Correlation Matrix Code Manual August 2017

The code in question computes the mode-mode correlation matrices of the quotient spaces of the three-torus. Additionally, it will compute the pixel-pixel correlation matrix of a chosen mode-mode correlation matrix for the HEALpix pixelization scheme for the surface of a sphere. The correct implementation of this code is detailed below, along with the Python files necessary for handling/generating requisite input files and post-run plotting.

1 C++ Code

The C++ code is comprised of a set of files for handling various aspects of the computation, and they must share the same directory.

Header Files:

```
eigenbasis_lin_T3.hpp
eigenbasis_sph_T3.hpp
correlation_matrices.hpp
readToVec.hpp
rotAngles.hpp
Spherical.hpp
spline.hpp
```

Source Files:

```
eigenbasis_lin_T3.cpp
eigenbasis_sph_T3.cpp
correlation_matrices.cpp
main.cpp
readToVec.cpp
rotAngles.cpp
Spherical.cpp
```

Note: Prior to running on a machine that has access to fewer than 10 threads, the `main.cpp` *file must be edited, and the code rebuilt, in order to function properly.*

There is also a set of **.csv** files that are input into the **main.cpp** file. These are:

```
angleData.csv
lengthData.csv
dataMask.csv
eulerAngData.csv
positionData.csv
transFuncData.csv
```

1.1 HEALPix: External Library

The **HEALPix** Library is available from:
http://healpix.sourceforge.net/downloads.php

Its installation instructions can be found at:
http://healpix.sourceforge.net/html/install.htm

Note that **HEALPix** also depends on **CFITSIO**, which is available from:
http://heasarc.gsfc.nasa.gov/fitsio
The methods for downloading and installing the **CFITSIO** library are detailed in the installation instructions for **HEALPix**.

1.2 GSL: External Library

The **GSL** is available from:
https://www.gnu.org/software/gsl

Its installation instructions can be found at:
http://www2.lawrence.edu/fast/GREGGJ/CMSC210/gsl/gsl.html

1.3 Running From a makefile

To use the code on a new machine, open the makefile and edit it by setting the following variables:

1. `INC_DEBUG = $(INC) -I/<filepath>/Healpix_3.31/src/C/subs`
   `-I/<filepath>/ T3-correlations`
   `-I/<filepath>/gsl-x.xx+dfsg/specfunc`
   `-I/<filepath>/T3_correlations`
2. `LIB_DEBUG = $(LIB)<filepath>/Healpix_3.31/lib/libchealpix.a`
   `<filepath>/libcifitsio.a <filepath>/libgsl.a`
3. `LIBDIR_DEBUG = $(LIBDIR) -L<filepath-to-libcfitsio.a>`
   `-L<filepath-to-Healpix_3.31>`
   `-L<filepath-to-libgsl.a>`
4. `INC_RELEASE = $(INC) -I/<filepath>/Healpix_3.31/src/C/subs`
   `-I/<filepath>/ T3-correlations`
   `-I/<filepath>/gsl-x.xx+dfsg/specfunc`
   `-I/<filepath>/T3_correlations`
5. `LIB_RELEASE = $(LIB)<filepath>/Healpix_3.31/lib/libchealpix.a`
   `<filepath>/libcifitsio.a <filepath>/libgsl.a`

6. `LIBDIR_RELEASE = $(LIBDIR) -L<filepath-to-libcfitsio.a>`
   `-L<filepath-to-Healpix_3.31>`
   `-L<filepath-to-libgsl.a>`

The `<filepath>` of each may change on different machines. It is prudent to ensure their correctness prior to running the code once the `T3_correlations` folder and its contents are downloaded.

1.4 Output

Once the makefile variables are set, simply

1. Run the **make** command while in the `T3_correlations` directory
2. From the `T3_correlations/obj/Release` directory run the following two commands:

   (A)
   ```
   g++ -Wall -o T3-correlations correlation_matrices.o eigenbasis_lin_T3.o
   main.o readToVec.o rotAngles.o Spherical.o <filepath>/libcfitsio.a
   <filepath>/Healpix_3.31/lib/libchealpix.a
   ```
   (B)
   ```
   cp T3-correlations <filepath>/T3_correlations
   ```
3. Run `./T3-correlations`

This will create **.csv** files containing the elements of the desired correlation matrix named as "`<matrix type>-<domn>-<i>-<j>-<m>-<n>-<Nmax>.csv`"; where `<domn>` indexes the quotient space, `<i>` indexes the choice of three tilt angles, `<j>` indexes the choice of three principle length scales, `<m>` indexes the choice of three euler angles of the topology, and `<n>` indexes the observer position within the topology. The matrix type will be `R_Cxx` for the real part and `I_Cxx` for the imaginary part, also `X_Cpp`, `X_Cvv`, and `X_Cuu` refer to the pixel-pixel, mode-mode, and cut sky mode-mode correlation matrices respectively. These can be found in the `T3_correlations/Outputs` folder.

1.4.1 Running in Code::Blocks IDE

Code::Blocks is an open-source IDE for C, C++, and Fortran. It is available from:
http://www.codeblocks.org/downloads

The file

`T3_correlations.cbp`

must be in the same directory as the header and source files. These are already included in the project. The only additional steps to build and run the project successfully are to

1. Make sure the “Selected Compiler” is set to “GNU GCC Compiler”.
2. Properly link to the HealPix and CFITSIO libraries.

After installing **HEALPix**, complete Step 2 by:

1. Under “Build Options” → “Linker Settings” → “Link Libraries” add

   `<filepath>/Healpix_3.31/lib/chealpix.a`

   and

   `<filepath>/lib/libcifitsio.a`

2. Under “Build Options” → “Search Directories” → “Compiler” add

   `<filepath>/Healpix_3.31/src/C/subs`

   and

   `<filepath>/T3_correlations`

3. Under “Build Options” → “Search Directories” → “Linker” add

   `<filepath-to-libcfitsio.a>`

   and

   `<filepath>/Healpix_3.31/lib`

The output will be formatted identically to code built with the makefile, however, it may not be useful for all applications of this code to be run directly out of the Code::Blocks IDE. This method is, however, useful for testing, editing, and debugging.

2 Python Code

The elements of the computation that are not performance prohibitive (they only need to be performed once) are implemented in Python rather than C++ so that they can take advantage of Python modules and also remain simple to edit.

There are three files needed to obtain all necessary input files for the C++ code to run. These files do not need to be run in the same directory as the C++ files, but their output files must be in that directory.

1. `CAMBtransferFunctions.py` → Output: `transFuncData.csv`

2. `dom_params.py` → Outputs: `angleData.csv`, `lengthData.csv`, `eulerAngData.csv`, `positionData.csv`
3. `fits_read.py` → Output: `dataMask.csv`

Additionally, to plot the ouput of the C++ code:

`mollweide-post.py`

2.1 Requisite Modules

Generating the C++ code inputs with the provided files requires three non-native python modules:

1. pycamb: http://camb.readthedocs.io/en/latest/
2. healpy: https://pypi.python.org/pypi/healpy/1.8.4
3. astropy: http://docs.astropy.org/en/stable/install.html

pycamb requires a Fortran90 or later fortran compiler and the CAMB software package to be installed.

healpy requires the **HEALPix** library to be installed.