ASE-6002

TA-4: Design Optimization

Team Name: Raiders of the Lost Ark

MixQuick

Team Members:

Kwasi Darkwah

Scarlett Marzo

Aaron Seemungal

David Sehrt

Ash Thomas

Date: December 13, 2021

Table of Contents

List of Figures

List of Tables

Executive Summary

In this report, we explore potential optimal values for the design variables within the MixQuick blender model while factoring in uncertainty and risk preferences. The goal of this document is to explain how our combined set of models (Technical, Cost and Demand) were used to solve a complete decision problem. It details the structured approach used to formulate a base-line solution and subsequently, solve the same problem by maximizing the expected utility of the MixQuick model.

The initial design decision was guided by design theory and systems thinking. The team defined the design variables and performance attributes, as well as considered the relationship between these elements. Design variables were identified, calculated into Performance Attributes, and an Overall Profitability was simulated and optimized. The team obtained values for the design variables through research of existing blending units on the market. The model was then made to be more robust by integrating uncertainty and risk preferences while predicting the performance of the system.

Initially, the design space was explored to find a deterministic solution. The optimal design solution and optimum utility was maximized while maintaining the risk tolerance that was previously researched. First, the deterministic approach evaluated an unconstrained design space to find the absolute optimal design. The resulting optimal design was found to be an infeasible solution. Thus, constraints were added to the model, as needed, to generate a feasible deterministic solution.

Next, the design problem was explored under uncertainty. That is, solving the same problem with uncertainties while maximizing the average expected utility. This was achieved by implementing a nested loop in our optimization analysis to allow for more comprehensive probabilistic analysis. The Latin Hypercube Sampling (LHS) method was introduced and was found to converge faster than the Monte Carlo (MC) sampling. While comparing the results of the deterministic and stochastic approaches, it was observed that, in many cases, LHS outperformed MC substantially, even with several uncertain variables incorporated.

Finding the optimal solution to the MixQuick design problem using two different approaches provided several key insights on how to conduct effective optimizations, ranging from the effective selection of algorithms to the evaluation of the results and even performance of the optimizer tool. Perhaps the most important lesson surrounds the number of iterations selected within runs. It was noted that multiple iterations are necessary for a good model. However, as the number of iterations increases, so does time to complete a simulation. As such, there is a fine line after which increasing the number of runs further does not significantly improve the simulation results compared to resources invested.

There were limitations and sources of error within the MixQuick model. As the adage goes, "garbage in, garbage out." This was a serious consideration for the design team as models are only as good as the information you put into them. In instances where formulas were created or combined for modeling purposes, the team felt that physical equations could be further developed or improved given more time and incorporating an iterative and incremental approach. Another key concern was the accuracy of the information ascertained from the demand model since it provides a rough estimation for user preferences. The design team noted that modeling design variables as exponential functions returned more realistic results.

The MixQuick design problem was focused on designing a smart blender that would be most profitable. The model started as a simple one that relied on linear equations and did not factor in uncertainties. Now, the model considers uncertainties, risks, probabilities and relies on exponential equations. Its development into a mature model along with the implementation of optimization tools allowed us to find an optimal and feasible design solution for the MixQuick blender.

1 **Deterministic Solution**

1.1 Optimization

The MixQuick model was updated to include uncertainty and optimization in this assignment, as shown in Figure 1. The initial optimization was performed in a deterministic manner with the uncertainty variables set to their most likely values. After a review of the unconstrained results, a requirement that *Jar Volume* must be greater than zero and less than 1.5 L was created for a more realistic optimization.

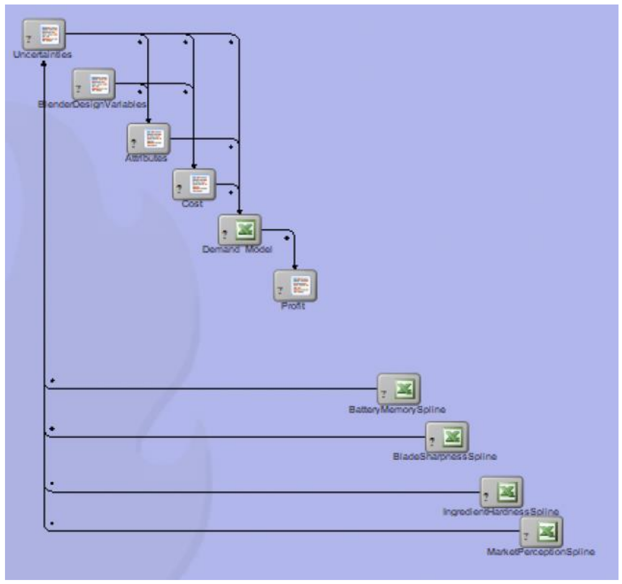

Figure 1 MixQuick Model at Constant Uncertainty

1.1.1 Unconstrained Optimization

To understand the unconstrained space, an initial Design of Experiments (DOE) was performed. The focus of this optimization was on evaluating two of three design variables at a time, compared against *Utility*. The results are shown below in Figure 2.

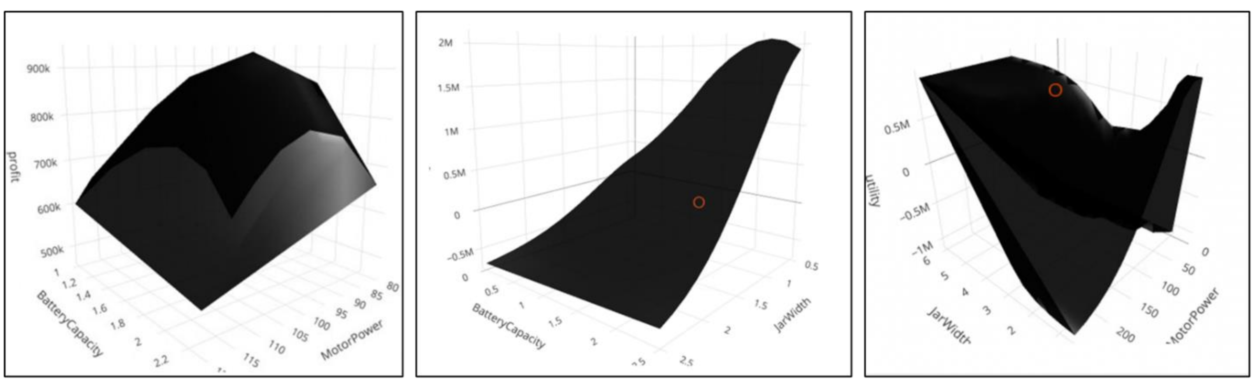

Figure 2 Unconstrained Optimization DOE Surface Plots

The DOE revealed that there was a maximum located within the identified bounds of *Battery Capacity* (approximately 1.7Amp-hrs) and *Motor Power* (approximately 110 W). Jar Width had two interesting features outside of the intended design range of 1.0 to 2.5”. *Jar Width* showed a maximum greater than approximately 6” and slowly increased as *Jar Width* increased. An excessively large *Jar Width* is not realistic. This finding was likely a result of the Demand Model. The model considers *Jar Width* as optimized at the maximum values and maintains *Blend Time* and other Performance Attributes at large sizes. Once *Jar Width* was bounded to the max of 2.5”, another maximum was identified at approximately 1.0”. Again, this is another unrealistic result below of our intended design range. The targeted Market should have a *Jar Volume* of 0.25 to 1.5L, in line with the options available on the team’s Design Survey Tool. This drove the need to apply a constraint on all Optimizations requiring *Jar Volume* to exist between 0.25 and 1.5L. Another notable feature of *Jar Width* versus *Motor Power* is the saddle feature. As *Motor Power* increases, the *Battery Life* decreases which in turn decreases demand. When *Motor Power* was too low, the *Blend Time* would increase which similarly depressed the demand.

To understand the effects of unconstrained design variable on utility, profit, demand, and price, the Optimization Tool was applied to the updated model, as shown in Figure 3.

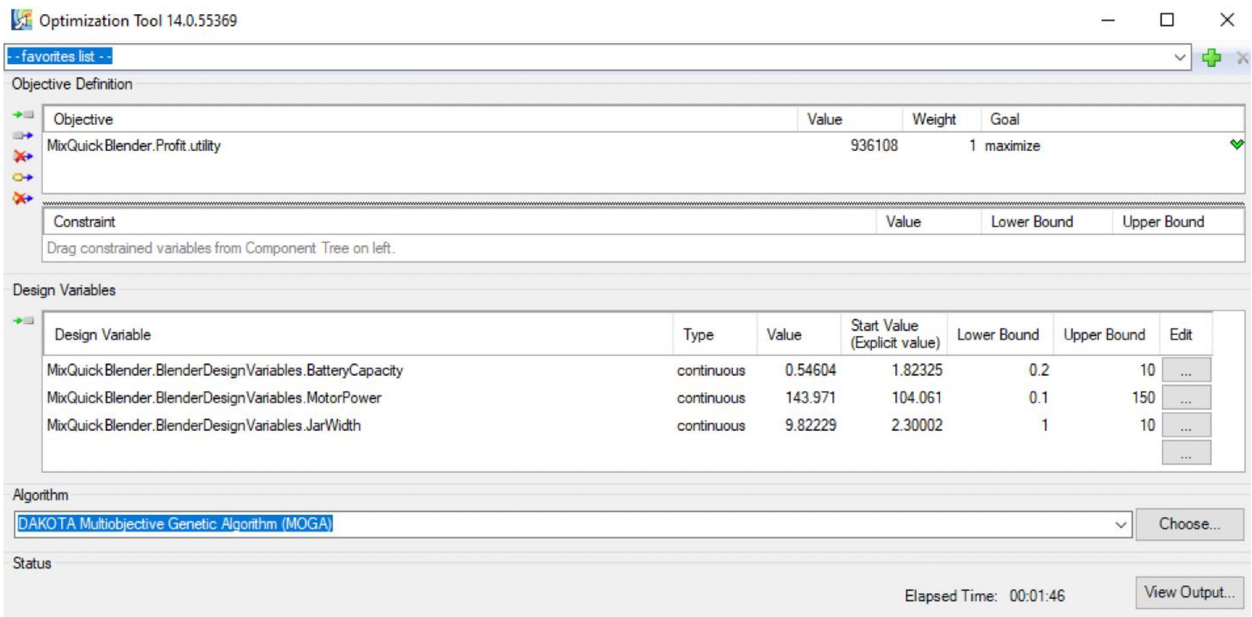

Figure 3 Set Up of the Optimization Tool for Unbounded Optimization

The outcomes of the optimization align with the results of the DOE in the unconstrained state. The unrealistically large jar volume is the optimum solution per the algorithm. On the three-dimensional plot, Figure 4, the utility is increasing as *Jar Width* increases. Additionally, this is driving the *Battery Capacity* down to zero, which is also unrealistic. The *Motor Power* is also being pushed to the maximum extent of the study, which is another odd result. The performance attributes for this unbounded optimization would result in a *Jar Volume* of 21 L (unacceptably high), a *Blend Time* of 429 s (unacceptably high), and a *Battery Life* of 1.2 min (low but could be acceptable). Through several additional optimization trials, a constraint on *Jar Width* solves the unacceptable performance attribute problems.

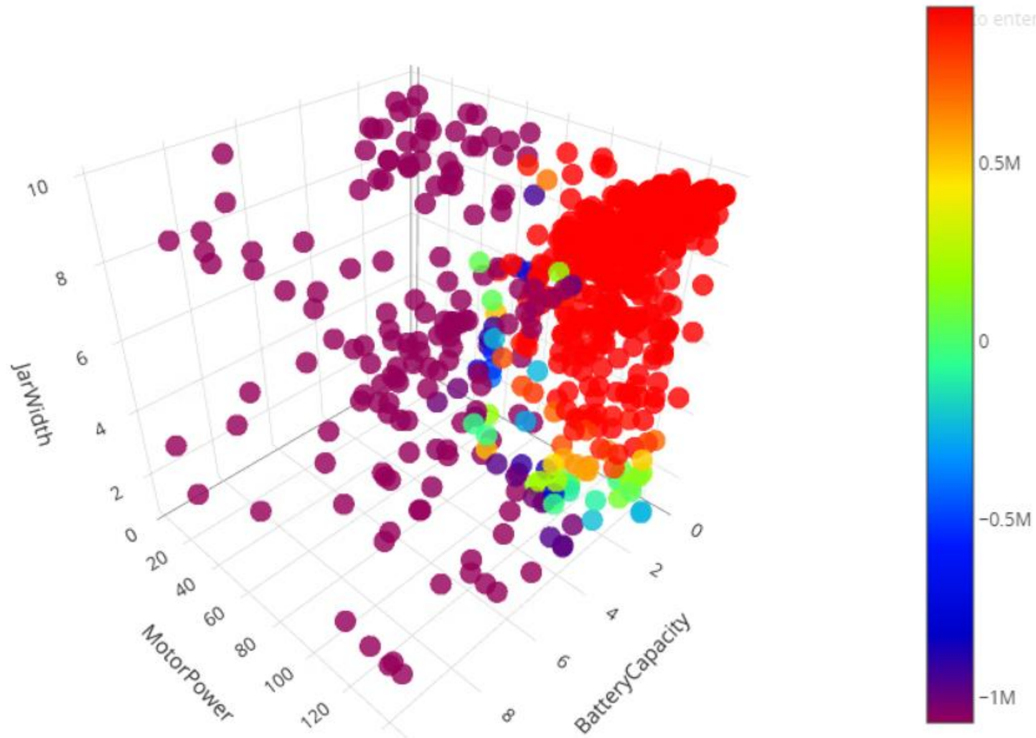

Figure 4 DAKOTA MOGA Results for Unbounded Optimization, Utility (Two Angles)

1.1.2 Constrained Optimization

Once the problems from the unbound model were evaluated, a constraint of *Jar Volume* greater than 0.25 L and less than 1.25 L was applied. (Note that this constraint can be applied in the Optimization Tool Constraints or by applying an upper bound on the *Jar Width* of approximately 1.4 in. to 2.5 in.) The optimization trade study was reperformed, per Figure 5, to ensure a realistic product.

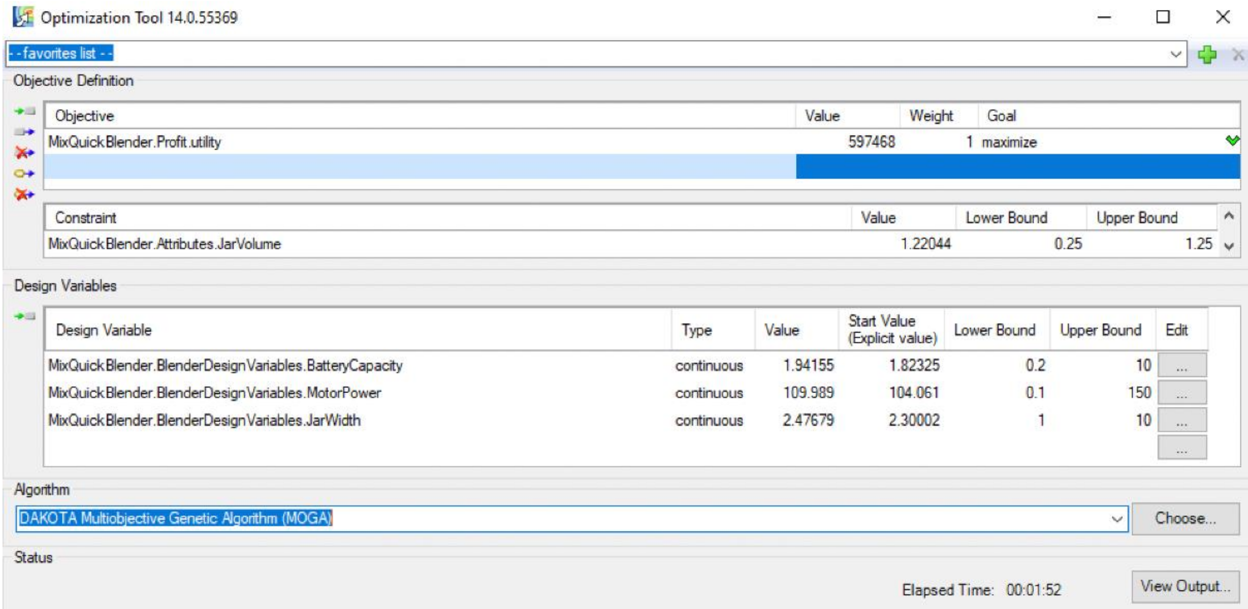

Figure 5 Set Up of the Optimization Tool for Constrained Optimization

The results of the constrained optimization were more acceptable. The model matched our DOE as the *Battery Capacity* was 1.94 Amp·Hrs, *Motor Power* was 110 W, and *Jar Width* (radius) was 2.76 in (nearly the largest), which were all appropriate. The resulting performance attributes values are as follows: *Battery Life* of 4.98 min, a *Blend Time* of 1 min, and a *Jar Volume* of 1.22 L These results were all deemed acceptable. The *Utility* did decrease from $936,000.00 to $597,000.00; however, the accuracy of the unconstrained model is questionable at best.

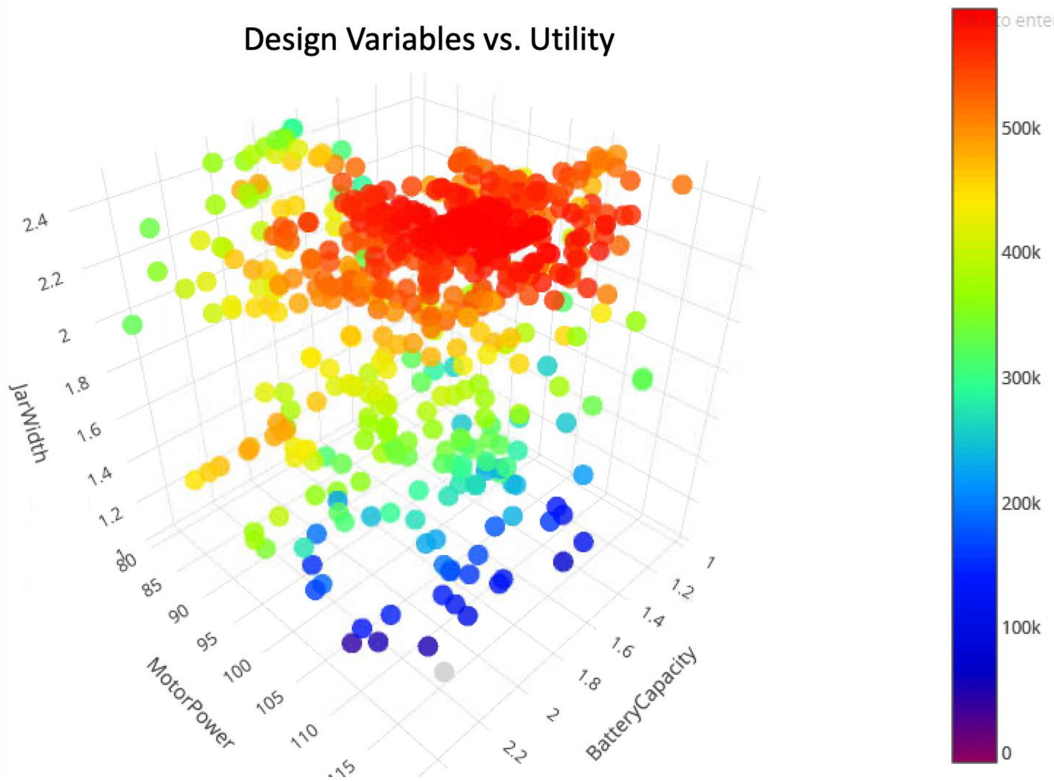

Figure 6 DAKOTA MOGA Results for Contrained Optimization, Utility

The surface plots of *Profit*, *Demand*, and *Unit Cost* also provided insight into the practicality of the optimization, shown in Figure 7. The *Profit* looked very similar to *Utility* curve, except that *Utility* flattens the high profits due to our risk aversion and increases lower profits (see the z=2.4 plane for comparison). The *Profit* and *Demand* plots look identical, which seems reasonable. The *Unit Cost* helps identify the max profit point as high *Motor Power* and high *Battery Capacity* increase cost. This *Unit Cost* also meets expectation as *Jar Width* does not change much over the constrained range.

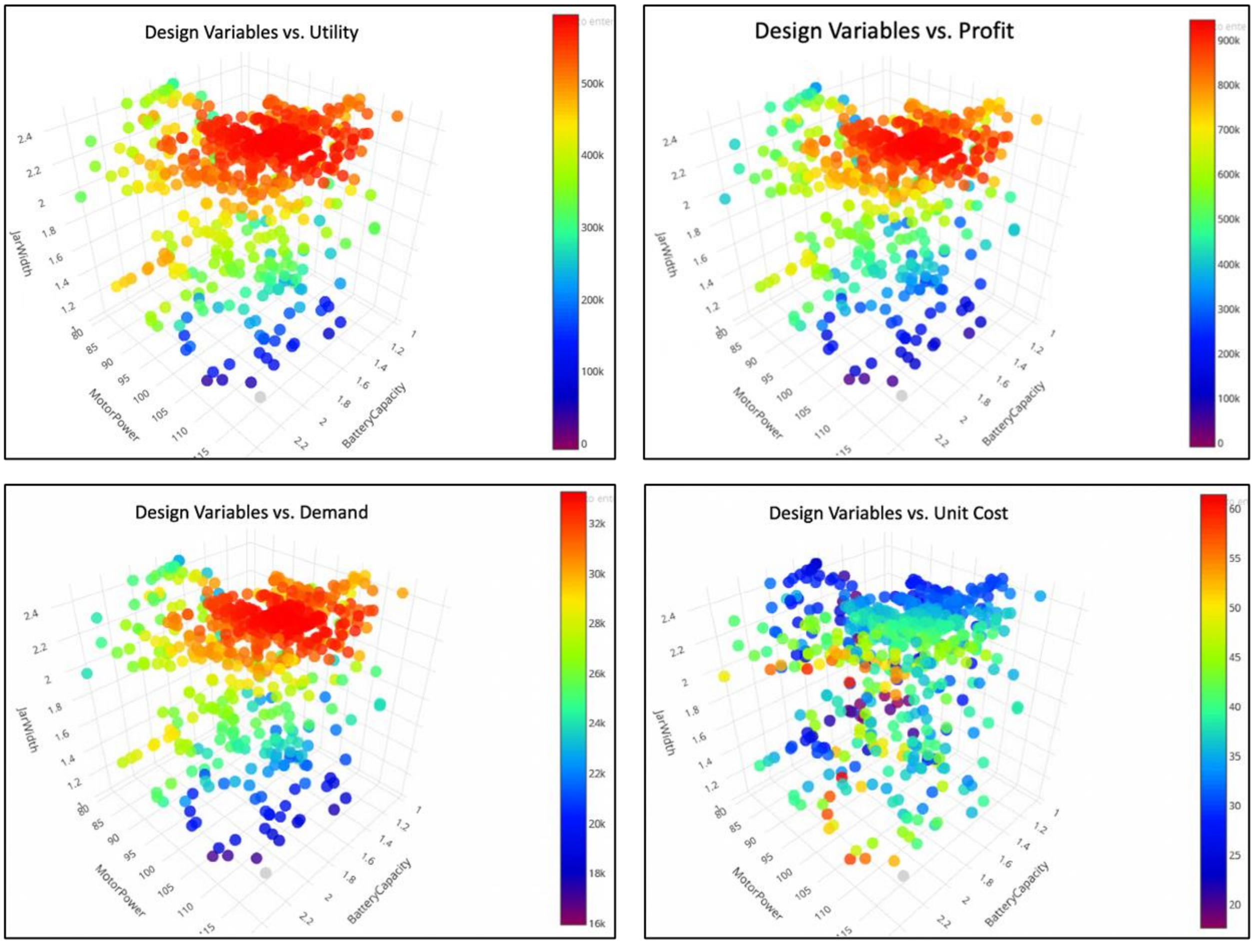

Figure 7 Additional Constrained Optimization Plots for Profit, Demand, and Unit Cost

1.2 Convergence to Optimality

After evaluating the optimized solution with DAKOTA MOGA, several other optimization algorithms were explored. The results of these optimization algorithms are summarized in Table 1 below. All optimizations identified similar results.

No.	Algorithm	Utility ($)	Design Variables			Perf. Attribute	Constraint	Run Time
			Battery Capacity	Motor Power	Jar Width	Jar Volume		
1	Noesis Adaptive Region	610,871.00	1.79179	109.746	2.50003	1.2460	0 < JarVolume < 1.25	52s
2	DAKOTA MOGA	607,831.00	1.75295	108.018	2.49409	1.24035	0 < JarVolume < 1.25	1m22s
3	DAKOTA MOGA	936,108.00	1.54052	139.132	9.85778	22.2554	0 < JarVolume < 22.955	1m32s
4	DAKOTA MOGA	935,950.00	2.48199	13.9799	0.95325	0.0104	0 < JarVolume < 1.25	2m38s
5	Noesis Adaptive Region	610,871.00	1.79266	109.735	2.50003	1.24722	0.25 < JarVolume < 1.25	54s
6	Evolve	610,860.00	1.792	109.724	2.5	1.24719	0.25 < JarVolume < 1.25	11m21s
7	Darwin	610,860.00	1.792	109.7	2.5	1.24719	0.25 < JarVolume < 1.25	8m18s
8	Noesis Sequential Quadratic	610,728.00	1.8263	110.084	2.5004	1.24765	0.25 < JarVolume < 1.25	6s
9	DAKOTA MOGA	606,398.00	1.79951	109.353	2.48782	1.23312	0.25< JarVolume > 0	2m18s
10	Darwin	535,291.00	1.824	103.4	2.2783	1.00189	0.25 < JarVolume < 1	10m55s
11	Noesis Adaptive Region	534,740.00	1.82435	103.444	2.2765	1.0	0.25 < JarVolume < 1	18s
12	Noesis Mixed Integer	(370,903.00)	2.5	153.915	1.7	0.46918	0.25 < JarVolume < 1.25	3s

Table 1 Unconstrained and Constrained Deterministic Optimization Trade Studies

The Noesis Adaptive Region Model optimizing algorithm was particularly interesting. The Noesis Adaptive Algorithm appears to work very linearly and moves through the design space to find the optimal value, as shown in Figure 8. It does not give the appearance of a wide random sampling as seen in other algorithms (e.g. DAKOTA MOGA) but then narrows down to a solution in a very linear method. The analysis highlights how *Utility* continues to improve until the optimal value is found using small increments to change the design variables, as is indicated in Figure 9.

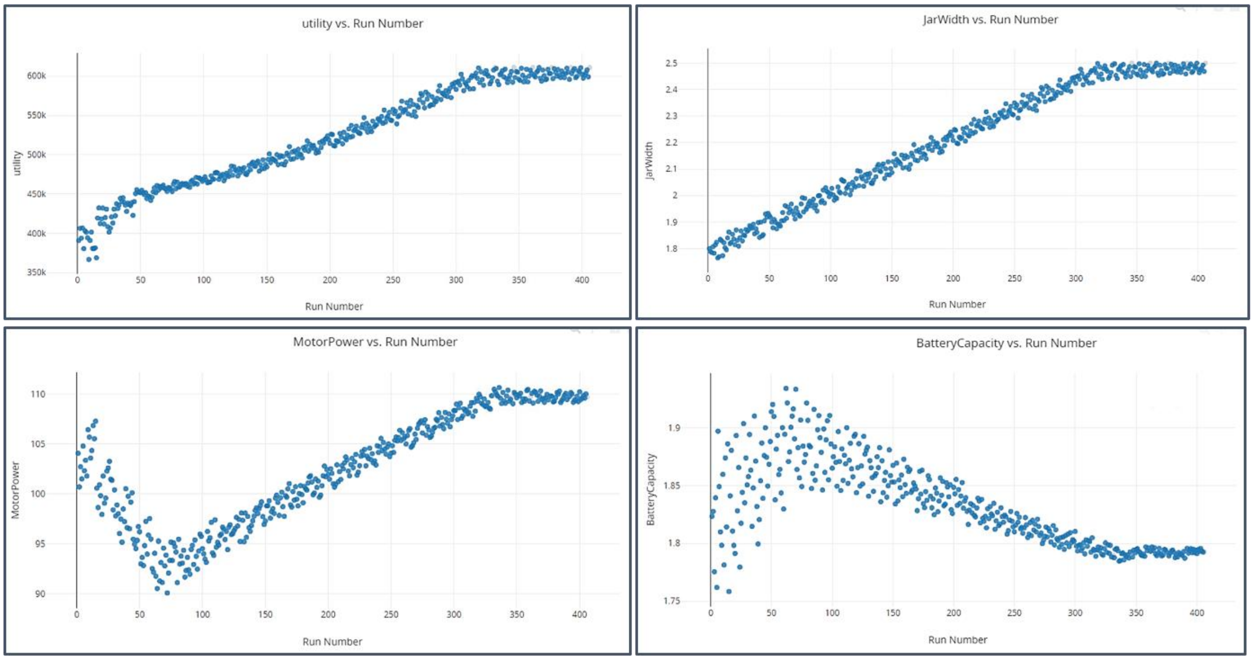

Figure 8 Constrained Optimal Solution Scatter Plots

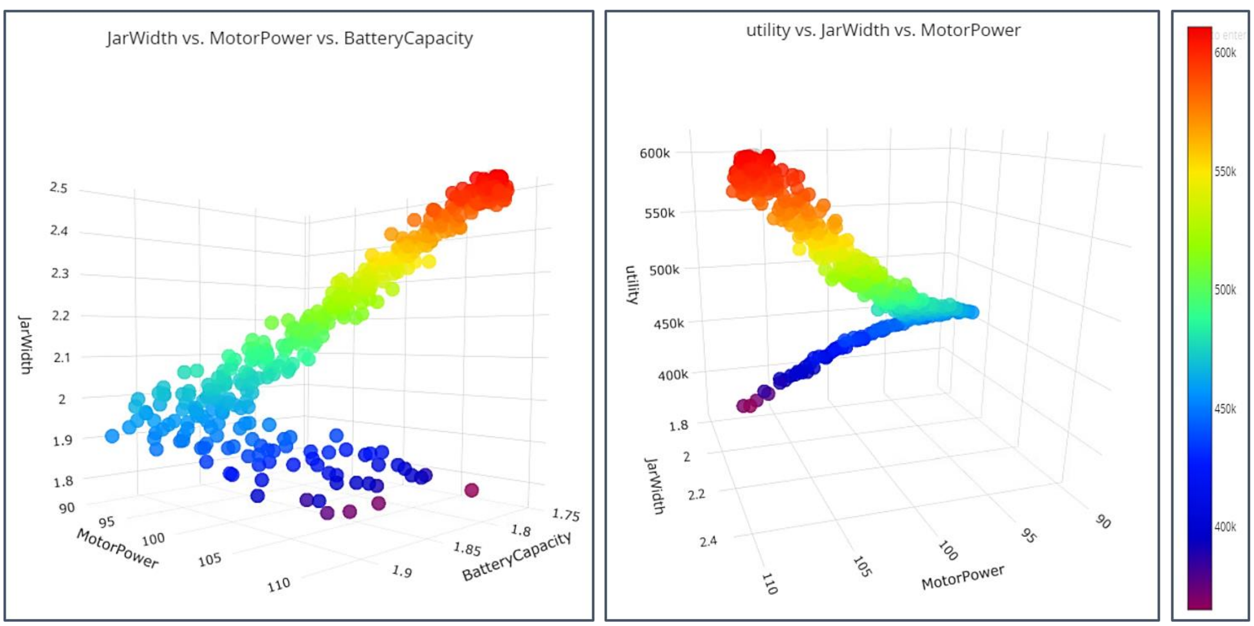

Figure 9 Constrained Optimal Solution 3D Scatter Plots, Utility

1.3 Optimal Design Solution

All the algorithms used to evaluate the deterministic model provided similar optimized results. The performance attributes all fell within the range of our initial survey questionnaire which gives us confidence in the accuracy of the model. The design variables were deemed reasonable as they align with readily available technologies. The *Demand* and *Utility* were also deemed reasonable for the given *Market Size*. Our careful exploration of design variables in our initial DOE, along with our iterative and incremental approach of performing the constrained optimization, provided a reasonably acceptable solution.

1.4 Final Design in the Market

The table below outlines a summary of the results from our optimized solution, Table 2. This product is intended for the everyday consumer use and primarily to make drinks that require powerful mixing on-the-go (e.g., smoothies, health drinks, protein shakes). As researched in our Demand Survey:

1) A *Battery Life* of 5 min is competitive with the market.

2) A *Blend Time* of 62 s is competitive, but a little longer compared to the best blender in the market.

3) A *Jar Volume* of 1.2 L is superior to the market.

Constrained Optimized Solution							
Battery Capacity	Motor Power	Jar Width	Battery Life	Blend Time	Jar Volume	Demand	Price
1.9 Amp·hr	110 W	2.5 in.	5.0 min	62 s	1.2 L	33,000	$92.37

Table 2 Constrained Optimized Solution Summary

The price of $92.37 is also competitive with high-end, well-regarded blenders on the market. (e.g., Blendjet). The MixQuick blender would do well against the Blendjet for more than just individual use-cases (e.g., couples or family use, officemate use, graduate-student study group use). Table 3 summarizes the research previously completed on competitive blenders.

Attribute Name	Units	Blendjet	VOTSUPKITDINOK	Hamilton
Price	$	99.95	23.99	18.99
Battery Life	min	5	2	5
Blend Time	s	25	60	40
Capacity	L	0.475	0.4	0.4

Table 3 Market Research

1.5 Trade-Offs

There were a few trade-offs identified in this project. One of the trade-offs is the *Jar Width* against *Motor Power*. In the DOE, we saw a “saddle” where there was an optimal *Motor Power*. At high *Motor Power* this will improve *Blend Time* but will decrease *Battery Life*. At low *Motor Power* this would increase *Battery Life* but decrease *Blend Time*. There was also an optimal *Motor Power* versus *Battery Capacity* relationship. Increasing *Motor Power* would decrease *Battery Life*, so *Battery Capacity* would need to increase but this would negatively impact *Cost*.

We also found that increasing *Battery Capacity* will positively improve *Battery Life* but will incur an increased *Unit Cost*. Increasing *Motor Power* would improve *Blend Time* but would negatively affect *Battery Life* and *Cost*. In the optimization, the knock-on effect of increasing *Motor Power* is that *Battery Capacity* would increase to improve *Battery Life* but would incur a cost to do so. Another trade-off is with *Jar Width*. Increasing *Jar Width* would increase *Capacity* but would negatively affect *Blend Time* and *Cost*. In the optimization, increasing *Jar Width* would increase *Motor Power*, which would have the knock-on effects mentioned above. Having multiple trade-offs added to the complexity of this solution. Figure 10 shows how each design variable would affect the performance attributes.

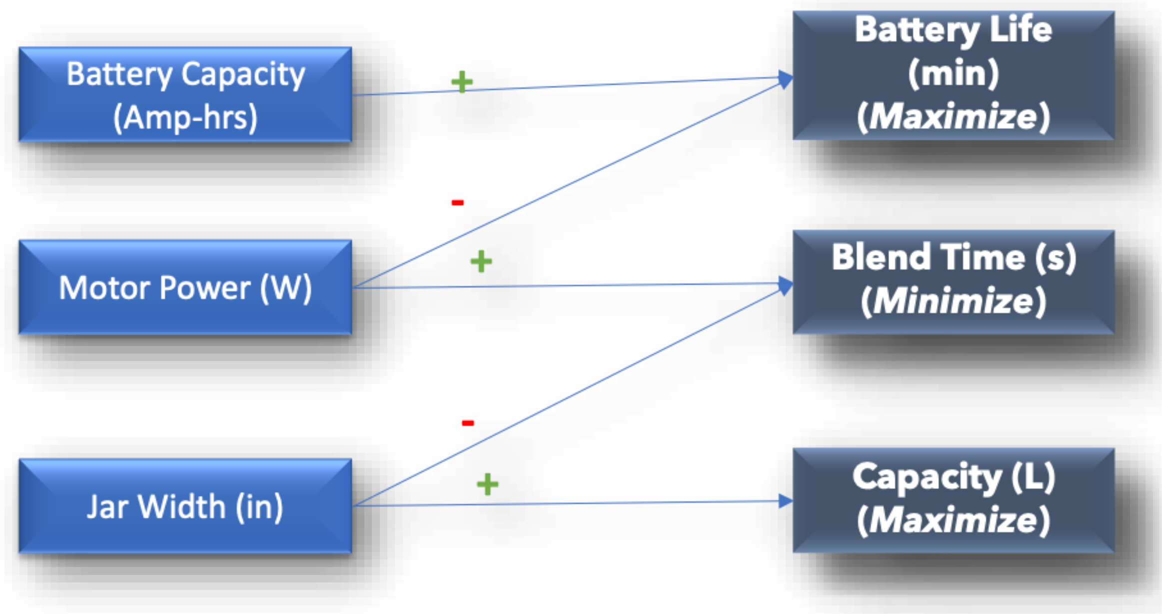

Figure 10 Design Variable Trade-Offs

(Note: Increasing any design variable would negatively impact Production Costs.)

An additional trade-off is design complexity versus optimization time. In our simple model, algorithms could operate faster with tighter bounds but would slow down with constraints. A more complex model, like one that references multiple external files rather than leveraging scripts, would demand longer evaluation times. Adding additional design variables would further slow the evaluation.

1.6 Requirement Satisfied

This section has been addressed above in Section 1.1.2 Constrained Optimization. We successfully re-ran our deterministic optimization with the requirement constraints bounding *Jar Width*. We no longer have an excessively large cup solution, nor a tiny/negative volume cup solution.

2 Stochastic Solution

After evaluating the design problem in a deterministic approach, the team sought out to find the optimum of the same design problem under uncertainty. The model used in the previous analysis was updated to include the Latin Hypercube Sampling (LHS) module. The LHS module ran a specified number of trials to calculate an expected utility average, which was then used to perform the optimization trade study. The MixQuick model was updated as shown in Figure 11 to solve the design problem under uncertainty. As shown in the figure, the LHS module was added on the bottom right.

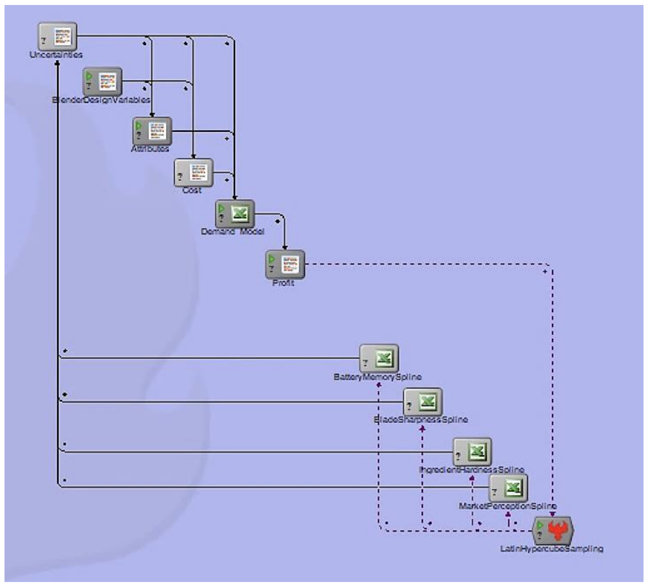

Figure 11 MixQuick Model Under Uncertainty

The Optimization Tool in ModelCenter was set up as shown in Figure 12. The objective used was the *Average Expected Utility* with a constraint on *Jar Volume* bounded between 0.25 L and 1.25 L. The design variables were also bounded as follows: *Battery Capacity* between 0 Amp·hrs and 2.5 Amp·hrs, *Motor Power* between 0 W and 150 W, and *Jar Width* between 1.7 in and 2.5 in. As explained in the previous section, the model appreciates a larger *Jar Width* and, thus, a larger *Jar Volume*, which is why we used 1.25 L as the upper bound. The Algorithm Wizard within the Optimization Tool provided a list of viable options for which algorithm to use for evaluation of the optimal design solution, shown in Figure 13.

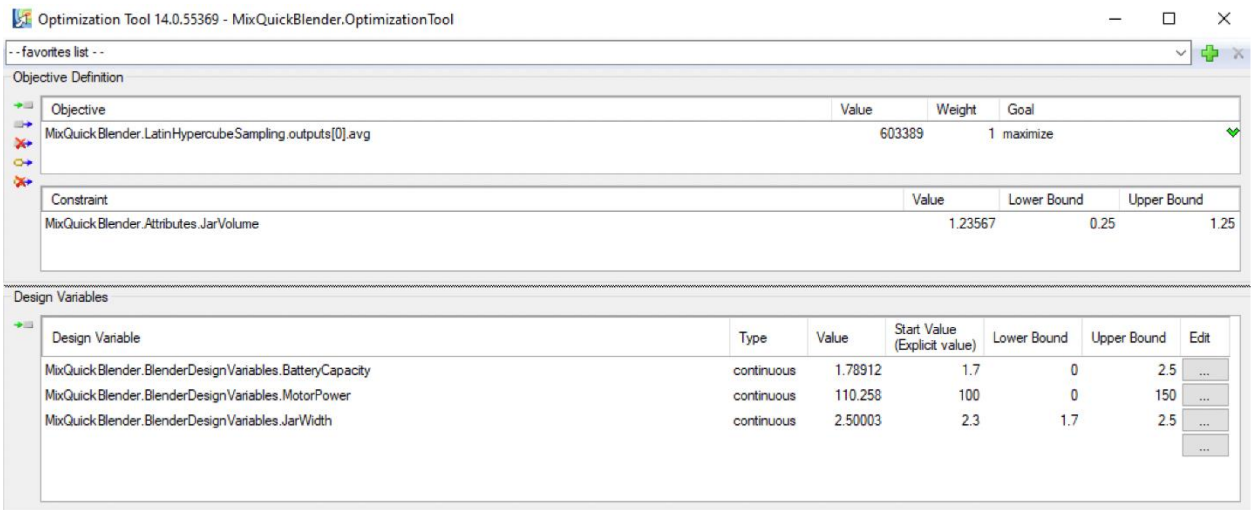

Figure 12 Set Up of the Optimization Tool for Uncertainty Solution

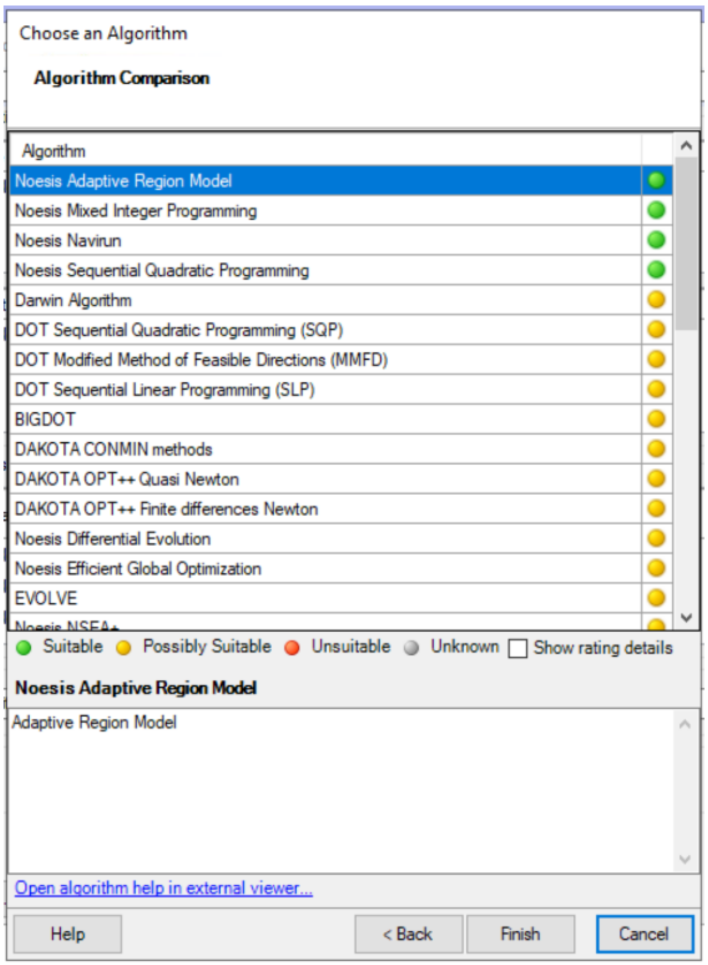

Figure 13 Algorithm Wizard Results

To select the best algorithm for the MixQuick model, the team ran trade studies using different algorithms with the model, each set to 5 trials. The algorithm that predicted the most expected average utility would be selected for further analysis. Table 4 summarizes the results of this preliminary study. As shown, the algorithms we initially tested were Noesis Adaptive Region Model, Darwin, Noesis Sequential Quadratic Programming, and DAKOTA MOGA. These were selected based on the results from the Algorithm Wizard and from what we observed during the deterministic solution. The number of trials refers to how many trials the LHS module would run before computing the average that would be used in the optimization trade study. The value of the *Expected Utility*, *Battery Capacity*, *Motor Power*, *Jar Width*, and *Jar Volume* shown in the table are the optimal values that were found in each trade study. We noted that the expected average utility and the design parameters that resulted using the Noesis Adaptive Region Model and Darwin algorithms were very similar, with minor differences. Although both algorithms were valuable options to experiment with, Darwin was too computationally expensive. Therefore, the team selected Noesis Adaptive Region Model for further analysis in finding the optimum.

Algorithm	No. of Trials	Expected Utility ($)	Design Variables			Perf. Attribute	Constraint	Run Time
			Battery Capacity	Motor Power	Jar Width	Jar Volume		
Noesis Adaptive Region Model	5	603,389	1.78912	110.258	2.50003	1.23567	0.25 < JarVolume < 1.25	4m27s
Darwin	5	603,378	1.788	110.23	2.5	1.23563	0.25 < JarVolume < 1.25	1h36m5s
Noesis Sequential Quadratic Programming	5	602,192	1.84246	107.901	2.5004	1.23609	0.25 < JarVolume < 1.25	27s
DAKOTA MOGA	5	587,564	1.78912	109.108	2.45633	1.18571	1.0 < JarVolume < 1.25	19m48s

Table 4 Summary of Preliminary Optimization Results

While the final decision of which algorithm to use was based off the previous table, we did perform additional optimization studies to experiment and observe the behavior of the resulting plots. All trade studies performed are summarized in Table 5. As stated previously DAKOTA MOGA was not a feasible option due to the prolonged computation times, seen again here in the 50-trial run. During our experimentation, we noticed that Noesis Sequential Quadratic Programming may have been the best in computation time, but the resulting scatter plots for *Expected Utility* and the design variables were illogical with several plateaus of different values.

Algorithm	No. of Trials	Expected Utility ($)	Design Variables			Perf. Attribute	Constraint	Run Time
			Battery Capacity	Motor Power	Jar Width	Jar Volume		
DAKOTA MOGA	5	587,564	1.78912	109.108	2.45633	1.18571	1.0 < JarVolume < 1.25	19m48s
DAKOTA MOGA	50	601,744	1.79232	110.215	2.48608	1.21963	1.0 < JarVolume < 1.25	4h8m43s
Darwin	5	603,378	1.788	110.23	2.5	1.23563	0.25 < JarVolume < 1.25	1h36m5s
Noesis Adaptive Region Model	5	603,389	1.78912	110.258	2.50003	1.23567	0.25 < JarVolume < 1.25	4m27s
Noesis Adaptive Region Model	50	606,806	1.78578	110.19	2.50003	1.23567	0.25 < JarVolume < 1.25	49m9s
Noesis Adaptive Region Model	100	607,024	1.78871	110.119	2.50003	1.23567	0.25 < JarVolume < 1.25	1h11m30s
Noesis Adaptive Region Model	250	606,863	1.78844	110.066	2.50003	1.23567	0.25 < JarVolume < 1.25	4h25m2s
Noesis Sequential Quadratic Programming	5	602,192	1.84246	107.901	2.5004	1.23609	0.25 < JarVolume < 1.25	27s
Noesis Sequential Quadratic Programming	25	603,929	1.84237	107.727	2.5004	1.23609	0.25 < JarVolume < 1.25	2m10s
Noesis Sequential Quadratic Programming	50	605,065	1.84548	107.133	2.5004	1.23609	0.25 < JarVolume < 1.25	4m14s
Noesis Sequential Quadratic Programming	100	605,217	1.84643	107.112	2.5004	1.23609	0.25 < JarVolume < 1.25	8m49s
Noesis Sequential Quadratic Programming	500	604,617	1.84407	107.511	2.5004	1.23609	0.25 < JarVolume < 1.25	42m16s

Table 5 Summary of All Optimization Studies Under Uncertainty

Even so, Noesis Adaptive Region Model was still the algorithm of choice because of the highest *Expected Utility*. Table 6 summarizes the results of the optimization studies using the algorithm of choice. We conducted trade studies multiple times, increasing the number of trials that the LHS module would complete for optimization for each iteration. As shown in Table 6, we conducted trade studies using 5, 50, 100, and 250 trials. We attempted to conduct a study with 500 trials twice but were unsuccessful because the tool timed out and failed. Again, we notice the positive influence a larger jar has on our model, as seen by the optimal value that was found for *Jar Width*. It was found to be at its maximum each iteration and, therefore, so was *Jar Volume*.

No. of Trials	Expected Utility ($)	Design Variables			Perf. Attribute	Constraint	Run Time
		Battery Capacity	Motor Power	Jar Width	Jar Volume		
5	603,389	1.78912	110.258	2.50003	1.23567	0.25 < JarVolume < 1.25	4m27s
50	606,806	1.78578	110.19	2.50003	1.23567	0.25 < JarVolume < 1.25	49m9s
100	607,024	1.78871	110.119	2.50003	1.23567	0.25 < JarVolume < 1.25	1h11m30s
250	606,863	1.78844	110.066	2.50003	1.23567	0.25 < JarVolume < 1.25	4h25m2s

Table 6 Summary of Selected Optimization Studies Under Uncertainty (Using Noesis Adaptive Region Model)

The plots that follow (Figures 14 through 17) are two-dimensional scatter plots, where the x-axis is the run number and the y-axis is the computed value of *Expected Utility*, *Battery Capac*ity, *Motor Power*, or *Jar Width*, as applicable. Each set of plots are considered one figure and each individual plot is labeled on the bottom left corner denoting how many trials were used to generate it. In all these plots, we can make the following observations:

- As the number of trials are increased, the optimal value is found sooner. In other words, the plots converge at an earlier run number than its predecessor (less trials). Take the *Expected Utility* plots, Figure 14, as an example. The number of runs completed to find the optimal *Expected Utility* at 5 trials is approximately 165 runs. At 250 trials, however, it's found after approximately 155 runs.
- As the number of trials are increased, less clusters are formed on the scatter plots. They become more precise with less noise. This is expected due to the nature of LHS, which aims to spread the sample points more evenly across all possible values.
- With 250 trials, the predicted values for each parameter, or the optimal solution, are as follows:
  - *Expected Utility* = $606,863
  - *Battery Capacity* = 1.8 Amp·hrs
  - *Motor Power* = 110.1 W
  - *Jar Width* = 2.5 in.
  - *Jar Volume* = 1.2 L

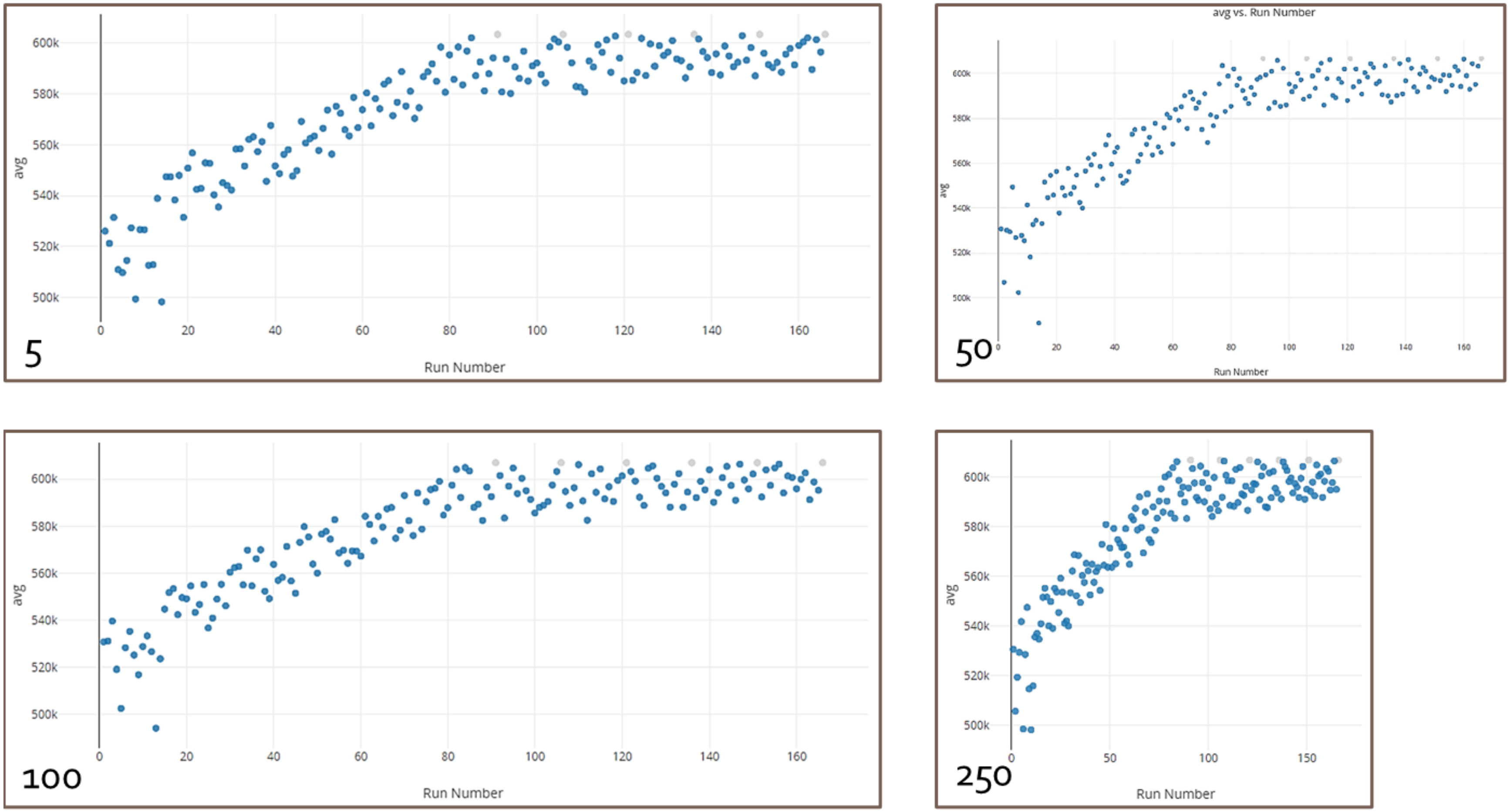

Figure 14 Expected Utility Scatter Plots

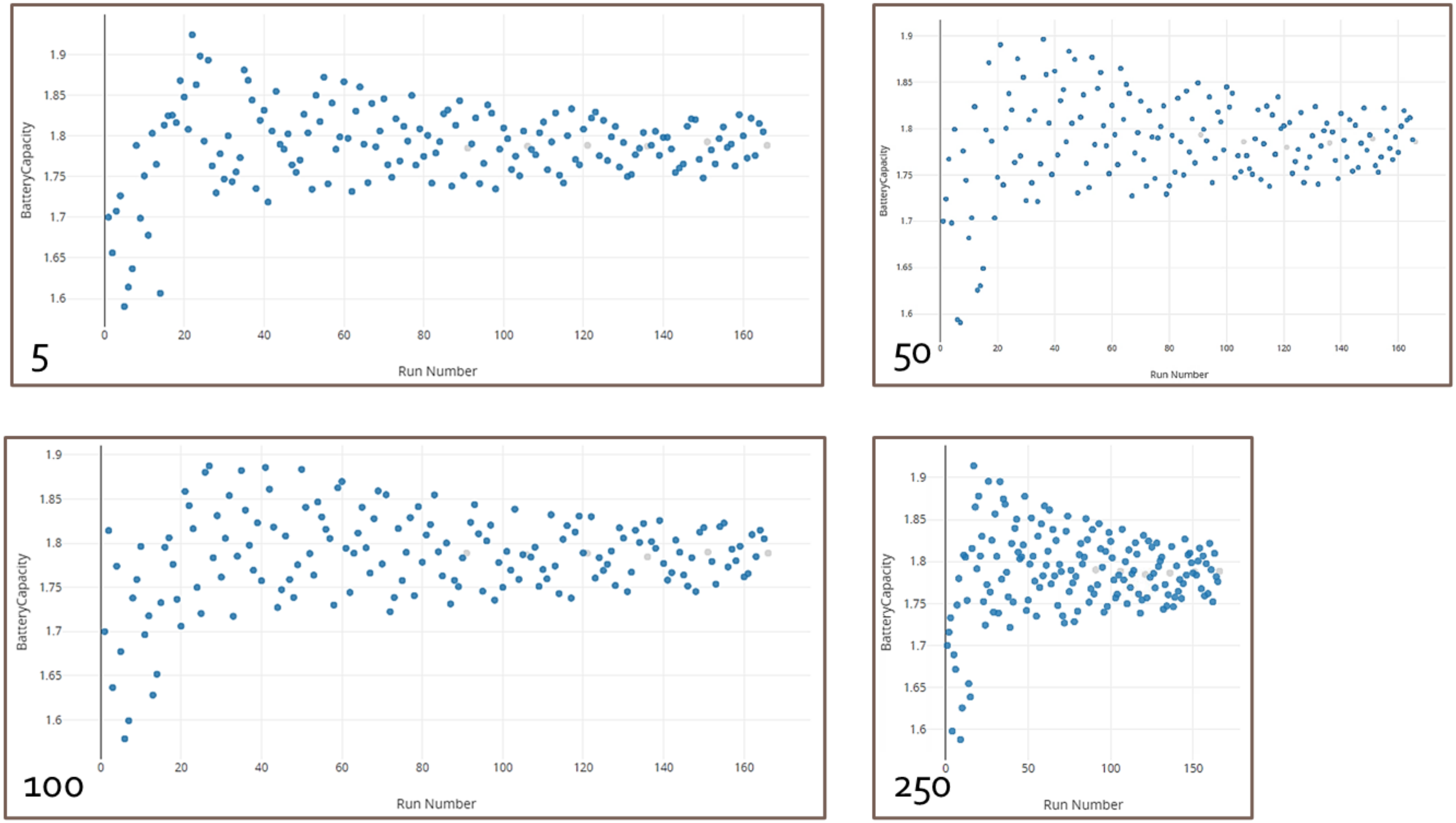

Figure 15 Battery Capacity Scatter Plots

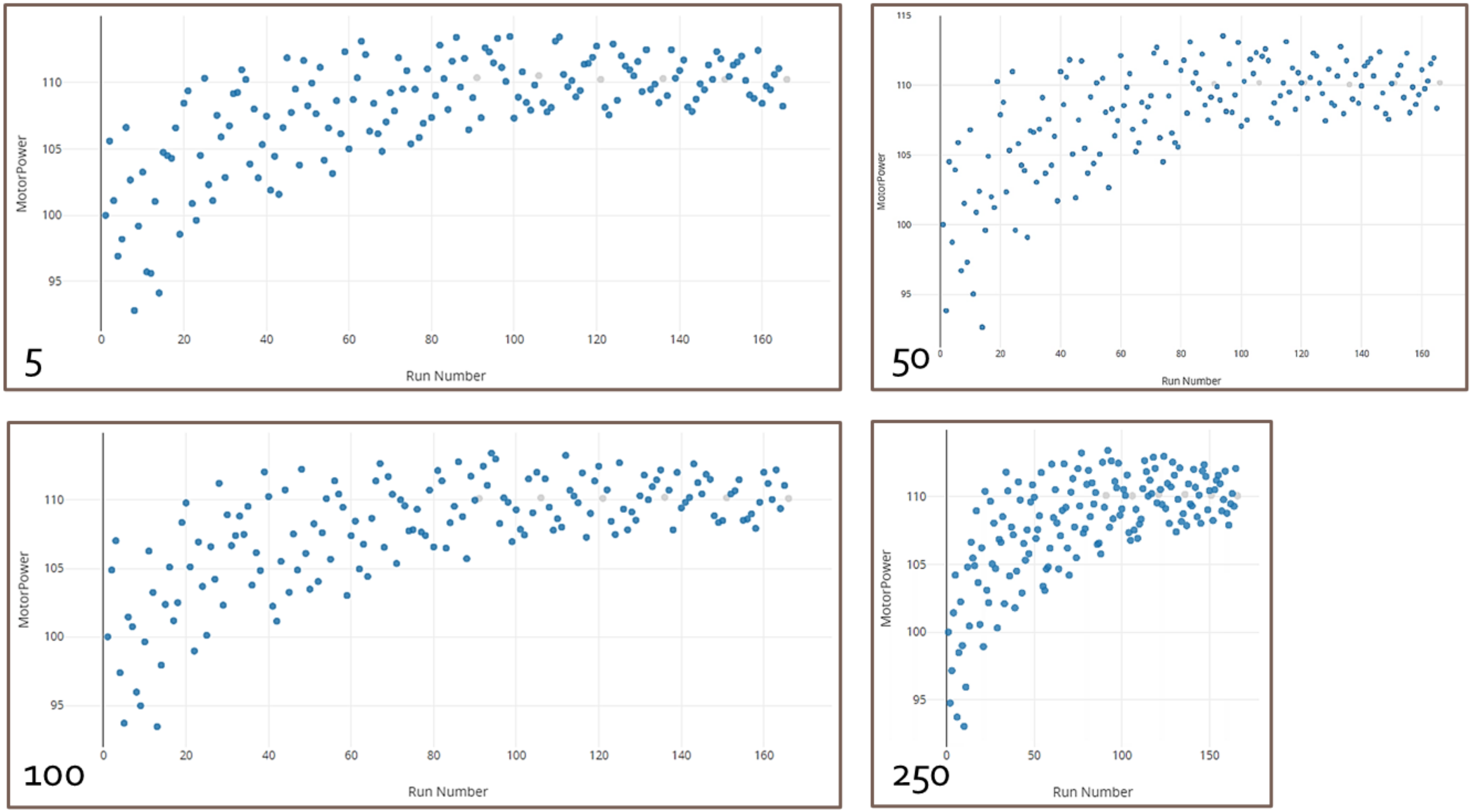

Figure 16 Motor Power Scatter Plots

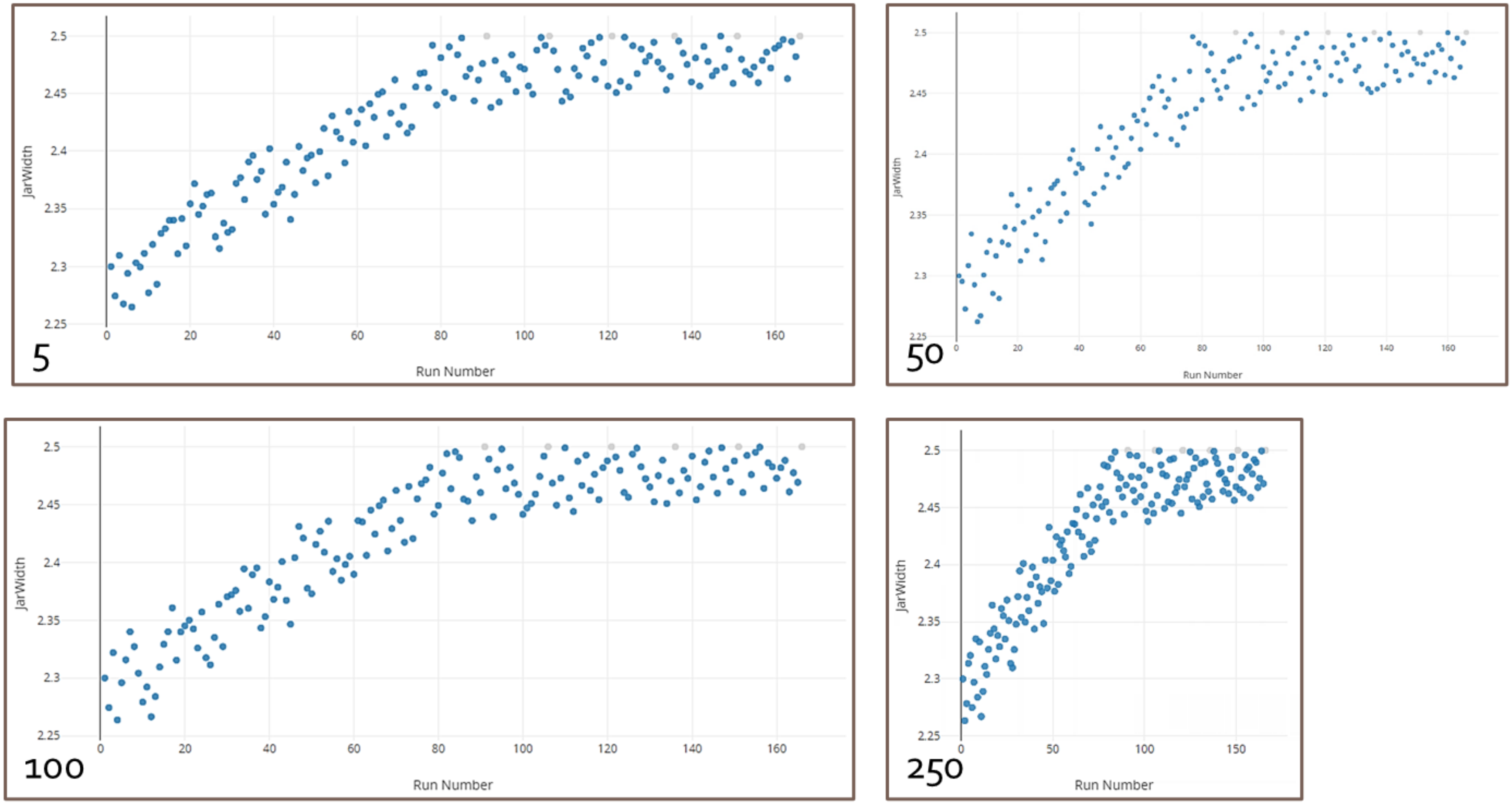

Figure 17 Jar Width Scatter Plots

A parallel coordinates plot was also generated for each one of these trade studies, so that we could observe how the different values relate to each other. Like the plots shown above, the set of plots are considered one figure and each individual plot is labeled on the bottom left corner denoting how many trials were used to generate it. The color legend is also included in the figure. The parallel coordinates plot has 5 parallel axes representing *Expected Utility*, *Jar Volume*, *Battery Capacity*, *Motor Power*, and *Jar Width* (in order), each scaled differently as is appropriate for that specific parameter. The connected line segments represent the combination of the data points for the five parameters. The colored lines identify the *Expected Utility* for each of these combinations, where red represents the highest utility and purple represents the lowest, ranging from $500,000.00 to $600,000.00. Figure 18 displays the parallel coordinates plots that were generated. From these plots, we can make the following observations:

- The more trials that are used for analysis, the less clusters of data we see. In these plots, this is seen in how the lines are broadened each time the trials are increased. It becomes really distinct when comparing the red lines in the plots. The red is denser in some areas within the plots of less trials in comparison to the 250-trial plot.
- The optimal solution that would generate the most *Expected Utility* (shown in red) is at the greater end of the bounds for *Motor Power* and *Jar Width*, while *Battery Capacity* is mostly in the mid-high range within the bound, as we observed in the scatter plots.

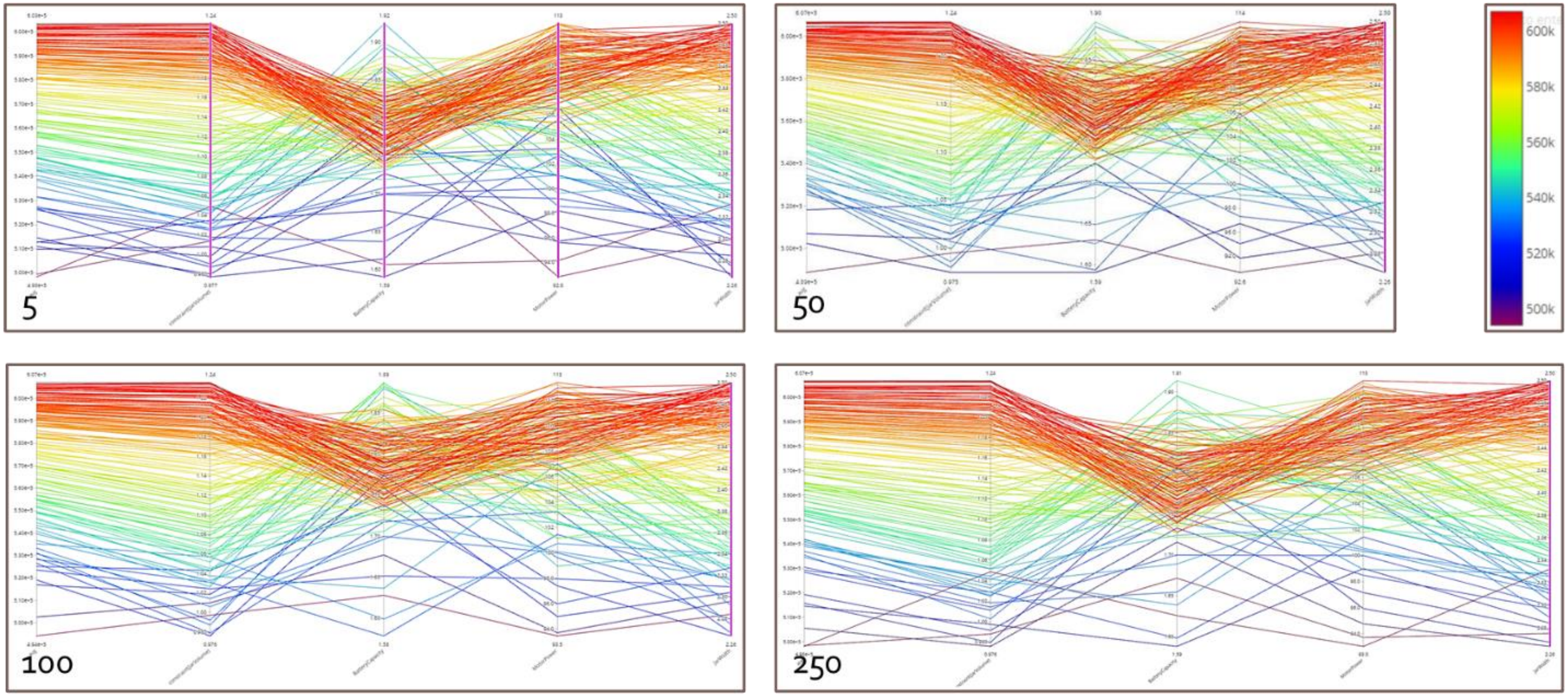

Figure 18 Parallel Coordinates Plots

2.1 Deterministic vs. Stochastic Solution

The deterministic and stochastic solutions indicated very similar design solutions. As discussed in this report, both solutions were found using the Noesis Adaptive Region Model algorithm in the Optimization Tool. Table 7 summarizes the results of both solutions. From the summary, we note that the design variables, specifically *Battery Capacity* and *Motor Power*, are within ± 0.5. *Jar Width* is predicted to be the same value – at the maximum value of its bounds. Interestingly, we found that the *Jar Volume*, however, were not equal. Upon further investigation, we found that a variable of the *Jar Volume* equation, *UC_JarThickness*, was inadvertently changed from 0 to 0.01 while changing tasks. This small change caused for different results in the *Jar Volume* but is still small enough to be considered negligible. The most significant difference between the two solutions is in the *Expected Utility*. The deterministic solution, which found its optimum after 400 runs, predicts a utility of $610,871.00. The stochastic solution, on the other hand, predicts a utility of $606,389.00 after completing about 155 runs.

Noesis Adaptive Region Model		**Deterministic Solution**	**Stochastic Solution *(250 trials)***
Expected Utility ($)		610,871.00	606,389.00
Design Variables	**Battery Capacity (Amp·hrs)**	1.79	1.79
	Motor Power (W)	109.74	110.07
	Jar Width (in.)	2.50	2.50
Perf. Attribute	**Blend Time (s)**	63.64	63.07
	Battery Life (min.)	4.60	4.58
	Jar Volume (L)	1.247	1.24

Table 7 Deterministic vs. Stochastic Solution Summary

There is a difference of $4,482.00 between the Expected Utility of both solutions. The reason for this difference lies in the different methods that were used to solve the design problems. The deterministic solution set the four uncertainty variables at a constant – the most likely value for that uncertainty. Additionally, the Monte Carlo analysis used for the deterministic approach relies on repeated random sampling to obtain numerical results. In random sampling, new sample points are generated without considering the previously generate sample points. Because it relies on pure randomness, it can be inefficient, resulting in some points clustered closely, while other intervals within the spaced get no samples. Alternatively, the stochastic solution was found without setting the uncertainty values at a constant and using LHS. LHS spreads the sample points more evenly across all possible values, reducing clusters of data. In many cases, LHS outperforms Monte Carlo analysis substantially yielding more accurate results. This is also observed by the number of runs completed. The Monte Carlo approach used for the deterministic solution ran just over 400 runs, whereas the LHS approach used in the stochastic solution ran just over 150 runs. Figure 19 shows the *Expected Utility* plots of both solutions to demonstrate where it converges.

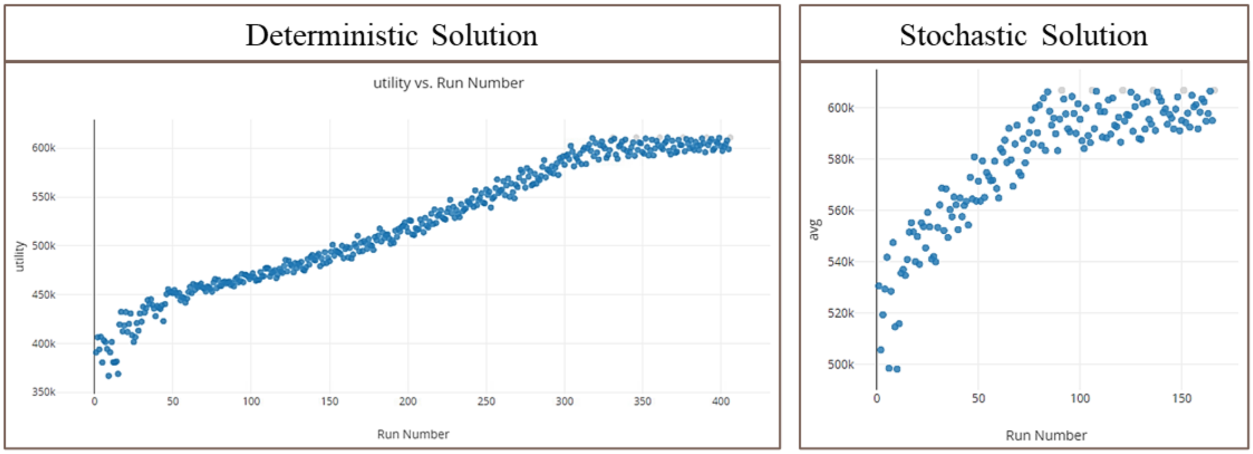

Figure 19 Deterministic vs. Stochastic Solution, Expected Utility Plots

3 Lessons Learned

3.1 Insights

Initially, the design team struggled to fully appreciate the unconstrained model. The team began with odd results, but as constraints were incorporated, the model yielded more reasonably acceptable results.

The best results were achieved when multiple iterations of the model are conducted. This is a good practice that's necessary to obtain reasonably acceptable models. However, it must be noted that multiple iterations are directly proportional to an increase in simulation times. The design team exercised much patience with each increase in runs, as the simulation times could sometimes become excessive. Although multiple iterations generally lead to better results, there is a cutoff point after which the improvement in results is marginal compared to the resources necessary to find the convergence point.

The design team noted that the model returned more reasonable results when exponential functions were incorporated, that is, local minimums/maximums for non-trivial results. This is compared with relying solely on linear equations. Exponential relationships add greater complexity to optimization than linear relationships and allow for a more interesting problem in general. The use of scripts in ModelCenter versus in Excel reduced model complexity. With scripts embedded in the model, it is much easier and quicker for it to extract information required to execute computations and simulations subsequently.

It was determined that the Demand Surveys work most effectively when it spans over a wide range to identify maximum and minimum preferences. For instance, our design model was unbounded at the maximum value for *Jar Volume*; despite the highest utility value achieved, this was not a feasible solution since having an over-sized cup is not practical for the consumer. In another scenario with constraints in place, the result was also a high utility value but was another infeasible solution with an extremely low value for *Jar Volume* was achieved. This scenario indicates an exceedingly small cup size unrealistic for use by the consumer.

3.2 Unanticipated Obstacles and Challenges

Given the time constraints to develop a design model for MixQuick, we down selected to three design variables early in the process. We believe we did a decent job selecting motor power, battery capacity, and jar width. If time permitted, we could have refined our variables better by adding other design variables. Jar height, for example, could have been added as a fourth variable to improve the blend time equation. We could have also included blade sharpness, or even the number of blades. We believe we made reasonably acceptable decisions early on considering the accelerated modeling schedule we had to accommodate.

Without a shared drive, it was quite a daunting task maintaining links in ModelCenter to external files (e.g. Demand Survey). Life would have been much easier with the availability of a shared drive to recognize and capture files needed to synchronize with ModelCenter.

Installing/extracting LHS after each ModelCenter simulation was an unforeseen inconvenience following the first driver installation. Each task in ModelCenter could not simply be executed without the installation of the LHS driver again, taking away precious time that could have otherwise been allocated to other key activities. Additionally, the optimization runs for the utility plots times out in some instances. After several hours of runs, one would think the utility plots would converge at some point, but, unfortunately, all hopes were lost when they stopped progressing.

3.3 Evaluating the MixQuick Model

Models are only as good as the information you put into them. As such the design team did thorough research using reputable sources in determining baseline ranges for the design variables, uncertainty variables and the market comparisons. Even with this research, there are bound to be inaccuracies, and these can be rolled forward into the model. The model aims to provide a realistic estimation at best.

In order to create an effective model that linked the design variables to the performance attributes, physical equations needed to be crafted (e.g. Blend Time is not obvious). These physical equations were not perfect and started out quite rudimentary. As the team's understanding of the design decision improved, the physical formulae were adjusted in an iterative and incremental fashion. There is still much room for improvement, but this model provides a solid foundation upon which to further develop.

Finally, the team considered the impact from inaccuracies with the Demand Survey. Inaccuracies present here would be carried forward into the model. The updated demand model used the elicited responses on user preferences from the entire PMASE cohort (41 persons). It is possible that more realistic estimations could have been captured by increasing the number of people that completed the survey. In short, the design team may not have identified user preferences accurately (e.g. *Jar Volume*).

References

- https://towardsdatascience.com/parallel-coordinates-plots-6fcfa066dcb3
- https://www.statisticshowto.com/latin-hypercube-sampling/
- https://analytica.com/latin-hypercube-vs-monte-carlo-sampling/