GENETIC ALGORITHM BASED DESIGN OPTIMISATION FOR PERMANENT MAGNET SYNCHRONOUS MOTORS

Gunaratnam Sooriyakumar†, Roy Perryman‡, Stephen J. Dodds‡
†Control Techniques Dynamics, Walworth Industrial Estate, Andover, Hampshire SP10 5AB
‡CITE, University of East London, United Kingdom
soori@ctdynamics.com ; r.perryman@uel.ac.uk; stephen.dodds@spacecon.co.

Abstract: This research work presents a new and efficient design methodology for the specification, development and manufacture of permanent magnet synchronous motors (PMSMs). In this paper a genetic algorithm based design optimisation technique for PMSMs is presented in which the multi-criteria considered in the optimisation are the electromagnetic performance, the thermal performance and the material cost. Models have been developed for each criterion in order to calculate the objective vector. A software tool called PMSM Analyser was developed to assist the motor design methodology. The optimisation algorithms and the electromagnetic, thermal and cost models were integrated and interfaced using this software. The programme is demonstrated for the design of a 12 slot 10 pole PMSM. The design parameter vector contains stator bore diameter, stator tooth thickness and stator back iron thickness. For the base design the outer diameter of the stator is 180mm and the stack length of the motor is 90mm. The base design refers to the design before optimisation and the optimal design refers to the design with optimised dimensions. The optimisation programme predicts significant improvements over the baseline design and experimental results are presented which indicate good agreement with the predictions of the programme. The new approach has been used successfully in the development and design of a PMSM with a stall torque of 125Nm, rated torque of 75Nm at 1500r/min and output power of 12kW. The strengths of the design methodology are summarised with the genetic algorithm optimisation, innovative multi-objective handling and design models for the various disciplines of PMSM development.

1. Introduction:

The genetic algorithm was developed and inspired by the natural selection of living beings which is a very successful organising principle for optimising individuals and populations of individuals. The genetic algorithm is not the only algorithm inspired by natural selection. The genetic algorithm was mainly developed by (Holland, 1975). Evolutionary strategies, were developed in Germany by (Rechenberg, 1973) and (Schwefel 1981). (Fogel, et. al., 1966) used evolutionary programming as a learning process aiming to generate artificial intelligence. These natural selection processes inspired methods or algorithms which are called evolutionary algorithms. If it is possible to mimic natural selection, then the optimisation task can be carried out more successfully. The design of a system using a selected design vector is analogous to an individual who is fighting for survival within a larger population. Only the fittest survives and fitness is assessed by the objective function value.

As stated previously, the gradient based methods suffer from inaccuracies in estimating the gradient and finding local optima only whereas the enumerative method is time consuming. Gradient based methods and enumerative methods are therefore not robust. The genetic algorithm on the other hand is a robust method (Holland, 1975) and it differs from the traditional methods as follows (Goldberg, 1989).

- GAs work with a coding of the parameter set, not the parameters themselves.
- GAs search from a population of points, not a single point.
- GAs use an objective function, not derivatives or other auxiliary knowledge.
- GAs use probabilistic transition rules, not deterministic rules.
- There can be multiple solutions for a given problem using GAs and alternative solutions can be selected for solving in a problem.

In view of these desirable features, the GA has been selected as the optimisation tool to support this research and is explained in detail in the following section.

2. A Detailed Description of the Genetic Algorithm:

2.1. Basic Principles:

Fig. 2.1 shows a flow chart of how the genetic algorithm works in a single population.
At the start, a population is randomly initialised and its fitness is calculated. The genetic algorithm uses three operations to create a new generation, sometimes referred to as offspring or children from the parents. They are reproduction, mating or crossover and mutation.
Based on the fitness of individuals they are selected for further operations. Copying the individuals, based on the objective values, into a new generation is called reproduction which is an artificial version of the natural election. The higher the objective value, there is a better reproduction. Selected individuals are then recombined or mated randomly, after this they are mutated. Their fitness is calculated and inserted into the population which produces a new generation. These activities are repeated until the stopping criterion is met.

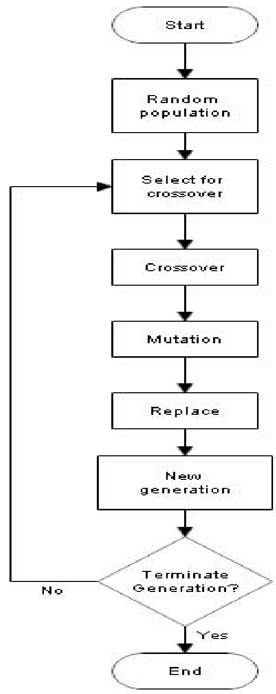

Fig. 2.1: Generic genetic optimisation flow chart

To obtain better results or for special problems, the multi-population evolutionary algorithm, as shown in Fig. 2.2, can be applied.

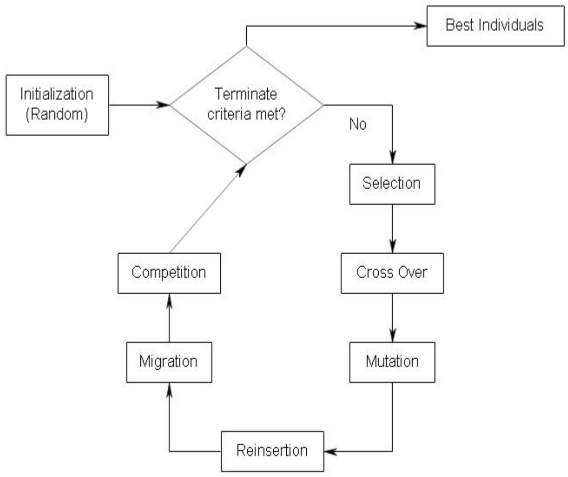

Fig. 2.2: Generic genetic optimisation flow chart - multi-population

2.2. Reproduction:

Which individuals and how many individuals have to be selected for recombination is determined by reproduction. At first individuals have to be assigned with fitness. The objective function value is not directly used by the GA but is converted to a fitness value which is a function of the objective value. The following methods can be used for fitness assignment.

- Proportional fitness assignment
- Rank-based fitness assignment
- Multi-objective ranking

Based on the fitness of individuals they can be selected for mating in the following ways:

- Roulette-wheel selection
- Stochastic universal sampling
- Local selection
- Truncation selection
- Tournament selection

Fitness assignment by scaling (proportional fitness assignment) Scaling can be linear or non-linear. Linear scaling is just the proportional value of the objective value. If the objective function value is negative, then it is first offset to make it positive and then it is scaled. In non-linear scaling a non-linear function of the objective function value is offset, if required, and then linearly scaled to achieve the fitness. For most of the optimisation problem fitness assignment by scaling is sufficient, but in this case, the optimisation sometimes suffers from stagnation and premature convergence which are not desirable. Stagnation occurs when the selective pressure is small. Selective pressure means the probability of the best individual being selected compared to the average probability of selection of all the individuals. Premature convergence occurs where reproduction has caused the search to narrow down too quickly (Matlab Documentation, 2005). These undesirable effects can be improved by sigma-scaling (Hancock, 1994) in which the required offset is calculated from the average and standard deviation of fitness values of the population.

A generalised multi-objective evolutionary optimisation process (Fonseca) is shown in Fig. 2.3.

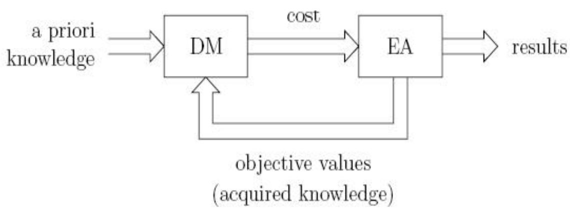

Fig. 2.3: A generalised multi-objective minimisation

The evolutionary algorithm (EA) produces new solutions based on the cost of the current solutions evaluated by a decision maker (DM).

Based on how the optimisation and decision making are combined, multi-objective minimisation can be categorised in three ways.

- Priori articulation of preferences: The decision maker combines all the objectives into a scalar cost function, so that the problem can be handled as if it was a single-objective optimisation.

• Posteriori articulation of preferences: A set of non-inferior solutions will be presented to the decision maker and a compromised solution will be selected by the decision maker.

• Progressive articulation of preference: At each step of the optimisation, partial preference information is supplied to the optimiser by the decision maker.

2.2.1. Weighted sum method: In this method objectives are multiplied by weights and added together to produce a single objective function.

The advantage of an objective function is that it can be controlled by its weight. The main problem is to determine the weights corresponding to each objective. The solution depends on the weights used.

2.2.2. Min-max method: In this method maximum difference between the objectives and their target values (optima or demand level) is minimised. This method can also be used in goal programming. In all aforementioned methods, the solution is a single point solution. In practical problems decision makers (DM) often need alternatives for decision making as some of the objectives conflict with each other. Also, if the objectives are noisy and discontinuous, these methods do not work very well and are very sensitive to the weights or the demand level. The methods require prior knowledge of each objective to decide the weights or demand values.

2.2.4. Multi-objective ranking: For ranking the individuals in the population, multi-objectives have to be compared between individuals, and based on the comparison the individuals have to be ranked. Then linear or non-linear ranking can be applied to assign fitness values to the individuals.

2.2.5. Pareto-ranking: In a minimisation problem, if x, y are two solution vectors, then if x and y are not dominating each other, they are called Pareto optimal solutions. The space formed by the objective vectors of Pareto optimal solutions which are non-dominant to each other is known as the Pareto optimal front. Any final design solution should preferably be a member of the Pareto optimal set.

In ranking, the Pareto-optimal solutions are normally regarded as equivalent and equal rank is given. The rank of an individual within the population r depends on the number of individuals Nd dominating this individual (Fonseca). Instead of giving equal ranking to Pareto-optimal solutions, they can be differentiated and ranked.

• Extreme cases can be ranked lower. For example a motor design which produces a lower cogging torque than others so that it cannot be dominated by other solutions but the cost is extremely high, has to be ranked lower.

• In optimisation problems there are some objectives which are more important than others. By taking account of this fact the Pareto-optimal solutions can be ranked.

• Some objectives need not be minimised or maximised. They need to satisfy only the minimum requirement. For example in a motor design, back e.m.f. does not need to be maximised. As long as it is above the required value, then the design is acceptable. So inequalities (constraints) can be set for some objectives and Pareto-optimal solutions can be ranked based on these inequalities.

In determining the Pareto-optimal solutions in the multi-objective optimisation, a normal evolutionary algorithm may converge at a single solution (premature convergence), and this is called genetic drift. It is important in these cases that special methods are used to maintain population diversity.

Fitness sharing (Fonseca), (Horn and Nafpliotis, 1993) can be used to overcome genetic drifting. In this method individuals which are closer to another individual are lowered in their fitness level.

There are several methods developed for searching non-dominated individuals in a population based multi-objective optimisation. (Scaffer, 1984, 1985) described a method called the Vector Evaluated Genetic Algorithm (VEGA) in which sub-populations were selected from the whole population according to objectives. After shuffling the sub-populations together, crossover and mutation was applied. As the population evolved, non-dominated individuals were identified. One of the problems reported in the VEGA method was called speciation (Fonseca). Speciation can be minimised by employing the Non-Dominated Sorting Algorithm (NSGA) method (Goldberg, 1989), (Fonseca), (Deb, 2001).

(Fourman, 1985) described a method called lexicographic ordering in which individuals are compared pair by pair. The objectives are assigned priorities. The most important objective is compared first. If the objective is similar for both individuals then the second most important objective is compared and so on.

2.2.6. Constraints handling: The optimisation constraints can be equality or inequality constraints. However, for the motor design, only inequality constraints occur and only these will be considered and discussed in this work.

GAs do not have explicit objective constraints, but constraints can be handled as follows:

• implicitly via the fitness function with a penalty for violation

• via the selection operator with rejection of constraint violators.

If the constraints are violated, then the solution is not feasible and it can be rejected by the selection operator. This is suitable where the constraints are rigorous (hard constraints). If the constraints are soft, then the fitness of the solutions can be degraded in relation to the degree of violation (Black, 1993). This method is called the penalty method.

(Powell and Skolnick, 1993) proposed a method in which the objective function was rescaled to less than one if it is feasible, and greater than one if it is not feasible, hence in the ranking and fitness assignment a feasible solution is allocated a higher fitness level.

2.2.7. Selection for recombination: After assigning fitness to individuals a partial set of the population can be selected for mating. One selection scheme called 'roulette wheel selection' can be thought of as follows: A wheel with an arrow indicator is segmented proportional to fitness or selection probability of individuals in the population. The wheel is rotated a number of times equal to the number of individuals that have to be selected. In each rotation the individual indicated by the arrow indicator will be selected. In the software implementation, a random number generator is used instead of a rotation of the wheel.

Another method called Stochastic Universal Sampling can be thought as follows: A roulette wheel as described above is used with more than one arrow indicator equal in number to the number of individuals that have to be selected. The indicators are equally spaced. The wheel is rotated only once. The individuals indicated by the arrows are selected.

Local selection (Voight, et. al., 1991), truncation selection (Blickle and Thiele, 1995) and tournament selection (Goldberg and Deb, 1991) are the other common selection methods.

2.3 Recombination:

In recombination two individuals are used to create a new individual by combining the characteristics (variables) of them. The recombination can be discrete recombination or real valued recombination. Binary recombination is also a discrete recombination.

2.4 Mutation:

The offspring are mutated after the crossover with specified probability. Mutation can be real or binary.

2.5. Re-insertion

Once offspring are created by reproduction, recombination and mutation, they have to be inserted into the current population to create the next generation. The number of offspring can be higher or equal or less than the population size. The insertion scheme depends on the selection scheme used. Local insertion and global insertion are the schemes used in the insertion corresponding to selection and global selection respectively.

3: Application of GA in Motor Design Optimisation:

The optimisation tab in PMSMAnalyser is shown in Figure 3.1.

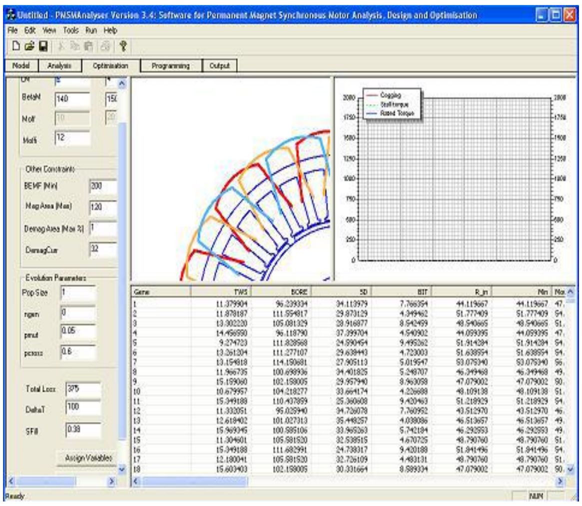

Fig. 3.1: Optimisation tab in PMSMAnalyser

In this PMSMAnalyser, C++ code the material cost/rated torque is defined as the main objective function. The minus sign is used to defined the optimisation as a minimisation of the objective function. Top part of the code defines the limits for some other objective functions. In this example they are cogging torque percentage and
demagnetisation area. The designs which exceeds these limits get low probability to be selected for next generation. The genetic algorithm (single population) parameters used to control the optimisation are as follows:

- Number of population in a generation
- Number of generations
- Cross over probability
- Mutation probability

These parameters can be entered through the PMSMAnalyser optimisation tab.

4: Preliminary Optimisation Results:

The genetic algorithm optimisation of the permanent magnet synchronous motors are demonstrated in the following sections.

4.1 Base design 1:

The design parameter vector contains stator bore diameter, stator tooth thickness and stator back iron thickness. For the base design the outer diameter of the stator is 180mm and the stack length of the motor is 90mm. The design has 12 slots and 10 poles. The base design refers to the design before optimisation and the optimal design refers to the design with optimised dimensions. The geometry and the winding configuration of the base design is shown in Figure 3.2.
The constraints of this design parameter vector are set according to the following lower and upper limits:

- Bore diameter = 95mm - 115mm
- Tooth thickness = 10.0mm - 16.0mm
- Back iron thickness = 4mm - 10mm

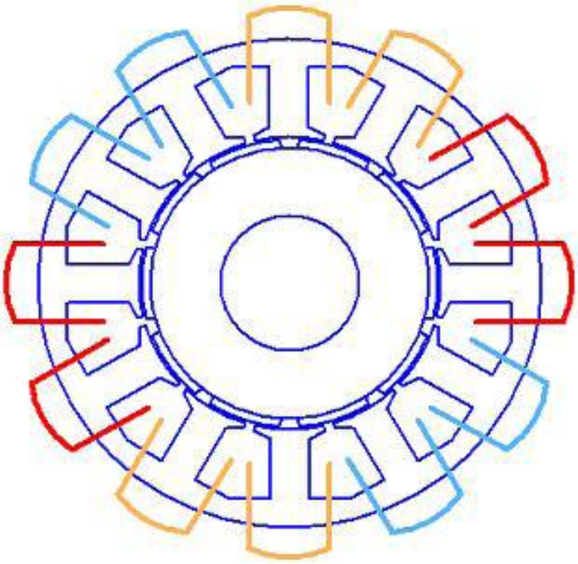

Fig. 3.2: Base design 1.

The following parameter values are fixed and are the same for the base design as well as the optimal design.

- Magnet span angle (electrical) = 1400
- Magnet thickness = 3mm
- Air gap = 1.0 mm
- Stack length = 90mm
- Stator outer diameter = 180mm
- Slot fill = 50%

The genetic algorithm parameters used are as follows:

- Number of generations = 30
- Number of genes in a population = 15
- Crossover probability = 0.6
- Mutation probability = 0.05

Fig. 3.3 gives stall torque optimisation results.

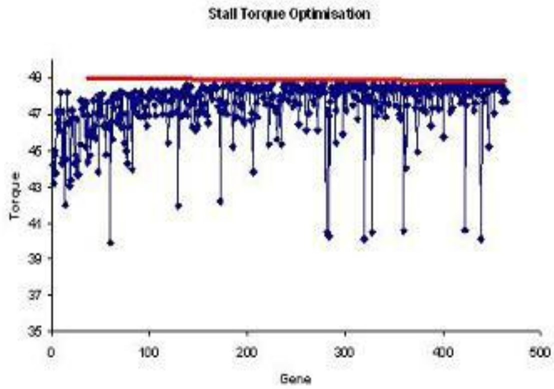

Fig. 3.3: Base design 1- Stall torque optimisation

This optimisation is named as BD1-OR1 (optimisation run 1). The objective function is the stall torque. It can be observed that the stall torque was improving during the optimisation from Figure 3.3. The optimal design is named as BD1-OD1. Figs. 3.4, 3.5, 3.6, 3.7 and 3.8 give the rated torque and active material cost optimisation results.

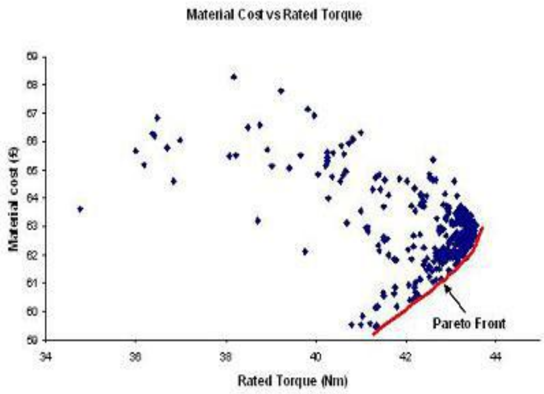

Fig. 3.4: Base design 1- Rated torque optimisation

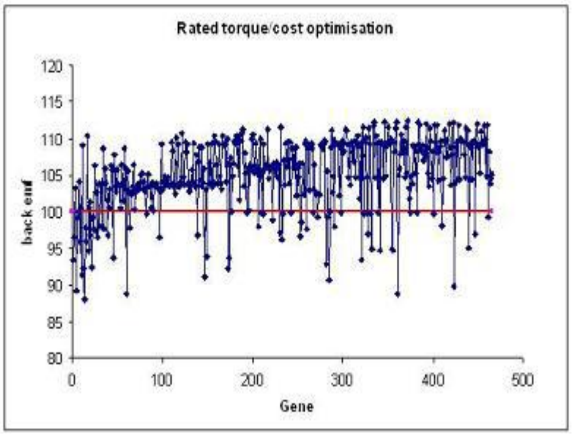

Fig. 3.5: Base design 1- Rated torque optimisation: back e.m.f.

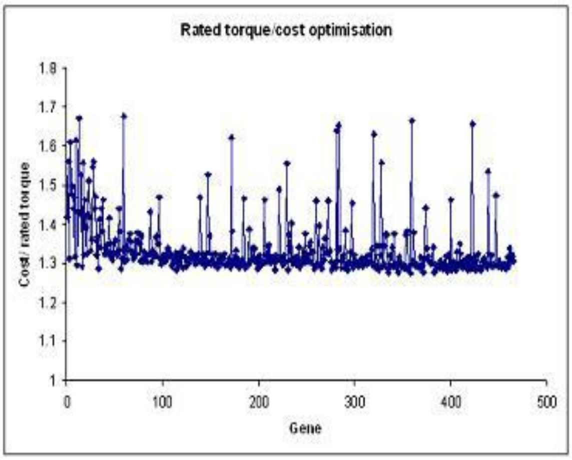

Fig. 3.6: Base design 1- Rated torque optimisation: cost/rated torque.

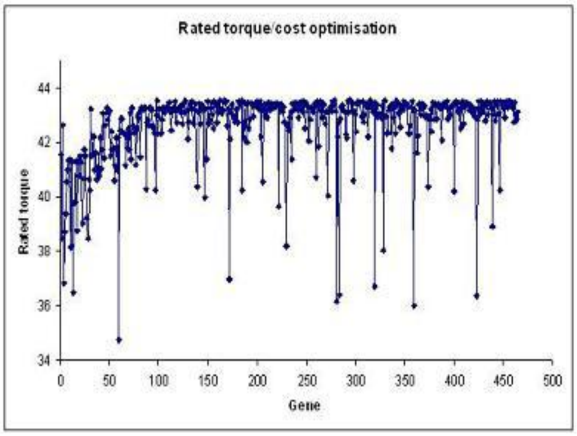

Fig. 3.7: Base design 1- Rated torque optimisation.

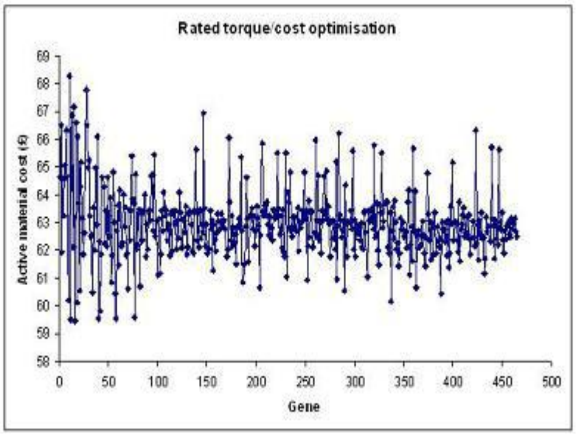

Fig. 3.8: Base design 1- Rated torque optimisation: active material cost.

The objective function is the ratio between active material cost and rated torque. This optimisation is named as BD1-OR2. The optimisation was defined as a maximisation problem by assigning a negative sign to the objective function. The optimal design is named as BD1-OD2. The rated torque is also improving during the optimisation as shown in Figure 3.7. But the active material cost does not show obvious improvement during the optimisation as shown in Figure 3.8. However the objective function of the ratio between active material cost and rated torque improves during the optimisation as shown in Figure 3.6 but this strengthens the fact that defining the objective function as a meaningful function of more than one objectives gives better optimal design than optimising the objectives individually. Also in the optimisation OD1-OR2, for the back e.m.f. and cogging torque, inequality constraints were used. The minimum back e.m.f. is defined as 100 V at 1000 r/min. In the optimisation, if the back e.m.f. drops below 100V, the objective function returns to a negative maximum of -1000 regardless of the ratio between active material cost and rated torque. If the back e.m.f. is above 100V, then the back e.m.f. does not have any effect in the objective function. Similar constraint can be set for the cogging torque with a maximum peak to peak cogging torque is 1% of the stall torque. Figure 3.5 shows that the back e.m.f. is increasing during the optimisation and more and more designs are producing back e.m.f. more than 100V during the optimisation. Figure 3.9 shows the cogging torque during the optimisation.

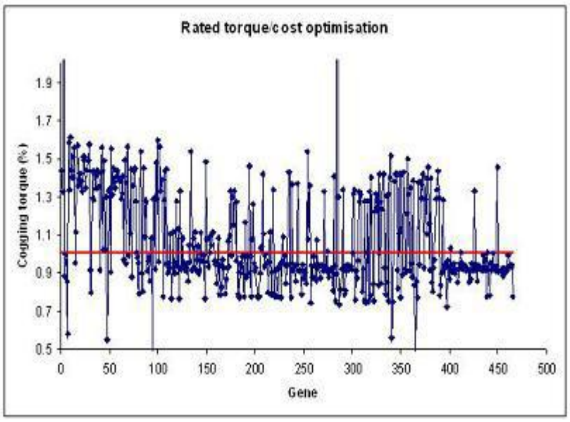

Figure 3.9: Base design 1- Rated torque optimisation: Cogging torque during optimisation

In Figure 3.4 it can be clearly identified that non-dominated solutions intensify along a line which is called the Pareto-optimal front. Table 3.1 compares the performance between the base design and the optimal design.

Quantity	Base Design	Optimal Design
TWS	14.0	12.9
BIT	8.0	7.1
Stator ID (Bore)	108	111.8
Rated Torque (Nm)	39.2	43.5
Active material cost	61.6	63.3
Cost/Torque	1.57	1.39
Back emf	101.1	109.3

Table 3.1: Comparison of base design 1 (BD1) and optimal design 2 (BD1-OD1)

The predicted and measured cogging torque are shown in Figure 3.10 and 3.11, respectively.

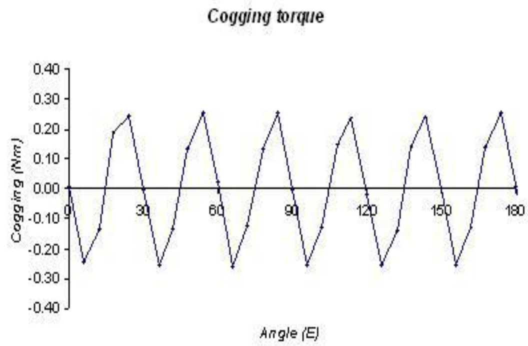

Fig. 3.10: Predicted cogging torque - BD1-OD2.

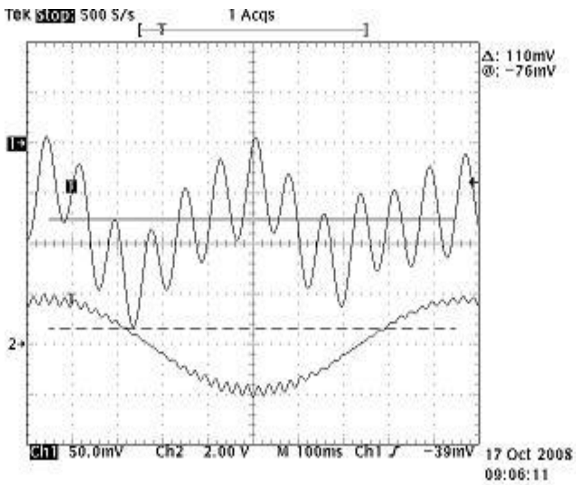

Fig. 3.11: Measured cogging torque - BD1-OD2.

The predicted and measured back e.m.f. are shown in Figs. 3.12 and 3.13 respectively.

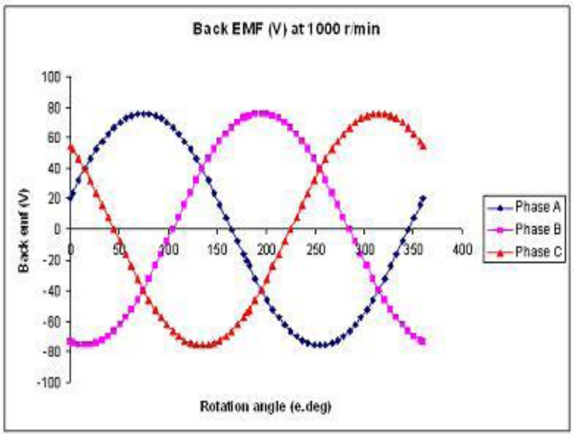

Figure 3.12: Predicted back emf - BD1-OD2

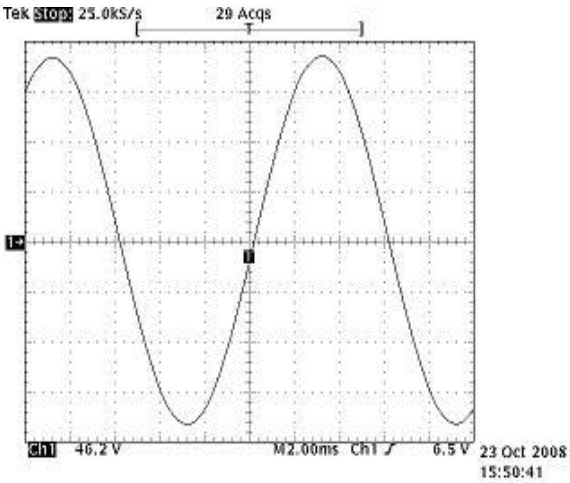

Figure 3.13: Measured back emf - BD1-OD2

5: Conclusion

The requirement for a multi-objective optimisation approach was analysed in terms of the design process for a PMSM. Traditional methods were investigated and considered in relation to their suitability for the application. This included gradient based methods, artificial neural networks and simulated annealing but they did not meet the necessary criteria. The genetic algorithm methods were considered to provide the optimum solution for this application and this technique has been adopted for the motor design process.

6. References:

Black, T., “Optimal Mutation Rates in Genetic Search”, ICGA5, 1993, 2–8

Blickle, T., and Thiele, L. A., *Comparison of Selection Schemes used In Genetic Algorithms*, TIK Report No. 11, 1995.

Deb, K., *Multi-objective Optimisation using Evolutionary Algorithms,* John Wiley & Sons, Chichester, 2001.

Fogel, L. J., Owens, A. J. and Walsh, M. J. *Artificial Intelligence through Simulated Evolution*, John Wiley, New York, 1966.

Fonseca, C. M., “Multiobjective Genetic Algorithms with Application to Control Engineering Problems”, *Ph.D. the is.*

Fourman, M. P., *Compaction of Symbolic Layout using Genetic Algorithms,* Grefenstette (1985), 141–153.

Goldberg, D. E., *Genetic Algorithms in Search, Optimization & Machine Learning*, Addison-Wesley, United States, Jan 1989.

Goldberg, D. E. and Deb, K., “A Comparative Analysis of Selection Schemes used in Genetic Algorithms”, FGA1, 1991, 69–93.

Hancock, P. J. B., *An Empirical Comparison of Selection Methods in Evolutionary Algorithms*, Fogarty, 1994, 80–94.

Holland, J. H., *Adaptation in Natural and Artificial Systems,* Ann Arbor: The University of Michigan Press, 1975.

Horn, J., and Nafpliotis, N., “Multi-objective Optimisation using the Niched Pareto Genetic Algorithm”, *IlliGAL Report 93005: Illinois Genetic Algorithms Laboratory, University of Illinios, 1993.*