Application of Genetic Algorithms to Motor Parameter Determination

by

R. Nolan, P. Pillay, and T. Haque

Electrical Engineering Department, University of New Orleans,
Lake front, New Orleans, LA 70148

ABSTRACT

This paper applies genetic algorithms to the problem of induction motor parameter determination. Generally available manufacturers published data like starting torque, breakdown torque, full load torque, full load power factor etc, are used to determine the motor parameters for subsequent use in studying machine transients. Results from several versions of the genetic algorithm are presented as well as a comparison with the Newton-Raphson method.

1. INTRODUCTION

The industry is becoming increasingly concerned about the ability of motors to ride through power system disturbances such as voltage dips or short duration outages [1,2]. An improved ride-through capability improves the reliability of the plant, particularly in the process industry where the failure of a motor can result in considerable downtime. The calculation of reclosing transients after an autoreclose for example, requires knowledge of the motor's electrical and mechanical parameters, which are not always readily available. From the motors' nameplate, the readily available data from the manufacturer are starting torque, breakdown torque, full load torque, full load power factor, full load efficiency etc. It is desirable to be able to extract the motor parameters from such data, which is the purpose of this paper. The Newton-Raphson method has been previously used, but with convergence problems relating to the initial starting points and the requirement for iteration [3]. In this paper, two different techniques, the Newton-Raphson and the genetic algorithms [4], are used to extract the motor parameters from readily available performance data. Several different induction machines are tested and the results are compared.

2. NEWTON-RAPHSON OPTIMIZATION USING QUATTRO PRO

Quattro Pro uses the Newton-Raphson method to solve nonlinear equations that may encompass several variables and constraints. The equivalent circuit parameters of an induction machine, which include stator and rotor resistances, and stator, rotor, and magnetizing reactances, can be obtained from Quattro Pro using its Newton-Raphson based optimizer function. The Quattro Pro spreadsheet can be set up to include the torque and power factor equations, an initial estimate for each parameter, and relevant nameplate and performance data. The relevant performance data consists of full load, locked rotor, and breakdown torque values, full load power factor, full load slip, and supply voltage. Quattro Pro begins by using the Newton-Raphson optimizer to adjust each parameter and recalculate the spreadsheet. Based on the new results, the optimizer continues to make adjustments until a solution is reached that meets all of the requirements. The optimizers recommended solutions appear in the designated cells, but the solutions vary depending on the initial estimates of the equivalent circuit parameters. In general, the more realistic the starting values are, the closer the results are to the correct optimal solution.

The major drawback of the Newton-Raphson method is that its success depends on the selection of good initial estimates. Although the optimization process may only take a few minutes, a considerable amount of time and effort can be spent selecting the initial estimates which require familiarity with the particular machine size and parameters. Even when the initial solutions appear reasonable, the optimizer still may not converge to the correct solution.

3. GENETIC ALGORITHMS

3.1 Introduction

The genetic algorithm is another method which may be used to solve a system of nonlinear equations. The genetic algorithm uses objective functions based on some performance criterion to calculate an error. However, the genetic algorithm is based on natural selection using random numbers, and does not require a good initial estimate. That is, solutions to complex problems could evolve from poor initial estimates in a game of survival of the fittest. Genetic algorithms manipulate strings of binary digits, and measures each string's strength with a fitness value. The stronger strings advance, and mate with other strong strings to produce offspring. Eventually one string emerges as the best. One of the most important advantages of the genetic algorithm over the Newton-Raphson technique is that they are able to find the global minimum instead of a local minimum and that the initial estimate need not be close to the actual values.

Another advantage is that they do not require the use of the derivative of the function, which is not always easily obtainable or may not even exist for example when dealing with real measurements involving noisy data.

3.2 The main operators

The mechanics of the genetic algorithm are elementary, involving nothing more than copying strings, random number generation, and swapping partial strings [4]. A simple genetic algorithm that produces good results in many practical problems is composed of three operators:

1. Reproduction
2. Crossover
3. Mutation

Reproduction is a process in which individual strings are selected according to their fitness. The fitness is determined by calculating how well each string fits an objective function. Copying strings according to their fitness value implies that strings that fit the objective function well have a higher probability of contributing one or more offspring in the next generation. This process of reproduction is of course an artificial version of natural selection. Here the objective function is the final arbiter of the string-creature's life or death.

Stochastic sampling with replacement is the name given to a simple reproduction scheme. This scheme is based on placing the string probabilities on a weighted roulette wheel and spinning the wheel to select a string. The probabilities on the roulette wheel are determined by the string's fitness as a percentage of the total population fitness. The roulette wheel selection scheme utilizes random numbers to simulate a spin of the wheel. Once a string is selected by the reproduction operator, the string is copied into a mating pool and waits to be selected for further genetic operator action. The roulette wheel scheme does not guarantee that the fittest strings will be selected albeit their probability for selection is high. Therefore, this method may not produce the best results, especially for problems with small populations.

Crossover is a two step process that involves mating and swapping of partial strings. Each time the crossover operator takes action, two randomly selected strings from the mating pool are mated. Then, in the case of simple crossover, a position along one string is selected at random, and all binary digits following the position are swapped with the second string. The result is two entirely new strings that move on to the next generation. This can be more clearly understood by the following example in which string 1 and string 2 have already been chosen to mate as shown in figure 3.1.

```
                    crossover
                      point
String A    1 0 0 1 1|0 0 1
String B    1 0 1 0 1|1 1 0

String A'   1 0 0 1 1|1 1 0
String B'   1 0 1 0 1|0 0 1
```

Figure 3.1. The crossover operator

Mutation follows crossover and protects against the loss of useful genetic information (1's and 0's). The operator works by randomly selecting one string and one bit location, and changing that strings bit from a 1 to a 0 or vice versa as shown in figure 3.2. The probability for mutation to occur is usually very small, roughly one mutation per thousand bit transfers.

```
                          Bit selected
                          for mutation
                                |
String A     1 0 0 0 1 1 0
String A'    1 0 0 0 1 1 1
```

Figure 3.2. The mutation operator

The three genetic operators, reproduction, crossover, and mutation, provide an effective search technique using natural selection and random number generation. Advanced operators, such as, dominance, inversion, and segregation exist, but are generally not essential for good results to many problems. In some cases the advanced operators can degrade the performance of the genetic algorithm.

3.3 Implementation of the genetic algorithm

The genetic algorithm can be used to calculate the equivalent circuit parameters of an induction machine as shown in figure 3.3. The locked rotor, breakdown, and full load torque equations form a multiobjective optimization problem, where each equation is a function of three or more machine parameters. The three torque functions can be written as follows.

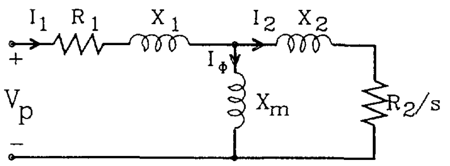

Figure 3.3. Induction Motor Equivalent Circuit

$$F1(R_1,R_2,X_1) = \frac{V^2\frac{R_2}{s}}{w_s[(R_1+\frac{R_2}{s})^2 + X_1^2]} - T_{fl} \qquad (3.1)$$

$$F2(R_1,R_2,X_1) = \frac{V^2 R_2}{w_s[(R_1+R_2)^2 + X_1^2]} - T_{lr} \qquad (3.2)$$

$$F3(R_1,X_1) = \frac{V^2}{2w_s[R_1 + \sqrt{R_1^2 + X_1^2}]} - T_{bd} \qquad (3.3)$$

where F1 is the error in the full load torque, F2 is the error in the locked rotor torque, F3 is the error in the breakdown torque, R2 is the rotor resistance, R1 is the stator resistance, X2 is the rotor reactance, X1 is the stator reactance, and Xm is the magnetizing reactance.

For simplicity, the stator and rotor leakage reactances are combined into one leakage reactance (Xl). The stator and rotor reactances can be extracted after the optimization by knowing the design class of the machine. The magnetizing reactance (Xm) can be calculated using the full load power factor equation after R1, R2, and Xl have been calculated, using the genetic algorithm.

Each parameter is coded as a 14 bit unsigned binary number, and together they form one 42 bit string as shown in figure 3.4. The maximum value each parameter can have, based on an accuracy of three decimal places, is 16.384 ohms.

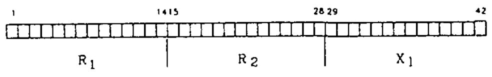

Figure 3.4. The genetic algorithm string

In this case, the error function is chosen as the sum of the squares of the torque error functions, while the fitness function is the inverse of the error. The aim of the genetic algorithm is to minimize the error or to maximize the fitness.

$$\varepsilon = F1(\cdot)^2 + F2(\cdot)^2 + F3(\cdot)^2 \qquad (3.4)$$

$$Fitness = \frac{1}{\varepsilon} \qquad (3.5)$$

4. RESULTS

4.1 Steady state parameter and torque results

Three induction motors (table 4.1) with known equivalent circuit parameters were used to test four different versions of the genetic algorithm.

Table 4.1. Actual Machine Parameters

Hp	Rpm	Volt	R_1	R_2	X_l	X_m
5	1750	230	0.434	0.303	2.511	24.61
50	1705	460	0.087	0.228	0.604	13.08
500	1773	2300	0.262	0.187	2.412	54.02

Each version uses the same population size, random number generator, string size, objective functions, and fitness function, yet each is slightly different in its approach. A description of each version follows.

V1: Version 1 uses stochastic sampling with replacement (weighted roulette wheel) as described earlier for reproduction. Simple crossover and mutation are also used, that is, one randomly selected crossover point and one bit change per thousand bit transfers for each string. This version is the simple genetic algorithm.

V2: Version 2 is identical to version 1 except that deterministic sampling is used instead of stochastic sampling for its reproduction scheme. The deterministic sampling scheme calculates the probabilities of selection as usual, the string fitness divided by the total fitness. Then each string is assigned an expected number based on its probability of selection. The actual number of times a string is copied into the mating pool is found from the integer part of the expected number. If additional strings are needed to fill the new population then the fractional parts of the expected number are sorted and the strings are selected from the top of the sorted list. This selection scheme has proved superior to straight roulette wheel selection since it guarantees that fit strings will be copied into the mating pool.

V3: Version 3 uses the deterministic sampling scheme with two-point crossover. The two-point crossover operator swaps all binary digits between two randomly selected points along the string.

V4: Version 4 uses the deterministic sampling selection scheme with a crossover operator for each parameter. This means that there is one randomly selected crossover point for each parameter or three crossover points along the entire string. This algorithm is superior to the other three since it produces consistently good results.

The results of each version of the genetic algorithm are given in the tables below. Comparisons can be made with table 1 which shows the actual equivalent circuit parameters for each induction machine. In addition, results from Quattro Pro's Newton-Raphson search routine is provided.

The results in using version 1 of the genetic algorithm are shown in table 4.2 for 3 different horsepower sizes. The estimated parameters are compared with the actual parameters and the errors calculated. Considerable errors are produced for some parameters, for example 85% error in R1 and even larger errors in R2. This version would be unacceptable. Table 4.3 has the parameter results using version 2. Although the parameter errors are not as large as for version 1, they are still considerable. Version 3 does not produce substantially better results than version 2 as shown in the parameter errors of table 4.4. Version 4 has the best results with acceptable errors in R2, Xl, and Xm as shown in table 4.5. Larger errors were produced in R1 for the small and large motors. However, table 4.6 shows that errors in R1 do not affect the torque calculations significantly. For example, while there were 15% and 24% errors in R1, for the 5 hp and 500 hp motors, the maximum error produced in the estimation of any torque is 2%. This is not unexpected since the error function was defined to minimize the torques, not the electrical parameters. Tables 4.7 and 4.8 show that acceptable results are obtainable using Newton-Raphson techniques provided good initial estimates of parameters are used. However table 4.9 shows that a slight change in the initial estimate of a parameter can cause the Newton-Raphson to converge to an entirely wrong solution as shown for the leakage and magnetizing reactances. The genetic algorithm is more robust in this regard.

Table 4.2. Results of genetic algorithm V1

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
R_1 (est)	0.063	0.083	0.666
R_1 (act)	0.434	0.087	0.262
% error	85.48	4.59	154.19
R_2 (est)	0.336	1.714	0.180
R_2 (act)	0.303	0.228	0.187
% error	10.89	651.75	3.74
X_l (est)	2.690	0.642	2.046
X_l (act)	2.511	0.604	2.412
% error	7.13	6.29	15.17
X_m (est)	28.44	71.77	44.31
X_m (act)	24.61	13.08	54.02
% error	15.56	448.70	17.97

Table 4.3. Results of genetic algorithm V2

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
R_1 (est)	0.707	0.245	0.334
R_1 (act)	0.434	0.087	0.262
% error	62.90	181.61	27.48
R_2 (est)	0.286	0.184	0.206
R_2 (act)	0.303	0.228	0.187
% error	5.61	19.29	10.16
X_l (est)	2.038	0.440	2.438
X_l (act)	2.511	0.604	2.412
% error	18.84	27.15	1.08
X_m (est)	20.39	9.39	55.18
X_m (act)	24.61	13.08	54.02
% error	17.15	28.21	2.15

Table 4.4. Results of genetic algorithm V3

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
R_1 (est)	0.194	0.305	0.297
R_1 (act)	0.434	0.087	0.262
% error	55.29	250.57	13.36
R_2 (est)	0.370	0.163	0.179
R_2 (act)	0.303	0.228	0.187
% error	22.11	28.51	4.27
X_l (est)	2.529	0.332	2.496
X_l (act)	2.511	0.604	2.412
% error	0.717	45.03	3.48
X_m (est)	29.29	7.81	55.05
X_m (act)	24.61	13.08	54.02
% error	19.02	40.29	1.91

Table 4.5. Results of genetic algorithm V4

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
R_1 (est)	0.367	0.087	0.325
R_1 (act)	0.434	0.087	0.262
% error	15.44	0.00	24.04
R_2 (est)	0.314	0.239	0.191
R_2 (act)	0.303	0.228	0.187
% error	3.63	4.82	2.14
X_1 (est)	2.386	0.641	2.470
X_1 (act)	2.511	0.604	2.412
% error	4.98	6.13	2.40
X_m (est)	25.39	13.73	54.72
X_m (act)	24.61	13.08	54.02
% error	3.17	4.97	1.29

Table 4.6. Torque errors for genetic algorithm V4

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
T_{fl} (est)	22.36	234.17	2023.23
T_{fl} (act)	22.35	234.55	1997.9
% error	0.045	0.162	1.267
T_{lr} (est)	14.31	519.10	841.85
T_{lr} (act)	14.3	529.708	835.57
% error	0.069	2.00	0.752
T_{bd} (est)	50.45	765.39	4982.38
T_{bd} (act)	50.14	773.987	5017.68
% error	0.618	1.111	0.704

Table 4.7. Results of Newton-Raphson search method

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
R_1 (est)	0.392	0.087	0.262
R_1 (act)	0.434	0.087	0.262
% error	9.67	0.00	0.00
R_2 (est)	0.313	0.238	0.195
R_2 (act)	0.303	0.228	0.187
% error	3.30	4.39	4.28
X_1 (est)	2.375	0.631	2.521
X_1 (act)	2.511	0.604	2.412
% error	5.42	4.47	4.52
X_m (est)	24.24	13.73	56.09
X_m (act)	24.61	13.08	54.02
% error	1.50	5.35	3.83

Table 4.8. Torque errors for Newton-Raphson method

	Motor 1 5 hp	Motor 2 50 hp	Motor 3 500 hp
T_{fl} (est)	22.33	235.16	2001.95
T_{fl} (act)	22.35	234.55	1997.9
% error	0.089	0.260	0.203
T_{lr} (est)	14.31	530.32	833.66
T_{lr} (act)	14.3	529.708	835.57
% error	0.069	0.116	0.229
T_{bd} (est)	50.13	775.27	5017.5
T_{bd} (act)	50.14	773.987	5017.68
% error	0.019	0.166	0.004

Table 4.9. Divergence of Newton-Raphson method

	Actual Machine Parameter	Case 1 Starting Parameter	Case 1 Final Parameter	Case 2 Starting Parameter	Case 2 Final Parameter
R_1	0.434	0.10	0.391	0.10	1.400
R_2	0.303	0.10	0.312	0.10	15.590
X_1	2.511	1.05	2.370	1.00	0.000
X_m	24.610	13.00	24.240	13.00	0.000

The performance of the error function when each version of the genetic algorithm is used is compared in figure 4.1. The results show that all versions eventually converge, thus producing low errors in the torques, but not necessarily low errors in the parameters. Version 4 however converges fastest with acceptable errors in the parameters as well as the torques. Figure 4.2 shows the convergence of machine parameters using the Newton-Raphson method. Two cases are used to test the convergence of the Newton-Raphson method. In case 1 the initial guess of the machine parameters was good, and the optimizer converged to the correct machine parameters. In case 2 the parameter xl was changed from 1.05 to 1.00 while all other parameters remained the same, and the optimizer failed to converge.

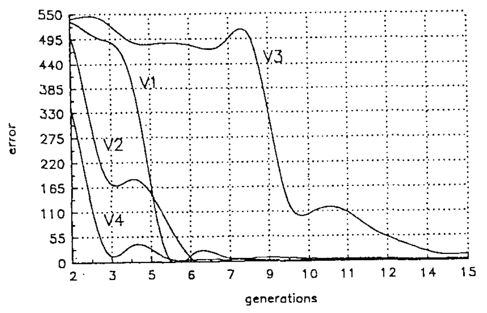

Figure 4.1. Convergence of parameters using genetic algorithms

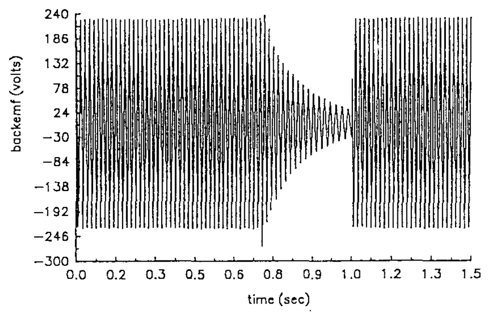

Figure 4.3. Backemf using actual parameters

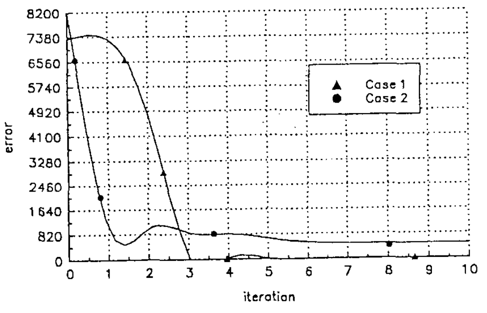

Figure 4.2. Convergence of parameters using Newton-Raphson

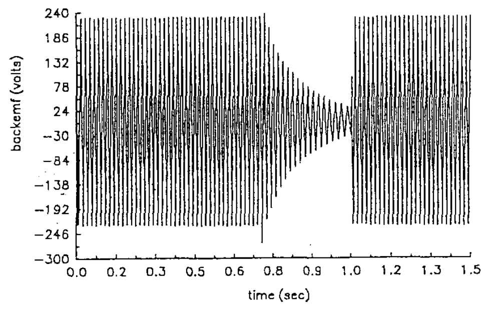

Figure 4.4. Backemf using genetic algorithm V4 parameters

4.2 Transient torque results

In order to demonstrate the accuracy during transient operation, the parameters generated by the genetic algorithm are used to predict the motor backemf waveforms for a 5 hp induction motor during a power outage. V4 of the genetic algorithm is used and compared against the results using the known original parameters as shown in figures 4.3 and 4.4. The waveforms were generated using EMTP [5,6], and it is clear that there is little difference between the waveforms.

While the backemf waveforms are virtually identical, the torque-speed curves are slightly dissimilar, mainly due to the differences in the rotor resistance. This is apparent in figures 4.5 and 4.6 for the genetic generated parameters and the actual parameters respectively.

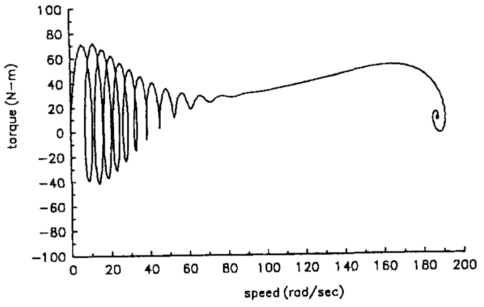

Figure 4.5. Torque-speed curve using actual parameters

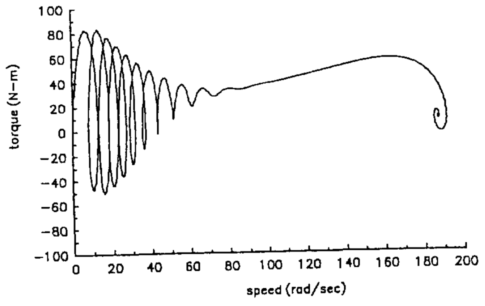

Figure 4.6. Torque-speed curve using genetic algorithm V4 parameters

4.3 Effect of population size and bit length on the algorithm performance

While 14 bits were used for each parameter and a population size of 250 strings was used in this paper, it is of interest to examine the effect of different bit lengths and population numbers on the ability of the algorithm to converge. The results are presented in figure 4.7 showing a bit length variation from 8 to 18 and the number of strings or population size from 100 to 500 for version V4. The number of generations is basically an indication of the time to converge. If the number of bits is less than 10 then poor results are obtained regardless of the population size. If the population size is less than 150 strings then poor results are obtained, regardless of the bit length. Also if the population is too large, greater than say 450 strings, the performance deteriorates regardless of bit size, indicating that perhaps other advanced operators like dominance and segregation may have to be used to procure reasonable results. The graph shows that bit sizes between 10 and 18 and population sizes between 150 to 400 would give consistently good results. A smaller bit size and population size has advantages in computer space requirements and processing time.

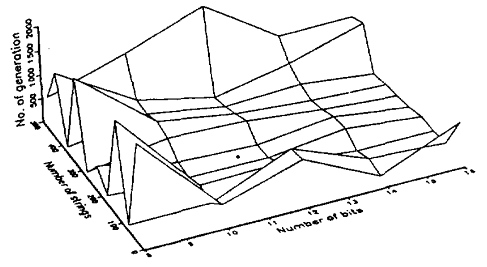

Figure 4.7. Effect of bit length and population size on generation number

4.4. Manufacturer performance data

Different manufacturers may calculate the performance data of a machine using a slightly different method. In addition, the designs of different manufacturers can lead to different errors from using the 5 parameter model used in this paper. Thus the performance of the genetic algorithm would differ for different manufacturers resulting in different errors in full load, locked rotor, and breakdown torque. This section examines these torque errors when using manufacturer's data of starting, full load, and breakdown torques. Thus no prior knowledge of the parameters was available. This section differs from section 4.1 in that there, known parameters were used to calculate the starting, full load, and breakdown torques using the induction motor equivalent circuit. Thus the equivalent circuit was assumed to be exact, with the same parameters being applied to starting as well as running conditions.

The performance data from seven different manufacturers was gathered from MotorMaster, a package developed by the Washington State Energy Office to collect motor manufacturer's data. This was used as input for genetic algorithm V4. The parameters for the simple equivalent circuit were calculated and the torques recalculated and the errors determined. Now the errors are larger as shown in figure 4.8 for a commercial 5 hp motor from seven different manufacturers, as a result of inaccuracies in the model (constant parameters with 5 variables). However, in spite of the simple model, manufacturers M1, M3, M4, M5, and M7 all produce acceptable results (around 10% error or less) while the torque calculations for manufacturers M2 and M6 have larger errors. This indicates that either a deep bar model or a 5 parameter model with variable parameters must be used to reduce the errors in predicting the torques for these two manufacturers.

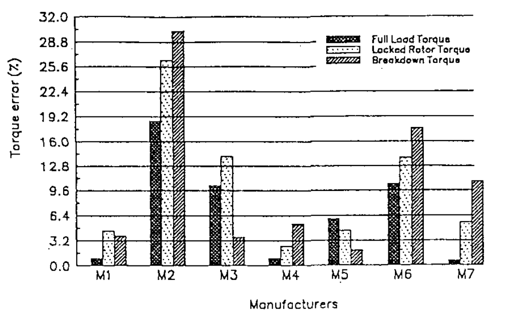

Figure 4.8. Genetic performance for 5 hp induction machine

5. CONCLUSION

This paper has applied the genetic algorithm to the problem of motor parameter determination. Several different versions were examined by calculating the parameters for a small (5 hp), medium (50 hp), and large (500 hp) induction motor. Version 4 produced extremely good results when the torques were generated from the equivalent circuit with known parameters. Larger errors were produced when using actual data from several manufacturers due to the neglect of parameter variations and deep bar effects in the model. The results were still acceptable for 5 of the 7 manufacturers. The use of the Newton-Raphson method was also demonstrated and its sensitivity to the initial starting values highlighted.

6. ACKNOWLEDGEMENT

The authors acknowledge EPRI, Entergy Services, Louisiana State and Mobil Oil for their support.

7. REFERENCES

[1] T.A Higgins, W.L. Snider, P.L. Young, and H.J. Holley, "Report on Bus Transfer: Part I - Assessment and Application", IEEE Transactions on Energy Conversion, vol. 5, no. 3, September 1990.

[2] Sarma S. Mulukutla and Edward M. Gulachenski, "A Critical Survey of Considerations in Maintaining Process Continuity during Voltage Dips while Protecting Motors with Reclosing and Bus-Transfer Practices", IEEE Transactions on Power Systems, vol. 7, no. 3, August 1992.

[3] B.K. Johnson and J.R. Willis, "Tailoring Induction Motor Analytical Models to Fit Known Motor Performance Characteristics and Satisfy Particular Study Needs", IEEE Transactions on Power Sytems, vol. 6, no. 3, August 1991.

[4] David E. Goldberg, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, Massachusetts, 1989.

[5] EMTP Revised Rule Book Version 2.0, EPRI EL-6421-L, Volume 1, Version 2.0, June 1989.

[6] G.J. Rogers and D. Shirmohammadi, "Induction Machine Modelling For Electromagnetic Transient Program", IEEE Transactions on Energy Conversion, vol. EC-2, no. 4, December 1987.