CruiseAuto Project – Milestone 4

ANSWER SHEET: Algorithm Refinement and Final Deliverables

Table of Contents

Part 1. Assignment Header

Section and Team ID: 222_30

Team Member Name	Purdue Career Account Login	Programmer Number	Detailed Description of the Work	Percent of Work
TJ Bielefeld	tbielefe	1	Made improvements to subfunction 2,3 & performance. Filled in M4 values for parameters of experimental and benchmark data.	25
Bridget O’Donnell	odonne62	4	Filled out 4a tables with benchmark values and M3 values. Entered values for 4b M3 algorithm parameters. Completed The second half of the technical brief.	25
Arvin Bhasin	bhasin15	3	Wrote out our improvement plan, as well as resume block, for this project. I also worked on our technical brief.	25
Olivia Stauffer	stauffeo	2	Filled out the introduction and parameters section of the technical brief part 6. Filled out the references in part 8 and the MATLAB functions we used in part 9.	25

Part 2. Milestone 3 Feedback and Reflection

Strength: The strengths found in our M3 is the deliverables we were able to properly execute, as we gave parameter output comparison and ACC performance boundaries correctly. This showcases our group’s strength in using SSE and also comparing our work with the benchmark data in a productive and correct manner.

Limitation: Our only limitation was our usage of in-text citations, as we failed to reference our citation in the text we used it in.

How could the feedback from M3 lead to improvements? It can lead us to make sure to reference our sources in our text rather than just the references.

What concrete steps will you take to incorporate the M3 feedback to improve your algorithm?

The concrete step we would take is to make sure to look at our references, and make sure each source cited can be found in our text and is properly cited.

Part 3. Algorithm Improvement Plan

Improvement #1

Parameter(s) Targeted: Time constant

Describe the improvement that your team is introducing to the M3 algorithm: We'll be taking our raw data, cleaning each data set, by taking the average around a NAN value, to complete any holes, and run each data set through our time constant function, and take the average of the time constants we find per vehicle's tire, giving us a more accurate reading.

Why are these refinements necessary and how will they improve your algorithm?: Due to the way our data is cleaned, with averaging the data in our method, we see this led to an inaccurate slope. So, fixing the way we apply the way we clean our data, will improve the accuracy of our time constant, because the time constant is just reliant on when the data increased by 63.7%, and if we cleaned the data by averaging it, shifting some data, then the time constant would be directly affected. So the improvement of cleaning the data set without averaging, we will get a better time constant, with us not shifting the points of our data.

Improvement #2

Parameter(s) Targeted: Initial Speed

Describe the improvement that your team is introducing to the M3 algorithm: We're changing the use of the mean to median in the use of a 4th sub-function, which finds the initial speed.

Why are these refinements necessary and how will they improve your algorithm?: This refinement makes the initial speed more accurate due to mean being more sensitive to outliers, while median is less affected by outliers, making the algorithm improved. This is refinement is needed to make our initial speed more accurate.

Part 4. Algorithm Refinements Implementation and Results

4a. Refinement Results with Benchmark Data

Table 4a.1 - Results for Compact Hatchback Benchmark Data

Parameter	Benchmark Values	M3 Algorithm Values	M3 Percent Error	M4 Algorithm Values	M4 Percent Error
Acceleration start time [s]	6.21	6.30	1.45%	6.11	1.61%
Time constant [s]	1.51	1.44	4.64%	1.47	2.65%
Initial speed [m/s]	-0.09	-0.20	122.02%	-0.17	94.44%
Final speed [m/s]	25.08	24.96	0.46%	25.04	0.16%

Table 4a.2 - Results for Midsize Sedan Benchmark Data

Parameter	Benchmark Values	M3 Algorithm Values	M3 Percent Error	M4 Algorithm Values	M4 Percent Error
Acceleration start time [s]	9.39	9.52	1.38%	9.30	0.96%
Time constant [s]	1.96	1.90	3.06%	1.88	4.08%
Initial speed [m/s]	-0.22	-0.31	41.83%	-0.26	20.00%
Final speed [m/s]	24.72	24.60	0.48%	24.64	0.31%

Table 4a.3 - Results for SUV Benchmark Data

Parameter	Benchmark Values	M3 Algorithm Values	M3 Percent Error	M4 Algorithm Values	M4 Percent Error
Acceleration start time [s]	6.85	7.17	4.67%	6.79	0.88%
Time constant [s]	2.80	2.45	12.50%	2.63	6.07%

Initial speed [m/s]	0.19	0.16	17.57%	0.16	16.32%
Final speed [m/s]	24.18	24.11	0.28%	24.16	0.08%

Table 4a.4 – Results for SSE_{mod}

Vehicle	SSE_{mod} from Benchmark Parameters (Part 4a of M3)	SSE_{mod} from M3 Algorithm Parameters (Part 4b of M3)	SSE_{mod} from M4 Algorithm Parameters
Compact Hatchback	.5828	0.5950	0.6350
Midsize Sedan	.5080	0.5392	0.5553
Large SUV	.5088	0.5884	0.5945

4b. Refinement Results with Experimental Data

Table 4b.1 – M3 and M4 Algorithm Comparison of Experimental Data Parameters

Vehicle	Tire Type	M3 Algorithm Parameters				M4 Algorithm Parameters			
		Start time [s]	Time constant [s]	Initial speed [m/s]	Final speed [m/s]	Start time [s]	Time constant [s]	Initial speed [m/s]	Final speed [m/s]
Compact Hatchback	Summer	6.88	2.16	-0.0767	24.6684	7.73	1.53	-0.0741	24.7084
Compact Hatchback	All-Season	5.93	3.92	-0.0808	24.9143	7.75	1.76	-0.0543	24.9618
Compact Hatchback	Winter	6.76	4.24	-0.0670	25.0583	8.09	2.62	-0.0791	25.0864
Midsize Sedan	Summer	5.97	4.01	-0.0712	24.9581	7.80	1.39	-0.0676	24.9741
Midsize Sedan	All-Season	5.44	2.32	-0.0875	24.8941	6.17	1.84	-0.0747	24.9316
Midsize Sedan	Winter	7.33	3.86	-0.0725	24.9415	8.27	2.57	-0.0777	24.9844
Large SUV	Summer	5.96	3.82	-0.0575	24.0656	7.45	3.06	-0.0650	24.0992

Large SUV	All-Season	5.54	4.72	-0.0746	24.0346	7.26	2.27	-0.0789	24.0790
Large SUV	Winter	6.81	3.99	-0.0685	23.6413	7.66	3.27	-0.0585	23.6758

4c. Performance Check with Experimental Parameters

Table 4c.1 – Performance Boundary Results

Vehicle	Tire Type	Within Bounds or Outside Bounds?
Compact Hatchback	Summer	The compact hatchback with summer tires is within the bounds for both final velocity and the time constant.
Compact Hatchback	All-Season	The compact hatchback with all- season tires is within the bounds for both final velocity and the time constant.
Compact Hatchback	Winter	The compact hatchback with winter tires is within the bounds for both final velocity and the time constant.
Midsize Sedan	Summer	The midsize sedan with summer tires is within the bounds for both final velocity and the time constant.
Midsize Sedan	All-Season	The midsize sedan with all-season tires is within the bounds for both final velocity and the time constant.
Midsize Sedan	Winter	The midsize sedan with winter tires is within the bounds for both final velocity and the time constant.
Large SUV	Summer	The large SUV with summer tires is within the bounds for both final velocity and the time constant.
Large SUV	All-Season	The large SUV with all-season tires is within the bounds for both final velocity and the time constant.
Large SUV	Winter	The large SUV with winter tires is within the bounds for the final velocity but NOT for the time constant.

Part 5. Algorithm Performance Discussion

Do you believe your algorithm accurately reflects the true performance of the system? Why or why not?

I believe our algorithm accurately reflects the true performance of the system, as our time constants and final velocities matches the range that was given to compare, as well as our model graphs being very identical to the raw graphs given to us as well. As well as the benchmark data given in M3, being very similar and comparable to our data as we saw in M3, and above with all the algorithms and benchmark data being very similar. Also, we can see that we're within bounds for both time constant and final velocity for all tires for Sedan and Hatchback, with the only exception of the Large SUV, where we have everything in bound except for summer tires, with the time constant being off. But with everything else being in, except for the time constant, I think it can be said our algorithm is accurate.

Does your algorithm need more work that you are unable to complete because of the due date? If yes, describe what you would do. If no, justify why your analysis is complete as-is.

Our team worked vigorously, and stayed on top of all due dates, so time is not an issue to us, and the due date had no effect on our completion of our algorithm. We can see that we're within bounds for both time constant and final velocity for all tires for Sedan and Hatchback, with the only exception of the Large SUV, where we have everything in bound except for winter tires, with the time constant being off. But with everything else being in, except for the time constant, I think it can be said our algorithm is complete as-is.

Does your technical brief reflect your critique of your algorithm's performance? Remember, it is vital in engineering to accurately represent your work.

Yes, in our technical brief we critique our acceleration system and the way it's coded, and the results not being 100% accurate for all vehicles. Critiquing our work is vital for engineering, as we want to make sure that the CruiseAuto is aware of the limitations for our system, so they know how and where the limits of our system stretch, as well as are able to implement our system in different softwares or languages as they know the details of not only how it works, but how it doesn't, which increases the implementation for someone looking to interpret/use this system for their desired use.

Part 6. Technical Brief

Use the template document provided. Upload all deliverables to Gradescope.

Part 7. Resumé Insert

Design Projects

Cruise Control System, *Purdue University* Spring 2025

- → **Designed and programmed a multi-function MATLAB algorithm** to identify vehicle acceleration parameters (start time, time constant, initial/final speeds) from noisy sensor data by creating user-defined functions and structured pseudocode planning.
- → **Iteratively improved algorithm performance** by evaluating noise error, benchmark comparisons, and performance boundary visualizations, leading to a refined model with reduced parameter identification error rates.
- → **Delivered a fully-tested, client-ready technical package** including updated scripts, data analysis plots, and a technical brief summarizing accuracy and improvement justifications for our client's adaptive cruise control system assessment.

Part 8. References

Floating-Point Formats. (2024). Mathworks.com. https://www.mathworks.com/help/matlab/ref/string.sprintf.html

Determine NaN Elements in Real Numbers. (2024). Mathworks.com. https://www.mathworks.com/help/matlab/ref/double.isnan.html

Part 9. MATLAB Built-in Functions

Fill out the following information **for each** MATLAB built-in function that your algorithm uses that was not explicitly taught in class. Add additional rows as needed. If you did not use any new built-in MATLAB functions, then delete the table below and write "No new function used."

Function Name & Call (include inputs/outputs)	Where in your algorithm do you use the function?	Describe the inputs needed to run the function.	Describe the outputs from the function.	Describe the theory and/or mathematics behind how the function operates.
sprintf() input: a string output: array of strings	We used this function in the main inside a loop that's job was to print different figures. We used it to help us print the different titles for each graph.	A string is a needed input to run the function as we want a string to be output (Mathworks 2024). The output could be something that is not a string but that would not produce the string output we want. Thus, for this program our input must be a string.	If the input is a string, then the output is also a string. If the input is not a string, then the output is a character (Mathworks 2024). Since our input was a string, our output was a string	When writing this code this function looks similar to the fprintf function. However, in this function we can assign it to a variable that can then be called later. Thus, a whole print statement can be assigned to a variable. This is used when the programmer wants a formatted output. It is common to use this for a title for a graph, labeling axes, or a text object.
Isnan() input: an array Output: a logical array	We use this function to clean our data in subfunction 2. This is used as a condition in an if else statement. We use this to	The input is an array of characters that can include numbers or Nan values (Mathworks 2024).	The output is a logical array the same size as the input array. Where there are NaN values in the input array there are zeros	This function evaluates an array. First it creates an array of zeros the same size as the input function. It goes through each index

	test if a value in the velocity matrix is a number or not.		in the output array. Where there are numbers in the input array there are ones in the output array (Mathworks 2024).	and evaluates if a given index is a number or not. If the value is NaN it does not change the logical array however if the value is a number, it changes that index value to a one. It then prints out the value.