‘Checkers’ - Highly Efficient Derivative-Free Optimization

Paul Belitz and Thomas Bewley

Flow Control Laboratory, Dept. of MAE, UC San Diego, La Jolla CA 92093 USA

Derivative-free algorithms are frequently required for the optimization of nonsmooth functions defined by physical experiments or by averaging of the statistics of numerical simulations of chaotic systems such as turbulent flows. The core idea of all efficient algorithms for problems of this class is to keep function evaluations far apart until convergence is approached. Generalized Pattern Search (GPS) algorithms, such as the present, accomplish this by coordinating the search with an underlying grid which is refined and coarsened as appropriate. Rather than using the Cartesian grid (the typical choice), the present work introduces for this purpose the use of lattices derived from *n*-dimensional sphere packings (for a comprehensive review of such lattices and their properties see Conway & Sloane 1999). Such lattices are significantly more uniform and have much higher kissing numbers (that is, they have many more nearest neighbors) than their Cartesian counterparts; both of these facts make them much better suited for coordinating GPS algorithms. One of the most efficient subclasses of GPS algorithms, known as the Surrogate Management Framework (SMF; see Booker *et al*, 1999), alternates between an exploratory Search over a surrogate function interpolating all existing function evaluations (and thus summarizing the trends which they represent), and an exhaustive Poll which checks the function on neighboring points to confirm or confute the local optimality of any given Candidate Minimum Point (CMP) on the underlying grid. The present work combines the SMF with efficient lattices based on *n*-dimensional sphere packings, and additionally incorporates one of the highly efficient global search strategies meticulously examined by Jones (2001), thereby developing an extremely efficient lattice-based derivative-free optimization algorithm. Our code implementing this algorithm, dubbed Checkers, compares quite favorably to competing algorithms on a range of well-known optimization test problems.

I. Background

Under the nonsmooth assumption mentioned in the abstract, derivative-based optimization strategies such as Conjugate Gradient and BFGS perform poorly, relegating the class of available algorithms to derivative-free routines. Given an expensive function to minimize, the convergence rate of the selected algorithm is of utmost importance.

Perhaps the simplest grid-based derivative-free optimization algorithm available, identified in this paper as Successive Polling (SP), proceeds as follows. Start with a coarse grid and evaluate the function on a starting point on this grid, defined as the first candidate minimum point (CMP). Then, Poll (that is, evaluate) the function values on gridpoints which neighbor the CMP in parameter space, at a sufficient number of gridpoints to *positively span*[a] the feasible neighborhood of the CMP [this step ensures convergence, as discussed further in Torczon 1997, Booker *et al.* 1999, and Coope & Price 2001]. When polling:

(a) If any poll point is found to have a function value lower than that of the CMP, define this new point as the new CMP and immediately terminate the current Poll step.
(b) If all poll points are found to have function values higher than that of the CMP, refine the grid by a factor of two.

A new poll step is then initiated and the process repeated until terminated. Though the basic SP algorithm described above is not at all efficient, it guarantees eventual convergence to a local minimum, and there exist a wide variety of effective ways to accelerate it. The most efficient subclass of such modified SP algorithms, known as the Surrogate Management Framework (SMF; see Booker *et al*., 1999), leverages inexpensive “surrogate” functions to interpolate the available function evaluations and provide suggested regions of parameter space in which to perform new function evaluations between each Poll step. SMF algorithms thus alternate beween two steps:

[a]That is, such that any feasible point in the neighborhood of the CMP can be reached via a *linear combination with non-negative coefficients* of the vectors from the CMP to the poll points.

(i) Search over the surrogate to identify the most promising gridpoint at which to sample the function, evaluate the function at this gridpoint, update the surrogate function, and repeat. This Search is terminated when it fails to return an improved CMP.
(ii) Poll the neighborhood of the current CMP, following rules (a) and (b) above.

Note the substantial flexibility in the design of the Search algorithm. An efficient Search greatly increases the efficiency of the SMF algorithm, whereas an inefficient Search effectively reduces SMF to SP. As the SP and Search algorithms are effectively independent of one another in the SMF, we analyze them separately in the discussion that follows.

At the heart of the SMF algorithm lies the n-dimensional discretizing grid or 'lattice' that all function evaluations are restricted to and from which each Poll set is selected. To the best of the authors' knowledge, all previous work using such grid-based optimization strategies have utilized Cartesian grids. From n-dimensional sphere packing theory, a wide variety of alternative lattices are available (Conway & Sloane 1999). All of these alternative lattices are significantly more uniform than the Cartesian grid, as measured by common performance metrics such as packing density, covering thickness, and an appropriately normalized measure of average quantization error. Thus, the use of an alternative lattice offers potential increases in the efficiency of SP-based algorithms.

An even more significant disadvantage of the Cartesian approach to the coordination of a derivative-free optimization code is the relatively poor placement of the nearest-neighbor grid points. At each SP step, the poll points must be selected to form a positive basis around the CMP. In the interest of efficiency, minimizing the number of poll points at each step is of key importance. As explained in detail in Belitz *et al*. (2008), and summarized briefly below, the Cartesian grid induces a very nonuniform Poll set in the SP algorithm, and thus the use of alternative lattices is of great interest in the SP and SMF algorithms.

II. Successive Polling Test Results

To test the effect of the underlying lattice on the poll step alone, several simulations were run on known test functions $J(\mathbf{x})$ to gather statistical data on the performance of an A_n-based SP code as compared to an otherwise identical Cartesian-based SP code. The initial test functions consisted of positive definite quadratic bowls with random condition number, random minimum location, and random algorithm start point. The initial grid spacings of the Cartesian-based and A_n-based algorithms were selected to provide a constant initial average step length. The number of function evaluations necessary to reach a given level of convergence were recorded for a large number of tests to provide statistically representative comparisons of algorithm efficiency. The SP search above was run for $n = 2, 3, 4,$ and 5. A_n outperformed Cartesian in 81% to 100% of the runs, with the efficiency difference increasing significantly as the dimension n of the problem was increased.

These results show clearly that the A_n-based SP algorithm significantly outperforms the Cartesian-based SP algorithm. The mechanism by which the Cartesian-based algorithm performs so poorly is worth examining in detail. The lack of uniformity of the Cartesian basis decreases the likelyhood of the steepest-descent direction being well approximated by the basis vectors. The more uniform A_n search is more likely to have a vector more closely approximating the true gradient, thereby converging at a higher rate. Further, as the dimension of the problem increases, not only does A_n tend to converge in as many or fewer steps than Cartesian, but the amount by which A_n improves on Cartesian increases. In $n = 5$ we frequently realize an order of magnitude difference in the efficiencies of the two approaches.

These results offer solid evidence that, as predicted by their significantly improved uniformity and distribution, the convergence rate of Generalized Pattern Search (GPS) algorithms can be improved greatly by implementing an efficient lattice to discretize the parameter space.

III. Kriging Interpolating Surrogate Functions

The purpose of the Search in the SMF is to identify, based on the trends evident in the current set of function evaluations, as well as the "voids" of information existing in this set, the most promising areas where the function value might be lower than the current CMP. For maximum performance, one would obviously like to find the most efficient Search algorithm possible. To model the function as effectively as possible, the present algorithm leverages Kriging interpolants.

Often described as "modelling a function as a realization of a stochastic process", Simple Kriging interpolation provides a Predictor $\hat{J}(\mathbf{x})$ and a Predictor Uncertainty $\sigma(\mathbf{x})$. The Predictor is that function which maximizes the likelyhood of the observed data (see Jones, 2001). The Predictor Uncertainty is modeled based on the distance to the nearby function evaluations and the correlation (or lack thereof) of their corresponding values. The Predictor matches the function value at each sampled point, and the Predictor Uncertainty goes to zero at each sampled point. The derivation of Simple Kriging as well as the details of the numerical implementation can be found in Belitz & Bewley (2008).

IV. Building the Surrogate Search

Having a Predictor and its corresponding Uncertainty from the Kriging model, a sophisticated Search can now be built. One obvious strategy commonly used in such scenarios consists of simply evaluating the Predictor at its minimum, discretized onto the underlying grid. This simple, intuitive approach has been implemented in a variety of examples with reasonably good results. However, such a strategy offers no clear way to "explore" the function far from the Predictor minimum, and the resulting Search tends to be local.

To increase the exploratory quality of the Search, schemes such as that implemented by Marsden (2005) can be used, in which the Search is defined by a Poll step performed around the surrogate minimum. Another option explored by Booker *et al*, (1999) considered evaluating the function at a number of points in parameter space, some chosen for minimum surrogate value, others for maximum surrogate uncertainty. Such a heuristic provides a measure of global convergence behavior, as the 'space-filling' maximum uncertainty points allow the surrogate to explore and accurately model the function relatively far from the surrogate minimum.

The efficacy of such Predictor-based Search strategies can be evaluated with a one-dimensional example (see Figure 1). In this case we take a simple function of a scalar argument x with multiple minima: $J(x) = \sin(x) + x^2$ on the interval $[-5, 5]$. Two initial points are used as the starting points for the Search. The function is sampled at the point returned by the search; the surrogate is updated, and the search is repeated.

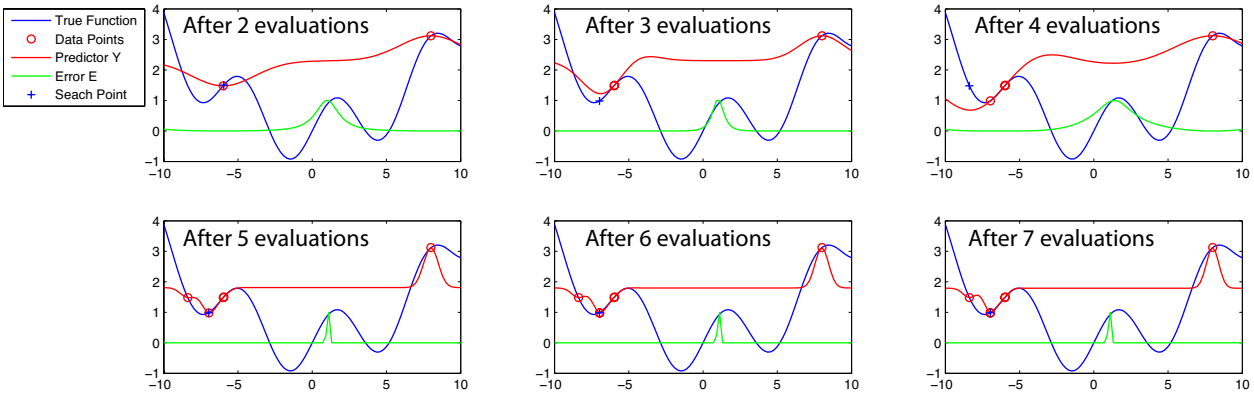

Figure 1. Search based on $J(x)$ above. The algorithm does not necessarily converge even to a local minimum, and stalls after just six function evaluations.

In Figure 1, sampling at the surrogate minimum is explored. This Search stalls quite quickly - in the case shown, after only five function evaluations. Even more notably, the algorithm does not even converge to a local minimum! While in an SMF scenario the grid can prevent such early stalling by keeping function evaluations far apart, this simple example illustrates that this search strategy in its purest form is clearly lacking. Due to this easily demonstrable inadequacy, we conclude that more effective Search strategies are worth exploring. In particular, a search algorithm that drives the Search to explore *near*, not *at*, the Predictor minimum seems likely to greatly improve the convergence of the Search algorithm.

A more promising idea includes the Predictor Uncertainty in the search algorithm. By minimizing $\hat{J}(\mathbf{x}) - l\sigma(\mathbf{x})$, where l is an adjustable scalar, the Search will return points where the Predictor is low and the Uncertainty is high (Cox & John, 1997). Thus, the Search is necessarily driven away from previously evaluated points, thereby avoiding the fundamental problem with the simpler Search described above. This strategy provides natural flexibility in the exploratory nature of the Search, and appears to be a very effective algorithm. The scalar l governs the locality of the Search - the greater l, the more "exploratory" a search pattern is generated (that is, the more the search simply fills the

voids in the existing set of data). As $l \to \infty$, the Search becomes dense in parameter space as the number of evaluations is increased, which therefore satisfies the requirement for guaranteed convergence of such an algorithm to the global minimum as estabilished by Torn & Zilinskas (1992).

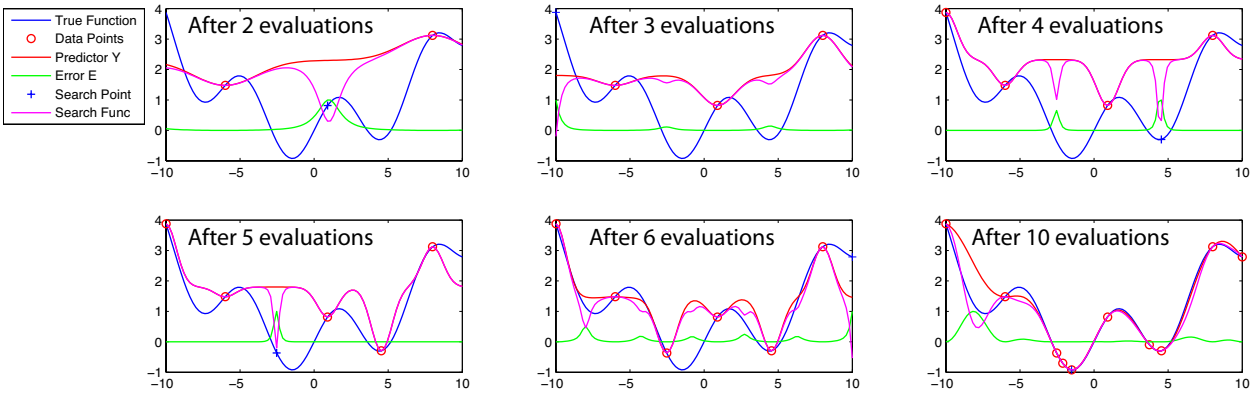

Figure 2. Search Point based on $\hat{f}(x) - l\sigma(x)$. The algorithm rapidly explores the function and quickly locates the global minimum.

As demonstrated in Figure 2, this search (with an appropriate value of l) provides very good performance, capturing the behavior of the test function very well after a relatively small number of function evaluations. Ever more sophisticated search strategies have been explored (see Jones, 2001); however, the performance advantages of such algorithms over the simpler strategy demonstrated here do not appear to be particularly compelling.

Numerically, implementing this Search leads to some challenges. As $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ has multiple minima, developing an efficient algorithm to perform the minimization of $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ for a given l is necessary. Noting that the search function $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ is itself quite smooth, a derivative-based approach to minimizing $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ was devised. As the Uncertainty is zero at each sampled point, the minima of $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ will generally lie between the sampled points. Thus, a derivative-based search starting very near but not quite at each sampled point is used. This search moves in a variety of directions from each sampled point and reliably converges into *all* local minima of $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$.

V. An A_n-Based Surrogate Management Framework

To recap: the SMF algorithm consists of performing Searches until the Search fails, upon which a Poll step is performed. If the Poll is successful, the algorithm returns to the Search; else, the grid size is refined by a factor of two, and a new Search is performed. In the current study, we test the effect of changing the underlying grid coordinating this algorithm from Cartesian to A_n. In §II we showed that the use of a highly uniform lattice greatly improves the efficiency of the Poll step which forms an important component of the full SMF algorithm. We now show that such lattices also greatly improve the efficiency of the full SMF algorithm.

As described above, the Search step of the SMF algorithm is performed by minimizing the function $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$, where $\hat{f}(\mathbf{x})$ is the predicted value of the function, l is a chosen scalar, and $\sigma(\mathbf{x})$ is the Kriging Uncertainty. In the interest of robust performance, the chosen method of minimizing $\hat{f}(\mathbf{x}) - l\sigma(\mathbf{x})$ for multiple l offers excellent performance, good customization properties for nonlocal optimization, and low complexity.

The key features of our lattice-based SMF algorithm are now completely described. Our code implementing this algorithm, dubbed "Checkers", aims to be a maximally efficient derivative-free optimzation routine for locating local minima, while providing a customizable degree of "exploratory" emphasis, via the selection of l, for global optimization. Checkers has been tested in this work for $n = 2$ to $n = 6$ dimensions. The test functions chosen here consist of the n-dimensional Rosenbrock function (a local minimization problem), and the Branin test function (a global optimization problem). The objective of these tests were to quantify the convergence rate of an A_n SMF minimization as compared to its Cartesian equivalent, as well as to attempt to compare the performance of Checkers to various other popular derivative-free optimization schemes.

VI. Results

The Rosenbrock test function is uniquely difficult for the SMF to minimize, as seen in Figure 3. The "valley" in which the true minimum lies is narrow, curved, and characterized by a vanishing second derivative on its floor, making it extremely difficult for any search algorithm to approximate the gradient sufficiently accurately to move towards the minimum. Thus, Rosenbrock is an excellent test for the efficiency of an algorithm such as the SMF. Note that Rosenbrock is a convex function with only one minimum, and is defined as

$$J(\mathbf{x}) = \sum_{i=0}^{n-1} [(1-x_i)^2 + (-1)^n 5(x_{i+1} - x_i^2)^2]$$

This function was further modified to shift the global minimum to a random point to avoid biasing the test. Two SMF schemes were run on this function: both utilizing the $\hat{J}(\mathbf{x}) - l\sigma(\mathbf{x})$ search. The sole difference between these two versions of SMF is the underlying lattices (A_n versus Cartesian) coordinating the optimization. As in the SP tests described earlier, the average poll vector length was normalized between the Cartesian and A_n versions of the SMF.

To evaluate the efficiency of these algorithms, the Cartesian version of the algorithm was run for 200 to 300 function evaluations. Then, the A_n code was run until the minimum function value returned was comparable to that returned by the Cartesian code. The number of function evaluations required to converge to the level found by both thus gives a metric by which to compare the efficiency of the two algorithms.

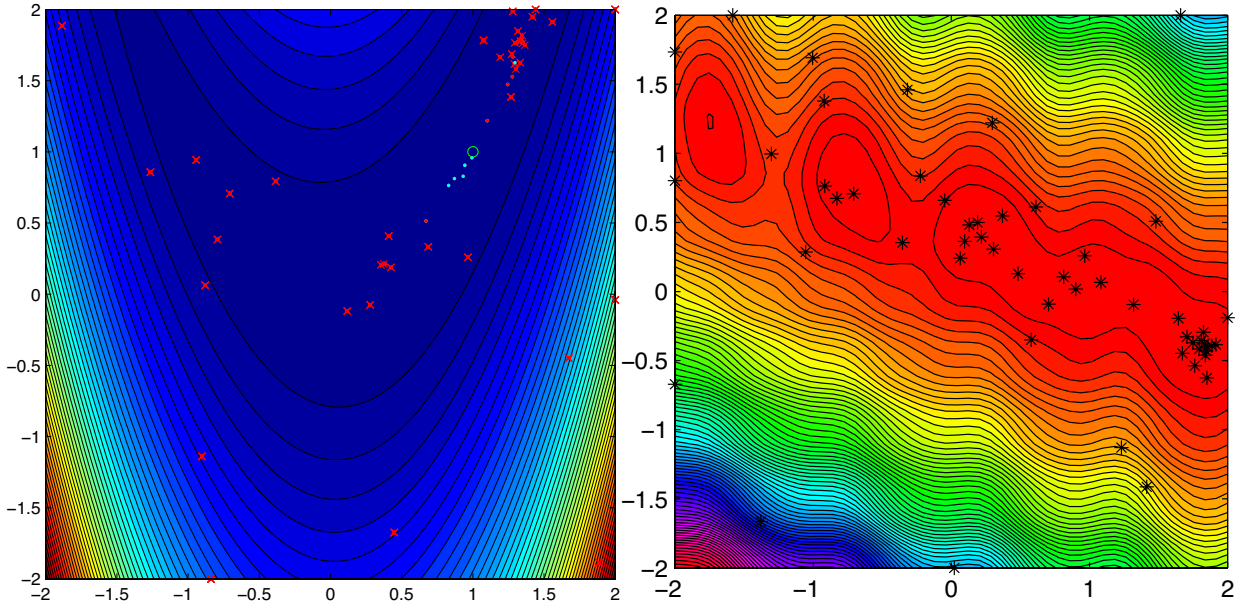

Figure 3. Optimization of the Rosenbrock (left) and Branin (right) test functions. Note how the Search accelerates the convergence to the minimum (green circle) on the Rosenbrock function, returning points (green) that converge at an increased rate compared to the SP Poll (red crosses). Also note the Search algorithm's nonlocal optimization properties on the Branin function - the Search locates the region of the global minimum; the Poll ensures convergence once said region is located.

The two versions of SMF were run on the modified Rosenbrock function described above in $n = 2, 3, 4, 5,$ and 6 dimensions. In these tests A_n outperformed Cartesian from 73% to 90% of the runs, indicating a very substantial improvement realized by changing nothing other than the lattice underlying the optimization algorithm.
These results show a clear conclusion: efficient lattices provide a significant advantage over Cartesian grids in SP-based numerical optimization algorithms. In the SMF code Checkers, the difference in efficiencies even with a sophisticated Search algorithm employed regularly result in a up to a factor of 2 increase in algorithm efficiency; that is, convergence to a given level routinely requires half as many function evaluations just by coordinating the algorithm with the more uniform lattice A_n.

Thus far, only results on local minimization have been discused. As Checkers' Search has the capacity to provide non-local optimization, preliminary testing was performed on a sample function with multiple minima. The Branin test function was selected for this preliminary testing. The Branin test function is defined as

$$J(\mathbf{x}) = (1 - 2x_2 + 0.05\sin(4\pi x_2) - x_1)^2 + (x_2 - 0.5\sin(2\pi x_1))^2$$

On the interval $-2 < x_1 < 2, -2 < x_2 < 2$, the Branin Test Function has five local minima. As can be seen in Figure 3, with the Search parameter $l = 50$, Checkers does an excellent job of exploring all the function minima, eventually converging to the true minimum. At l is increased, the Search becomes more 'space-filling', driving the maximum uncertainty of the Kriging surrogate towards zero. As the number of function evaluations is increased, the points evaluated by this search algorithm become dense in parameter space. The Poll drives the convergence of the algorithm to the global minimum once the viscinity of the global minimum has been identified by the Search.

While these preliminary results indicate the non-local aspect of the Search behavior, additional testing is warranted. Further algorithm refinement to provide the best combination of local and nonlocal Search behavior remains to be performed; however, these initail results clearly demonstrate the outstanding capabilities of the Checkers SMF code.

VII. Conclusion

The results from the SP algorithm on quadratic bowl tests show a clear and substantial advantage when utilizing a lattice-based Poll step in Successive Polling-based algorithms. Simply put, switching from a Cartesian grid to a lattice-based SP significantly improves the algorithm's convergence rate. Testing on other functions substantiates these results. There is no doubt remaining that lattice-based SP searches are the preferred choice over Cartesian-based algorithms. Independent of the convergence efficiency of Checkers on local optimization problems, the rigorous Search implemented in Checkers has demonstrated efficient global convergence of the algorithm in non-convex optimization problems. With appropriate tuning, the algorithm proves to do an excellent job locating a global minimum on standard test problems. Further work remains to be done in this area, but all testing indicates Checkers to be a efficient, robust code for a wide variety of optimization applications.

Acknowledgments

The authors thank Sebastien Michelin, Haoxiang Luo, and Alison Marsden for helpful discussions related to this work.

References

[1]Conway, JH, & Sloane, NJA (1999) *Sphere Packings, Lattices, and Groups*, Springer.

[2]Booker, A, Dennis, JR, Frank, P, Serafini, D, Torczon, V, & Trosset, M (1999) A rigorous framework for optimization of expensive functions by surrogates. *Structural and Multidisciplinary Optimization* **17**, 113.

[3]Jones, DR (2001) A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of Global Optimization* **21**, 345-383.

[4]Belitz, P, & Bewley T (2008) "Checkers", an efficient lattice-based algorithm for derivative-free optimization. *SIAM Optimization*, submitted.

[5]Marsden, AL, Wang, M, Dennis, JE, & Moin, P (2004) Optimal aeroacoustic shape design using the surrogate management framework. *Optimization and Engineering* **5** (2), 235-262.

[6]Coope, ID, & Price, CJ (2001) On the convergence of grid-based methods for unconstrained optimization. *SIAM J. Optim.*, **11**, 859869.

[7]Torczon, V (1997) On the convergence of pattern search algorithms. *SIAM J. Optim.*, **7**, 1-25.