Parameter Optimization in PEM Fuel Cells: A Genetic Algorithm Approach Versus A Point-Based Heuristic Comparison

Chris Henson
5/10/17

Created as a Final Project for:
CS 767 Machine Learning
Boston University

I. Problem

PEM fuel cells are a promising technology that can use air or hydrogen as its fuel to create a completely pollution-free source of electricity. Although research has gone into this field to improve the technology, the problem of modeling the fuel cell stack is computationally complex and is an impediment to optimizing the output of these cells. The goal of the two approaches outlined below is to attempt to optimize seven of the parameters within the given PEM fuel cell model, and to compare their outcomes both to the optimized model fuel cell as well as to each other. The approach and the results are outlined below.

II. Approach

The two approaches to this problem developed in this paper are a population-based genetic algorithm and a point-based heuristic. The genetic algorithm uses stochastic selection, a stochastic crossover function, with random members of the population generated to fill any gaps left by this process. Finally, a mutation function is employed to complete the algorithm.

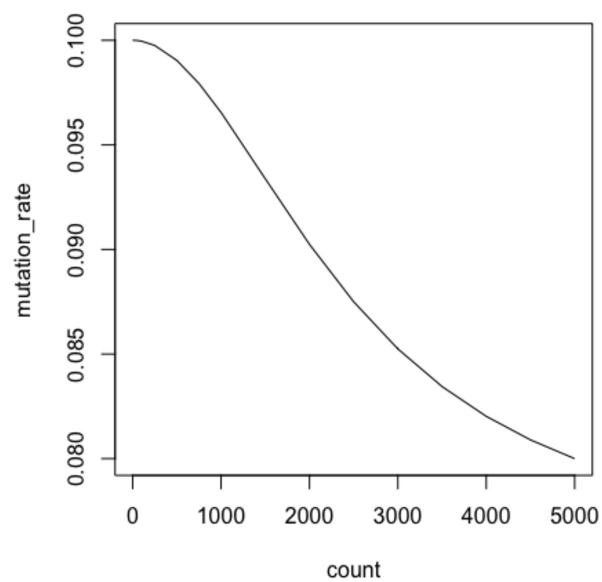
The point-based heuristic uses a mutation-based generation to improve on the solution, but creates an evaluation-based probability to choose a higher cost option to prevent being locked into a local optimum. The probability of choosing a higher cost option decreases as the number of iterations increases, creating a "cooling" effect that is more likely to choose an option based on its better characteristics later in the iterative process. Both approaches produced outcomes that show promise, and the results are outlined below. For this process, the target fitness was set at 0.10, a measure of the sum squared error (SSE) between the voltage output of the generated parameters as compared to the model output.

The Genetic Algorithm Approach

The source code for the genetic algorithm approach is attached at the end of this paper. It uses a population of 100 individual candidates of the seven parameters, and were generated initially randomly within the given specifications for each. These were then used to create a polarization curve based on a range of currents (i, in amps). This output was compared to the given model polarization curve based on the optimal parameters given, and assigned a fitness score based on the SSE between the two outputs. The goal was to optimize the parameters to match as closely as possible the given parameters of the model PEM.

The algorithm starts by using a stochastic process for the selection, creating a cutoff between 1% and 90% of the population after sorting by the best fit. The best option for that iteration will always survive the selection process, with the survival of the rest based at random. The crossover process with the random generation will refill the population numbers. The parents used for the crossover are selected at random, and the generated child is created using a randomly generated crossover point for the parents. These children are then placed in the population. Any remaining population numbers needed will be filled by randomly generating a new member within the range of each parameter.   A stochastic process determines how many of the population needed to replenish the population come from crossover, and how many from random generation.

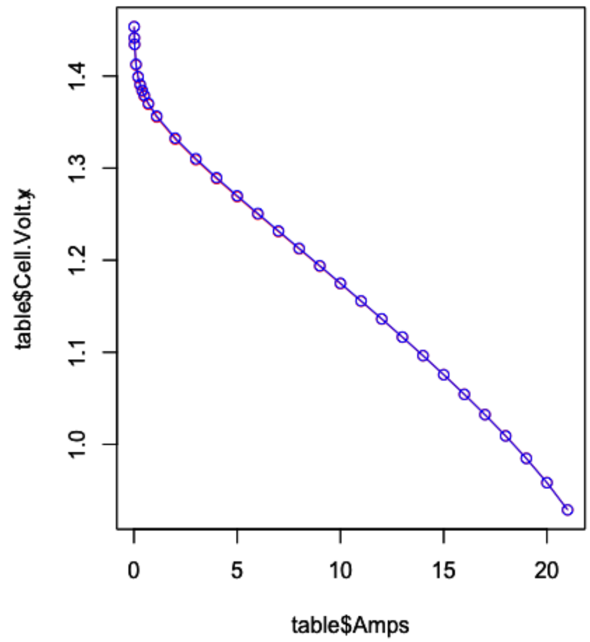
Several mutation functions were then applied to the population, using a component-wise mutation probability. The mutation rate of 0.10, 0.075, and several variants of a reverse-sigmoid function with a rate of 0.10 going down to around 0.05, 0.06, and 0.8 respectively were evaluated for this process. This creates its own "cooling process" with respect to mutation. The results for each shown and compared in the results data. An example graph of the output of the mutation rate from the reverse sigmoid function is shown below:



The Point-Based Heuristic Approach

The point-based approach initially creates a fifty-member population of randomly generated values, each with the parameter specifications and range of its corresponding model specification. These are then applied to a mutation function, which uses a stochastic element to determine the incremental change to the original value. The fitness of the original group is then compared to the fitness of the mutated group, and an average fitness is determined. Both groups are sorted, and the best fitness of all the population is then chosen as the initial point to move forward. This initial point is then mutated based on the same algorithm used previously, and the fitness of this function is compared to the original. As stated previously, if the mutated fitness is deemed worse than the previous, it may still be chosen based on a heuristic that uses a probability based on the number of evaluations already generated, and whether the fitness difference is comparable to the average fitness difference measured from the population. The full heuristic is shown in the attached source code.

In both approaches, a single global best is tracked and stored to determine the best value generated during the iterative process. Once the target function is achieved, or the maximum allowed iterations is reached, the parameters and fitness of the global best is returned. The graph in the results section shows that every evaluation of both approaches produced a graphical model that very closely represented the model polarization curve:



III. Results

The summary of the results collected are shown in the table below:

Algorithm	Parameters	Best Fitness After 500,000 Evaluations				No. of Eval. To Target Fitness (0.10)		
GA		Mean	Std Dev	Best Found	Fail	Mean	Std. Dev	Fail
	Rev. Sig. mut--0.1 to 0.6	0.03768004	0.02348721	0.00622023	0	1138.83	939.975425	0
	Rev.Sig mut--0.1 to 0.8	0.03588645	0.02344675	0.00345171	0	1122.53333	915.530965	0
	Mut. = 0.075	0.03793192	0.02576404	0.00031007	1	1236.33333	1207.57567	1
	Mut. = 0.1	0.04819411	0.0323703	0.00567972	3	1473.86667	1555.84552	3
	Rev. Sig. mut--0.1 to 0.5	not evaluated				1732.00	1909.89395	6
Point-Based Heuristic	x=u{0,1}, alpha=0.999, eval frac=0.05*	0.03545485	0.03099033	0.00040095	1	1351.01	1549.87562	1

*Results shown are per 100 evaluations

The point-based results in the table have been standardized to a per 100 evaluations average and standard deviation in order to compare it equally with the genetic algorithm, which evaluates a population of 100 at a time. Also note that, due to the high failure rate and high average fitness, there was no 5000 iteration evaluation done to the reverse sigmoid mutation function that went from 0.1 to 0.5. Based on the results collected, the stochastic nature of the selection and crossover of the genetic algorithm paired with the “muted” reverse sigmoid function that moves from a mutation rate of 0.1 to one of 0.08 reach the target fitness on average better than the others, with a smaller standard deviation, and with no failures. The 0.1 to 0.06 reverse sigmoid does also does well comparatively, again with no failures. Assigning a mutation rate of 0.075 through the duration of the iterations also shows comparatively good results in terms of average mean to target fitness. This method, however, did produce one failure during the run. The point based approach comes in next in terms of mean time to target fitness, and also performs within the range of several of the genetic algorithms. It did, however, have one failure compared to none for the two top performing GA algorithms.

When the program was allowed to run to the maximum evaluations (5,000 for the genetic algorithms, and 500,000 for the point heuristic) the heuristic has the best average fitness compared to the other variations, and was able to find the absolute best fitness out of all of the iterations performed. It did, however, have a failure to reach even the target fitness rate over the entire iteration compared to no failures for the top two performing genetic algorithms. Both failures for the point based approach came from the same seed, suggesting there may be some corner cases or certain conditions that may create problems for this method. More research in this area will need to be performed. Without this failure to skew the data, the average iterations to find the target fitness for the point-based heuristic goes from 1351 down to 1225, and the average fitness found after 500,000 evaluations goes to 0.03318.

IV. Conclusion

Depending on how accurate this simulation should be in comparison with the model characteristics, either of these approaches provide accurate model simulations down to very small levels. The genetic algorithm with the stochastic selection, random single point crossover and random generation, with a reverse sigmoidal curve to control the mutation rate not only performed better in the average iterations to the target fitness, but also does very comparable with the point-based heuristic approach with respect to average fitness levels after a maximum evaluation period. These results also are reached without any failures, which was not the case with any other process evaluated in this paper. Optimizing the point-based approach was not the intent of this paper, but further research is suggested to attempt to attempt this for better performance and compare those results with those shown here. The source code for both techniques is included in this package, along with the raw data collected, and a summary table of all the raw data for each iteration. One note: if this will be replicated, using R as was done in this experiment would not be advised. The scripting language take much longer to process an iteration of this form versus a more traditional language such as C++, Java, or GO.