4 east = 1.0

Definition at line 26 of file Div1D.m.

9.1.1.5 div kk

Initial value:

```
= 2
[ss, mb, qq] = MMTKDiv1D(kk)
```

Definition at line 28 of file Div1D.m.

9.1.1.6 ss mb qq num_cells = 3∗kk - 1

Definition at line 32 of file Div1D.m.

9.1.1.7 west = 0.0

Definition at line 25 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
00014 % Website: http://www-rohan.sdsu.edu/~sanche94/
00015 % 2015-09-10 14:18:35
00016
00017 %------------- BEGIN CODE --------------
00018
00019 addpath('../mexsrc');
00020
00021 close all;
00022 clear all;
00023 clc;
00024
00025 west = 0.0;
00026 east = 1.0;
00027
00028 kk = 2
00029 [ss, mb, qq] = MMTKDiv1D(kk);
00030 ss
00031
00032 num_cells = 3*kk - 1;
00033 div = MMTKDiv1D(kk, west, east, num_cells);
00034 div
00035
00036 kk = 4
00037 [ss, mb, qq] = MMTKDiv1D(kk);
00038 ss
00039 mb
00040 qq
00041
00042 num_cells = 3*kk - 1;
00043 div = MMTKDiv1D(kk, west, east, num_cells);
00044 div
00045
00046 %------------- 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

```
00001
00013 /*
00014 Copyright (C) 2015, Computational Science Research Center, San Diego State
00015 University. All rights reserved.
00016
00017 Redistribution and use in source and binary forms, with or without modification,
00018 are permitted provided that the following conditions are met:
00019
00020 1. Modifications to source code should be reported to: esanchez@mail.sdsu.edu
00021 and a copy of the modified files should be reported once modifications are
00022 completed. Documentation related to said modifications should be included.
00023
00024 2. Redistributions of source code must be done through direct
00025 downloads from the project's GitHub page: http://www.csrc.sdsu.edu/mtk
00026
00027 3. Redistributions of source code must retain the above copyright notice, this
00028 list of conditions and the following disclaimer.
00029
00030 4. Redistributions in binary form must reproduce the above copyright notice,
00031 this list of conditions and the following disclaimer in the documentation and/or
00032 other materials provided with the distribution.
00033
00034 5. Usage of the binary form on proprietary applications shall require explicit
00035 prior written permission from the the copyright holders.
00036
00037 6. Neither the name of the copyright holder nor the names of its contributors
00038 may be used to endorse or promote products derived from this software without
00039 specific prior written permission.
00040
00041 The copyright holders provide no reassurances that the source code provided does
00042 not infringe any patent, copyright, or any other intellectual property rights of
00043 third parties. The copyright holders disclaim any liability to any recipient for
00044 claims brought against recipient by any third party for infringement of that
00045 parties intellectual property rights.
00046
00047 THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
00048 ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
00049 WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00050 DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
00051 ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
00052 (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
00053 LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
00054 ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
00055 (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
00056 SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00057 */
```

9.5 Makefile.inc File Reference

9.6 Makefile.inc

```
00001 # Makefile setup file for MMTK.
00002
00003 SHELL := /bin/bash
00004
00005 # Please set the following variables up:
00006
00007 #   2. Absolute path to base directory of the MTK... where is the MTK?
00008 #   ____________________________________________________________________
00009
00010 BASE = /home/ejspeiro/Dropbox/MTK
00011
00012 #   2. The machine (platform) identifier and required precision.
00013 #   ____________________________________________________________________
00014
00015 # Options are:
00016 # - LINUX: A LINUX box installation.
00017 # - OSX: Uses OS X optimized solvers.
00018
00019 PLAT = LINUX
00020
00021 # Options are:
00022 # - SINGLE: Use 4 B floating point numbers.
00023 # - DOUBLE: Use 8 B floating point numbers.
00024
00025 PRECISION = DOUBLE
00026
00027 #   3. Optimized solvers and operations by means of ATLAS in Linux?
00028 #   ____________________________________________________________________
00029
00030 # If you have selected OSX in step 1, then you don't need to worry about this.
00031
00032 # Options are ON xor OFF:
00033
00034 ATL_OPT = OFF
00035
00036 #   4. Paths to dependencies (header files for compiling).
00037 #   ____________________________________________________________________
00038
00039 # GLPK include path (soon to go):
00040
00041 GLPK_INC = $(HOME)/Libraries/glpk-4.55/include
00042
00043 # Linux: If ATLAS optimization is ON, users should only provide the path to
00044 # ATLAS:
00045
00046 ATLAS_INC = $(HOME)/Libraries/ATLAS_3.8.4-CORE/include
00047
00048 # OS X: Do nothing.
00049
00050 #   5. Paths to dependencies (archive files for (static) linking).
00051 #   ____________________________________________________________________
00052
00053 # GLPK linking path (soon to go):
00054
00055 GLPK_LIB = $(HOME)/Libraries/glpk-4.55/lib/libglpk.a
00056
00057 # If optimization is OFF, then provide the paths for:
00058
00059 BLAS_LIB = $(HOME)/Libraries/BLAS/libblas.a
00060 LAPACK_LIB = $(HOME)/Libraries/lapack-3.4.1/liblapack.a
00061
00062 # WARNING: Vendor libraries should be used whenever they are available.
00063
00064 # However, if optimization is ON, please provide the path the ATLAS' archive:
00065
00066 ATLAS_LIB   = $(HOME)/Libraries/ATLAS_3.8.4-CORE/ATLAS_3.8.4-BUILD-Citadel/lib
00067
00068 #   6. Compiler and its flags.
00069 #   ____________________________________________________________________
00070
00071 # Debug Level. Options are:
00072 # 0. NO debug at all NOR any run-time checks... be cautious!
00073 # 1. Verbose (execution messages) AND run-time checks.
```

```
00074 # 2. Level 1 plus intermediate scalar-valued results.
00075 # 3. Level 2 plus intermediate array-valued results.
00076
00077 DEBUG_LEVEL = 3
00078
00079 # Flags recommended for release code:
00080
00081 CCFLAGS = -O2
00082
00083 # Flags recommended for debugging code:
00084
00085 CCFLAGS =
00086
00087 #   7. Archiver, its flags, and ranlib:
00088 #   ____________________________________________________________________
00089
00090 ARCH      = ar
00091 ARCHFLAGS = cr
00092
00093 # If your system does not have "ranlib" then set: "RANLIB = echo":
00094
00095 RANLIB = echo
00096
00097 # But, if possible:
00098
00099 RANLIB = ranlib
00100
00101 #   8. Valgrind's memcheck options:
00102 #   ____________________________________________________________________
00103
00104 MEMCHECK_OPTS = -v --tool=memcheck --leak-check=full --show-leak-kinds=all \
00105 --track-origins=yes --freelist-vol=20000000
00106
00107 # Done!
00108
00109 #   ____________________________________________________________________
00110 #   ____________________________________________________________________
00111 #   ____________________________________________________________________
00112
00113 #   MMTK-related.
00114 #   ____________________________________________________________________
00115
00116 SRC       = $(BASE)/src
00117 INCLUDE   = $(BASE)/include
00118 LIB       = $(BASE)/lib
00119 MTK_LIB   = $(LIB)/libmtk.a
00120 TESTS     = $(BASE)/tests
00121 EXAMPLES  = $(BASE)/examples
00122
00123 #   Compiling-related.
00124 #   ____________________________________________________________________
00125
00126 CC = mex
00127
00128 CCFLAGS += -largeArrayDims -g
00129
00130 ifeq ($(PRECISION),DOUBLE)
00131   CCFLAGS += -DMTK_PRECISION_DOUBLE
00132 else
00133   CCFLAGS += -DMTK_PRECISION_SINGLE
00134 endif
00135
00136 CCFLAGS += -DMTK_DEBUG_LEVEL=$(DEBUG_LEVEL) -I$(INCLUDE)
00137
00138 # Only the GLPK is included because the other dependencies are coded in Fortran.
00139
00140 ifeq ($(ATL_OPT),ON)
00141   CCFLAGS  += -I$(GLPK_INC) $(ATLAS_INC)
00142 else
00143   CCFLAGS  += -I$(GLPK_INC)
00144 endif
00145
00146 #   Linking-related.
00147 #   ____________________________________________________________________
00148
00149 NOOPT_LIBS  = $(LAPACK_LIB) $(GLPK_LIB) $(BLAS_LIB) -lstdc++ -lgfortran
00150
00151 OPT_LIBS = -L$(ATLAS_LIB) -latlas -llapack -lblas -lm -latlas \
00152 -lstdc++ -lgfortran
00153
00154 LIBS = $(MTK_LIB)
```

```
00155
00156 ifeq ($(PLAT),OSX)
00157   LIBS  += -framework Accelerate $(GLPK_LIB)
00158 else
00159   ifeq ($(ATL_OPT),ON)
00160     LIBS += $(OPT_LIBS)
00161   else
00162     LIBS += $(NOOPT_LIBS)
00163   endif
00164 endif
00165
00166 #   Documentation-related.
00167 #   ____________________________________________________________
00168
00169 DOCGEN      = doxygen
00170 DOCFILENAME = doc_config.dxcf
00171 DOC         = $(HOME)/Dropbox/MMTK/doc
00172 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

```
00001
00008 /*
00009 Copyright (C) 2015, Computational Science Research Center, San Diego State
00010 University. All rights reserved.
00011
00012 Redistribution and use in source and binary forms, with or without modification,
00013 are permitted provided that the following conditions are met:
00014
00015 1. Modifications to source code should be reported to: esanchez@mail.sdsu.edu
00016 and a copy of the modified files should be reported once modifications are
00017 completed. Documentation related to said modifications should be included.
00018
00019 2. Redistributions of source code must be done through direct
00020 downloads from the project's GitHub page: http://www.csrc.sdsu.edu/mtk
00021
00022 3. Redistributions of source code must retain the above copyright notice, this
00023 list of conditions and the following disclaimer.
00024
00025 4. Redistributions in binary form must reproduce the above copyright notice,
00026 this list of conditions and the following disclaimer in the documentation and/or
00027 other materials provided with the distribution.
00028
00029 5. Usage of the binary form on proprietary applications shall require explicit
00030 prior written permission from the the copyright holders.
00031
00032 6. Neither the name of the copyright holder nor the names of its contributors
00033 may be used to endorse or promote products derived from this software without
00034 specific prior written permission.
00035
00036 The copyright holders provide no reassurances that the source code provided does
00037 not infringe any patent, copyright, or any other intellectual property rights of
00038 third parties. The copyright holders disclaim any liability to any recipient for
00039 claims brought against recipient by any third party for infringement of that
00040 parties intellectual property rights.
00041
```

```
00042 THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND
00043 ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED
00044 WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE
00045 DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR
00046 ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES
00047 (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES;
00048 LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON
00049 ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT
00050 (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS
00051 SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
00052 */
00053
00054 #if __cplusplus == 201103L
00055
00056 #include <iostream>
00057 #include <iomanip>
00058 #include <cmath>
00059 #include <cstring>
00060
00061 #include "mex.h"
00062
00063 #include "mtk.h"
00064
00065 using namespace std;
00066
00067 static int MTKDiv1D(int order_accuracy,
00068                     mtk::Real tau,
00069                     mtk::Real *outS,
00070                     mtk::Real *outM,
00071                     mtk::Real *outQ);
00072
00073 static int MTKDiv1DReturnAsDenseMatrix(int order_accuracy,
00074                                        mtk::Real west_bndy_x,
00075                                        mtk::Real east_bndy_x,
00076                                        int num_cells,
00077                                        mtk::Real tau,
00078                                        mtk::Real *outMatrix);
00079
00080 void mexFunction(int nlhs, mxArray *plhs[], int nrhs, const mxArray *prhs[]) {
00081
00082   int order_accuracy;  // kk inside MTK.
00083   int num_cells;       // scaler to matrix size.
00084
00085   mtk::Real *outMatrix;  // Where to place our output.
00086   mtk::Real *outS;       // Output Stencil.
00087   mtk::Real *outQ;       // Output Weights.
00088   mtk::Real *outM;       // Output Mimetic Coefficients.
00089
00090   mtk::Real west_bndy_x; // Start of the Interval.
00091   mtk::Real east_bndy_x; // End of the Interval.
00092   mtk::Real tau;         // Mimetic tolerance (optional).
00093
00094   if (nrhs >= 1 && nrhs <= 2) {
00095
00096     order_accuracy = mxGetScalar(prhs[0]);
00097
00098     if (nrhs == 1 && order_accuracy >= mtk::kCriticalOrderAccuracyDiv) {
00099       tau = mtk::kDefaultMimeticThreshold;
00100       std::cout << "Default mimetic threshold set as" << tau << std::endl;
00101     }
00102     if (nrhs == 2 && order_accuracy < mtk::kCriticalOrderAccuracyDiv) {
00103       tau = mxGetScalar(prhs[1]);
00104       std::cout << "Order of accuracy does not need mimetic threshold. If provided it will be ignored." <<
      std::endl;
00105     }
00106     if(nrhs == 2 && order_accuracy >= mtk::kCriticalOrderAccuracyDiv) {
00107       tau = mxGetScalar(prhs[1]);
00108     }
00109     if (nlhs != 3) {
00110       mexErrMsgIdAndTxt("MMTK:Div1D:nlhs","3 outputs are needed: S, M, Q.");
00111     }
00112     if (nlhs == 3) {
00113
00114       // Stencil.
00115       plhs[0] = mxCreateDoubleMatrix((mwSize) 1,
00116                                      (mwSize) order_accuracy,
00117                                      mxREAL);
00118
00119       // Mimetic coefficients rows.
00120       plhs[1] = mxCreateDoubleMatrix((mwSize) (order_accuracy/2 - 1),
00121                                      (mwSize) (3*order_accuracy/2),
```

```
00122                                     mxREAL);
00123
00124       // Weights.
00125       plhs[2] = mxCreateDoubleMatrix((mwSize) 1,
00126                                      (mwSize) order_accuracy,
00127                                      mxREAL);
00128
00129       outS = mxGetPr(plhs[0]);
00130       outQ = mxGetPr(plhs[2]);
00131       outM = mxGetPr(plhs[1]);
00132
00133       MTKDiv1D(order_accuracy, tau, outS, outQ, outM);
00134     }
00135   } else {
00136
00137     if (nrhs > 5) {
00138       mexErrMsgIdAndTxt("MMTK:Div1D:nrhs",
00139                         "At most 5 inputs are permitted: (order, west, east, cells, tau).");
00140     }
00141     if (nrhs < 4) {
00142       mexErrMsgIdAndTxt("MMTK:Div1D:nrhs",
00143                         "At leats 4 inputs are required: (order, west, east, cells).");
00144     }
00145     if (nlhs != 1) {
00146       mexErrMsgIdAndTxt("MMTK:Div1D:nlhs","Only one output is needed.");
00147     }
00148
00149     order_accuracy = mxGetScalar(prhs[0]);
00150
00151     west_bndy_x = mxGetScalar(prhs[1]);
00152
00153     east_bndy_x = mxGetScalar(prhs[2]);
00154
00155     num_cells = mxGetScalar(prhs[3]);
00156
00157     if (num_cells < 3*order_accuracy - 1) {
00158       mexErrMsgIdAndTxt("MMTK:Div1D:nrhs", "Number of cells too small for required order.");
00159     }
00160     if (order_accuracy >= mtk::kCriticalOrderAccuracyDiv) {
00161       std::cout << "Order of accuracy too high. Mimetic threshold will be applied." << std::endl;
00162     }
00163     if (nrhs == 5 && order_accuracy >= mtk::kCriticalOrderAccuracyDiv) {
00164       tau = mxGetScalar(prhs[4]);
00165     }
00166     if(nrhs == 5 && order_accuracy < mtk::kCriticalOrderAccuracyDiv) {
00167       std::cout << "Order of accuracy does not need mimetic threshold. If provided it will be ignored."<<
      std::endl;
00168       tau = mxGetScalar(prhs[4]);
00169     }
00170     if(nrhs == 4 && order_accuracy >= mtk::kCriticalOrderAccuracyDiv) {
00171       //Default threshold, used only if order >= mtk::kCriticalOrderAccuracyDiv.
00172       tau = mtk::kDefaultMimeticThreshold;
00173       std::cout << "Default mimetic threshold set as" << tau << std::endl;
00174     }
00175
00176     plhs[0] = mxCreateDoubleMatrix((mwSize) (num_cells + 2),
00177                                    (mwSize) (num_cells + 1),
00178                                    mxREAL);
00179
00180     outMatrix =  mxGetPr(plhs[0]);
00181
00182     MTKDiv1DReturnAsDenseMatrix(order_accuracy,
00183                                 west_bndy_x, east_bndy_x, num_cells,
00184                                 tau, outMatrix);
00185   }
00186 }
00187
00188 static int MTKDiv1D(int order_accuracy,
00189                     mtk::Real tau,
00190                     mtk::Real *outS,
00191                     mtk::Real *outQ,
00192                     mtk::Real *outM) {
00193
00194   mtk::Div1D div;
00195
00196   bool info = div.ConstructDiv1D(order_accuracy, tau);
00197
00198   if (!info) {
00199     std::cerr << "Mimetic div could not be built." << std::endl;
00200   }
00201
```

```
00202   int number_of_extra_rows = order_accuracy/2 - 1;
00203
00204   if (order_accuracy > mtk::kDefaultOrderAccuracy) {
00205     for(auto ii = 0; ii < order_accuracy; ++ii) {
00206       outS[ii] = div.coeffs_interior()[ii];
00207       outQ[ii] = div.weights_cbs()[ii];
00208     }
00209     mtk::DenseMatrix ee(div.mim_bndy());
00210
00211     ee.OrderColMajor();
00212
00213     for(auto ii = 0; ii < (3*order_accuracy/2)*number_of_extra_rows; ++ii) {
00214       outM[ii] = ee.data()[ii];
00215     }
00216   } else {
00217     for(auto ii = 0; ii < order_accuracy; ++ii) {
00218       outS[ii] = div.coeffs_interior()[ii];
00219     }
00220   }
00221 }
00222
00223 static int MTKDiv1DReturnAsDenseMatrix(int order_accuracy,
00224                                        mtk::Real west_bndy_x,
00225                                        mtk::Real east_bndy_x,
00226                                        int num_cells,
00227                                        mtk::Real tau,
00228                                        mtk::Real *outMatrix) {
00229
00230   int nn = num_cells + 2;
00231   int mm = num_cells + 1;
00232
00233   mtk::Div1D div;
00234
00235   bool info = div.ConstructDiv1D(order_accuracy, tau);
00236
00237   if (!info) {
00238     std::cerr << "Mimetic div could not be built." << std::endl;
00239   }
00240
00241   mtk::UniStgGrid1D grid(west_bndy_x, east_bndy_x, num_cells);
00242
00243   mtk::DenseMatrix divm(div.ReturnAsDenseMatrix(grid));
00244
00245   divm.OrderColMajor();
00246
00247   for(auto ii = 0; ii < mm*nn; ++ii) {
00248     outMatrix[ii] = divm.data()[ii];
00249   }
00250 }
00251
00252 #endif
```

9.9 mexsrc/MMTKDiv1D.m File Reference

9.10 MMTKDiv1D.m

```
00001 % MMTKDiv1D Create a mimetic one-dimensional divergence operator.
00002 %
00003 %   div = MMTKDiv1D(k, w, e, n, tau) Returns the divergence order k as a n + 2
00004 %                                    by n + 1 matrix, in the interval (w,e) with
00005 %                                    discretized using n cells with the given
00006 %                                    mimetic threshold tau (optional).
00007 %
00008 %   [s, m, q] = MMTKDiv1D(k, tau)    Returns the divergence order k as the
00009 %                                    interior stencil s, mimetic coefficients m,
00010 %                                    mimetic weights q and mimetic threshold tau
00011 %                                    (optional).
00012 %
00013 % For more information, see
00014 % <a href="http://www.csrc.sdsu.edu/mimetic-book/">
00015 % Mimetic Discretization Methods</a>.
00016 %
```

9.11 README.md File Reference

9.12 README.md

```
00001 # MATLAB wrappers for the Mimetic Methods Toolkit (MMTK)
00002
00003 By: **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu**
00004     ____________________________________________________________
00005
00006 ## 1. Description
00007
00008 We define numerical methods that are based on discretizations preserving the
00009 properties of their continuum counterparts to be **mimetic**.
00010
00011 The **Mimetic Methods Toolkit (MTK)** is a C++ library for mimetic numerical
00012 methods. It is arranged as a set of classes for **mimetic quadratures**,
00013 **mimetic interpolation**, and **mimetic discretization** methods for the
00014 numerical solution of ordinary and partial differential equations.
00015
00016 This collection of **MATLAB Wrappers for the MTK (MMTK)** allows developers to
00017 invoke the MTK from a MATLAB environment.
00018
00019 Finally, a collection of grid visualization routines that is compatible with
00020 these wrappers can be found in:
00021
00022 [MATLAB Visualizers for Uniform Staggered Grids](https://github.com/ejspeiro/UniStgGrid-Visualizers)
00023     ____________________________________________________________
00024
00025 ## 2. Dependencies
00026
00027 This README assumes all of these dependencies are installed in the following
00028 folder:
00029
00030 ```
00031 $(HOME)/Libraries/
00032 ```
00033
00034 In this version, the MTK optionally uses ATLAS-optimized BLAS and LAPACK
00035 routines for the internal computation on some of the layers. However, ATLAS
00036 requires both BLAS and LAPACK in order to create their optimized distributions.
00037 Therefore, the following dependencies tree arises:
00038
00039 ### For Linux and OS X:
00040
00041 1. MATLAB R2014a or greater - Available from: http://www.mathworks.com/
00042
00043 1. MTK - Available from: http://csrc.sdsu.edu/mtk/
00044
00045     ____________________________________________________________
00046
00047 ## 3. Installation
00048
00049 The following instructions assume MATLAB R2014a or greater.
00050
00051 You have two options, either follow the instructions given, **expert install**,
00052 or use the provided **patch file**, **naive install**.
00053
00054 ### EXPERT PART 1. CONFIGURATION OF THE MEX COMPILER.
00055
00056 From your MATLAB prompt, type:
00057
00058 ```
00059 >> mex -setup C++
00060 MEX configured to use 'g++' for C++ language compilation.
00061 Warning: The MATLAB C and Fortran API has changed to support MATLAB
00062      variables with more than 2^32-1 elements. In the near future
00063      you will be required to update your code to utilize the
00064      new API. You can find more information about this at:
00065
00066 http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-
00067 64-bit-api.html.
00068 >>
00069 ```
00070
00071 ### EXPERT PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.
00072
00073 The previous step creates the following file:
```

```
00074
00075 ```
00076 $(HOME)/.matlab/R2014a/mex_C++_glnxa64.xml
00077 ```
00078
00079 The purpose of this section is to configure the MATLAB R2014a Mex compiler so
00080 that it can work with the latest C++ standard (C++11).
00081
00082 Please execute the following changes on the aforementioned file (line numbers
00083 may differ):
00084
00085 Lines 26 and 27:
00086
00087 ```
00088 CMDLINE1="$CXX -std=c++11 -c $DEFINES $INCLUDE $CXXFLAGS $OPTIM $SRC -o $OBJ"
00089 CMDLINE2="$LDXX -std=c++11 $LDFLAGS $LDTYPE $LINKOPTIM $LINKEXPORT $OBJS $CXXLIBS $LINKLIBS -o $EXE"
00090 ```
00091
00092 Line 33:
00093
00094 ```
00095 CXXFLAGS="-std=c++11 -ansi -fexceptions -fPIC -fno-omit-frame-pointer -pthread"
00096 ```
00097
00098 Line 35 and 26:
00099
00100 ```
00101 CXXOPTIMFLAGS="-std=c++11 -O -DNDEBUG"
00102 CXXDEBUGFLAGS="-std=c++11 -g"
00103 ```
00104
00105 Line 38:
00106
00107 ```
00108 LDXX="gfortran"
00109 ```
00110
00111 Line 59:
00112
00113 ```
00114 CXXFLAGS="-std=c++11 -ansi -pthread"
00115 ```
00116
00117 ### NAIVE PART 1. CONFIGURATION OF THE MEX COMPILER.
00118
00119 From your MATLAB prompt, type:
00120
00121 ```
00122 >> mex -setup C++
00123 MEX configured to use 'g++' for C++ language compilation.
00124 Warning: The MATLAB C and Fortran API has changed to support MATLAB
00125      variables with more than 2^32-1 elements. In the near future
00126      you will be required to update your code to utilize the
00127      new API. You can find more information about this at:
00128
00129 http://www.mathworks.com/help/matlab/matlab_external/upgrading-mex-files-to-use-
00130 64-bit-api.html.
00131 >>
00132 ```
00133
00134 ### NAIVE PART 2. CONFIGURATION OF THE MEX COMPILER SETUP FILE.
00135
00136 ```
00137 cd $HOME/.matlab/R2014a
00138 chmod +w mex_C++_glnxa64.xml
00139 patch < mex_C++_glnxa64.patch
00140 chmod -w mex_C++_glnxa64.xml
00141 ```
00142 Exit terminal, and restart MATLAB. You can use C++11 to create MEX files now!
00143
00144 ### PART 3: CONFIGURATION OF THE MAKEFILE.
00145
00146 The following steps are required the build and test the MTK. Please use the
00147 accompanying `Makefile.inc` file, which should provide a solid template to
00148 start with. The following command provides help on the options for make:
00149
00150 ```
00151 $ make help
00152 -----
00153 Makefile for the MMTK.
00154
```

```
00155 Options are:
00156 - all: builds he library, the tests, and examples.
00157
00158 - gendoc: generates the documentation for the library.
00159
00160 - clean: cleans ALL the generated files.
00161 -----
00162 ```
00163
00164 ### PART 4. BUILD THE MMTK.
00165
00166 From your shell, at the base folder of the MMTK, just type:
00167
00168 ```
00169 make
00170 ```
00171
00172 If successful you'll read:
00173
00174 ```
00175 ----- Library created! Check in /home/ejspeiro/Dropbox/MTK/lib
00176 ```
00177     ____________________________________________________________________
00178
00179 ## 4. Frequently Asked Questions
00180
00181 Q: Why haven't you guys implemented GBS to build the library?
00182 A: I'm on it as we speak! ;)
00183
00184 Q: When will the other flavors be ready?
00185 A: Soon! I'm working on getting help on developing those.
00186
00187 Q: Is there any main reference when it comes to the theory on Mimetic Methods?
00188 A: Yes! Check: http://www.csrc.sdsu.edu/mimetic-book
00189
00190 Q: Do I need to generate the documentation myself?
00191 A: You can if you want to... but if you DO NOT want to, just go to our website.
00192     ____________________________________________________________________
00193
00194 ## 5. Contact, Support, and Credits
00195
00196 The MTK is developed by researchers and adjuncts to the
00197 [Computational Science Research Center (CSRC)](http://www.csrc.sdsu.edu/)
00198 at [San Diego State University (SDSU)](http://www.sdsu.edu/).
00199
00200 Developers are members of:
00201
00202 1. Mimetic Numerical Methods Research and Development Group.
00203 2. Computational Geoscience Research and Development Group.
00204 3. Ocean Modeling Research and Development Group.
00205
00206 Currently the developers are:
00207
00208 - **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu** - @ejspeiro
00209 - Jose E. Castillo, Ph.D. - jcastillo at mail dot sdsu dot edu
00210 - Guillermo F. Miranda, Ph.D. - unigrav at hotmail dot com
00211 - Christopher P. Paolini, Ph.D. - paolini at engineering dot sdsu dot edu
00212 - Angel Boada.
00213 - Johnny Corbino.
00214 - Raul Vargas-Navarro.
00215
00216 Finally, please feel free to contact me with suggestions or corrections:
00217
00218 **Eduardo J. Sanchez, Ph.D. - esanchez at mail dot sdsu dot edu** - @ejspeiro
00219
00220 Thanks and happy coding!
```

Index