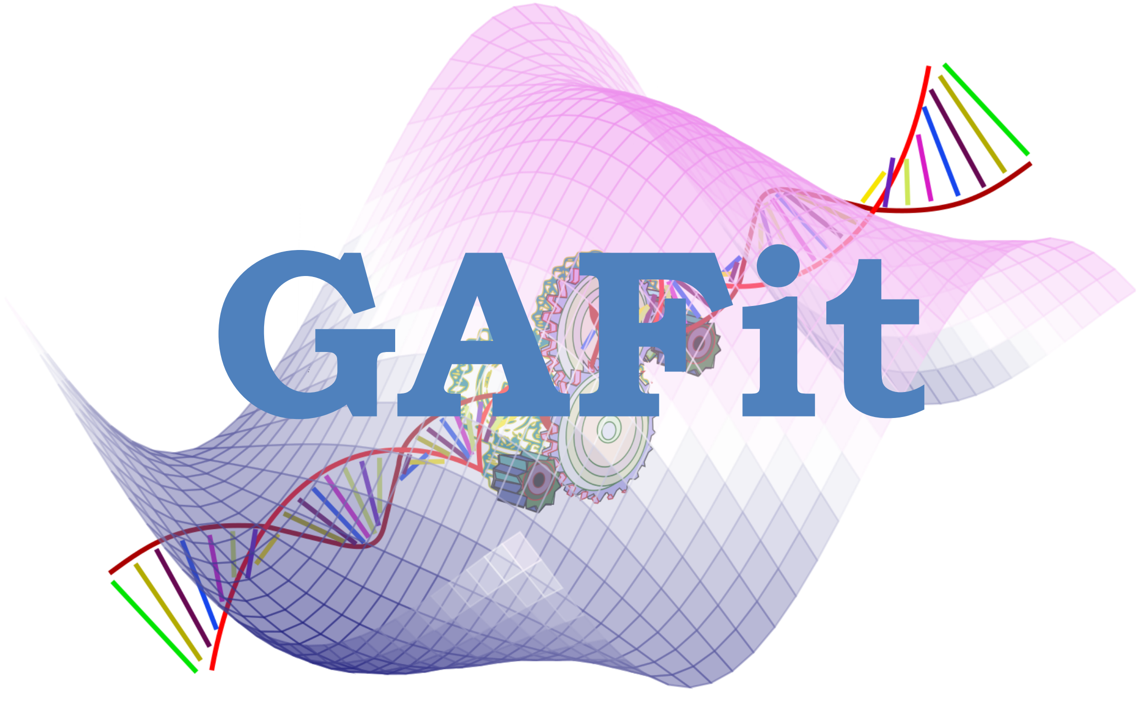
A Genetic Algorithm program for fitting potential energy surfaces



Simplified user guide

Contents

1. Introduction.. 3

2. GAFit citation.. 3

3. Installation.. 4

4. Fitting pairwise intermolecular potentials.................. 6

5. Interface with the CHARMM program...................... 15

6. Interface with the MOPAC program.......................... 22

7. Fitting a user-defined function to a set of data........... 30

8. Recommendations... 33

Appendix.. 35

1. Introduction

GAFit is a software designed to help users parameterize force fields and potential energy surfaces. It may also be used for other optimization problems. The program uses a Genetic Algorithm (GA) to perform least-squares nonlinear fittings. In general, the fittings are conducted by minimizing one of the following objective or merit functions, χ^2

$$\chi^2(\mathbf{a}) = \sum_{i=1}^{N} [y_i - f(x_i; \mathbf{a})]^2 \times w_i \qquad (1)$$

$$\chi^2(\mathbf{a}) = \sum_{i=1}^{N} \left[\frac{y_i - f(x_i; \mathbf{a})}{y_i}\right]^2 \times w_i \qquad (2)$$

In these equations, N is the total number of data points (x_i, y_i) used for the fitting, **a** is a collective variable formed by n fitting parameters and $f(x_i; \mathbf{a})$ is the value of the model function at x_i. The square difference between y_i and the corresponding model value, in absolute (eq 1) or relative (eq 2) terms, can be multiplied by a weight (w_i) attributed to each data point.

GAFit has been written in several programming languages (Fortran 90, C, Perl and Java) and can run in all Linux based workstations.

The present manual is written as a simplified guide or tutorial, and illustrates the main applications of GAFit through some examples. For these types of applications, we tried to make the program as handy as possible. The software package also includes some advanced features not covered in this guide, which extend significantly the range of applicability of the program. They are described in a detailed manual included in the distribution.

The users are encouraged to contact the authors for help with GAFit. Comments and suggestions are also very welcome.

2. GAFit citation

The main features of GAFit are described the following papers:

1. R. Rodríguez-Fernández, F. B. Pereira, J. M. C. Marques, E. Martínez-Núñez and S. A. Vázquez, “GAFit: a general-purpose, user-friendly program for fitting potential energy surfaces based on genetic algorithms”, submitted to publication in Computer Physics Communications.
2. J.M.C. Marques, F.V. Prudente, F.B. Pereira, M.M. Almeida, A.M. Maniero, C.E. Fellows, “A new genetic algorithm to be used in the direct fit of potential energy curves to ab initio and spectroscopic data”, J. Phys. B: At. Mol. Opt. Phys., 41 (2008) 085103.

Please, cite these articles in every scientific work that reports results obtained with GAFit.

3. Installation

The configuration, compilation and installation phases are done by the GNU autotools utilities:

```
tar -xvzf gafit-VERSION.tar.gz
cd gafit-VERSION
./configure
make
make install
```

The compilation generates the executable “gafit”, as well as a series of utility programs. All these binaries go into $HOME/bin and other files (tests and documentation) into $HOME/share. To install it into /usr/local (note that you need superuser permissions.), use:

```
./configure --prefix=/usr/local
make
sudo make install
```

To force a Fortran compiler (e.g. ifort), use:

```
./configure FC=ifort
```

To force a C compiler (e.g. icc), use:

```
./configure FC=icc
```

You can also employ any combination of the above:

```
./configure --prefix=/usr/local FC=ifort CC=icc
```

Many of the binaries created in the $HOME/bin directory are used for internal manipulations. For details, the reader may consult the extended manual. Here, we only list the most relevant utilities for the examples shown in this simplified user guide:

bedit: interactive editor for files *atom2type.txt*, *charges.txt* and *bounds.txt*.
chmfinal: used for the GAFit-CHARMM interface. Evaluates the objective function and writes a file with a comparison between the target data and the calculated values.
fitview: generates data for plotting the parameterized pairwise potentials as well as a comparison with the target intermolecular potentials.
gafit: main executable
mvariable: used to fit data to a generic multivariable function specified in the *job.txt* file.
mvtest: after finishing a multivariable fitting, this utility can be used to evaluate the differences between the target and the calculated data.
needle: generates the files *atom2type.txt* and *charges.txt*.

4. Fitting pairwise intermolecular potentials

For this type of application, we will consider, as an example, the system of Xe interacting with the uracil-Li$^+$ cation (see Figure 1). GAFit needs the following data to parameterize two-body intermolecular potentials.

(1) A single file containing a series of geometries for two interacting molecules, specified in Cartesian coordinates (in Å). The default name for this file is *geometries.txt*. All the geometries must follow the same atom numbering and have xyz format:

```
14

C 0.000000 0.000000 0.000000
N 0.000000 0.000000 1.362384
C 1.155049 0.000000 2.139980
N 2.320779 0.000000 1.355224
C 2.402238 0.000000 -0.011058
C 1.159894 0.000000 -0.718415
O 1.170643 0.000000 3.357110
O 3.545236 0.000000 -0.554967
Li 5.008730 0.000000 -1.523927
H 3.191434 0.000000 1.888428
H 1.154019 0.000000 -1.800728
H -0.977216 0.000000 -0.473394
H -0.873789 0.000000 1.886165
Xe 17.515861 0.000000 -9.804728
14

C 0.000000 0.000000 0.000000
N 0.000000 0.000000 1.362384
C 1.155049 0.000000 2.139980
N 2.320779 0.000000 1.355224
C 2.402238 0.000000 -0.011058
C 1.159894 0.000000 -0.718415
O 1.170643 0.000000 3.357110
O 3.545236 0.000000 -0.554967
Li 5.008730 0.000000 -1.523927
H 3.191434 0.000000 1.888428
H 1.154019 0.000000 -1.800728
H -0.977216 0.000000 -0.473394
H -0.873789 0.000000 1.886165
Xe 15.848244 0.000000 -8.700621
14
...
```

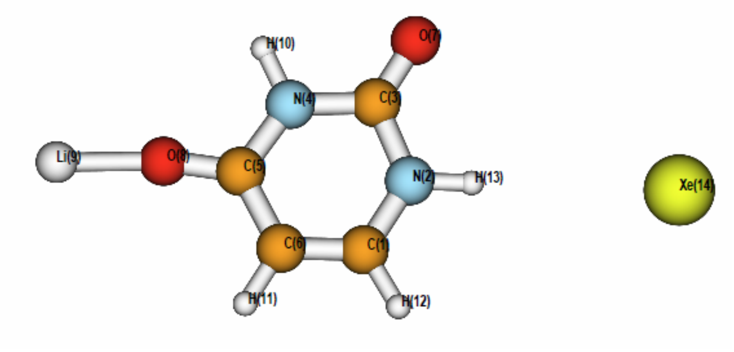


Figure 1. Atom numbering for Xe interacting with the uracil-Li$^+$ cation.

(2) A single file containing the target interaction energies (first column), specified in kcal/mol, as well as the corresponding weights (second column):

```
-0.00885  1
-0.01544  1
-0.02934  1
-0.06465  1
-0.17906  1
-0.24340  1
-0.34349  1
-0.50651  1
-0.65862  1
-0.86925  1
-1.16044  1
-1.58535  1000
-2.19203  1000
-2.72424  5000
-3.37999  5000
-4.15432  5000
-5.00328  12000
-5.77258  22000
-6.03815  35000
-6.13657  35000
-5.96945  12000
-5.39592  9200
-4.21698  9000
-2.15687  5000
1.16822  200000
6.28815  1000
13.94390  1000
25.15173  1
41.30632  1
64.31607  1
96.77263  1
142.12125  60
204.88776 800
...
```

The default filename is *energies.txt*. For this type of fittings, GAFit employs eq 2 as the default objective function.

(3) A file specifying the atom numbering and the types of atoms (default filename: *atom2type.txt*). The first line specifies the number of atoms in molecule 1, that is, the uracil-Li$^+$ cation according to our atom numbering, as well as the total number of atoms (14). The following lines specify, for each atom, the atom numbering (first field), the atomic symbol (second field), and the atom type (third field). In our example, this file is:

```
13  14
    1    C      1
    2    N      2
    3    C      3
    4    N      4
    5    C      5
    6    C      6
    7    O      7
    8    O      8
    9   Li      9
   10    H     10
   11    H     11
   12    H     12
   13    H     13
   14   Xe     14
```

Notice that, in this example, all the atoms have different types. If there are chemically equivalent atoms in your system, they should be of the same type. For very large systems, the user may want to use the utility *needle,* which builds the *atom2type.txt* file from *geometries.txt*:

```
needle -p 13 -o geometries.txt
```

The option “-p 13” indicates that the first molecule has 13 atoms, and the option “-o” tells *needle* to create the files *atom2type.txt* and *charges.txt*. The user should check the former file, and change it if necessary. By default, all the atomic charges written in the latter file are 0.0. If, for the fittings, you want to use pairwise functions represented by Lennard-Jones potentials plus point-charge electrostatic terms, then you have to modify the file *charges.txt* appropriately (charges must be specified in a.u.). This can be done by editing the file or, alternatively, by using *bedit*. This utility is an interactive editor for files *atom2type.txt*, *charges.txt* and *bounds.txt* (this file is described next).

(4) A file showing the lower and upper limits for the parameters that will be fitted (default filename: *bounds.txt*). The total number of parameters will depend on the form of the intermolecular potentials used in the fittings. At present, GAFit includes the potentials shown in Table 1. If you want to use a different pairwise potential, or include many-body terms, you should modify the file *userpotential.f*, written in Fortran language. Details are given in the Appendix. In our example, we have used two-body potentials of type 2, which comprise 6 parameters for each kind of interaction (78 different parameters in all). The format for *bounds.txt* is illustrated as follows:

```
Pair potential 1: C(1)-Xe(14)
    13000.0     20000.0    0
    2.3     2.8    0
    -850.0     -200.0    0
    3.8     4.6    0
    20.0     80.0    0
    10.0     16.0    0
Pair potential 2: N(2)-Xe(14)
    48000.0     75000.0    0
    2.6     3.4    0
    -1500.0     -800.0    0
    5.0     5.9    0
    60.0     150.0    0
    10.0     13.5    0

...

Pair potential 13: H(13)-Xe(14)
    40000.0     70000.0    0
    3.8     4.3    0
    -1400.0     -600.0    0
    6.0     7.0    0
    50.0     900.0    0
    11.5     18.0    0
```

Here we show the limits for three different pairwise potentials or interaction types (for simplicity, we do not include the remaining data). By default, the file must include the limits for all the parameters. The ordering in this file is very important because it is associated with the atom numbering. The first group of data refers to the pairwise interactions between the first atom type of molecule 1 and the first atom type of molecule 2. The second group of data corresponds to the pairwise interactions between the first atom type of molecule 1 and the second atom type of molecule 2, and so on. In our example, the second molecule is, in fact, an atom. Therefore, the first group corresponds to the pairwise interaction between atom types 1 and 14, the second group concerns the interaction between atom types 2 and 14, and so on. The first line of each group of data is a title. The second line includes the lower and upper limits for the first parameter (*A*), the third line specifies the limits for the second parameter (*B*), and so on. The numbers in the last column indicate whether the optimized parameters should be floating point (0) or integer (1) numbers.

Table 1: Pairwise potentials internally defined in GAFit.

Type	Coefficients	Pairwise potential[a]
–1	any	User function to be coded in “userpotential.f”
1	4	$V = Ae^{-Br} + \frac{C}{r^D}$
2	6	$V = Ae^{-Br} + \frac{C}{r^D} + \frac{E}{r^F}$
3	8	$V = Ae^{-Br} + \frac{C}{r^D} + \frac{E}{r^F} + \frac{G}{r^H}$
4	2	$V = A\left[\left(\frac{B}{r}\right)^{12} - \left(\frac{B}{r}\right)^{6}\right] + 332.0637813\frac{q_i q_j}{r}$

[a] Coefficients *A* to *H* are the pairwise parameters. In function 4, q_i and q_j are partial charges (in a.u.) for interacting atoms, and the real number is a constant that gives the electrostatic term in kcal/mol when the distance *r* between atoms *i* and *j* is specified in Å.

If you want to use the same limits for all the parameters of the same kind (i.e., *A*, *B*, ...), you can simplify the file *bounds.txt*. An example may be:

```
Lower and upper limits for A-F parameters
    0.0  80000.0    0
    2.0      6.0    0
-1000.0      0.0    0
    3        8      1
    0.0   1000.0    0
   10       16      1
```

In this case, and because this is not the default format, the user must specify this alternative in a file called *job.txt*, which is described below.

(5) If you select the pairwise potential of type 4, which is the widely employed potential in most force fields, GAFit will need the abovementioned file *charges.txt*.

(6) Finally, the *job.txt* file, which gives GAFit the relevant information for the fittings. For our example, using the default filenames already mentioned, the data included in this file may be as simple as:

```
[Job]
Evaluations: 1000000
Application: intermolecular
Potential: 2
```

In general, the data specified in the file *job.txt* can be divided into different sections, and each of them is headed by a keyword in brackets (case insensitive). Only one section, with the keyword "Job", is mandatory. In this example, GAFit performs a single run comprising one million evaluations, using eq 2 for the objective function (default for this type of application). If you want to use eq 1, you must specify in the [Job] section: "Fitting: absolute". Also, if you want to use a common set of lower and upper limits, as noticed above, you must include in this section an additional line with: "All coefficients: no".

Besides the [Job] section, the user may include in the *job.txt* file the sections [Parameters] and [Print]. The last section controls the different alternatives for printing the output data, as detailed in Table 2. The section [Parameters] refers to the parameters associated with the genetic algorithm. In general, we recommend using the default values, which are indicated in Table 2.

Table 2. Default options for the [Parameters] and [Print] sections in the *job.txt* file.[a]

Parameter	Type	Valid set	Default
[Parameters]			
population	integer		100
crossover rate	real		0.75
crossover	string	{spc, dpc, blax, sbx}	sbx
blx_alpha	real		0.5
eta_sbx	real	non negative	2.0
mutation rate	real		0.1
mutation	string	{random, sigma}	sigma
sigma	real		0.1
integer mutation	string	{random, adjacent}	random
elitism	bool	{yes, no}	yes
tournament size	integer		5
direction	string	{min, max}	min
[Print]			
geometries	bool	{yes, no}	yes
runs	bool	{yes, no}	yes
GA settings	bool	{yes, no}	no

[a] For details about the GA parameters, see reference 2 of section “GAFit citation”.

Once we have all the above data prepared, we can run GAFit:

```
gafit > output.txt &
```

where *output.txt* is the name of the main output file generated by GAFit. Some information written in this file is given as follows:

```
*************
  gafit 1.1
  Build: 12
*************
application settings: intermolecular

Job type: internal

Settings for job
-----------------------------
Coordinates:[geometries.txt]
Energies:[energies.txt]
Atom2type:[atom2type.txt]
Bounds:[bounds.txt]
Charges:[charges.txt]
Potential read: 2
```

```
All coefficients: yes, Read whole set
Fitting: relative
Auto weights: no

Print options:
        geometries yes
        runs yes
        ga settings no
        analytical no
        auto weights no
...now reading data
          14
 C    0.00000000       0.00000000       0.00000000
 N    0.00000000       0.00000000       1.36238396
 C    1.15504897       0.00000000       2.13998008
 N    2.32077909       0.00000000       1.35522401
 C    2.40223789       0.00000000      -1.10579999E-02
 C    1.15989399       0.00000000     -0.718415022
 O    1.17064297       0.00000000       3.35711002
 O    3.54523611       0.00000000     -0.554966986
 Li   5.00872993       0.00000000      -1.52392697
 H    3.19143391       0.00000000       1.88842797
 H    1.15401900       0.00000000      -1.80072796
 H  -0.977216005       0.00000000     -0.473394006
 H  -0.873789012       0.00000000       1.88616502
 Xe   17.5158615       0.00000000      -9.80472755
          14
...
Different interaction types: 13,
        with 6 coefficients each,
        so, we need a 78 elements vector.
        Choosen potential=2
Fragment A atoms: 13, Fragment B atoms: 1
Fragment A types: 13, Fragment B types: 1
A and B common types: 0
Different interactions: 13

Reading bounds for 6 coefficients

            1          A    +13000.00000 -    +20000.00000 (real)
            2          B        +2.30000 -        +2.80000 (real)
            3          C      -850.00000 -      -200.00000 (real)
            4          D        +3.80000 -        +4.60000 (real)
            5          E       +20.00000 -       +80.00000 (real)
            6          F       +10.00000 -       +16.00000 (real)

...

78 BOUNDS VECTOR
================================================================

INTERACTION TYPE 1
----------------------
C(1)-Xe(14)
        Coefficients:
            1          A    +13000.00000 -    +20000.00000 (real)
            2          B        +2.30000 -        +2.80000 (real)
            3          C      -850.00000 -      -200.00000 (real)
            4          D        +3.80000 -        +4.60000 (real)
            5          E       +20.00000 -       +80.00000 (real)
            6          F       +10.00000 -       +16.00000 (real)

...

#seed#1468426708#seed#

run 1
Eval.           Best fit.
------------------------------------
100             1.236266675242e+07
10200           1.487424926316e+05
20300           1.335889505418e+05
30400           1.244792007801e+05
40500           1.180134679772e+05
50600           1.160364163593e+05
60700           1.157183966786e+05
70800           1.146496840250e+05
...

#
#Results
#

INTERACTION TYPE 1
----------------------
C(1)-Xe(14)
        Coefficients:
            1 A    +17995.7364524386
            2 B        +2.6249900376
            3 C      -281.7139874216
            4 D        +4.3197620184
            5 E       +32.8522517088
            6 F       +15.7925252683

...

#
#Evaluation
#
#Geometry     Energy        Calculated      Difference     Weight
#========     ======        ==========      ==========     ======
     1   -0.008850000000  -0.006086307564  -31.23 %       +1.00
     2   -0.015440000000  -0.011384983190  -26.26 %       +1.00
     3   -0.029340000000  -0.023678617472  -19.30 %       +1.00
     4   -0.064650000000  -0.057208979939  -11.51 %       +1.00
     5   -0.179060000000  -0.173571660358   -3.07 %       +1.00
     6   -0.243400000000  -0.241066465936   -0.96 %       +1.00
     7   -0.343490000000  -0.343906926998   +0.12 %       +1.00
     8   -0.506510000000  -0.506142762731   -0.07 %       +1.00
     9   -0.658620000000  -0.649107650390   -1.44 %       +1.00
    10   -0.869250000000  -0.844044075207   -2.90 %       +1.00
    11   -1.160440000000  -1.113387404431   -4.05 %       +1.00
    12   -1.585350000000  -1.489331481910   -6.06 %      +1000.00
    13   -2.192030000000  -2.015574606192   -8.05 %      +1000.00
    14   -2.724240000000  -2.475184206607   -9.14 %      +5000.00
    15   -3.379990000000  -3.036920269255  -10.15 %      +5000.00
    16   -4.154320000000  -3.700910521647  -10.91 %      +5000.00
    17   -5.003280000000  -4.433813015452  -11.38 %      +12000.00
    18   -5.772580000000  -5.123951695418  -11.24 %      +22000.00
    19   -6.038150000000  -5.375506696459  -10.97 %      +35000.00
    20   -6.136570000000  -5.488768545945  -10.56 %      +35000.00
    21   -5.969450000000  -5.371468392937  -10.02 %      +12000.00
    22   -5.395920000000  -4.888580485662   -9.40 %      +9200.00
    23   -4.216980000000  -3.844642589493   -8.83 %      +9000.00
    24   -2.156870000000  -1.959273337804   -9.16 %      +5000.00
    25   +1.168220000000  +1.166525389514   -0.15 %      +200000.00
    26   +6.288150000000  +6.096264071038   -3.05 %      +1000.00
    27   +13.943900000000  +13.619902158748   -2.32 %      +1000.00
    28   +25.151730000000  +24.835973205299   -1.26 %       +1.00
    29   +41.306320000000  +41.258659760509   -0.12 %       +1.00
    30   +64.316070000000  +64.951212352573   +0.99 %       +1.00
    31   +96.772630000000  +98.683947050039   +1.98 %       +1.00
    32   +142.121250000000  +146.098586650258   +2.80 %       +60.00
    33   +204.887760000000  +211.830627817500   +3.39 %       +800.00
...
```

The last section of this file prints the set of fitted parameters and a comparison between the target values (column under the header “energy”) and those calculated with the fitted potentials. As pointed out previously, the potential energies are given in kcal/mol. We notice that the GAFit execution generates a series of files with additional information. Among them, the most relevant is *best.txt*, which contains the best set of fitted parameters as well as the corresponding value of the objective function.

The utility *fitview* (used without arguments) may be used to generate data files (with extensions “.dat” and “.plt”) for plotting the fitted pairwise potentials with GNUPLOT. In addition, this utility generates the files *general_evaluation.dat* and *general_evaluation.plt* for plotting a comparison between the target values and those evaluated with the fitted potentials.

Finally, it is worth mentioning that there is a Fortran subroutine in which the user may implement new potentials not included in Table 1. This subroutine, called *userpotential.f*, is described in the Appendix.

5. Interface with the CHARMM program

As described in the first reference given in the Citation Section, GAFit was interfaced with CHARMM to facilitate direct parameterizations of force fields. Here, and following this paper, we show an example that illustrates this type of application. In particular, we will consider the zwitterionic form of glycine: $NH_3^+CH_2COO^-$. Suppose we want to reparameterize the torsional terms associated with the central C-C bond, that is, the NCCO and the HCCO torsions. We will need the following data.

(1) A collection of geometries, with the corresponding target energies and weights, appropriately selected for the parameterization of the desired force field terms. Here, for illustration purposes, we used geometries and energies obtained with the standard CHARMM force field, by a rigid scan performed with a step size of 5° and starting from the optimized, global minimum structure. The geometries must be written in Cartesian coordinates (in Å), in separate files, with the standard CRD format of CHARMM. All these files must be gathered in a directory called *geometries*. No additional files should be included in this directory. For these geometry files, it is important to use filenames that facilitate file operations within a CHARMM input file. Specifically, for this example we employed 37 different geometries named as "geo-*n*.crd", where *n* is an integer starting from 1 (i.e., geo-1.crd, geo-2.crd and so on). For example, the geo-1.crd has the form:

```
* 1 : -42.6026 : 1.0
   10
    1    1 GLY  N      0.00326  -0.04316  -0.06921 GLY  1      0.00000
    2    1 GLY  HT1   -0.35888  -0.63209   0.69825 GLY  1      0.00000
    3    1 GLY  HT2   -0.43484  -0.22526  -0.98675 GLY  1      0.00000
    4    1 GLY  HT3   -0.13323   0.98292   0.18470 GLY  1      0.00000
    5    1 GLY  CA     1.49972  -0.03258  -0.13410 GLY  1      0.00000
    6    1 GLY  HA1    1.82427  -0.25275  -1.14048 GLY  1      0.00000
    7    1 GLY  HA2    1.90407  -0.68024   0.62987 GLY  1      0.00000
    8    1 GLY  C      1.94104   1.39101   0.18975 GLY  1      0.00000
    9    1 GLY  OT1    1.04484   2.24755   0.43692 GLY  1      0.00000
   10    1 GLY  OT2    3.16784   1.61874   0.18952 GLY  1      0.00000
```

The first line starts with an asterisk (*), which tells CHARMM that this line is a title. However, it contains important data for GAFit. The second field of this line contains an integer, which corresponds to the geometry number. The next number in this line, written as a floating point, is the target potential energy (in kcal/mol) for this geometry, and the last number is the weight. Notice that the numbers must be separated by colons. The remaining lines define the geometry according to the CRD format.

(2) A CHARMM topology file, which may be one of the topology files distributed within the CHARMM package, for example, *top_all36_prot.rtf*.

(3) A CHARMM parameter file. The default filename for GAFit is *template.prm*. This file may be one of the parameter files distributed with CHARMM, but you must do some changes or include new lines if you want to parameterize new parameters. In our example, and only for illustration purposes, we want to reparametrize the NCCO and the HCCO torsions. The lines for these torsions in the original parameter file (CHARMM22 force field) read:

```
...
OC   CC   CT2  NH3      3.2000  2   180.00 ! ALLOW PEP PRO
...
X    CT2  CC   X        0.0500  6   180.00 ! ALLOW   POL PEP
...
```

In each of these lines, the first numbers correspond to the force constants (in kcal/mol), the integer numbers are the multiplicities, and the third numbers are the phases. You should change these lines (in *template.prm*) to something like:

```
...
OC   CC   CT2  NH3      @fc1(0.,5.)  @mult1(1,6) @phase1 (0;180)
...
X    CT2  CC   X        @fc2(0.,5.)  @mult2(1,6) @phase2 (0;180)
...
```

The symbol “@” precedes the data for a given parameter. Here, in all, we have 6 independent parameters that will be fitted. In particular, “fc1” is the name we use for the force constant for torsions involving atoms whose CHARMM types are OC, CC, CT2 and NH3 (in this order). The values in parenthesis (0.,5.) indicate that this parameter is represented by a floating point number, ranging from 0.0 to 5.0 kcal/mol. The multiplicity for OC-CC-CT2-NH3 torsions (mult1) is represented by an integer, which may vary from 1 to 6. Finally, the phase for this type of torsions (phase1) may take only two values: 0 or 180. Notice that, in this case, the values within the parenthesis are separated by a semicolon, rather than by a comma.

(4) A CHARMM input file. Specifically, for this example we employed the following file (default filename: *fitting.dat*):

```
* Glycine in zwitterionic form
*

open unit 1 card read name top_all36_prot.rtf
read rtf card unit 1
```

```
open unit 2 card read name parameters.prm
read para card unit 2

read sequence cards
* gly
*
1
gly

generate GLY setup first GLYP last CTER

set point 1
set loopsize 37

! Atoms defining N-C-C-O dihedral
set a 1
set b 5
set c 8
set d 9

! Set up a file to keep track of energies and dihedral angles
open write card unit 21 name calculated.energies

! Loop for calculating the energies of all the geometries
label loop

! Read coordinates
open unit 29 card read name geometries/geo-@point.crd
read coor card unit 29
close unit 29

! Compute the energy for selected geometry
energy

! Determine dihedral angle
quick @a @b @c @d
set angle ?phi

! Write out the current dihedral angle and energy
set name geo-@point.crd

write title unit 21
* @name @angle ?ener
*

incr point
if @point lt @loopsize.5  goto loop

close unit 21

stop
```

Using this input file, CHARMM reads the topology and parameter files (i.e., "top_all36_prot.rtf" and "parameters.prm"), generates the glycine segment, and then calculates the energy for each geometry. Also, CHARMM writes, in a file called "calculated.energies" (default filename), the CRD filename, the value of the monitored dihedral angle (i.e., N1-C5-C8-O9), in degrees, and the calculated energy (in kcal/mol) for each geometry. An example showing a few lines of this file is:

```
 GEO-1.CRD -2.063507319E-03 -42.3130251
 GEO-2.CRD -5.00226022 -42.2326496
 GEO-3.CRD -10.0022356 -41.9329715
 GEO-4.CRD -15.0021843 -41.1610537
 GEO-5.CRD -20.0020657 -40.2801482
 GEO-6.CRD -25.0022808 -39.1410691
...
```

(5) The *job.txt* file, which in our example is:

```
[Job]
Evaluations: 50000
Application: CHARMM
Exec: /usr/programs/charmm/exec/gnu/charmm
Refgeom: geo-1.crd
Calculated energies: 1    3
```

As shown in this *job* file, the user must indicate the complete path of the CHARMM executable. Also, it is necessary to specify the geometry that will be used to scale all the potential energies. For this geometry, both the target energy and that calculated with CHARMM will be set to zero. Finally, after the keyword "Calculated energies", the user will have to specify the numbers of the columns, in the file "calculated.energies", that contain the CRD filenames and the energies calculated by CHARMM (1 and 3 in our example).

Once all the data were prepared, you can run GAFit in the same way as for fittings of intermolecular potentials:

```
gafit > output.txt &
```

The form of the *output.txt* file is:

```

*************
  gafit 1.1
  Build: 12
*************
application settings: charmm
autoconfiguring...
Energy reference:geo-1.crd
```

```
Job type: external bulk
Command : ./chmm.sh

Settings for job
-----------------------------
Command:[./chmm.sh]
Bounds:[bounds.txt]
External input:[charmm.input]
External fit:[charmm.fit]
Coefficients: 6

Print options:
        runs yes
        ga settings no
...now reading data

Reading bounds for 6 coefficients

            1        fc1        +0.00000 -        +5.00000 (real)
            2      mult1        +1.00000 -        +6.00000 (integer)
            3     phase1        +1.00000 -        +2.00000 (integer)
            4        fc2        +0.00000 -        +5.00000 (real)
            5      mult2        +1.00000 -        +6.00000 (integer)
            6     phase2        +1.00000 -        +2.00000 (integer)


 6 BOUNDS VECTOR
            1        fc1        +0.00000 -        +5.00000 (real)
            2      mult1        +1.00000 -        +6.00000 (integer)
            3     phase1        +1.00000 -        +2.00000 (integer)
            4        fc2        +0.00000 -        +5.00000 (real)
            5      mult2        +1.00000 -        +6.00000 (integer)
            6     phase2        +1.00000 -        +2.00000 (integer)

#seed#1468401578#seed#

run 1
Eval.           Best fit.
------------------------------------
100             7.814998210000e+02
200             2.972104790000e+02
300             1.976430000000e+00
400             1.746275000000e+00
500             1.746275000000e+00
600             1.052513000000e+00
700             8.883390000000e-01
800             7.168480000000e-01
900             1.747860000000e-01
1000            3.458300000000e-02
1100            3.458300000000e-02
1200            9.518000000000e-03
1300            8.877000000000e-03
1400            8.877000000000e-03
1500            7.503000000000e-03
1600            2.114000000000e-03
1700            3.790000000000e-04
1800            2.190000000000e-04
1900            1.460000000000e-04
```

```
2000            6.600000000000e-05
...

#
#Results
#
     1        fc1   +3.200062247511
     2      mult1   +2.000000000000
     3     phase1   +2.000000000000
     4        fc2   +0.050002780746
     5      mult2   +6.000000000000
     6     phase2   +2.000000000000
```

The section with the results at the end of the file, showing the optimized parameters, needs some clarification. Although multiplicities and phases are treated as integers, they are presented as floating points in this section. In addition, only two different phases were considered in the fittings: 0 or 180 degrees. Internally, the program assigns 0 to 1 and 180 to 2. Therefore, in this example, the optimized phases correspond to 180 degrees.

Apart from the main output file, GAFit generates other files that contain information that might be useful for the user. One of them is *best.txt*, which shows the optimized parameters as well as the corresponding value of the objective function. This file is used by the utility *chmfinal*, which performs a comparison between the target values and those calculated by CHARMM with the parameters specified in *best.txt*. The user may run this utility in the following way:

```
chmfinal > comparison.txt
```

An example of a *comparison.txt* file is displayed below. The value of the objective function is given as “Fitness”.

```


#
#FINAL EVALUATION
#
                           COEFFICIENTS

        0                   fc1:        3.200554499075
        1                  per1:        2.000000000000
        2                phase1:        180
        3                   fc2:        0.050011361767
        4                  per2:        6.000000000000
        5                phase2:        180
```

```
                           Fitness: 6.20415e-05
#
# (relative values to none)
#
#   Geometry       Reference   Calculated   Difference
#==============   ==========  ===========  ===========
  GEO-1.CRD       -42.602600   -42.602645     -0.0001%
  GEO-10.CRD      -34.253600   -34.252492      0.0032%
  GEO-11.CRD      -32.867200   -32.865990      0.0037%
  GEO-12.CRD      -31.491200   -31.489795      0.0045%
  GEO-13.CRD      -30.154100   -30.152552      0.0051%
  GEO-14.CRD      -28.893700   -28.891982      0.0059%
  GEO-15.CRD      -27.756800   -27.755000      0.0065%
  GEO-16.CRD      -26.796200   -26.794178      0.0075%
  GEO-17.CRD      -26.064900   -26.062795      0.0081%
  GEO-18.CRD      -25.608100   -25.605983      0.0083%
  GEO-19.CRD      -25.456300   -25.454086      0.0087%
  GEO-2.CRD       -42.460900   -42.460905     -0.0000%
  GEO-20.CRD      -25.619800   -25.617529      0.0089%
  GEO-21.CRD      -26.086800   -26.084660      0.0082%
  GEO-22.CRD      -26.825600   -26.823583      0.0075%
  GEO-23.CRD      -27.789500   -27.787633      0.0067%
  GEO-24.CRD      -28.924500   -28.922808      0.0058%
  GEO-25.CRD      -30.176600   -30.175070      0.0051%
  GEO-26.CRD      -31.498100   -31.496733      0.0043%
  GEO-27.CRD      -32.850700   -32.849448      0.0038%
  GEO-28.CRD      -34.205500   -34.204332      0.0034%
  GEO-29.CRD      -35.540000   -35.539156      0.0024%
  GEO-3.CRD       -42.044300   -42.044256      0.0001%
  GEO-30.CRD      -36.833400   -36.832870      0.0014%
  GEO-31.CRD      -38.062100   -38.061541      0.0015%
  GEO-32.CRD      -39.196600   -39.196159      0.0011%
  GEO-33.CRD      -40.201400   -40.201128      0.0007%
  GEO-34.CRD      -41.037500   -41.037367      0.0003%
  GEO-35.CRD      -41.667100   -41.667199     -0.0002%
  GEO-36.CRD      -42.058700   -42.058894     -0.0005%
  GEO-37.CRD      -42.191600   -42.191763     -0.0004%
  GEO-4.CRD       -41.376400   -41.376225      0.0004%
  GEO-5.CRD       -40.492600   -40.492286      0.0008%
  GEO-6.CRD       -39.435000   -39.434586      0.0010%
  GEO-7.CRD       -38.246800   -38.246263      0.0014%
  GEO-8.CRD       -36.966900   -36.966187      0.0019%
  GEO-9.CRD       -35.627400   -35.626489      0.0026%
```

6. Interface with the MOPAC program

This interface was designed for reparameterizations of semiempirical Hamiltonians, which may be useful for direct dynamics simulations of chemical reactions. As in Ref. 1, for illustration purposes we considered the reaction depicted in Figure 2, which is one of the elementary steps involved in the unimolecular decomposition of methyl cyanoformate (MCF). Our aim is to refine the original PM3 parameters in order to describe more accurately the exit channel, that is, the region of the potential energy surface from the transition state to products.

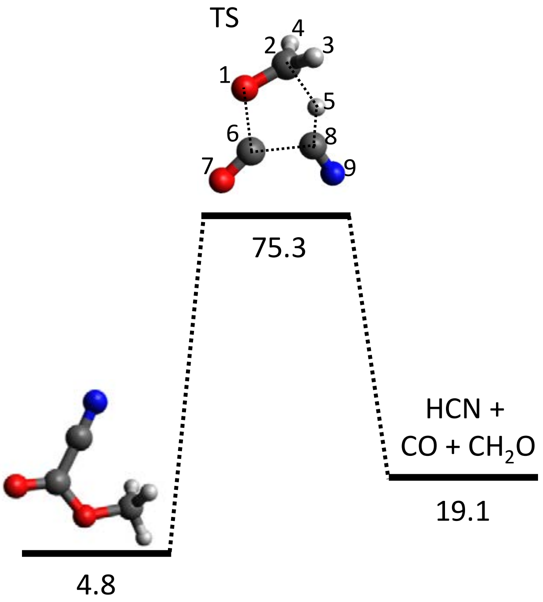


Figure 2. Schematic of the major HCN formation channel from MCF in the ground electronic state. Relative energies (in kcal/mol) include zero-point energy contributions.

For this type of fittings we need the following data files:

(1) The *job.txt* file. An example is

```
[Job]
Evaluations: 100000
Application: MOPAC
Exec: /usr/programs/mopac/MOPAC2016.exe
```

In this case, GAFit will perform one hundred thousand evaluations. Resembling the interface with CHARMM, you must specify the complete path of the MOPAC2016 executable.

(2) A MOPAC input file, called *template.mop*, containing the data for a series of MOPAC calculations. In our example, this file includes the following data:

```
pm3 ts precise external=@
ts opt Calc number  1

 O     0.602377  1   1.747672  1  1.468819 1
 C     0.720763  1   1.673557  1  0.196945 1
 H     1.282434  1   0.845622  1 -0.234559 1
 H     0.548946  1   2.566613  1 -0.403118 1
 H    -0.677585  1   1.045195  1 -0.133712 1
 C    -1.028709  1   1.109413  1  2.049872 1
 O    -1.431149  1   1.042754  1  3.120518 1
 C    -1.665193  1   0.674903  1  0.383195 1
 N    -2.725474  1   0.201184  1  0.160337 1

oldgeo pm3 force external=@
ts freq Calc number  2

pm3 precise external=@
prods Calc number  3

 C     0.720763  1  1.673557  1  0.196945 1
 O     0.602377  0  1.747672  0  1.468819 0
 C   -12.655732  0 -3.440348  0  6.191846 0
 O   -13.058172  1 -3.507007  1  7.262492 1
 C   -17.855127  0 -6.989832  0 -7.423126 0
 N   -18.915409  1 -7.463551  1 -7.645984 1
 H     1.282434  1  0.845622  1 -0.234559 1
 H     0.548946  1  2.566613  1 -0.403118 1
 H   -16.867519  1 -6.619539  1 -7.940033 1

pm3 precise external=@
hcn opt calc number 4

C      -17.855127000 1   -6.989832000 1   -7.423126000 1
N      -18.010807521 1   -7.155299755 1   -8.568415518 1
H      -17.713969613 1   -6.839703368 1   -6.383986092 1

oldgeo pm3 force external=@
hcn freqs calc number 5
```

With this input file, MOPAC will use the PM3 Hamiltonian with parameters read from an external file. It is mandatory to include “external=@” in all the command lines. Notice that this input file tells MOPAC to run five calculations. The first one is an optimization of the transition state, which is followed by the calculation of the vibrational frequencies on the optimized structure. The third calculation optimizes the structures of the product species and gives the total heat of formation for products.

Finally, the last two calculations are performed to evaluate the vibrational frequencies of the HCN product, which will be used for the parameterization.

(3) A file containing the target data used for the reparameterization (default filename: *conditions.txt*). In our example, we used the following data:

```
dist 1 1 2 1.280 19.2367
dist 1 1 3 2.044 10.7922
dist 1 1 4 2.044 10.7922
dist 1 1 5 2.168 9.96391
dist 1 1 6 1.845 12.3548
dist 1 1 7 2.713 7.25354
dist 1 1 8 2.733 7.17603
dist 1 1 9 3.896 4.17174
dist 1 2 3 1.090 22.8932
dist 1 2 4 1.090 22.8932
dist 1 2 5 1.569 15.1521
dist 1 2 6 2.610 7.67336
dist 1 2 7 3.685 4.55597
dist 1 2 8 2.593 7.74614
dist 1 2 9 3.748 4.43587
dist 1 3 4 1.878 12.0731
dist 1 3 5 1.973 11.3138
dist 1 3 6 3.260 5.51037
dist 1 3 7 4.320 3.53327
dist 1 3 8 3.016 6.2003
dist 1 3 9 4.079 3.87653
dist 1 4 5 1.973 11.3138
dist 1 4 6 3.260 5.51037
dist 1 4 7 4.320 3.53327
dist 1 4 8 3.016 6.2003
dist 1 4 9 4.079 3.87653
dist 1 5 6 2.213 9.68676
dist 1 5 7 3.340 5.3091
dist 1 5 8 1.175 21.1389
dist 1 5 9 2.234 9.56137
dist 1 6 7 1.146 21.714
dist 1 6 8 1.836 12.4333
dist 1 6 9 2.697 7.31646
dist 1 7 8 2.772 7.0284
dist 1 7 9 3.339 5.31154
dist 1 8 9 1.182 21.0034
dist 3 1 2 1.213 20.4191
dist 3 2 7 2.019 10.9717
dist 3 2 8 2.019 10.9717
dist 3 1 7 1.098 22.719
dist 3 1 8 1.098 22.719
dist 3 7 8 1.870 12.1405
dist 3 3 4 1.138 21.8768
dist 3 5 9 1.063 23.4964
dist 3 6 9 2.230 9.58506
dist 3 5 6 1.167 21.2952
freq 2 1 -786 0.000161455
freq 2 2 130 0.00582717
freq 2 3 167 0.00354308
freq 2 4 168 0.00350128
freq 2 5 271 0.00135164
freq 2 6 368 0.000734425
freq 2 7 481 0.000430433
freq 2 8 539 0.000342936
freq 2 9 570 0.00030671
freq 2 10 648 0.000237416
freq 2 11 832 0.000144115
freq 2 12 1010 9.78358e-05
freq 2 13 1210 6.81886e-05
freq 2 14 1234 6.55641e-05
freq 2 15 1376 5.2739e-05
freq 2 16 1553 4.14093e-05
freq 2 17 1828 2.98932e-05
freq 2 18 2080 2.30917e-05
freq 2 19 2267 1.94408e-05
freq 2 20 3052 1.07287e-05
freq 2 21 3164 9.9828e-06
freq 5 1 715 0.000161455
freq 5 2 2014.99 2.30917e-05
freq 5 3 3464 9.9828e-06
delt 1 3 53.1 0.0341669
```

In each line, the first field defines the type of data. Specifically, “dist” is used for a distance between two atoms, “freq” for vibrational frequencies and “delt” for energy differences. Regarding geometrical data, you can also use “angle” and “dihedral”. For geometrical data and frequencies, the second field in the row is an integer corresponding to the calculation number. Notice that in this case every MOPAC run performs five different calculations, as detailed above. The next two (for “dist”), three (for “angle”) or four (for “dihedral”) numbers in each row indicate the indices of the atoms involved in the calculation of that particular distance, angle or dihedral (the atom numbering is specified in Figure 2). The next number is the target value and the last number is the weight. For example, the data “`dist 1 1 6 1.845 12.3548`” indicates that the target distance between atoms 1 and 6 in the transition state (calculation number 1) is 1.845 Å, and that the weight for this data point is 12.3548.

For GAFit-MOPAC fittings we use, as default, the following objective function

$$\chi^2 = \sum_{i=1}^{N} \sum_{j=1}^{M_i} \left({y_{j,i}}^{ab\ initio} - {y_{j,i}}^{PM3-SRP} \right)^2 w_{y_{j,i}} \qquad (3)$$

where M_i is the number of data points for calculation number i and $y_{j,i}$ refers to each of the data points employed in the fitting. Also, as default, we used weights calculated with the following formula

$$w_{y_{j,i}} = \left(\frac{10}{1+|y_{j,i}^{ab\ initio}|}\right)^2 \qquad (4)$$

For frequencies, the integer number shown in the third field specifies the normal mode, and the number (integer or real) in the fourth field gives the corresponding frequency (in cm^{-1}). Notice that the normal modes are ordered by increasing frequencies. If the frequency calculation involves a transition state (2 in our example), the first normal mode corresponds to the reaction coordinate, having an imaginary (or negative) frequency (–786 cm^{-1} in our example). When the first column is “delt”, the next two numbers refer to the calculations from which the energy difference will be computed (more exactly, the difference between the corresponding heats of formation, in kcal/mol). For example, “`delt 1 3 53.1 0.0341669`” indicates that the difference between the heats of formation obtained from calculation 1 (transition state) and calculation 3 (products) will be compared with the target value of 53.1 kcal/mol. This value is the reverse barrier height, without zero-point energy corrections, obtained by ab initio calculations for the reaction depicted in Figure 2.

(4) A file specifying the semiempirical parameters that will be refitted (default filename: *template.coefs*). In our example, this file reads

```
BETAS  H        -6.173787
ZS     H         1.188078
ALP    H         2.882324
GSS    H        12.848
USS    C       -52.028658
UPP    C       -39.614239
BETAS  C       -15.715783
BETAP  C        -7.719283
ZS     C         1.808665
ZP     C         1.685116
ALP    C         2.648274
GSS    C        12.23
GSP    C        11.47
GPP    C        11.08
GP2    C         9.84
HSP    C         2.43
```

As can be seen, in each line, the first field specifies the nature of the parameter, the second field the atomic symbol, and the third field gives the standard value in MOPAC.

(5) Finally, the user must include the *bounds.txt* file. For this type of fittings, we usually set the parameter limits to ±10% of the original values. Accordingly, in our example we used the following file:

```
    ±10% of original values
    -6.7911657   -5.5564083    0
    1.0692702   1.3068858    0
    2.5940916   3.1705564    0
    11.5632   14.1328    0
    -57.2315238   -46.8257922    0
    -43.5756629   -35.6528151    0
    -17.2873613   -14.1442047    0
    -8.4912113   -6.9473547    0
    1.6277985   1.9895315    0
    1.5166044   1.8536276    0
    2.3834466   2.9131014    0
    11.007   13.453    0
    10.323   12.617    0
    9.972   12.188    0
    8.856   10.824    0
    2.187   2.673    0
```

Once you have prepared all the data files, you can run GAFit as usual. After execution, the main output file will have the form:

```
*************
  gafit 1.1
  Build: 12
*************
application settings: mopac
autoconfiguring...

Job type: external bulk
Command : ./external-mopac.sh

Settings for job
-----------------------------
Command:[./external-mopac.sh]
Bounds:[bounds.txt]
External input:[mopac.input]
External fit:[mopac.fit]
Coefficients: 16

Print options:
        runs yes
        ga settings no
...now reading data

Reading bounds for 16 coefficients
```

```
            1    BETAS H        -5.55641 -        -6.79117 (real)
            2       ZS H        +1.06927 -        +1.30689 (real)
            3      ALP H        +2.59409 -        +3.17056 (real)
            4      GSS H       +11.56320 -       +14.13280 (real)
            5      USS C       -46.82579 -       -57.23152 (real)
            6      UPP C       -35.65282 -       -43.57566 (real)
            7    BETAS C       -14.14420 -       -17.28736 (real)
            8    BETAP C        -6.94735 -        -8.49121 (real)
            9       ZS C        +1.62780 -        +1.98953 (real)
           10       ZP C        +1.51660 -        +1.85363 (real)
           11      ALP C        +2.38345 -        +2.91310 (real)
           12      GSS C       +11.00700 -       +13.45300 (real)
           13      GSP C       +10.32300 -       +12.61700 (real)
           14      GPP C        +9.97200 -       +12.18800 (real)
           15      GP2 C        +8.85600 -       +10.82400 (real)
           16      HSP C        +2.18700 -        +2.67300 (real)


 16 BOUNDS VECTOR
================================================================
            1    BETAS H        -5.55641 -        -6.79117 (real)
            2       ZS H        +1.06927 -        +1.30689 (real)
            3      ALP H        +2.59409 -        +3.17056 (real)
            4      GSS H       +11.56320 -       +14.13280 (real)
            5      USS C       -46.82579 -       -57.23152 (real)
            6      UPP C       -35.65282 -       -43.57566 (real)
            7    BETAS C       -14.14420 -       -17.28736 (real)
            8    BETAP C        -6.94735 -        -8.49121 (real)
            9       ZS C        +1.62780 -        +1.98953 (real)
           10       ZP C        +1.51660 -        +1.85363 (real)
           11      ALP C        +2.38345 -        +2.91310 (real)
           12      GSS C       +11.00700 -       +13.45300 (real)
           13      GSP C       +10.32300 -       +12.61700 (real)
           14      GPP C        +9.97200 -       +12.18800 (real)
           15      GP2 C        +8.85600 -       +10.82400 (real)
           16      HSP C        +2.18700 -        +2.67300 (real)
run 1

#seed#1472638378#seed#

Eval.           Best fit.
------------------------------------
100             69.5771
300             69.5771
500             69.5771
700             69.5771
900             65.8821
1100            64.9055
1300            63.0892
...
99500           26.8279
99700           26.8279
99900           26.8279



#
#Results
#
     1    BETAS H   -6.657298928288
     2       ZS H   +1.108147455140
     3      ALP H   +3.093853871338
     4      GSS H   +12.796695984177
     5      USS C   -54.913255848784
     6      UPP C   -39.237708830507
     7    BETAS C   -17.287361300000
     8    BETAP C   -8.491211300000
     9       ZS C   +1.660454994750
    10       ZP C   +1.686683543508
    11      ALP C   +2.643096657816
    12      GSS C   +12.520292179122
    13      GSP C   +10.709243986667
    14      GPP C   +11.101172729281
    15      GP2 C   +9.756803735777
    16      HSP C   +2.310144875043
```

The optimized parameters are specified in the last section of the file.

7. Fitting a user-defined function to a set of data

GAFit can be used to fit a multivariable function to a set of data points. Here we show the easiest way to perform this type of fittings. For the present example, suppose we have a series of data points that can be fitted to a multiexponential function. Specifically, our function, *f*(*x*), will be a linear combination of four exponentials:

$$f(x) = \sum_{i=1}^{4} a_i \exp(-b_i x) \qquad (5)$$

In all, we have eight parameters to fit. To run GAFit, we will need:

(1) The data, collected in a single file. In this example, this file contains two columns of data:

```
#
#  x   y
590    0.46106
600    0.45758
610    0.45309
620    0.44883
630    0.44518
640    0.44103
650    0.43637
660    0.43205
...
```

(2) The *job.txt* file, which in this case has the following form:

```
[Job]
evaluations:   100000
application: mvariable

[Multi variable]
Coefficients:a1(0.0,4000.0),b1(0.0,0.05),a2(0.,0.7),b2(0.,0.004),a3(0.,0.7),b
3(0.,0.001),a4(0.,0.2),b4(0.,0.0003)

data file: data2fit.txt
data columns: x, y
data headers: 2

[Objective function]
func=a1*exp(-b1*x) + a2*exp(-b2*x) + a3*exp(-b3*x) + a4*exp(-b4*x) ;
fit=((y - func)/y)^2;
```

The [Job] section tells GAFit to perform a multivariable fitting (“Application: mvariable”), with one hundred thousand evaluations. The coefficients to be fitted, their nature (integer or real) and their lower and upper limits are specified in the section [Multi variable], and in a single line after the keyword “Coefficients”. Also in this section, the user has to specify the filename of the data file, the number of title lines in this file (“data headers”), and assign the column numbers to the variables. In this example, the values in the first column correspond to the *x* variable, and those in the second column to the *y* variable.

Finally, you need to include in the *job.txt* file a section specifying the form of function that will be fitted as well as the expression of the objective function. For the latter, we used the default name “fit”. If different, you must indicate it in section [Multi variable], using the keyword “Fit variable”.

It is important to mention that, for this type of applications, the function specified in the [Objective function] section of the *job.txt* file is interpreted by GAFit, rather than compiled to machine code. Therefore, the executions of these fittings are relatively slow.

The execution of GAFit generates a series of files and, as for the previous cases, the most relevant are the main *output.txt* file and *best.txt*. The former file has the form:

```
  *************
    gafit 1.1
    Build: 12
  *************
  application settings: mvariable
  autoconfiguring...
  Mvariable Analysis
  ============================
  external inp: external.input
  external fit: external.fit
  bounds file : bounds.txt
  coefficients:
a1(0.0,4000.0),b1(0.0,0.05),a2(0.,0.7),b2(0.,0.004),a3(0.,0.7),b3(0.,0.001),a
4(0.,0.2),b4(0.,0.0003)
  fit variable: fit
  data file   : data2fit.txt
  columns     : x, y
  headers     : 2
  expression  : objective function
  print code  : no

          func = a1 * exp( - b1 * x) + a2 * exp( - b2 * x) + a3 * exp( - b3 *
x) + a4 * exp( - b4 * x);

          fit = ((y - func) / y)^2;



  ============================

  Job type: external bulk
  Command : mvariable

  Settings for job
  -----------------------------
  Command:[mvariable]
  Bounds:[bounds.txt]
  External input:[external.input]
  External fit:[external.fit]
Coefficients: 8

  Print options:
          runs yes
          ga settings no
  ...now reading data

  Reading bounds for 8 coefficients

              1         a1        +0.00000 -     +4000.00000 (real)
              2         b1        +0.00000 -        +0.05000 (real)
              3         a2        +0.00000 -        +0.70000 (real)
              4         b2        +0.00000 -        +0.00400 (real)
              5         a3        +0.00000 -        +0.70000 (real)
              6         b3        +0.00000 -        +0.00100 (real)
              7         a4        +0.00000 -        +0.20000 (real)
              8         b4        +0.00000 -        +0.00030 (real)
```

```
   8 BOUNDS VECTOR
  ================================================================
              1        a1        +0.00000 -     +4000.00000 (real)
              2        b1        +0.00000 -        +0.05000 (real)
              3        a2        +0.00000 -        +0.70000 (real)
              4        b2        +0.00000 -        +0.00400 (real)
              5        a3        +0.00000 -        +0.70000 (real)
              6        b3        +0.00000 -        +0.00100 (real)
              7        a4        +0.00000 -        +0.20000 (real)
              8        b4        +0.00000 -        +0.00030 (real)
  run 1

  #seed#1470306630#seed#

  Eval.           Best fit.
  ------------------------------------
  100             196.518
  300             48.4782
  500             37.7606
  700             36.0062
  900             34.2722
 1100             28.0344
...
  99300           21.4176
  99500           21.4176
  99700           21.4176
  99900           21.4176


  #
  #Results
  #
       1         a1   +3998.713670701100
       2         b1   +0.024031913357
       3         a2   +0.344297314205
       4         b2   +0.002208887725
       5         a3   +0.383170951000
       6         b3   +0.000536519343
       7         a4   +0.092065737482
       8         b4   +0.000143392802
```

The difference between the target data (*y* values) and those calculated by the function ("func") can be obtained by using the utility *mvtest* (without arguments).

8. Recommendations

In general, we recommend the following strategy for fittings with GAFit. Start the process by running a few (e.g., five) independent fittings, in separate directories, using a wide spectrum for each parameter and the same weight for each data point. In

GAFit, the seed for the generation of random numbers is initialized using the computer’s real time clock, unless specified by the user in the [Job] section (e.g., “Test: 615427”). Therefore, independent fittings will lead, in general, to different solutions. GAFit can perform several independent fittings in only one execution. This can be done by indicating the number of fittings in the [Job] section of the file *job.txt*, using the keyword “Runs” (e.g., “runs: 10”). However, the different fittings will be executed sequentially, and therefore this way is recommended only if the total CPU time needed for the fittings is small.

From the analysis of the results, you may change weights and parameter ranges. When possible, try to confine the parameter ranges to realistic or reasonable values. For example, for reparameterizations of semiempirical Hamiltonians, we recommend to employ parameter ranges within ±10% of the original values. With the new weights and limits, run a new series of independent fittings. In general, this process should be repeated several times until you are satisfied with the results or until you reach the conclusion that the function or potential is not appropriate to model the target data or the system that is being studied.

Appendix

Below is the Fortran subroutine *userpotential.f*, which may be modified by the user to implement a pairwise potential not included in the GAFit library. This subroutine is located in the directory “src”. Here, as an example, we consider a Buckingham type potential, which has four different parameters: *A*, *B*, *C* and *D*. The number of parameters of the pair potential is specified by the variable *usetcoefs*. In most cases, the user will need to change only the value of this variable and the Fortran function *userv*(*r*,*i*,*j*,*x*,*m*). In this function, *i* and *j* are atom indexes for molecules 1 and 2, respectively, and *r* is the distance (in Å) between these atoms. The integer *m* is the total number of parameters (in this case 4 times the number of different pair potentials). Finally, *x* is an array containing all the parameters.

```
c USER POTENTIAL
c please change as needed


c USER DATA MODULE

      module userdata
      implicit none
      save
c v-------CHANGE-ME-------------------v
c define your variables here

c ^-------CHANGE-ME-------------------^
      end module userdata


c USERREAD SUBROUTINE

      subroutine userread()
      use userdata
c v-------CHANGE-ME-------------------v
c your code to read external files here


c ^-------CHANGE-ME-------------------^
      end


C USETCOEFS FUNCTION

      integer function usetcoefs()
c here specify the number of coefficients
c v-------CHANGE-ME-------------------v
      usetcoefs=4
c ^-------CHANGE-ME-------------------^
```

```
      end


c UGETCHARGES FUNCTION

      logical function ugetcharges()
c specify if you need a charges file
c v-------CHANGE-ME-------------------v
      ugetcharges=.false.
c ^-------CHANGE-ME-------------------^
      end

c USERPOT SUBROUTINE

      subroutine userpot(geo,x,nmax,vpot)
      use vglobales
c ------------------------------------
c to use your external data
      use userdata
c ------------------------------------
      integer nmax,geo,i,j,k
      double precision d,vpot,userv
      double precision X(nmax)
c v-------CHANGE-ME-IF-NEEDED----------v
      vpot=0.0d0
c note: here all interactions are calculated
      do i=1,nprox
       do j=1,nsam
       k=j+nprox
       d=r(geo,i,j)
       vpot=vpot+userv(d,i,k,x,nmax)
       enddo
      enddo
c ^-------CHANGE-ME-IF-NEEDED----------^
      return
      end


c FUNCTION USER POTENTIAL
c  write userv using ix function to access
c  individual coefficients.
c  use CALL coordinates(geometry,atom,x,y,z)
c  to access individual coordinates.

      double precision FUNCTION userv(r,i,j,x,m)
      implicit none
      integer i,j,m,ix
      dimension x(m)
c note: here ONE interaction is calculated
c v-------CHANGE-ME-------------------v
      double precision x,r,a,b,c,d
      A=x(ix(i,j,1))
      B=x(ix(i,j,2))
      C=x(ix(i,j,3))
      D=x(ix(i,j,4))
      userv=A*EXP(-B*R)+C/R**D
c ^-------CHANGE-ME-------------------^
      RETURN
      END
```

```
c USER FITTING FUNCTION
c  write here the user fitting function
c  if you only need the fitting function
c  leave the line "call potRouter..." unchanged
c  and  change the line "userfitting=..." with your
c  fitting function.
c  if you have a userv function (above this), you can
c  use it here, or access it via potRouter

      double precision function userfitting(x,m,geo)
      use vglobales
      use userdata
      double precision x,vpot
      integer m,geo
      dimension x(m)
c v-------CHANGE-ME-------------------v
      call potRouter(geo,x,m,vpot)
      userfitting=(v(geo)-vpot)*(v(geo)-vpot)
c ^-------CHANGE-ME-------------------^
      return
      end
```