XIX SBQT
Simpósio Brasileiro de Química Teórica 2017
12 a 17/Nov, 2017, Águas de Lindóia/SP, Brasil

High Performance Collision Cross Section Calculation - HPCCS

Gabriel Heerdt[a], Leandro N. Zanotto[a,b], Paulo C. T. Souza[a,c], Guido C. S. Araujo[b], Munir S. Skaf[a]

[a]*Institute of Chemistry, University of Campinas, Barão Geraldo, 13083-852, Campinas - São Paulo, Brazil.*

[b]*Institute of Computing, University of Campinas, Barão Geraldo, 13083-970, Campinas - São Paulo, Brazil.*

[c]*Faculty of Mathematics and Natural Sciences, University of Groningen, Groningen 9747, AG, The Netherlands.*

Ion Mobility Spectrometry (IMS) is a widely used and 'well-known' technique of ion separation in the gaseous phase, based on the differences in ion shape and charge. Despite IMS is a fairly rapid experiment, the data interpretation is still a challenge and depends on accurate theoretical estimates of the molecule Collision Cross Section (CCS).

The *Mobcal* software [1-3], developed in 1996 by Jarrold et al., is the most widely used program for theoretical CCS calculations. *Mobcal* uses three different algorithms to calculate CCS: Projection Approximation (PA), Exact Hard Sphere Scattering (EHSS) and Trajectory Method (TM). TM is the most accurate one, being the best choice for CCS estimates, especially for highly-charged macromolecules as proteins.[4] As demonstrated in Fig. 1 (B), the orientationally averaged collision integral ($\Omega_{avg}^{(1,1)}$) can be calculated by integrating over all scattering angles (χ). However, such estimates involve performing thousands molecular dynamics simulations per analyzed molecule (Fig. 1 (A)), with a very high computational cost.

Since the original Mobcal were written for serial execution with Fortran language, it is not prepared for the technology we have today. Herein, we present a new software to High Performance Collision Cross Section Calculation (HPCCS). Our software, focused only in Trajectory Method, was totally rewritten in C++ language, improved with new features and increased the performance using High Performance Computing (HPC) techniques.

On the first release, the code rewrite and CPU parallelization and vectorization [5] were completed. Large biomolecules can be simulated in a short time, within high accuracy, which helps the interpretation of experimental IM-MS data. As can be seen in Fig. 1 (C), the theoretical CCS results, performed with the TM by our program, are in good agreement with experimental data.

XIX SBQT
Simpósio Brasileiro de Química Teórica 2017
12 a 17/Nov, 2017, Águas de Lindóia/SP, Brasil

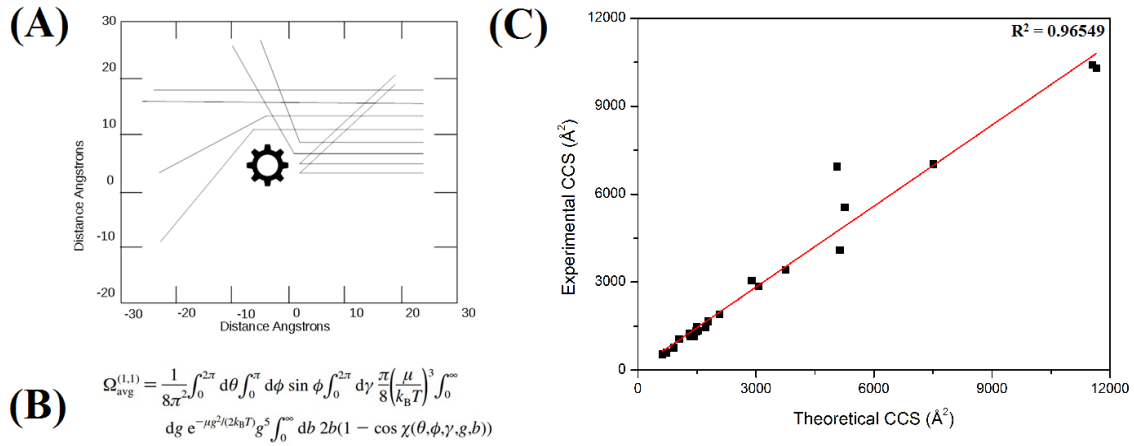

Figure 1: Actually, the accurate estimation of CCS (Ω in the formula**)** requires the calculation of the all possible collision angles (χ, indicated at (B)) between a buffer gas and a target molecule, as shown in (A). Such angles can be generated by individual MD simulations, they depend on the relative orientation of the molecule, size, shape and interaction with the gas. (C) Experimental versus Theoretical Collision Cross Section, Å^2.

As for the computational time, for the largest simulated protein complex, which contains approximately 32,000 atoms, the total time considering 32 threads was about 4 hours. This means that for a common user it is perfectly possible to simulate on your desktop, usually with 2 - 4 threads, CCSs for proteins and their complexes in a viable time. The IM-MS experiments combined with appropriate simulations are an excellent tool to obtain biomolecule structure information.

Key-words: Ion Mobility, Mass Spectrometry, Collision Cross Section, Trajectory Method, High Performance Computing
Support: This work has been supported by FAPESP: 2016/04963-6, 2013/08293-7.
References:
[1] A. A. Shvartsburg, M. F. Jarrold, *Chem. Phys. Lett.*, 261, 86 (1996).
[2] M. F. Mesleh, J. M. Hunter, A. A. Shvartsburg, G. C. Schatz, M. F. Jarrold, *J. Phys. Chem.*, 100, 16082 (1996).
[3] A. A. Shvartsburg, G. C. Schatz, M. F. Jarrold, *J. Chem. Phys.*, 108, 2416 (1998).
[4] F. Lanucara, S. W. Holman, C. J. Gray, C. E. Eyers, *Nat. Chem.*, 6, 281 (2014).
[5] J. Jeffers, J. Reinders, "Intel Xeon Phi coprocessor high performance programming" (2013), Elsevier, Newnes.