Calibration

Purpose:

To find values of λj , Pgen, Pann that will give a simulated σ-ε curve that approximately matches the experimental curve.

(\*and then what?)

Tools given and used:

An Windows Executable was given.   
It takes in λj , Pgen, Pann as read from the “INPUT.txt” file in the same directory for 1000°C, 900°C, 800°C, 700°C, 600°C (i.e. 15 variables in total, three for each temperature.), and generate seven .txt file containing the simulated σ-ε expected to be given these variables (one .txt file for each of the above temperature at slow straining speed, and then two .txt file for 1000°C and 900°C at high straining speed respectively.)

Using Python to create a script called Supervisor.py, the process of running this .exe file is procedurized and automated as follows: first the values inside the INPUT.txt is changed; the executable file is then called; lastly the outputted .txt files are moved into a sub-folder and renamed to prevent overwriting of the 1000°C and 900°C simulated σ-ε .txt files, as well as for better organization.

Procedure:

The residual between the simulated curve and the experimental curve was found by the “Residual” subprogram in Fitter.py.

Using the module of scipy.Optimize.curve\_fit() in Controller.py, which imports and uses Supervisor.py to change the inputted variables in INPUT.txt to minimize the values obtained by the “Residual” subprogram in Fitter.py, the optimal values for λj, Pgen, Pann, were found for each temperature.

Since the lower and upper bounds are provided for each variable,

(4E-8<λj<1E-7; 1E-6<Pgen<1E-3; 1E-3<Pann<0.1)

We can be certain that scipy.Optimize.curve\_fit() will be able to find at least one value for each of these variables to minimize the residuals. The reason is that once bounds are given, scipy.Optimize.curve\_fit() will use the method of TRF to minimize the residual obtained, which always converges.

The resulting best fitting parameter is stored in the “Individually Optimized” directory.

As per secondary supervisor’s instruction, the experimental data for 600°C is regarded as invalid, thus the calibration was done against 1000°C (fast & slow), 900°C (fast & slow), 800°C (slow only), 700°C (slow only) only.

After the best fitting parameter was found for each of the temperatures, the Pgen and Pann for all temperature were chosen to be the same and then optimized by having the collective residuals for all temperatures minimized while tweaking Pgen and Pann (using a slightly altered version of the Controller.py script called ControllerTwoVar.py); while λj for all temperatures were fixed. The results were stored in “CollectiveOptimizationIteration1”

The best fitting λj for each temperature may have differed after Pgen and Pann were collectively changed. Therefore, λj for each temperature was found again using a second slightly altered version of the Controller.py script. The results were stored in “CollectiveOptimizationIteration1.5”.

The best fitting collective value of Pgen and Pann may also have changed after iteration 1.5. Therefore, the same procedure of fixing λj and then varying Pgen and Pann were done again, producing the results in “CollectiveOptimizationIteration2”.

The argument goes for “CollectiveOptimizationIteration2.5”, “CollectiveOptimizationIteration3”, “CollectiveOptimizationIteration3.5”.

The process of iterating ended at 3.5 , at which point the observed change between variables in 2.5 and 3.5 were at the 5th (or later) significant digit, and thus further iteration were considered as time-inefficient.

Results:

The λj obtained for each temperature are as below:

λj1000°C = 8.299086322111734e-08

λj900°C = 5.8310559074891014e-08

λj800°C = 4.726905702584226e-08

λj700°C = 4.000000000000001e-08

Note that the lower bound of λj for all temperatures was set to be 4E-8. Therefore, it is very likely that the 700°C case has hit the lower bound while the computer was attempting to tweak it, and thus this value cannot be trusted.

And the probability of generation and annihilation of dislocations are as follows:

Pgen = 1.0000000000008298e-06

Pann = 0.036954870741079456

Note that the lower bound was set to be 1E-6 for Pgen. Therefore, it is very likely that the program has hit the lower bound when attempting to optimize the result by tweaking Pgen.