2025
International Internship Pilot Program in Taiwan

Final Report

Name：Fairchild, Reese

Internship Inst.：Tunghai University

Name of PI/Department/School：Ta-Chen Lin, Department of Chemical and Materials Engineering, Tunghai University

Submission Date：2025/08/30 (August 30th, 2025)

Notes for Use

(Added Sept. 4th, 2025 by Reese Owen Fairchild)

- This is an unpublished, unreferred research report, outlining the results of a summer research internship.
- This internship was graciously funded by the International Internship Pilot Program (IIPP), which is held by the National Science and Technology Council (NSTC) of Taiwan. The author of this report would like to kindly thank the NSTC for their funding.
- This internship was conducted under the guidance of Dr. Ta-Chen Lin of Tunghai University, whom the author would like to kindly thank.
- In accordance with the IIPP agreement, this work, and other work produced during the internship, may be used or may authorized to be used by the NSTC or relevant parties.
- The author, to the best of their ability, guarantees that this final report does not infringe upon the intellectual property rights or related rights of others and does not violate moral customs, academic ethics, or the laws of Taiwan.

Biochemical Process Optimization with Simulated Annealing and Artificial Neural Networks

I. Introduction

A. Overview

1. Problem

Biochemical process optimization involves finding the optimal conditions in order to yield more favorable results in a biochemical process. Our work involved optimizing Polyhydroxyalkanoate (PHA) production based on two factors: concentration (PCC) and content (PCN). We work to increase the concentration as much as possible (measured in g/L) and increase the content percentage as close to 100% as possible to produce high quality PHAs. These PHAs have potential usage in producing compostable, biodegradable plastics, a good alternative to other plastics that stay around as pollutants for a long time.

2. Datasets

Our optimization involved two separate datasets from experiments of the chemical process: batch and fed-batch. Batch data was gathered by adding all ingredients at once. This dataset has five unique independent variables. Fed-batch data is gathered by adding reactants slowly over a period of time; this dataset's entries only vary based on time, giving us only one independent variable.

3. General Process

Our general process for optimization involves fitting our dataset to a regression model, giving us the coefficients for an equation that we may use to estimate the dependent variable values based on some independent variable values. Then, we use simulated annealing (SA) in order to try a large number of independent variable values, calculate the resulting dependent variable values through an objective function, and keep the best result. This is done three times; one annealing run optimizes based on PHA content (PCN), one based on PHA concentration (PCC), and one partially based on both values (referred to as "Mix").

B. Related Works

Our approaches were informed by previous experiments involving artificial neural networks (ANNs). Khanlou et al.[1] used artificial neural networks to optimize fiber

[1] Khanlou, H.M., Sadollah, A., Ang, B.C. et al. (2014) Prediction and optimization of electrospinning parameters for polymethyl methacrylate nanofiber fabrication using response surface methodology and artificial neural networks. *Neural Computing & Applications, 25*. https://doi.org/10.1007/s00521-014-1554-8

diameter of electrospun fiber, showing the potential for ANN usage, and Pen et al.[2] used an ANN to optimize fed-batch fermentation, much like our fed-batch dataset optimization. Lastly, Elmeligy et al.[3] used artificial neural networks to help optimize biobutanol production in a multiple objective optimization problem much like ours.

II. Optimization Improvement Methods

A. Outlier Removal

One method to improve our process is to remove outlier data from our dataset. By comparing the residuals from our regression models (which are generated using MATLAB's "fitlm" function), we identified which datapoints were fit worst to our model, allowing us to remove them from the dataset. We did this by generating scatterplot graphs of the residual data (such as Figure 1).

Our analysis concluded that out of 16 datapoints, the 1st, 2nd, and 15th datapoints from the batch dataset were outliers. For the fed-batch dataset, we concluded that out of 7 datapoints, the 3rd datapoint was an outlier for the PCC values, while the 2nd datapoint was an outlier for the PCN values.

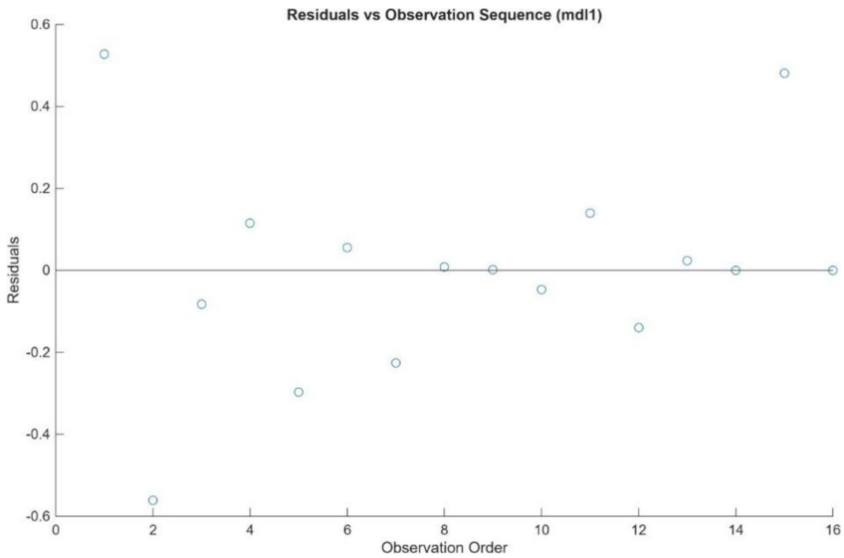

Figure 1. A scatter plot showing the residuals from the regression model based on the batch data PCC values.

B. Custom Annealing

While MATLAB offers a default simulated annealing (MSA) function ("simulannealbnd"), this function does not allow us complete control over the process

---

[2] Peng, W., Zhong, J., Yang, J. et al. (2014) The artificial neural network approach based on uniform design to optimize the fed-batch fermentation condition: application to the production of iturin A. *Microbial Cell Factories, 13* (54). https://doi.org/10.1186/1475-2859-13-54

[3] Elmeligy, A., Mehrani, P., & Thibault, J. (2018). Artificial Neural Networks as Metamodels for the Multiobjective Optimization of Biobutanol Production. *Applied Sciences*, *8*(6). https://doi.org/10.3390/app8060961

and can be quite difficult to edit. Thus, many of our annealing involved a custom annealing method (CSA). The basic functionality is similar: it generates a default point, changes the point by small amounts (perturbation), compares the resulting dependent values (based on some objective function we give), and graphs the results. The perturbation strategy we used mimics that used by "simulannealbnd." However, by using a custom function, we can edit the independent variable values in the objective function. This allows us to apply constraints and bounds to our variables easier. We can also define a custom metropolis function, which alters what values are accepted and allows us to occasionally accept "worse" objective values and potentially escape local optima.

C. Bounding and Constraints

Table 1. The annealing results from an early program that had constraint issues.

Algorithm	Process Mode (Dataset)	PCC Annealing	PCN Annealing	Mix Annealing
MATLAB Functions	Fed-Batch	PCC: 446.2375 g/L	PCN: 71.2901%	PCC: 446.2375 g/L
				PCN: -190470%

One problem that arose from our annealing results were values that fall outside of feasible ranges. This can be observed in the data shown in Table 1. As we can see, the Mix Annealing PCN value is -190470, when it should be between 0 and 100. Thus, this annealing data is not a feasible result.

When using "simulannealbnd," we can prevent this by using penalties. If the found PCC or PCN values are problematic, we have our objective function return some arbitrarily poor objective value. Thus, we ensure that this annealing run is not selected as the best version. When using our custom annealing, we can check that our independent values fall within lower and upper bounds that we define based on our dataset. We also calculate the roots of our independent variables from the regression function and use these to bound our independent variables. By doing this, we ensure our dependent variables will not be unfeasibly large or small.

D. Artificial Neural Network for Regression Model

Artificial Neural Networks (ANNs) are complex machine learning tools inspired by the human brain. They consist of a variety of layers containing neurons (an input layer, output layer, and some number of hidden layers). Each neuron receives input, processes it, and passes it to the next layer. Overall, the system can be a powerful tool, and we use it here to generate our regression model coefficients, rather than rely on

MATLAB's "fitlm" function. We use a neural network with two hidden layers that have one neuron each. We train this network and then use a Levenberg–Marquardt function to express its data in a useful regression model.

III. Results

A. Model Fit Results

We show our complete R-squared value results in Table 2. R-squared is a measure of model fit, and a R-squared value closer to one is more favorable. "fitlm" is the MATLAB function used to generate the regression model in the "MATLAB functions (MSA)" and "Custom SA (CSA)" annealing methods, while "ANN-Based" uses the artificial neural network method described above. As we can see, removing outliers from our datasets greatly improves the model fit. In addition, our ANN-based method for generating the regression model seems to perform similarly to MATLAB's "fitlm" with both the batch and fed-batch processes, although it is slightly worse with the batch dataset. Overall, all the models that use the "outliers removed" datasets seem to show good fit.

Table 2. The model fit (R-Squared) results from multiple regression model approaches and processes.

Algorithm	**Process Mode (Dataset)**	**R-Squared (PCC Model)**	**R-Squared (PCN Model)**
fitlm	Batch	0.777	0.777
fitlm	Batch (Outliers Removed)	0.971	0.974
fitlm	Fed-Batch	0.876	0.879
fitlm	Fed-Batch (Outliers Removed)	0.972	0.987
ANN-Based	Batch (Outliers Removed)	0.9693	0.9711
ANN-Based	Fed-Batch (Outliers Removed)	0.9721	0.9871

Notes: The "fitlm" algorithm is used by both MSA and CSA, and it is deterministic, meaning it yields the same results each time. The ANN-Based method varies each time, so we present our highest found R-Squared values in this table.

B. Dependent Variable Results

We show our annealing results in Table 3. We ran three different algorithms that each mitigate the issues described above, such as bounding problems. "MATLAB Functions" uses MATLAB's "simulannealbnd" and "fitlm" for the annealing and regression model. "Custom SA" uses our custom annealing function and "fitlm" for the regression model, and "ANN-Based" uses our custom annealing function and the ANN based approach for the regression model.

We can see that all three methods seem to perform similarly on the batch data, with the ANN-based method being slightly better. With our fed-batch data, the ANN-based method seems to not have caused any improvement. This seems to indicate that our ANN-based method is a viable method for generating our regression model, yielding results at least as good as the other methods.

Table 3. The optimization results from multiple algorithms and datasets (outliers removed).

Algorithm	**Process Mode (Dataset)**	**PCC Objective Function**	**PCN Objective Function**	**Mix Objective Function**
MATLAB SA	Batch	PCC: 1.0436	PCC: 1.0436	PCC: 1.0437
		PCN: 99.9073	PCN: 99.9065	PCN: 99.9101
Custom SA	Batch	PCC: 1.0437	PCC: 1.0437	PCC: 1.0437
		PCN: 99.9101	PCN: 99.9102	PCN: 99.9091
ANN-Based	Batch	PCC: 1.0883	PCC: 0.9690	PCC: 1.0883
		PCN: 100.0000	PCN: 100.0000	PCN: 100.0000
Custom SA	Fed-Batch	PCC: 2.4916	PCC: 1.7824	PCC: 2.4916
		PCN: 55.0251	PCN: 71.1321	PCN: 55.0251
ANN-Based	Fed-Batch	PCC: 2.4857	PCC: 1.7049	PCC: 2.4920
		PCN: 55.2948	PCN: 71.2906	PCN: 55.0466

Note: Results are shown from annealing based on PCC values, PCN values, and a mix of both. PCC units are g/L. PCN is a percentage (%).

C. Concluding Thoughts

Future work should involve refining our annealing methods and ANN program to produce better results. The experimental data sets used could also be expanded, which would like improve the accuracy of our simulations. Overall, despite struggles with our ANN, our work shows good progress towards optimizing this chemical process. In our experimental batch dataset, the best results were a PCN of 27.929% and a PCC of 0.282 g/L. As we can see, our annealing predicts that we could yield much better results. It also supports that the processes described above are useful methods of optimization and could be applied to more chemical processes.