

# **A finite element based optimisation tool for electrical machines**

Stiaan Gerber  
University of Stellenbosch

Centre for Renewable and Sustainable Energy Studies

## **Abstract**

This paper discusses the development of a finite element simulation package for electrical machines. This software is being used at the Electrical Machines Lab in the Department of Electrical and Electronic Engineering at the University of Stellenbosch where research in electrical machines for applications such as electric vehicles, wind generators, wave energy converters and Stirling engines is conducted. A couple of the main components of this software package are described and its capabilities are discussed.

The tool is designed to couple with a variety of optimisation tools which enables the design of electrical machines using finite element based numerical optimisation. Typically, these optimisation processes are computationally expensive. For this reason, the tool was designed to make use of parallel processors to speed the optimisation process.

The competency of the finite element package is illustrated by means of two case studies. These case studies show that the developed finite element package is capable of producing results that match those obtained using commercial finite element packages. One of these machines, a linear generator for a Stirling engine application, is used to illustrate how the tool can be used to optimise a machine.

Recommendations regarding possible future development of the package are made.

*Keywords:* electrical machines, finite element analysis, design optimisation

## **1 Introduction**

### **1.1 Modern electrical machine design challenges**

One of the greatest challenges of our time is to supply the ever growing population of the earth with enough energy. In modern times, the dominant form of energy used by humans has become electrical energy because it can be transported relatively easily and because it can readily be converted into other forms of energy that is useful to us such as heat or rotational energy. However, we as the human race are also growing ever more conscious of the fact that our way of life is having a negative impact on other forms of life and this has motivated us to consider, among many other things, the ways in which we generate electricity.

Electric generators are one of the critical components in the electricity generation process and as such the quality and cost of these machines have a significant impact on the performance and cost of our electrical systems. Today, the main challenges in designing electric generators are:

- designing generators suited to alternative forms of mechanical input power such as wind power, Stirling engines or wave energy converters

- pushing generators to be as efficient as possible
- designing generators that are inexpensive, thus lowering the cost and the impact associated with electricity generation or making renewable energy generation financially more viable

Electric vehicles, on the other hand, hold the promise of cleaner, more efficient transportation and although electrical machines are not necessarily the most prominent impediment to the widespread use of these vehicles, they remain a critical component with strict design constraints.

In order to meet the requirements on all types of electrical machines, the ability to accurately model their performance is of vital importance. The most popular technique currently available for the modelling of electrical machines is finite element analysis. This method allows many machine parameters, such as torque, power and efficiency to be calculated accurately and allows designs to be evaluated for many different criteria.

## 1.2 The finite element method

Here follows a discussion on the finite element method, as is applicable to the modelling of electrical machines considered in this paper.

In order to accurately calculate machine parameters, the magnetic field inside a machine must be solved for different positions of the moving component in the machine. The equations that govern this field are Maxwell's equations. In the magnetostatic case which is considered in this work, the problem of solving the magnetic field can be reduced to solving

$$\nabla \times (\nabla \times \mathbf{A}) = \mu \mathbf{J} \quad (1)$$

with  $\mathbf{A}$  the magnetic vector potential,  $\mathbf{J}$  the current density and  $\mu$  the permeability. For the two-dimensional cases considered in this work, the unknown vector function  $\mathbf{A}$  can be reduced to a single component dependent on two spatial variables, e.g.  $\mathbf{A}(x, y, z) = A_z(x, y)$ .

Using the finite element method the problem domain is broken up into small triangular elements on which the unknown function  $A_z(x, y)$  is approximated by

$$A_z(x, y) = \alpha_1 + \alpha_2 x + \alpha_3 y = N_1 u_1 + N_2 u_2 + N_3 u_3 \quad (2)$$

where the  $N_i$  are shape functions and the  $u_i$  are the (unknown) nodal values of the vector potential at the vertices of the triangle.

Eventually, the problem can be reduced to solving a system of linear equations for each triangular element,

$$\mathbf{K}^{le} \mathbf{u}^{le} = \mathbf{f}^{le} \quad (3)$$

where the superscripts designate the local element system. These can be merged into one global system of linear equations from which the unknown nodal values can be solved.

$$\mathbf{K} \mathbf{u} = \mathbf{f} \quad (4)$$

To assemble the global stiffness matrix,  $\mathbf{K}$ , refer to Fig. 1. To calculate a term,  $K_{ij}$  of the global stiffness matrix, terms from different local stiffness matrices may have to be summed. For example,  $K_{25}$  and  $f_2$  is calculated as

$$K_{25} = K_{13}^2 + K_{23}^1 \quad f_2 = f_2^1 + f_1^2 \quad (5)$$

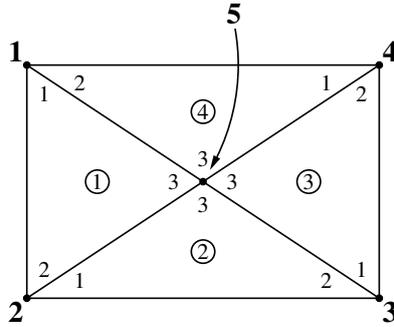


Figure 1: A portion of a finite element mesh. The large bold numbers indicate global node numbers, the small numbers indicate local node numbers and the circled numbers indicate element numbers

where the superscripts indicate local element systems.

### 1.3 Overview of this work

In the past, a custom finite element program has been used with good success by members of the Electrical Machines Group at the University of Stellenbosch. Because of the importance of finite element analysis in electrical machine design and the versatility offered by having the source code of a finite element program available, this program was deemed a significant asset. Unfortunately, there were also a couple of negative aspects to the implementation that required some attention. It was considered a worthwhile undertaking to sort out these issues and make some improvements that would allow the continued use of this program.

The goal of this work was to transform the original finite element program into a more powerful and usable tool, making it more competitive with commercially available finite element simulation packages for certain classes of problems. The classes of problems targeted in this work were two-dimensional magnetostatic problems with non-linear magnetic materials. These classes, illustrated in figures 2 to 4, can be defined as follows:

1. 2D rotating machines
2. 2D flat linear machines
3. 2D axisymmetric or tubular linear machines

The first two classes require the solution of (1) in Cartesian coordinates,

$$\frac{\partial^2 A_z}{\partial x^2} + \frac{\partial^2 A_z}{\partial y^2} = -\mu J_z \quad (6)$$

and the third class requires the solution of (1) in cylindrical coordinates

$$\frac{\partial^2 A_\phi}{\partial z^2} + \frac{\partial^2 A_\phi}{\partial r^2} + \frac{1}{r} \frac{\partial A_\phi}{\partial r} - \frac{1}{r^2} A_\phi = -\mu J_\phi \quad (7)$$

where  $\mu$  is a function of the magnetic field strength, defined by a single valued B-H curve, and the current densities  $J_z$  and  $J_\phi$  are prescribed.

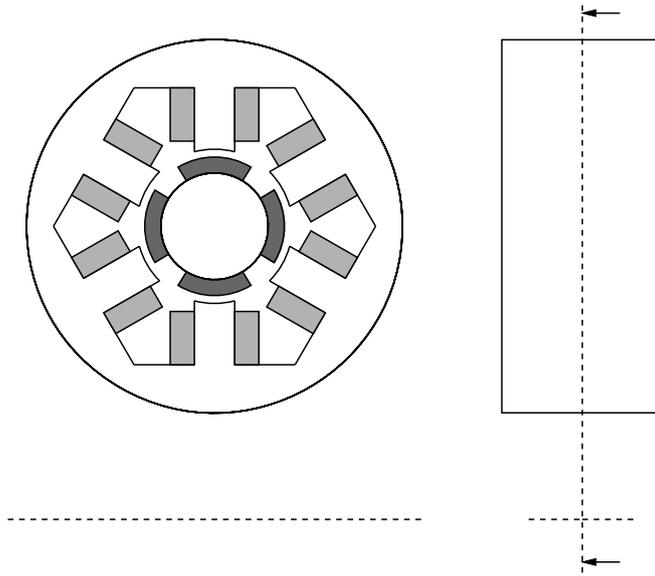


Figure 2: Problem class I: 2D rotating machines

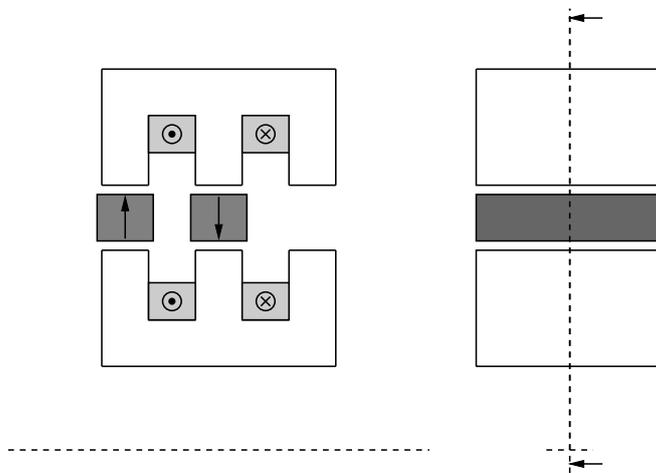


Figure 3: Problem class II: 2D flat linear machines

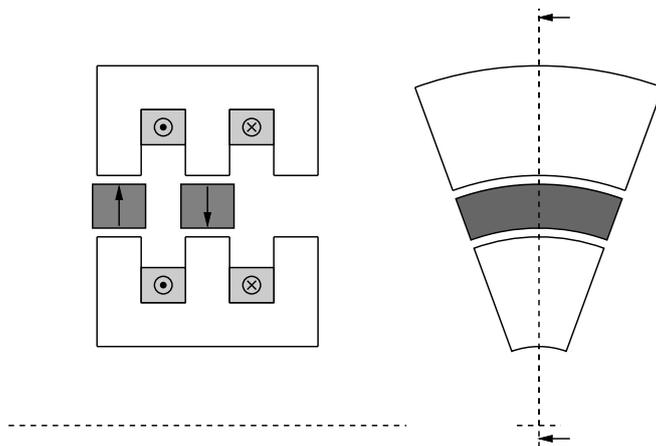


Figure 4: Problem class III: 2D axisymmetric linear machines

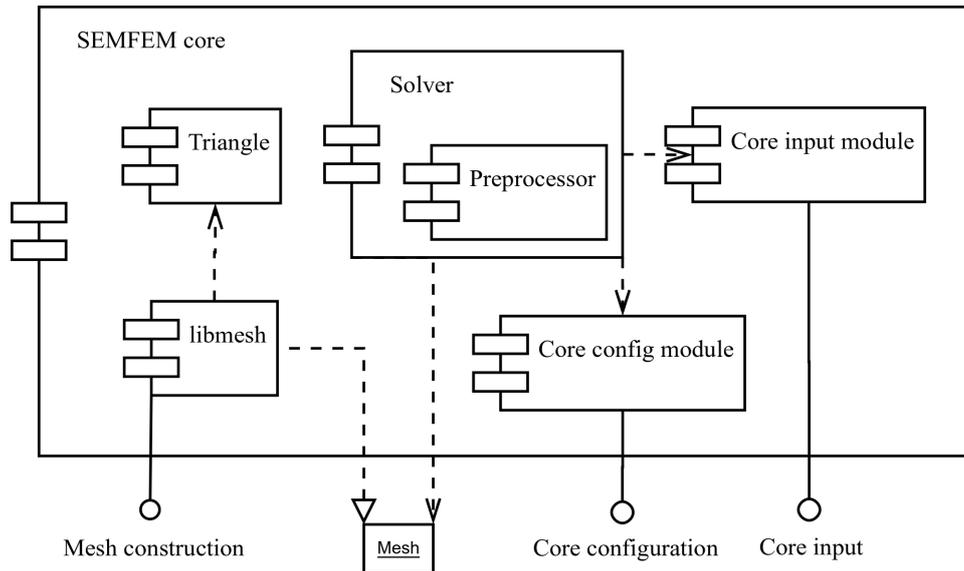


Figure 5: The inner structure of the core. Dashed arrowheads indicate dependencies, solid arrowheads indicate implementation.

The program, in its original form as received by the author, was capable of solving class I and II problems with the limitation that only a single air-gap could be modelled. The extension of the program for class III problems and multiple air-gaps is discussed in section 2.

Another requirement was that the developed finite element program should be easy to use within different optimisation environments. If a robust, tried and tested, environment for the optimisation of generators could be created, it would be a very powerful tool.

## 2 Components

In this section the main software components of the simulation package will be discussed briefly. Improvements and extensions of the original program will be highlighted.

### 2.1 Overview

For many reasons, it was necessary to completely restructure the original program to improve its performance and usability. Thus, the first part of this work was mainly concerned with the form of the program rather than the mathematics behind it. Originally, two separate programs existed for class I and II problems. These were merged and many other improvements were made. The resulting restructured package has been dubbed SEMFEM - Stellenbosch Electrical Machines Finite Element Method.

The main components of the restructured package and the interfaces provided to a user is illustrated in figure 5. The configuration module stores data on the required problem class, the amount of air-gaps and other configuration parameters. The input module stores data such as the amount of time-steps required, position values of the moving component, winding currents at each time-step, etc. The libmesh component provides an interface that allows the user to easily construct a mesh (see section 2.2). The preprocessor is responsible for calculating the stiffness matrices of the air-gap elements (see section 2.3), setting boundary conditions and setting up other data structures needed by the solver. The solver solves the system equation

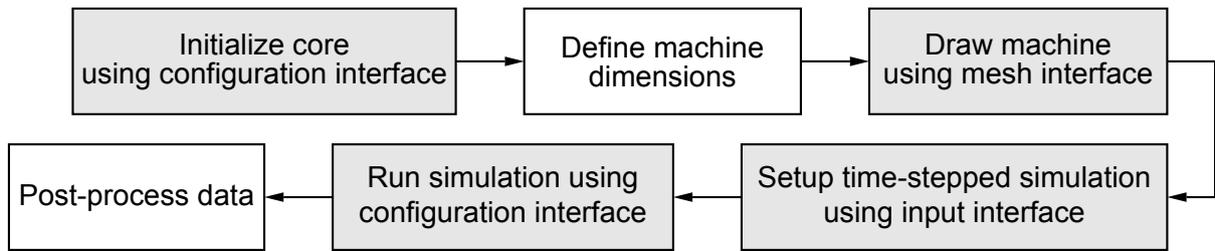


Figure 6: Flow diagram of a typical simulation

(4), taking into account the non-linearity of  $\mu$ , and calculates torque or force and flux linkages. With this data available the user can obtain all the required characteristics of the machine.

A user would typically use the interfaces as illustrated by the shaded blocks in figure 6.

## 2.2 Mesh generation

From the user's perspective, the most work in preparing a simulation often lies in constructing the finite element mesh. Thus, it is vitally important that the process of constructing a mesh is powerful, yet easy to use. The original program did not meet this requirement. Mesh construction was a tedious process and varying the coarseness of a mesh was almost as much work as constructing a new mesh from scratch. This is highly undesirable because the mesh coarseness has a great effect on the simulation accuracy, but also on computational time. A user typically wants to experiment with the coarseness of the mesh in various parts of the model in order to obtain an acceptable compromise between accuracy and speed and this requires a simple, powerful method of varying the coarseness.

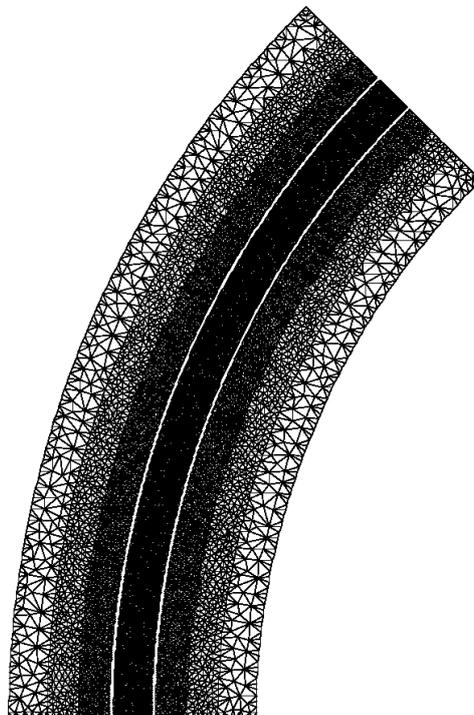


Figure 7: A finite element mesh generated with SEMFEM using *Triangle*

For these reasons, the original method of generating a mesh was discarded and a specialised open source mesh generator, *Triangle* (Shewchuk, 1996), was incorporated into the program. This is accomplished by the libmesh component shown in figure 5. This library, like *Triangle* itself, is implemented in C. It is approximately 900 lines long and contains all the functionality to convert data structures from the representation used by *Triangle* to the representation used by SEMFEM as well as the subroutines that users of the program need to construct a mesh.

The new method of mesh generation greatly simplifies the process from the user's perspective and is capable of varying the mesh coarseness in different parts of the model by simply setting a maximum element area for specific regions. An example of a mesh generated using this method is shown in figure 7.

### 2.3 Air-gap elements

The single most important distinction between the presented program and other packages used to simulate electrical machines in the present day is the way this program handles relative movement between different sections of the finite-element mesh. The method used in this program was first proposed by Abdel-Razek et al. (1982). It is unique because the air-gap is not meshed, but the field in a special air-gap element is calculated analytically. Thus, the accuracy of the field in the air-gap, a crucial part of the model, is only subject to the accuracy of the surrounding elements. No additional discretisation error is introduced in the air-gap. Another advantage of using this method is that the forces acting on the moving component can be calculated simply and accurately using Maxwell's tensor, as is described in (Abdel-Razek et al., 1981). A linear version of the air-gap element method using the Cartesian coordinate system was derived by Wang (2003). This air-gap element featured in the linear version of the original program.

Originally, the air-gap element method was computationally very expensive, but Flack and Volschenk greatly reduced the cost of the method using the technique discussed in (Flack and Volschenk, 1994). This technique was implemented in the original versions of the program and contributed to its good performance.

Air-gap elements also produce a local stiffness matrix of the form of (3), but where the local stiffness matrices of ordinary first order triangular elements are  $3 \times 3$  matrices, that of an air-gap element is much larger because it is connected to many more than 3 nodes.

A big limitation of the original version of the program was that it was only capable of analysing models with a single air-gap using the air-gap element method. This prohibited the use of the program in the analysis of many topologies. This limitation was removed during the course of this work.

The greatest challenge in adding multiple air-gap functionality to the program lay in understanding how air-gap elements are coupled to the rest of the finite element mesh and identifying the pieces of code that implement this coupling. Mathematically, the coupling of air-gap elements to the traditional mesh can be expressed as

$$K_{ij} = \sum K_{mn}^e + \sum_{k=1}^{N_G} \sum K_{mn}^{ek} \quad (8)$$

where the first term represents contributions from the traditional mesh and the second term represents the contributions from a total of  $N_G$  air-gaps. Once this was mastered, modifying

the code to allow multiple air-gaps was a relatively simple matter. This was done in the following manner. Firstly, additional variables were declared to provide storage for the additional air-gap element data. Secondly, the preprocessor was modified to setup more than one air-gap correctly. Thirdly, the solver was modified to take the contribution to the stiffness matrix coefficients from all air-gap elements into account. Lastly, the torque or force calculations were modified to take the contribution from both neighbouring air-gap elements into account.

## 2.4 Nonlinear solver

Referring to (4), note that the coefficients of the matrix  $\mathbf{K}$  are functions of  $\mu$  and  $\mu$  is a function of the solution variables  $\mathbf{u}$ . This implies that an iterative process must be used to converge to the correct solution of (4). The Newton-Raphson method is used in the presented program because of its fast rate of convergence. Using this method, the solution vector is updated at each iteration according to the following equation (Binns et al., 1992)

$$\mathbf{u}^{m+1} = \mathbf{u}^m - \mathbf{J}^{-1}[\mathbf{K}^m \mathbf{u}^m - \mathbf{f}] \quad (9)$$

where  $\mathbf{J}$  is the Jacobian matrix of the vector

$$\mathbf{F} = \mathbf{K}^m \mathbf{u}^m - \mathbf{f} \quad (10)$$

with respect to the vector  $\mathbf{u}$ .

## 2.5 Axisymmetric problems

The extension of the program to allow axisymmetric problems (class III) to be solved required the correct calculation of the coefficients of the system matrix equation. The contribution due to ordinary triangular elements as well as that of axisymmetric air-gap elements needed to be calculated. The derivation of the coefficients due to ordinary first order triangular elements was obtained from (Binns et al., 1992). These coefficients for the local element system are

$$K_{ij} = \int_{\Omega_e} \frac{1}{4A^2\mu} \left[ c_i c_j + \frac{1}{r^2} (a_i + 2b_i r + c_i z)(a_j + 2b_j r + c_j z) \right] r dr dz \quad (11)$$

$$f_i = \int_{\Omega_e} J_\phi \left( \frac{a_i + b_i r + c_i z}{2A} \right) r dr dz \quad (12)$$

which, unlike the coefficients for class I and II problems, is typically evaluated with a numerical integration scheme such as Gaussian quadrature. The constants  $a_i$ ,  $b_i$  and  $c_i$  only depend on the geometry of the triangle and  $A$  is the area of the triangle.

Axisymmetric air-gap elements have never been used before and so the coefficients of the system matrix for this element was derived by the author. For an axisymmetric air-gap element with inner radius  $a$ , outer radius  $b$  and length  $z_0$ , the coefficients are given by

$$K_{ij}^e = \frac{z_0(b^2 - a^2)}{2(c_i' - \frac{c_i^2}{c_i'})(c_j' - \frac{c_j^2}{c_j'})} \cdot a_{0i} a_{0j} + \frac{z_0}{2} \sum_{n=1}^{\infty} (a_{ni} a_{nj} + b_{ni} b_{nj}) \int_a^b \left[ \lambda_n^2 f_{2i}(r) f_{2j}(r) r dr + \frac{1}{r^2} f_{4i}(r) f_{4j}(r) \right] r dr \quad (13)$$

with the functions  $f_{2i}(r)$  and  $f_{4i}(r)$  defined as

$$f_{2i}(r) = \left( \frac{I_1(\lambda_n r) - \frac{I_1(\lambda_n c_i)}{K_1(\lambda_n c_i)} K_1(\lambda_n r)}{I_1(\lambda_n c'_i) - \frac{I_1(\lambda_n c_i)}{K_1(\lambda_n c_i)} K_1(\lambda_n c'_i)} \right) \quad (14)$$

$$f_{4i}(r) = r \lambda_n \left( \frac{I_0(\lambda_n r) + \frac{I_1(\lambda_n c_i)}{K_1(\lambda_n c_i)} K_0(\lambda_n r)}{I_1(\lambda_n c'_i) - \frac{I_1(\lambda_n c_i)}{K_1(\lambda_n c_i)} K_1(\lambda_n c'_i)} \right) \quad (15)$$

The values of the constants  $a_{0i}$ ,  $a_{ni}$ ,  $b_{ni}$ ,  $c_i$ ,  $c'_i$  and  $\lambda_n$  can be obtained from Wang (2003). The integral in (13) is evaluated numerically and the infinite series is truncated at some point.

An expression for the coefficients of the Jacobian matrix used in the Newton-Raphson iterations of the nonlinear solver is not reported in Binns et al. (1992). The author derived an expression for these coefficients following the process of a similar derivation in Binns et al. (1992). The coefficients are given by

$$J_{ij} = K_{ij} + \frac{1}{8A^4} \cdot \frac{\partial \kappa}{\partial (p^2)} \cdot \sum_{h=1}^3 \sum_{k=1}^3 \left( \int_{\Omega^e} t_{kj} t_{ih} r dr dz \right) u_k u_h \quad (16)$$

with

$$t_{ij} = \left[ c_i c_j + \frac{1}{r^2} (a_i + 2b_i r + c_i z)(a_j + 2b_j r + c_j z) \right] \quad (17)$$

In (16),  $\kappa$  is the reluctivity and  $p$  is the magnitude of the flux density. The integral is evaluated numerically using Gaussian quadrature.

### 3 Optimisation and parallelisation

Although it is possible to optimise machine designs based on analytical solutions of machine parameters, this method is usually less accurate and not as powerful as multivariable optimisation using finite element analysis. The latter process is, which is focused on in this work, is illustrated in figure 8. The idea is to define the optimisation problem as follows,

$$\text{Minimise : } F(\mathbf{X}) \quad (18)$$

$$\text{Subject to : } G_1(\mathbf{X}) > g_1 \quad (19)$$

$$G_2(\mathbf{X}) > g_2 \quad (20)$$

$$G_3(\mathbf{X}) > g_3 \quad (21)$$

$$G_4(\mathbf{X}) > g_4 \quad (22)$$

$$X_{li} \leq X_i \leq X_{ui} \quad (23)$$

where  $\mathbf{X}$  is the vector of design variables. Equation 18 is the objective function and equations 19 to 22 represent constraints which are also a function of  $\mathbf{X}$ . The search space is defined by lower and upper bounds on each design variable,  $X_{li}$  and  $X_{ui}$ , as in (23). Referring to figure 8, the finite element simulation is responsible for evaluating the objective function and constraints for a vector of design variables. The optimiser's task is to find the optimal choice of design variables. In order to do this, it is necessary to evaluate the objective function and the constraints many times.

Considering this process, it is clear why it is necessary to minimise the computational time of finite element simulations: A single optimisation process requires many simulations to be

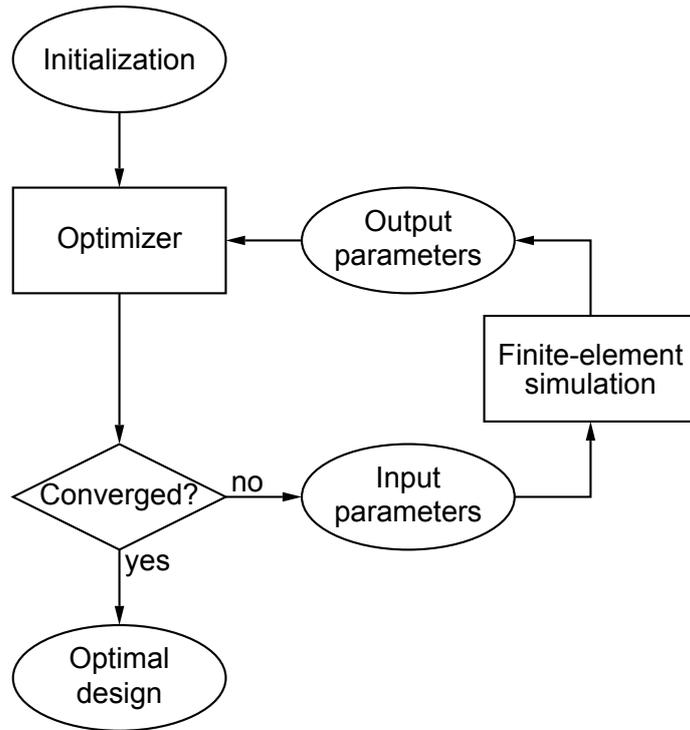


Figure 8: Design optimisation using finite element analysis

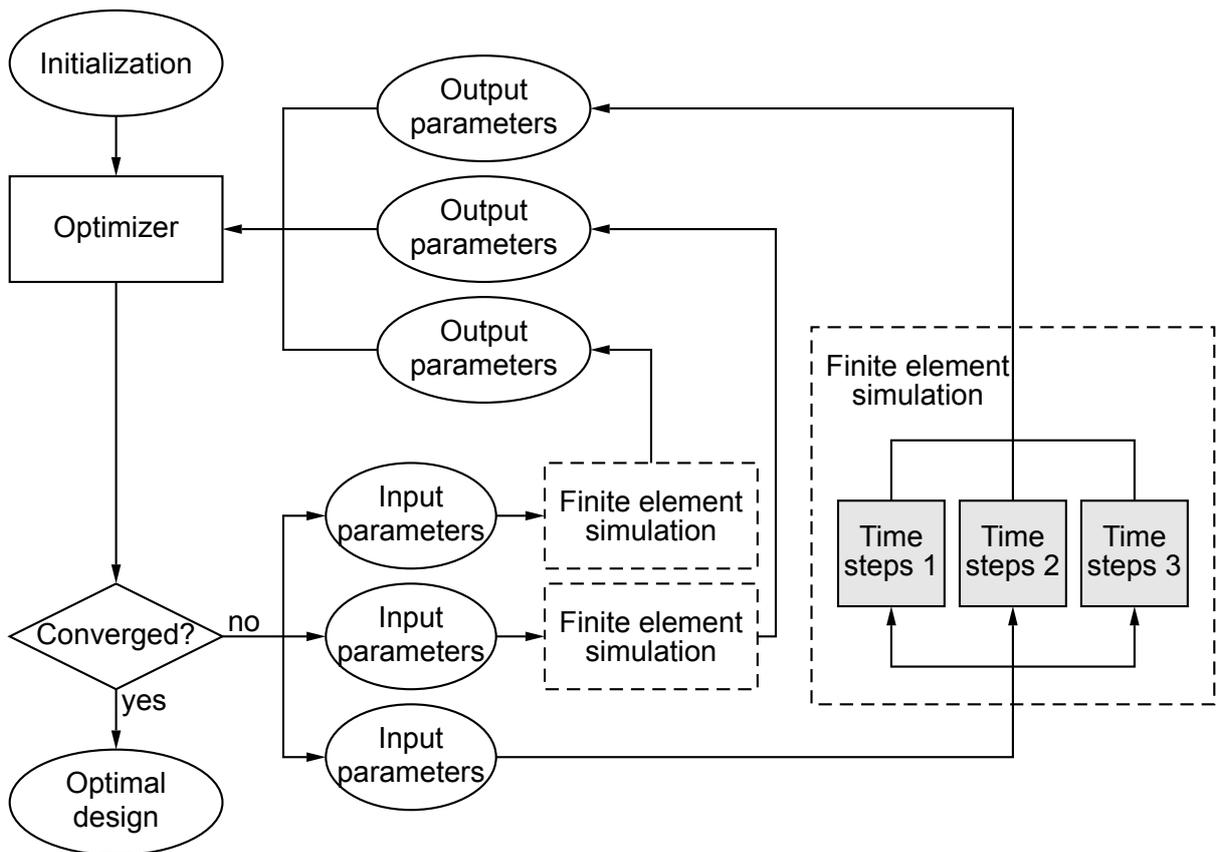


Figure 9: Parallel design optimisation using parallel finite element analysis

run. Depending on the complexity of the problem, this means that the effect of computational efficiency for a single optimisation is not measured in seconds or in minutes, but in hours or days.

In order to alleviate the problem of high computational cost, the optimisation process can make use of parallelisation in several areas. The parallelisation implemented in this work is illustrated in figure 9. Comparing figures 8 and 9, it can be seen that the serial process has been parallelised in two areas: the optimiser requests three simulations simultaneously and the time steps of a single simulation is divided and solved on three separate CPU cores, illustrated by the shaded blocks. Thus, in this example, a total of nine CPU cores can be used simultaneously.

At the time of writing, a suitable example of parallel optimisation was not yet available, although successful tests were carried out. The optimisation described in section 4.2 did not make use of any parallelisation.

## 4 Application

### 4.1 Case Study: Rotating Machine

The simulation of the double rotor, radial flux, air-cored, permanent magnet machine shown in figure 10 was presented as part of an investigation into the accuracy and performance of SEMFEM with support for machines with dual air-gaps (Gerber et al., 2010). This generator is intended for wind power applications. The investigation showed that SEMFEM can produce results that match those of commercial packages in comparable simulation times.

The machine was simulated for a single electrical period using SEMFEM and *Ansoft Maxwell 2D*. A single simulation consisted of 100 time-steps. Figure 11 shows a comparison of the torque and flux linkage waveforms obtained with SEMFEM and *Maxwell*. The slight difference in the offset of the torque waveforms is attributed to small inconsistencies in the models used in SEMFEM and *Maxwell*. Both simulation packages produced noisy torque ripples using less dense meshes. In the case of the flux linkage waveforms, the results are indistinguishable from each other.

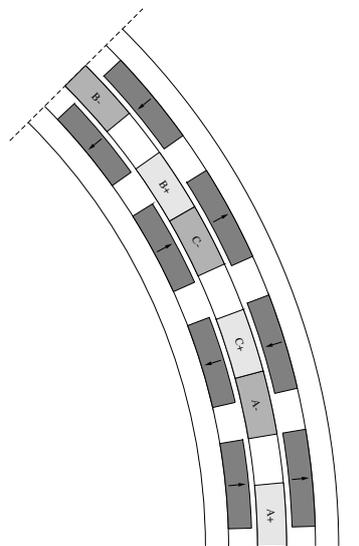


Figure 10: A radial flux, air-cored permanent magnet machine

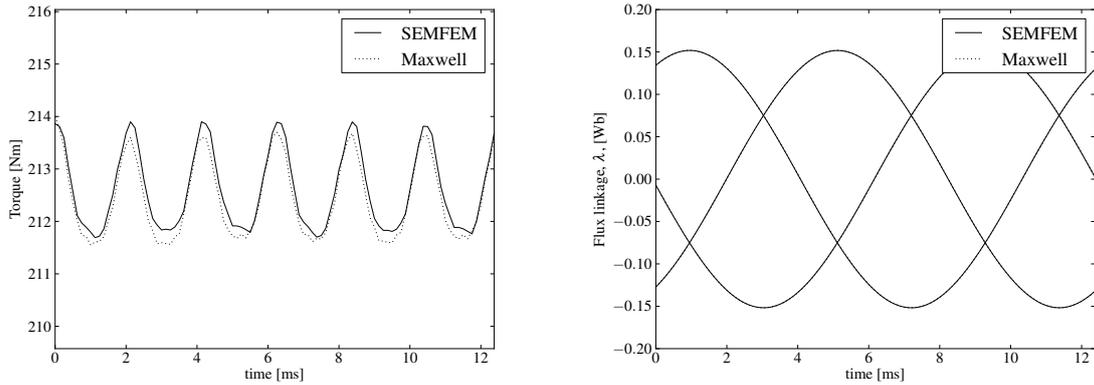


Figure 11: Comparison of torque and flux linkage waveforms for the rotating machine from SEMFEM and *Ansoft Maxwell 2D*.

## 4.2 Linear machine

The topology shown in figure 12 is presented as an example of a short stroke linear machine for application in a Stirling engine which can be used in solar dish collectors. The dimensions 1 through 8 shown in figure 12 as well as the phase of the current  $i$  were chosen as the design variables of the optimisation problem. The goal of the optimisation was to obtain a design which can meet required power and efficiency specifications while being as light as possible. The optimisation problem was formulated as

$$\begin{aligned}
 \text{Minimise} & : F(X) = m \\
 \text{Subject to} & : P_{out} > 10 \text{ kW} \\
 & \eta > 0.95 \\
 & B_{yoke} < 1.6 \text{ T} \\
 & B_{tooth} < 1.6 \text{ T}
 \end{aligned}$$

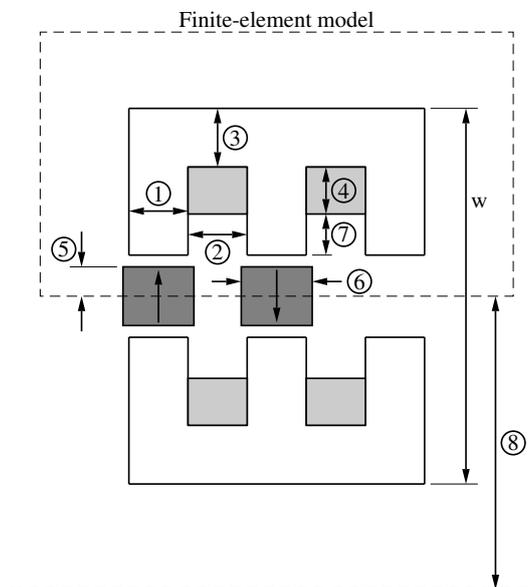


Figure 12: An example of a short stroke linear generator

Table 1: Optimisation results: Objective function and constraint values

Method	MFD	SLP	SQP	PSO
<b>m</b>	35.1 kg	32.5 kg	33.0 kg	33.7 kg
<b>P<sub>out</sub></b>	9.98 kW	9.99 kW	10.0 kW	9.98 kW
$\eta$	97.8 %	96.8 %	96.7 %	96.8 %
<b>B<sub>yoke</sub></b>	1.60 T	1.59 T	1.57 T	1.60 T
<b>B<sub>tooth</sub></b>	1.60 T	1.60 T	1.60 T	1.60 T

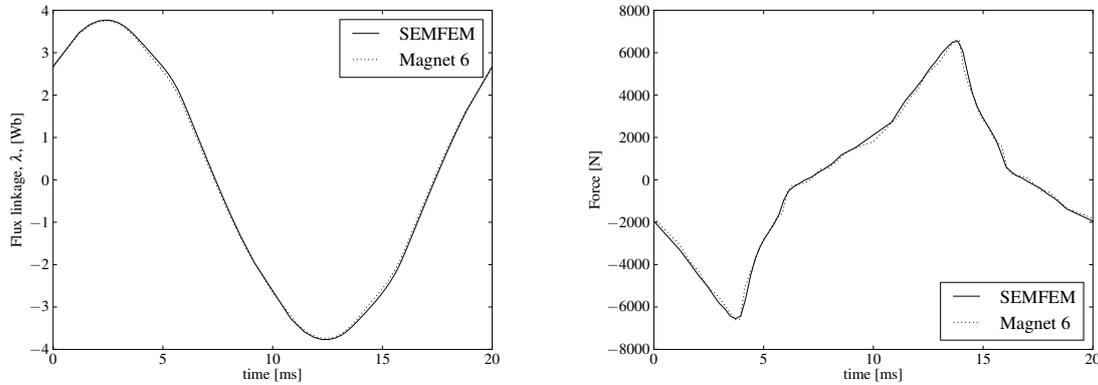


Figure 13: Comparison of force and flux linkage waveforms for the linear machine from SEMFEM and *Magnet 6*.

where  $X$  is the vector of design variables,  $m$  is the total mass of the machine,  $B_{yoke}$  is the peak flux density in the stator yoke and  $B_{tooth}$  is the peak flux density in a stator tooth. Reasonable upper and lower bounds were also placed on all of the design variables. This is necessary to ensure that the optimiser does not generate a set of design variables that will crash the simulation and cause the optimisation process to terminate before completion. The RMS current density in the coils was fixed at 4 A/mm<sup>2</sup> and the desired output voltage was set at 400 V for all simulations. The stroke of the machine was fixed at 40 mm.

The optimisation problem was solved using four different optimisation methods, namely the modified feasible directions method (MFD), sequential linear programming (SLP), sequential quadratic programming (SQP) and particle swarm optimisation (PSO). The first three methods are part of *dot*, a commercial Fortran optimisation program. These methods are all gradient-based. The fourth method is an implementation of the particle swarm algorithm as described by Vanderplaats Vanderplaats (2007) with a few additions by the author. The gradient-based optimisers have the advantage that an optimum is found using far less objective function evaluations. Global optimisation algorithms, such as the particle swarm algorithm, are generally considered to be more robust than gradient-based algorithms.

The results for the objective function and constraint values are tabulated in table 1. In this case, the different optimisation algorithms report roughly the same optimum objective function value. Table 1 shows that the results from the SLP, SQP and PSO methods are especially similar. This suggests that the optimum results obtained with these methods are trustworthy.

The optimal design was also simulated using Infolytica Corporation's *Magnet 6* to verify the accuracy of SEMFEM. A comparison of the results is shown in figure 13. Once again, the results are in very good agreement.

## **5 Conclusions and recommendations**

Finite element analysis coupled with numerical optimisation is a powerful method for the design of electrical machines. During this work, great progress was made in the development of software for this purpose. The versatility offered by having a customisable finite element simulation package available can be a great advantage in this design process. The package has proved that - for the targeted problem classes - it's capabilities are at least on par with that of commercial packages.

Possible future extensions of SEMFEM include

- Time-harmonic and transient simulation capabilities
- Eddy-current and hysteresis loss calculations
- Further parallelisation

The creation of a detailed user's guide, which would make this powerful tool easily accessible to users, is also a high priority.

## References

- Abdel-Razek, A., Coulomb, J., Feliachi, M. and Sabonnadiere, J. (1981), 'The calculation of electromagnetic torque in saturated electric machines within combined numerical and analytical solutions of the field equations', *IEEE Trans. Magn.* **17**(6), 3250--3252.
- Abdel-Razek, A., Coulomb, J., Feliachi, M. and Sabonnadiere, J. (1982), 'Conception of an air-gap element for the dynamic analysis of the electromagnetic field in electric machines', *IEEE Trans. Magn.* **18**(2), 655--659.
- Binns, K., Lawrenson, P. and Trowbridge, C. (1992), *The Analytical and Numerical Solution of Electric and Magnetic Fields*, John Wiley & Sons Ltd., Baffins Lane, Chichester, West Sussex PO191UD, England.
- Flack, T. and Volschenk, A. (1994), 'Computational aspects of time-stepping finite element analysis using an air-gap element', *Proceedings of ICEM'94, Paris* .
- Gerber, S., Strauss, J. and Randewijk, P. (2010), Evaluation of a hybrid finite element analysis package featuring dual air-gap elements, in 'Proceedings of the XIX International Conference on Electrical Machines', Rome.
- Shewchuk, J. R. (1996), Triangle: Engineering a 2D Quality Mesh Generator and Delaunay Triangulator, in M. C. Lin and D. Manocha, eds, 'Applied Computational Geometry: Towards Geometric Engineering', Vol. 1148 of *Lecture Notes in Computer Science*, Springer-Verlag, pp. 203--222. From the First ACM Workshop on Applied Computational Geometry.
- Vanderplaats, G. (2007), *Multidiscipline Design Optimization*, Vanderplaats Research & Development, Inc., 126 Bonifacio Place, Suite F, Monterey, CA 93940.
- Wang, R.-J. (2003), Design aspects and optimisation of an axial field permanent magnet machine with an ironless stator, PhD thesis, University of Stellenbosch.