Machine Learning to Discover Simple Terminal Velocity Equations

Trevor Semeraro

June 2025

1 Introduction

The goal of this project is to fit an equation using the diameter, pressure, temperature alongside additional calculated parameters such as density and dynamic viscosity to estimate the terminal velocity of cloud droplets. Using the outputs of the methodology described in Beard et al.[2] as the ground truth, we aim to derive a model that is computationally faster than Beard as well as simpler models such as Simmel et al. [3], whilst minimizing our error.

2 Methodology

2.1 Symbolic-Regression Framework

We frame terminal-velocity estimation as a symbolic-regression problem. Equations are discovered with `SymbolicRegression.jl` [5]. The search is restricted to the binary operators

$$\{+,\ -,\ *,\ /,\ x^y, \min^*,\ \max^*\},$$

operators with $(*)$ are only active for the correction model, and unary operators

$$\{\sqrt{x},\ x^2,\ x^3,\ x^{-1},\ |x|\}.$$

We chose to exclude $\sin, \cos, \log$ because of their infrequency in cloud-microphysics parameterizations.

Model complexity. `PySR` defines complexity as the sum of node costs in the expression tree. All operators default to cost 1, but we reweigh computationally expensive operations to ensure higher complexities are representative of slower models (Table 1). We assign weights in accordance to operators reciprocal throughput. Throughput is the maximum number of instructions of the same kind that can be executed per clock cycle. The reciprocal throughput is therefore the average number of clock cycles per instruction.[4] As these values vary from CPU to CPU, we selected the AMD K7 as our reference, if you are building a model for a specific machine, these values would be worth adjusting. Candidate expressions were capped at total complexitys of 100 and 200 respectively.

Operator	$\lvert x\rvert$	min	max	x^3	x^4	x^8	x^{12}	x^{16}	x^{-1}	$/$	$\sqrt{x}$	x^y
Cost	0.5	0.5	0.5	2	2	3	4	4	4	4	11	20

Table 1: Adjusted node costs reflecting reciprocal throughput's on AMD K7 processor [4]. Unlisted operators keep the default cost 1.

3 Training

3.1 Model

We split the equation discovery task into two segments. First, we aim to learn the relationship between the drop diameter and the terminal velocity at a reference temperature and pressure, T_{ref} and p_{ref}, respectively:

$$v_{\text{ref}}(d) = v(d, T_{\text{ref}}, p_{\text{ref}}) \tag{1}$$

where v_{ref} is the learned function dependent only on d. Second, we aim to learn a thermodynamical correction term f_{thermo} (unitless) relative to the reference velocity such that:

$$f_{\text{thermo}}(d, T, \rho, \eta) \equiv v(d, T, \rho, \eta)/v_{\text{ref}}(d). \tag{2}$$

where f_{thermo} takes inputs diameter, temperature, density, and dynamic viscosity respectively. The product of these two functions' outputs yields a drop's terminal velocity v (m s^{-1}):

$$v(d, T, p) = f_{\text{thermo}}(d, T, p)v_{\text{ref}}(d). \tag{3}$$

There was a wide range of terminal velocity output values for the reference equation, $(2.85\times10^{-6}, 12)$m/s. For the thermodynamic correction term we did not want to penalize larger errors on smaller correction values compared to larger correction values. Therefore, we trained both steps with respect to mean squared relative loss.

3.2 Data Collection

	Diameter D [μm]	Pressure p [Pa]	Temperature T [K]
Lower bound	1	6×10^4	230
Upper bound	7×10^3	1.02×10^5	310

Table 2: Sampling ranges. Diameter and pressure are drawn in log space; temperature in linear space.

Reference Training Data Diameter was taken in log space over the diameter range illustrated in (Table 2). Temperature was kept constant at 285K and pressure was kept constant at 90,000Pa. We sampled this space over 100,000 points.

Correction Training Data A Latin Hypercube Sampling (LHS) design of N = 10,000 points ensures stratified coverage of the input space spanned by drop diameter D, air temperature T, and air pressure p (Table 2). The D and p axes are sampled logarithmically to prevent oversampling large raindrops and under sampling small raindrops. The T–p range corresponds to conditions where liquid water is thermodynamically stable in the Earth's troposphere (SOURCE).

Additionally, to capture small-scale discrepancies in the final model, we generate a 50^3 Cartesian mesh on $\log D$, $\log p$, and T, yielding 125 000 additional points for fine-tuning.

3.3 Ground-Truth Computation

For every sampled triplet (D, T, p) we compute the terminal velocity V_{term} using the formulation of Beard et al. [2]. Air density ρ_{air} follows the ideal-gas law,

$$\rho_{\text{air}} = \frac{p}{R_{\text{air}}T},$$

and dynamic viscosity $\mu(T)$ is obtained from Sutherland's law with constants $C_1 = 1.72 \times 10^{-5}$ Pa s, $S = 120$ K [1]. Water density is fixed at 998 kg m^{-3}.

3.4 Training Procedure

We trained both the reference and correction models with respect to minimizing the mean squared relative error. We choose this error metric because the range of possible terminal velocity values $(2\times10^{-6}\text{m/s}, 12\text{m/s})$ spans several orders of magnitude.

3.5 Benchmarking

To quantify computation speed, we re-implement each candidate terminal velocity formula in Fortran. Every function is evaluated on the same test set of $N = 10^6$ (D, p, T) tuples. We repeat this full evaluation 101 times within a single process to minimize startup overhead and to smooth out timing noise.

We report throughput as "millions of equation evaluations per second", and calculate it as the mean wall-clock time of the last 100 runs (the first run is discarded to eliminate cache-warm-up effects). All benchmarks are compiled with `gfortran -Ofast -march=native`.

4 Results

$$
\begin{aligned}
v_{\text{ref}}(d) = \Big(c_0\Big(c_1 + c_{19}x_1^{c_3} + c_2\Big(c_{14}\big(c_{15} + c_{16}x_1\big(c_{17} + c_{18}x_1\big)^{c_3}\big)^{c_3} \\
+ c_3\Big|c_4 + c_5\big(c_6 + c_7x_1^{c_3}\big)^{c_3} + c_8x_1^{c_9}\big(c_{10} + c_{11}x_1\big)^{c_3}\big(c_{12} + c_{13}x_1\big)^{c_3}\Big|\Big)^{c_3}\Big)^{c_3}\Big)^{c_9}
\end{aligned} \tag{4}
$$

$$
\begin{aligned}
f_{corr}(d,T,p) = \Big|\big(x_2\Big(c_0 + c_1x_2 + c_2^{c_3}x_2 + c_4\Big(c_{15} + c_{16}x_3^{c_9}\big(c_{17} + x_2\big)\Big)\big(c_{18}^{c_{19}} + x_3\big) \\
\cdot\Big(x_2 + \big(c_{10}c_{12}^{c_{13}}x_1^{c_{11}} + c_{14}x_1x_2\big)|c_5 + c_6x_1^{c_7} + c_8x_2|^{c_9}\Big)^{c_9}\big)^{c_{20}}\Big|^{c_{19}}
\end{aligned} \tag{5}
$$

These equations are written in python and Fortran in the github. They are then combined as described in (3).

4.1 Pareto Front Analysis

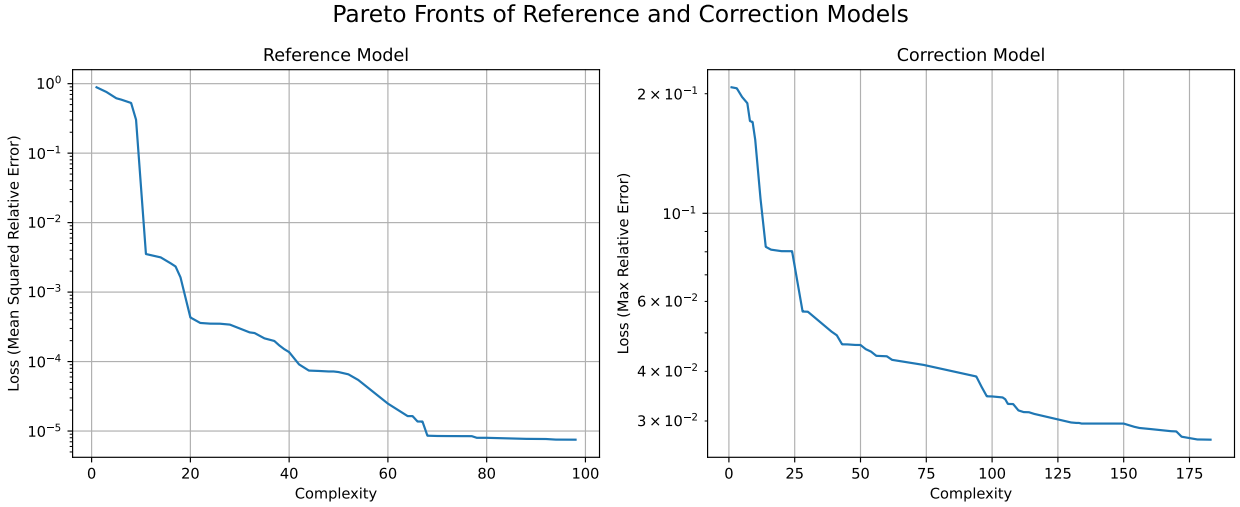

Figure 1: Enter Caption

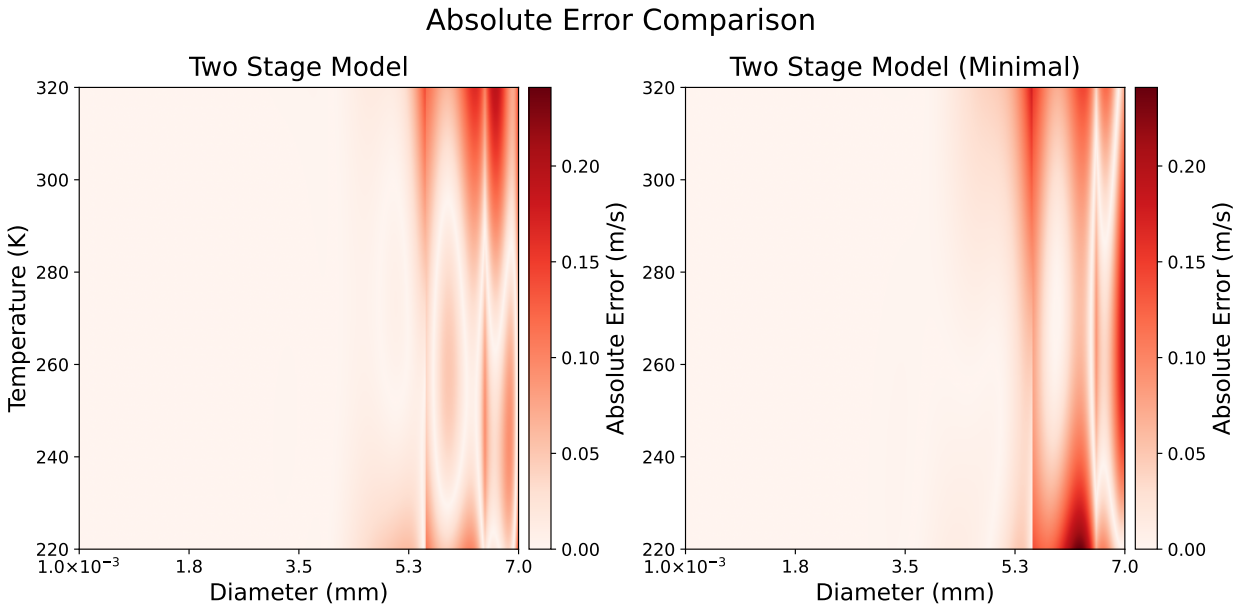

Figure 2: Enter Caption

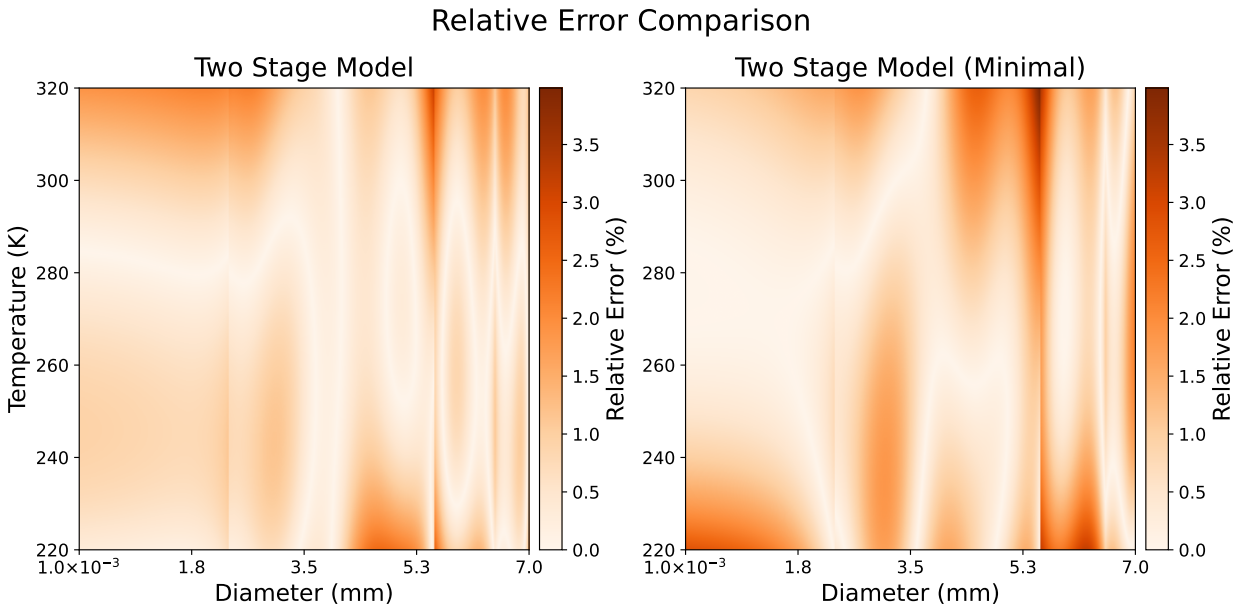

Figure 3: Enter Caption

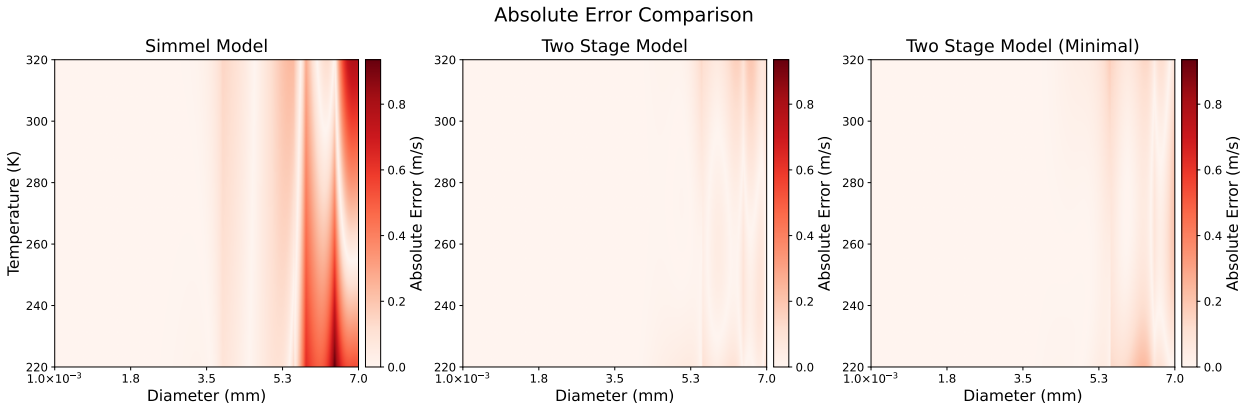

Figure 4: Enter Caption

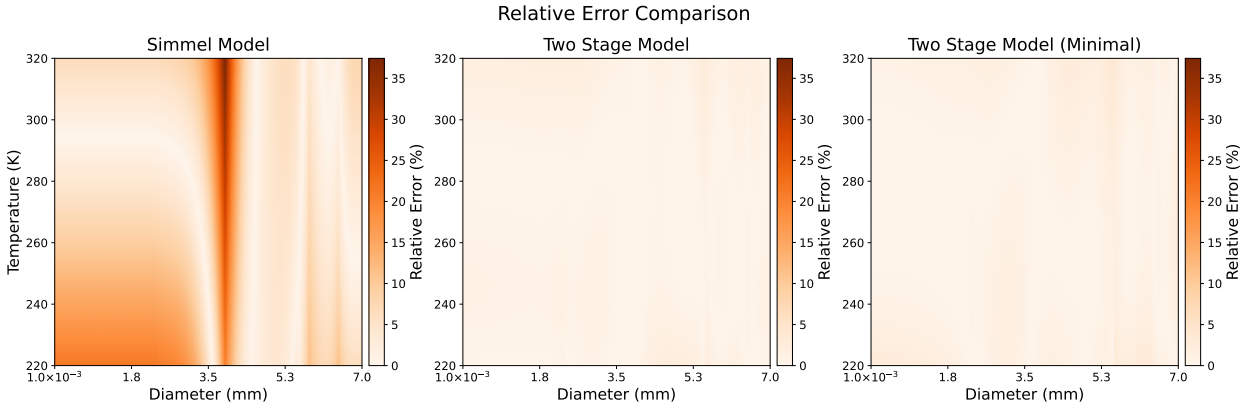

Figure 5: Enter Caption

Metric	BEARD	SIMMEL	TWO STAGE	TWO STAGE MINI
Time	1.35×10^{-1}	5.48×10^{-2}	8.02×10^{-2}	1.08×10^{-2}
Max Relative Error	2.09×10^{-4}	3.84×10^{1}	2.86	3.73
Mean Relative Error	1.62×10^{-5}	7.83	5.77×10^{-1}	6.91×10^{-1}
Max Absolute Error	2.10×10^{-5}	2.88	2.32×10^{-1}	3.37×10^{-1}
Mean Absolute Error	1.11×10^{-6}	1.65×10^{-1}	1.33×10^{-2}	1.74×10^{-2}

Table 3: Performance results on a validation dataset of 1,000,000 samples.

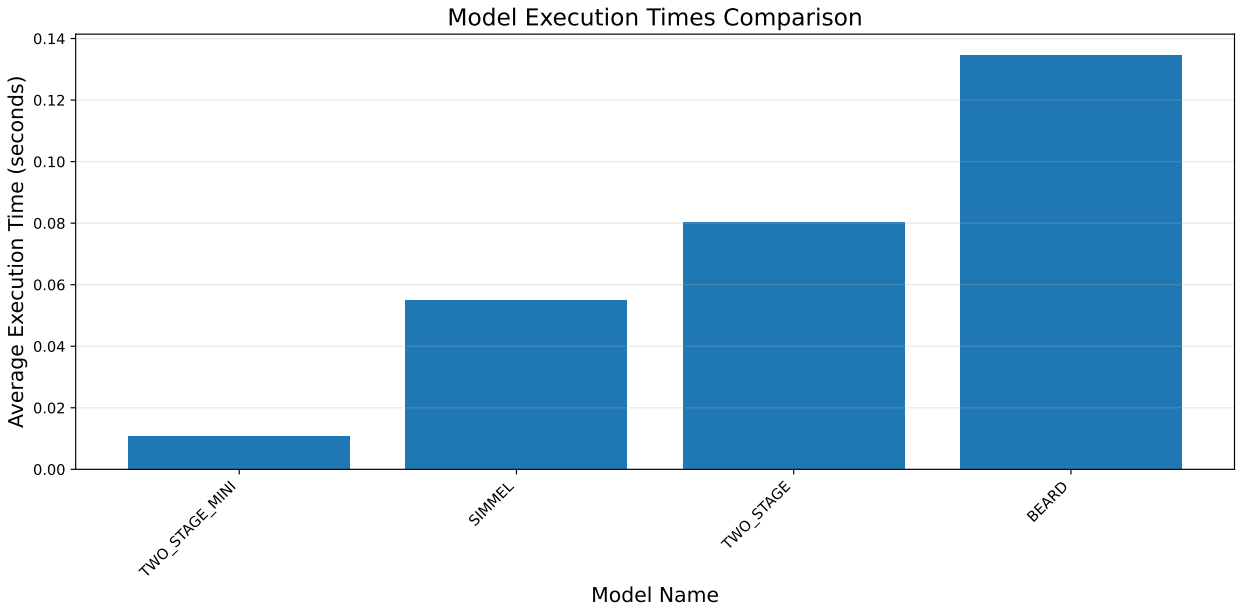

Figure 6: Enter Caption

4.2 Performance Metrics

Table 3 shows the max relative error, mean relative error, max absolute error, and mean absolute error for the simmel funciton as well as the 3 models in this paper. The results were calculating on the validation dataset, running each respective function and calculating metrics in relation to the ground truth. The metrics illustrate that the One Shot (30) model has a similar runtime to the Simmel model, whereas the One Shot (20) and Two Stage Models have roughly half the runtime. All models perform better then the Simmel model on all error metrics, besides for the Two Stage Model with respect to Max Absolute Error.

Below are figures illustrating the error rates of the different models for ease of comparison. Additionally, they indicate the regimes where the models tend to misrepresent the true terminal velocities. All absolute errors use the same color mapping, and all relative errors use the same color mapping for ease of comparison across charts.

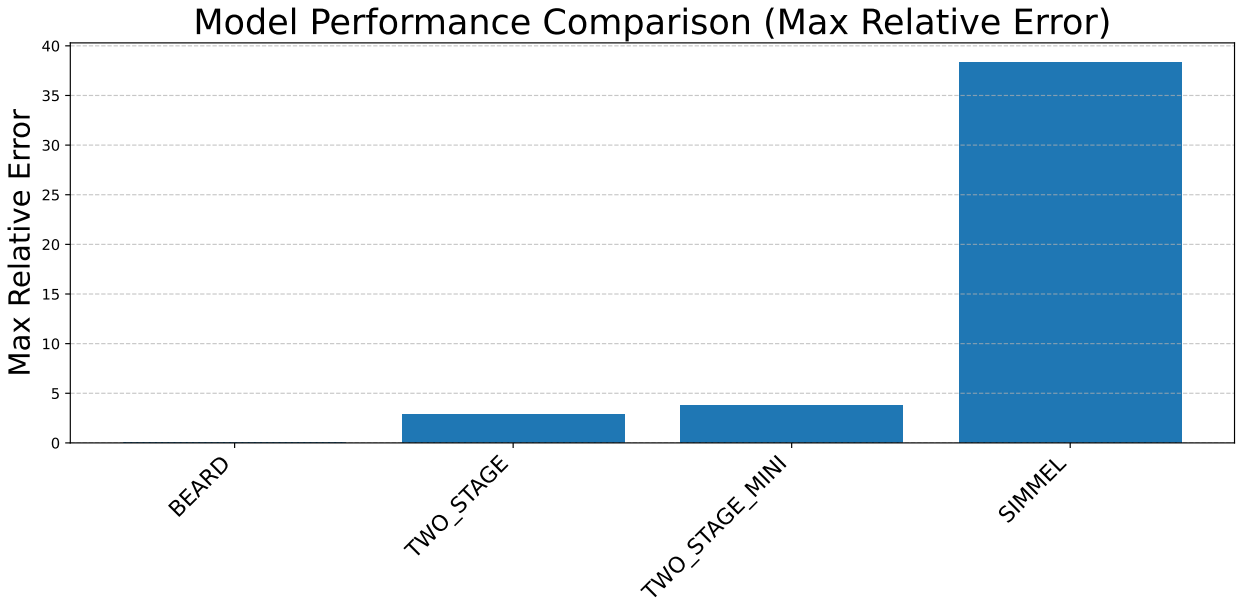

Figure 7: Enter Caption

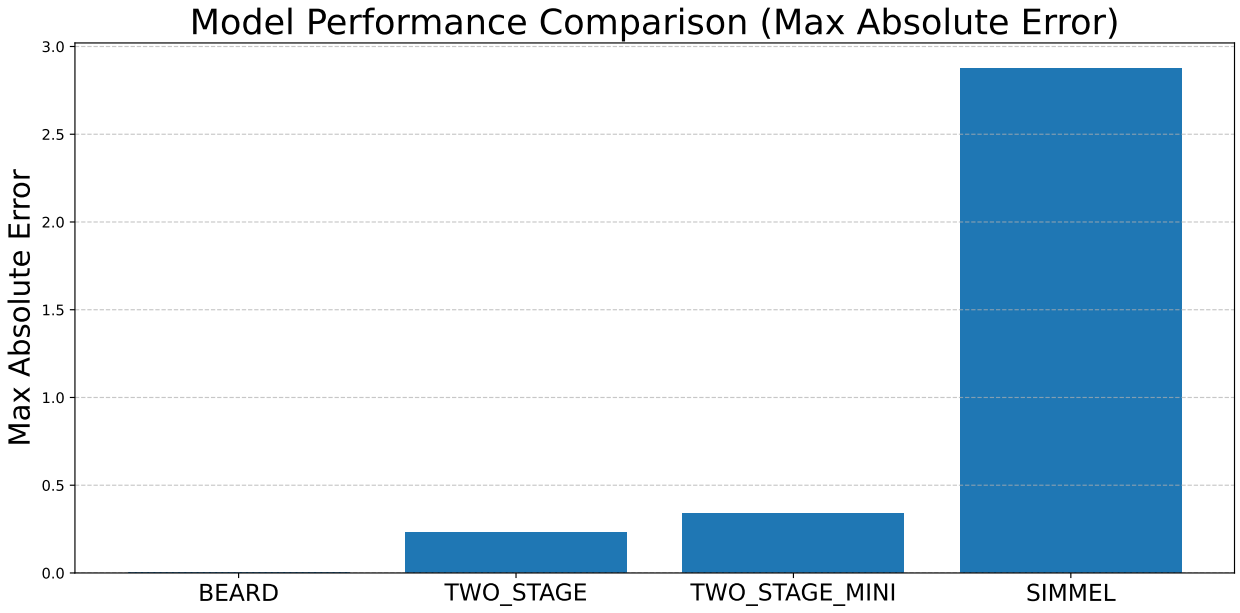

Figure 8: Enter Caption

References

[1] William Sutherland. "The viscosity of gases and molecular force". In: *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*. Series 5 36.223 (1893), pp. 507–531. DOI: 10.1080/14786449308620508. URL: https://doi.org/10.1080/14786449308620508.

[2] K. V. Beard. "Terminal Velocity and Shape of Cloud and Precipitation Drops Aloft". In: *Journal of Atmospheric Sciences* 33.5 (1976), pp. 851–864. DOI: 10.1175/1520-0469(1976)033<0851:TVASOC>2.0.CO;2. URL: https://journals.ametsoc.org/view/journals/atsc/33/5/1520-0469_1976_033_0851_tvasoc_2_0_co_2.xml.

[3] Martin Simmel, Thomas Trautmann, and Gerd Tetzlaff. "Numerical solution of the stochastic collection equation—comparison of the Linear Discrete Method with other methods". In: *Atmospheric Research* 61.2 (2002), pp. 135–148. ISSN: 0169-8095. DOI: https://doi.org/10.1016/S0169-8095(01)00131-4. URL: https://www.sciencedirect.com/science/article/pii/S0169809501001314.

[4] Agner Fog. "Instruction tables: Lists of instruction latencies, throughputs and micro-operation breakdowns for Intel, AMD and VIA CPUs". In: *Agner.org manuals* (2022). Version 4, last updated 4 Nov 2022. URL: https://www.agner.org/optimize/instruction_tables.pdf (visited on 07/14/2025).

[5] Miles Cranmer. *Interpretable Machine Learning for Science with PySR and SymbolicRegression.jl*. 2023. arXiv: 2305.01582 [astro-ph.IM]. URL: https://arxiv.org/abs/2305.01582.