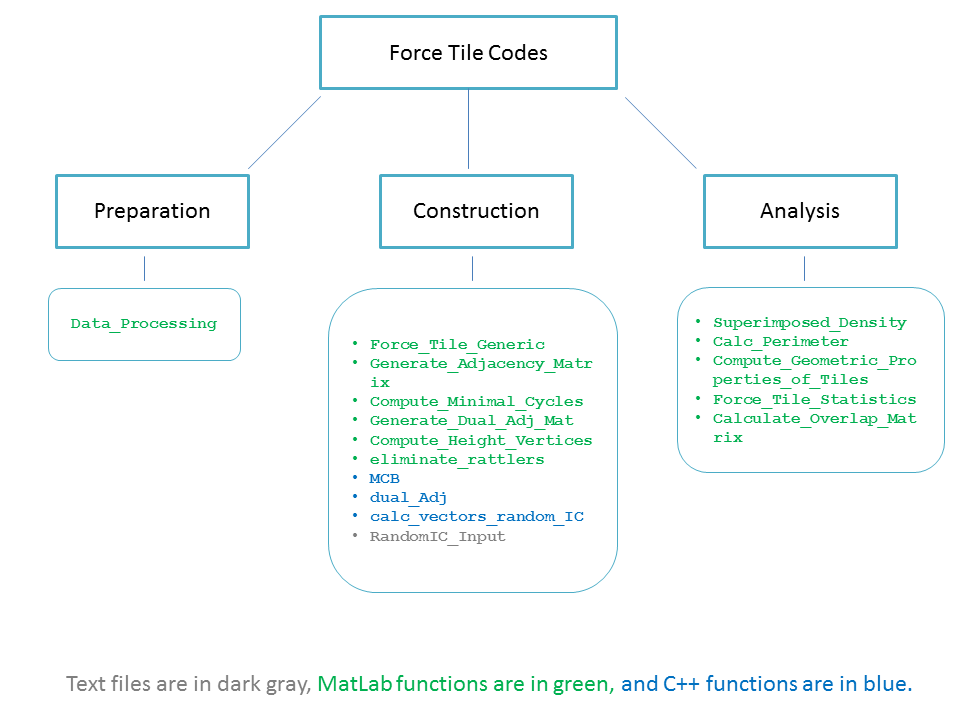
Force Tiling Cookbook

The force tiling algorithm creates a tiling of forces starting from a force balanced configuration. The force tiling idea has been explained in detail elsewhere, so in this cookbook, I'll refrain from going over it again. Instead, this document will serve as a step by step guideline to use the code I've written to generate the force tiling. Force tiling will be referred to as FT henceforth.

The FT code consists of multiple source files written in MatLab (ML) and C++. Computation intensive part of the code (e.g., finding the Minimum Cycle Basis, topology of the dual graph, etc. ) is written in C++. ML acts as an interface, since it's easier to manipulate and plot data in ML than C++. Some of the computationally easy codes are also written in ML.

The codes can be broadly divided into three categories: (a) Data Preparation, (b) Force Tile Computation, and (c) Analysis. A summary of the codes in each category is shown in the following diagram.



**(a) Data Preparation:** The data preparation code reads raw data obtained from experiments and simulation, and removes unnecessary information. The FT code requires the input data in the following format:

* A file containing the index and coordinates of the grains (4 columns). The format should be:

*grain id x-cor y-cor diameter*

* Another file containing the contact forces in the following format (5 columns):

*grain id #1 grain id #2 signed force magnitude x-component of the unit vector along the force y-component of the unit vector along the force*

It's important to specify the signed magnitude of the forces, because with attractive interaction, the force can be negative.

The data preparation code takes the raw datafile from an experiment or simulation and creates a ML binary file with some specified name with the above mentioned formatting.

The format of the actual data obtained from different experiments vary, so make sure to change the data preparation code accordingly. For example, in expt #1, the positions may be in a file called *“cm\_xyz.txt”,* where *xyz* may be step number or some other number identifying a particular set of experiments. In this file, the grain id may be in col 1, the coordinates in col 4 and 5, the diameter in col 7, while rest of the columns are unimportant. Expt #2, however, may have the name *“posn\_abc.dat”* for the same file and the corresponding columns may be quite different. Same is true for the file containing the forces.

**(b) Force Tile Computation:** The FT computation is an involved process and requires multiple steps to generate the force tiling.

1. Generate the adjacency matrix of the grain contact network. Typically, the adjacency matrix is scalar valued and the scalar provides the weight of that particular contact. In this code, the weight is vector valued and is equal to the force acting through that contact.
2. Compute the **minimum cycle basis** or MCB. The minimum cycles are required to obtain the topology of the dual graphs. Please use a book on graph theory or Wikipedia to learn about MCB and how they may be used to obtain the dual of a planar graph.
3. Generate the dual graph topology from the MCB. Generates the scalar valued adjacency matrix of the dual graph. This is the topology of the FT.
4. Generate the vector valued adjacency matrix for the FT. This is what will be used to construct the FT vertices.
5. Obtain the FT vertices using the vectors in the adjacency matrix. This process is similar to how one obtains the crystal lattice from the primitive vectors. However, in that case, there are handful (2 for 2D, 3 for 3D) of primitive vectors, and the task is simpler. Since the FT is disordered, there are as many unique contact vectors as there are contacts. Hence, we construct the vertices iteratively through an optimization process.
6. After certain number of steps (hard coded), we obtain the FT network. We have information about the vertices, and the topology.

**(c) Analysis of the FT:** Different kinds of analysis can be done on the FT. However, the most useful analyses are the following:

1. Overlap. Measures the persistence of the FT. Higher overlap indicates a persistent structure, which is a signature of a rigid object.
2. Statistics of the area and perimeter of the FT. The perimeter roughly maps to the pressure and the area to the square of the deviatoric stress. These measures can be compared against the pressure and deviatoric stress obtained directly from the experiment or simulation.
3. Variance and Correlation of the perimeter and area.
4. Fraction of non-convex polygons.

Using the code:

1. First compile the c++ codes in CPP Source folder to generate the required executables. The codes are written with C++11 standard. While compiling, please keep that in mind. I have compiled the codes in 64 bit Ubuntu with a rather inelegant method as described below and they work. I've compiled the same code in Windows using Code Blocks, and it works.

* To comile the MCB code, go to the MCB directory, and use the following command to generate the executable.

$ g++ -std=c++11 \*.cpp \*.h -o MCB

* To compile other cpp codes use a similar command. For example, for codename.cpp,

$ g++ -std=c++11 codename.cpp -o codename

2. Once they are compiled, you're all set to use the force tiling code written in MatLab. Figure out how to run a matlab code from the internet. Here are the steps to create a force tiling.

* First use the Data\_Processing\_xyz.m, where xyz is a generic name for some experiments. For example, here I have given the name “Frictionless”. This code will process the raw data from experiment and simulation in a format that can be used to generate the tiling. Also see the README file. As an example, I've included a sample data file in the Sample folder.
* Use Force\_Tiling\_Generic to generate the force tiling and plot it. This code will also save create and save the data in a folder called Data.

* You're good to go. Analyze the code and figure out what you need to change to accommodate your data.

**Debugging:** Using code written by another person is a nightmarish experience. To exacerbate that, I'm a horrible code writer. If you run into a problem, which is highly probable, and need to contact me, please follow these steps for quick response.

1. Copy and paste the detailed error message from MATLAB.
2. Send it to [forcetileenquiries@gmail.com](mailto:forcetileenquiries@gmail.com)
3. Send the email sometime between 0-9 AM Boston Time.
4. Often there are mistakes due to a missed semicolon or a misplaced comma or a quote. Please make sure that your error doesn't belong to this category. If that's the case, expect no reply from me.
5. If that's not the case, I'll be happy to help you through the ordeal.

I'll periodically update this document with the FAQs and other tips. Until then, happy tiling!