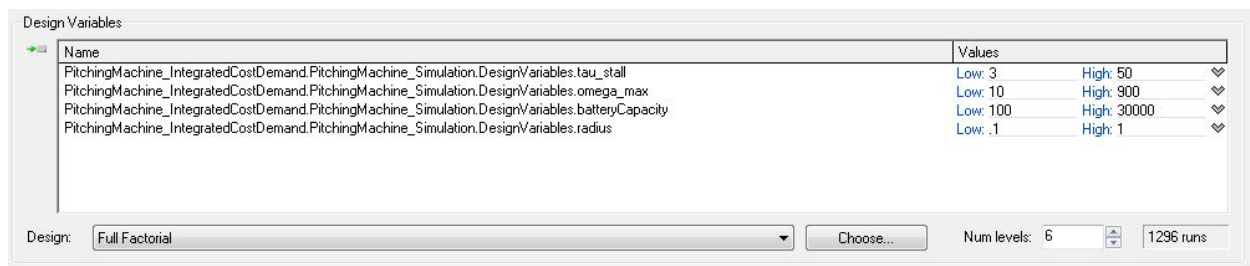


ME6105: HWG5
Group 11: Baseball
13 December 2017

Task 1: Solve the Design Problem Deterministically

DOE Revisit

Prior to running the optimizer deterministically, the DOE from HWG4 Task 5 was revisited/rerun due to changes that needed to be made in the model. It also makes a good starting point for the optimization performed in Task 2 of this assignment. The changes in the model included changing the variable types (this resulted in stratified/discretized values in the previous assignment) and updating the cost calculator for the DC motor to be a function of power (this function was also based on the larger data set collected by the entire class). The DOE was run with the following bounds for the study's design variables:



Name	Values
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.tau_stall	Low: 3 High: 50
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.omega_max	Low: 10 High: 900
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.batteryCapacity	Low: 100 High: 30000
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.radius	Low: .1 High: 1

Design: Full Factorial Choose... Num levels: 6 1296 runs

Figure 1: Revisited design variables

These values are different from those used in HWG4. The results from that DOE showed trivial optima at small/unrealistic values for the stall torque and no-load speed, so the lower bounds were increased in this study to try and eliminate that issue. Also, some of the results from the DOE in HWG4 showed a significant tradeoff between operating time and maximum pitch speed (in the realm of pitch speeds at <10mph with an operating time of multiple hours compared to pitch speeds >100mph and running for a total of less than 5 minutes). The realization was that the battery capacity upper bound (5000 mAh) was much too small as this is roughly the twice the size of a modern smartphone Li-ion battery. The upper bound for battery capacity is increased in this DOE to 30Ah, which is still expected to be too high but should be all encompassing within the design space. A reference of a small car battery being roughly 45Ah was used in setting this limit.

With these considerations in mind, the DOE was run again in an effort to avoid these scenarios. The responses were the same as HWG4, shown below in the figure.

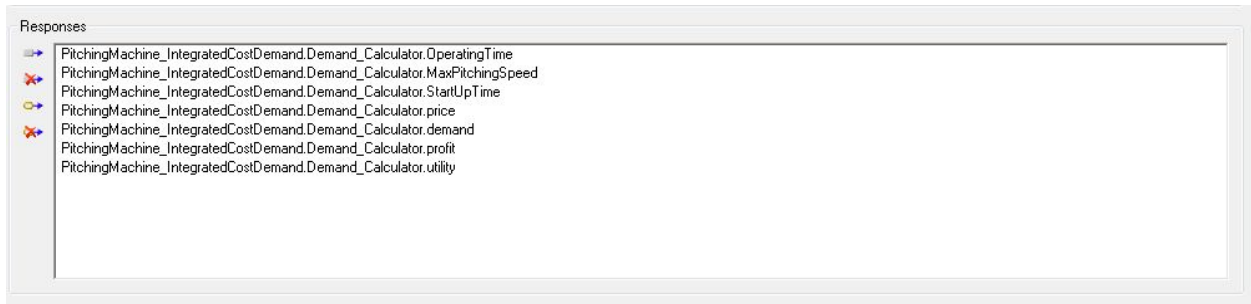


Figure 2: The responses for the DOE

Also equivalent to HWG4, the DOE run was a 6 level full factorial. The resulting utility surfaces as a function of 2 design variables are shown in the six figures below.

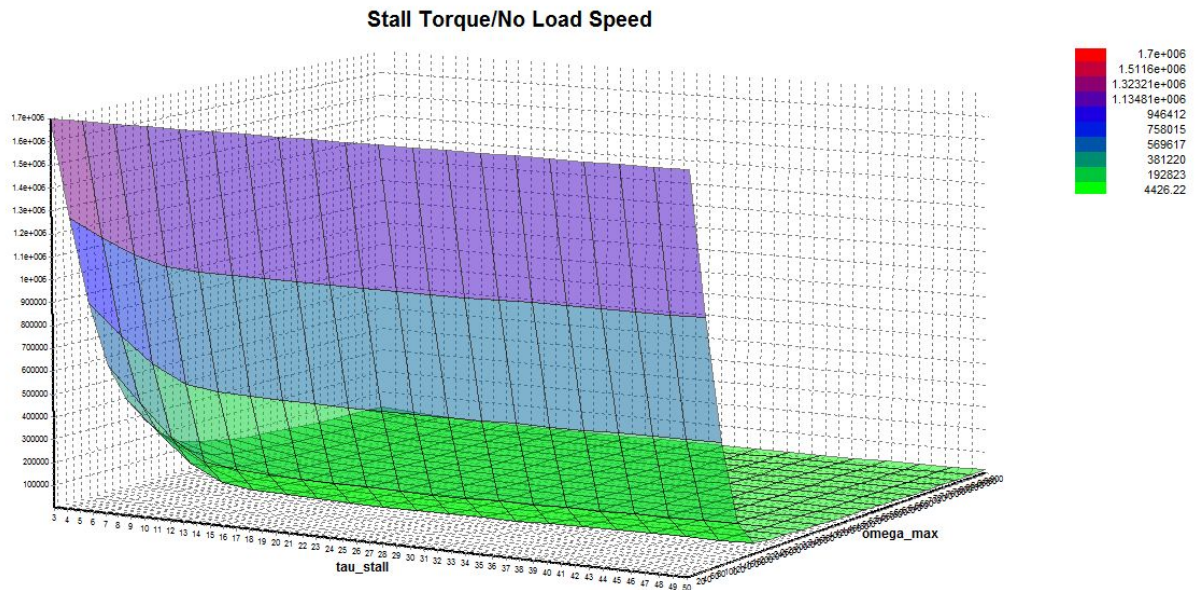


Figure 3: The utility as a function of the stall torque and no load speed

The stall torque/no load speed surface plot displays a similar issue as G4; a trivial optima exists for a condition of no load speed being as small as possible. This is likely a direct output of the motor cost model; for a given stall torque, a smaller no load speed indicates lower power usage. Given the cost function defined as it is relative to power, a lower no load speed will return lower cost and higher utility. The utility surface being relatively constant while moving along a given value of no load speed is likely due to the inverse relationship between the stall torque and no load speed as discussed in lecture.

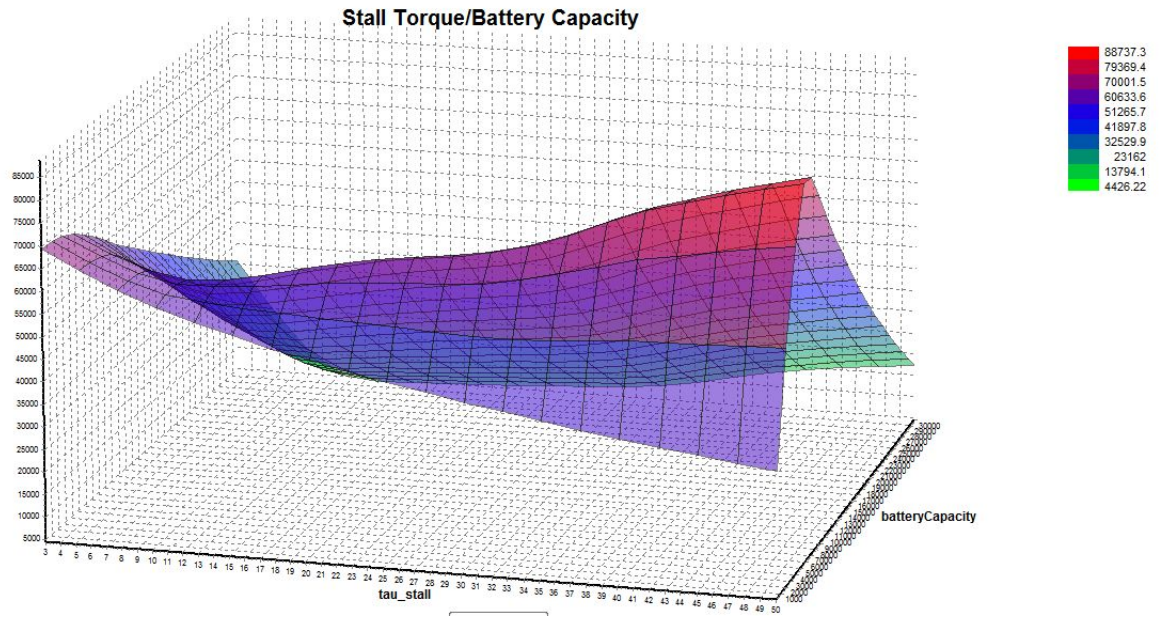


Figure 4: The utility as a function of the stall torque and the battery capacity

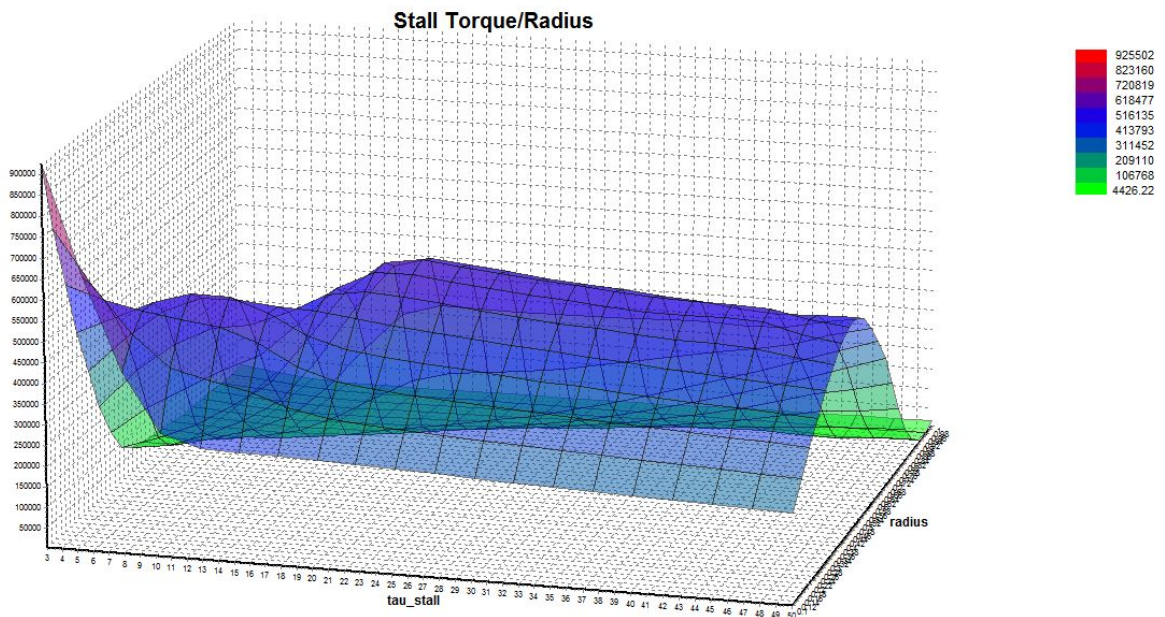


Figure 5: The utility as a function of the stall torque and the wheel radius

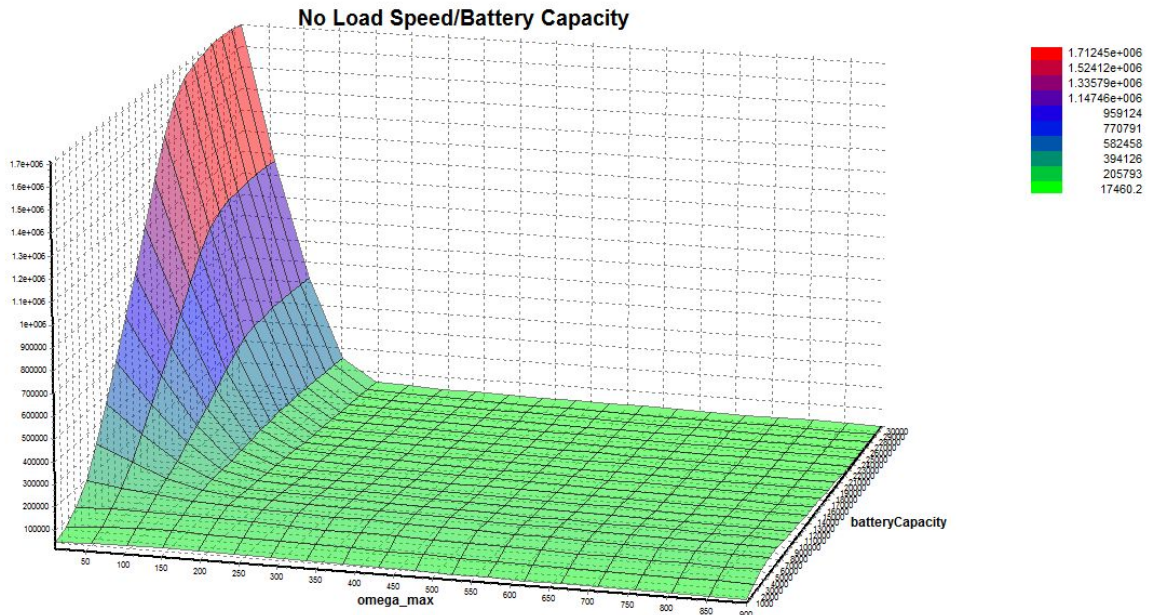


Figure 6: The utility as a function of the no load speed and battery capacity

Similar behavior is seen in the utility surfaces for no load speed and battery capacity. As previously mentioned, a lower no load speed corresponds with lower power for a given torque. Lower power consumption and a higher availability of power will result in a lower cost and a longer operating time, thus the trivial utility for a 'smaller is better' no load speed condition and a 'larger is better' condition for the battery capacity.

The condition shown in the no load speed/radius utility surface is likely the combined result of the lower power consumption as well as a failure of the utility model in highly penalizing low pitch speeds. A small no load speed should in theory return a smaller pitch speed and thus a lower utility. When comparing the response of operating time and pitch speed as a function of no load speed and radius, the similarity in surface shape between operating time and utility is easily seen. As the no load speed increases, the pitch speed follows a general trend of increasing - despite this, the utility generally decreases as the no load speed increases. Thus, the utility is strongly sensitive to the operating time and somewhat insensitive to the pitch speed.

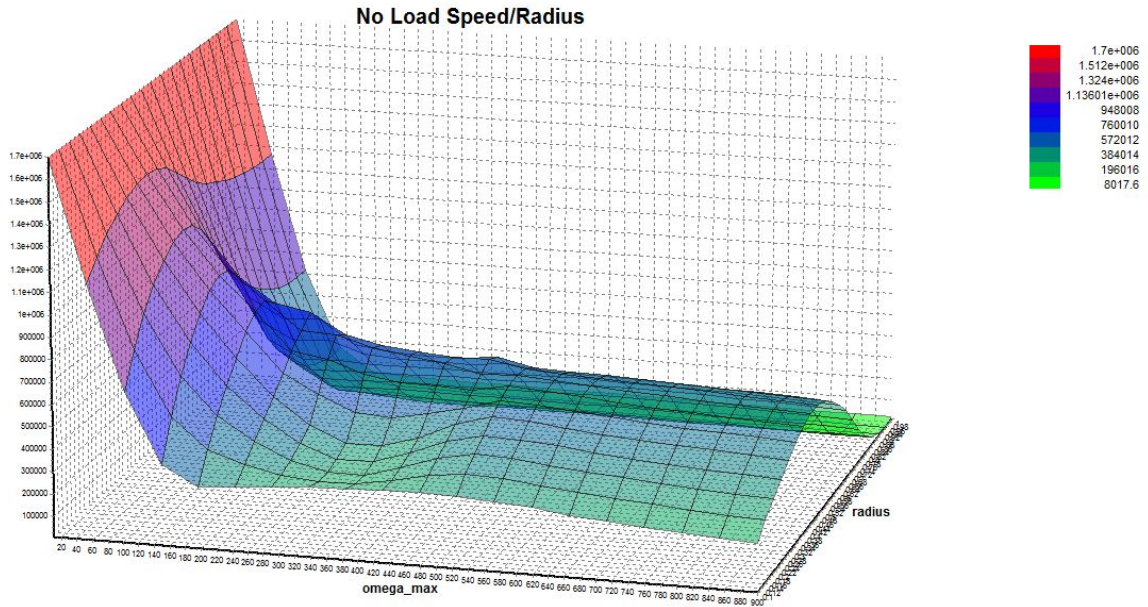


Figure 7: The utility as a function of the no load speed and the wheel radius

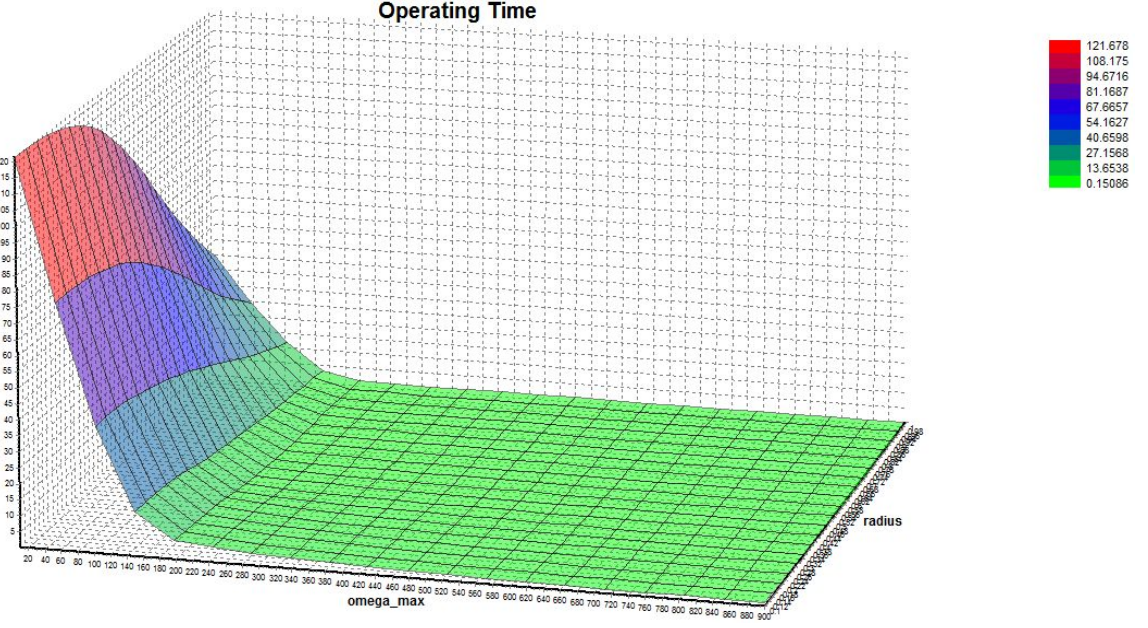


Figure 8: The Operation time vs. the wheel radius and no load speed

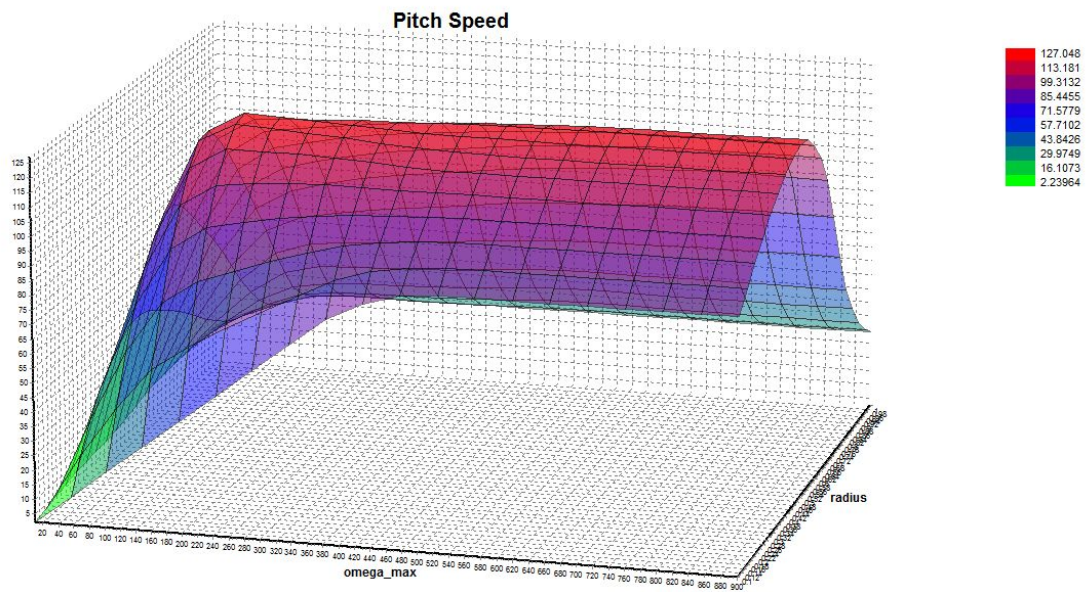


Figure 9: The pitching speed vs. no load speed and the wheel radius

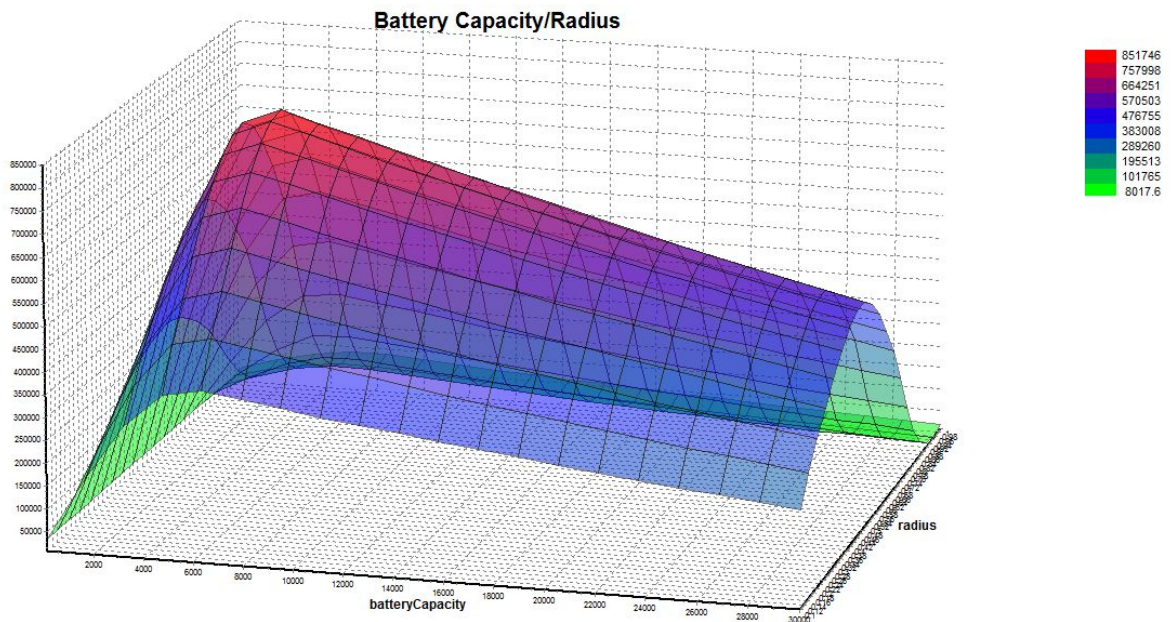


Figure 10: The utility vs. the wheel radius and the battery capacity

The utility surfaces are still shown to be smooth and thus a gradient-based approach is expected to be successful. Based on the utility values returned by the DOE, the best 'location' at which to begin the optimization is characterized by the design variables at the following values:

Stall Torque: 26.1338N*m
No Load Speed: 321.483 rad/s
Battery Capacity: 2599.97 mAh
Wheel Radius: .41236m

The bounds for the optimization are selected based on the utility surface data to be the following:

Stall Torque: [8,50]
No Load Speed: [50,350]
Battery Capacity: [2000, 15000]
Wheel Radius: [0.15, 0.7]

Another limitation of the model realized during these analyses was how certain design alternatives could result in the return of a 0 start time. Obviously, this goes against physical principles given the inertia of the wheels. The startup time calculation is a derivative-based approach that records the simulation time when the angular acceleration of the wheels comes within a specified tolerance of zero. If the wheels never reach this tolerance, then the startup time is recorded as zero (the initialized value of the variable). This is a limitation as in this case the model should ideally recognize that the wheels have never reached steady state, so the startup time is some time greater than the simulation time. Ideally this case should be penalized by the model as a zero startup time is essentially rewarded within the demand model. This skews the optimization results, which will be discussed further in Task 2.

Deterministic Optimization

Utilizing the updated cost calculator, the deterministic optimization was run using the values listed above as start values. To begin, a non-gradient-based algorithm was selected (Hooke-Jeeves) with the intention of later comparing this non-gradient-based approach with both gradient based and pattern search algorithms. The setup details from the optimization results are shown in the Figure below:

Problem Definition			
<i>PitchingMachine_IntegratedCostDemand.OptimizationTool</i>			
Objective Function(s)			
Name	Weight	Goal	Solve for Value
PitchingMachine_IntegratedCostDemand.Demand_Calculator.utility	1	maximize	0
Continuous Design Variables			
Name	Start Value	Lower Bound	Upper Bound
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.radius	0.41236	0.15	0.7
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.tau_stall	26.1338	8	50
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.omega_max	321.483	50	350
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.batteryCapacity	2599.97	2000	15000
Algorithm Used:			
Hooke-Jeeves algorithm			
Algorithm Options			
Optimization Parameters			
MaxEvaluations			1000
MaxIterations			100
StepSizeReductionFactor			0.5
StepSizeTolerance			1e-006

Figure 11: The optimization results (Hooke-Jeeves)

The results (returned as the best design) are shown in the next figure below; the no load speed is shown to present the largest difference from the start value at roughly a 50% difference, but the remaining design variables are within about 20% of the original starting value.

Best Design		
<i>Run Number 307</i>		
Objective(s)		
Name	Value	
PitchingMachine_IntegratedCostDemand.Demand_Calculator.utility	1.11461e+006	
Design Variable(s)		
Name	Start Value	Value
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.radius	0.41236	0.43813
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.tau_stall	26.1338	35.4824
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.omega_max	321.483	160.741
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.batteryCapacity	2599.97	2016.5

Figure 12: The best design (Hooke-Jeeves)

The simulation converged in 316 runs (noted for comparison to the other algorithms below). The run history and convergence history are shown in the figures below. The run history illustrates results that align with the nature of Hooke-Jeeves (regarding its nature of stepping 'back and forth'). It is easily seen that the utility spikes and bottoms out while following a general increasing trend.

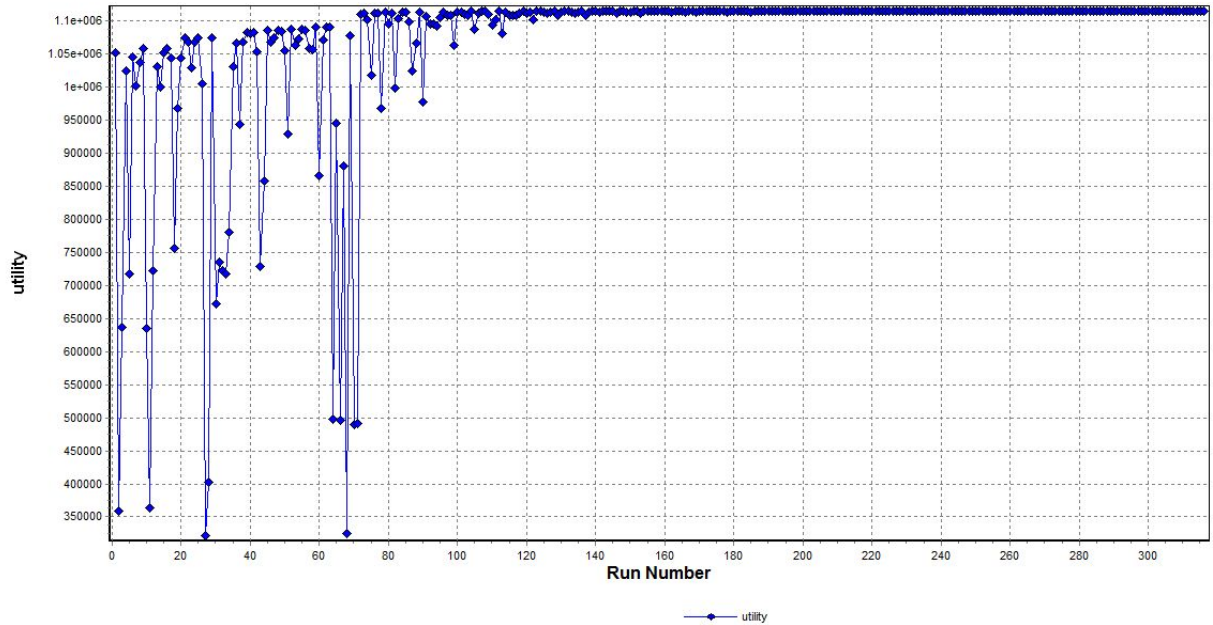


Figure 13: The utility vs. number of runs

The convergence history shows a monotonically increasing utility as the optimizer progresses through each design, confirming the upward trend in utility. It is interesting to note that the algorithm was set up with a small enough convergence criteria to pass through a local maximum.

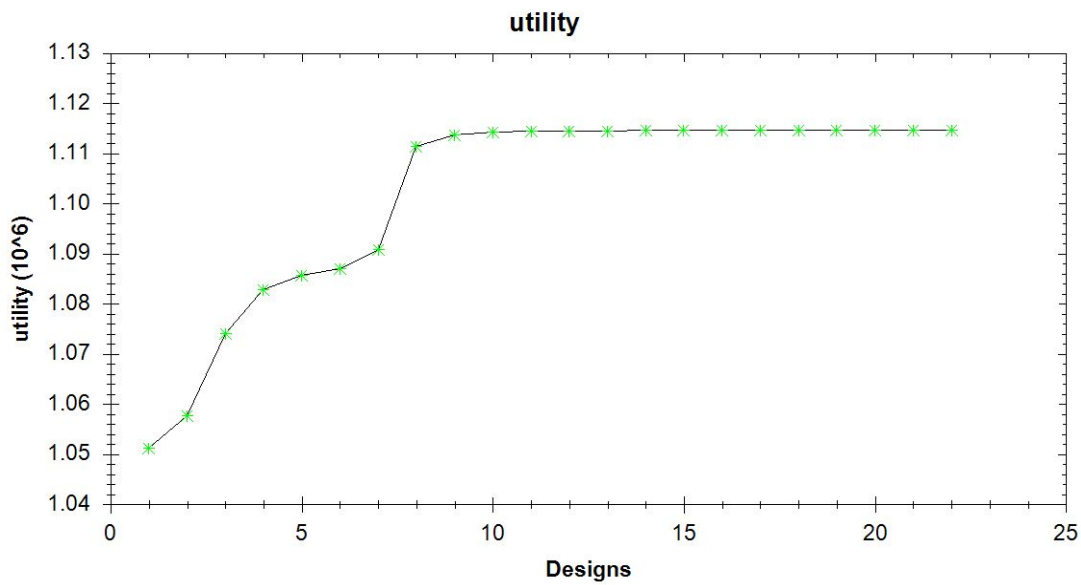


Figure 14: The utility as a function of designs

Next, the same simulation setup (with regards to start values and bounds) was run with a gradient-based algorithm - specifically, BFGS. The simulation setup/parameters are shown in the figure below:

Algorithm Options	
Optimization Parameters	
ConstraintThreshold	-0.03
ConstraintViolationThreshold	0.003
Scaling	0
Optimization Parameters for first iteration	
AbsoluteObjectiveDelta	0
MaxAbsoluteVariableDelta	0
MaxRelativeVariableDelta	0.01
RelativeObjectiveDelta	0.1
Optimization Parameters for Gradients	
Gradients	Forward
MaxGradientConstraints	0
MinFiniteDifferenceStep	0.0001
NearActiveConstraints	False
RelativeFiniteDifferenceStep	0.001
Stopping Criteria	
AbsoluteConvergenceTolerance	0
ConvergenceIterations	2
MaxEvaluations	1000
MaxIterations	100
RelativeConvergenceTolerance	0.001

Figure 15: The simulation setup (BFGS)

The best design as determined by the algorithm differs from the starting values by 12.3%, 0.35%, .044%, and 0% for radius, stall torque, no load speed and battery capacity, respectively - much closer to the starting values than the design alternative determined best by the Hooke-Jeeves algorithm. The returned values for the best design are shown in the figure below.

Best Design Run Number 33		
Objective(s)		
Name	Value	
PitchingMachine_IntegratedCostDemand.Demand_Calculator.utility	1.09207e+006	
Design Variable(s)		
Name	Start Value	Value
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.radius	0.41236	0.46317
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.tau_stall	26.1338	26.04
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.omega_max	321.483	321.341
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.batteryCapacity	2599.97	2599.97

Figure 16: The best design (BFGS)

In addition to BFGS being closer to the starting values on a percentage basis, the algorithm also converged with roughly 10% of the required runs as compared to the Hooke-Jeeves-based optimization (34 as compared to 316).. The run history and convergence history are shown in the figures below.

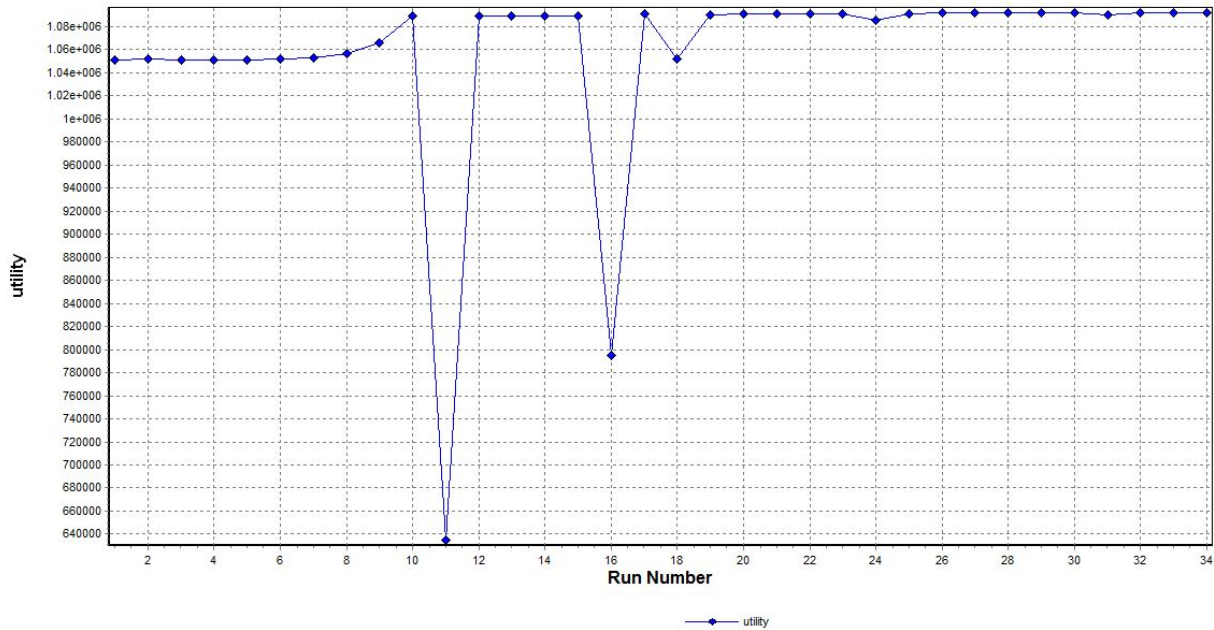


Figure 17: The utility vs. number of runs

The run history displays behavior consistent with BFGS/gradient-based approaches with a largely increasing utility as the runs progress. Run 11 is an interesting result as it is such a steep drop in utility that is presented by the gradient in run 10 as being ‘uphill’. A similar result though less extreme is returned in run 16. These runs indicate roughness in the utility surface; consulting the table shows that these runs are unique in that they have a higher radius value than the points surround them (for example, a value of 0.53535 for run 11 as compared to values of .45934 and .4598 for runs 10 and 12, respectively). This behavior aligns with the utility surfaces returned by the new DOE; the surfaces show a maximum at a radius value of ~0.45, dropping off relatively steeply in either direction from this point.

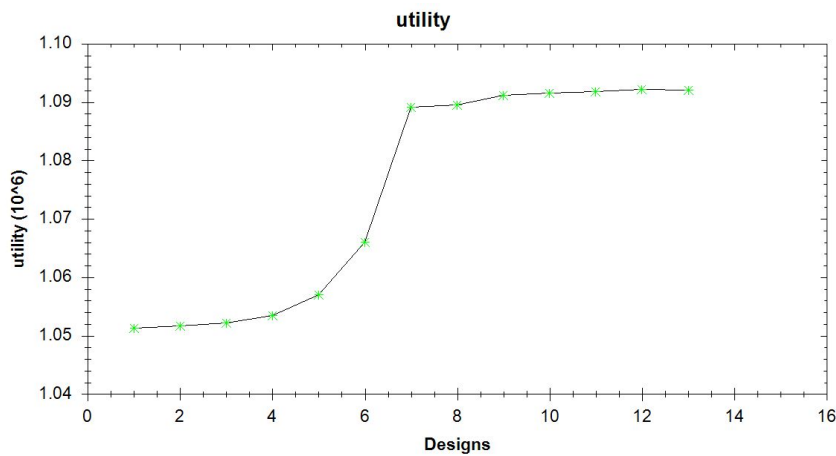


Figure 18: The utility as a function of designs

The convergence history shows the fast ascension in utility allowed by the gradient-based BSGF. Again it is noted that the algorithm moved past a local maximum (right at the beginning of the simulation).

Based on the absolute values for utility returned by these two algorithms, it seems that there are multiple maxima on the utility surface with comparable utilities; from the same starting point, two different optimization algorithms have yielded different design variables representing optimality. Taking into account the operating principle of each algorithm, it seems the result returned by BFGS is a 'steeper hill' with a slightly lower peak than that returned by the Hooke-Jeeves algorithm.

In the interest of curiosity and as a third measurement between two contrasting results, a pattern search algorithm was lastly used with the same starting values and bounds for the design variables. The intent here is to utilize a global search algorithm in comparing the so far contrasting results between gradient and non-gradient based algorithms. The simulation setup is shown in the figure below.

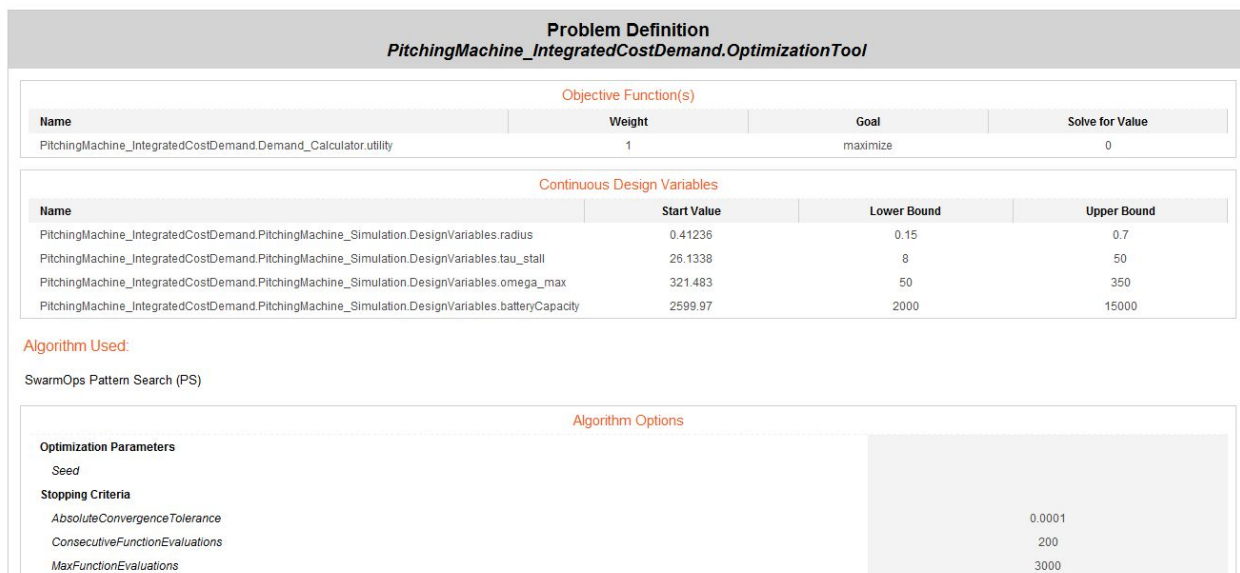


Figure 19: The optimization setup (pattern search algorithm)

The pattern search took a bit longer to run than Hooke-Jeeves with 419 runs (unsurprising given the nature of the algorithm). The design returned by the algorithm as best is calculated to be a lower utility than both the Hooke-Jeeves and BFGS approach, and with values for the design variables that differ from the starting value by 0%, 20.89%, 23.53%, and 22.44% for radius, stall torque, no load speed and battery capacity, respectively. The results are displayed in the figure below.

Best Design Run Number 312		
Objective(s)		
Name	Value	
PitchingMachine_IntegratedCostDemand.Demand_Calculator.utility	1.0896e+006	
Design Variable(s)		
Name	Start Value	Value
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.radius	0.41236	0.41236
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.tau_stall	26.1338	20.6723
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.omega_max	321.483	245.848
PitchingMachine_IntegratedCostDemand.PitchingMachine_Simulation.DesignVariables.batteryCapacity	2599.97	2016.5

Figure 20: The best design (pattern search algorithm)

This is interesting as a theoretically global search yielded a utility that is less than two other algorithms. It seems that there are multiple maxima contained within the utility surface, for which the maxima are close enough in absolute value to avoid convergence to the same maximum by multiple algorithms. The run history and convergence history of the pattern search algorithm are shown below:

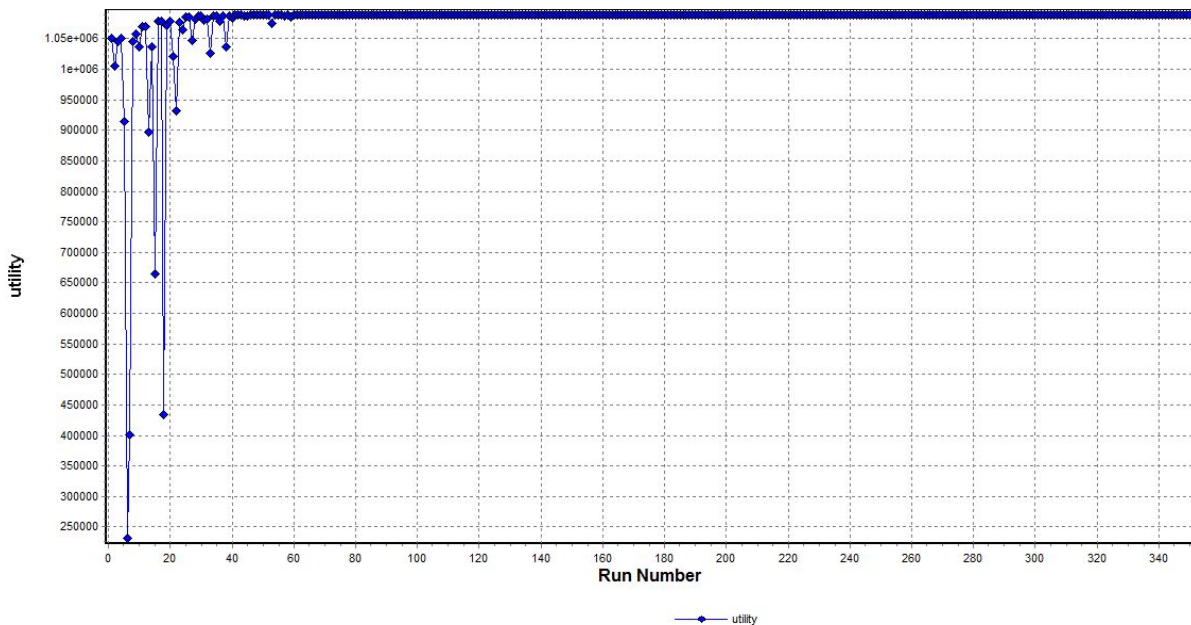


Figure 21: The utility as a function of runs

The run history displays behavior expected from the pattern search; the beginning of the simulation shows relatively scattered utility values as the algorithm ‘jumps’ around. Once it hits a maxima at ~run 40, it eventually meets the convergence criteria. It’s interesting to note how many more runs near the maxima/how many more design iterations in the convergence history are required with the pattern search relative to Hooke-Jeeves and BFGS.

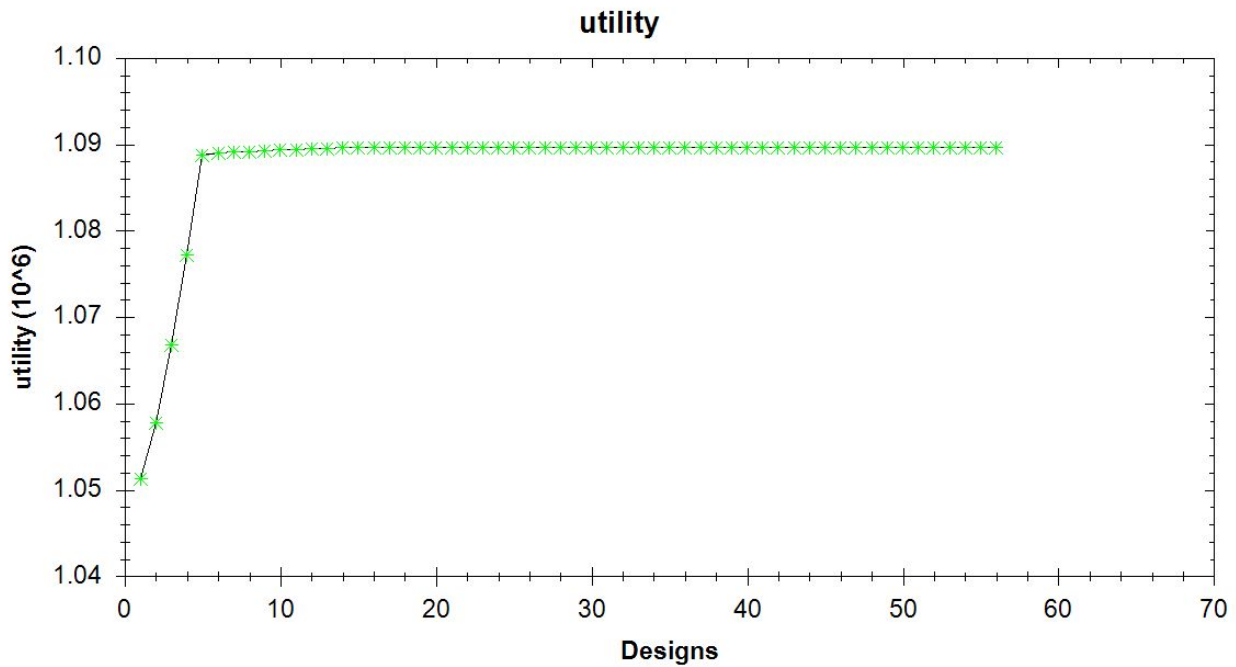


Figure 22: The utility as a function of designs

The results from these three approaches are summarized in the table below, and illustrate what are likely three comparative maxima on the utility surface characterized within the specified design space. The percent values listed represent the percentage difference from the starting value (consistent for each algorithm).

Table 1: Optimization Results

Algorithm	% Stall Torque	% No Load Speed	% Radius	% Battery Capacity	Utility
Hooke-Jeeves	35.78	49.99	6.25	22.46	1.115e+006
BFGS	0.35	0.44	12.3	0	1.092e+006
Swarm (Pattern)	20.89	23.53	0	22.44	1.0896e+006

Task 2: Solve the Design Problem under Uncertainty

Uncertainty was added to the ModelCenter model by adding an LHS driver loop on top of the variability LHS driver. This allowed for a probabilistic analysis of the variability and uncertainty, with the averages of each uncertain and variability variable taken as outputs to allow for

optimization. The model setup is shown in Figure 23. The setup of the optimization tool is shown in Figure 24. The ranges for the design variables in the optimizer are based on the DOE provided in Task 1. The ranges were set to capture the maximum peaks seen in the DOE, while eliminating what appear to be trivial optimums, such as the spike in utility for stall torques decreasing to zero.

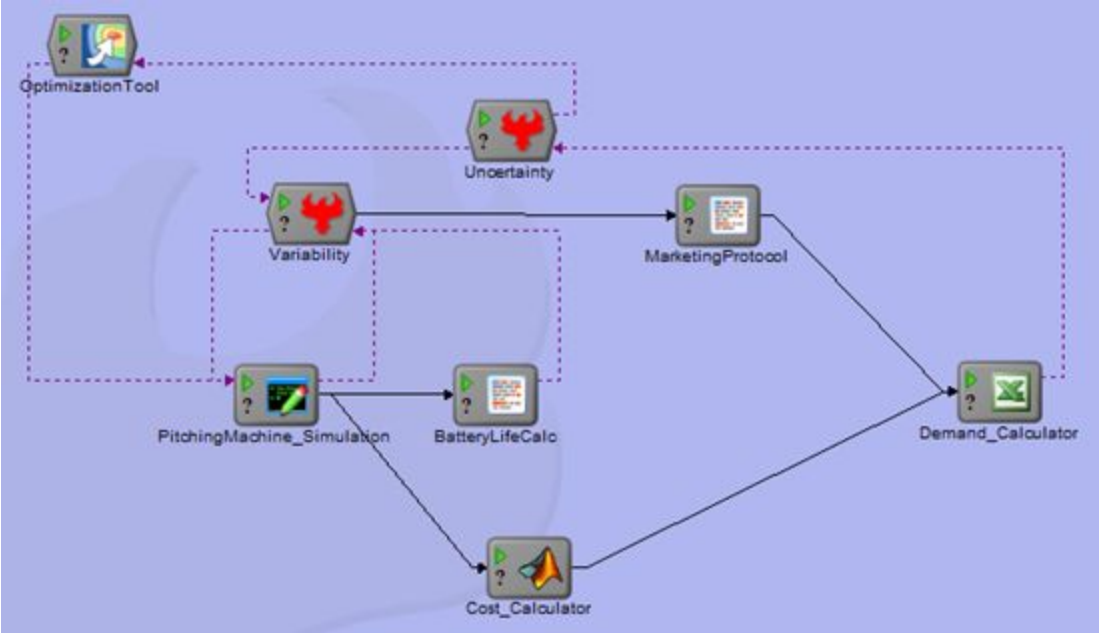


Figure 23: ModelCenter Setup for Optimization under uncertainty

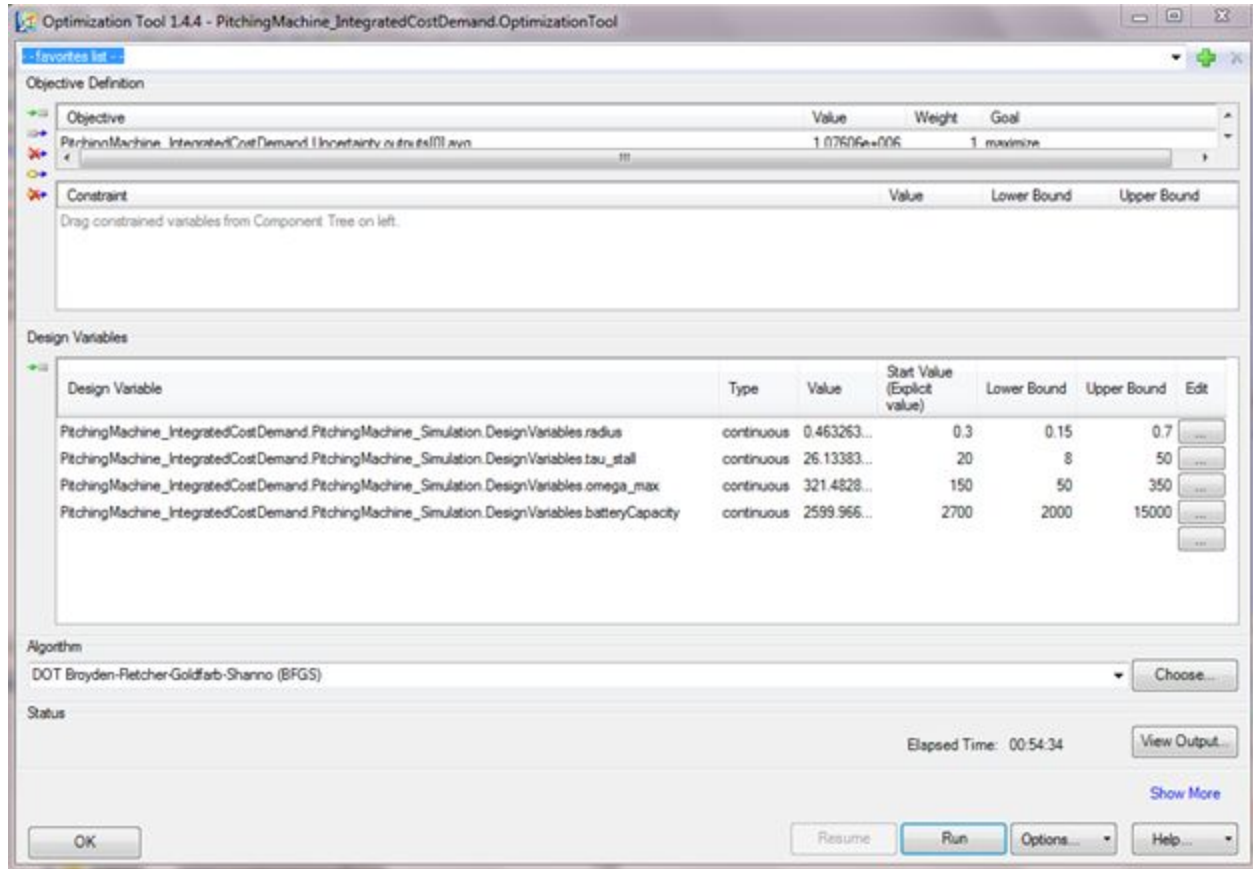


Figure 24: Setup of Optimization Tool

The optimization tool was run first with 5 LHS samples and then with 10. After each optimization, the final optimized design variable values were set as the start values for the next optimization. The results of these optimization studies are shown in Table 2.

Table 2: Optimization Results

Design Variable	5 LHS Samples	10 LHS Samples
Wheel Radius (m)	0.46326	0.58936
Stall Torque (N-m)	26.1338	25.851
No load speed (rpm)	321.483	321.029
Battery Capacity (mA-hrs)	2599.97	2599.98
Utility	1.07606×10^6	1.16827×10^6

It is interesting to note that the first optimization found almost the exact design variable values as the DOE results in Task 1, and after increasing the number of LHS samples, only the wheel radius changed significantly. The stall torque, no load speed, and battery capacity all remained effectively the same, while the wheel radius increased.

The final optimized design has much larger wheel radius than expected. This is likely a result of the error in the model's calculation of startup time as discussed in Task 1. The model initializes startup time as zero, and then resets the start-up time to the simulation time when the wheel radial speed reaches approximate steady state; however, for large wheel sizes and smaller stall torques, the wheel never reaches steady state within the given Dymola simulation time. As a result, the model returns a startup time of zero seconds for large radii and smaller stall torques. Within the demand model for the pitching machine, smaller startup times are more desirable, and in addition, larger wheels are not significantly more expensive than smaller wheels. The result is that the wheel size determined by the optimizer is significantly larger than expected. Given more time, the model would be adjusted to take the total time of the simulation as startup time if the wheel never comes to steady state. This would give the proper penalty for the larger wheels.

While the no load speed and wheel radius appear extremely high, the maximum pitching speed for the machine is 69 mph with these design variable values. This appears low given the no-load speed and the large wheel radius; however, based on the geometry of the ball-wheel interaction and the resulting friction, the ball and wheel never reaches the no-slip condition. The result is that the ball speed is still reasonable for a pitching machine.

The battery capacity results in a total battery life of only about 0.1 hours. This is much lower than would be expected for an ideal pitching machine. This extremely low optimum could be the result of the cost model inhibiting large battery sizes and customer demand not driving high operating times to be important enough. Additionally, the large wheel size could be impacting the results of this variable. The larger wheel takes significantly more energy to spin up than smaller wheels, so inherently a machine with the same battery size and a larger wheel will run for less time. However, based on the error in the model discussed above, the optimizer finds that large wheel sizes are desirable because of the zero startup time, and this likely drives the optimum design to revolve mostly around this wheel size rather than the other design variables.

Adding uncertainty to the optimization did not have a large effect on the results compared to the results in Task 1. The only design variable that changed significantly during the optimization process was the wheel radius, which increased. This shows that the uncertainty distributions we chose do not have a large effect on the overall design of the machine. This was somewhat seen in homework G4, where the uncertainty analysis showed larger spreads in results and some shifts in the averages of the distributions; however, there were no major shifts in results. The optimization results echo this. With further time and a more detailed analysis, the uncertainty analysis could be refined to see the effects on other uncertainty variables that were

not initially chosen. Additionally, different distributions for uncertainty variables could result in larger effects.

Task 3: Lessons learned

One of the major lessons learned throughout this project has been the importance of thoroughly verifying and qualifying the different parts of the model separately before adding in additional analyses. This became most clear in this portion of the project with the startup time error. The optimum found in this project is clearly not the actual optimum design for a pitching machine, which primarily results from the startup time error. This problem could have been mitigated earlier on in the project if the full design space had been more thoroughly vetted. However, the exact combination of the large wheel radius and the small stall torque had not been vetted early enough in the process to catch the error.

In this project we also learned the importance of vetting the design space before beginning an optimization. The DOE tool enabled us to decrease the ranges on the design variables before beginning the optimizations, which likely saved computational time overall. This is a methodology that can easily be extended to other branches of model building and design decisions, where parametric studies of different variables are not uncommon in the start of a design. These types of analyses are often computationally expensive and time intensive, and exploring how each of the variables separately affects the design is a good way to gain understanding of the overall system and also potentially catch errors before starting larger analyses.