9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization
4-6 September 2002, Atlanta, Georgia

AIAA 2002-5507

EMPIRICAL COMPARISON OF GRADIENT-BASED METHODS ON AN ENGINE-INLET SHAPE OPTIMIZATION PROBLEM

András Sóbester* and Andy J. Keane†
University of Southampton
Southampton, UK

ABSTRACT

With the development of increasingly sophisticated adjoint flow-solvers capable of providing objective function gradients at reasonable computational costs, modern deterministic gradient-based search methods have come to be regarded as amongst the most powerful tools in aerodynamic shape optimization and MDO problems. However, their performance can be disappointing when the objective function landscape features multiple local optima, long valleys, noise or discontinuities. Equally, stochastic global explorers, such as Genetic Algorithms (GAs), while less affected by these problems, are relatively slow to converge. In this paper we propose GLOSSY (Global/Local Search Strategy), a generic hybrid approach, which combines a global exploration method with gradient-based exploitation. We analyze the performance of two optimizers based on the GLOSSY framework (fusing a GA with a quasi-Newton local search method) and we show through a set of comparative tests that on the moderately noisy objective landscape of a jet-engine inlet shape optimization problem the hybrid outperforms both of its components used individually. We also look at the issue of what global / local search effort ratio gives the hybrid the best performance.

INTRODUCTION

In the optimization community in general and in the field of aerodynamic shape optimization in particular, there has been a long-running debate about the use of gradient-based local improvement procedures versus stochastic global exploration methods. Both categories have seen significant developments over past decades. While in the early days of numerical aerodynamic shape optimization[1] (late 70's) the concept of gradient-based search was almost equivalent to that of the steepest-descent algorithm, by now a wide range of sophisticated optimizers of this class have entered everyday design practice. Steepest-descent is still used occasionally, but it has been gradually superseded by modern algorithms based on the Newton method (with line-search or trust-region-type implementations), quasi-Newton methods (BFGS, DFP) and conjugate gradient optimizers (Fletcher-Reeves, Polak-Ribiere)[2]. On the other side of the argument, stochastic exploration methods have also evolved from tentative simulations of physical and biological phenomena into a set of powerful search tools, the most popular being Simulated Annealing (SA) and Genetic Algorithms (GA). Hajela[3] provides a recent survey of these and other zeroth-order methods from a Multidisciplinary Optimization (MDO) perspective.

As the body of experience on these two main categories of search techniques grew, so did the design community's awareness of their respective limitations. Gradient-based local searches, while very efficient on many smooth, unimodal objective function landscapes, often provide less then satisfactory results when the problem exhibits valleys and/or multiple local optima. Once trapped in a valley or at a local optimum the search needs to be re-launched from a new (commonly random) starting point. This operation usually involves wasteful, lengthy exploration of unpromising regions of the search space, such as those with very poor objective values or virtually flat regions (until the neighborhood of a local optimum is reached) and one can only hope that the new starting point is in the basin of attraction of a thus far unexploited local (or perhaps the global) optimum. Conversely, global explorers, such as GAs, are good at leaving poor objective value regions behind quickly, while simultaneously exploring several basins of attraction. What *they* lack is high convergence speed and precision in the exploitation of individual local optima.

To summarize: global explorers are good at locating basins of attraction – gradient-based local searches are good at descending into them. Additionally, like most zeroth-order methods, global explorers are less affected by noise in the objective landscape. A solution that suggests itself is attempting to get the best of both worlds by combining a local gradient-based improvement procedure with a global explorer to form a hybrid method that allocates the available computational resources between the two in an efficient manner. As the search engine that we are proposing in this paper is based on the global/local hybridization

* Graduate Student.
† Professor, School of Engineering Sciences

principle, we delve more deeply into this issue in the next section.

A crucial factor that affects the performance of any gradient-guided search ("pure" or hybrid) is the computational cost and quality of the gradient. Finite differencing, the most straightforward gradient calculation method is problematic on both counts. As the evaluation of each component of the gradient in a given point of the search space requires an evaluation of the objective function, its computational burden can be immense. This is a particularly great drawback in an MDO context, where the number of dimensions of the search space (and hence the number of components of the gradient) is likely to be high. Using cheaper, partially converged objective function values is not a solution, as the noise on the gradient thus obtained by finite differencing is usually too high to be of any value for an optimization algorithm.[4,5]

A paper published by Jameson[6] in 1988 signaled a breakthrough in the gradient evaluation problem. In this seminal contribution he introduced the adjoint method for aeronautical computational fluid dynamics (CFD), a technique whereby a new set of equations can be constructed (based on the solution of the flow equations), which, at a computational cost similar to that of the solution of the flow equations, yields all components of the objective function gradient. Jameson, Reuther and other coworkers developed the method for potential flow, the Euler equations and the Navier-Stokes equations, at the same time proving its benefits from the optimization point of view with applications ranging from 2D aerofoil design[7] to the optimization of high-lift systems[33] and full aircraft configurations.[5, 8-10]

A number of other research groups have also performed aerodynamic design optimization using adjoint flow solvers: Monge and Tobio[11], Eliott and Peraire[12], Giles and Pierce[13], Anderson and Venkatakrishnan[14], Kim, Obayashi *et al.*[15,16], Iollo, Salas and Ta'asan[17], Arian and Vatsa[41], Nemec and Zingg[34], etc. (for a comprehensive survey see Ref. 18).

To date, as far as we were able to ascertain, all reported aerodynamic design applications of the adjoint method have relied solely on deterministic gradient-based optimization. Amongst these, quasi-Newton methods (BFGS, in particular) and the smoothed gradient method[10] (based on the principle that aerodynamic shapes are predominantly smooth) appear to have the best performance.

A detailed comparative study of these approaches has been conducted by Jameson and Vassberg[19] on the brachistochrone, a classic calculus of variations problem (with a known analytical solution), considering every mesh point on the curve (the shape of which is to be optimized) as a design variable. However, very few comparative studies exist in the literature that assess the relative merits of gradient-based deterministic optimizers and stochastic search methods (comparisons between single runs of SA, gradient-descent and GA have been described by Obayashi[20] and Sasaki[21]). As pointed out earlier, pure gradient-based methods may be the best choice on some smooth and unimodal landscapes, but when several local optima and/or convergence noise are present, their superiority is far from obvious.[22] Additionally, as Ta'asan[35] points out, high dimensional landscapes resulting from the discretization of partial differential equations (such as the conservation equations of a flow field) often have badly conditioned Hessians (i.e., the level curves around optima are long ellipses, rather than circles), which reduce the efficiency of pure gradient-based searches.

For such design cases we propose a global/local hybrid approach and we measure its performance against that of its two components, two optimizers often used separately in aerodynamic design applications: a GA and a quasi-Newton search using the BFGS update[36]. We also look at one of the critical issues of global/local hybridization: how to divide the available search time between the two components of the hybrid.

The problem used in the empirical comparative tests described here is the shape optimization of a jet engine inlet. A full potential flow solver is employed to evaluate the objective function and its gradient via an adjoint solution.

GLOBAL/LOCAL HYBRIDS

An increasingly popular way of overcoming the shortfalls of global exploration methods (e.g., GA) and gradient-based local searches is to combine them into a hybrid. The backbone of such a hybrid is the scheme that allocates the available search-time between the exploration and the exploitation process and determines which designs will take part in exploration or exploitation.

The optimization literature provides a wide variety of search-time division strategies. In an aerodynamic shape optimization context, Vicini and Quagliarella[23] describe an application of one of the most popular and most straightforward such heuristics. They interrupt a GA search once every second generation and then perform 2-3 steps of conjugate gradient local search (the gradients being evaluated by finite differencing) on the best individuals (a conjecture is made that running the local improvement procedure to complete convergence would have a negative effect on the overall performance of the algorithm). The improved individuals are then fed back into the population where they continue to evolve within the GA framework (this is termed Lamarckian learning).

A similar approach is to run the GA for a preset number of generations and then improve the best individual(s) with the local search method. However, such two-phase techniques involve a serious implementation difficulty: there is no reliable way of determining the best moment for switching from exploration to exploitation. This problem exposes one of the major conundra in hybrid optimizer design: "what percentage of the total available search time should be devoted to exploration as opposed to exploitation?" Several studies[24-26] have looked at mathematical models aimed at calculating the optimum search-time division ratio – however, the results are difficult to apply in practice, mostly because the proposed mathematical models require an in-depth knowledge of the optimizers as well as of the landscape under scrutiny.

A possible way of deciding when to shift from global search to exploitation is to control the search time division adaptively. In such hybrids the responses of the system govern the resource allocation[28,29]. The allocation heuristic usually takes feedback from the search by measuring the improvements made by both approaches and, based on this "reward" data, it decides whether exploration or exploitation should have its share of CPU time increased or decreased.

An added problem in devising evolutionary/local search hybrids is that in most cases it is not obvious *which of the individuals* resulting from the evolutionary process should undergo local improvement. Common practice is to pick the fittest member of the GA population – however, as there is no reliable way of telling how far from convergence a GA population is, there is no guarantee that this will lead to the global optimum[‡].

With these considerations in mind we have constructed a generic search time division control framework, which allows the combination of a population-based global explorer and a local improvement procedure using either simple (deterministic) or adaptive control of search time allocation. Our template is based on the principle of *reproductive isolation*. This idea (often encountered in the evolutionary optimization literature[37-40]), spawned by the natural metaphor of species competing for the same resources, involves running the local and global optimizers in separate populations with periodical spells of migration. The adaptation is accomplished by resizing the populations before the migration periods according to the relative average objective function value improvements achieved by the two methods in their respective populations (i.e., the more successful species grows at the expense of the less successful).

We next describe this family of hybrids in more detail.

THE GLOSSY TEMPLATE AND TWO IMPLEMENTATIONS

GLOSSY (Global/Local Search Strategy), the hybridization framework that we propose in this paper is shown in figure 1. This general structure can be used for the implementation of several types of exploration/exploitation methods.

First, an initial population of designs is generated. This is followed by an allocation step, which splits up these individuals into two populations (**GP** and **LP**) according to a pre-established initial size ratio. The designs in the Global Population (**GP**) will participate in the exploration procedure (for a number of *SG* generations – this value is also set at the allocation step), while those in the Local Population (**LP**) will each be improved using the exploitation procedure (for a number of *SL* steps – again, *SL* is set at the initial allocation stage).

On a parallel architecture the two procedures would take place simultaneously – in that case *SG* and *SL* have to be chosen such that approximately the same wall-time/cycle will be required by the two methods. If the flow solver itself is capable of domain decomposition and thus efficient parallel computation, the values of *SG* and *SL* can be selected to optimize the convergence speed of the algorithm (in terms of overall number of evaluations of the objective function and its gradient) without compromising overall computational efficiency.

Each cycle is concluded by a reallocation step, which decides the size and composition of the two populations for the next cycle. The number of global search generations or local improvement steps per cycle can also be adjusted here – again, this may become necessary in a parallel implementation if during the search, say, the number of objective function evaluations per local search iteration increases and therefore idle time starts appearing on the nodes processing the global population. The reallocation step determines the nature of the search-time division control scheme (deterministic or adaptive). Many reallocation schemes are possible, so it is perhaps more enlightening to look at some possible techniques with reference to specific global and local search methods; therefore, we first tackle the issue of what strategies we

‡ This difficulty suggests another line of attack: the application of statistical population models for assessing the modality of the search space. The interested reader is referred to a preliminary study conducted on an artificial test function by Hacker *et al.*[27]

want to combine for the application presented in this paper.

Choosing an optimizer for a particular type of application is still a black art, since, as we mentioned in the introduction, there are very few truly conclusive comparative studies. In keeping with what seems to be the prevailing opinion in the optimization community, we have selected a quasi-Newton method with the BFGS update scheme as the local (exploitation) element of our hybrid searcher.

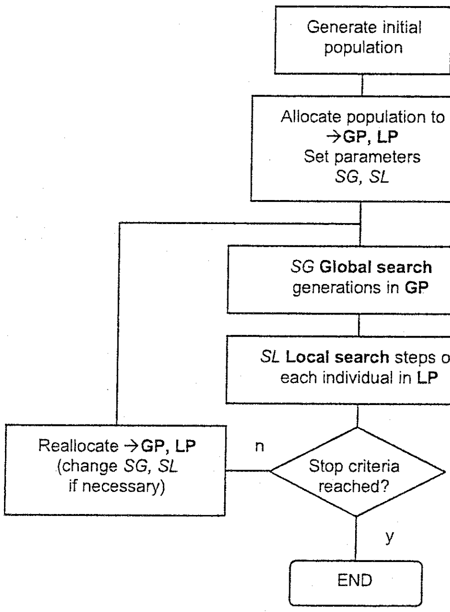

Figure 1. Flowchart of the GLOSSY hybrid template.

With regards to the global searcher, the most difficult task in applying methods of this class to real-life problems is the appropriate tuning of their parameters (mutation step sizes, operator probabilities, SA cooling schedules, etc.). We have selected a simple GA as the global component of the hybrid mostly because a substantial amount of historical data is available (both in the literature and in the authors' own work) on parameter choice for this type of algorithm. We stress here that the GLOSSY framework is by no means limited to these two types of searches – other combinations could include GA+SA, GA+Simplex, SA+BFGS, etc.

Returning now to the reallocation scheme, perhaps the simplest strategy is to maintain the population sizes constant and after a number of *SG* GA generations (in the global population) and *SL* BFGS iterations applied to each individual in the local population reshuffle the populations. The migration of individuals can be done in several ways – the simple heuristic we have opted for is illustrated in figure 2. In the following we will refer to this strategy as GLOSSY Mk1.

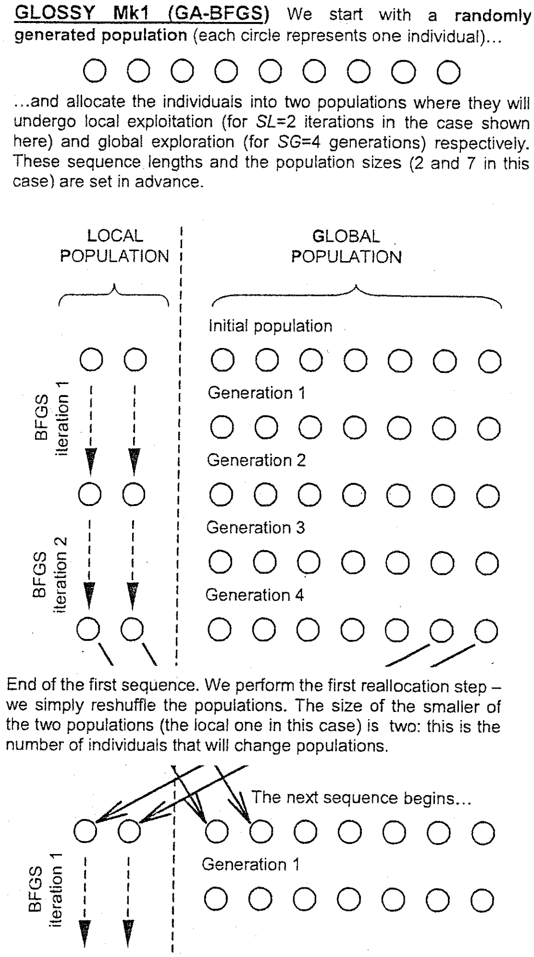

Figure 2. GLOSSY Mk1.

As with all deterministic hybrid schemes one needs to know (or guess) before starting the search what the optimum resource allocation ratio is. We will examine this problem using an empirical approach in the following sections of this paper. Our application may be, to some extent, representative from the optimum resource allocation point of view of aerodynamic optimization problems in general. Nevertheless, the ratio found to give the best results here may not be optimal in other circumstances (objective function values generated by higher fidelity flow solvers, other global/local optimizers).

A possible way to overcome this problem is to start with a "sensible" resource (population) division, measure the performances of the two methods before each reallocation and resize the populations according to the relative improvements per individual per objective function evaluation achieved by the two methods. The varying population size raises the question of which individuals to migrate from one population to the other. Again, several schemes are possible – the one we have adopted for is shown in figure 3. We call this adaptive strategy GLOSSY Mk2.

In order to assess the feasibility and performance of these hybrid schemes we have conducted a number of test runs on an aerodynamic shape optimization problem. We now go on to briefly describe this application.

APPLICATION – SHAPE OPTIMIZATION OF A TURBINE INLET

When planning the comparative study presented here we had to reconcile two conflicting objectives. First, we had to fit into the available computing time a sufficiently large number of optimization runs to ensure the statistical relevance of the collected optimizer performance data. In our experience about 50 runs are required to get a reasonable estimate of the mean and the variance of the objective function values obtained by the optimizers at various stages (this is particularly important when comparing the performance of the same optimizer with slightly different parameter settings). Therefore, we have opted for 50 runs on each optimizer / parameter set.

The second objective has been to use a computational model of a real aerodynamics problem that is accurate enough for useful conclusions to be drawn (with respect to relative optimizer performance) that could be applied to some extent in the case of high-fidelity flow models as well.

With this tradeoff in mind, we have chosen the two-dimensional model of the steady potential flow around a turbine inlet lip. An input mass flow rate has been specified on the right-hand boundary (the fan face). The flow is subsonic throughout the domain. We have used a solver with an adjoint capability to determine the peak value of the surface flow velocity (around the inlet) as well as the gradient of this objective function.

In order to be able to achieve reasonably fast full convergence (again, having in mind the large number of runs required for each optimizer / parameter setting) we have adopted a low variable-number shape parameterization approach: the bump function method introduced by Hicks and Henne[1] (as recommended, e.g., in Refs. 30, 31).

Two such bump functions have been added to the profile: one to the outside skin of the inlet and one to the inside (the leading edge point separates the two regions– see figure 4).

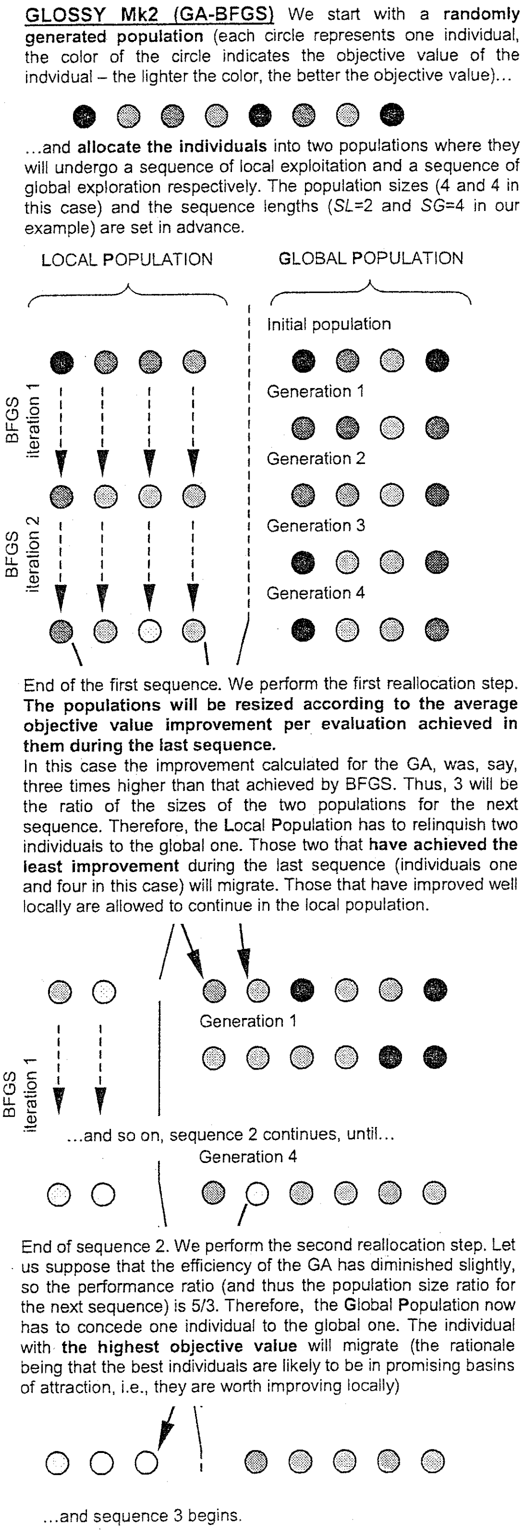

Figure 3. GLOSSY Mk2.

They have the following general form:

$$b(x) = A\left[\sin\left(\pi x^{-\frac{\ln 2}{\ln x_p}}\right)\right]^t, \quad x \in [0,1] \tag{1}$$

where A is the height of the bump, x_p is the location of the peak of the bump and t is a parameter that controls the width of the bump (large values of t correspond to sharp bumps). Thus, we used six variables in total – the two A, x_p, t triplets defining the two bumps.

Running the flow solver complete with the solution of the adjoint system takes about 130 seconds to full convergence – for the purposes of these experiments we have only converged the solutions partially. This results in a slight noise in the objective function (particularly in its gradient), but the run time was cut down to about 80 seconds (such noise is present in more sophisticated flow solvers anyway).

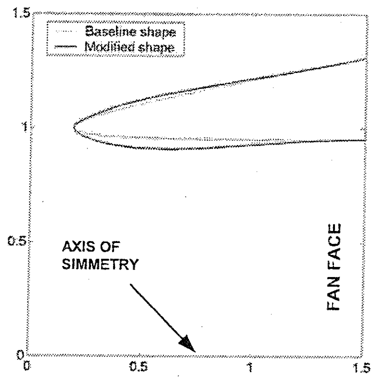

Figure 4. Half-section through the inlet. The baseline shape (dotted line) is visible inside the contour resulting from the addition of two Hicks-Henne bumps (randomly generated in the case shown here).

EMPIRICAL COMPARISONS

As one of the central issues of global/local hybridization is how to split the available search time and allocate designs to the two components of the combined optimizer, we have devoted most of our test runs to finding an empirical answer to this question. We have conducted the bulk of the tests using the GA-BFGS hybrid based on the GLOSSY Mk1 strategy, which offers two possibilities for obtaining various resource division ratios. One can modify either the relative sizes of the two populations or the relative sequence lengths (number of GA generations and number of BFGS iterations). Of course, the effects of modifying the exploration/exploitation time ratio with these two methods will not be the same (for example, improving three individuals with two BFGS iterations each at every GLOSSY Mk1 cycle can lead to a very different overall result than having one individual in the local population, improved with six iterations in every cycle – even though the total exploitation effort would generally be the same).

The comparative tests presented include results generated on optimization runs with a maximum budget of 200 equivalent evaluations of the objective function. The word "equivalent" refers to the fact that the BFGS and the GLOSSY hybrids require not only the objective function values but also gradient information. Using the adjoint flow solver mentioned in the previous section the latter *together with the former* can be obtained for a design at about 2.5 times the computational cost of one direct objective value. Therefore, if the cost of the objective function alone is 1 unit then the cost together with the gradient is about 2.5 units. All comparative results contain data collected over 50 independent optimization runs each.

In our experience the majority of evolutionary algorithms work most efficiently on low-dimensional problems of this type with relatively small population sizes. Therefore, all runs presented here operate on populations of N=8 individuals. The pure selecto-recombinative GA, as well as the GA components of the GLOSSY hybrids (the same GA algorithm has been built into the hybrids) have been run with a probability of crossover of 0.8 and a probability of mutation of 0.1[§].

The results of the comparative tests are shown in table 1, which contains the mean objective values obtained after 200 equivalent evaluations of the objective function using the various optimizers / parameter settings. In addition to the sample means, the table also shows the standard deviations of the objective function samples (over the 50 runs).

Let us make one final point before going on to the discussion of these results. Testing heuristics is a notoriously difficult task for several reasons[32] and the results are often controversial. First, one tends to fine-tune one's own proposed technique and compare it with "the others" run with default parameter settings. Furthermore, stochastic methods, almost by definition,

§ A small number of trial runs suggested that on this problem these settings lead to better results than those usually used as defaults – 0.6 and 0.05. However, we note here that no substantial effort was invested into fine-tuning the GA itself, as our main goal was to compare the GA with a GA-BFGS hybrid, as well as GA-BFGS performance results obtained with different settings of the GLOSSY hybridization scheme, rather than the GA with some other "pure" method.

often perform very differently from one run to the next. Therefore, any comparison is only as good as the statistical relevance of the results. While admitting that no comparison is perfectly fair, we believe that using the same population size for every optimizer, employing the same GA with the same parameter settings in both its "pure" and hybridized form and looking at both the mean and the variance of the results over relatively large samples (50 runs) goes a long way towards providing fair comparisons.

The first set of tests was aimed at investigating the behavior of GLOSSY Mk1 with various global / local population sizes. The performance figures are shown in rows 1-4 of the table. The ultimate objective values, as well as the standard deviations are clearly the lowest for those cases where only one or two individuals are in the local exploitation population, with the rest undergoing global exploration.

In order to test the adaptation capabilities of GLOSSY Mk2, we set up a test run where we started the adaptive hybrid with equal global / local population sizes. As figure 5 indicates, the adaptive heuristic also recognizes that the algorithm works best with a small local population. The plot shows the variation of the average global population size, with the final value (after 200 equivalent evaluations) being just over 6.

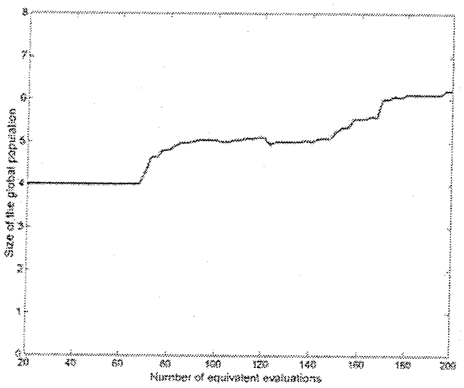

Figure 5. Variation of the global population size when using the adaptive hybrid (GLOSSY Mk2), starting with equally split (4-4) populations. The plot is averaged over 50 runs.

This matches the conclusions reached from the GLOSSY Mk1 runs with different population sizes. The actual performance figures (see row 12 in table 1) also confirm the conclusion frequently encountered in the adaptive optimization literature, that the performance of a finely tuned optimizer is always better than that of an adaptive one (i.e., GLOSSY Mk1 when correctly set up outperforms GLOSSY Mk2). In this case the ultimate objective value is similar to those obtained with the GLOSSY Mk1 runs with the largest global population sizes, however the standard deviation of the ultimate objectives reached by the adaptive heuristic is considerably worse.

Returning now to our investigation of optimum resource division in GLOSSY Mk1, there is another aspect to this problem: the sequence lengths of the two optimizers. Initial tests suggested that the performance of the algorithm is more sensitive to variations in the length of the BFGS sequences (*SL*) than in the number of GA generations per cycle (*SG*). Therefore, we have performed a series of seven test runs with *SL*=1..7 and *SG* held constant at 3. The population sizes, based on our previous conclusions (from runs with both GLOSSY Mk1 and Mk2), were set to 1 in the local improvement population and 7 in the GA population.

The results are shown in rows 5-11 of table 1. The best results are achieved with BFGS iteration/cycle numbers of 4 and 5 (rows 8 and 9) – both the mean ultimate objective achieved and the standard deviations are the best of the whole test series.

As the plot of the sample characteristics after 200 equivalent evaluations (figure 6) indicates, an iteration number of 4 is probably the best choice, as the standard deviation is already fairly large at an *SL* value of 5, in spite of a slightly better mean.

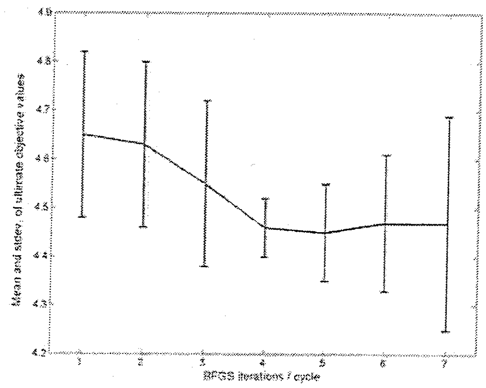

Figure 6. Mean ultimate objective values ($\mu_1 \ldots \mu_7$) achieved by GLOSSY Mk1 with different numbers of BFGS iterations/cycle. The error bars indicate the ranges ($\mu_i+\sigma_i$, $\mu_i-\sigma_i$), i=1..7, where $\sigma_1 \ldots \sigma_7$ are the corresponding standard deviations (the values represented are those from table 1, rows 5-11).

Finally, let us look at the other main point of this study: how does the GLOSSY-based GA-BFGS hybrid compare with its components when they are used separately (as they are in all the adjoint-optimization work seen to date by the authors)? Rows 13 and 14 of table 1 contain the performance results for the pure BFGS and the pure GA. Figures 7 and 8 give a clearer insight into the comparative performances of these optimizers. Figure 7 shows the averaged optimization histories of the GA and GLOSSY Mk1 (run with a local population of a single individual, undergoing 4 BFGS iterations of local improvement and 3 GA generations per cycle, i.e., as per row 8 of table 1). The two history curves as well as the variation of their standard deviations have similar shapes, indicating the effect of the GA in the hybrid. However, the use of the adjoint gradients via the Lamarckian local improvement component (the BFGS) clearly makes a difference: the GLOSSY objective values are better throughout the runs and the standard deviation of the ultimate objective value sample is also slightly better than that of the GA.

A slightly different picture emerges from figure 8, a comparison between the averaged history of a set of pure BFGS runs and the same GLOSSY Mk1 history as shown on figure 7. The initial convergence (up to about 120 equivalent evaluations) of BFGS is better, but from that point onwards the hybrid gives considerably better improvement, accompanied by a dramatic reduction in the standard deviation of the objective value sample (0.06 compared to BFGS's 0.48 at the end of the run).

The mean ultimate objective values for GA, BFGS and for GLOSSY Mk1 are 4.67, 4.65 and 4.45 respectively. The best design overall (objective function value of 4.36) has been found using GLOSSY Mk1. None of the pure BFGS or GA runs has found this optimum. With the parameter settings shown in row 8 of table 1 the 50 ultimate objective values produced by GLOSSY optimizer fall into the range (4.36,4.92). The ranges for GA and pure BFGS are (4.38,5.06) and (4.37,6.08) respectively.

CONCLUSIONS

The advent of adjoint flow solvers opened up a whole range of new possibilities in aerodynamic shape optimization. The availability at reasonable computational cost of objective function gradients makes gradient-based searches vastly more efficient than they used to be with gradient information obtained by finite differencing. Nevertheless, even the most sophisticated techniques of this kind still have a major shortcoming: their inability to tackle multimodal and / or noisy landscapes efficiently. The GLOSSY hybridization framework proposed in this paper is aimed at overcoming this problem. We have shown through a set of comparative tests on a simple aerodynamic shape optimization application that such an approach is feasible and that it is an efficient way of combining the global exploration capability of a GA with the fast local convergence of the BFGS.

We do not claim that the conclusions drawn from the simple problem used in this paper are universal. Indeed, we plan to test our scheme using more complex problems and high fidelity flow solvers in the future. Yet, we do believe that the experimental results presented here can serve as a useful set of guidelines for designers.

Should the optimum population sizes found here for our non-adaptive deterministic hybrid work badly on some other problem, we recommend the adaptive variant. This was shown to confirm the results obtained with different settings of the deterministic optimizer.

The comparative tests presented here were based on the assumption that the optimization process is serial (i.e., the results apply on parallel architectures only when the flow solver itself is parallel). Nevertheless, the GLOSSY framework does allow parallel implementations, the only constraint being that for an efficient parallelization one needs to choose the sequence lengths in such a way that none of the methods "runs over" the other by a significant amount on each cycle.

The same general framework can also be used for the implementation of a wide range of different global and local search methods; we merely demonstrated the capabilities of the scheme using GA and BFGS.

ACKNOWLEDGEMENTS

This work has been supported by the University Technology Partnership for Design, a collaboration between BAE Systems, Rolls-Royce and the Universities of Sheffield, Cambridge and Southampton.

The authors would also like to thank Mike Giles, Mihai Duta and Juri Papay for their help in obtaining the results presented.

REFERENCES

[1] Hicks, R., Henne, P., "Wing Design by Numerical Optimization", *Journal of Aircraft,* Vol. 15, No. 7, 1978. pp. 407-412.

[2] More, J. J., Wright, S. J., "Optimization Software Guide", *Frontiers in Applied Mathematics 14,* SIAM 1993.

[3] Hajela, P., "Nongradient Methods in Multidisciplinary Design Optimization – Status and Potential", *Journal of Aircraft,* Vol. 36, No. 1, 1999.

[4] Reuther, J. J., "Aerodynamic shape optimization using control theory", PhD Thesis, University of California, Davis, May 1996.

[5] Reuther, J. J., Jameson, A., Alonso J., Rimlinger, M. J., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 1", *Journal of Aircraft*, Vol. 36, No. 1, pp. 51-60, 1999.

[6] Jameson, A., "Aerodynamic Design via Control Theory", *Journal of Scientific Computing*, Vol. 3, No. 3, pp. 233-260, 1988.

[7] Kim, S., Alonso, J., Jameson, A., "Two-dimensional High-Lift Aerodynamic Optimization Using the Continuous Adjoint Method", AIAA 2000-4741, *8th AIAA / USAF / NASA / ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA, 2000.

[8] Reuther, J. J., Jameson, A., Alonso J., Rimlinger, M. J., "Constrained Multipoint Aerodynamic Shape Optimization Using an Adjoint Formulation and Parallel Computers, Part 2", *Journal of Aircraft*, Vol. 36, No. 1, pp. 61-74, 1999.

[9] Jameson, A., "Re-Engineering the Design Process Through Computation", *Journal of Aircraft*, Vol. 36., No. 1, pp. 36-50, 1999.

[10] Jameson, A., Vassberg, J. C., "Computational Fluid Dynamics for Aerodynamic Design – its Current and Future Impact", AIAA 2001-0538, *39th AIAA Aerospace Sciences Meeting and Exhibit*, Reno, NV, 2001.

[11] Monge, F., Tobio, B., "Aerodynamic Design and Optimization by Means of Control Theory", *in Computational Mechanics – New Trends and Applications*, Barcelona, 1988.

[12] Eliott, J., Peraire, J., "Practical 3D Aerodynamic Design and Optimization Using Unstructured Meshes", *AIAA Journal*, Vol. 35, No. 9., pp. 1479-1485.

[13] Giles, M. B., Pierce, N. A., "An Introduction to the Adjoint Approach to Design", *Flow, Turbulence and Combustion*, 65(3/4):393-415, 2000.

[14] Anderson, K. W., Venkatakrishnan, V., "Aerodynamic Design Optimization on Unstructured Grids Using a Continuous Adjoint Formulation", *Computers and Fluids*, Vol. 28, No. 4-5, pp. 443-480, 1999.

[15] Kim, H-J., Sasaki, D., Obayashi, S., Nakahashi, K, "Aerodynamic Optimization of Supersonic Transport Wing Using Unstructured Adjoint Method", *AIAA Journal*, Vol. 39, No. 6, 2001. pp 1011-1020.

[16] Kim, H-J., Obayashi, S., Nakahashi, K., "Flap-Deflection Optimization for Transonic Cruise Performance Improvement of Supersonic Transport Wing", *Journal of Aircraft*, Vol. 38, No. 4, 2001. pp 709-717.

[17] Iollo, A., Salas, M., Ta'asan S., "Shape Optimization Governed by the Euler Equations Using an Adjoint Method", *Institute for Computer Applications in Science and Engineering*, Report 93-78, 1993.

[18] Newman, J. C. III., Taylor, A. C. III., Barnwell, R. W., Newman, P. A., Hou, G. J.-W., "Overview of Sensitivity Analysis and Shape Optimization for Complex Aerodynamic Configurations", *Journal of Aircraft*, Vol. 36., No. 1, pp. 87-96, 1999.

[19] Jameson, A., Vassberg, J. C., "Studies of Alternative Numerical Optimization Methods Applied to the Brachistochrone Problem", *Computational Fluid Dynamics Journal*, Vol. 9, No. 3., 2000. pp. 281-296.

[20] Obayashi, S., Tsukahara, T., "Comparison of Optimization Algorithms for Aerodynamic Shape Design", *AIAA Journal*, Vol. 35, No. 8, pp. 1413-1415, 1997.

[21] Sasaki, D., Obayashi, S., Kim, H-J., "Evolutionary Algorithm vs. Adjoint Method Applied to SST Shape Optimization", *CFD2001, The Annual Conference of the CFD Society of Canada*.

[22] Alonso, J., Kroo, I. M., Jameson, A., "Advanced Algorithms for Design and optimization of Quiet Supersonic Platforms", *40th Aerospace Sciences Meeting and Exhibit*, Reno, NV, 2002.

[23] Vicini, A., Quagliarella, D., "Airfoil and Wing Design Through Hybrid Optimization Strategies", *AIAA Journal*, Vol. 37., No. 5., pp. 634-641, 1999.

[24] Whitley, D., "Modeling hybrid genetic algorithms" *in Winter G., Cuesta P. (eds): Genetic Algorithms in Engineering and Computer Science*, pp. 191—201, John Wiley, 1995.

[25] Goldberg, D. E., Voessner, S., "Optimizing Global-Local Search Hybrids", *GECCO-99: Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 220-228, 1999.

[26] Sinha, A., Goldberg, D, "Verification and Extension of the Theory of Global-Local Hybrids", *IlliGAL Report* No. 2001010, 2001.

[27] Hacker, K., Eddy, J., Lewis, K., "Tuning a Hybrid Optimization Algorithm by Determining the Modality of the Design Space", *in Proceedings of ASME Design Engineering Technical Conferences and Computers and Information in Engineering*, Pittsburgh, September 9-12, 2001, p.10.

[28] Lobo, F. G., Goldberg, D. E., "Decision making in a hybrid genetic algorithm" *in Proceedings of the IEEE International Conference on Evolutionary Computation*, 1997.

[29] Magyar, G., Johnsson, M., Nevalainen, O., "An Adaptive Hybrid Genetic Algorithm for the Three-Matching Problem", *IEEE Transactions on Evolutionary Computation*, Vol. 4, No. 2, 2000.

[30] Jameson, A., "A Perspective on Computational Algorithms for Aerodynamic Analysis and Design", *Progress in Aerospace Sciences*, No. 37, 2001, pp 197-243.

[31] Samareh, J., "Survey of Shape Parameterization Techniques for High-Fidelity Multidisciplinary Shape

Optimization", *AIAA Journal,* Vol. 39, No. 5, May 2001.

[32] Hooker, J. N., "Testing heuristics: We have it all wrong", *Journal of Heuristics,* No. 1, pp 33-42, 1996.

[33] Kim, S., Alonso, J., Jameson, A., "Design Optimization of High-Lift Configurations Using a Viscous Continuous Adjoint Method", *40th Aerospace Sciences Meeting and Exhibit,* Reno, NV, 2002.

[34] Nemec, M., Zingg, D. W., "Towards Efficient Aerodynamic Shape Optimization Based on the Navier-Stokes Equations", *15th AIAA Computational Fluid Dynamics Conference,* Anaheim, CA, 2001.

[35] Ta'asan, S., "Introduction to Shape Design and Control", *VKI Lecture Series on Inverse Design and Optimisation Methods,* 1997.

[36] Kelley, C. T., "Iterative Methods for Optimization", *Frontiers in Applied Mathematics,* SIAM, 1999.

[37] Herdy, M., "Reproductive Isolation as Strategy Parameter in Hierarchically Organized Evolution Stategies", *Parallel Problem Solving from Nature II,* pp. 207-217, Elsevier, 1992.

[38] Autere, A., "Employing Gradient Information in a Genetic Algorithm", *Second European Congress on Intelligent Techniques and Soft Computing* EUFIT 94, Aachen, 1994.

[39] Schlierkamp-Voosen, D.,Muhlenbein, H., "Strategy Adaptation by Competing Subpopulations", *Proceedings of Parallel Problem Solving From Nature III,* Jerusalem, pp. 199-208, Springer, 1994.

[40] Eiben, A. E., Sprinkhuizen-Kuyper, I. G., Thijssen, B. A., "Competing crossovers in an adaptive GA framework", *Proceedings of the 5th IEEE Conference on Evolutionary Computation*, IEEE Press, pp. 787-792, 1998.

[41] Arian, E., Vatsa, V. N., "A Preconditioning Method for Shape Optimization Governed by the Euler Equations", *ICASE Report* No. 98-14, 1998.

Row nr.	Optimizer parameters			Mean / standard deviation of objective value samples after 200 equivalent evaluations	
	Local sequence length (*SL*)	Global sequence length (*SG*)	Global popsize. (GA)	μ	σ
GLOSSY Mk1					
1	3	3	*4*	4.71	0.24
2	3	3	*5*	4.68	0.28
3	3	3	*6*	4.57	0.17
4	3	3	*7*	4.55	0.17
5	*1*	3	7	4.65	0.17
6	*2*	3	7	4.63	0.17
7	*3*	3	7	4.55	0.17
8	*4*	3	7	4.46	0.06
9	*5*	3	7	4.45	0.10
10	*6*	3	7	4.47	0.14
11	*7*	3	7	4.47	0.22
GLOSSY Mk2 (GA-BFGS)					
12	3	3	4 (initial)	4.56	0.28
pure BFGS (random initial designs)					
13	n/a	n/a	n/a	4.65	0.48
pure GA					
14	n/a	n/a	8	4.67	0.20

Table 1. Mean and standard deviation of objective function value samples collected over 50 runs of GLOSSY Mk1, the adaptive GLOSSY Mk2, the pure BFGS (started from random initial designs) and the pure GA.

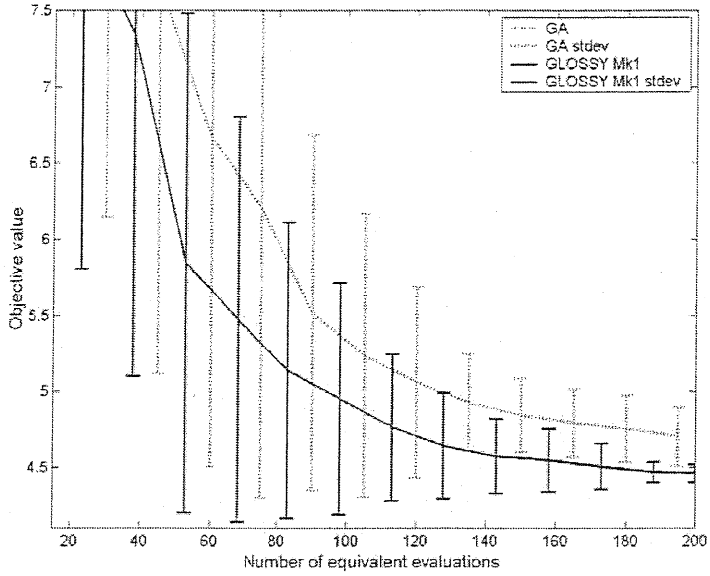

Figure 7. GLOSSY Mk1 and GA optimization histories on the inlet problem. The error bars indicate the sample standard deviations of the objective function values achieved over 50 runs.

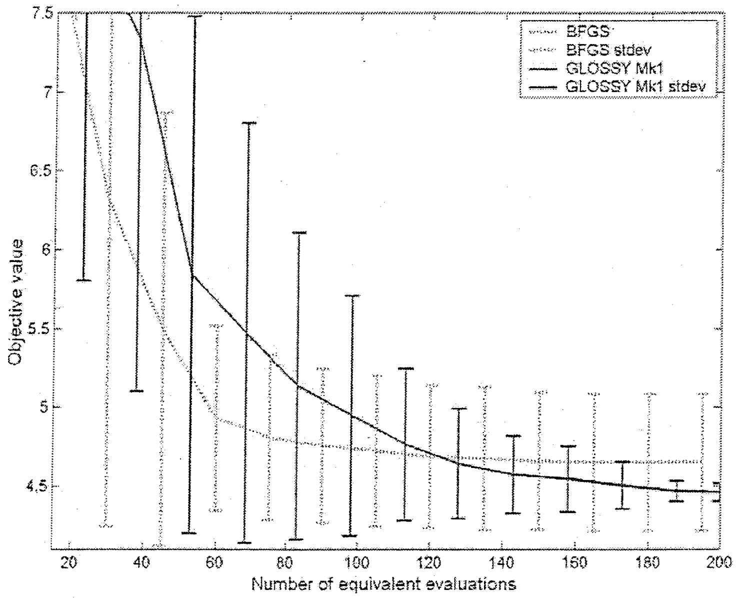

Figure 8. GLOSSY Mk1 and BFGS optimization histories on the inlet problem. The error bars indicate the sample standard deviations of the objective function values achieved over 50 runs.