Summer Research Project: Optimisation of Tidal Arrays through the use of Surrogate modelling and the ParEGO Optimisation Algorithm

Ben Ashby

Contents

Introduction

This project aims to find a methodology for the optimisation of an array of Tidal Turbines through the use of Computational Fluid Dynamics (CFD) and Evolutionary Algorithms. An actuator disk model (Svenning, 2010) was used to represent the turbines in the open source CFD package OpenFOAM, and an array of these disks was optimised through the use of the ParEGO optimisation algorithm (Knowles, 2005).

The following report details the steps taken to achieve this and the relevant theory, followed by a discussion of the results found using this approach.

Background and theory

Actuator Disk Model

The model used throughout this work was the Volume Force Actuator Disk Model first introduced by Svenning (2010), which was further developed for use with tidal turbines rather than wind turbines. The wake structure behind such turbines is recreated in the flow through the use of volume forces which act across the disk area, retarding the flow and introducing swirl. For a more detailed description of this model please see (Svenning, 2010).

This actuator disk representation of a Tidal Turbine provides a relatively inexpensive CFD model, allowing for the simulation of arrays of multiple turbines where other approaches would prove prohibitively expensive to run.

Surrogate Modelling

When dealing with expensive computational models, it can be more efficient to produce a statistical representation of the system in order to explore the model's characteristics quickly and easily. A model can be produced through a Kriging process, where the original model is sampled and interpolation used to predict the results for untested inputs.

Kriging provides an added benefit over other surrogate modelling techniques, such as Artificial Neural Networks, in that it provides a measure of the uncertainty of its prediction for each point. This uncertainty can be incorporated into a metric known as the Expected Improvement, which combines the absolute value predicted by the surrogate model with the uncertainty of the prediction. The Expected Improvement can be used to identify areas of the search space with the potential to contain optimal solutions that may otherwise be missed.

The surrogate modelling throughout this work made use of the DACE (Design and Analysis of Computer Experiments) toolbox designed for use with Matlab (Lophaven, et al., 2002). The algorithm runs using this surrogate model to identify the next solution to be tested on the original model at each iteration, and the Kriging model is refitted using the outputs of the original model increasing the certainty of the predictions at each iteration.

ParEGO algorithm

ParEGO is a multiple objective optimisation algorithm which incorporates surrogate modelling with an internal GA to produce a Pareto front of non-dominated solutions (Fu, et al., 2009). At each iteration, the algorithm constructs a Kriging model to the set of previous solutions to the problem, and uses a genetic algorithm to find a solution that maximises the Expected Improvement. This optimal solution is then run on the original model and the solution set updated.

The algorithm uses an aggregation function to combine the multiple objectives to a single scalar value for the solution set before constructing a surrogate model and running the GA. The aggregation function uses a weighting vector to compute the single output value, and this weighting is changed at each iteration to ensure that all areas of the Pareto front are explored by the algorithm.

For a more in depth description of the ParEGO algorithm please see Knowles (2005).

The Hypervolume Indicator

The Hypervolume indicator is a performance indicator used to compare the results of multi objective optimisation algorithms; it works by finding the size of the region of the objective space dominated by the solution set. The score is also linked to the range of the Pareto front. In order to calculate the Hypervolume later in this report, a matlab toolbox designed for this purpose was used (Cao, 2008).

Methodology

Set up of Domain

The domain used for the modelling was set up is shown in Figure 1, which represents a section of an array with three rows, the dashed lines show the boundary of the domain, with symmetry conditions applied to the side walls in order to shrink the domain modelled in CFD, and thus reduce the computational cost.

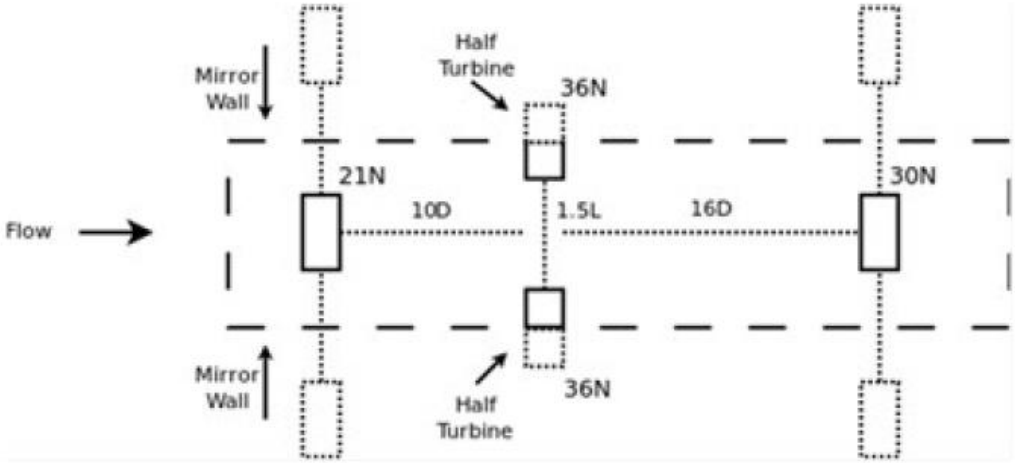

Figure 1: plan view of the domain used to explore array optimisation in this study (Guidolin, et al., n.d.)

By using this domain, the performance of the array is then dependent on 6 parameters to be varied throughout the study: the loading of each row of turbines, the downstream spacing between turbines in the 1st and 2nd row and the 2nd and 3rd row, and the lateral spacing between the turbines.

The loadings of the turbines were explored within the range found through a parameterisation study, which is detailed in the next section of this report, while the three geometric variables were kept between ranges specified by (Guidolin, et al., n.d.).

Parameterisation of Turbine

In order to find a suitable range of turbine loadings to explore within the optimisation process, a parameterisation study was carried out to find the power curve produced by the turbine under the flow conditions to be used throughout the study.

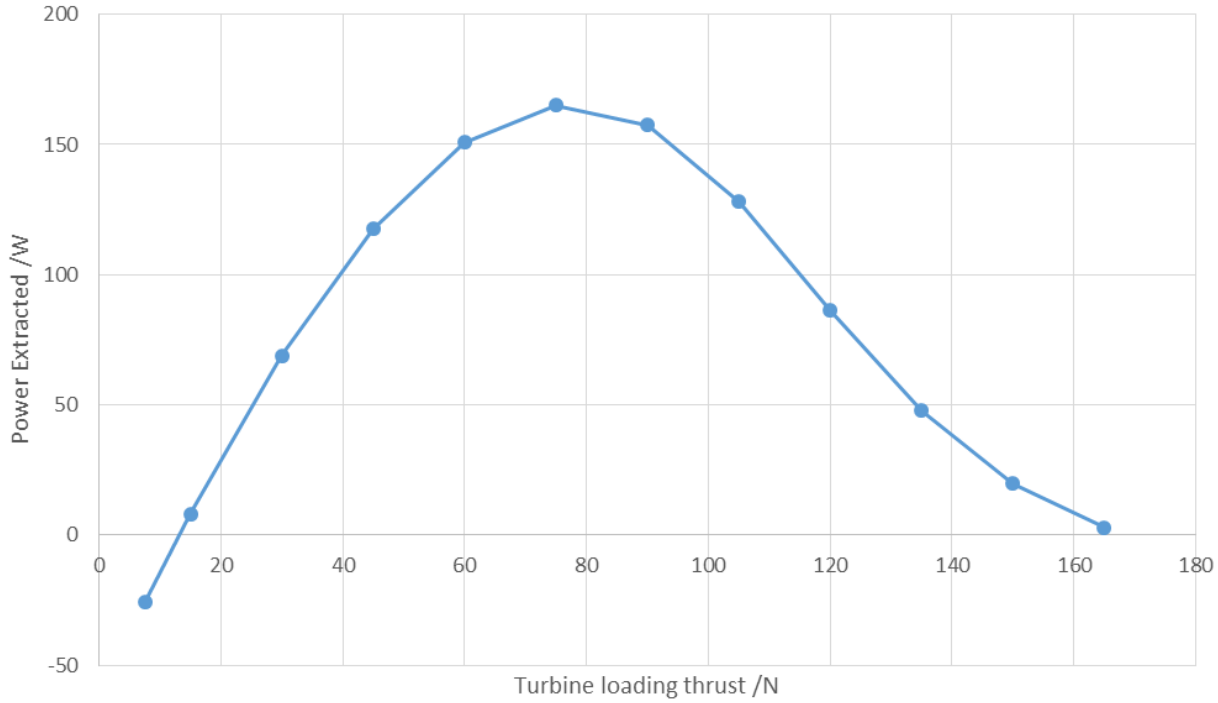

Figure 2: power curve of actuator disk model under the inlet conditions used throughout the study

Figure 2 shows the Power curve produced for a range of loading thrust set by the model, from this the upper and lower bounds on the thrust were set at 15N and 160N for the study.

The thrust to torque ration was maintained throughout the study and the corresponding limits on the torque were 0.25Nm and 2.75Nm respectively.

Thrust and Torque are the two inputs required by the actuator disk model, which are combined into a single loading parameter for each row. This parameter was varied between 0 and 1 and the appropriate thrust and torque inputs to the model were found by using the loading parameter as a weighting to select a value in the range found from the parameterisation study.

Latin Hypercube Sampling

In order to explore the search space effectively before constructing a surrogate model, a Latin Hypercube sample of 30 test cases was set up and run on the original CFD model. The sampling carried out was continuous, though for practical implementation the outputs were rounded to 3 d.p for the loadings and 2 d.p for the geometric parameters. The initial sample size was small considering the number of variables, though the ParEGO algorithm quickly identifies areas of the search space likely to hold optimal solutions and so the effects of this should be minimal.

Implementation of ParEGO algorithm

In order to implement the ParEGO algorithm, a matlab script was written to read in the solution set and perform the optimisation at each step. Since no way of automating the generation and analysis

of each case in OpenFoam was available, or easily attainable at this stage, the algorithm had to be reset and restarted for each iteration.

Due to the computation time per iteration, a parallel ParEGO approach was investigated, where at the end of each iteration 5 solutions are analysed using the CFD model, with each solution exploring a different area of the Pareto front. This approach should dramatically reduce the time required to produce an optimised set of solutions to the problem.

In order to assess the performance of the parallel approach, the traditional ParEGO algorithm was also used, and the results compared.

Results and Analysis

Parallel ParEGO Optimisation

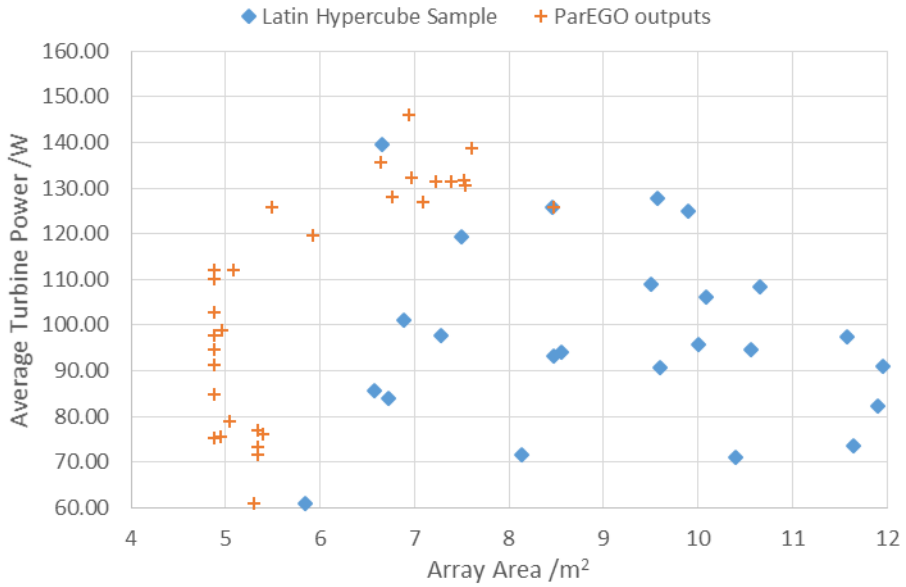

Figure 3: illustrating the outputs of the Parallel ParEGO optimisation after 6 iterations

Figure 3 shows the results of the parallel ParEGO implementation after 6 iterations, with a total of 30 cases run. There is a Pareto front forming towards the top left of the cluster, with the algorithm attaining a Hypervolume score of 83.75%. This shows that the algorithm is progressing quickly towards the true optimal set, if the algorithm were allowed to progress further a more complete Pareto set would be formed.

These results are encouraging, showing that this technique can generate a good set of results from a small number of iterations, though in practice the algorithm would need to be run for longer to achieve a full Pareto set with less clustering.

Traditional ParEGO Optimisation and Performance Indicators

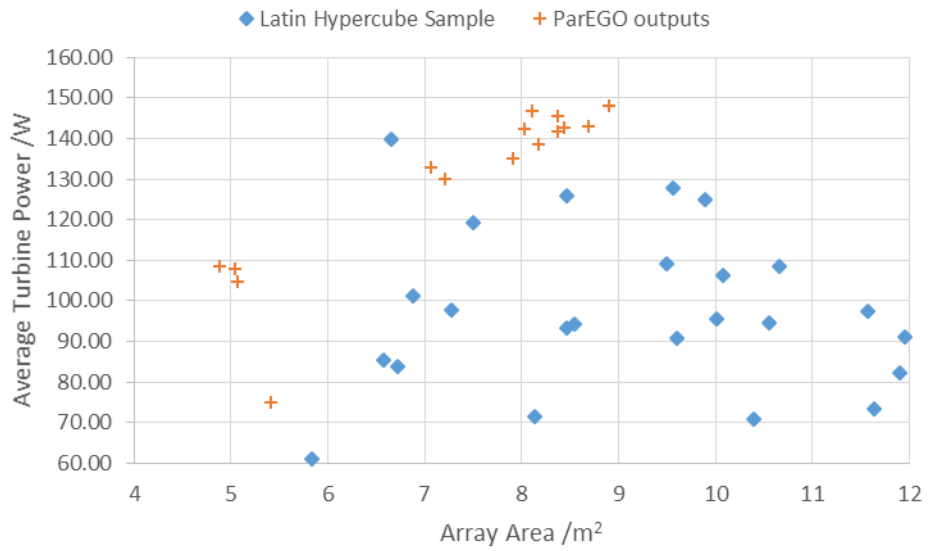

Figure 4: illustrating the outputs of the traditional ParEGO optimisation after 15 iterations

Figure 4 shows the results from the traditional run of ParEGO after 15 iterations, with a Hypervolume score of 66.71%. The time required per iteration limited the number of iterations carried out in this work.

The results of the traditional run of ParEGO can be compared with the parallel implementation after 3 iterations, as at this stage both algorithms have required 15 cases to be evaluated on the original CFD model.

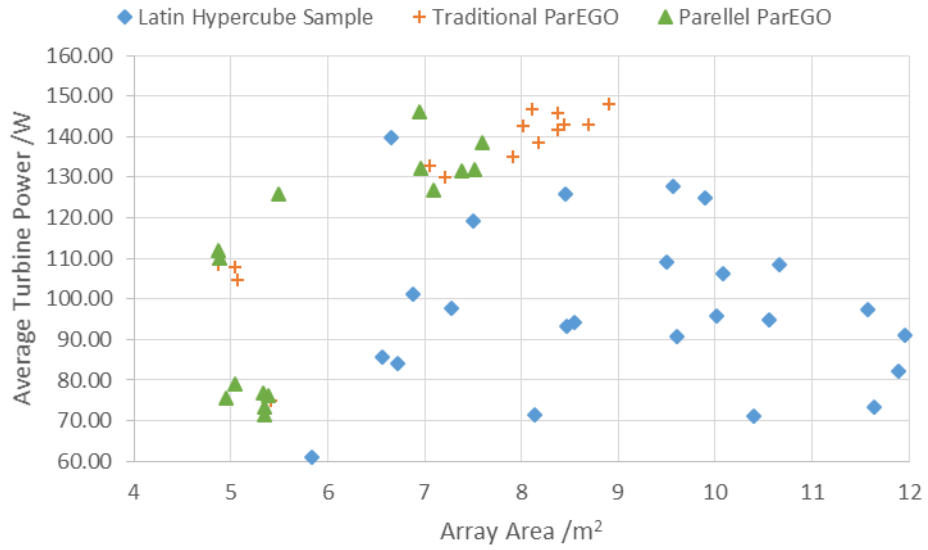

Figure 5: Comparison of traditional ParEGO and a parallel implementation after 15 cases evaluated using the CFD model

Figure 5 shows, somewhat surprisingly, that the Parallel implementation produced a better fit after 15 cases were tested. Given that the solution set is updated after each iteration in the traditional approach, providing more information to the algorithm at each subsequent test case and so the results would be expected to be more advanced than the Parallel approach. However the

hypervolume shows that the traditional ParEGO score of 66.71% is far lower than the parallel implementation score of 81.75% for the same number of cases.

This difference can be in part attributed to the distribution of the points along the Pareto front for each variant of ParEGO; the traditional algorithm formed a cluster around a region with a high power output and larger area, with 9 of the 15 cases being found in this region. In comparison the results from the Parallel algorithm were more evenly spaced. Since the spread of the Pareto front contributes to the hypervolume score this explains the difference in this indicator. When selecting the next batch of cases to be run in parallel care was taken to avoid any duplicate cases, this would have ensured that a wider range of outcomes were found than the traditional case where the algorithm could search a similar area at each iteration.

The clustering of values for both implementations of the algorithm may also be linked to the initial sampling, due to the time required to run each case the initial sample size of 30 was small considering the complexity of the scenario to be modelled by the surrogate. This low resolution initial model would mean that the algorithms would require a higher number of iterations to produce a complete set of optimal solutions along the Pareto front, and meant that the model was more likely to search for solutions near to the better solutions found in the original sample.

Another factor in the performance of the surrogate model was that the prediction for the area of an array took into account all 6 input parameters, when in fact it was solely linked to the geometric parameters, which would have introduced some noise to the model.

Conclusions

The results of this study suggest that this approach could be of much use for the design of tidal arrays in the future, ParEGO can achieve a set of strong solutions from a small number of iterations, and given the computational cost of each iteration this is crucial for this approach. By running ParEGO in parallel the time taken to run this process is reduced dramatically, though some information is lost in comparison to the original implementation, where the solution set is updated after each test case. This suggests that through running ParEGO in parallel, the results is a fast and powerful multi-objective optimisation tool.

A good Pareto set was achieved from the parallel implementation of ParEGO after just 6 iterations, and although there were areas of the Pareto front where there are few solutions, the algorithm would better explore these regions if it were allowed to run further.

This study was carried out under the assumption that the parameters used in the design of a tidal array could all be continuous, whereas in reality the geometry may need to be considered as discrete due to the tolerances available when installing tidal turbines. In this case the approach taken in this study would still be of use, and would provide a fast and efficient method for identifying the ideal solution set.

Bibliography

Cao, Y., 2008. *Hypervolume Indicator.* [Online]
Available at: http://uk.mathworks.com/matlabcentral/fileexchange/19651-hypervolume-indicator
[Accessed 27 August 2015].

Fu, G., Khu, S. & Butler, D., 2009. Use of surrogate modelling for multiobjective optimisation of urban wastewater systems. *Water Science and Technology,* 60(6), pp. 1641-7.

Guidolin, M. et al., n.d. Toward the optimal design of a tidal farm layout using a CFD model. *Not yet published.*

Knowles, J., 2005. ParEGO: A Hybrid Algorithm With On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems. *IEEE TRANSACTIONS ON EVOLUTIONARY COMPUTATION,* 10(1), pp. 50-66.

Lophaven, S., Nielson, H. & Søndergaard, J., 2002. *DACE: A Matlab Kriging Toolbox,* Lyngby: Technical University of Denmark.

Svenning, E., 2010. *Implementation of an actuator disk in OpenFOAM,* Gothenburg: Chalmers University of Technology.