J Glob Optim (2007) 39:39–48
DOI 10.1007/s10898-006-9110-4

ORIGINAL PAPER

Multi-funnel optimization using Gaussian underestimation

Roummel F. Marcia · Julie C. Mitchell · J. Ben Rosen

Received: 16 March 2006 / Accepted: 21 October 2006 / Published online: 24 November 2006

Abstract In several applications, underestimation of functions has proven to be a helpful tool for global optimization. In protein–ligand docking problems as well as in protein structure prediction, single convex quadratic underestimators have been used to approximate the location of the global minimum point. While this approach has been successful for basin-shaped functions, it is not suitable for energy functions with more than one distinct local minimum with a large magnitude. Such functions may contain several basin-shaped components and, thus, cannot be underfitted by a single convex underestimator. In this paper, we propose using an underestimator composed of several negative Gaussian functions. Such an underestimator can be computed by solving a nonlinear programming problem, which minimizes the error between the data points and the underestimator in the L^1 norm. Numerical results for simulated and actual docking energy functions are presented.

Keywords Gaussian functions · Underestimators · Nonlinear programming · Protein docking

1 Introduction

The problem of estimating the location of the global minimum point of a basin-shaped energy function $f(x): \Re^n \to \Re$, with a very large number of local minima, is important in a number of computational biology applications, including protein–ligand docking and the prediction of protein structure from sequence. This problem has been

R. F. Marcia (✉)· J. C. Mitchell
Departments of Biochemistry and Mathematics, University of Wisconsin-Madison, 433 Babcock Drive, Madison, WI 53706, USA
e-mail: marcia@math.wisc.edu

J. Ben Rosen
Department of Computer Science and Engineering, University of California, San Diego, La Jolla, CA 92037, USA

considered in several earlier papers, where a set of $m \gg n$, local minima is computed in an initial large search domain $\mathcal{D}$ in $\Re^n$, and these data points are approximated by a strictly convex quadratic function $q(x,p)$ which underestimates all of them and minimizes the error in the L^1 norm [3,6,7,9,12,13]. In $q(x,p)$, the variable $p \in \Re^s$, with $s \leq m$, represents the parameters which are determined so as to minimize the L^1 distance between $q(x,p)$ and $f(x)$ at each data point. The unique minimum point $x_{\text{pred}} \in \mathcal{D}$ of $q(x,p)$ is then a good prediction for the global minimum x_{gmin} of the true energy function $f(x)$. A more detailed search is then carried out in a greatly reduced domain with x_{pred} at its center, to determine the actual global minimum of $f(x)$. The convex quadratic approximation has been successfully applied to a realistic docking energy function, as well as a large number of simulated docking energy test functions [8].

However, there are situations when an approximation by a single convex function gives a poor prediction of the global minimum of the true energy function $f(x)$. This occurs when $f(x)$ is not basin-shaped, but contains two, or more, local minima with large magnitudes at distinct locations. In this situation the predicted global minimum will be located at some point between the two large magnitude local minima, rather than at the correct global minimum point. For this situation, a much better approach is to use an underestimating function $g(x,p)$ consisting of the sum of l negative Gaussian functions, in effect, clustering the data points into distinct energy wells (see Fig. 1). Specifically we define

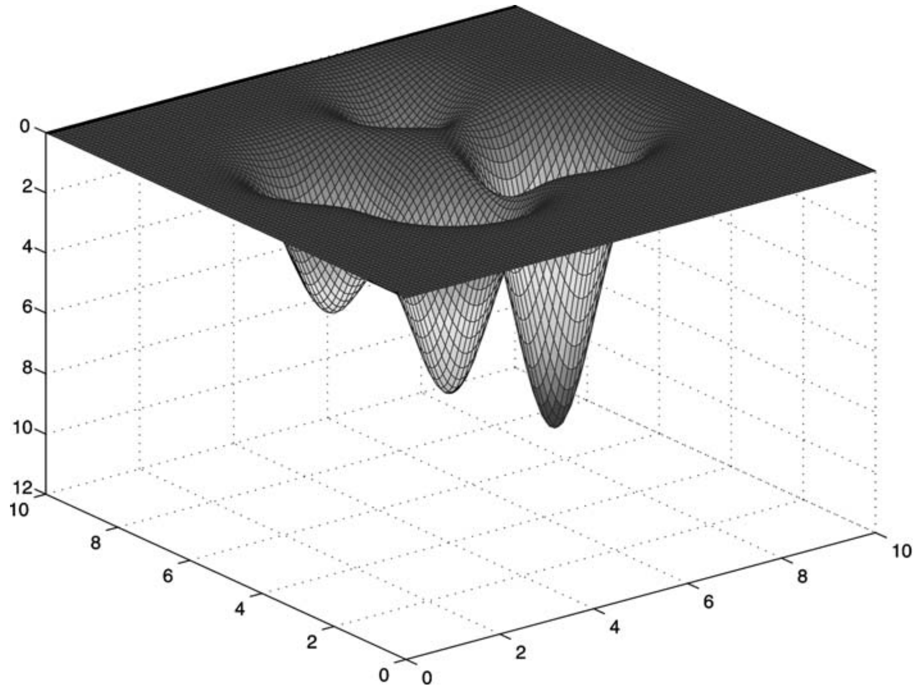

Fig. 1 A sum of 2D negative Gaussian functions with $l = 4$

$$g(x,p) \equiv \alpha + c^T x - \sum_{j=1}^{l} \beta_j e^{-\rho_j \|x-z_j\|^2}, \quad \beta_j \geq 0,\ \rho_j > 0, \tag{1}$$

where p represents the s parameters $(\alpha, c, \beta_j, \rho_j, z_j)$, to be determined. If $\beta_k > \beta_j$, $j = 1,2,\ldots,l$, with $k \neq j$, then the approximate global minimum of $g(x,p)$ will occur at $x = z_k$. The number of negative Gaussian functions, l, is typically between two and five and can be estimated by considering how well the lowest data points are clustered. For functions with more than five strong local minima that are well separated, l must be increased accordingly. From experience in our application, four proved to be sufficient. Note that the function $g(x,p)$ consists of radial basis functions (negative Gaussian functions, in particular), which have been previously used in global optimization (e.g., [2,5,10]). Other radial basis functions include thin-plate splines and multiquadratics, for example. However, whereas these functions are used to interpolate data points in global optimization, we use the Gaussian underestimator to *underfit* these points. Underestimating such points ensures that the minimizer in each Gaussian underestimator lies below each point in the Gaussian function, making the minimizer a plausible predictor of the global minimum.

An algorithm based on Gaussian Underestimation has been developed and implemented. We call this the GU algorithm. As in other underestimation methods, the parameters $(\alpha, c, \beta_j, \rho_j, z_j)$ for defining the Gaussian underestimator are obtained by solving an optimization problem that minimizes the L^1 distance between the data points $f(x^{(k)})$ and the underestimator $g(p;x^{(k)})$ subject to $g(p;x)$ underfitting all the data points. We describe the GU algorithm and its computational implementation and testing in this paper, which is organized as follows. In Sect. 2, we formulate GU as a two-phase optimization program, where the solution to a linear optimization problem is used as an initial estimate for the solution in the general nonlinear program formulation in the second phase. We describe in Sect. 3 the problem type to which we will apply the GU algorithm. We present numerical results in Sect. 4 and conclude with some closing remarks in Sect. 5.

2 Formulation and solution as nonlinear program

In order to determine the parameters p in $g(x,p)$ we formulate the problem as a nonlinear programming problem which minimizes the error at the m data points in the L^1 norm, and requires that $g(x,p)$ underestimates $f(x)$ at every data point x_i, $i = 1,2,\ldots,m$ (see Fig. 2). The number of such data points is usually $m = 4s$, where $s = (n+2)(l+1) - 1$ is the number of parameters necessary to define the Gaussian underestimator, and n and l are the dimension of x and the number of Gaussian functions, respectively (see Eq. 1). Although this many data points are sufficient to determine a Gaussian underestimator, it would be preferable to compute significantly more points for the Gaussian underestimator to be a more accurate approximation of the energy landscape and, therefore, a better predictor of the global minima. Equally important is the quality of these data points. Coarse uniform sampling over the domain space can be performed to obtain these points, but the corresponding Gaussian underestimator might not necessarily capture the "topographic" behavior of the function in regions of low energy conformations and deeper basins containing better optima might be missed. Ideally, these initial data

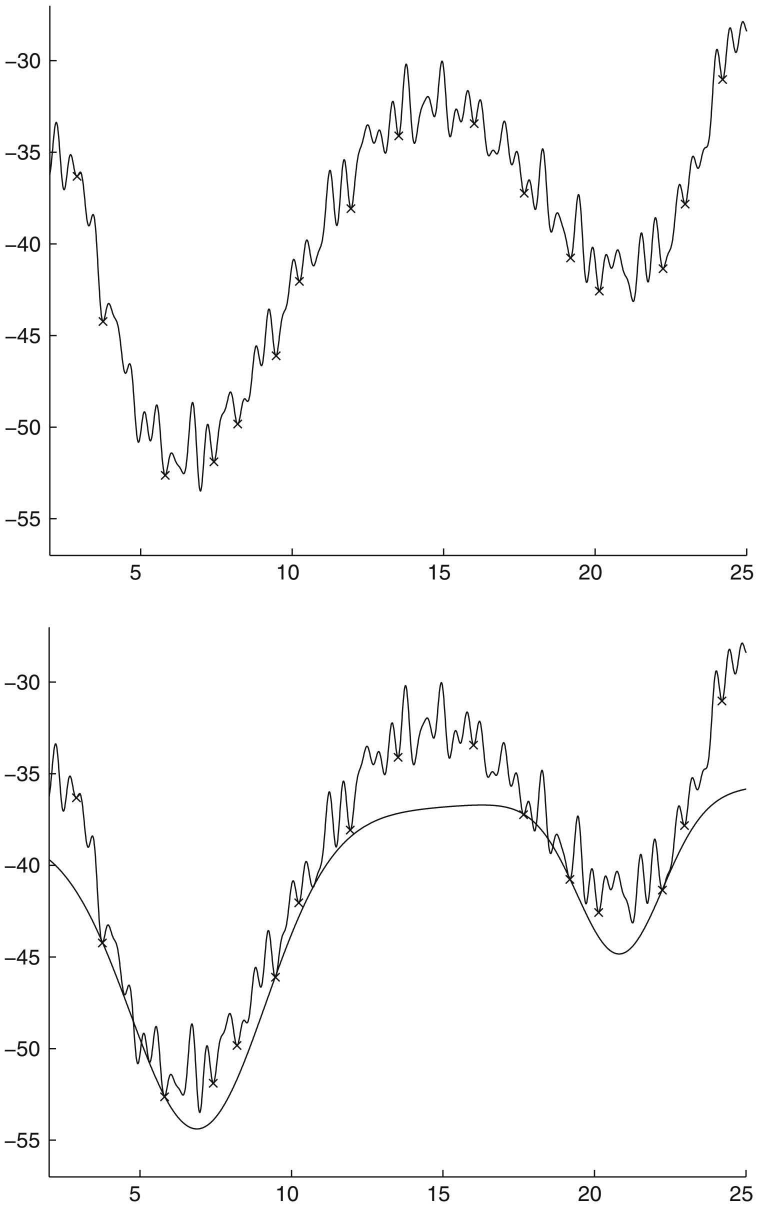

Fig. 2 Gaussian underestimation in 1D

points are local minima with low energy values. In our application, the potential energy function is sufficiently smooth and differentiable so that local optimization is viable.

Let $y_i = f(x_i)$. Then we require $g(x_i, p) \le y_i$, $i = 1, 2, \ldots, m$. The formulation is then given by:

$$\begin{array}{lll} \underset{p \in \Re^s}{\text{minimize}} & G(p) \equiv \sum_{i=1}^{m} y_i - g(x_i,p), \quad g(x,p) \text{ given by (1)} & \\ \text{subject to} & y_i - g(x_i,p) \geq 0, \quad i = 1,2,\ldots,m & \text{(GU)} \\ & \text{All parameters in } p \text{ bounded.} & \end{array}$$

This nonlinear programming problem (GU) is a key part of the GU algorithm. Note that both the objective function $G(p)$ and the constraints in (GU) are linear in the parameters α, c, and β_j, and nonlinear in ρ_j and z_j. This formulation is similar to that used in [13], except that $g(x,p)$ is nonlinear in some of the parameters, whereas the convex quadratic underestimator $q(x,p)$ in [13] is linear in all the parameters.

Because the vectors z_j occur in $g(x,p)$ nonlinearly, it is possible to obtain solutions to (GU) which do not give the smallest possible L^1 error. That is, we may get a local minimum of the error rather than the desired global minimum. This will depend primarily on the initial values chosen for the z_j at the start of the optimization. Since z_k represents the approximate minimum location of the kth Gaussian function, $k = 1,2,\ldots,l$, we would like to start the solution of (GU) with a reasonable estimate of these locations in the initial search domain $\mathcal{D}$. Fortunately, such an estimate is usually known from a coarse grid constructed on $\mathcal{D}$, and an evaluation of $f(x)$ at each coarse grid point. This information is used to specify both the initial values of z_j, and also a lower and upper bound on each element of each z_j. This coarse grid evaluation will also permit an initial value, and bounds, on each of the parameters ρ_j, $j = 1,2,\ldots,l$.

The solution to (GU) will give optimal values $\hat{p}$ of the parameter vector p, including $\hat{z}_j$ and $\hat{\rho}_j, j = 1,2,\ldots,l$. One of the $\hat{z}_j$, say $\hat{z}_k$, is then a good predictor for the location of the global minimum of the true energy function $f(x)$. This $\hat{z}_k$ is such that $g(\hat{z}_k,\hat{p})$ is its predicted value, and

$$g(\hat{z}_k,\hat{p}) \leq g(\hat{z}_j,\hat{p}), \quad j = 1,2,\ldots,l.$$

We then do a final search in a much reduced search domain, with its center at $\hat{z}_k$. This final search uses a relatively large number of local minimizations in the reduced search domain to determine the location of x_{gmin}, the global minimum point, and its value $f(x_{\text{gmin}})$. In some cases it is possible that one of the other locations $\hat{z}_j$ gives a function value $g(\hat{z}_j,\hat{p})$ close that to that of $g(\hat{z}_k,\hat{p})$. If that occurs, then a similar detailed search is also done in the neighborhood of this $\hat{z}_j$, and the point with the smallest value of $f(x)$ is then chosen as the global minimum point.

3 Docking energy functions

The true protein–ligand docking energy surface is represented by a function $f(x)$, $x \in \mathcal{D} \subset \Re^n$, which is assumed to have a very large number (possibly exponential in n) of local minima. Typically the computation of the energy surface value at a single point x, may require several minutes. Furthermore, many local minima are needed to insure that the underestimating function gives a good prediction of the global minimum. This means that m (the number of local minimum data points) may be large, and may increase exponentially with n. The number of minimum will typically be $O(n^2)$, so that using the true energy surface for test problems is not practical. Instead, we follow the approach presented in [6,7] which simulates the energy landscape commonly found in protein docking problems. The majority of results presented here use this simulated energy function. However, we also present results of the GU algorithm to a potential energy function used in an actual docking software called Docking Mesh Evaluator [8].

3.1 Simulated energy functions

In the prediction of the global minimum for a real docking energy surface we typically compute a relatively large number m of local minima of $f(x)$, to obtain $f(x_i)$, $i = 1, 2, \dots, m$. We then compute the underestimating function $g(x, p)$ as described earlier. The only information used by the GU algorithm in approximating the real docking energy surface $f(x)$, consists of the function values $f(x_i)$, $i = 1, 2, \dots, m$, at the m local minimum points x_i. For a real docking energy surface the computation of the local minima is by far the most time consuming part of the calculation. In our simulated test problems we replace this slow calculations as follows.

First we generate the sum of negative Gaussian functions (1), choosing l (the number of functions) and the parameters $p = \bar{p}$, to represent the general surface of the true $f(x)$. We then simulate lm local minima by first generating (usually randomly) the points x_i, $i = 1, 2, \dots, m$, with function values $g(x_i, \bar{p})$, $i = 1, 2, \dots, m$. We then perturb these values by a random perturbation η_i, $-\gamma \le \eta_i \le \gamma$, for selected values of γ, and define

$$y_i = g(x_i, \bar{p}) + \eta_i, \quad i = 1, 2, \dots, m. \tag{2}$$

We define a simulated energy surface $Sf(x)$, as the function $g(x, \bar{p})$, with m known local minima y_i, as given by Eq. 2. The set of points (x_i, y_i) is the only information we use in (GU) to determine the underestimator. By an appropriate choice of the parameter vector $\bar{p}$, we can ensure that one of the $\bar{z}_j$, (say $\bar{z}_1$), is the global minimum point, $x_{\text{gmin}} = \bar{z}_1$, of $g(x, \bar{p})$. This is done by choosing $\beta > \beta_j$, $j = 2, 3, \dots, l$, and $\rho_j \ge \rho_{\min} > 0$, for some appropriate value of $\rho_{\min}$. We then assume that x_{gmin} is the global minimum point of the simulated energy surface $Sf(x)$. In order to make the test using $Sf(x)$ as close as possible to the situation with a true docking energy surface, we eliminate any random point x_i which is close to $\bar{z}_1$. This makes $Sf(x)$ a better representation of $f(x)$, since it is highly unlikely that any random point used when applying (GU) to a true energy function $f(x)$ will be very close to its true global minimum point. Each point x_i, with function value y_i, is assumed to represent a local minimum of $f(x)$. Since y_i is very easily computed, these simulated local minima of $f(x_i)$ make it much faster to test the algorithm, than using $f(x)$ would be.

Given the (x_i, y_i) values we apply (GU) exactly as we would with a true docking energy surface $f(x)$, using the data points $(x_i, f(x_i))$. The GU algorithm has been tested on problems in $\Re^2$ and $\Re^6$. In both cases we used the sum of four negative Gaussians, so that $l = 4$. In each case the initial search domain $\mathcal{D}$, was a hypercube with edge length equal to 10, so that its volume was 10^n.

In order to evaluate the computation test results obtained with the simulated docking energy function, we need some measure of how well the underestimating function $g(x, \hat{p})$ predicts x_{gmin}, the global minimum of $Sf(x)$. The GU algorithm determines $\hat{p}$, so we know the point $\hat{z}_1$. This will typically be very close to the global minimum point of $g(x, \hat{p})$. We let $x_{\text{pred}} = \hat{z}_1$, and call x_{pred} the predicted global minimum of $Sf(x)$. The algorithm is therefore successful if $\|x_{\text{pred}} - x_{\text{gmin}}\|_\infty$ is sufficiently small.

Given x_{pred}, we construct the smallest hypercube in $\Re^n$, with x_{pred} at its center, that contains x_{gmin}. This hypercube will have an edge length of d_2, where $d_2 = 2\|x_{\text{pred}} - x_{\text{gmin}}\|_\infty$. The initial search domain $\mathcal{D}$, known to contain x_{gmin}, will be a hypercube of edge length d_1, with a volume d_1^n. The new hypercube, given by x_{pred} and d_2, will have a volume of d_2^n. The ratio of these two volumes is therefore $(d_2/d_1)^n$. If the GU algorithm is successful, we will have reduced the search volume by the volume reduction ratio:

$$\text{VRR} \equiv \left(\frac{d_2}{d_1}\right)^n \ll 1. \tag{3}$$

The VRR is used to evaluate the success of the GU algorithm as shown in Tables 1–3 in the next section.

3.2 Docking Mesh Evaluator

The Docking Mesh Evaluator (DoME) is software for predicting the active site of a protein to an ion, ligand, DNA, and other macromolecules upon binding. It describes the molecular interaction by defining a potential energy function whose global

Table 1 Results for a simulated energy function

	$\gamma = 0.0$	$\gamma = 0.1$	$\gamma = 0.2$	$\gamma = 0.4$	$\gamma = 0.8$	$\gamma = 1.6$
$G(p^{(0)})$	106.1	112.3	122.9	136.0	186.7	285.4
$G(p_f^*)$	0.0	5.7	11.9	24.7	53.5	128.7
`LP NPSOL Iter`	1	2	1	2	2	3
`NLP NPSOL Iter`	1	4	4	5	10	11
$\|\|\hat{z}_1 - \bar{z}_1\|\|_2$	0.0	0.053	0.144	0.175	0.356	0.841
VRR	0.00e-00	1.28e − 13	4.98e − 11	5.99e − 10	2.52e − 08	1.04e − 05

Table 2 Results for a simulated energy function

	$\gamma = 0.0$	$\gamma = 0.1$	$\gamma = 0.2$	$\gamma = 0.4$	$\gamma = 0.8$	$\gamma = 1.6$
$G(p^{(0)})$	107.9	117.2	119.9	135.2	183.5	276.5
$G(p_f^*)$	0.0	6.6	11.9	27.1	49.9	107.0
`LP NPSOL Iter`	2	2	2	3	1	1
`NLP NPSOL Iter`	1	7	6	7	5	8
$\|\|\hat{z}_1 - \bar{z}_1\|\|_2$	0.000	0.097	0.113	0.203	0.347	0.951
VRR	0.00e − 00	6.10e − 12	1.21e − 11	6.26e − 10	1.24e − 08	1.05e − 05

Table 3 Results for a simulated energy function

	$\gamma = 0.0$	$\gamma = 0.1$	$\gamma = 0.2$	$\gamma = 0.4$	$\gamma = 0.8$	$\gamma = 1.6$
$G(p^{(0)})$	107.2	117.8	112.4	138.4	181.8	262.5
$G(p_f^*)$	0.0	6.6	13.4	24.7	55.3	116.5
`LP NPSOL Iter`	1	2	2	3	4	2
`NLP NPSOL Iter`	1	4	6	11	8	16
$\|\|\hat{z}_1 - \bar{z}_1\|\|_2$	0.000	0.046	0.097	0.291	0.395	0.766
VRR	0.00e − 00	9.12e − 14	2.20e − 12	5.20e − 09	8.32e − 08	2.02e − 06

minimum corresponds to the known docked configuration. This energy model takes into account both long-range forces (electrostatic potential) and short-range forces (hydrogen bonds, desolvation energy, and van der Waals interactions (dipole moments and steric repulsion)). DoME uses an exhaustive scan for favorable configurations, treating each molecule as a rigid body and allowing one to move and rotate in relation to the other, inducing six degrees of freedom (three translational and three rotational). DoME's scanning method can be viewed as a tinker toy approach where the receptor and ligand molecules, represented by spools, are connected by sticks of various lengths, and their orientations in relation to one another are determined by the holes in the spool. The level of sampling coarseness (the lengths of the sticks and the holes in the spools) can be specified by the user. In our application, about 2 million sampling points, representing roughly a spacing of 30° in the rotational variables and 2.5 Å in the translational variables, are used in the scanning. Such scanning can be done in parallel to reduce the total CPU time. The lowest 900 points are used for local minimization. Underestimation is used to reduce the search area, and the process is iterated. Currently, a single convex quadratic approximation is used to underestimate the best local minima obtained in the scanning. The GU algorithm improves on this approach by allowing for the possibility of multiple funnels within the energy landscape, which is the more likely general shape of the potential energy function. Such underfitting is thus a more reasonable approach.

4 Computational implementation and results

The computational implementation of GU consists of two phases: Phase-1 LP and Phase-2 NLP. The first phase fixes the initial estimates for the nonlinear variables ρ_j and z_j so that (GU) reduces to a linear programming problem on the variables α, c, and β_j. The global minimum to this linear program is easily computed (often within a few iterations) and is then used as an initial estimate for the nonlinear program (GU), where the parameters ρ_j and z_j are now allowed to vary. By solving the linear program first, we obtain a reasonably good estimate for the global minimum of the nonlinear program.

4.1 Algorithmic details

The Gaussian Underestimation approach is summarized by the following algorithm:

Algorithm. Gaussian Underestimation (GU) algorithm.

```
Define Phase-1 initial point $p^{(0)} = (\alpha^{(0)}, c^{(0)}, \beta_j^{(0)}, \rho_j^{(0)}, z_j^{(0)})$ and set
```

$$\alpha^{(0)} = \min_k \left\{ f(x^{(k)}) - \left({c^{(0)}}^T x^{(0)} - \sum_{j=1}^{l} \beta_j^{(0)} e^{-\rho_j^{(0)} \|x^{(k)} - z_j^{(0)}\|^2} \right) \right\};$$

```
    to ensure feasibility of GU.
Solve Phase-1 LP to obtain $p^* = (\alpha^*, c^*, \beta_j^*, \rho_j^{(0)}, z_j^{(0)})$;
Solve Phase-2 NLP using Phase-1 LP solution $p^*$ as initial estimate solution;
end;
```

Computational details. In the simulated docking energy function cases, we define $Sf(x) : \Re^6 \to \Re$, whose domain $\mathcal{D} \in \Re^6$ is the six-dimensional hypercube $[0, 10] \times \cdots \times [0, 10]$.

Table 4 Results for 1TAB in DoME

	1TAB
$G(p^{(0)})$	1379.6
$G(p_f^*)$	898.5
`LP NPSOL Iter`	3
`NLP NPSOL Iter`	727
$\|\|x_{\text{pred}} - x_{g\,\text{min}}\|\|_2$	10.61
$\mathcal{D}_{\text{reduced}}$	8.89e02

The function $Sf(x)$ is a sum of four Gaussian functions, i.e., $l = 4$, with distinct local minima. The bound γ on the perturbation has values $2.0^r \times 10^{-1}$, where $r = 0, 1, \ldots, 5$. The parameters z_j are required to remain feasible in $\mathcal{D}$. In the DoME test case, we used the 1TAB system from the Protein Data Bank [1]. The 1TAB complex consists of the enzyme trypsin and BBI, the Bowman–Birk trypsin-inhibitor, which is a polypeptide chain of 71 amino acids highly cross-linked by seven disulfide bridges [11]. Elevated levels of trypsin have been found in pancreatic tumors, and BBI, commonly found in soybeans, has been shown to suppress this type of tumor in various animals. In all test cases, $m = 111$ points were used to construct the Gaussian underestimator.

In our implementation, the Phase-1 LP and the Phase-2 NLP are solved using the NPSOL software of Gill et al. (see [4]). All runs were made on a 1.33 GHz Mac OS X PowerBook G4 with 768 MB of RAM. The algorithm was written in C. The C version of NPSOL was generated using the f2c Fortran to C translator.

4.2 Results

The initial value of the objective function in Phase 1 and the final value of the objective function in Phase 2 are given by $G(p^{(0)})$ and $G(p_f^*)$, respectively. `LP NPSOL Iter` is the iteration count for the Phase 1 linear program and `NLP NPSOL Iter` is the count for the Phase 2 nonlinear program. The point $\bar{z}_1$ is the known global minimum of the simulated docking energy function, and $\hat{z}_1$ is the approximate global minimum computed by GU. The GU algorithm were tested on several simulated energy functions, and Tables 1–3 are representative of these runs. Table 4 contains the result for the 1TAB complex using DoME.

Simulated energy function. From the number of iterations in Phase 1 for all three examples, solving the linear program is relatively easy. This is to be expected since Phase 1 is a linear problem in only 13 variables. On the other hand, the nonlinear program Phase 2 requires more iterations, especially for the cases with larger perturbations. We note that even with significant perturbations ($\gamma \leq 0.8$), the computed approximate global minimum is within 0.5 in Euclidean distance from the known global minimum.

These are representative examples of the results we get using different $Sf(x)$ energy functions to define a typical problem with different types of local minima. These results show that the VRR is always less than 1.04×10^{-5} which then reduces the search domain from 10^6 down to approximately a single unit cube in the n-dimension space of z_j.

DoME. As in the simulated energy function cases, the number of Phase-1 iterations is smaller than the number for Phase-2 iterations. The high number of Phase-2 iterations is consistent with the fact that the energy function in DoME is highly nonlinear, which makes underfitting with a smooth function difficult. Although the GU algorithm predicted a global minimum x_{pred} that is 10.61 Å away from the known global minimum x_{gmin}, the initial domain volume of 3.1×10^7 is reduced to $\mathcal{D}_{\text{reduced}} = 9.0\times10^2$ after the GU algorithm is applied. With the domain significantly reduced, an iterative application of the convex quadratic approximation method [13] can then be applied to further reduce the domain and more accurately predict the location of the global minimum.

5 Conclusions

We have presented an approach for underfitting functions with more than one local minima with a large function value in magnitude. This approach uses an underestimating function $g(p;x)$ consisting of a sum of negative Gaussian functions. The parameters needed to define $g(p;x)$ are obtained by solving a two-phase optimization problem, where the solution of the linear program in the first phase is used as an approximate solution to the nonlinear program in the second phase. We demonstrated numerically that this two-phase approach is effective in giving estimates for the location of the global minimum of a protein docking function.

Acknowledgments This research was funded by the National Library of Medicine Training Grant 5T15LM007359-03, Department of Energy Grant DE-FG03-01ER25497, and NSF Grant #0513121.

References

1. Bernstein, F.C., Koetzle, T.F., Williams, G.J.B., Meyer, E.F. Jr., Brice, M.D., Rogers, J.R., Kennard, O., Shimanouchi, T., Tasumi, M.: The Protein Data Bank: a computer-based archival file for macromolecular structures. J. Mol. Biol. **112**, 535–542 (1977)
2. Björkman, M., Holmström, K.: Global optimization of costly nonconvex functions using radial basis functions. Optim. Eng. **1**(4), 373–297 (2001)
3. Dill K.A., Phillips A.T., Rosen J.B.: CGU an algorithm for molecular structure prediction, in Large-scale optimization with applications, Part III (Minneapolis, MN, 1995), IMA Vol. Math. Appl., vol. 94, pp 1–21 Springer, New York (1997)
4. Gill P.E., Murray W., Saunders M.A., Wright M.H.: User's guide for NPSOL (version 4.0): A Fortran package for nonlinear programming, Tech. Rep. SOL-86-2, Systems Optimization Laboratory, Stanford University, Stanford, CA (1986)
5. Gutmann, H.-M.: A radial basis function method for global optimization. J. Global Optim. **19**(3), 201–227 (2001)
6. Mangasarian, O.L., Rosen, J.B., Thompson, M.E.: Global minimization via piecewise-linear underestimation. J. Global Optim. **32**, 1–9 (2005)
7. Mangasarian, O.L., Rosen, J.B., Thompson, M.E.: Convex kernel estimation of functions with multiple local minima. Comp. Optim. Appl. **34**, 35–45 (2006)
8. Marcia, R.F., Mitchell, J.C., Rosen, J.B.: Iterative convex quadratic approximation for global optimization in protein docking. Comp. Optim. Appl. **32**, 285–297 (2005)
9. Mitchell J.C., Rosen J.B., Phillips A.T., Ten Eyck L.F.: Coupled optimization in protein docking. In Proceedings of the Third Annual International Conference on Computational Molecular Biology, ACM Press, 280–284 (1999)
10. Regis, R.G., Shoemaker, C.A.: Constrained global optimization of expensive black box functions using radial basis functions. J. Global Optim. **31**, 153–171 (2005)
11. Tsunogae, Y., Tanaka, I., Yamane, T., Kikkawa, J., Ashida, T., Ishikawa, C., Watanabe, K., Nakamura, S., Takahashi, K.: Structure of the trypsin-binding domain of bowman-birk type protease inhibitor and its interaction with trypsin. J. Biochem. (Tokyo) **100**, 1637–1646 (1986)
12. Rosen, J.B., Dill, K.A., Phillips, A.T.: Protein structure and energy landscape dependence on sequence using a continuous energy function. J. Comput. Bio. **4**, 227–239 (1997)
13. Rosen, J.B., Marcia, R.F.: Convex quadratic approximation. Comp. Optim. Appl. **28**, 173–184 (2004)