Stress Relaxation in Prusa Filaments

Prepared for:

Dr. Ahmad Mohammadpanah Foroutaghe

Dr. Steven Rogak

MECH 305

Prepared by: Team #26

Bhumika Arora

Ethan Alexander

Kim Ristimaki

Hugh Rawling

10th April, 2022

Department of Mechanical Engineering

University of British Columbia

Abstract

The choice of what type of filament to use when 3-D printing significantly impacts the mechanical properties of the final product. Stress relaxation is one of those mechanical properties and is especially important when analyzing the long-term strength of mechanical components in a system. This project compared the stress relaxation behaviour of four common 3-D printing materials manufactured by Prusa in order to determine which one retained the largest stress at a steady state. The four materials are polylactic acid (PLA), polyethylene terephthalate glycol (PETG), acrylonitrile styrene acrylate (ASA) and polycarbonate (PC). Because the price of PC is $50 compared to the rest being $30, we also wanted to know if the price of the material had any relationship with the stress relaxation behaviour. So we also endeavoured to compare the cost-effectiveness of the best performing materials.

Using a load cell in series with the filament, we tensioned the specimen to 15 lbs and recorded the relaxation in stress for 30 minutes. Five trials were completed for each specimen (except ASA) one after the other on the same day to ensure room temperature did not vary significantly over the course of testing. Furthermore, the average room temperature throughout the whole testing process was consistent ensuring there was no bias in results. We fitted our data to a Prony series model in order to determine the steady-state stress characteristic of the specimen. This data is recorded as a percentage of the initial stress to let us compare performance between materials.

The results from our tests show us that PC had the highest steady-state stress characteristic, followed by PETG, ASA and finally PLA. Their performances are summarized in the graphic below. In terms of cost-effectiveness, PC performed 4% better than PETG despite PETG being 60% of the price of PC. PETG is the better option if there are budgetary concerns because it is more cost-effective. Figure 1 illustrates this point.

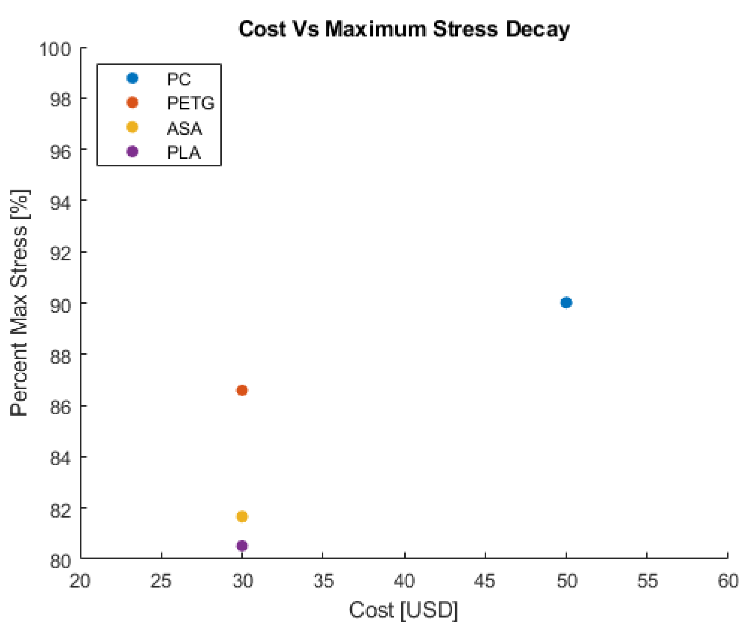


Figure 1: Graphic Abstract

Introduction

The motivation for this project arose when Ethan noticed a fault in his 3-D printed torsional springs as part of the filament feeding mechanism on his 3-D printer. Although they were functional when they were first made, after moderate use the coils would deform and greatly diminish their strength. So we wanted to know which material out of four Prusa filaments would retain the most of their strength over time. Those four materials are PC, PETG, PLA and ASA. These materials were chosen because they are popular filaments from Prusa. PC is the most expensive option, with a price of $50, out of the rest with a price of $30 each. For this reason, we also wanted to know if there were tradeoffs in performance with cost.

Our literature search uncovered a paper that looked at the change in stress concentration as a function of temperature[1] and print orientation[2] but none that directly compared material performance. However, we found a NASA report[3] that gave us an analytical model that we could use to fit our data and another paper that shows that this model is applicable to 3D printing materials[4].

There's a proportional relationship between the creep and stress relaxation of viscoelastic materials[3], giving us options on how we could approach testing the filaments in order to determine the best material. In the end, we still decided upon testing the stress relaxation because it would be much simpler and we would be able to collect more accurate results. A creep test would require very accurate instruments to measure such small changes in strain, which we were uncertain we could find.

Methods

The overall procedure of this project, as depicted in figure 2 below, involves 3 main steps. First, 3D printing filaments are physically tested using the apparatus, this is in tandem with data collection using an Arduino microcontroller. Finally, the data obtained is processed and analyzed using MATLAB according to techniques we designed ourselves and found from our literature review.

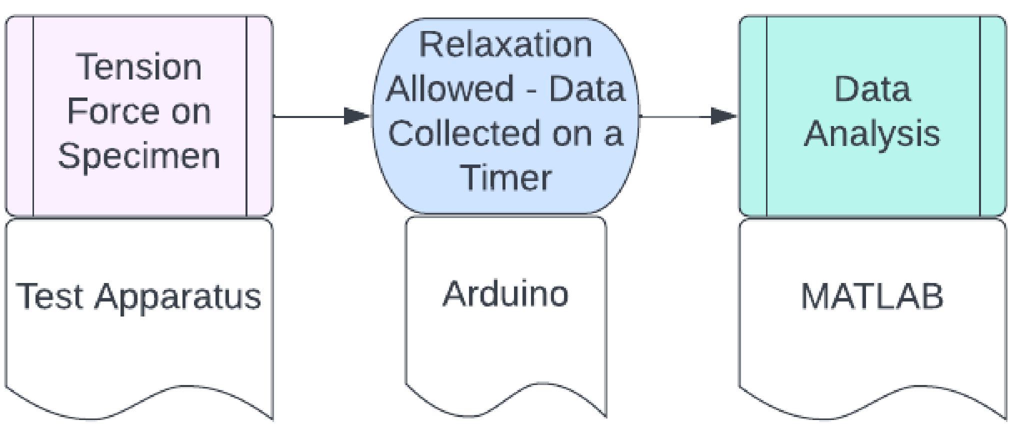


Figure 2: Experiment Procedure Overview - Flowchart

The physical portion of this experiment involved a test setup whose main components are labelled using callouts in the figure below.

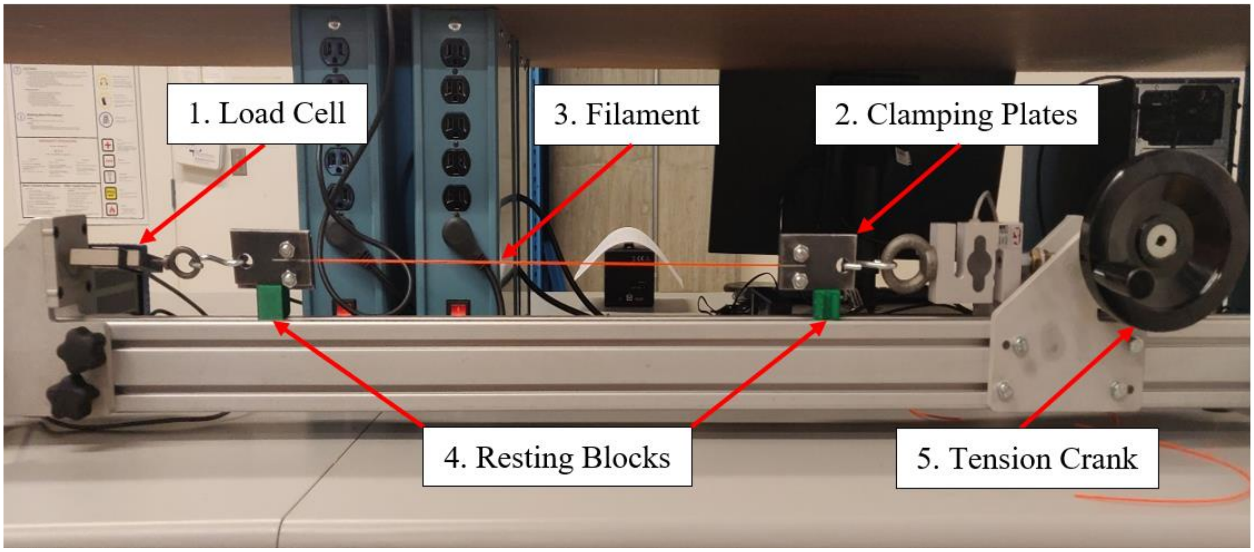


Figure 3: Test Apparatus Components in Final Build

The way this apparatus, as shown above, and its components were employed to extract the data is as follows:

1. **Load Cell:** A standard 50 lbs S-Load Cell, calibrated according to the procedures outlined on pages 12, 13 & 14 of the Mech 305 Solid Mechanics Experiment but we used a 46.7 lb calibration weight. As shown in figure 3, this load cell was mounted on the left-hand side of the rig. To ensure consistency, before and after each round of testing, the load cell was confirmed to be calibrated to within 0.1 lb.

2. **Clamping Plates:** On each side, two pairs of plates, machined according to the CAD Model in Appendix A, were used to clamp the filament between its grooves. On the left, the clamping plate was connected to the load cell. The pair on the right was attached to the tension crank. These plates were sandwich plates, which means that two separate plates were used to clamp the filament in between them and secured with bolts, this provided a secure, non-crushing grip on the filament. To ensure consistency, the securing step was standardized down to the number of turns on the bolts.

3. **Filament:** Four different types of commonly available PRUSA brand raw filaments were tested in this project. The materials were: PLA, PETG, ASA, and PC. To ensure consistency, all filaments used were raw since their cross-sectional area was more accurate and repeatable than anything printed. It was from freshly opened packs and was cut into 10-inch samples for a uniform length. All samples were brand new and thereby ensured to have no prior moisture absorption, which is known to cause brittleness in 3D printing filaments.

4. **Resting Blocks:** Two resting blocks, 3D printed cubes, were placed below each clamping plate pair. This was to keep their weight from impacting the tension through the filament and prevent sagging due to the weight of the plates. There was concern about this while building the rig because breaking the filament in procedural mistakes, or otherwise damaging it, needed to be strictly avoided and minimized. To ensure consistency, even the orientation of how these blocks were placed was noted in the first test and kept the same for the ones that followed.

5. **Tension Crank:** The tension crank, which is located at the far-right end of the rig, was available to us in KAIS 1160 and was used to apply tension to the filaments. Once again, to ensure consistency efforts were made to keep the load applied to between 15 to 16 lbs each time to keep the relaxation profiles the same.

Combining the standardized components and preparations above, the test setup was employed to obtain data on the stress relaxation in the filaments. Each test was run for a minimum of 30 minutes. Even the temperature in the room - the variation during testing was insignificant and multiple materials were tested on overlapping days. One person was present at all tests to ensure that all tests were done consistently and accurately to minimize errors. Determining how to standardize these steps was done through trial and error. We chose the 15 lb testing load because there were issues where the filament samples were failing early at higher loads, and we wanted to ensure that this was not a factor in our tests.

Overall, filaments were clamped within the rig, tension was applied by turning the tension crank, and the relaxation in the filaments was observed for 30 minutes. Data was collected through a load cell connected to an Arduino that streamed the data to a Matlab program to log it. Finally, by following this method, we obtained raw data as a force in pounds.

Results

From our testing we were able to successfully rank all the materials. The plot below shows the different materials extrapolating the results out to 3 hours using the fit formula discussed in the analysis section below. While they decay at different rates after 9 minutes the ranking is the same as the eventual steady states of the materials. In table 1, below all the materials are ranked in order and have their average steady-state percent force listed alongside.

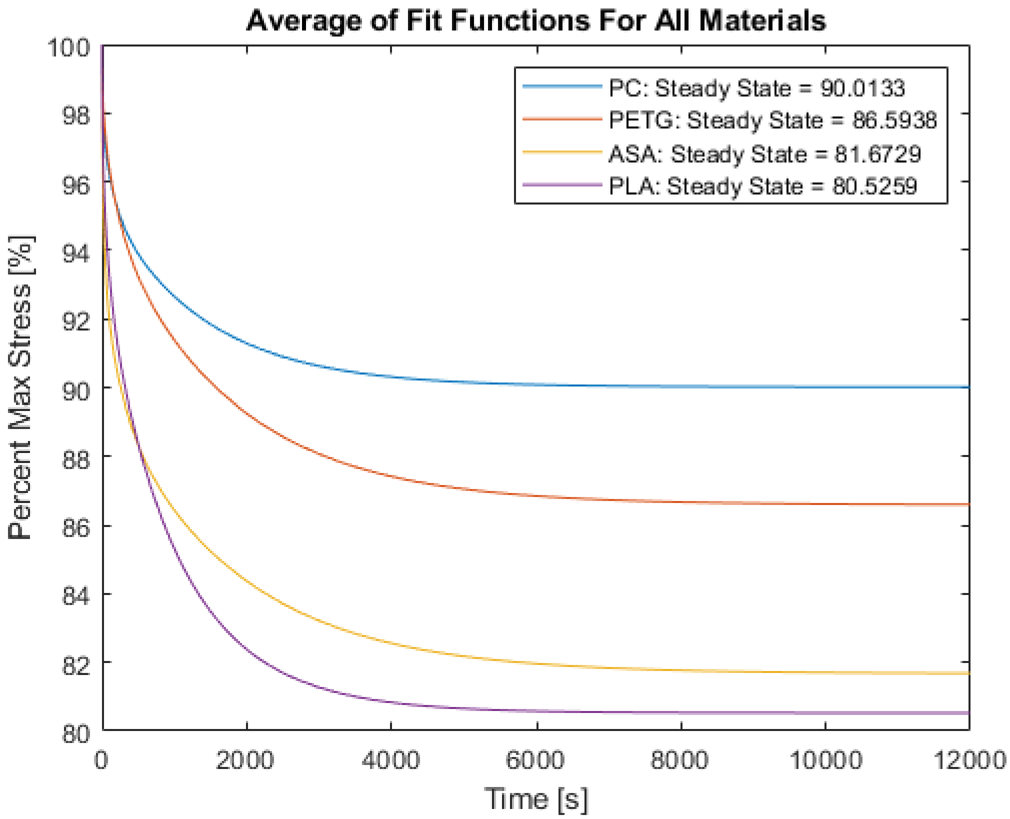


Figure 4: Averaged Prony fit for all materials tested

Table 1: Ranking of materials, units in % of full stress

Material	**PC**	**PETG**	**ASA**	**PLA**
Ranking	1	2	3	4
Trial 1	89.8	86.9	82.0	78.6
Trial 2	90.2	86.2	79.5	80.1
Trial 3	90.8	87.5	82.7	82.4
Trial 4	89.1	84.2	82.5	81.1
Trial 5	90.1	88.2	N/A	80.4
Average	90.0	86.6	81.7	80.5
Standard Deviation	0.6	1.5	1.4	1.4

Analysis

After collecting the raw data, we first needed to find a way to normalize all the data collected and decide what metric we would use to compare the materials. We chose to use a prediction of the steady-state stress at time equals infinity because this is what fits our application the best. Since all the materials had the same diameter, the stress is proportional to the force. The filament measurements are presented below in table 2. There was also the issue where the peak force would vary slightly between the tests, so not all the tests would start at the same point within the materials. To remove this variation, we changed all the forces into a percent full force that allowed all the tests to start at one and decay to a consistent steady-state. We decided that we wanted to use the hypothetical steady state of the material when time is infinity. As discussed earlier we used a Prony series which is the sum of several exponential decay functions with the formula shown below.

$$\sigma(t) = 100 + \sum_{n=1}^{6} \left\{ a(n) \cdot \left[e^{b(n) \cdot t} - 1 \right] \right\}$$

We chose to use a Prony series of length six because the fit was better than length four and the computing model did not take too long. This model allows for several time constants in the material and will enable you to find the steady-state of the material by subtracting the sum of the **a** coefficients from the initial value. They used a computer to find the coefficients in the paper, so we did the same. We decided to use a Prony series of length six because when we fit with fewer, the curve did not match the raw data as closely as we wanted. This fit was applied to all the material tests individually, so we got around five steady-state values for each material. We then performed a T-test between all the materials with a confidence value of 0.01, and if they were unique, we determined which had the higher mean and ranked it higher. To view the raw data refer to appendix B, and to view the fits of each of the tests refer to appendix C.

Table 2: Summary of raw filament data

Material	Diameter (mm)	Standard Deviation (um)	Ovality (%)
PLA	1.75 +/- 0.012	3.42	1.2
PC	1.75 +/- 0.023	7.43	2.2
PETG	1.75 +/- 0.011	3.47	1.8
ASA	1.75 +/- 0.019	5.03	1.2

Discussion

Our data analysis shows that the tested polymers do perform differently in stress relaxation, and can be clearly ranked in terms of their ability to retain stress. PC retains the greatest proportion of the initial stress at 90.0% while PLA performs the worst at 80.5%. A T-test upholds this result with a confidence value of 0.01. As expected, the force-time curves generated from our testing could be accurately modeled using a Prony series, which is in line with the results of our research into stress relaxation in viscoelastic materials. This gives additional evidence that our results are valid.

One of the motivations of our experiment was to determine if the more expensive types of 3D printing filament perform better than the cheaper polymers. Of the materials tested, PC is the most expensive, with a price ⅔ more than the other materials. From the data analysis, we can say with certainty that PC does perform the best of the materials tested by a substantial margin but it is not necessarily enough to justify the price increase. The PC samples still relaxed by about 10% of the initial load so while it is better than the other tested materials it is still not suitable for the desired application. This makes it difficult to justify the increased cost of PC for use in springs.

There are a couple of sources of error in this experiment that could be addressed easily given additional time. The first is that due to the relatively short characteristic times of stress relaxation for plastic samples, the time it took to manually apply load with the tensile setup would allow some stress relaxation to occur at force values that were lower than the intended initial force. If the force was not applied quickly, the force-time curve would be offset in a way that was difficult to correct for during the analysis. The second issue is that while setting the initial force in the sample manually, it is difficult to achieve a consistent initial force value. Both of these problems could be solved by creating a system to automatically set the initial force in the sample. This could be achieved easily using a servo motor attached to the power screw mechanism in the tensile tester using feedback from the load cell.

Conclusion

The following list is the performance of the filaments, ranked from best to worst: PC, PETG, ASA, PLA. Paired t-tests allow us to say with high confidence ($p = 0.01$) that PC has the highest steady-state value for stress relaxation. The Prony series was a very accurate model to determine the stress relaxation behavior of viscoelastic materials such as 3-D printing filament. Given the relative rankings of performance, PETG might be a viable alternative to PC if a cheaper option is more desirable. Alternatively, we can say that 40% of the price of PC accounts for a 4% increase in performance compared to PETG.

Personal Contributions and Reflections

Ethan

I adapted the data collection scripts from the Thermofluids and solid mech labs to allow for efficient and simple data collection. I also created the data processing script that automatically imported, cleaned, processed, analyzed, and plotted all the results. I designed the clamping plates that we used to secure the filaments and acquired the tooling required to make them. I was also present during all data collection sessions to make sure that everything was set up consistently to make sure that the data was consistent. To improve the capstone project I think that the abstract proposal should be due earlier and reviewed earlier. This earlier due date would allow groups to select their experiment earlier and enable them to start collecting materials for their experiments earlier.

Kim

I purchased materials that were used to assemble the test apparatus (i.e. bolts, s-hooks, washers) and helped to assemble it. I also spent several hours collecting raw data. We decided to proceed with this project as it was the most practical out of the other concepts we explored. The data would be easy to interpret and the apparatus and test procedure was intuitive. Given the short time span given to us, this was the best option. For the future, MECH 305 should tell us quicker the available test apparatuses we can use provided by the university, as that would guide decision-making for student-teams as to what tests would be technically feasible or not.

Hugh

I manufactured the clamping plates we used to hold the filament in the test apparatus and helped with the data collection. While we were planning how to carry out the experiment we came up with some ideas to make a custom test apparatus, which became unnecessary after the equipment in the student lab was made available to us. This wasn't a big problem but it did cost us some time. Having access to a comprehensive list of available materials and test equipment before we chose a project would help with project planning in future iterations of the course.

Bhumika

I primarily focused on an extensive literature review on the materials used to make the filaments and to determine the relaxation and creep moduli of the same material for more background and feasibility regarding our project. While there were many papers on the general topic, nothing compared the available filaments directly so it was concluded that there is a need for such a project. The other significant task to which I contributed was data collection . I was responsible for the timeline and project planning during the first phase of the project, and once we moved onto the data collection phase I also spent time in set up and several hours in collecting data. Time permitting, I would've liked to figure out how to test filament that had been preheated as well. This is because this would've been its actual operating temperature and it would be interesting and important to see if our results translated to heated filaments as well. Towards the final stage of the project, I drew up the skeleton for our presentation and then contributed just like each member to the presentation and reports once we divided up the work. Overall, even though a more out of the box topic that's more personally relevant would have been interesting to select while we had the course framework supporting us, I see a lot of value in our results and know that they will definitely be of use in our many upcoming 3D printing experiences. Additionally, since this is the part of the course I enjoyed the most but was most limited in terms of things to do, if Mech 305 were to cut back on labs and introduce the project earlier a more extensive and complex experiment could've been conducted as well; I would value that greatly.

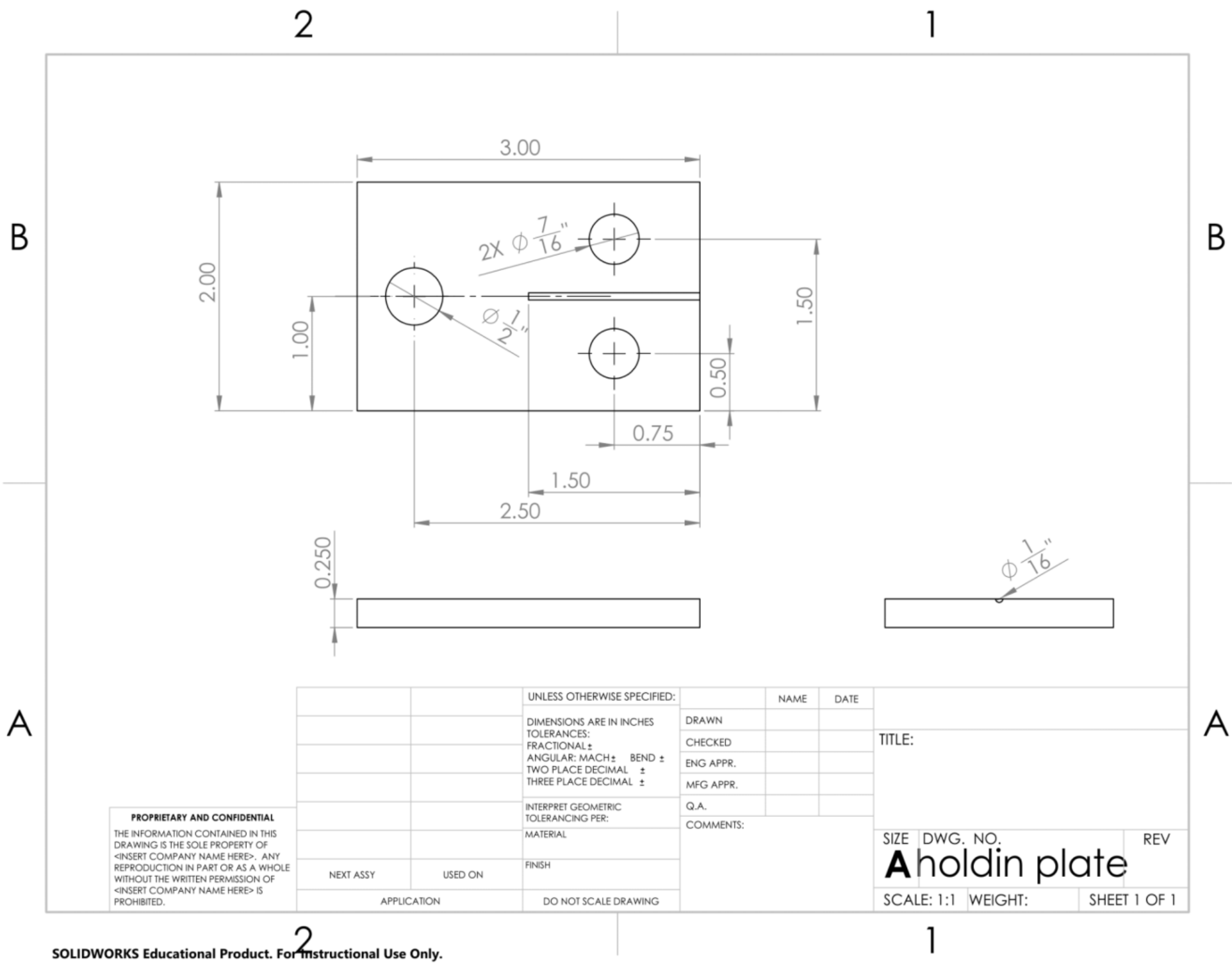
Acknowledgements

We would like to acknowledge the support of the MECH department, Sean Buxton and Simon Szoke for guiding us through our project. They provided advice and equipment which we used to make our experiments successful.

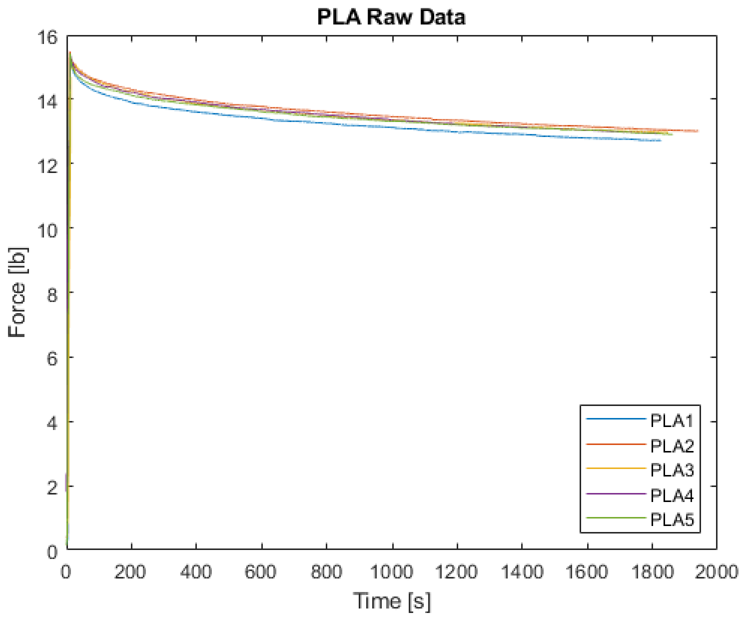
References

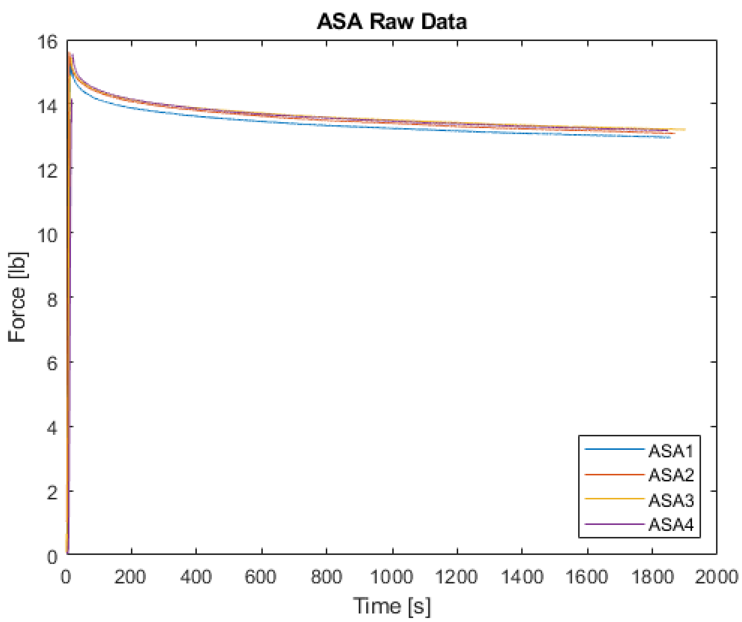
1. Chen, T., 2000, “Determining a Prony Series for a Viscoelastic material From Time Varying Strain Data”, U.S. Army Research Laboratory, https://ntrs.nasa.gov/api/citations/20000052499/downloads/20000052499.pdf
2. Ibrulj, J., Obucina, M., and Dzaferovic, E., 2020, “Modeling of Viscoelastic Behaviour of 3D Printed Polymeric Materials”, DAAAM International,https://www.researchgate.net/publication/341142275_Determination_of_Relaxation_and_Creep_Modulus_of_Polymer_Materials_Obtained_by_3D_Printing
3. Zhang, H., 2016, “Characterization of Tensile, Creep, and Fatigue Properties of 3S Printed Acrylonitrile Butadiene Styrene”, Purdue University, https://scholarworks.iupui.edu/bitstream/handle/1805/11024/ZHANG_HANYIN_THESIS.pdf?sequence=1&isAllowed=y
4. Dusunceli, N., Drozdov, A. D., Theilgaard, N., 2016, “Effects of Temperature on the Relaxation Behavior of Poly(lactic acid)”, Society of Plastics Engineers, https://www.researchgate.net/publication/309680865_Effects_of_temperature_on_the_relaxation_behavior_of_polylactic_acid

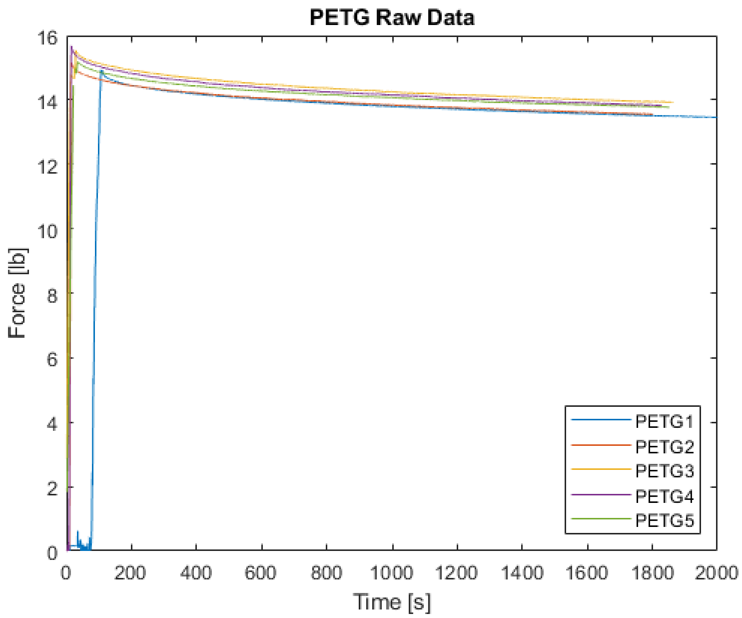
Appendix A - Clamping Plate CAD Drawing

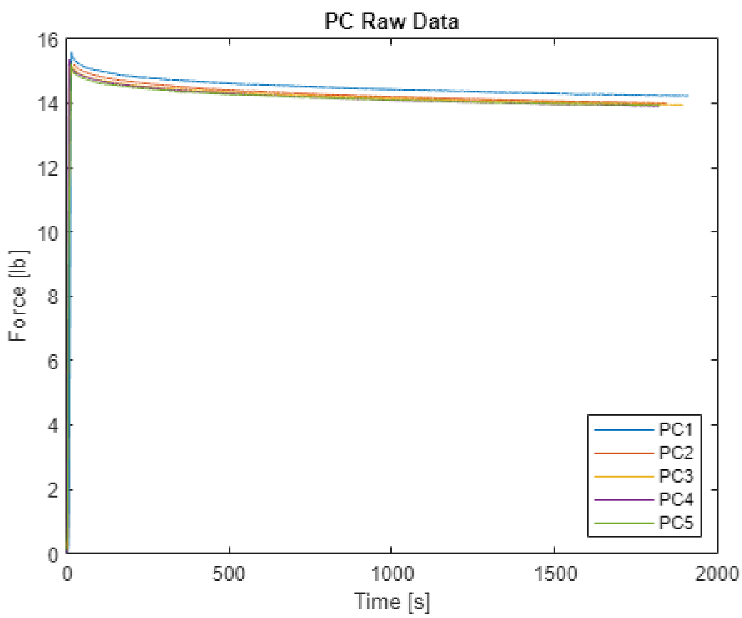


Appendix B - Raw Data Plots

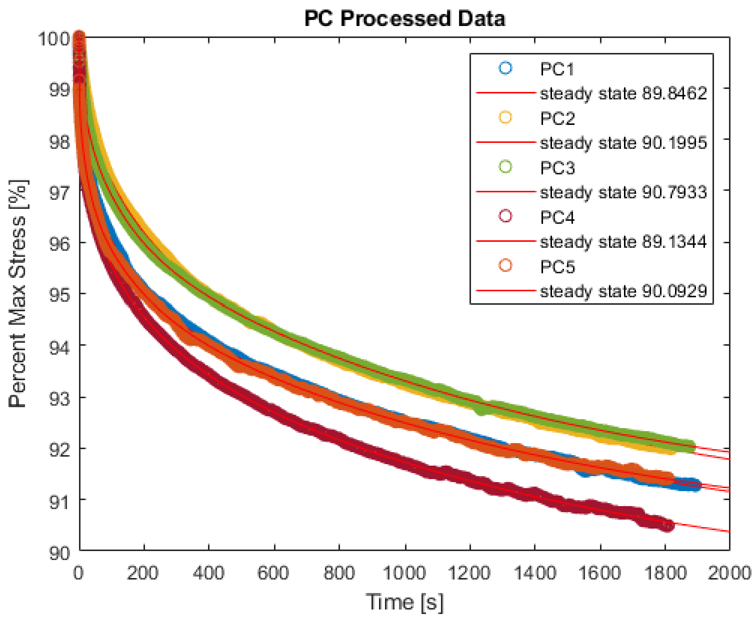





PETG Raw Data
Force [lb]
Time [s]
PETG1
PETG2
PETG3
PETG4
PETG5
0
2
4
6
8
10
12
14
16
0
200
400
600
800
1000
1200
1400
1600
1800
2000


PC Raw Data
Force [lb]
Time [s]
PC1
PC2
PC3
PC4
PC5
0
2
4
6
8
10
12
14
16
0
500
1000
1500
2000

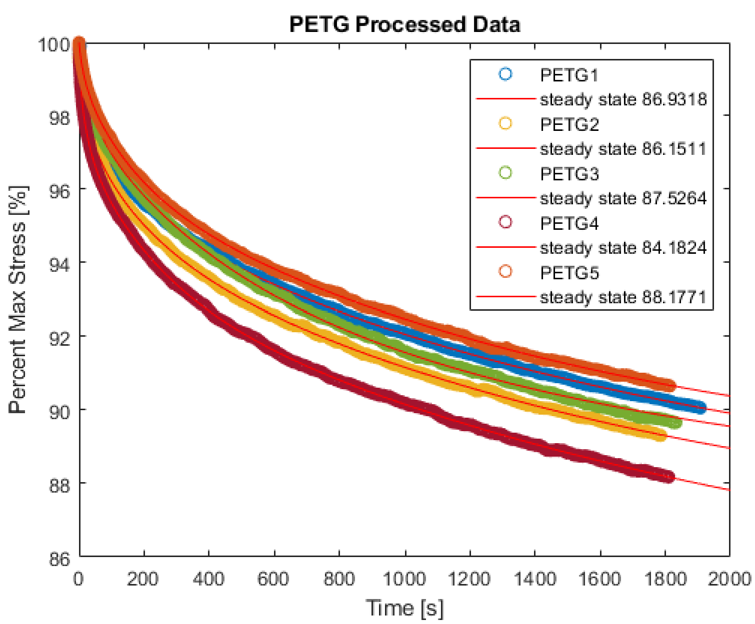
Appendix C - Processed Data




```
PC1
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =        5.489  (5.484, 5.495)
       b =   -0.0007163  (-0.0007226, -0.00071)
       c =        2.223  (1.89, 2.557)
       d =    -0.007758  (-0.008248, -0.007268)
       e =        1.546  (1.422, 1.669)
       f =     -0.09565  (-0.1028, -0.0885)
       g =       0.4249  (-872.3, 873.2)
       h =     -0.02136  (-1.116, 1.074)
       k =       0.3343  (0.3002, 0.3684)
       l =      -0.8015  (-0.9249, -0.678)
       m =       0.1366  (-872.7, 872.9)
       n =     -0.02244  (-3.522, 3.477)
gof = struct with fields:
           sse: 8.8062
       rsquare: 0.9998
           dfe: 18885
    adjrsquare: 0.9998
          rmse: 0.0216
PC2
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =        5.884  (5.877, 5.892)
       b =   -0.0006568  (-0.0006627, -0.0006508)
       c =        2.537  (2.523, 2.552)
       d =    -0.005629  (-0.005699, -0.005559)
```

```
       e =       1.078  (-717.2, 719.3)
       f =    -0.03668  (-1.309, 1.236)
       g =      0.1903  (-718.1, 718.4)
       h =    -0.03664  (-7.24, 7.167)
       k =     0.06698  (-1516, 1516)
       l =     -0.2312  (-130.9, 130.4)
       m =     0.04424  (-1516, 1516)
       n =     -0.2197  (-195.9, 195.5)
gof = struct with fields:
           sse: 4.7373
       rsquare: 0.9999
           dfe: 18156
    adjrsquare: 0.9999
          rmse: 0.0162
PC3
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       5.603  (5.601, 5.606)
       b =  -0.0008011  (-0.0008039, -0.0007984)
       c =       2.164  (2.149, 2.178)
       d =   -0.008349  (-0.008422, -0.008275)
       e =       1.155  (0.5936, 1.717)
       f =    -0.05514  (-0.06498, -0.0453)
       g =      0.2183  (-1668, 1669)
       h =     -0.1489  (-24.89, 24.6)
       k =     0.03909  (-0.01981, 0.09798)
       l =      -1.184  (-3.178, 0.8107)
       m =     0.02694  (-1668, 1668)
       n =     -0.1425  (-202.8, 202.5)
gof = struct with fields:
           sse: 5.3183
       rsquare: 0.9999
           dfe: 18704
    adjrsquare: 0.9999
          rmse: 0.0169
PC4
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       5.336  (5.303, 5.369)
       b =  -0.0007295  (-0.0007557, -0.0007033)
       c =       1.904  (1.714, 2.094)
       d =   -0.004643  (-0.005165, -0.004121)
       e =       1.107  (0.8222, 1.391)
       f =    -0.03779  (-0.04942, -0.02615)
       g =      0.8726  (0.6673, 1.078)
       h =     -0.5092  (-0.5852, -0.4332)
       k =      0.7419  (0.61, 0.8738)
       l =     -0.1484  (-0.2015, -0.09536)
       m =      0.9042  (0.6758, 1.133)
       n =    -0.01402  (-0.01972, -0.008322)
gof = struct with fields:
           sse: 10.7140
       rsquare: 0.9998
           dfe: 18041
    adjrsquare: 0.9998
```

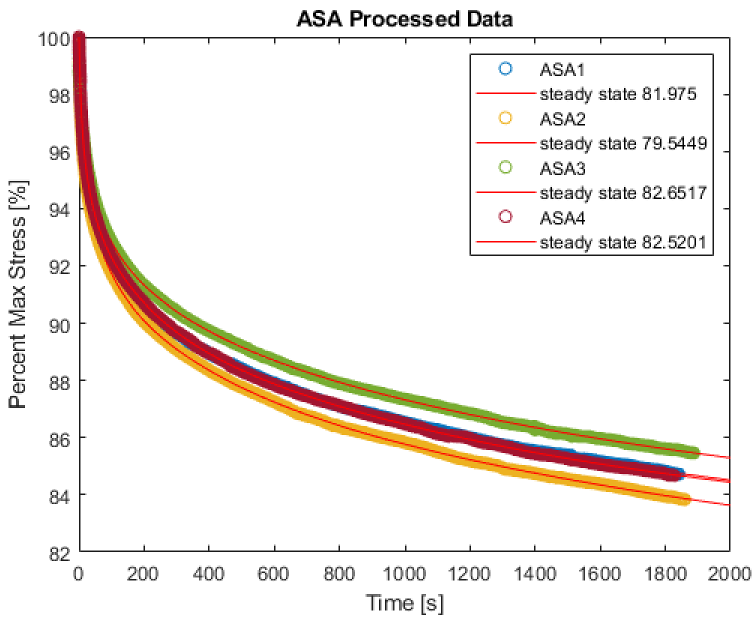
```
           rmse: 0.0244
PC5
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       5.082  (5.073, 5.09)
       b =  -0.0007511  (-0.0007598, -0.0007423)
       c =       2.003  (1.985, 2.021)
       d =    -0.00676  (-0.006888, -0.006632)
       e =       1.398  (1.361, 1.435)
       f =     -0.0403  (-0.04208, -0.03851)
       g =      0.6779  (-1.401e+04, 1.401e+04)
       h =      -1.664  (-338.7, 335.4)
       k =      0.4636  (0.3083, 0.619)
       l =     -0.2712  (-0.371, -0.1714)
       m =      0.2832  (-1.401e+04, 1.401e+04)
       n =      -1.632  (-809.8, 806.5)
gof = struct with fields:
            sse: 20.6549
        rsquare: 0.9994
            dfe: 18032
     adjrsquare: 0.9994
           rmse: 0.0338
```




```
PETG1
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.728  (8.704, 8.753)
       b =  -0.0005367  (-0.0005495, -0.000524)
       c =       1.875  (1.782, 1.968)
       d =   -0.004189  (-0.004497, -0.00388)
       e =       1.034  (0.8204, 1.248)
       f =    -0.01441  (-0.018, -0.01081)
       g =      0.6765  (-482.2, 483.5)
       h =     -0.1344  (-5.451, 5.182)
       k =      0.4436  (-479.3, 480.2)
```

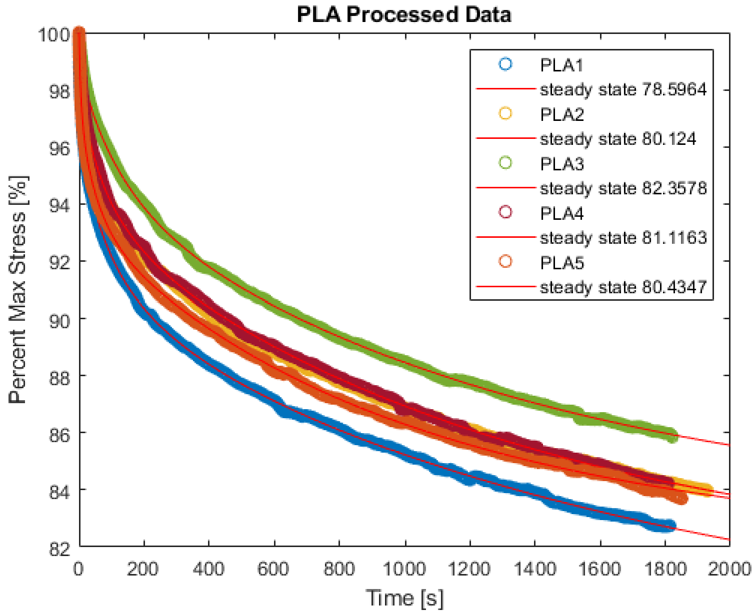
```
       l =     -0.1187  (-9.485, 9.248)
       m =      0.3105  (-2.639, 3.26)
       n =    -0.04943  (-0.2475, 0.1486)
gof = struct with fields:
           sse: 8.4052
       rsquare: 0.9999
           dfe: 19025
    adjrsquare: 0.9999
          rmse: 0.0210
PETG2
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.896
       b =  -0.0005761
       c =       2.644
       d =   -0.005152
       e =       1.122
       f =    -0.02586
       g =       1.051
       h =     -0.1454
       k =     0.08116
       l =      -76.48
       m =     0.05511
       n =      -192.7
gof = struct with fields:
           sse: 13.5558
       rsquare: 0.9998
           dfe: 17803
    adjrsquare: 0.9998
          rmse: 0.0276
PETG3
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       6.967  (-2461, 2475)
       b =  -0.0006472  (-0.02052, 0.01923)
       c =       3.244  (-1.217, 7.705)
       d =   -0.002745  (-0.004248, -0.001242)
       e =       1.387  (1.292, 1.481)
       f =    -0.02301  (-0.02594, -0.02009)
       g =      0.3956  (0.1152, 0.676)
       h =     -0.4612  (-0.7298, -0.1925)
       k =      0.3256  (-2466, 2467)
       l =  -0.0005402  (-0.416, 0.415)
       m =      0.1549  (-0.05136, 0.3612)
       n =     -0.1165  (-0.3495, 0.1165)
gof = struct with fields:
           sse: 82.0225
       rsquare: 0.9991
           dfe: 18290
    adjrsquare: 0.9991
          rmse: 0.0670
PETG4
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
```

```
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =        9.71  (9.656, 9.764)
       b =  -0.0004906  (-0.0004998, -0.0004815)
       c =       3.453  (3.433, 3.473)
       d =    -0.00469  (-0.004768, -0.004612)
       e =       1.145  (-1107, 1109)
       f =    -0.03865  (-1.121, 1.044)
       g =      0.7843  (0.6624, 0.9061)
       h =     -0.2037  (-0.2626, -0.1448)
       k =      0.5057  (-1107, 1108)
       l =    -0.03651  (-2.273, 2.2)
       m =      0.2201  (0.0572, 0.383)
       n =     -0.8487  (-1.367, -0.3306)
gof = struct with fields:
           sse: 15.0087
       rsquare: 0.9998
           dfe: 18071
    adjrsquare: 0.9998
          rmse: 0.0288
PETG5
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.355  (8.344, 8.365)
       b =  -0.0006653  (-0.0006694, -0.0006611)
       c =       2.511  (2.5, 2.522)
       d =   -0.006405  (-0.006471, -0.00634)
       e =      0.7481  (0.6879, 0.8082)
       f =    -0.04675  (-0.05048, -0.04302)
       g =      0.1926  (0.137, 0.2481)
       h =     -0.1914  (-0.2811, -0.1018)
       k =    0.009938  (-36.75, 36.77)
       l =      -4.866  (-2596, 2586)
       m =    0.006925  (-36.73, 36.75)
       n =      -3.678  (-2722, 2715)
gof = struct with fields:
           sse: 11.5063
       rsquare: 0.9999
           dfe: 18105
    adjrsquare: 0.9999
          rmse: 0.0252
```




```
ASA1
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       7.884  (7.869, 7.9)
       b =   -0.000567  (-0.0005786, -0.0005554)
       c =       3.282  (3.258, 3.307)
       d =    -0.00389  (-0.003976, -0.003803)
       e =       3.083  (2.796, 3.369)
       f =    -0.01982  (-0.0209, -0.01874)
       g =       1.406  (1.238, 1.573)
       h =    -0.05058  (-0.06243, -0.03873)
       k =       1.702  (1.51, 1.894)
       l =     -0.1562  (-0.1801, -0.1324)
       m =      0.6684  (0.5259, 0.8109)
       n =     -0.5884  (-0.6716, -0.5052)
gof = struct with fields:
           sse: 5.5858
       rsquare: 0.9999
           dfe: 18369
    adjrsquare: 0.9999
          rmse: 0.0174
ASA2
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.899  (8.694, 9.105)
       b =  -0.0003902  (-0.0004225, -0.0003579)
       c =       3.885  (3.781, 3.99)
       d =   -0.002962  (-0.003088, -0.002837)
       e =       3.334  (3.268, 3.4)
       f =    -0.01513  (-0.01565, -0.01462)
       g =        2.07  (1.319, 2.821)
       h =    -0.06888  (-0.08272, -0.05505)
       k =       1.242  (0.5423, 1.941)
       l =     -0.1587  (-0.2237, -0.09368)
```

```
       m =       1.025  (0.8475, 1.202)
       n =     -0.7099  (-0.8035, -0.6163)
gof = struct with fields:
           sse: 14.4016
       rsquare: 0.9999
           dfe: 18549
    adjrsquare: 0.9999
          rmse: 0.0279
ASA3
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.081  (8.06, 8.102)
       b =  -0.0005592  (-0.0005714, -0.000547)
       c =       3.062  (3.036, 3.089)
       d =   -0.004043  (-0.00415, -0.003937)
       e =       3.015  (2.967, 3.062)
       f =    -0.01803  (-0.01855, -0.01751)
       g =      0.4222  (0.1578, 0.6866)
       h =      -0.227  (-0.3252, -0.1288)
       k =       1.703  (1.472, 1.934)
       l =    -0.08227  (-0.09219, -0.07236)
       m =       1.065  (1.03, 1.1)
       n =      -3.434  (-3.709, -3.16)
gof = struct with fields:
           sse: 10.8697
       rsquare: 0.9999
           dfe: 18815
    adjrsquare: 0.9999
          rmse: 0.0240
ASA4
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.036  (8.027, 8.044)
       b =   -0.000715  (-0.0007235, -0.0007065)
       c =       3.733  (3.607, 3.859)
       d =   -0.005672  (-0.005844, -0.0055)
       e =       2.661  (2.139, 3.183)
       f =    -0.04462  (-0.05291, -0.03634)
       g =       1.419  (-2290, 2292)
       h =     -0.1925  (-5.489, 5.104)
       k =      0.9689  (0.3755, 1.562)
       l =    -0.01932  (-0.0272, -0.01144)
       m =      0.6618  (-2290, 2292)
       n =     -0.1859  (-11.54, 11.17)
gof = struct with fields:
           sse: 10.3831
       rsquare: 0.9999
           dfe: 18251
    adjrsquare: 0.9999
          rmse: 0.0239
```




```
PLA1
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       11.98  (11.94, 12.03)
       b =  -0.0005931  (-0.0006087, -0.0005776)
       c =       2.716  (2.455, 2.977)
       d =   -0.005202  (-0.005696, -0.004709)
       e =       2.913  (2.679, 3.148)
       f =    -0.01494  (-0.01643, -0.01345)
       g =       2.317  (2.012, 2.623)
       h =    -0.07533  (-0.08675, -0.0639)
       k =      0.8428  (0.5713, 1.114)
       l =     -0.2654  (-0.4195, -0.1112)
       m =      0.6322  (0.3484, 0.916)
       n =      -1.259  (-1.78, -0.7375)
gof = struct with fields:
           sse: 50.5093
       rsquare: 0.9997
           dfe: 18096
    adjrsquare: 0.9997
          rmse: 0.0528
PLA2
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.313  (-354.3, 370.9)
       b =  -0.0007413  (-0.007827, 0.006345)
       c =       4.211  (-341.3, 349.8)
       d =  -0.0004169  (-0.01643, 0.01559)
       e =        3.63  (3.259, 4)
       f =   -0.006259  (-0.006846, -0.005673)
       g =       1.693  (-2263, 2266)
       h =    -0.03886  (-0.5975, 0.5197)
       k =       1.217  (1.156, 1.279)
       l =     -0.3244  (-0.3444, -0.3044)
```

```
       m =      0.8121  (-2264, 2265)
       n =     -0.0397  (-1.236, 1.156)
gof = struct with fields:
           sse: 31.9418
       rsquare: 0.9998
           dfe: 19240
    adjrsquare: 0.9998
          rmse: 0.0407
PLA3
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.276  (-184.3, 200.8)
       b =  -0.0009137  (-0.007372, 0.005545)
       c =       4.125  (-159.8, 168.1)
       d =  -0.0003943  (-0.01535, 0.01456)
       e =       2.681  (-1905, 1910)
       f =   -0.007829  (-0.07679, 0.06113)
       g =       1.136  (-1907, 1909)
       h =   -0.007662  (-0.1742, 0.1589)
       k =      0.9327  (0.7555, 1.11)
       l =    -0.05585  (-0.07067, -0.04102)
       m =      0.4924  (0.2429, 0.7419)
       n =     -0.1803  (-0.2374, -0.1231)
gof = struct with fields:
           sse: 42.9661
       rsquare: 0.9997
           dfe: 18159
    adjrsquare: 0.9997
          rmse: 0.0486
PLA4
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
     Coefficients (with 95% confidence bounds):
       a =       8.496  (-8571, 8588)
       b =  -0.0007324  (-0.04762, 0.04616)
       c =       3.955  (-8578, 8586)
       d =  -0.0008244  (-0.1005, 0.09889)
       e =       3.208  (2.951, 3.464)
       f =   -0.009211  (-0.01019, -0.008227)
       g =       1.633  (1.39, 1.876)
       h =    -0.02939  (-0.03626, -0.02251)
       k =      0.9672  (-8033, 8035)
       l =     -0.1723  (-16.54, 16.19)
       m =      0.6253  (-8033, 8034)
       n =     -0.1684  (-25.3, 24.97)
gof = struct with fields:
           sse: 91.7376
       rsquare: 0.9995
           dfe: 18059
    adjrsquare: 0.9995
          rmse: 0.0713
PLA5
curve =
     General model:
     curve(x) = 100+a*(exp(b*x)-1)+c*(exp(d*x)-1)+e*(exp(f*x)-1)+g*(exp(h*x)-
                    1)+k*(exp(l*x)-1)+m*(exp(n*x)-1)
```

```
     Coefficients (with 95% confidence bounds):
       a =       7.934  (1.169, 14.7)
       b =     -0.0013  (-0.001766, -0.0008349)
       c =       5.096  (4.51, 5.681)
       d =    -0.00032  (-0.001409, 0.0007687)
       e =       3.607  (3.4, 3.814)
       f =    -0.01772  (-0.0188, -0.01663)
       g =       1.385  (-32.15, 34.92)
       h =      -0.141  (-0.8306, 0.5486)
       k =       0.983  (-32.74, 34.7)
       l =    -0.09851  (-0.7119, 0.5149)
       m =      0.5611  (0.08623, 1.036)
       n =     -0.8136  (-1.392, -0.2348)
gof = struct with fields:
           sse: 140.7435
       rsquare: 0.9991
           dfe: 18461
    adjrsquare: 0.9991
          rmse: 0.0873
```