Computational technology for the study of atomic-molecular Morse clusters of extremely large dimensions

P Sorokovikov, A Gornov and A Anikin
Matrosov Institute for System Dynamics and Control Theory of SB RAS 134, Lermontov Street, Irkutsk, 664033, Russian Federation

E-mail: sorokovikov.p.s@gmail.com

Abstract. The paper considers the problem of finding low-potential Morse clusters, which reduces to the global minimization of the non-convex potential function. The main difficulty of this class of problems is the astronomical increase in the number of local extremums with increasing dimension. We propose algorithms and computational techniques for the study of Morse clusters. Problems with a small, a medium and a large number of atoms are investigated to check the performance of the developed methods. System computational experiments were performed to search for a global extremum in the Morse model of extremely large dimensions (from 241 to 300 atoms). The authors are not aware of other attempts to carry out system computations for Morse clusters of the specified number of atoms.

1. Introduction
Modern experimental methods of molecular physics (see, for example, [1]) allow researchers to obtain a sufficiently large amount of information about the structure of various substances, which makes it possible to form mathematical models that quite adequately reflect their internal structure. In recent years, a large number of mathematical models have been proposed focused on various types and microstates of substances – metallic clusters, gas clusters, “molecular” clusters, biomolecules, ionic clusters, “binary clusters”, supercooled liquids, and others (see, for example, [2, 3]). By now, according to experts, the number of proposed models has reached about several hundred.

A distinctive feature of atomic-molecular clusters is the non-monotonic dependence of their properties on the number of atoms. As a result, the computational problems arising from the study on models of stable states of substance turned out to be very nontrivial in most cases. From a mathematical point of view, these problems can be considered as the problem of searching for a global minimum of non-convex potential functions. The number of local extremes in these models grows extremely rapidly, depending on the dimension of the clusters, according to various estimates, as an exponent of the square of the number of atoms. However, modern optimization algorithms running on high-performance computing techniques can find the “best of known” solutions that may be global minima.

2. Morse model
The paper considers the problem of finding atomic-molecular Morse clusters [2] of extremely large dimensions. The following box-constrained global optimization problem is considered:
\[
\begin{align*}
& f(x) \to \min, \ x \in B, \ B = \left\{ x \mid x = (x_1, x_2, \ldots, x_n), \alpha_i \leq x_i \leq \beta_i \right\}, \\
& f(x) = \sum_{i=1}^{N} \sum_{j=1}^{N} e^{x(i, j)} \left( e^{x(i, j)} - 2 \right), \\
& r_{ij} = \sqrt{\left(x_{(i-j)+1} - x_{(j-i)+1}\right)^2 + \left(x_{(i-j)+2} - x_{(j-i)+2}\right)^2 + \left(x_{(i-j)+3} - x_{(j-i)+3}\right)^2}.
\end{align*}
\]

Here \( f(x) \) is the non-convex and smooth objective function; \( r_{ij} \) is the distance between particles \( i \) and \( j \); \( \rho \) is the special parameter (\( \rho = 3 \)); \( N \) is the number of atoms, \( n = 3N \) is the dimension of the vector \( x \); \( \alpha, \beta \) are vectors of parallelepiped constraints.

3. Description of computational technology

To solve the problems of optimizing atomic-molecular clusters, a three-stage computational technology has been implemented, which includes various methods from a set of basic at each stage.

The first stage is focused on the stochastic approximation of a neighborhood of a known record point and relies on a set of algorithms-generators of various initial approximations.

At the second stage, the problem of fast descent of the approximations generated at the first stage [4] is solved using one of the “starter algorithms” – algorithms that allow finding low-potential clusters with minimal computational costs. Three variants of “starter algorithms” have been implemented: the modification of B.T. Polyak’s method [5, 6], the “raider method” [7] and the variant of the nonlocal dichotomy method [8, 9].

For local descents implemented in the third stage, the L-BFGS method with a memorization bandwidth of three is used (see, for example, [10, 11, 12]).

Also, to solve the problem of optimizing atomic-molecular clusters, the modification of the genetic algorithm (see, for example, [13, 14, 15]) was proposed and implemented.

3.1. Generators of starting approximations

To obtain new approximations for local descent, a specialized set of 7 generators of quasi-random vectors is used:

- LG – “level” generator;
- AG – “averaged” generator;
- GG – “gradient” generator;
- LAG – combined “level-averaged” generator;
- LGG – combined “level-gradient” generator;
- AGG – combined “averaged-gradient” generator;
- LAGG – combined “level-averaged-gradient” generator.

The LG algorithm (“level” generator) implements the classical scheme for generating a quasi-random vector using one of the standard algorithms for generating quasi-random numbers (the URAND algorithm [16]). For each component of the vector, a random number is searched for in the double-width interval, centered at the current record point. The implemented modification of the algorithm consists in finding a start vector in which the value of the function is not too large, since the operation of local algorithms for unsuccessful initial points may be impossible. The algorithm iteratively searches for a starting approximation. If the starting point could not be found for the given number of iterations, the last generated one is taken as the result of the algorithm.

The AG algorithm (“averaged” generator) implements the idea of using the experience of searching for local extrema already found by the algorithm. The iteration accumulates the average value of the shift (“drift”) of points, which occurs as a result of the work of local optimization algorithms. When
generating a new approximation, this allows us to take into account the individual “need” of each variable to change, not allowing the interval for all points to be excessively expanded due to several “greedy” ones. A new approximation is quasi-randomly generated from an interval that is equilaterally increased from the average by the value of the algorithmic parameter. Also, to cut off too inconvenient points, the same mechanism is used as in the LG algorithm, the parameters of which are used to control the AG algorithm.

The GG algorithm (“gradient” generator) implements the idea of the informativeness and utility of the gradient function at points far from extremes. The algorithm allows, having executed a certain number of iterations, to find among the generated points an approximation that has the lowest norm of gradient.

The LAG algorithm (“level-averaged” generator) is designed as a combination of LG and AG generators, switched alternately on even (AG) and odd (LG) iterations of the algorithm. In this case, the first few iterations (by default, 4) the AG algorithm is not included since at this stage of the computation a situation with an inadequate estimate of the averaged “drift” is possible.

The LGG algorithm (“level-gradient” generator) also alternately turns on LG and GG through iteration and does not depend on the “drift” parameters.

The AGG algorithm (“averaged-gradient” generator) is designed as a combination of AG and GG generators, switched alternately on even (AG) and odd (GG) iterations of the algorithm. In this case, the first few iterations (by default, 4) the AG algorithm is not included for the same reason as in the LAG generator.

The stochastic selection of the generator algorithm is implemented in the LAGG generator (“level-averaged-gradient” generator). The LG, AG and GG modules are used as basic bricks. The choice of one of them at each iteration is performed with the same probability. An exception is the use of the AG module, which becomes possible after the 4th iteration when adequate estimates of the “drift” appear.

3.2. “Algorithms-starters”

“Starters” are designed to provide the initial stage of local descent, in which regular methods (quasi-Newton, conjugate gradients, Nesterov’s “optimal” methods, direct dual methods, and others) with competitive characteristics “on average” may not be very effective.

Besides, with the stochastic generation, starting points are possible in the region of the variable set, in which there are “computational peculiarities”. For example, the existence of points at which the values of a function and/or gradient are difficult to imagine within the framework of the machine arithmetic used. An attempt to overcome similar, typical in computational practice, real optimization problems can be made based on heuristic methods that don’t guarantee, in the general case, solutions, but simplify and/or speed up the start of the computation steps.

Modification of the Polyak’s method (“Polyak”), decomposition gradient type method (“Raider”) and modification of the multidimensional dichotomy method (“Dichotomy”) are used as “starters” in the software package.

3.2.1. Modification of the Polyak’s method. The method proposed by B.T. Polyak in 1969 [5] is one of the oldest (sub)gradient algorithms. To its shortcomings, experts rightly attribute the availability of mandatory information about the value of a function at an extreme point, which led to its oblivion for many years. But in our case, for carrying out preliminary descent from points far from optimal points, the method may be useful due to its simple structure and high computational stability. The proposed modification (see, for example, [6]) allows one to obtain monotonic, by the value of the function, successive approximations by crushing the step in the direction of the gradient. In the implemented version, the specificity of the problem is used – knowledge of the fact that the minimized energy functional must necessarily be negative (maximized energy functional must be positive) and the presence of the record value of the function as a lower estimate for the calculation at iteration.
3.2.2. **Raider method.** The decomposition gradient method is called by us the raider method (see, for example, [7]). The main idea of the method is to work at each iteration not with a full gradient, but only with its largest components. At the same time, there is the possibility of rapid advancement over the “active” (“energetic”) components, which in this case are not constrained by the “sluggish” components. The method allows, in some cases, to promptly correct those cluster components that were generated extremely unsuccessfully and thereby create a convenient starting position for regular methods of local descent.

3.2.3. **Modification of the multivariate dichotomy method.** This algorithm (see, for example, [8]) can be interpreted as a generalization of the coordinate search method using gradient information. The basic idea of the algorithm is to regularly compress the search set (parallelepiped) by vigorously clipping all its quadrants into which the gradient is directed. The sequence of approximations generated by the algorithm is generally non-monotonic. The implemented stochastic algorithm of the heuristic type can be interpreted as a modification of the method of centered sections (see, for example, [9]). The algorithm, in some cases, may not find points at all that improve the starting position, but with a good arrangement of the generated approximations, it can be very useful, including in non-local progress.

3.3. **Genetic algorithm**

Global optimization methods can be divided into single-point and multipoint (population) algorithms. Population methods, unlike single-point algorithms, operate on a set of points (population). Particular cases of population methods are evolutionary (genetic) algorithms that simulate natural selection. The general scheme of genetic algorithms is shown in figure 1.

![Figure 1. The general scheme of genetic algorithms.](image)

Evolutionary operators implemented as follows (table 1).

| Operator   | Realization                                      |
|------------|--------------------------------------------------|
| Selection  | Tournament selection [13; 14]                    |
| Crossover  | $k$-point crossover [13; 15]                     |
| Mutation   | $x = \text{uniform}(B_r(x))$ – stochastic selection of a point in the ball of random radius $r$ centered at the point $x$ |
| Truncation | Selection of the $p$ fittest individuals from the current population and |
the mutant population into a new population

The values of genetic algorithm parameters are presented in table 2.

| Parameter                        | Value | Comment                                                                 |
|----------------------------------|-------|-------------------------------------------------------------------------|
| Population size, \( p \)         | 10    | The population size is constant.                                        |
| Tournament size                  | 3     |                                                                          |
| The number of breakpoints, \( k \) | 3     | Breakpoints are selected randomly.                                      |
| Parents size                     | 5     | 5 parents are selected, from which randomly constructed 5 pairs.         |
| Children size                    | 5     | each pair gives 2 persons, the best of which is counted as a single child.|
| Mutation probability             | 0.9   | Mutation with specified probability applies only to the children.       |

4. Testing of algorithms and computational technology

Using the implemented algorithms and computational technology, computations were performed to search for optimal configurations for the atomic-molecular Morse potential. Computational experiments were performed using a server computer with the following characteristics: 2 x Intel Xeon E5-2680 v2 2.8 GHz (20 cores, 40 threads); 128 Gb DDR3 1866 MHz.

4.1. Verifying the performance of the proposed methods

A performance test of three-stage computational technology for clusters of 5–240 atoms was performed. The resulting value was compared with the best of known one: \( \delta = |\tilde{f} - f| \), where \( \tilde{f} \) is the value found by the algorithm, \( f \) is the probable value of the global minimum. The values obtained during testing coincided with the well-known results published in [3].

The implemented genetic algorithm was also tested for these clusters. Table 3 shows the results of computational experiments.

| \( N \) | \( \delta \) | \( N \) | \( \delta \) |
|---------|-------------|---------|-------------|
| 5       | 0.000000    | 20      | 0.000000    |
| 8       | 0.000000    | 23      | 0.000000    |
| 11      | 0.000000    | 26      | 0.284601    |
| 14      | 0.000000    | 29      | 0.860821    |
| 17      | 0.000000    | 32      | 1.364192    |

The obtained values coincided with the best known for problems up to 23 atoms, inclusive. For problem sizes of more than 23 atoms, errors \( \delta > 0 \) were observed in the solutions achieved.

To study models with an extremely large number of atoms, three-stage computational technology was used, since with its help problems of smaller dimensions were successfully solved.
4.2. Testing of computational technology for extremely large cluster dimensions

Extremely large problems for clusters with the number of atoms from 241 to 300 were investigated (solutions are available only for clusters up to 240 atoms in the public Cambridge Energy Landscape Database [3] and other sources).

For each generator of the initial approximations and each “starter algorithm”, computations were performed for clusters with the number of atoms from 241 to 300. As a stopping criterion, an excess of the specified time limit (86400 seconds, i.e. 24 hours) was used. Table 4 shows the results of computations for the problem with a dimension of 262 atoms.

| Generator | Starter | Value     | Starter | Value     |
|-----------|---------|-----------|---------|-----------|
| 1 (LG)    | Raider  | -3158.18043 | Dichotomy | -3159.31282 | Polyak | -3150.27152 |
| 2 (AG)    | Raider  | -3152.97692 | Dichotomy | -3163.20340 | Polyak | -3151.86394 |
| 3 (GG)    | Raider  | -3160.45951 | Dichotomy | -3160.96551 | Polyak | -3151.26523 |
| 4 (LAG)   | Raider  | -3159.63533 | Dichotomy | -3163.20340 | Polyak | -3152.68241 |
| 5 (LGG)   | Raider  | -3159.19448 | Dichotomy | -3160.09538 | Polyak | -3151.08223 |
| 6 (AGG)   | Raider  | -3162.74753 | Dichotomy | -3159.60526 | Polyak | -3150.33721 |
| 7 (LAGG)  | Raider  | -3164.56854 | Dichotomy | -3164.56854 | Polyak | -3153.24328 |

As can be seen from table 4, among all the tested variants, there are two leading computational technologies with a combination of the starting approximation generator number 7 (LAGG) and the raider method (first variant), modification of the dichotomy method (second variant) as a “starter”.

Figure 2 shows two plots with the results of computational experiments. The first plot (left) shows a comparison of “starter algorithms” (LAGG was chosen as the generator of initial approximations in this computation). The second plot (right) shows the dependence of the best-found values on the number of atoms.

![Figure 2](image-url)
As can be seen from figure 2, the best result was shown by the raider method and modification of the dichotomy method. The modification of Polyak’s method is worse than its competitors.

Table 5 presents the best-found values (probably “best of known“). We didn’t succeed in comparing the results of the computations with the works of other authors, because we are not aware of other attempts to carry out system computations for Morse clusters of the specified dimensions.

Table 5. The best-found values (number of atoms: from 241 to 300).

| N   | Value       | N   | Value       | N   | Value       |
|-----|-------------|-----|-------------|-----|-------------|
| 241 | -2852.824893 | 266 | -3226.262063 | 286 | -3514.900966 |
| 243 | -2882.570362 | 268 | -3256.769025 | 288 | -3543.544683 |
| 245 | -2910.707950 | 270 | -3283.366175 | 290 | -3576.270498 |
| 247 | -2940.026081 | 272 | -3311.874597 | 292 | -3607.169943 |
| 249 | -2971.203337 | 274 | -3340.319691 | 294 | -3636.816229 |
| 250 | -2985.771711 | 276 | -3368.675534 | 296 | -3664.748040 |
| 255 | -3058.984226 | 278 | -3398.079040 | 297 | -3680.123441 |
| 260 | -3133.960446 | 280 | -3427.879852 | 298 | -3695.432549 |
| 262 | -3164.568536 | 282 | -3455.618262 | 299 | -3713.203750 |
| 264 | -3195.212792 | 284 | -3488.811918 | 300 | -3728.226966 |

Based on the conducted computational experiments, we can formulate the following findings:

- Using the proposed genetic algorithm, it was possible to achieve the best of known values for clusters up to 23 atoms, inclusive.
- On average, the raider method and modification of the multivariate dichotomy method showed the best results. The best-found values were obtained using these algorithms.
- The modification of the Polyak’s method is worse than the raider method and modification of the multivariate dichotomy method.
- Among the starting approximation generators, the “level-averaged-gradient” generator (LAGG) is leading.

5. Conclusion

The paper proposed algorithms and computational techniques for the study of atomic-molecular Morse clusters. The performance of the developed methods for problems of small, medium and large dimensions was tested.

Using the developed algorithms and software, system computational experiments were performed comparing the proposed initial approximation generators and “starter algorithms”. In the course of testing, probable optimal configurations were obtained for clusters of very large dimensions (from 241 to 300 atoms).

Acknowledgments

The reported study was funded by Russian Foundation for Basic Research, project No. 19-37-90065.

References

[1] Brooks R L 2013 The fundamentals of atomic and molecular physics (New York: Springer)
[2] Doye J P K and Wales D J 1997 J. Chem. Soc. Faraday Trans. 93(24) 4233–43
[3] The Cambridge Energy Landscape Database at: http://www.wales.ch.cam.ac.uk/CCD.html
[4] Wales D J and Doye J P K 1997 J. Phys. Chem. A 101(28) 5111–6
[5] Polyak B T 1969 Computational Mathematics and Mathematical Physics 9(3) 14–29
[6] Gornov A Yu and Anikin A S 2015 Conference proceedings “Lyapunov readings 2015” pp 20
[7] Gornov A Yu 2018 Conference proceedings “Lyapunov readings 2018” pp 28
[8] Gornov A Yu, Andrianov A N and Anikin A S 2016 Proceedings of the International Workshop “Situational Management, Intellectual, Agent-based Computing and Cybersecurity in Critical Infrastructures 2016” pp 76
[9] Levin A Yu 1965 Reports of the Academy of Sciences of the USSR 160(6) 1244–7
[10] Liu D C and Nocedal J 1989 Mathematical programming 45(1-3) 503–28
[11] Mokhtari A and Ribeiro A 2015 The Journal of Machine Learning Research 16(1) 3151–81
[12] Zhu C et al 1997 ACM Transactions on Mathematical Software (TOMS) 23(4) pp 550–60
[13] Whitley D 1994 A genetic algorithm tutorial Statistics and computing 4(2) pp 65–85
[14] Back T 1996 Evolutionary Algorithms in Theory and Practice (Oxford: University Press)
[15] Mitchell M 1999 An Introduction to Genetic Algorithms (Cambridge: Cambridge MIT Press)
[16] Forsait D et al 1980 Machine methods for mathematical calculations (Moscow: Mir)