MATLAS TOOLBOX:
A MEX GATEWAY FOR UTILIZING LASTOOLS READ AND WRITE FUNCTIONS IN MATLAB

CONTENTS

1. INTRODUCTION

1.1 INITIAL DESCRIPTION

Authors

The authors welcome all comments for improving the toolbox and its documentation.

Paula Litkey has written most of the code of *MATLAS* and documentation for using the toolbox in Linux.

Eetu Puttonen has contributed to the *MATLAS* coding and testing. He has written the documentation for compiling and using the toolbox in Windows.

Mariana Batista Campos has contributed to update *MATLAS* and documentation from version v.014 to v.015

Acknowledgements

Martin Isenburg, for creating, developing, and maintaining the LAStools and LASlib, rapidlasso.com

Joaquim Luis, for providing a test mex code, header files for .dll compilation, and MATLAB compatible dynamic libraries to start testing with in the first place. http://w3.ualg.pt/~jluis/

LAStools mailing list
http://groups.google.com/group/lastools

Funding

The work of authors has been partly supported from the following projects: Academy of Finland project no.316096/320075, "Upscaling of carbon intake and water balance models of individual trees to wider areas with short interval laser scanning time series" and from the Strategic Research Council at the Academy of Finland project no. 293389/314312, "Competence Based Growth Through Integrated Disruptive Technologies of 3D Digitalization, Robotics, Geospatial Information and Image Processing/Computing - Point Cloud Ecosystem (COMBAT)

1.2 FUNCTION OVERVIEW OF FGI MATLAS

MATLAS is a MATLAB mex gateway for using LasTools read and write functionalities from MATLAB. MATLAS consists of a reader function las2mat, a writer function MATLAS and collection of helper functions and definitions (mex_lasz_fun_fgi.cpp and mex_lasz_fun_fgi.hpp). MATLAS copies the data fields from LasTools point and header objects into a MATLAB structure. The point-wise data are stored in MATLAB as a structure of vectors, not as in the LasTools, where each point is an object (the more precise MATLAB presentation would be a vector of structures). The structure of vectors in MATLAB is both faster and suits our purposes better. There are 13 fields to the point data that are always copied / present and 3 fields that are present if the data field is in use in the LasTools point object. The persistent fields are present in each structure of points, even if they hold no data. They are: "x", "y", "z", "intensity", "return_number", "number_of_returns", "scan_direction_flag", "edge_of_flight_line", "classification", "scan_angle_rank", "user_data and point_source_ID". The fields "gps_time", "rgb" and "attributes" only appear if they are found in the data. The size of the fields "rgb" and "attributes" is column times row (m x n). The "rgb" field is of size N x 3, where N is the number of points. This is important to note especially when constructing a structure in MATLAB for writing into a las file. The attributes appear both in the header and in the point data. In the header, field "attributes" is a substructure that has fields "name", "type", "description"," scale" and "offset" for each extra attribute. In the data, the values are saved in a column x row matrix in the same order they are listed in the header. The attribute values are already scaled using the parameters in the header.

2. INSTRUCTIONS FOR COMPILING

2.1 MATLAS IN LINUX

These instructions have been tested with Ubuntu 18.04 and LASTools v 201124.

First, extract the MATLAS package, path denotes the path to matlas_tools.

Download the LASTools library (in the example below, it is extracted to the MATLAS folder). If you have LASTools already, you still need to compile the shared library (edit the Makefile in /lastools/LASlib/src directory to build .so) and fix the paths in the mex commands. Make sure your LASlib version is not older than 131025.

Copy and rename the /path/matlas_tools/lastools/LASlib/src/Makefile to /path/matlas_tools/lastools/LASlib/src/Makefile_orig

Edit the Makefile so that the COPTS line at the top has the "-fPIC" as the example Makefile in this package also edit lines all: add liblas.so.
NOTE: if you copy the Makefile from MATLAS to path/matlas_tools/lastools/LASlib/src/ it is important to check that the list of files (with .o ending) is the same as in Makefile_orig because there might be changes between the LASlib versions!

In the terminal go to /path/matlas_tools/lastools/LASlib/src and run make (type make and hit enter).

In Matlab go to directory /path/matlas_tools/ if you installed the lastools in the /path/matlas_tools/ folder do option A, otherwise do option B.

A. Type compile_matlas_tools, if there are no errors the reader is ready for use.

B. Compile the fgi_las2mat.cpp and fgi_MATLAS with the commands below:
NOTE! You need to replace 'path' with the directory path that you are using!
mex las2mat.cpp /path/matlas_tools/lastools/LASlib/lib/liblas.a
-I/path/matlas_tools/lastools/LASlib/inc
-I/path/matlas_tools/lastools/src
-I/path/matlas_tools/lastools/LASzip/inc
-I/path/matlas_tools/lastools/LASzip/src
mex MATLAS .cpp /path/matlas_tools/lastools/LASlib/lib/liblas.a
- I/path/matlas_tools/lastools/LASlib/inc
-I/path/matlas_tools/lastools/src
-I/path/matlas_tools/lastools/LASzip/inc
-I/path/matlas_tools/lastools/LASzip/src

To read a .las/.laz file, use the read command:
[hdr,str] = las2mat ('-i /path/matlas_tools/lastools/data/house.laz');
And to write the structure back into .laz file, simple write:
MATLAS (str,'-o house_copy.laz'); OR MATLAS (str,hdr,'-o house_copy.laz');
For more examples, see the file **Use_Example.m**

2.2 MATLAS IN WINDOWS

This is a short guide for compiling Matlas and LAStools libraries with Code Blocks (MinGW GCC). Please, check the compatibility between your MATLAB version and the GCC compiler used in Code Blocks. These instructions have been tested with MATLAB R2019b, MinGW GCC 6.3 and LASTools v 201124.

MATLAB R2015b, R2016a, R2016b, R2017a: MinGW GCC 4.9.2 from TDM
MATLAB R2017b and R2018a: MinGW GCC 5.3 from mingw-w64.org
MATLAB R2018b and beyond: MinGW GCC 6.3 from mingw-w64.org

PROJECT CONFIGURATION

First, install Code Blocks (v.16.01 was tested) [http://www.codeblocks.org/]

Download from Matworks MinGW GCC 6.3 - MATLAB executable
https://se.mathworks.com/matlabcentral/fileexchange/52848-matlab-support-for-mingw-w64-c-c-compiler

Download LASTools source code: http://lastools.github.io/download/LAStools.zip

At Code Blocks → Open static library project
Change compiler at: Settings → Compiler → Global compiler settings →Toolchain Executable → Find Matworks MinGW GCC 6.3. Common path:

C:\ProgramData\MATLAB\SupportPackages\R2019b\3P.instrset\mingw_w64.instrset
x86_64-w64-mingw32-gcc-6.3.0.exe
x86_64-w64-mingw32-g++.exe
x86_64-w64-mingw32-g++.exe
x86_64-w64-mingw32-gcc-ar.exe

PROJECT → Build options → Compiler settings → Compiler Flags [-std=c++11]

PROJECT → Build options → Compiler settings → OtherCompilerOptions:

-std=gnu++11
-std=gnu++0x
-fexceptions

PROJECT → Build options → Linker settings → Linker options →include standard C/C++ libs:
*See Lastools example configuration

kernel32
user32
gdi32
winspool
comdlg32
advapi32
shell32
ole32
oleaut32
uuid
odbc32

MATLAS 20.11.2020 v.015

PROJECT → Build options → Search directory → compiler/linker → point to

Lastools/Laslib/src
Lastools/Laslib/inc
Lastools/Laszip/src
Lastools/Laszip/dll
C:\Program Files\MATLAB\R2019b\extern\include

INPUT FILES

On the workspace → Right click on the project name → add files

HEADERS:

LASZIP – ALL HEADERS (.h) from LASzip/src and LASzip/dll
LASLIB – ALL HEADERS (.h) at LASlib/inc
MEX.H – C:\Program Files\MATLAB\R2019b\extern\include

SOURCES:

LASZIP – ALL SOURCES(.cpp) from LASzip/src and LASzip/dll
LASLIB – ALL SOURCES(.cpp) at LASlib/src
MATLAS SOURCE CODES - las2mat.cpp, lashdr2mat.cpp, MATLAS .cpp

BUILD THE PROJECT

OUTPUT: Object files (.o) and static library file (.a) at bin/Debug folder or bin/Release

MAKE BINARY FILES

At MATLAB check if your mex file is setup with the command:

If mex is not setup yet, write in the command window and MATLAB should recognize the Microsoft compiler:
[mex -setup] or [mex -setup c]

With the mex setup, copy the static library file (.a) from bin/Debug or bin/Release folder to\OS_Win\Libraries\Win64 (Please Download folder OS_Win). Run **Make_binaries.m** script The binary files can be found at \OS_Win\Binaries\Win64

SIMPLE USE TO TEST

Add binary files path to your script:
[addpath ('…\OS_Win\Binaries\Win64')]

Call the read function:
[hdr,str] = las2mat('-i FilePath\File.laz')
For more examples, see file **Use_Example.m.**