Contents lists available at ScienceDirect

Computer Physics Communications

www.elsevier.com/locate/cpc

Gmat. A software tool for the computation of the rovibrational G matrix ☆

M.E. Castro, A. Niño, C. Muñoz-Caro *

Grupo de Química Computacional y Computación de Alto Rendimiento, Escuela Superior de Informática, Universidad de Castilla-La Mancha, Paseo de la Universidad 4, 13071 Ciudad Real, Spain

ARTICLE INFO

Article history:
Received 10 September 2008
Accepted 7 January 2009
Available online xxxx

PACS:
31.15.xv
33.15.Mt
33.20.Vq
02.60.Jh

Keywords:
Rovibrational G matrix
Principal inertia axes
Vibrational coordinates
Object-oriented programming

ABSTRACT

Gmat is a C++ program able to compute the rovibrational G matrix in molecules of arbitrary size. This allows the building of arbitrary rovibrational Hamiltonians. In particular, the program is designed to work with the structural results of potential energy hypersurface mappings computed in computer clusters or computational Grid environments. In the present version, 1.0, the program uses internal coordinates as vibrational coordinates, with the principal axes of inertia as body-fixed system. The main design implements a complete separation of the interface and functional parts of the program. The interface part permits the automatic reading of the molecular structures from the output files of different electronic structure codes. At present, Gamess and Gaussian output files are allowed. To such an end, use is made of the object orientation polymorphism characteristic. The functional part computes numerically the derivatives of the nuclear positions respect to the vibrational coordinates. Very accurate derivatives are obtained by using central differences embedded in a nine levels Richardson extrapolation procedure.

Program summary

Program title: Gmat
Catalogue identifier: AECZ_v1_0
Program summary URL: http://cpc.cs.qub.ac.uk/summaries/AECZ_v1_0.html
Program obtainable from: CPC Program Library, Queen's University, Belfast, N. Ireland
Licensing provisions: Standard CPC licence, http://cpc.cs.qub.ac.uk/licence/licence.html
No. of lines in distributed program, including test data, etc.: 17 023
No. of bytes in distributed program, including test data, etc.: 274 714
Distribution format: tar.gz
Programming language: Standard C++
Computer: All running Linux/Windows
Operating system: Linux, Windows
Classification: 16.2
Nature of problem: Computation of the rovibrational G matrix in molecules of any size. This allows the building of arbitrary rovibrational Hamiltonians. It must be possible to obtain the input data from the output files of standard electronic structure codes. In addition, the program should handle the large number of files generated in massive explorations of molecular potential energy hypersurfaces. In these cases, Gmat will provide the G matrix as a function of the molecular structure.
Solution method: To reach its objectives, Gmat has been organized in two components: an interface and a functional part. This organization allows for separating the input/output tasks, which are dependent on the human–machine interaction model selected, from the functional requirements, which are not. An object-oriented approach has been used in both parts. In the interface, polymorphism is used to allow the data acquisition from output files of different electronic structure codes. In the functional part, Gmat computes numerically the derivatives of the Cartesian coordinates respect to the vibrational coordinates needed to build the G matrix. Extremely accurate numerical derivatives are obtained in a double procedure. First, the truncation plus roundoff errors are minimized in the central differences expression. Second, the result is embedding in a nine levels Richardson extrapolation process. In the present version, the program allows the use of internal coordinates as vibrational coordinates, with the principal axes of inertia as body-fixed system.

☆ This paper and its associated computer program are available via the Computer Physics Communications homepage on ScienceDirect (http://www.sciencedirect.com/science/journal/00104655).

* Corresponding author. Tel.: +34 926295362; fax: +34 926295354.
E-mail address: camelia.munoz@uclm.es (C. Muñoz-Caro).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2009.01.006

Running time: Sample test runs provided with the distribution take a few seconds to execute.

1. Introduction

The G matrix fully defines the kinetic operator of the molecular rotational–vibrational Hamiltonian [1–4]. In other words, the G matrix defines the kinetic energy for the molecular rotation and for any intramolecular motion. After computing the G matrix, the rotational, rotation–vibration and pure vibrational kinetic terms are provided. Therefore, except the potential function, it provides all the information needed to build any non-rigid, anharmonic, rovibrational Hamiltonian. Thus, for instance, in pure vibrational studies, reliable Hamiltonians including anharmonicity and coupling among several vibrational motions [5–8] can be constructed.

In order to develop and use rovibrational Hamiltonians, we need to compute the rovibrational G matrix in molecules of arbitrary size. It would be desirable to include as many internal, vibrational, degrees of freedom as needed. In these kinds of studies, one interesting point is the evaluation of an accurate potential energy hypersurface for the nuclear motion of the molecular system, by using electronic structure models. In particular, the massive exploration of the hypersurface, as a function of the molecular structure, is possible using electronic structure codes running on computer clusters or Internet-based Grid of computers [9]. In these cases, it would be desirable to process automatically this information, and compute the G matrix elements for the different molecular structures as a function of the vibrational coordinates. To address the previous issues we have designed a software tool: Gmat. Gmat is written in C++ because, apart from being an efficient language, it has useful object-oriented characteristics such as encapsulation, inheritance and polymorphism [10]. Such characteristics make easy the future maintainability and extensibility of the program.

2. Theory

Using coordinates with origin at the center of mass, the quantum-mechanical kinetic operator for the nuclear motion [1–5] has the following form,

$$\hat{\mathrm{T}} = -\frac{\hbar^2}{2}\sum_{i}^{3N-3}\sum_{j}^{3N-3}\left[g_{ij}\frac{\partial^2}{\partial q_i \partial q_j} + \left(\frac{\partial g_{ij}}{\partial q_i}\right)\frac{\partial}{\partial q_j}\right], \tag{1}$$

where g_{ij} are the elements of the rovibrational G matrix, and the i and j indexes run on the rovibrational degrees of freedom $(3N-3)$. In turn, the rovibrational G matrix is defined as [4,5],

$$\mathrm{G} = \begin{pmatrix} \mathrm{I} & \mathrm{X} \\ \mathrm{X}^T & \mathrm{Y} \end{pmatrix}^{-1}. \tag{2}$$

Here, I represents the inertial tensor, *i.e.*, the overall rotation contribution. Y corresponds to the pure vibration contribution, and X represents the vibration–rotation interaction contribution. The elements of these matrices are obtained from the molecular structure as,

$$I_{ii} = \sum_{\alpha}^{n} m_\alpha\left[(r_\alpha)^2 - (r_{\alpha i})^2\right],$$

$$I_{ij} = -\sum_{\alpha}^{n} m_\alpha r_{\alpha i} r_{\alpha j},$$

$$X_{ij} = \sum_{\alpha}^{n} m_\alpha\left[r_\alpha \times \left(\frac{\partial r_\alpha}{\partial q_i}\right)\right]_j,$$

$$Y_{ij} = \sum_{\alpha}^{n} m_\alpha\left(\frac{\partial r_\alpha}{\partial q_i}\right)\cdot\left(\frac{\partial r_\alpha}{\partial q_j}\right), \tag{3}$$

where the α index runs on all the atoms in the molecule.

Accurate molecular structures can be obtained from the results of electronic structure calculations, *i.e.*, molecular structure optimizations. Therefore, for a given molecular structure, we can build the G matrix if we can compute the derivatives of the nuclear positions respect to the vibrational coordinates appearing in Eq. (3).

3. Numerical

We evaluate numerically, rather than analytically, the derivatives in Eq. (3). In this form, the use of arbitrary vibrational coordinates is simplified. To obtain accurate numerical derivatives the general approach described in Ref. [8] is used. Thus, we start by determining the optimal increment of the vibrational coordinate, Δq_i, which minimizes the truncation plus roundoff errors in the central differences expression. Therefore, for a given component, $r_{\alpha j}$, of the α atom position vector we have,

$$\Delta q_i = \left[\varepsilon_f \frac{r_{\alpha j}(q_i)}{(\partial^3 r_{\alpha j}(q_i)/\partial q_i^3)}\right]^{1/3}. \tag{4}$$

Here ε_f is the fractional accuracy with which $r_{\alpha j}(q_i)$ is computed. This value is at least equal to the machine epsilon (the machine precision). So, in a first approach we have $\varepsilon_f \approx \varepsilon_m$. The ε_m for a floating point value in double precision using the IEEE 754 standard representation is about 10^{-16} [11].

Considering the set of $3N$ derivatives, we compute an overall optimal increment as the square root of the quadratic average. So, the optimal Δq overall increment (the step size, h) would be,

$$h = \left[\sum_{i}^{N}\sum_{j}^{3}\left[3\varepsilon_f \cdot \frac{r_{\alpha j}(q_i)}{(\partial^3 r_{\alpha j}(q_i)/\partial q_i^3)}\right]^{2/3} \Big/ 3N\right]^{1/2}. \tag{5}$$

Eq. (5) represents the optimal step size for the computation of the whole set of derivatives. Finally, to enhance the accuracy of the computed derivatives we apply a Richardson extrapolation [8]. Using Richardson extrapolation accurate numerical derivatives are obtained in an iterative procedure with the following recurrence relation between two applications, i and $i+1$, of the extrapolation,

$$D_{i+1}(h) \cong \frac{2^{2(i+1)} D_i(h/2) - D_i(h)}{2^{2(i+1)} - 1}. \tag{6}$$

As shown in Eq. (6), we need different numerical derivatives starting with a step interval h, which is successively halved. The optimal h value obtained in Eq. (5) is considered as the limiting h value in the Richardson procedure. As error we use an index measuring the overall difference among two successive applications of the Richardson extrapolation. This index has the advantage of being related to the number of significant digits. To define the index, we use an approach similar to that applied in Eq. (5), defining the error index for a whole set of m internal coordinates as,

$$\varepsilon = \left[\sum_{k}^{m}\sum_{i}^{N}\sum_{j}^{3}\left[D_{ik}(h/2) - D_{ik}(h)\right]^2 / [3Nm]\right]^{1/2}. \tag{7}$$

Ref. [8] applies the previous procedure to the case of internal coordinates (bond distances, valence and dihedral angles) on

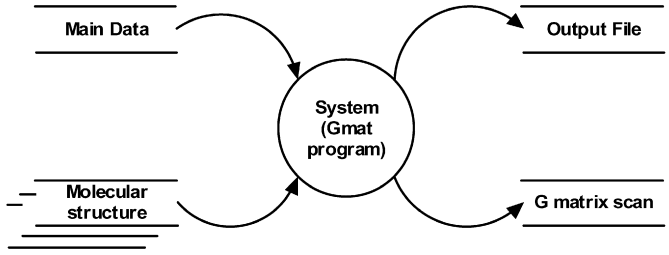

Fig. 1. System context diagram for the Gmat program.

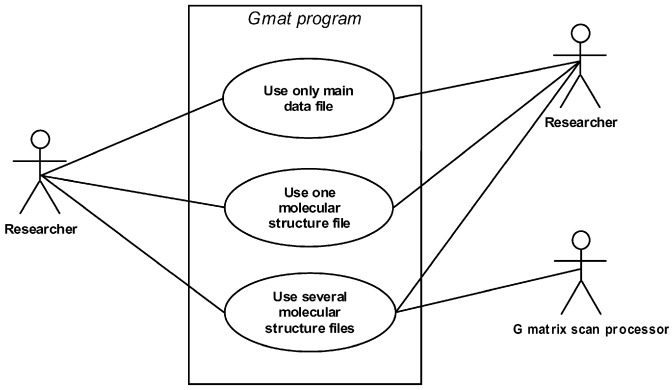

Fig. 2. UML use case diagram for the Gmat program.

machines using the IEEE 754 standard double precision representation. Here, different combinations of initial h steps versus number of Richardson extrapolations are considered. The results in Ref. [8] show that a nine levels Richardson extrapolations procedure with initial step values of 10^{-2} Angstroms, for bond lengths and 10^{-4} degrees, for valence and dihedral angles, provides the more accurate results.

4. Structure of the program

The methodology described in Sections 2 and 3 was implemented in the C++ program Gmat. Standard C++ has been used to provide object-oriented capabilities [10]. All data structures are dynamically allocated in run time. The structure of each class (attributes and visibility) is fully documented in the (.h) definition file of each class. A description of the general functionality and general organization of the program is presented in the following subsections.

4.1. General functionality

The general functionality of the Gmat program is shown in a classical system context diagram [12] in Fig. 1. We can see that the program accepts one main data file and, optionally, one or several molecular structure files. These files are the output of standard electronic structure programs such as: Gamess, Gaussian, Dalton, NWChem, Molpro, MolCAS, etc. On output, the program generates a general output file and, optionally, an additional file containing the G matrix for a series of molecular structures.

According to the searched versatility of the program, three use cases are considered. These cases are represented in Fig. 2 as an UML use case diagram [13]. Fig. 2 shows that in the first case all the information, including the molecular structure, is supplied directly by the user in a single file, the main data file. In the second case, the general information is provided in the main data file, except the molecular structure data. This is directly read from the output file of one of the available molecular electronic structure

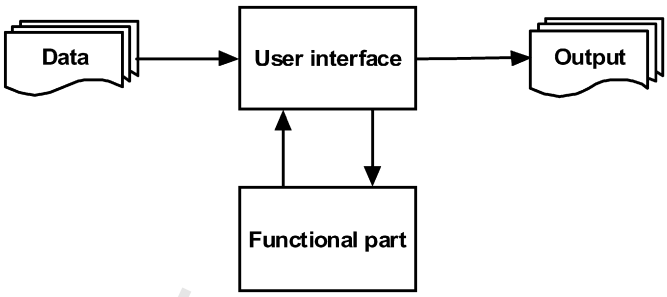

Fig. 3. Block diagram of the Gmat program.

packages. This case is intended to facilitate the use of information obtained from molecular structure optimizations. Finally, the third case is implemented for computing the G matrix for each one of the large number of molecular structures obtained from a molecular potential energy hypersurface scan. The result is a file containing the G matrices for all the molecular structures as a function of the considered vibrational coordinates. This model allows for further interfacing with different software tools able to build and process the rovibrational molecular Hamiltonian.

4.2. General organization

The general structure of the Gmat program is shown in Fig. 3. Gmat has been organized into two main components: an interface and a functional part. This organization allows the independent management of the input/output tasks, which are dependent on the human–machine interaction model, from the functional requirements, which are not. In both parts, an object-oriented approach has been used. The interface part is responsible for the acquisition of data from the general data file, and (if needed) from the output files of electronic structure packages, see Fig. 3. In this case, Gmat must identify and read the corresponding output format. On the other hand, the functional part is responsible for the computation of the G matrix elements.

The interface part of the program resorts to the object orientation characteristic of polymorphism. This is defined as the ability of objects of different classes to respond to method or function calls of the same name [14]. This allows the handling, in a unified way, of the different data formats found in the standard electronic structure codes. To such an end, an inheritance relationship is needed. Then, a reference of base (parent) class can be used to refer to any object of a descendant class. Fig. 4 shows an UML class diagram corresponding to the interface part of Gmat. The MolecularData class is the base (abstract) class that defines the reading data methods. This class uses the information from the AtomicData package about atomic symbols and atomic masses, implemented as a namespace. For the masses, the average atomic mass data provided by the NIST are used [15]. The classes derived from MolecularData contain the specific code for reading from each specific electronic structure package. GaussianMolecularData for Gaussian [16] and GamessMolecularData for Gamess [17] are available in the present, 1.0, version of the program.

On the other hand, the functional part is structured in several classes. Fig. 5 presents an UML diagram class for the functional part of the program. The Gmatrix class contains and handles all the information about the rovibrational G matrix. This is the base (parent) class for the inheritance hierarchy of classes related to the computation of Gmatrix in different coordinates. This class contains all the necessary methods to build the full rovibrational G matrix described in Section 2. In this class, the inertia tensor, I, the rotation–vibration interaction X, and the vibrational Y matrices are computed, see Eq. (3). The class includes the necessary methods to obtain the center of mass and the principal axes co-

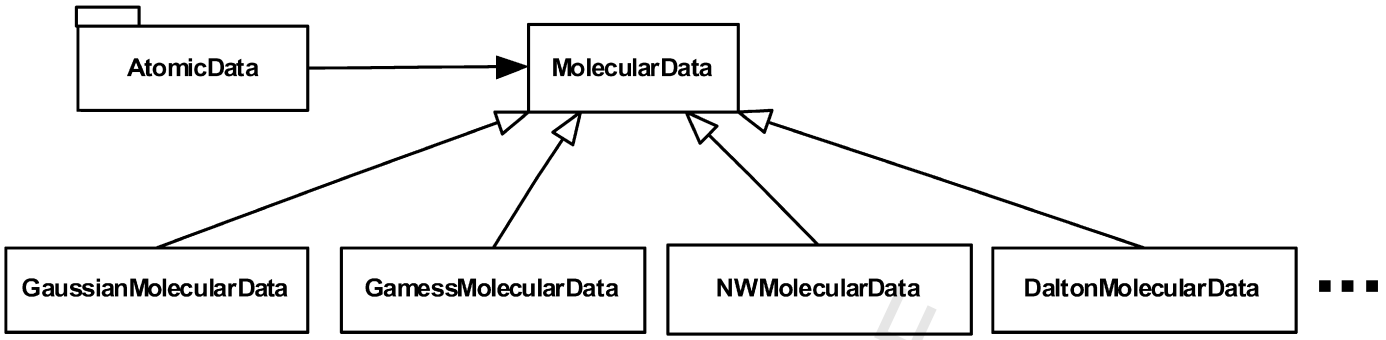

Fig. 4. UML class diagram for the interface part of the Gmat program.

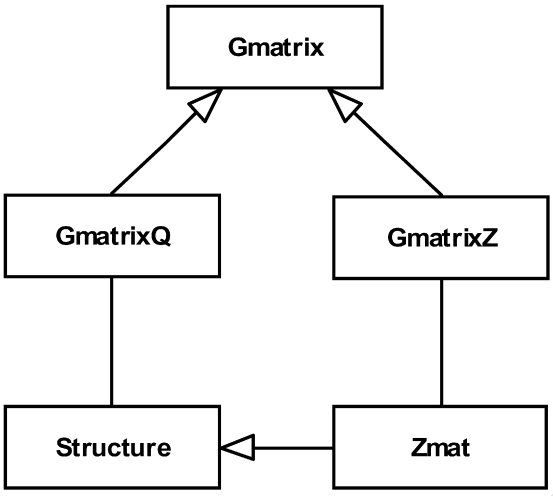

Fig. 5. UML class diagram for the functional part of the Gmat program.

ordinates. The GmatrixZ class extends Gmatrix by inheritance, and it handles the G matrix in arbitrary internal coordinates. This class implements the methodology developed in Section 3 for computing accurate numerical derivatives [8]. GmatrixQ extends Gmatrix by inheritance, allowing the use of normal modes as vibrational coordinates. In the present version of the program this class is not yet implemented.

The Structure class contains and handles the general information of a molecular system, like number of atoms, atomic numbers, Cartesian coordinates, etc. The continuous line in Fig. 5 shows that this class is used by GmatrixQ. The Zmat class contains the definition and values of the internal coordinates. This class also contains the methods for converting to internal from Cartesian coordinates, and to Cartesian from internal coordinates. Zmat exhibits an inheritance relationship with Structure, and it is used by GmatrixZ, see Fig. 5.

At present, Gmat, version 1.0, is able to compute the rovibrational G matrix for a molecular system with any number of atoms. It uses internal coordinates as vibrational coordinates. The vibrational coordinates can be defined as interatomic distances, valence angles or dihedral angles. Dummy atoms can be used in the molecular structure definition. In addition, isotopic substitution of hydrogen by deuterium is also allowed. The principal axis of inertia system is used as the reference body-fixed frame.

Gmat computes the G matrix for one or several molecular structures. When just one structure is used it can be input in the main data file (as Cartesian coordinates) or in an independent file. In this last case, the data are assumed to come from the output of a molecular electronic structure package. At present, both Gamess and Gaussian output files can be used. Also, Gmat works with a set of electronic structure result files. In this last case, Gmat allows to process in a single shot the large number of molecular structure files generated in molecular potential energy hypersurface scans.

Further releases are planned to extend the capabilities of Gmat, both in the interface and in the functional components as follows:

In the interface part:

- To allow the direct use of output files from additional electronic structure codes.

In the functional part:

- To introduce different sets of vibrational coordinates. For instance, the implementation of the GmatrixQ class allowing the use of normal modes.
- To work in different molecular structure reference frames, in addition to the principal axes of inertia system.

Acknowledgements

This work has been cofinanced by FEDER funds and the *Consejería de Educación y Ciencia de la Junta de Comunidades de Castilla-La Mancha* (grant #PBI08-0008). The *Ministerio de Educación y Ciencia* (grant #FIS2005-00293) is also acknowledged. M.E. Castro thanks the *Consejo Nacional de Ciencia y Tecnología*, *CONACyT* (Mexico) for a graduate grant (grant #171982).

References

[1] P.R. Bunker, P. Jensen, Molecular Symmetry and Spectroscopy, second ed., NRC-Research Press, Ottawa, 1998.
[2] D. Papousek, M.R. Aliev, Molecular Vibrational/Rotational Spectra, Academia, Prague, 1982.
[3] H.M. Pickett, J. Chem. Phys. 56 (1972) 1715.
[4] M.A. Harthcock, J. Laane, J. Chem. Phys. 89 (1985) 4231.
[5] (a) A. Niño, C. Muñoz-Caro, Computers Chem. 18 (1) (1994) 27;
(b) C. Muñoz-Caro, A. Niño, QCPE Bull. 13 (1993) 4.
[6] A. Niño, C. Muñoz-Caro, M. Mora, S. Reyes, J. Phys. Chem. A 107 (47) (2003) 10191.
[7] M.E. Castro, A. Niño, C. Muñoz-Caro, Theor. Chem. Account. 119 (4) (2008) 343.
[8] M.E. Castro, A. Niño, C. Muñoz-Caro, Lecture Notes in Computer Science 5072 (2008) 1011.
[9] S. Reyes, C. Muñoz-Caro, A. Niño, R.M. Badia, J.M. Cela, Concurrency and Computation: Practice and Experience 19 (2007) 463.
[10] S. Bjarne, The C++ Programming Language, Addison-Wesley Professional, Indianapolis, 2000.
[11] IEEE Std 754-1985, IEEE Standard for Binary Floating-Point Arithmetic.
[12] T. DeMarco, Structured Analysis and System Specification, Prentice-Hall, New Jersey, 1979, p. 75.
[13] J. Rumbaugh, I. Jacobson, G. Booch, The Unified Modeling Language Reference Manual, Addison-Wesley Object Technology Series, Addison-Wesley, Boston, 2005, p. 219.
[14] B. Meyer, Object-Oriented Software Construction, Prentice-Hall, New Jersey, 2000, p. 459.
[15] National Institute for Standards and Technology, http://www.nist.gov/. Last visit August 15, 2008.

[16] M.J. Frisch, G.W. Trucks, H.B. Schlegel, G.E. Scuseria, M.A. Robb, J.R. Cheeseman, J.A. Montgomery Jr., T. Vreven, K.N. Kudin, J.C. Burant, J.M. Millam, S.S. Iyengar, J. Tomasi, V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega, G.A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, M. Klene, X. Li, J.E. Knox, H.P. Hratchian, J.B. Cross, C. Adamo, J. Jaramillo, R. Gomperts, R.E. Stratmann, O. Yazyev, A.J. Austin, R. Cammi, C. Pomelli, J.W. Ochterski, P.Y. Ayala, K. Morokuma, G.A. Voth, P. Salvador, J.J. Dannenberg, V.G. Zakrzewski, S. Dapprich, A.D. Daniels, M.C. Strain, O. Farkas, D.K. Malick, D. Rabuck, K. Raghavachari, J.B. Foresman, J.V. Ortiz, Q. Cui, A.G. Baboul, S. Clifford, J.A. Cioslowski, B.B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R.L. Martin, D.J. Fox, T. Keith, M.A. Al-Laham, C.Y. Peng, A. Nanayakkara, M. Challacombe, P.M.W. Gill, B. Johnson, W. Chen, M.W. Wong, C. Gonzalez, J.A. Pople, Gaussian 03 program. Revision A. 1, Gaussian, Inc., Pittsburgh PA, 2003.

[17] M.W. Schmidt, K.K. Baldridge, J.A. Boatz, S.T. Elbert, M.S. Gordon, J.H. Jensen, S. Koseki, N. Matsunaga, K.A. Nguyen, S. Su, T.L. Windus, M. Dupuis, J.A. Montgomery, J. Comput. Chem. 14 (1993) 1347.

Please cite this article in press as: M.E. Castro et al., Gmat. A software tool for the computation of the rovibrational G matrix, Computer Physics Communications (2009), doi:10.1016/j.cpc.2009.01.006