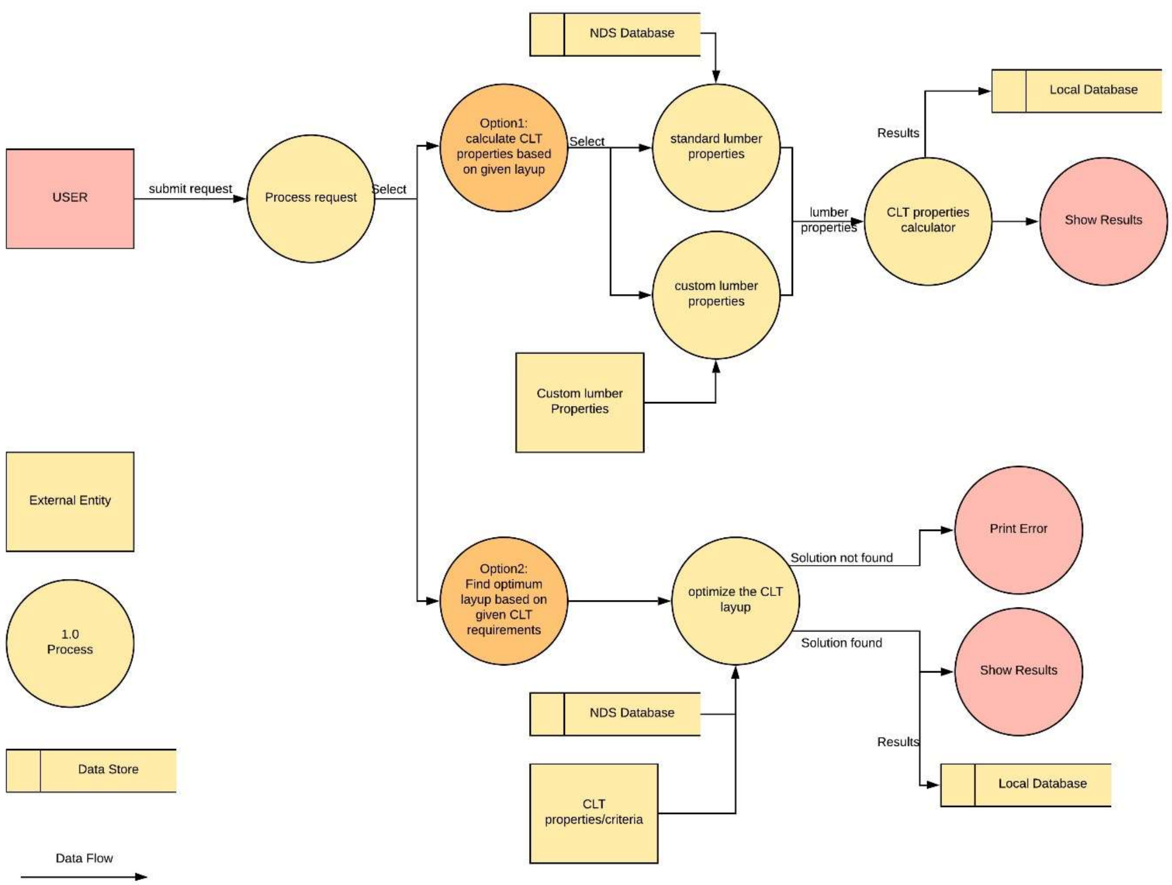
Documentation for CLT Properties Calculator Algorithm

Code and Documentation: Sina Jahedi, sina.jahedi@oregonstate.edu

Supervisor: Dr. Lech Muszynski, lech.muszynski@oregonstate.edu

---



Spring 2020

Oregon State University

Department of Wood Science and Engineering

Introduction

A Python script developed to calculate the structural properties of given CLT layup in accordance with PRG-320 2019 standard. The software accepts arbitrary properties of the laminations as well as desired CLT layup and prints the calculated structural properties in minor and major directions. A second script is developed to reverse the sequence of initial algorithm, meaning that, it calculates the most suitable CLT layup based on given desired CLT properties.

General Format

The developed projects should be attached with this documentation. The two projects can be found with the following folder names:

"~/Calculator": *software accepts arbitrary properties of the laminations as well as desired CLT layup and prints the calculated structural properties in minor and major directions.*

"~/Optimizer": *calculates the most suitable CLT layup based on given CLT properties and lamination properties.*

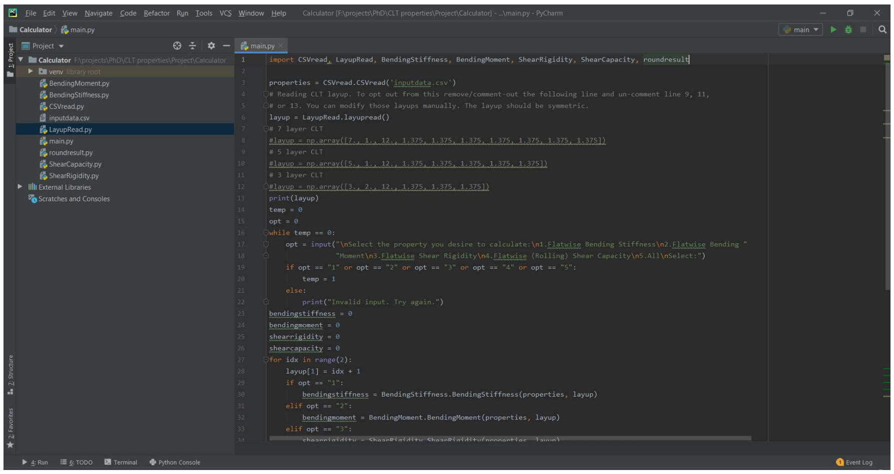
These projects can run individually without any help from each other. Changing the folder names will result in malfunction of the code. In order to change the name of the folders/projects refer to Pycharm tutorial.

Assumptions and Limitations

- It is assumed that the calculated CLT is symmetric. The number of plies is restricted to 3, 5, or 7 and the plies next to each other are perpendicular. Forcing otherwise will result in incorrect answers.
- The minimum allowable thickness of each ply is set to 0.625 inches in accordance to PRG-320 2019. Although mentioned otherwise in PRG the maximum is set to 5 inches for more flexibility. This can be changed in the code body by changing variable "limit" in "Optimizer/main.py".
- For calculation of all the mechanical properties we precisely followed PRG-320 2019, Appendix X3. Any assumption used for deriving those formulas is also true for this software.

How to Use

The software is developed in Python programming language. The best way to open and run the projects is to use Pycharm. After installation, open either project using the **file** tab, and select **open**. You should specify the directory which project is located at, e.g. "F:\CLT properties\Project\Calculator". Upon successful opening of the project you should be able to see the project folder on the toolbar located on the left. Click on "Calculator" and select "main.py". At this point you should be able to see the screen below:



For running the project for the first time you need to select **Run** tab, and select **Run** while you have opened "main.py". For running the program second time and forward you can simply click on the green triangle on the top right corner.

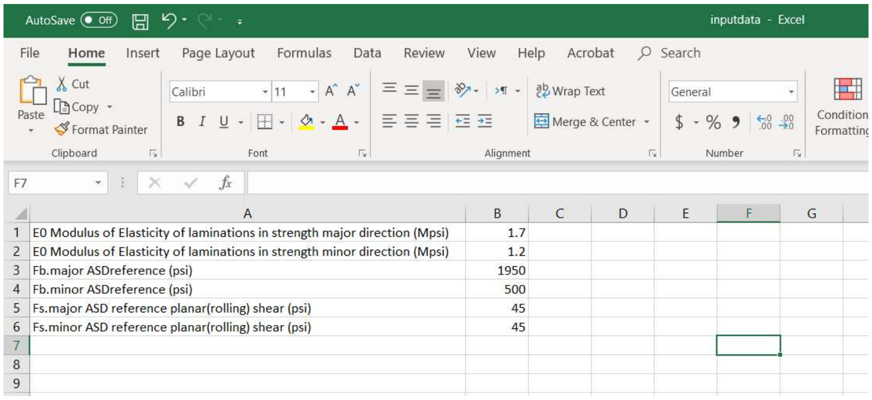
Troubleshoot: If you received an error indicating that the interpreter is not working properly you will need to create a new project. Follow these steps:

1- Create a new project.
2- Copy any file (e.g. main.py, BendingMoment.py, CSVread.py, inputdata.csv, etc.) from "~/Calculator" to your new project directory. DO NOT copy ".idea", "_pycache_", "venv" and other folders.
3- Import any missing library to the project. E.g. you will need to add numpy library to the project. Pycharm does a great job for suggesting and automatically importing the missing libraries.
4- Run the script.

The same procedure is true for "~/Optimizer" project.

Input

Either projects contain a file named "inputdata.csv" which specifies the properties of the laminations. You can open and edit the file using Microsoft Excel. You should keep the format of the .csv file as the original file. Image on the right shows the original format.



- Changing the position of rows and columns will result in malfunction of the script.
- Each section should be filled in the **exact unit** as provided in the file.
- **No blank** cells in any of the cells shown above.
- Do not change "inputdata.csv" file name and format. If you change the file name, you should change the code accordingly.

There should be a folder attached to this file named "PRG standard laminations". This folder provides the standard laminations which recommended by PRG-320 2019 in the correct format as required by the algorithm. For example, the file named "E1.csv" contains the lamination properties of E1 CLT grade ready to be plugged in to the code. For using "E1.csv" you will need to remove "inputdata.csv" in the according project, and replace it with your desired file. **Reminder:** Make sure to rename the file to "inputdata.csv".

Output

Currently both projects are designed to print the output on the command line. One may change the code body to print the output on an external file using CSV library. The output is in the same sequence as Table A2 in PRG-320 2019.

Units

Every input and output value used in the algorithm should be in imperial unit system and in accordance with PRG-320 2019, Appendix X3. Make sure to follow the guidelines offered in the command line when inserting input values.

Rounding

For rounding the results in "Calculator" project we followed the information given in PRG-320 2019 Table A2 footer, page 34. The numbers are rounded only in the final stage of the algorithm before showing the results. No value is rounded in the mean calculations.

In "Optimizer" project all the final properties are rounded to the closest 0.01. This is to help the user to find the true values of the output.

~/Calculator Project

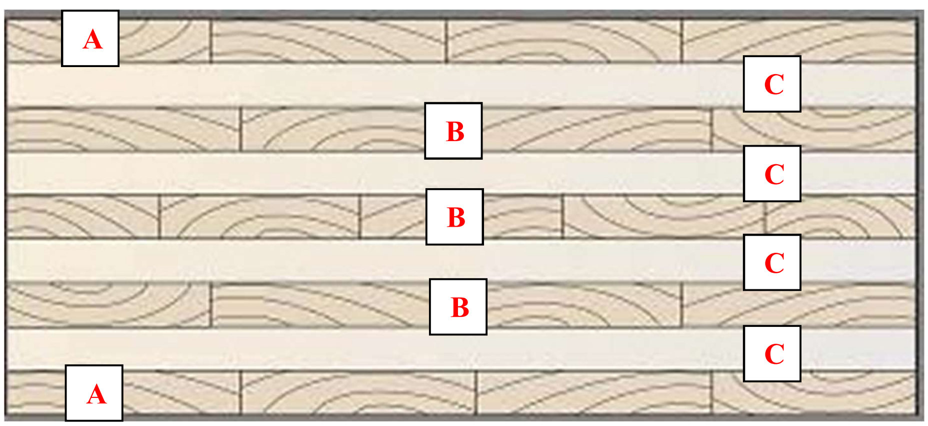
This part of the software calculates the properties of the desired CLT based on given lamination properties and layup. For running the project, you need to run "main.py" in the project folder. This file reads various functions stored in other files located in "~/Calculator" and prints the results.

Main.py

Running the main function will initiate the process by reading "inputdata.csv" file. If the file read successfully the input data will be printed on the command screen. At this point the software asks for input data to acquire CLT layup.

- The first question asks about the number of layers you intent to have for the CLT. You can either select 3, 5, or 7.
- Next question asks for design width. This value should be inserted as described by PRG-320 standard. To insert the default value type "**12**".
- Next the software will ask about the thickness of each lamination. The questions should be answered in accordance with the image and description below:
    - **A** is "Thickness (in) of the out most layer"
    - **B** is "Thickness (in) of the inner plies parallel to the out most layer"
    - **C** is "Thickness (in) of the inner plies perpendicular to the out most layer"

  Note: Thicknesses are the same for every ply marked with the same character.



- Lastly you need to say which properties of the CLT you desire to calculate. Typically users select the option which provides all properties, but the other options may come handy in the case that you lack some of the lamination properties.

Hardcoding Layup

LayupRead.py function gathers input data from user and return a np.array with the size of < number of plies + 2 > which later be used for calculation of properties. One may want to hardcode (manually inserting) the CLT layup instead of going through the questions every time for running the code. For this you need to follow the steps below:

1- Comment out layupread function by placing # sign in the beginning of line 6 in "main.py" function.
2- Remove the # sign from either line 8, 10, or 12 (depending on the desired number of plies) to activate those lines.

```
#layup = LayupRead.layupread()
# 7 plies CLT
layup = np.array([7., 1., 12., 1.375, 1.375, 1.375, 1.375, 1.375, 1.375, 1.375])
# 5 plies CLT
#layup = np.array([5., 1., 12., 1.375, 1.375, 1.375, 1.375, 1.375])
# 3 plies CLT
#layup = np.array([3., 2., 12., 1.375, 1.375, 1.375])
```

3- Change the layup array based on the information given in table below. A, B, and C refer to plies marked in previous figure. Strictly follow the format provided in table below, otherwise the code generates faulty results.

Valid Input	Number of plies	Const.	Width	Ply A	Ply C	Ply B	Ply C	Ply B	Ply C	Ply A
Example 7-plies	7	1	12	3	2.375	1.375	2.375	1.375	2.375	3
Valid Input	Number of plies	Const.	Width	Ply A	Ply C	Ply B	Ply C	Ply A		
Example 5-plies	5	1	12	3	2.375	1.375	2.375	3		
Valid Input	Number of plies	Const.	Width	Ply A	Ply C	Ply A				
Example 3-plies	3	1	12	3	2.375	1.375				

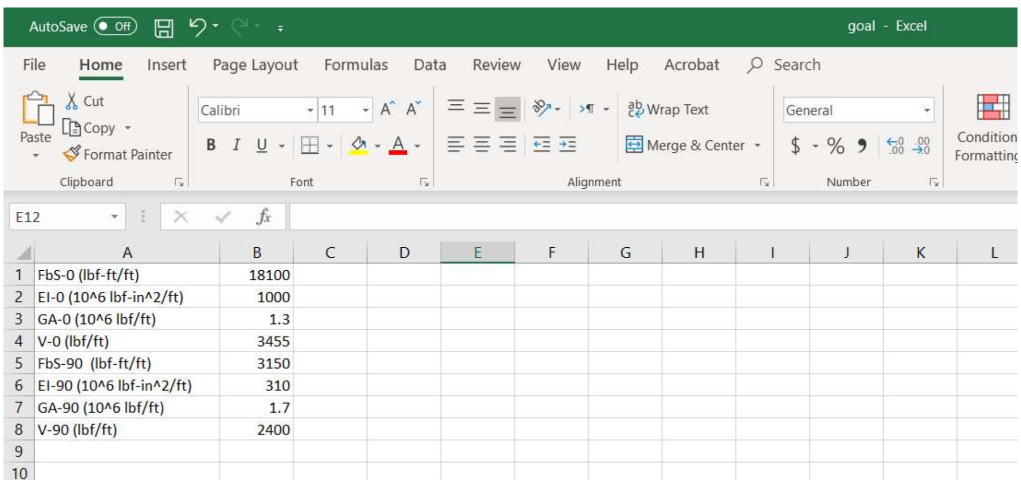
~/Optimizer Project

This part of software prints optimal layup based on given lamination properties and desired CLT structural properties. For running the project, you need to run “main.py” in the project folder. This file reads various functions stored in other files located in “~/Optimizer” and prints the results.

goal.csv

In this file you should specify the goal structural properties for the CLT you want. The goal properties are the minimum required values for each property. If one of the properties is not important for your case you can leave 0 in that field. The same as “inputdata.csv” changing the format of the file or leaving any cell blank will result in malfunction of the code. The units used for filling this file should be the same as mentioned in table A2 PRG-320 2019, page 33.

The properties marked with -0 are the properties in major direction and those marked with -90 are properties in minor direction. For example FbS-0 is the flatwise bending moment in major strength direction and FbS-90 is the property in minor strength direction.



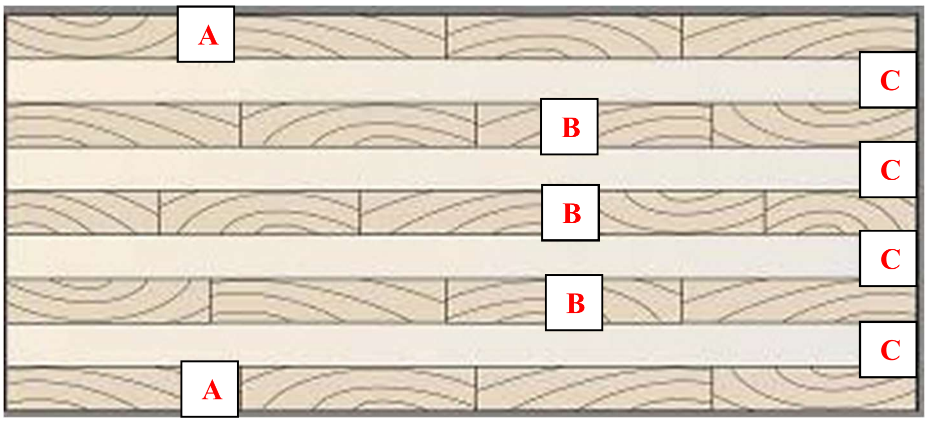
	A	B
1	FbS-0 (lbf-ft/ft)	18100
2	EI-0 (10^6 lbf-in^2/ft)	1000
3	GA-0 (10^6 lbf/ft)	1.3
4	V-0 (lbf/ft)	3455
5	FbS-90 (lbf-ft/ft)	3150
6	EI-90 (10^6 lbf-in^2/ft)	310
7	GA-90 (10^6 lbf/ft)	1.7
8	V-90 (lbf/ft)	2400

Note: Save “goal.csv” file after each change to see the effect in the main software.

Main.py

Running the main function will initiate the process by reading “inputdata.csv” and “goal.csv” file. If the file read successfully the input data will be printed on the command screen. Afterwards the software asks for the desired number of plies for CLT. Then you need to specify which method do you want to use for optimization. The options are:

- **Symmetric but un-uniform thicknesses of plies:** meaning that plies A, B, and C may have different thicknesses.
- **All of the plies have the same thickness:** Plies A, B, and C surely have the same thicknesses.
- **Plies parallel to each other have the same thickness:** Plies A and B surely have the same thicknesses, but C may be different.



The software starts with the hardcoded initial layup and calculates the structural properties. If the properties satisfy the goal it prints the output, otherwise increases the thickness of plies and tries again. If the software reaches the thickness limit and still it does not satisfy the goal it prints the error “We have reached the limit. We couldn't find any possible layup for the specified properties”. In the case of receiving this error you may want to increase input number of plies.

You are able to change the hardcoded variables to meet your needs. **Diff** specifies the thickness (in) that is added to each ply when algorithm iterates for finding the optimum layup. To comply with American lumber market we recommend 1/8, 1/4, 1/2, and 1 for this variable. **Plies** is the array defining the initial layup for iterations. The first value in the array is the thickness (in) of plies marked as A in the figure above, and the second and third values are B and C accordingly. The minimum allowable thickness of each ply is 5/8, PRG-320 2019. Finally, **limit** defines the maximum thickness (in) of each ply in the CLT.

```
# Iteration step in (in).
Diff = 0.125
# Starting layup. The values are Major, Major2, and minor directions in order (in).
Plies = [0.625, 0.625, 0.625]
# limit will define the maximum allowable thickness for each ply in CLT.
limit = 5
```