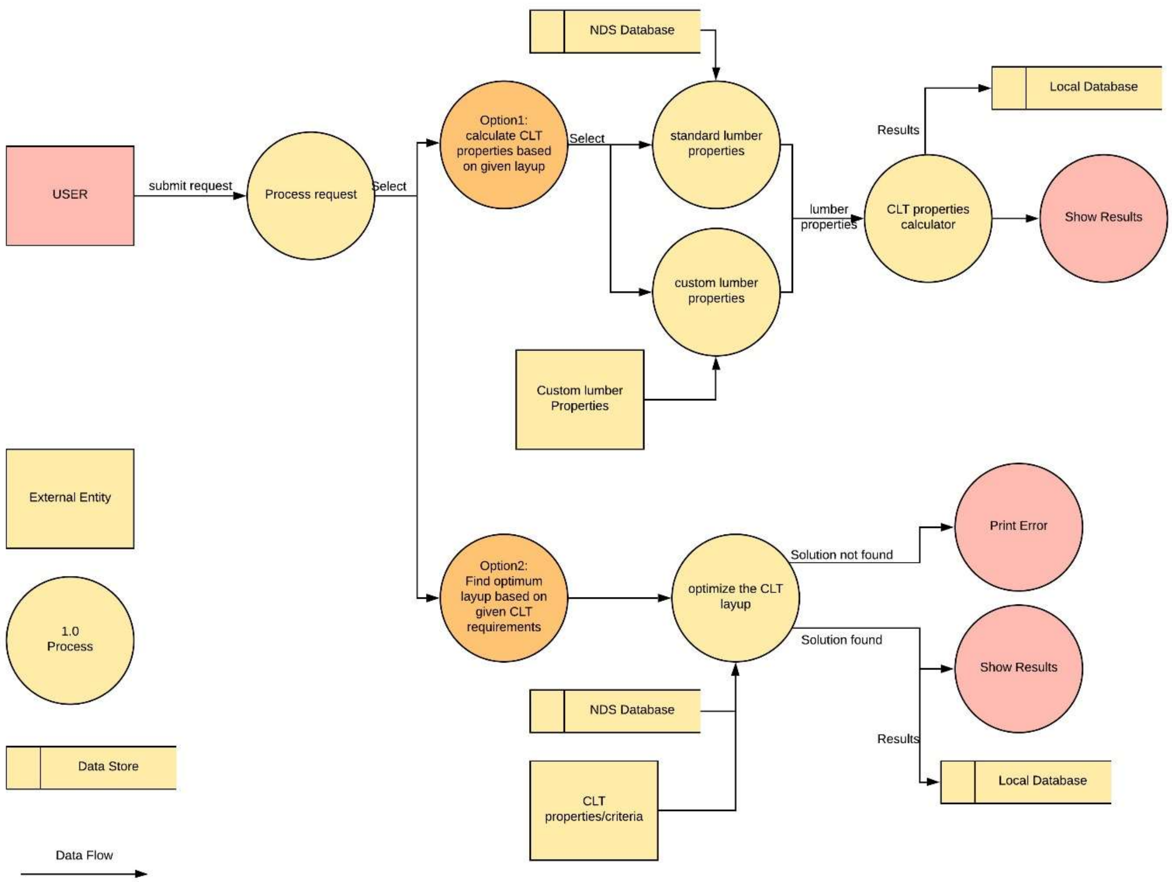
Documentation for CLT Design Values Calculator Application

Source Code Repository: https://github.com/sina-jahedi/CLT_Design_Values_Calculator

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 was developed to calculate the design values of a given CLT layup based on shear analogy method which is referred by PRG-320 2019 standard. The software accepts arbitrary properties of the laminations as well as characteristics of the CLT layup, e.g. thickness, number of plies, etc. and calculates CLT design values in the minor and major directions. A second script was developed to optimize the layup in a reverse sequence of initial algorithm, meaning that, it calculates the most suitable CLT layup based on a given set of desired design values CLT.

The objective of this document is to describe the methodology and assumptions used for development of this application and also provide a guideline on how to use the script.

General Format

The script of this projects is uploaded on an open-access GitHub repository linked below.

Source Code Repository: *https://github.com/sina-jahedi/CLT_Design_Values_Calculator*

The two programs developed in this project can be found in the following folder:

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

"~/Layup Optimizer": *calculates the most suitable CLT layup based on given criteria.*

These projects can run individually independent to each other.

Formulation of Shear Analogy Method

Shear analogy method (Kreuzinger 1999) following the exact formulation provided in PRG-320 Appendix X3 was used in this application. The presented methodology uses the arbitrary laminations design values and calculates the overall design values of a layer-wise composite panel. The input and output data are summarized in Figure 1. Refer to the standard for more details on the formulation.

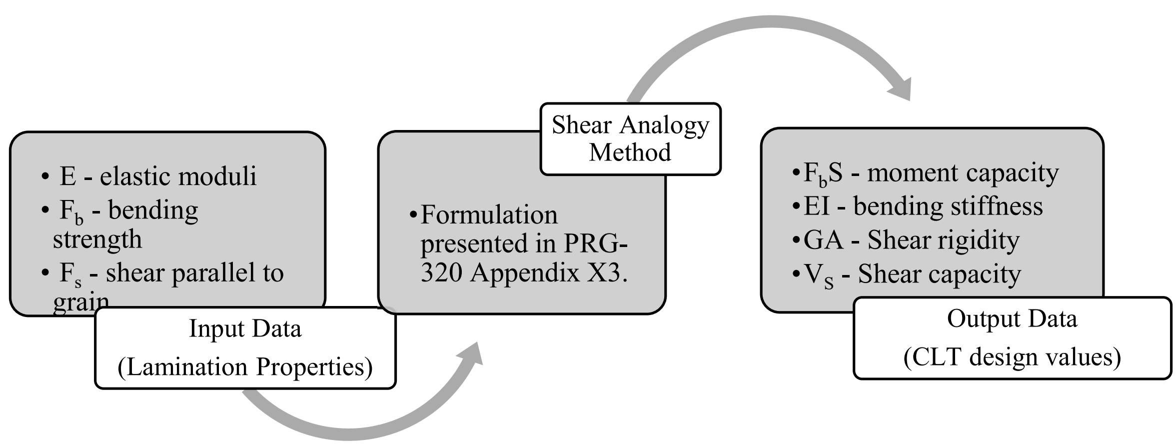


Figure 1, Input and output data for the shear analogy method formulation used in PRG-320.

It shall be noted that the properties of each lamination can be different from the other. For most practical uses, all longitudinal layers are selected with the same grade and properties and all transverse layers similar to each other as well.

The methodology is capable of calculating design values of CLT in major and minor strength directions. Following the standard, it is assumed that:

$E_{transvers} = E_{longitudinal}/30$

$G_{longitudinal} = E_{longitudinal}/16$

$G_{transvers} = G_{longitudinal}/10$

Where E is elastic moduli and G is shear moduli of the longitudinal or transverse laminations.

Limitations of the Software

- Although shear analogy method is capable of calculating unsymmetric layups, this software is limited to symmetric layups only. The number of plies is restricted to 3, 5, or 7 oriented cross-wise. Forcing otherwise will result in incorrect answers.
- PRG-320 2019, Appendix X3 was precisely followed. Any assumption used in PRG-320 is also true for this software.

HOW TO USE

The software is developed in Python. The best way to open and run the projects is to use Pycharm. After installation of Pycharm, open either project using the **file** tab, and select **open**. You should specify the directory in which the project is located at, e.g. "F:\CLT properties\Project\Calculator". Once opened, you should be able to see the project folder on the toolbar on the left. Click on "Calculator" and select "main.py". At this point you should be able to see the screen showed in Figure 2.

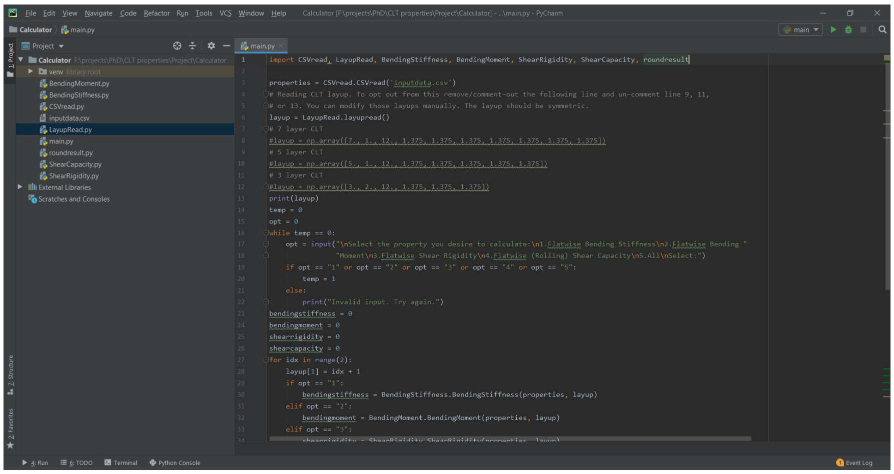


Figure 2, The user interface of Pycharm and opening the main script.

To run the project for the first time you need to select **Run** tab, and select **Run** while you have "main.py" script opened. After that, you can simply click on the run button (green triangle on the top right corner).

Troubleshoot: If you received an error saying 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 on suggesting and automatically importing the missing libraries.
4- Run the script.

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

Input

Both 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 file as ".csv". Please note that:

- 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.
- Do not change the name of the file and format or you should change the script in code accordingly.

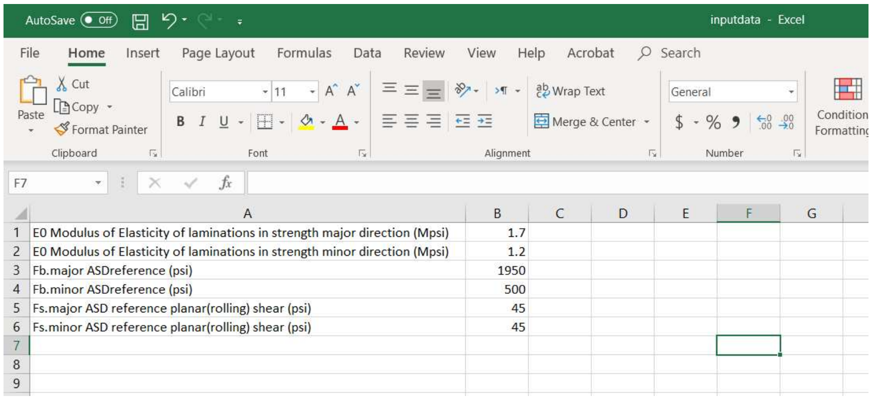


Figure 3, inputdata.csv file that contains the input data for lamination properties.

There is a folder uploaded on GitHub named “PRG standard laminations”. This folder provides the basic laminations that are provided in PRG-320 2019. The format and units of the input is adjusted accordingly. For example, the file named “E1.csv” contains the lamination properties of basic E1 CLT grade ready to be plugged in to the code. To use this, for example “E1.csv”, you will need to delete the “inputdata.csv” and replace it with your desired file. 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

In the “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.

In “Optimizer” project all the final properties are rounded to the closest 0.01.

~/Design Values Calculator

The project titled as "Design Values Calculator" contains a script that calculates the design values of a desired CLT layup based on given set of properties for the laminations. To run the application, you need to access the "main.py" file in the project's folder. This file reads and runs the functions available in the other .py files located in the subfolders. The results will be printed in the command line.

Main.py

The main script starts with reading the "inputdata.csv" file. If the file read successfully, the input data will be printed on the command screen and then the application asks for information about the desired CLT layup.

- Type the number of layers you desired for the CLT. You can either select 3, 5, or 7.
- Select the standard width of the panel (design width). The standard value presented by PRG-320 is 12 inches. To insert the default value type "12".
- Next the software asks about the thickness of each lamination in the following order, Figure 4:
    - **A** is "Thickness (in.) of the out most layers"
    - **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.

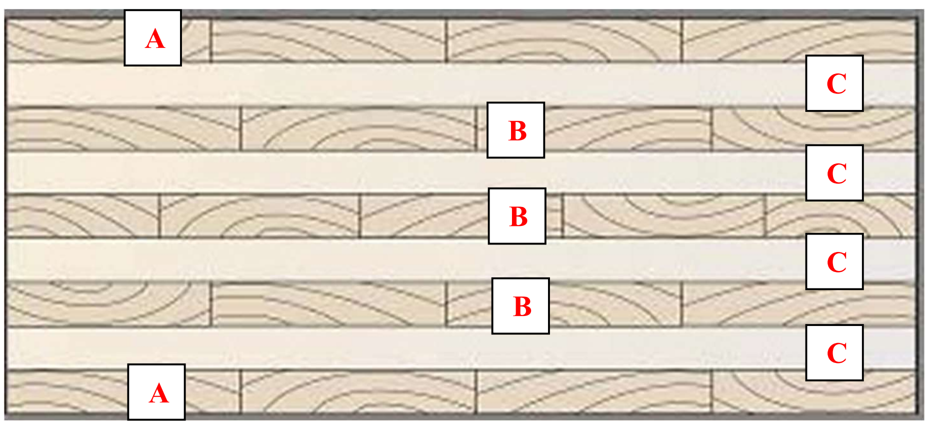


Figure 4, A guideline to insert lamination thicknesses in the Design Values Calculator application.

- Lastly you need to say which design values of the CLT you desire to calculate, e.g. moment capacity, stiffness, etc. Typically, users select the option that calculate all design values, however the other options may come handy if there is a lack of information on some the lamination properties.

Hardcoding The Layup

LayupRead.py script gathers input data from the user and returns a np.array with the size of <number of plies + 2 > that will be used for the calculations. One may hardcode the CLT layup. In other words, the goal is to insert the CLT layup as a variable in the body of code instead of going through the input questions. 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, Figure 5.

```
#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])
```

Figure 5, Hardcoding the layup information in the main.py file.

3- Change the array based on the information given below. A, B, and C refer to the plies marked in Figure 4. Strictly follow the format and sequence provided below, otherwise the code generates faulty results. All values are in inches. And the default value for width is 12.

For 7-ply: [7, 1, width, A, C, B, C, B, C, A]

For 5-ply: [5, 1, width, A, C, B, C, A]

For 3-ply: [3, 1, width, A, C, A]

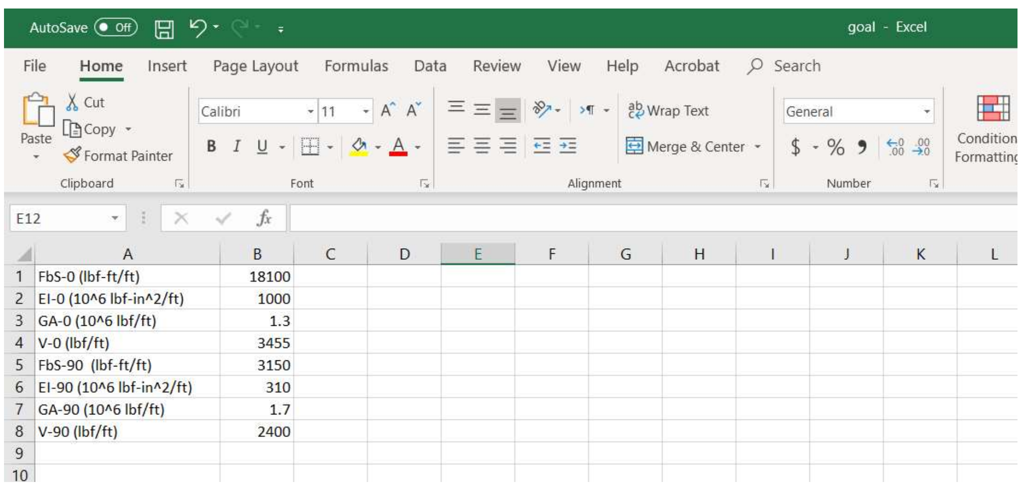
~/ Layup Optimizer

The project titled as "Layup Optimizer" contains a script that calculates the optimal CLT layup based on given lamination properties and the desired CLT design values. The optimization criterion is defined as the minimum allowable design value for each property, e.g. moment capacity, stiffness, etc. This script will run the "Design Value Calculator" application iteratively to match the criteria with the minimum possible CLT gross thickness.

To run the project, you need to access "main.py" in the optimizer folder. This file reads and runs the functions available in the other .py files located in the subfolders. The results will be printed in the command line.

goal.csv

In this file you should specify the criterion (optimization goal) for the desired CLT panel. If one of the properties is not important for your design, you can leave 0 in that field. Changing the format of the file or leaving any cell blank will result in compilation error. The design value units used in this file is the same as table A2 PRG-320. 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 effective flatwise moment capacity in major strength direction and FbS-90 is effective flatwise moment capacity in minor strength direction. Save the file after each change to see the effect in the main software.



	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

Figure 6, goal.csv file that contains the criteria for optimization algorithm.

Main.py

The main script starts with reading the "inputdata.csv" and "goal.csv" files. If the file read successfully, the input data will be printed on the command screen. Afterwards the application asks for the desired number of plies for CLT. 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 have the same thicknesses, but C might be different.

The software uses the "Design Values Calculator" code to derive the structural properties. If the derived design values meet the goal, the application prints the output on the command line, otherwise it increases the thickness of plies and tries again. If the software reaches the maximum thickness limit, it prints an error indicating "*We have reached the limit. We couldn't find any possible layup for the specified properties"*.

You might want to change the variables defined in the main.py. **Diff** specifies the thickness (in.) addition increments in each iteration. To comply with American lumber market we recommend 1/8, 1/4, 1/2, and 1 inch for this variable. **Plies** is an array defining the initial layup which is typically defined as the thinnest possible layup. Following PRG-320, the minimum allowable thickness for each ply is 5/8. The first value in the array is the thickness (in) of plies marked as A in Figure 4, and the second and third values are B and C accordingly. Finally, **limit** defines the maximum possible thickness (in) for each ply in the CLT.