TUCS

Napsu Karmitsa

On testing nonsmooth formulations of the Lennard-Jones potential in polyatomic clustering problems

Turku Centre *for* Computer Science

TUCS Technical Report
No 1132, April 2015

On testing nonsmooth formulations of the Lennard-Jones potential in polyatomic clustering problems

Napsu Karmitsa
University of Turku, Department of Mathematics and Statistics,
FI-20014 Turku, Finland
napsu@karmitsa.fi

TUCS Technical Report
No 1132, April 2015

Abstract

A cluster is a group of identical molecules or atoms loosely bound by inter-atomic forces. The optimal geometry minimizes the potential energy — usually modelled as the Lennard-Jones potential — of the cluster. The minimization of the Lennard-Jones potential is a very difficult global optimization problem with extremely many local minima. In addition to cluster problems, the Lennard-Jones potential represents an important component in many of the potential energy models used, for example, in protein folding, protein-peptide docking, and complex molecular conformation problems. In this paper we study different modifications of the Lennard-Jones potential in order to improve the success rate of finding the global minimum of the original potential. The main interest of the paper is in nonsmooth penalized form of the Lennard-Jones potential. The preliminary numerical experiments confirm that the success rate of finding the global minimum is clearly improved when using the new formulae.

Keywords: Lennard-Jones potential, clustering problem, molecular conformation, nonsmooth optimization, global optimization.

TUCS Laboratory
TUCS Laboratory

1 Introduction

A cluster is a group of identical molecules or atoms loosely bound by inter-atomic forces. The optimal geometry minimizes the potential energy of the cluster, expressed as a function of Cartesian coordinates

$$E(x, y, z) = \sum_{i=1}^{N} \sum_{j=i+1}^{N} v(r_{ij}), \tag{1}$$

where N is the number of atoms (molecules) in the cluster and r_{ij} is the distance between the centers of a pair of atoms (molecules). That is,

$$r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}.$$

The simplest model (yet extremely difficult to solve) uses the *Lennard-Jones* pairwise potential energy function

$$v(r_{ij}) = \frac{1}{r_{ij}^{12}} - \frac{2}{r_{ij}^{6}}. \tag{2}$$

Variations of this problem include carbon and argon clusters as well as water molecule clusters (see, e.g. [12, 22, 23]). In addition, the Lennard-Jones potential represents an important component in many of the potential energy models used, for instance, in complex molecular conformation, protein-peptide docking, and protein folding problems [13, 14, 20].

The objective function of the Lennard-Jones potential (1) and (2) is smooth (continuously differentiable) assuming that $r_{ij} > 0$ and easy to implement. However, it has extremely complicated landscape with huge number of local minima. In [17], a smooth penalized modification for the Lennard-Jones pairwise potential function (2) was introduced that allows a local search method to escape from the enormous number of local minima of the Lennard-Jones energy landscape. The formula of this *penalized Lennard-Jones potential* is

$$\bar{v}(r) = \frac{1}{r^{2p}} - \frac{2}{r^{p}} + \mu r + \beta(\max\{0, r^2 - D^2\})^2, \tag{3}$$

where $p > 0$, $\mu, \beta \geq 0$ are real constants and $D > 0$ is an underestimate of the diameter of the cluster.

The local minimum of the modified objective function (1) and (3) was then used as a starting point for a local optimization of the Lennard-Jones potential function (1) and (2). This procedure was reported to result convergence to global minimum with much greater success than when starting local optimization with random points [17].

The idea of penalized potential was further modified in [4] resulting a *nonsmooth penalized Lennard-Jones potential*

$$\bar{v}(r) = \frac{1}{r^{12}} - \frac{1}{r^{6}} + \mu r + \beta(\max\{0, r^2 - D^2\}). \tag{4}$$

According to very limited number of test cases used in [4], this formulation together with *discrete gradient method* [1] used for minimization, yields yet another improvement in the success rate of finding the global minimum.

In this paper we study the different parameter values for the nonsmooth penalized Lennard-Jones potential (4) and also some new modifications of this nonsmooth formulation. Our goal is to confirm the results obtained in [4] and to further improve the success rate of finding the global minimum of the original problem (1) and (2) .

As a solver for the minimization problem we use the *limited memory discrete gradient bundle method* (LDGB, [11]) that is a derivative free method for nonsmooth moderate large problems. The LDGB is a hybrid of the discrete gradient method [1] and the limited memory bundle method [9, 10]. The choice of the solver is reasoned by three facts: First, we need a solver that is capable to solve (locally) nonsmooth nonconvex problems. Second, the computation of subgradients (generalized gradients [6]) is not an easy task, since the problem is subdifferentially irregular (see, e.g. [2]) and, thus, the calculus exists only in the form of inclusions. Therefore, the choice of derivative free method is justified. Finally, the number of variables in the clustering problem is 3N. This means that our solution algorithm needs to be able to solve moderate large problems.

The paper is organized as follows. In the next section, we introduce formulae used in our experiment. Then, in Section 3, we briefly describe the basic ideas of LDGB. In section 4, we give the results of our numerical experiments and, finally, in Section 5, we conclude the paper.

2 Nonsmooth polyatomic clustering problem

In this paper we try to escape from the local minima of the Lennard-Jones energy landscape using a *nonsmooth penalized Lennard-Jones potential* of the form

$$\bar{v}(r) = \frac{1}{r^{2p}} - \frac{2}{r^p} + \mu r + \beta(\max\{0, r^2 - D^2\}), \tag{5}$$

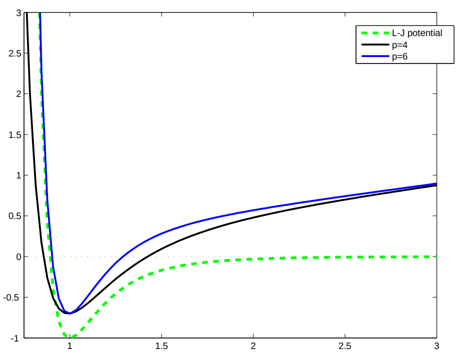
where $p > 0$, $\mu, \beta \geq 0$ are real constants and $D > 0$ is an underestimate of the diameter of the cluster. The local minimum of this modified objective function may be used as a starting point for a local optimization of the Lennard-Jones potential function (1) and (2). Note that by choosing $p = 6$ and $\mu, \beta = 0$, the penalized Lennard-Jones potential $\bar{v}$ coincides with the Lennard-Jones pairwise potential (2).

In addition, we use the formula where, instead of linear penalty μr used in (5), the first penalty is given with piecewise linear formula. That is, we use the formula

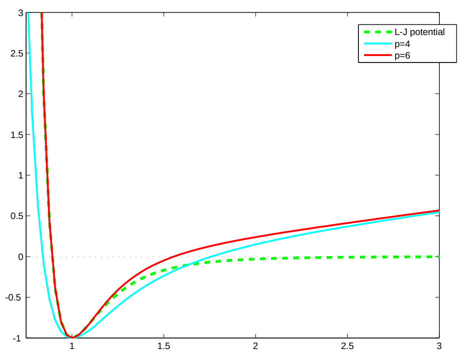
$$\bar{v}(r) = \frac{1}{r^{2p}} - \frac{2}{r^p} + \mu(\max\{0, r - 1.1\}) + \beta(\max\{0, r^2 - D^2\}). \tag{6}$$

In both of these formulae parameter p affects the rigidity of the model. By choosing $p < 6$ the atoms (molecules) can be moved more freely and by decreasing p the infinite barrier at $r = 0.0$, which prevents atoms from getting too close to each other, is also decreased. The first penalty term μr in (5) and $\mu(\max\{0, r-1.1\})$ in (6) gives a penalty to distances between the atoms. The penalty increases linearly as a function of distance. Nevertheless, in (6) we do not punish for distances smaller than 1.1. In addition, the first penalty terms have some smoothing effect (see [7] for the analysis of the first penalty term in (5)). On its turn, the second penalty term adds a penalty to the diameter of the cluster. It has no influence on pairs of atoms close to each other but it adds strong penalty to the atoms far away from each other. As in [17] the local minima of the modified objective functions (1) and (5) or (1) and (6) will be used as a starting point for a local optimization of the Lennard-Jones potential function (1) and (2).

In Figure 1 the formulae (5) and (6) with parameters $\mu = 0.3$, $\beta = 0$, $D = 0$ and $p = 6$ or $p = 4$ are displayed and compared with the Lennard-Jones pairwise potential (2). In Figure 2 the corresponding cases with diagonal penalization — that is, $\beta = 1.0$, $D = 2.0$ — are also displayed. Finally, in Figure 3, we compare the smooth formulation (3) with the nonsmooth one (6). Here, we have used two set of parameters: in Figure 3(a) we have set $p = 6$, $\mu = 0.3$, $\beta = 1.0$, and $D = 2.0$ and in Figure 3(b) we have set $p = 4$, $\mu = 1.0$, $\beta = 1.0$, and $D = 2.0$.

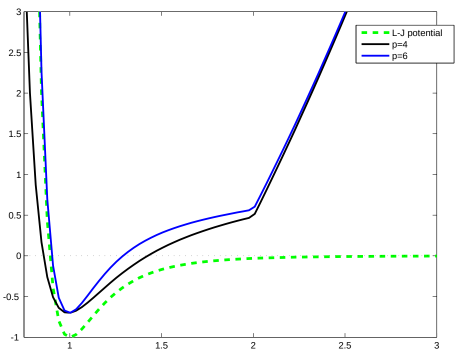


(a) Linear penalty

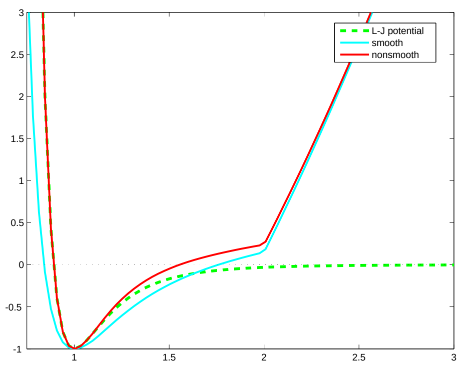


(b) Piecewise linear penalty

Figure 1: Comparison between Lennard-Jones and modified potentials.

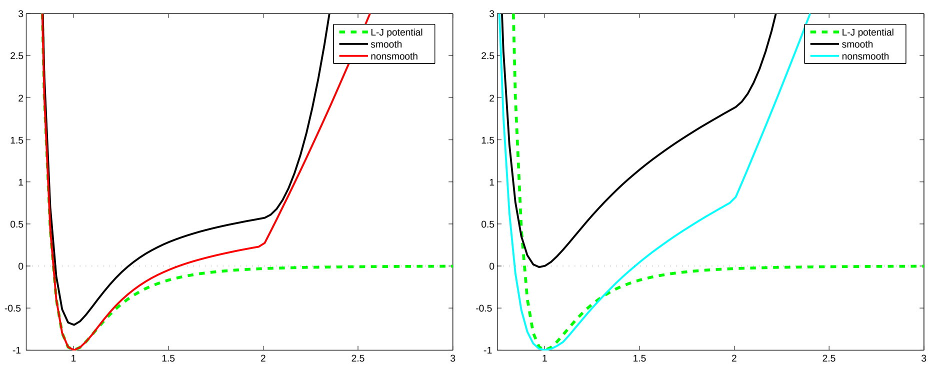


(a) Penalty (5)



(b) Penalty (6)

Figure 2: Comparison between Lennard-Jones and modified potentials (cont.).



(a) p=6 (b) p=4

Figure 3: Comparison between smooth and nonsmooth potentials.

We do not give here more detailed analysis of the effect of these penalized formulae. For a reader more interested, we recommend to see [7, 17, 18].

3 Limited memory discrete gradient bundle method

In this section, we describe the basic ideas of derivative free LDGB that is used as a solver for the minimization problem discussed in the previous chapter. The LDGB exploits the ideas of the variable metric bundle method [21]: namely, the utilization of null steps, simple aggregation, and the subgradient locality measures. Nevertheless, we use the discrete gradients instead of subgradient and the search direction is calculated using the limited memory approach. The only assumptions made are that the objective function is locally Lipschitz continuous and at every point $\boldsymbol{x} \in \mathbb{R}^n$ we can evaluate the value of the objective function $f(\boldsymbol{x})$. For a reader more interested in nonsmooth optimization and details of the method, we recommend to see [2] and [11].

We first give a simple flowchart of the method in Figure 4. Then we say few words of the algorithm.

Discrete Gradient. We start be defining the discrete gradient. Let us denote by

$$S_1 = \{\boldsymbol{g} \in \mathbb{R}^n \mid \|\boldsymbol{g}\| = 1\}$$

the sphere of the unit ball and by

$$P = \{z \mid z : \mathbb{R}_+ \to \mathbb{R}_+,\ \delta > 0,\ \delta^{-1}z(\delta) \to 0,\ \delta \to 0\}$$

the set of univariate positive infinitesimal functions. In addition, let

$$G = \{\boldsymbol{e} \in \mathbb{R}^n \mid \boldsymbol{e} = (e_1, \ldots, e_n), |e_j| = 1,\ j = 1, \ldots, n\}$$

be a set of all vertices of the unit hypercube in $\mathbb{R}^n$.

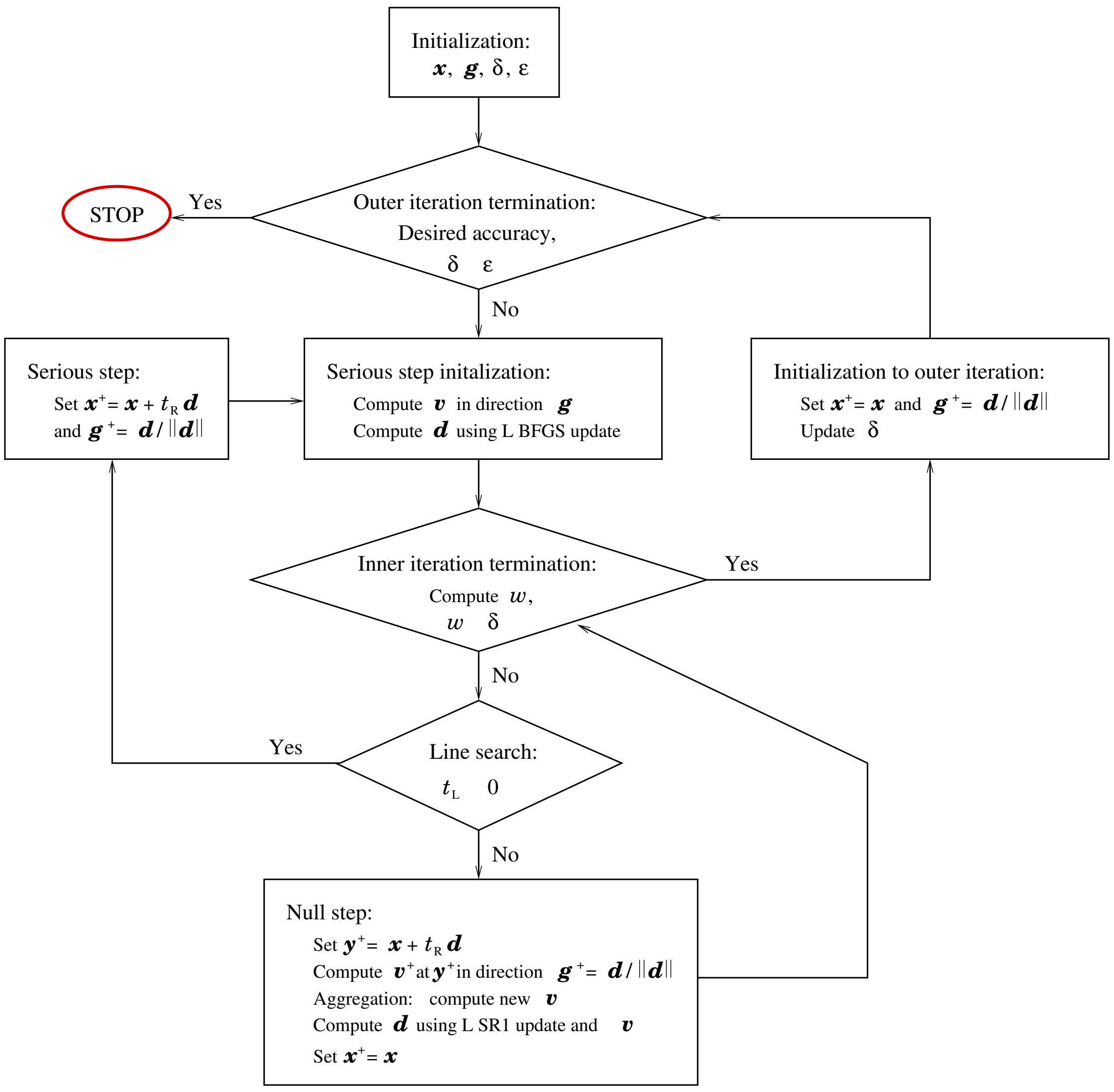

Initialization:
x, g, δ, ε
Outer iteration termination:
Desired accuracy,
δ ε
STOP
Yes
No
Serious step:
Set $x^+ = x + t_R d$
and $g^+ = d/\|d\|$
Serious step initalization:
Compute v in direction g
Compute d using L BFGS update
Initialization to outer iteration:
Set $x^+ = x$ and $g^+ = d/\|d\|$
Update δ
Inner iteration termination:
Compute w,
w δ
Yes
No
Line search:
t_L 0
Yes
No
Null step:
Set $y^+ = x + t_R d$
Compute v^+ at y^+ in direction $g^+ = d/\|d\|$
Aggregation: compute new v
Compute d using L SR1 update and v
Set $x^+ = x$

Figure 4: Program LDGB .

Now, take any $\boldsymbol{g} \in S_1$, $\boldsymbol{e} \in G$, $z \in P$, $\alpha \in (0,1]$, and compute $i = \operatorname{argmax}\{|g_j|,\ j = 1,\dots,n\}$. For $\boldsymbol{e} \in G$ define the sequence of n vectors

$$\boldsymbol{e}^j(\alpha) = (\alpha e_1, \alpha^2 e_2, \dots, \alpha^j e_j, 0, \dots, 0),$$

with $j = 1,\dots,n$ and for $\boldsymbol{x} \in \mathbb{R}^n$ and $\delta > 0$ consider the points

$$\boldsymbol{x}_0 = \boldsymbol{x} + \delta \boldsymbol{g}, \qquad \boldsymbol{x}_j = \boldsymbol{x}_0 + z(\delta)\boldsymbol{e}^j(\alpha), \qquad j = 1,\dots,n.$$

DEFINITION 3.1. The *discrete gradient* of the function $f : \mathbb{R}^n \to \mathbb{R}$ at the point $\boldsymbol{x} \in \mathbb{R}^n$ is the vector $\Gamma^i(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{e}, z, \delta, \alpha) = (\Gamma^i_1, \ldots, \Gamma^i_n) \in \mathbb{R}^n$ with the following coordinates:

$$\Gamma^i_j = [z(\delta)\alpha^j e_j)]^{-1} [f(\boldsymbol{x}_j) - f(\boldsymbol{x}_{j-1})], \qquad j = 1, \ldots, n, \ j \neq i,$$
$$\Gamma^i_i = (\delta g_i)^{-1} \left[f(\boldsymbol{x} + \delta \boldsymbol{g}) - f(\boldsymbol{x}) - \delta \sum_{j=1, j\neq i}^{n} \Gamma^i_j g_j \right].$$

The *closed convex set of discrete gradients*

$$V_0(\boldsymbol{x}, \delta) = \operatorname{cl}\operatorname{conv}\{\boldsymbol{v} \in \mathbb{R}^n \mid \exists\, \boldsymbol{g} \in S_1,\ \boldsymbol{e} \in G,\ z \in P, \alpha > 0 \text{ such that } \boldsymbol{v} = \Gamma^i(\boldsymbol{x}, \boldsymbol{g}, \boldsymbol{e}, z, \delta, \alpha)\}$$

is an approximation to the subdifferential $\partial f(\boldsymbol{x})$ for sufficiently small $\delta > 0$ [1, 2].

Outer and Inner Iterations. Both outer and inner iterations are used in the LDGB: The inner iteration of the LDGB is essentially same as the limited memory bundle method [9, 10] but now we use the discrete gradients instead of subgradient of the objective function. The outer iteration is used in order to avoid too tight approximations to the subgradients at the beginning of computation (thus, we have a derivative free method). That is, we start with "large" δ and make it smaller when we are closer to the optimum.

Search Direction. As already said, we use the discrete gradients instead of subgradient in our calculations and the search direction $\boldsymbol{d}_k$ is calculated using the limited memory approach. That is,

$$\boldsymbol{d}_k = -D^k \tilde{\boldsymbol{v}}_k,$$

where $\tilde{\boldsymbol{v}}_k$ is (an aggregate) discrete gradient and D^k is the limited memory variable metric update that, in the smooth case, represents the approximation of the inverse of the Hessian matrix. Note that the matrix D^k is not formed explicitly but the search direction $\boldsymbol{d}_k$ is calculated using the limited memory approach (to be described later).

Line Search. In order to determine a new step into the search direction $\boldsymbol{d}_k$, the LDGB uses the so-called *line search procedure* (see [10, 21]): a new iteration point $\boldsymbol{x}_{k+1}$ and a new auxiliary point $\boldsymbol{y}_{k+1}$ are produced such that

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{x}_k + t^k_L \boldsymbol{d}_k && \text{and} \\ \boldsymbol{y}_{k+1} &= \boldsymbol{x}_k + t^k_R \boldsymbol{d}_k, && \text{for } k \geq 1 \end{aligned}$$

with $\boldsymbol{y}_1 = \boldsymbol{x}_1$, where $t^k_R \in (0, t_{max}]$ and $t^k_L \in [0, t^k_R]$ are step sizes, and $t_{max} > 1$ is the upper bound for the step size. A necessary condition for a *serious step* is to have

$$t^k_R = t^k_L > 0 \qquad \text{and} \qquad f(\boldsymbol{y}_{k+1}) \leq f(\boldsymbol{x}_k) - \varepsilon^k_L t^k_R w_k, \tag{7}$$

where $\varepsilon_L^k \in (0, 1/2)$ is a line search parameter and $w_k > 0$ represents the desirable amount of descent of f at $\boldsymbol{x}_k$. If the condition (7) is satisfied, we set $\boldsymbol{x}_{k+1} = \boldsymbol{y}_{k+1}$ and a serious step is taken.

On the other hand, a *null step* is taken if

$$t_R^k > t_L^k = 0 \qquad \text{and} \qquad -\beta_{k+1} + \boldsymbol{d}_k^T \boldsymbol{v}_{k+1} \geq -\varepsilon_R^k w_k,$$

where $\varepsilon_R^k \in (\varepsilon_L^k, 1/2)$ is a line search parameter and $\boldsymbol{v}_{k+1} \in V_0(\boldsymbol{y}_{k+1}, \delta_k)$. Moreover, β_{k+1} is the subgradient locality measure [16, 19] similar to standard bundle methods, that is,

$$\beta_{k+1} = \max\{|f(\boldsymbol{x}_k) - f(\boldsymbol{y}_{k+1}) + (\boldsymbol{y}_{k+1} - \boldsymbol{x}_k)^T \boldsymbol{v}_{k+1})|, \gamma\|\boldsymbol{y}_{k+1} - \boldsymbol{x}_k\|^2\}.$$

Here $\gamma \geq 0$ is a distance measure parameter supplied by the user. Parameter γ can be set to zero when f is convex. In the case of a null step, we set $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$ but information about the objective function is increased because we store the auxiliary point $\boldsymbol{y}_{k+1}$ and the corresponding auxiliary discrete gradient $\boldsymbol{v}_{k+1} \in V_0(\boldsymbol{y}_{k+1}, \delta_k)$.

Under some semismoothness assumptions the line search procedure used with the LDGB is guaranteed to find the step sizes t_L^k and t_R^k such that exactly one of the two possibilities — a serious step or a null step — occurs [21].

Aggregation. The LDGB uses the original discrete gradient $\boldsymbol{v}_k$ after the serious step and the aggregate subgradient $\tilde{\boldsymbol{v}}_k$ after the null step for direction finding (i.e. we set $\tilde{\boldsymbol{v}}_k = \boldsymbol{v}_k$ if the previous step was a serious step). The *aggregation procedure* is carried out by determining multipliers λ_i^k satisfying $\lambda_i^k \geq 0$ for all $i \in \{1, 2, 3\}$, and $\sum_{i=1}^3 \lambda_i^k = 1$ that minimize a simple quadratic function

$$\begin{aligned}\varphi(\lambda_1, \lambda_2, \lambda_3) = {} & [\lambda_1 \boldsymbol{v}_m + \lambda_2 \boldsymbol{v}_{k+1} + \lambda_3 \tilde{\boldsymbol{v}}_k]^T D^k [\lambda_1 \boldsymbol{v}_m + \lambda_2 \boldsymbol{v}_{k+1} + \lambda_3 \tilde{\boldsymbol{v}}_k] \\ & + 2(\lambda_2 \beta_{k+1} + \lambda_3 \tilde{\beta}_k).\end{aligned}$$

Here $\boldsymbol{v}_m \in V_0(\boldsymbol{x}_k, \delta_k)$ is the current discrete gradient (m denotes the index of the iteration after the latest serious step, i.e. $\boldsymbol{x}_k = \boldsymbol{x}_m$), $\boldsymbol{v}_{k+1} \in V_0(\boldsymbol{y}_{k+1}, \delta_k)$ is the auxiliary discrete gradient, and $\tilde{\boldsymbol{v}}_k$ is the current aggregate discrete gradient from the previous iteration ($\tilde{\boldsymbol{v}}_1 = \boldsymbol{v}_1$). In addition, β_{k+1} is the current subgradient locality measure and $\tilde{\beta}_k$ is the current aggregate subgradient locality measure ($\tilde{\beta}_1 = 0$). The optimal values λ_i^k, $i \in \{1, 2, 3\}$ can be calculated by using simple formulae (see [21]).

The resulting aggregate discrete gradient $\tilde{\boldsymbol{v}}_{k+1}$ and aggregate subgradient locality measure $\tilde{\beta}_{k+1}$ are computed by the formulae

$$\tilde{\boldsymbol{v}}_{k+1} = \lambda_1^k \boldsymbol{v}_m + \lambda_2^k \boldsymbol{v}_{k+1} + \lambda_3^k \tilde{\boldsymbol{v}}_k \qquad \text{and} \qquad \tilde{\beta}_{k+1} = \lambda_2^k \beta_{k+1} + \lambda_3^k \tilde{\beta}_k.$$

Due to this simple aggregation procedure only one trial point $\boldsymbol{y}_{k+1}$ and the corresponding discrete gradient $\boldsymbol{v}_{k+1} \in V_0(\boldsymbol{y}_{k+1}, \delta_k)$ need to be stored.

The aggregation procedure gives us a possibility to retain the global convergence without solving the quite complicated quadratic direction finding problem (see e.g. [2]) appearing in standard bundle methods. Note that the aggregate values are computed only if the last step was a null step. Otherwise, we set $\tilde{\boldsymbol{v}}_{k+1} = \boldsymbol{v}_{k+1}$ and $\tilde{\beta}_{k+1} = 0$.

Matrix Updating. In the LDGB both the limited memory BFGS (L-BFGS) and the limited memory SR1 (L-SR1) update formulae [5] are used in calculations of the search direction and the aggregate values. The idea of limited memory matrix updating is that instead of storing large $n \times n$ matrices D^k, one stores a certain (usually small) number of vectors $\boldsymbol{s}_k = \boldsymbol{y}_{k+1} - \boldsymbol{x}_k$ and $\boldsymbol{u}_k = \boldsymbol{v}_{k+1} - \boldsymbol{v}_m$ obtained at the previous iterations of the algorithm, and uses these vectors to implicitly define the variable metric matrices. Note that, due to usage of null steps we may have $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$ and thus, we use here the auxiliary point $\boldsymbol{y}_{k+1}$ instead of $\boldsymbol{x}_{k+1}$.

Let us denote by $\hat{m}_c$ the user-specified maximum number of stored correction vectors ($3 \le \hat{m}_c$) and by $\hat{m}_k = \min\{\, k-1, \hat{m}_c \,\}$ the current number of stored correction vectors. Then the $n \times \hat{m}_k$ dimensional correction matrices S_k and U_k are defined by

$$\begin{aligned} S_k &= \begin{bmatrix} \boldsymbol{s}_{k-\hat{m}_k} & \dots & \boldsymbol{s}_{k-1} \end{bmatrix} \qquad \text{and} \\ U_k &= \begin{bmatrix} \boldsymbol{u}_{k-\hat{m}_k} & \dots & \boldsymbol{u}_{k-1} \end{bmatrix}. \end{aligned}$$

The inverse L-BFGS update is defined by the formula

$$D^k = \vartheta_k I + \begin{bmatrix} S_k & \vartheta_k U_k \end{bmatrix} \begin{bmatrix} (R_k^{-1})^T (C_k + \vartheta_k U_k^T U_k) R_k^{-1} & -(R_k^{-1})^T \\ -R_k^{-1} & 0 \end{bmatrix} \begin{bmatrix} S_k^T \\ \vartheta_k U_k^T \end{bmatrix},$$

where R_k is an upper triangular matrix of order $\hat{m}_k$ given by the form

$$(R_k)_{ij} = \begin{cases} (\boldsymbol{s}_{k-\hat{m}_k-1+i})^T (\boldsymbol{u}_{k-\hat{m}_k-1+j}), & \text{if } i \le j \\ 0, & \text{otherwise,} \end{cases}$$

C_k is a diagonal matrix of order $\hat{m}_k$ such that

$$C_k = \operatorname{diag}\,[\boldsymbol{s}_{k-\hat{m}_k}^T \boldsymbol{u}_{k-\hat{m}_k}, \dots, \boldsymbol{s}_{k-1}^T \boldsymbol{u}_{k-1}],$$

and ϑ_k is a positive scaling parameter.

In addition, the inverse L-SR1 update is defined by

$$D^k = \vartheta_k I - (\vartheta_k U_k - S_k)(\vartheta_k U_k^T U_k - R_k - R_k^T + C_k)^{-1} (\vartheta_k U_k - S_k)^T.$$

In the case of a null step, the LDGB uses the L-SR1 update formula, since this formula allows to preserve the boundedness and some other properties of generated matrices which guarantee the global convergence of the method. Otherwise, since these properties are not required after a serious step, the more efficient L-BFGS update is employed. In the LDGB, the individual updates that would violate positive definiteness are skipped (for more details, see [8, 9, 10, 11]).

Stopping Criterion. For smooth functions, a necessary condition for a local minimum is that the gradient has to be zero and by continuity it becomes small when we are close to an optimal point. This is no longer true when we replace the gradient by an arbitrary subgradient or a discrete gradient. Due to the aggregation procedure, we have quite a useful approximation to the gradient at our disposal, namely the aggregate discrete gradient $\tilde{\boldsymbol{v}}_k$. However, as a stopping criterion, the direct test $\|\tilde{\boldsymbol{v}}_k\| < \delta_k$, for some $\delta_k > 0$, is too uncertain, if the current piecewise linear approximation of the objective function is too rough. Therefore, we use the term $\tilde{\boldsymbol{v}}_k^T D^k \tilde{\boldsymbol{v}}_k = -\tilde{\boldsymbol{v}}_k^T \boldsymbol{d}_k$ and the aggregate subgradient locality measure $\tilde{\beta}_k$ to improve the accuracy of $\|\tilde{\boldsymbol{v}}_k\|$. Hence, the stopping parameter w_k at iteration k is defined by

$$w_k = -\tilde{\boldsymbol{v}}_k^T \boldsymbol{d}_k + 2\tilde{\beta}_k.$$

The inner iteration stops if $w_k \leq \delta_k$ and the outer iteration — and, thus, the algorithm — stops if $\delta_k \leq \varepsilon$ for some user specified $\varepsilon > 0$. The parameter w_k is also used during the line search procedure to represent the desirable amount of descent.

Global Convergence. If the LDGB algorithm terminates after a finite number of iterations, say at iteration k, then the point $\boldsymbol{x}_k$ is a stationary point of f. Otherwise, the accumulation point $\bar{\boldsymbol{x}}$ generated by LDGB algorithm is a a stationary point of f [11]).

4 Numerical experiments

We now give the results of our numerical experiments. To solve the problems we have used the solver `LDGB` discussed in the previous chapter. The Fortran 95 source code of the solver is available for downloading at http://napsu.karmitsa.fi/ldgb/. The experiments were performed on an Intel® Core™ 2 CPU 1.80GHz. To compile the code, we used `gfortran`, the GNU Fortran compiler.

In order to test the performance of modified formulae (5) and (6) we made a series of numerical experiments by running `LDGB` 1 000 times with $N = 2, \ldots, 40$. We started local optimization of the modified objective functions (1) and (5) or (1) and (6) with random points. That is, no special point generation procedure similar to [17] was used. Then the local minima of the modified potentials were used as starting points for a local optimization of the original Lennard-Jones potential function (1) and (2). In what follows we report the percentage of the trials which led to the putative global minimum as given in [15][1].

Let us first study only the linear and piecewise linear penalty term (that is, we set $\beta = 0$ in equations (5) and (6)) with different values of parameters p and μ. The results of these experiments are given in Tables 1 ($p = 6$) and 2 ($p \leq 4$), where "N" denotes the number of atoms, "L-J" denotes the success rate obtained with the original Lennard-Jones formulation (1) and (2), "Locatelli" stands for the results of Locatelli and Schoen [17] (we recall some of these results for comparison purposes), "linear" stands for the linear penalty (that is, equation (5) with $\beta = 0$) and "PW linear" stands for the piecewise linear penalty (equation (6) with $\beta = 0$).

First, it is worth to note that all the penalized formulations were superior when compared to the original Lennard-Jones formulation. For example, with $N = 13$ the

[1]The minimum obtained with $N = 13$ was smaller than that given in [15].

putative global minimum was found only 12 times when only the original Lennard-Jones formulation was used while with piecewise linear penalty with $p = 6$ and $\mu = 5$ it was found 989 times. In addition, with original Lennard-Jones formulation no global minimum was found with $N > 25$. Nevertheless, there was an exception: with $N = 8$ the piecewise linear penalty with $p = 6$ and $\mu = 5$ was worse than the original Lennard-Jones formulation (see Table 1).

In each table, the best values at all `LDGB`'s results are bolded. In addition, in Table 1 we have used red pen to show which one is the better (or equal): the smooth or the nonsmooth formulation with the same parameters. That is, the same values of p and μ are compared. It is now easy to see that the piecewise linear penalty usually gave better or equal results (in 75% of cases). This may follow from the fact that the minimizer of the piecewise linearly penalized formula (6) and that of the original formula (2) are the same while with the linearly penalized formula (5) there exist a small disruption (see Figure 1). Nevertheless, the magnitudes of the results are about the same.

In Table 1, we have emphasized with blue pen those values where Locatelli's results are better when compared to the run with `LDGB` with the same formula and parameters (that is, "linear" with $p = 6$ and $\mu = 0.3$). It can be seen that Locatelli's results are usually little bit better especially with larger N. This difference is probably due to specialized starting point generation procedure used in [17] rather than inferiority of our solution algorithm. In fact, since the results are of the same magnitude, it may be that `LDGB` does give some advantage — almost as strong as specialized starting point generation — when solving these kinds of problems.

When comparing different values of μ in Tables 1 and 2, it seems that large μ is well suited for small N but when N increases smaller value of μ is better. Indeed, when $N < 22$ the best success rates with $p = 6$ were usually obtained with piecewise linear penalty and $\mu = 5$. However, no successful runs were made when $N > 31$. With $p = 4$ and $\mu = 5$ no successful runs were made when $N > 13$. This trend can be seen both with the linear and with the piecewise linear penalty. Therefore, with $p = 4$, we also tested the values of μ that depends on the value of N. That is, $\mu = 10/N$ and $\mu = 1/N$. In Table 2 we see that this strategy gives us somewhat better results. Although the best success rate with small N is still usually obtained with either $\mu = 2$ or $\mu = 5$ the overall performance is best with $\mu = 10/N$. Nevertheless, it is worth of noting that $\mu = 1/N$ is the only choice of the parameter that gave the putative global optimum at least once with every N during our test drives.

In Table 2 we have compared Locatelli's results to piecewise linear formulation with the same parameters (blue pen). As before, Locatelli's results are slightly better from those of `LDGB` when N increases. However, the trend is not as clear as in Table 1. In addition, we compared Locatelli's results to our "best" parameters with $p = 4$, that is $\mu = 10/N$. In Table 2 we have emphasized with red pen those results that are better than or equal to Locatelli's results. That happens in 77% of the cases. It is also worth of noting that the improvements are often clear.

When comparing the different values of parameter p, we see that $p = 4$ usually gives better results than $p = 6$ when only the linear or the piecewise linear penalty is used:

Table 1: Success rate with linear penalty and $p = 6$.

N	L-J	Locatelli	linear	linear	linear	PW linear	PW linear	PW linear
		$p = 6$	$p = 6$	$p = 6$	$p = 6$	$p = 6$	$p = 6$	$p = 6$
		$\mu = 0.3$	$\mu = 0.3$	$\mu = 1.0$	$\mu = 5.0$	$\mu = 0.3$	$\mu = 1.0$	$\mu = 5$
2	88.7		99.9	99.9	**100.0**	99.9	99.8	99.9
3	78.4		93.5	96.9	99.3	95.2	97.9	**99.5**
4	67.4		91.2	94.7	98.4	91.7	94.4	**99.7**
5	63.9		98.6	99.0	**99.9**	98.7	98.9	**99.9**
6	1.3		11.6	23.2	99.2	10.1	40.5	**99.4**
7	11.1		23.0	28.2	35.8	25.6	27.7	**41.6**
8	19.8		**50.7**	45.9	25.5	48.8	43.7	14.2
9	6.6		21.0	25.8	**30.5**	20.9	25.8	26.3
10	1.7	9.0	8.4	18.4	38.0	8.5	25.3	**54.7**
11	1.2	14.8	14.4	31.6	58.9	14.2	39.4	**78.9**
12	1.2	24.4	26.3	55.4	87.2	29.5	61.4	**96.3**
13	1.2	21.0	21.4	55.9	91.1	23.6	65.3	**98.9**
14	2.6	39.3	45.3	73.1	95.7	43.6	78.5	**97.0**
15	1.7	17.4	19.7	14.2	9.1	**21.6**	16.0	19.8
16	0.9	10.5	11.1	16.2	3.5	14.5	**18.5**	8.3
17	0.2	4.0	**4.7**	4.1	2.1	4.2	3.9	0.7
18	–	2.5	2.8	12.1	22.8	3.5	14.7	**24.3**
19	0.2	6.1	5.9	14.9	21.8	7.8	14.8	**28.3**
20	0.2	8.3	8.9	18.9	27.7	9.9	21.6	**32.9**
21	–	3.1	2.8	7.7	14.7	4.6	10.8	**15.5**
22	–	6.4	5.6	16.7	**29.1**	8.8	17.4	27.5
23	0.1	3.4	3.3	10.6	18.5	2.4	11.5	**21.3**
24	–	5.3	5.5	13.6	**28.6**	6.7	18.4	28.5
25	0.1	8.3	7.2	18.9	**33.8**	8.7	22.3	33.2
26	–	2.4	2.0	11.8	29.4	2.4	13.8	**44.8**
27	–	0.5	0.2	0.2	0.1	0.4	**0.5**	–
28	–	1.0	0.7	1.0	–	0.8	0.8	**1.1**
29	–	1.1	1.0	4.4	–	1.2	4.3	**5.9**
30	–	0.1	**0.1**	–	–	**0.1**	–	–
31	–	0.2	–	0.1	–	0.2	**0.5**	0.3
32	–	0.4	**0.5**	0.1	–	0.3	0.2	–
33	–	0.5	0.7	**1.0**	–	0.8	**1.0**	–
34	–	0.1	**–**	**–**	**–**	**–**	**–**	**–**
35	–	0.2	**0.3**	0.2	–	0.1	–	–
36	–	0.2	–	**0.6**	–	0.1	0.2	–
37	–	0.1	–	–	–	**0.1**	–	–
38	–	0.2	0.1	1.6	–	0.4	**2.5**	–
39	–	0.2	**0.1**	–	–	**0.1**	**0.1**	–
40	–	0.1	0.4	**0.6**	–	0.4	0.2	–

Table 2: Success rate with piecewise linear penalty and $p = 4$.

N	Locatelli $p = 4$ $\mu = 0.3$	PW linear $p = 4$ $\mu = 0.3$	PW linear $p = 4$ $\mu = 2.0$	PW linear $p = 4$ $\mu = 5.0$	PW linear $p = 4$ $\mu = 10.0/N$	PW linear $p = 4$ $\mu = 1.0/N$	PW linear $p = 3$ $\mu = 10.0/N$
2		99.9	99.9	**100.0**	**100.0**	99.9	**100.0**
3		97.1	99.9	**100.0**	99.7	96.5	**100.0**
4		95.8	99.4	99.1	99.3	93.5	**99.5**
5		99.8	**99.9**	96.3	**99.9**	**99.9**	97.3
6		99.2	**99.9**	99.5	99.6	98.8	99.5
7		24.3	45.9	**60.1**	44.9	22.6	55.4
8		33.1	4.3	1.8	3.4	**36.9**	2.4
9		22.4	**24.4**	2.4	24.2	18.9	11.1
10	26.1	30.2	55.9	**57.6**	55.5	20.7	60.9
11	43.2	49.1	91.3	84.9	76.9	31.7	**95.7**
12	75.1	76.7	98.9	17.0	93.9	57.8	**99.6**
13	82.0	82.7	99.2	98.6	94.6	57.4	**99.4**
14	87.0	91.6	**99.7**	–	95.6	77.0	99.1
15	8.5	**21.9**	9.5	–	7.2	18.2	9.1
16	22.0	23.2	5.5	–	15.5	**27.5**	9.5
17	7.1	3.9	3.7	–	3.8	**7.2**	3.3
18	19.8	19.1	**31.7**	–	24.9	9.9	5.4
19	32.8	32.8	23.5	–	**37.7**	22.3	5.9
20	43.9	45.3	12.6	–	**50.1**	23.9	6.3
21	11.9	14.7	11.6	–	18.4	6.9	**25.5**
22	25.4	26.8	–	–	**30.4**	14.3	0.3
23	23.2	23.2	–	–	**28.8**	11.7	3.6
24	28.1	30.4	–	–	**33.1**	15.8	12.4
25	32.0	33.5	–	–	**34.0**	14.8	–
26	19.4	20.5	–	–	**24.7**	3.8	–
27	1.5	0.6	–	–	1.0	**2.3**	–
28	3.7	3.8	–	–	**3.9**	3.0	–
29	11.7	10.6	–	–	**14.0**	5.2	–
30	0.5	**0.7**	–	–	0.4	0.5	–
31	0.5	0.4	–	–	**1.2**	1.0	–
32	0.7	**2.1**	–	–	0.8	1.4	–
33	1.6	**2.0**	–	–	1.7	1.2	–
34	0.1	**0.4**	–	–	–	0.1	–
35	0.2	0.1	–	–	**0.4**	0.2	–
36	0.4	**0.2**	–	–	0.1	**0.2**	–
37	0.1	–	–	–	–	**0.2**	
38	1.2	1.1	–	–	**1.2**	0.7	–
39	0.1	–	–	–	0.1	**0.6**	–
40	0.2	0.2	–	–	**0.4**	0.3	–

the best values obtained with $p = 4$ were better than the best values obtained with $p = 6$ in 82% of cases. In addition to values $p = 6$ and $p = 4$, we made one trial set with piecewise linear penalty and $p = 3$ (see Table 2). Although $p = 3$ worked well for small N, the putative global minimum was not found within 1 000 trials with $N > 24$. In Table 2 we have used blue pen to point out those results with $p = 3$ that were better than the corresponding results with $p = 4$.

Now we start to study formulations with diameter penalizations. The results with different formulae and parameter values are given in Tables 3 – 5. Here, as before, "Locatelli" stands for the results of Locatelli and Schoen [17]. In addition, "smooth" stands for the smooth penalty (3) ran with `LDGB`, "lin+max" stands for formula (5) and "PWlin+max"stands for formula (6). In Tables 3 and 4 we study the case with $p = 6$ and in Table 5 we have results for $p = 4$. We have used the value $\beta = 1.0$ in all our trials. The best values at all `LDGB`'s results with different values of p are bolded.

None of the formulae and parameter combinations tested gave us the putative global optimum with every N within our 1000 test trials. With $p = 6$ the overall best results were obtained with formula (6) with parameters $\mu = 2.0$ and $D = 3.0$ (see Tables 3 and 4). However, this combination failed to find the putative global optimum (at least once) with six different Ns. In that sense the best results were obtained with the same formula but with $\mu = 10.0/N$ and $D = 5.0$, in which case we failed only with three different values of N. In addition, with $p = 4$ the overall best performance was obtained with formula (6). Here the best parameters were $\mu = 10.0/N$ and $D = 3.0$ and the number of failures in finding the putative global optimum was five (see Table 5). The same formula with parameters $\mu = 0.3$ and $D = 3.0$ succeeded in finding the putative global optimum in all but two different values of N.

As before, we compare Locatelli's results to the similar smooth formulation with $p = 6$, $\mu = 0.3$ and $D = 3$, and to formula (6) with $p = 4$, $\mu = 0.3$ and $D = 3$ both ran with `LDGB` (blue pen in Tables 3 and 5). Now, with $p = 6$ Locatelli's results were better with only three different values of N and with $p = 7$ they were better only in seven cases. This result — especially, when taking account the specialized starting point generation procedure used in [17] — means that with these more complex formulae the usage of the solver `LDGB` clearly gives us a small advantage.

Next we compare the smooth penalty (3) to the nonsmooth ones (5) and (6). In Table 3 we have again used red pen for those results of nonsmooth formulae which give better or equal results to the smooth formula with the same parameters. Here we can conclude that both the nonsmooth formulae give some improvement. When comparing formulae (5) and (6) there was not a big difference between the success rate of the formulae with the same parameters. Nevertheless, as before (6) was slightly better (see Table 3).

The formulae with diameter penalizations usually gave better results than that with only the linear or the piecewise linear penalization when the same parameter combinations were compared (see Tables 1 – 5). The clear exception for this rule is the smooth formulation with $D = 1.5$, where no successful runs were made with $N \geq 22$. The differences in the success rate are sometimes enormous. Two interesting examples occurs with $N = 6$ and $N = 38$. With these cases the optimal structure of the cluster is

Table 3: Success rate with diameter penalization and $p = 6$.

N	Locatelli $p=6$ $\mu=0.2$ $D=3.0$	smooth $p=6$ $\mu=0.3$ $D=1.5$	smooth $p=6$ $\mu=0.3$ $D=3.0$	lin+max $p=6$ $\mu=0.3$ $D=1.5$	lin+max $p=6$ $\mu=0.3$ $D=3.0$	PWlin+max $p=6$ $\mu=0.3$ $D=1.5$	PWlin+max $p=6$ $\mu=0.3$ $D=3.0$
2		**100.0**	99.9	**100.0**	**100.0**	**100.0**	**100.0**
3		99.9	98.2	**100.0**	97.2	99.8	97.0
4		99.0	95.5	**99.4**	94.6	99.1	94.8
5		97.8	99.6	94.3	99.6	97.4	99.7
6		65.8	12.7	98.2	11.7	98.0	11.8
7		**54.2**	28.7	49.0	31.2	44.8	31.0
8		20.6	54.8	26.0	52.3	22.2	53.7
9		11.5	29.8	17.6	29.5	14.4	28.8
10	12.2	53.2	19.2	36.4	20.2	33.4	21.4
11	14.9	73.3	24.5	60.6	29.3	64.6	31.1
12	22.5	**87.2**	37.5	83.1	42.9	84.6	45.9
13	21.9	**94.6**	35.2	89.1	39.1	89.0	37.9
14	39.8	**97.4**	56.0	95.9	61.2	96.1	63.2
15	22.3	0.4	24.4	2.1	**27.9**	1.8	22.0
16	13.4	0.2	15.7	2.2	12.7	1.4	13.2
17	5.4	**6.4**	5.9	4.4	5.2	2.8	4.1
18	3.3	–	6.0	**23.0**	6.9	20.7	7.2
19	4.8	0.3	8.1	9.1	7.6	9.0	9.6
20	8.0	1.0	11.6	12.1	10.5	12.7	13.6
21	4.5	1.9	5.0	13.0	7.1	12.5	6.8
22	8.2	–	11.4	18.4	10.2	19.9	10.2
23	4.3	–	6.9	12.4	4.9	13.6	5.7
24	9.7	–	10.1	22.6	12.7	25.7	9.9
25	16.0	–	17.9	29.3	16.3	28.5	18.1
26	5.6	–	7.2	30.5	7.7	**30.6**	8.9
27	0.1	–	–	–	0.2	–	–
28	0.6	–	0.6	–	0.5	0.1	0.9
29	0.8	–	1.3	1.2	0.5	0.8	1.1
30	–	–	–	–	–	–	–
31	0.1	–	0.4	0.3	0.2	0.3	–
32	–	–	–	0.1	0.1	–	0.2
33	–	–	0.1	1.5	0.2	1.6	0.2
34	–	–	–	–	–	–	–
35	0.1	–	–	–	–	–	–
36	–	–	0.2	–	–	–	–
37	–	–	–	–	–	**0.1**	–
38	8.9	–	8.0	–	6.7	–	6.7
39	–	–	–	–	–	–	–
40	–	–	–	–	–	–	–

Table 4: Number of successes with diameter penalization and $p = 6$ (cont.).

N	PWlin+max	PWlin+max	PWlin+max	PWlin+max	PWlin+max	PWlin+max
	$p = 6$	$p = 6$	$p = 6$	$p = 6$	$p = 6$	$p = 6$
	$\mu = 1.0$	$\mu = 2$	$\mu = 10.0/N$	$\mu = 10.0/N$	$\mu = 1.0/N$	$\mu = 1.0/N$
	$D = 3.0$	$D = 3.0$	$D = 3.0$	$D = 5.0$	$D = 3.0$	$D = 5.0$
2	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
3	98.3	99.2	**99.6**	99.4	97.4	95.1
4	96.9	98.0	97.9	**98.1**	93.9	92.7
5	99.7	**99.8**	**99.8**	99.6	99.6	99.7
6	41.2	**99.1**	97.9	98.6	10.4	6.4
7	30.8	**32.7**	31.3	31.6	28.5	24.5
8	46.1	35.2	43.3	39.8	**56.9**	51.7
9	**31.1**	28.2	27.1	27.0	28.2	23.6
10	30.5	**38.7**	31.8	24.3	18.2	11.6
11	45.7	**57.7**	43.1	39.3	22.4	13.8
12	71.2	**85.0**	69.7	61.1	28.2	15.4
13	67.7	**87.0**	61.7	55.8	25.1	14.9
14	80.4	**94.2**	75.1	70.2	45.8	31.4
15	15.9	22.1	15.2	17.3	**26.3**	18.2
16	16.8	16.1	**20.2**	17.6	15.3	10.5
17	3.1	3.0	4.6	3.7	**5.4**	5.0
18	14.6	**18.9**	11.5	9.2	2.5	1.7
19	13.1	**19.9**	11.2	11.3	3.1	2.2
20	20.3	**29.3**	16.1	15.3	5.8	4.1
21	11.7	**16.1**	9.0	6.7	3.2	1.6
22	19.2	**23.9**	14.9	13.1	6.4	2.1
23	10.9	**15.1**	6.4	5.1	2.9	2.2
24	20.4	**32.0**	12.0	11.1	8.2	2.7
25	27.0	**33.1**	20.8	13.3	11.5	3.3
26	15.4	**24.3**	8.1	4.7	4.2	0.4
27	–	–	–	0.3	0.3	**0.4**
28	0.2	0.7	0.6	**1.2**	–	0.9
29	2.4	**4.0**	0.6	1.3	0.4	0.2
30	**0.2**	–	–	–	0.1	**0.2**
31	**0.7**	0.3	–	0.5	0.2	0.1
32	–	0.1	–	**0.3**	0.1	**0.3**
33	1.1	**2.1**	0.3	0.2	0.1	–
34	–	**0.1**	–	–	–	**0.1**
35	–	–	–	–	**0.1**	–
36	–	–	–	**0.3**	–	0.1
37	–	–	–	**0.1**	–	–
38	4.7	4.7	6.8	0.4	**8.1**	–
39	0.1	0.1	**0.4**	0.3	–	**0.4**
40	**0.1**	–	–	**0.1**	–	–

Table 5: Number of successes with $p = 4$ and diameter penalization.

N	Locatelli	lin+max	PWlin+max	PWlin+max	PWlin+max	PWlin+max	PWlin+max
	$p = 4$	$p = 4$	$p = 4$	$p = 4$	$p = 4$	$p = 4$	$p = 4$
	$\mu = 0.2$	$\mu = 0.3$	$\mu = 0.3$	$\mu = 0.3$	$\mu = 0.3$	$\mu = 10.0/N$	$\mu = 1.0/N$
	$D = 3.0$	$D = 3.0$	$D = 1.5$	$D = 3.0$	$D = 5.0$	$D = 3.0$	$D = 3.0$
2		**100.0**	**100.0**	**100.0**	**100.0**	**100.0**	**100.0**
3		98.2	**100.0**	98.0	98.4	99.7	98.1
4		96.4	**99.8**	97.0	97.1	99.3	96.6
5		99.9	99.7	**100.0**	99.8	99.8	99.8
6		98.7	98.8	99.3	99.3	**99.8**	98.7
7		28.0	**52.6**	26.4	25.7	41.0	30.9
8		35.4	3.0	32.6	32.6	3.7	**40.7**
9		29.2	1.6	**29.8**	25.9	23.6	26.3
10	26.7	37.6	36.8	39.9	32.7	**55.6**	29.7
11	41.4	49.5	76.0	57.2	50.5	**79.3**	44.8
12	71.7	81.3	83.6	82.0	77.3	**94.1**	67.9
13	78.6	80.2	90.1	84.5	82.3	**94.7**	71.8
14	85.7	89.2	**97.8**	91.6	91.7	97.2	81.0
15	10.8	14.8	6.7	22.7	**24.3**	5.4	14.5
16	22.1	22.5	5.0	24.3	26.0	16.2	**28.7**
17	8.0	4.9	5.0	4.0	**5.4**	2.4	5.0
18	18.5	17.2	21.2	19.6	20.9	**29.0**	13.6
19	25.8	27.8	–	31.8	31.9	**36.2**	24.9
20	36.4	39.8	–	44.3	42.2	**46.8**	29.8
21	13.0	15.7	–	16.1	15.5	**16.6**	12.7
22	23.9	25.2	–	25.9	24.8	**28.0**	21.1
23	20.2	22.7	–	24.7	19.7	**25.2**	15.8
24	27.5	30.2	–	32.6	29.3	**38.1**	24.0
25	34.6	33.2	–	**36.9**	34.7	36.8	29.1
26	21.9	23.7	–	26.0	24.3	**28.7**	14.1
27	0.5	**0.6**	–	0.4	0.5	0.3	0.4
28	2.4	1.6	–	2.0	**2.7**	1.7	2.0
29	10.0	12.0	–	12.3	11.6	**12.6**	5.1
30	0.4	–	–	**0.3**	**0.3**	0.2	0.2
31	–	**1.2**	–	1.1	0.7	0.8	0.8
32	–	0.4	–	**0.8**	0.7	**0.8**	0.3
33	0.1	1.0	–	0.4	1.1	**1.4**	0.5
34	–	–	–	**0.1**	–	–	–
35	–	–	–	–	–	–	–
36	0.1	**0.3**	–	0.1	–	–	0.1
37	–	**0.1**	–	**0.1**	–	–	–
38	4.4	**2.8**	–	1.5	1.0	1.4	2.1
39	–	**0.2**	–	0.1	0.1	–	**0.2**
40	0.1	–	–	–	**0.2**	0.1	0.1

face-centered cubic (FCC) which is claimed to be one of the most difficult structures to find especially with large N (see e.g. [15, 17]). In both of these cases the formulae with diameter penalizations made large differences: with $N = 6$, $p = 6$ and $\mu = 0.3$ the formulations without diameter penalization gave only about 10% success, while with nonsmooth diameter penalizations the success rates were around 98%; with $N = 38$, $p = 6$ and $\mu = 0.3$ the corresponding values were less than 0.5% and 6.7%. Nevertheless, there are some results where the differences are the other way around: that is the case, for example, with $N = 15$, $p = 6$, $\mu = 0.3$ and $D = 1.5$. Moreover, when comparing the best values obtained with and without diameter penalization the choice between better formulae is not so clear. Indeed, the similar improvement as above for $N = 6$ and $p = 6$ can be done with only (piecewise) linear penalty by setting $\mu = 5$ (see Table 1). For $N = 38$ this kind of effect was not observed.

With smooth formulation the smaller D usually gave good results with small N (say $N < 20$) but the number of successes decreased dramatically when N increased. For example, with $p = 6$ the success rate with $D = 3$ was always better than that with $D = 1.5$ when $N \geq 18$ and, as already said, with $N \geq 22$ no successful runs were made with $D = 1.5$. However, when $n \leq 17$ the formula with $D = 1.5$ gave slightly better success rate than that with $D = 3.0$ (see Table 3). With nonsmooth formulations and $p = 6$, this trend was not so obvious. This is probably due to fact that the nonsmooth penalty term does not increase as fast as the smooth one. However, with $p = 4$ and $D = 1.5$ no successful runs were made when $N > 18$ although with smaller N the results with different values of D were comparable. The value $D = 3$ usually gave a little bit better success rate than $D = 5$, although this result might be different if larger clusters were optimized (see Tables 4 and 5, we have used red pen to emphasize the best result with different D).

Finally we say few words about the performance of the optimization algorithm: apart from some exceptional cases the average numbers of function evaluations needed to find a putative global minimum were clearly smaller with any of the formulae with $p = 4$ than those with $p = 6$. When comparing the different formulae the differences were not as clear. Nevertheless, the minimization of the smooth formula (3) usually used less evaluations than the minimization of the nonsmooth formulae (5) or (6), although, the number of average evaluations were of the same magnitude. When comparing formulae (5) with (6) with $p = 4$, $\mu = 0.3$, $D = 3$, formula (6) usually rose above (5). Nevertheless, again the differences were not large ones. In addition, when comparing the formulae with linear or piecewise linear penalties with parameters $p = 6$ and $\mu = 1.0$, the piecewise linear penalty usually used slightly less evaluations than the linear one. However, with $\mu = 0.3$ they used on an average the same amount of evaluations. The most interesting and unexpected result obtained, when comparing the evaluations needed with the piecewise linear penalty and formula (6) with $p = 4$, $\mu = 0.3$, $D = 3$ or $D = 5$: in all cases the average numbers of function evaluations needed to find a putative global minimum were smaller with the more complicated formula (6) than when only the piecewise linear penalty was used. A possible reason for this is that the diameter penalty forces the atoms that are far from the center of the cluster to move close enough more quickly.

5 Conclusions

In this paper we have studied different modifications of the Lennard-Jones potential in order to improve the success rate of finding the global minimum of the original potential. The main interest of the paper was in nonsmooth penalized form of the Lennard-Jones potential. Our goal was to improve the success rate of finding the global minimum of the original problem when using a local search optimization method. The preliminary numerical experiments confirm that with all the penalized formulae the success rates were greatly improved. The results obtained with nonsmooth formulae (5) or (6) were usually a little bit better than those with the smooth penalty (3). In addition, when both the linear penalty and the diameter penalization in (3) were used, the result obtained in our experiments were usually better than those given in [17], in spite of the fact that in [17] the special starting point generation procedure was used. This is probably caused by our solution algorithm LDGB. When comparing the different nonsmooth formulae, the one with the piecewise linear penalty term (that is, formula (6)) seems to be a little bit better one. Nevertheless, the differences were not large.

In our current research we have used a crude multistart method. The future research includes combination of some more sophisticated global optimization method and LDGB and usage of these new modified potentials to solve problems. One interesting idea would be to use an incremental approach (see e.g. [3]) with modified potentials to solve this kind of clustering problems.

References

[1] BAGIROV, A. M., KARASOZEN, B., AND SEZER, M. Discrete gradient method: A derivative free method for nonsmooth optimization. *Journal of Optimization Theory and Applications 137* (2008), 317–334.

[2] BAGIROV, A. M., KARMITSA, N., AND M. M. M. *Introduction to Nonsmooth Optimization: Theory, Practice and Software*. Springer, 2014.

[3] BAGIROV, A. M., UGON, J., AND MIRZAYEVA, H. G. Nonsmooth optimization algorithm for solving clusterwise linear regression problem. *Journal of Optimization Theory and Applications 164* (2015), 755–780.

[4] BELIAKOV, G., MONSALVE TOBON, J. E., AND BAGIROV, A. M. Parallelization of the discrete gradient method of non-smooth optimization and its applications. In *Computational Science — ICCS 2003*, Sloot et. al. , Ed., Lecture Notes in Computer Science. Springer Berlin, Heidelberg, 2003, pp. 592–601.

[5] BYRD, R. H., NOCEDAL, J., AND SCHNABEL, R. B. Representations of quasi-Newton matrices and their use in limited memory methods. *Mathematical Programming 63* (1994), 129–156.

[6] CLARKE, F. H. *Optimization and Nonsmooth Analysis*. Wiley-Interscience, New York, 1983.

[7] DOYE, J. P. K. The effect of compression on the global optimization of atomic clusters. *Physical Review E 62* (2000), 8753–8761.

[8] HAARALA, M. *Large-Scale Nonsmooth Optimization: Variable Metric Bundle Method with Limited Memory*. PhD thesis, University of Jyväskylä, Department of Mathematical Information Technology, 2004.

[9] HAARALA, M., MIETTINEN, K., AND MÄKELÄ, M. M. New limited memory bundle method for large-scale nonsmooth optimization. *Optimization Methods and Software 19*, 6 (2004), 673–692.

[10] HAARALA, N., MIETTINEN, K., AND MÄKELÄ, M. M. Globally convergent limited memory bundle method for large-scale nonsmooth optimization. *Mathematical Programming 109*, 1 (2007), 181–205.

[11] KARMITSA, N., AND BAGIROV, A. Limited memory discrete gradient bundle method for nonsmooth derivative free optimization. *Optimization: A Journal of Mathematical Programming and Operations Research 61*, 12 (2012), 1491–1509.

[12] KOSTROWICKI, J., PIELA, L., CHERAYI1, B. J., AND SCHERAGA, H. A. Performance of the diffusion equation method in searches for optimum structures of clusters of Lennard-Jones atoms. *Journal of Physical Cemistry 95* (1991), 4113–4119.

[13] LAMPARIELLO, F., AND LIUZZI, G. Global optimization of protein-peptide docking by a filling function method. *Journal of Optimization Theory and Applications 164* (2015), 1090–1108.

[14] LEACH, A. R. *Molecular Modelling: Principles and Applications*, 2nd edition ed. Pearson Education Limited, Harlow, England, 2001.

[15] LEARY, R. H. Global optima of Lennard-Jones Clusters. *Journal of Global Optimization 11* (1997), 35–53.

[16] LEMARÉCHAL, C., STRODIOT, J.-J., AND BIHAIN, A. On a bundle algorithm for nonsmooth optimization. In *Nonlinear Programming*, O. L. Mangasarian, R. R. Mayer, and S. M. Robinson, Eds. Academic Press, New York, 1981, pp. 245–281.

[17] LOCATELLI, M., AND SCHOEN, F. Fast global optimization of difficult Lennard-Jones clusters. *Computational Optimization and Applications 21* (2002), 55–70.

[18] LOCATELLI, M., AND SCHOEN, F. Efficient algorithems for large scale global optimization. *Computational Optimization and Applications 26* (2003), 173–190.

[19] MIFFLIN, R. A modification and an extension of Lemaréchal's algorithm for nonsmooth minimization. *Matematical Programming Study 17* (1982), 77–90.

[20] NEUMAIER, A. Molecular modeling of proteins and mathematical prediction of protein structure. *SIAM Rev. 39* (1997), 407–460.

[21] VLČEK, J., AND LUKŠAN, L. Globally convergent variable metric method for nonconvex nondifferentiable unconstrained minimization. *Journal of Optimization Theory and Applications 111*, 2 (2001), 407–430.

[22] WALES, D. J. Rearrangements of 55-atom Lennard-Jones and (C60)55 clusters. *Journal of Chemical Physics 101* (1994), 3750–3762.

[23] YEAK, S. H., NG, T. Y., AND LIEW, K. M. Multiscale modeling of carbon nanotubes under axial tension and compression. *Physical Review B 72* (2005).

ISBN 978-952-12-3193-3
ISSN 1239-1891