

# PACCMAN User/Developer Guide

(Python Analysis of Conformal Cooling Molds And/or designS)

This guide is meant to be used as a resource for using, adding to, or integrating the PACCMAN program. It is expected that users will have a package manager such as Anaconda or another program with Numpy, Scipy, Matplotlib, etc.

In order to install Pytest with Anaconda, open the Anaconda Powershell Prompt and run the command `"conda install -c anaconda pytest"`.

Feedback is welcome. Please report all bugs to [hughfeehan353](#) on Github. Thanks.

## **Introduction:**

This program is designed to aid in the optimization and design of conformal cooling molds. While sample variables are included, its main use is for user-provided data. From the chosen data file, PACCMAN performs functions and provides outputs of information about the mold that the inputted data describes. The outputs can then be compared to another design to determine which is the most efficient or saved as estimations of a particular mold's characteristics. A NumPy array can also be used as an input variable to see how different values of a variable affect the mold's performance. Multiple arrays can be used to compare things like the mold material that have multiple material property variables. However, it is important to consider that each variable for a specific material must be put in the same position in the arrays.

## Basic Use:

When run, the program first presents three options.

Selecting "N" uses the first mold material properties (KM1, etc.) and then asks the user to choose the heat transfer coefficient correlation.

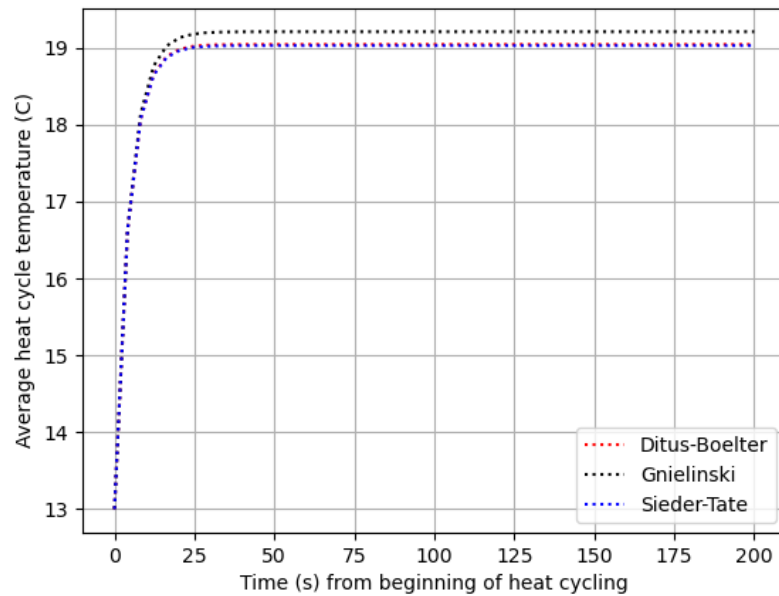
Selecting "M" compares the three mold materials using the Gnielinski correlation.

Selecting "H" compares the three heat transfer coefficient correlations using the first mold material property (KM1, etc.).

For all three choices, the program asks the user if they would like to save graphs of the average heat cycle temperature over time.

Choosing "Y", downloads the graph in .png and .eps formats.

Choosing "N" will not download the graphs



An example graph of the PACCMAN results.

The program is written so that by changing the variables, it can analyze any desired conformal cooling design and will report the following data:

flow velocity, kinematic viscosity, Reynolds number, Prandl number, Darcy friction factor, heat transfer coefficient, average heat cycle temperature of the mold, time constant, and coolant pressure drop.

## **Multiple Variables / Arrays**

Arrays can be used in this program to compare either multiple values for a single variable or multiple values for multiple values.

When comparing multiple values for a single variable, use a NumPy array and the program will compare the efficacy of the values entered.

When inputting multiple values for multiple variables, the number of values in each array must be the same. The values in array position 1 will always do arithmetic from the other values in position 1, the values in position 2 work with the other values in position 2, etc.. The final results that the program give will be the same as the number of array variables, which must be the same for every array. A recommended use of this is comparing different materials for the mold, coolant, or part and putting all the material properties of the same material in the same array slot.

## **Application Fundamentals for Advanced Use:**

At the beginning, the program initializes and assigns all the basic variables. By default, these are imported from the "basedata.py" file.

In order to change the values, the data in this file can either be edited directly or the import location can be changed. In order to do this, locate the import statement on line 12 of "paccman.py" and change "basedata" to the desired new file. All of the variables in "basedata.py" are necessary for the program to function properly so none should not be completely omitted if a new file is used.

Afterwards, all the necessary functions are defined. These cannot be removed as they are necessary for the program's function but could be added to.

At this point there is an "if" statement to check for the user's desired function for the program. This "if" statement contains all of the program's calculation functions in order to ensure that nothing will break if the user chooses an invalid letter.

The first thing inside the "if" statement are some calculations that must be completed no matter the chosen choice of the program's function.

The rest of the calculations contain nested "if" and "elif" statements so that the correct variables are used for the chosen program function.

Once the calculations are completed, the program creates a plot of the average heat cycle temperature over time that the user has the choice of downloading.

The program can be tested using Pytest by running the following command in the program's directory: "python -m pytest".

## Variables Used:

Part Thickness:  $L_P$

Coolant Line Diameter:  $D$

Average Height of Coolant Line Surface Irregularities:  $\varepsilon$

Coolant Line Length:  $L$

Coolant Thermal Conductivity:  $K_C$

Coolant Density:  $\rho_C$

Mold Specific Heat Capacity:  $C_C$

Coolant Temperature:  $T_C$

Part Density:  $\rho_P$

Part Specific Heat Capacity:  $C_P$

Thermal Conductivity of Mold:  $K_M$

Coolant Line Pitch Distance:  $W$

Distance from Coolant Line to Mold Wall:  $L_M$

Difference in Part Temperature Between Inserted Into Mold and Released:  $T$

Cycle Time:  $T_{Cycle}$

Mold Density:  $\rho_M$

Mold Specific Heat Capacity:  $C_M$

Initial Mold Temperature:  $T_{MO}$

Coolant Flow Rate:  $\dot{V}$

Coolant Dynamic Viscosity:  $\mu$

Coolant Dynamic Viscosity When Near Wall:  $\mu_W$

## Equations Used:

Flow Velocity of Coolant (U):  $\frac{\dot{V}}{\pi \cdot (\frac{D}{2})^2}$

Coolant Kinematic Viscosity (KV):  $\frac{\mu}{\rho_C}$

Reynolds Number (Re):  $\frac{U \cdot D}{KV}$

Prandtl Number (Pr):  $\frac{\mu \cdot C_C}{K_C}$

Darcy Friction Factor (Haaland Equation) (f):  $(\frac{1}{-1.8 \log[(\frac{\epsilon}{3.7 \cdot D})^{1.11} + (\frac{6.9}{Re})]})^2$

There are three chooseable equations for the Nusselt Number (Nu):

1. Dittus-Boelter:  $0.023 \cdot Re^{\frac{4}{5}} \cdot Pr^{0.4}$

2. Gnielinski:  $\frac{\frac{f}{8} \cdot (Re - 1000) \cdot Pr}{1 + [12.7 \cdot (\frac{f}{8})^{0.5} \cdot (Pr^{\frac{2}{3}} - 1)]}$

3. Sieder-Tate:  $0.027 \cdot Re^{\frac{4}{5}} \cdot Pr^{\frac{1}{3}} \cdot (\frac{\mu}{\mu_w})^{0.14}$

Heat Transfer Coefficient (h):  $\frac{K_C}{D} \cdot Nu$

Average Temperature of the Mold (ATM):  $T_C + \frac{\rho_P \cdot C_P \cdot L_P \cdot T \cdot [(2 \cdot K_M \cdot W) + (h \cdot \pi \cdot D \cdot L_M)]}{h \cdot \pi \cdot D \cdot K_M \cdot T_{Cycle}}$

Time Constant ( $T_{Constant}$ ):  $\frac{\rho_M \cdot C_M \cdot L_M^2}{K_M} \cdot (1 + \frac{2 \cdot W \cdot K_M}{h \cdot \pi \cdot D \cdot L_M})$

Pressure Drop:  $\frac{Df \cdot L}{D} \cdot \frac{\rho_C}{2} \cdot \dot{V}^2$

Equation for Graphs:  $y = ATM + (T_{MO} - ATM) \cdot e^{\frac{-x}{T_{Constant}}}$

## **Validation Cases:**

This folder contains two samples of data for use in the PACCMAN program. The data contained by the "xucomparison" file contains values from the 2001 paper: "The Design of Conformal Cooling Channels in Injection Molding Tooling" by Xiorong Xu, Emanuel Sachs, and Samuel Allen. "xucomparison.csv" is a table produced through WebPlotDigitizer of Figure 4 in the Xu paper. The data contained by the "guilongcomparison" file is from the 2010 paper "Analysis of thermal cycling efficiency and optimal design of heating/coolingsystems for rapid heat cycle injection molding process" by Wang Guilong, Zhao Guoqun, Li Huiping, and Guan Yanjin. "guilongcomparison.csv" contains the data from Figure 6 pulled with WebPlotDigitizer.

As seen in the .png files for each validation case, the data from both papers is extremely close to what the PACCMAN program calculates. This provides some affirmation that the program works both correctly and accurately.