QUANTUM: A Wolfram *Mathematica* add-on for Dirac Bra-Ket Notation, Non-Commutative Algebra, and Simulation of Quantum Computing Circuits

QUANTUM: A Wolfram *Mathematica* add-on for Dirac Bra-Ket Notation, Non-Commutative Algebra, and Simulation of Quantum Computing Circuits

J L Gómez Muñoz and F Delgado

Departamento de Física y Matemáticas, Escuela de Diseño, Ingeniería y Arquitectura, Tecnológico de Monterrey, Campus Estado de México, Atizapán, Estado de México, CP. 52926, México.

E-mail: `jose.luis.gomez@itesm.mx, fdelgado@itesm.mx`

Abstract. This paper introduces QUANTUM, a free library of commands of Wolfram *Mathematica* that can be used to perform calculations directly in Dirac braket and operator notation. Its development started several years ago, in order to study quantum random walks. Later, many other features were included, like operator and commutator algebra, simulation and graphing of quantum computing circuits, generation and solution of Heisenberg equations of motion, among others. To the best of our knowledge, QUANTUM remains a unique tool in its use of Dirac notation, because it is used both in the input and output of the calculations. This work depicts its usage and features in Quantum Computing and Quantum Hamilton Dynamics.

1. Introduction

QUANTUM is an add-on of *Mathematica* (versions 8 to 10) implementing calculations in Dirac notation [1], and allowing the end user to enter that notation using keyboard and palettes toolbars. As a simple example, the end user can easily write and obtain the following result:

$$\begin{aligned}\text{Input :}\quad & |\psi\rangle = \alpha\,|\phi_1\rangle + \beta\,|\phi_2\rangle\,; \\ & \text{Expand}[\,|\psi\rangle\cdot\langle\psi|\,] \\ \text{Output :}\quad & \alpha\alpha^*\,|\phi_1\rangle\,\langle\phi_1| + \alpha\beta^*\,|\phi_1\rangle\,\langle\phi_2| + \beta\alpha^*\,|\phi_2\rangle\,\langle\phi_1| + \beta\beta^*\,|\phi_2\rangle\,\langle\phi_2|\end{aligned} \tag{1}$$

The core of QUANTUM was constructed based on quantum information and quantum computing applications, therefore several of the advanced commands in the current version are restricted to discrete Hilbert spaces. The programming philosophy and implementation of QUANTUM are depicted in the second section. QUANTUM has three main modules; the third section describes the first one, Quantum'Notation', implementing bra-ket notation, non-commutative algebra of operators and commutators, and quantum measurements. The fourth section describes the second module, Quantum'Computing', including qubits and quantum gates for the simulation of algorithms and the automated drawing of quantum computing circuits. The third module in the fifth section, Quantum'QHD', implements the Quantized Hamilton Dynamics (QHD) approximation to the Heisenberg equations of motion. It can be used for the simulation of physical systems such as molecules. Last section is devoted to the conclusions.

2. Programming Philosophy and Implementation

Mathematica has a very consistent notation. For example, the standard command to graph a sine function from 0 to π is Plot[Sin[x],{x,0,Pi}], on the other hand, to integrate that function in the same interval the command is almost identical: Integrate[Sin[x],{x,0,Pi}]. We have implemented the same consistent behavior in the QUANTUM commands. As an example, the commands to obtain the bra-ket expression and the quantum computing circuit of a sequence of quantum computing gates are almost identical, as can be seen in Figure 1. Sometimes the Dirac notation, as it is used in books and papers of Quantum Mechanics and Quantum Computing, is not explicit and flexible enough for the calculations [2–4]. Therefore QUANTUM generalized this notation, as it is seen in the CNOT gate notation in Figure 1.

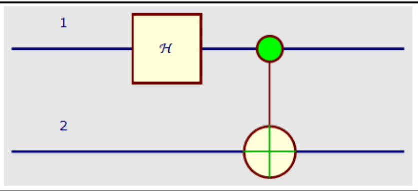
`QuantumEvaluate[`$C^{\{\hat{1}\}}[\mathcal{NOT}_{\hat{2}}]\cdot\mathcal{H}_{\hat{1}}$`]`	`QuantumPlot[`$C^{\{\hat{1}\}}[\mathcal{NOT}_{\hat{2}}]\cdot\mathcal{H}_{\hat{1}}$`]`
$\frac{\vert 0_{\hat{1}},0_{\hat{2}}\rangle\cdot\langle 0_{\hat{1}},0_{\hat{2}}\vert}{\sqrt{2}}+\frac{\vert 1_{\hat{1}},1_{\hat{2}}\rangle\cdot\langle 0_{\hat{1}},0_{\hat{2}}\vert}{\sqrt{2}}+\frac{\vert 0_{\hat{1}},1_{\hat{2}}\rangle\cdot\langle 0_{\hat{1}},1_{\hat{2}}\vert}{\sqrt{2}}+\frac{\vert 1_{\hat{1}},0_{\hat{2}}\rangle\cdot\langle 0_{\hat{1}},1_{\hat{2}}\vert}{\sqrt{2}}+\frac{\vert 0_{\hat{1}},0_{\hat{2}}\rangle\cdot\langle 1_{\hat{1}},0_{\hat{2}}\vert}{\sqrt{2}}-\frac{\vert 1_{\hat{1}},1_{\hat{2}}\rangle\cdot\langle 1_{\hat{1}},0_{\hat{2}}\vert}{\sqrt{2}}+\frac{\vert 0_{\hat{1}},1_{\hat{2}}\rangle\cdot\langle 1_{\hat{1}},1_{\hat{2}}\vert}{\sqrt{2}}-\frac{\vert 1_{\hat{1}},0_{\hat{2}}\rangle\cdot\langle 1_{\hat{1}},1_{\hat{2}}\vert}{\sqrt{2}}$	

Figure 1. The same syntax is used to obtain a bra-ket expression and to graph the corresponding circuit. This consistent behavior was implemented across all QUANTUM commands.

An important aspect of Quantum'Notation' module compared with other packages [5–7] implementing quantum computations (based on matrix notation) is the use of an algebraic management (based on Dirac notation). The first approach could be computationally inefficient because states and operators become typically big after tensor products. Then, matrix products could be a slow technique compared with the algebraic, where only non-zero terms appear. There are several outstanding aspects there: a) the natural and easier way to introduce operators and kets, in particular for problems using a big number n of qudits (d-level quantum systems); b) the comparative speed performance in the calculations between algebraic management $O(d^n log(d^n))$ versus straight matrix management of calculations $O(d^{2n})$, at least in the most common problems in quantum information, which can be expressed by sparse matrices of order $O(log(d^n))$ (despite other packages could implement sparse matrix multiplication techniques, in QUANTUM this process is automatic); and c) the capabilities of *Mathematica* to do parallel computations which has reduced in half the QUANTUM processing time for large number of qudits in a benchmarking for problems in Adiabatic Quantum Computation [8].

To implement QUANTUM, several advanced *Mathematica* techniques were used, including the unprotection and redefinition of standard *Mathematica* commands like Expand and Simplify, instead of just be included in the general commands of the tool; the use of operators without built-in meaning, like CenterDot; and the low-level programming of *Mathematica* box constructs, like RowBox, SuperscriptBox and TagBox [9].

3. Quantum Notation Module

The Quantum'Notation' module implements the non-commutative algebra of operators, commutators, and the Dirac bra-ket notation. It includes commands for the simulation of quantum measurements, the calculation of partial trace, partial transpose, among others.

Figure 2 shows a hierarchy of some of the commands in this module as well as groups of commands developed mainly based on Kets, Operators and Operations there. Thus, main basic operations used in Dirac notation for discrete basis as qubits, qudits, operators, adjoint, scalar,

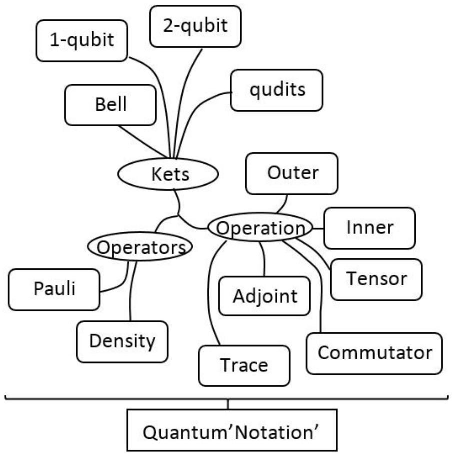
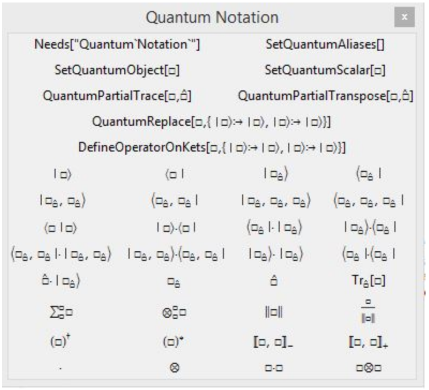


Figure 2. Main sections developed in Quantum'Notation' module integrating the basic aspects in Dirac notation: bra-ket's, operators and operations.

inner and tensor products, as well as other specific objects in the form of Bell states, Pauli matrices and Partial Trace operations are included. One of the palettes of this module ease the introduction of the notation containing the main elements in the module (Figure 3). Table 1 shows some of the main templates and their usage, which are entered either using the palettes or keyboard shortcuts. There, expressions after of symbol = represent traditional writing.



```
Quantum Notation
Needs["Quantum`Notation`"]                 SetQuantumAliases[]
SetQuantumObject[□]                        SetQuantumScalar[□]
QuantumPartialTrace[□,□̂]                   QuantumPartialTranspose[□,□̂]
QuantumReplace[□,{ |□⟩:→ |□⟩, |□⟩:→ |□⟩}]
DefineOperatorOnKets[□,{ |□⟩:→ |□⟩, |□⟩:→ |□⟩}]
```

Figure 3. Quantum'Notation' palette showing the principal commands implemented in the module: bra-ket's and basic operations.

Thus, with these basic constructions, it is possible set and calculate states, operators, projections, measurements, probabilities and traces from quantum objects operating on a tensor product basis of discrete Hilbert spaces, covering lots of possibilities for quantum systems.

4. Quantum Computing Module

The module named Quantum'Computing' includes quantum computing gates and qubits. The bra-ket evaluation of the quantum computing gates is delayed, so that they can be used to generate circuits, as in Figure 1, and can be also used in algebra, for example to expand them in terms of Pauli operators. Bell states, Quantum Fourier Transform and measurements are also included. Figure 4 shows a hierarchy of some of the commands in this module. As in Quantum'Notation', some palettes (Quantum Computing Kets and Quantum Computing

Ket-Bra's		Operations	
Template	Example	Template	Example
$\left\lvert \square_{\widehat{\square}}, \square_{\widehat{\square}}, \square_{\widehat{\square}}, \right\rangle$	$\lvert 0_{\hat{1}}, 0_{\hat{2}}, 1_{\hat{3}}\rangle$	$(\square)^{\dagger}$	$(\lvert 1_{\hat{2}}\rangle)^{\dagger} = \langle 1_2\rvert$
$\left\lvert \square_{\widehat{\square}} \right\rangle \cdot \left\langle \square_{\widehat{\square}} \right\rvert$	$\lvert 0_{\hat{1}}\rangle \cdot \langle 1_{\hat{1}}\rvert$	$\left\lvert \square_{\widehat{\square}} \right\rangle \otimes \left\lvert \square_{\widehat{\square}}, \square_{\widehat{\square}} \right\rangle$	$\lvert 1_{\hat{3}}\rangle \otimes \lvert 0_{\hat{1}}, 1_{\hat{2}},\rangle = \lvert 0_1 1_2 1_3\rangle$
$\left\lvert \square_{\widehat{\square}} \right\rangle \otimes \left\lvert \square_{\widehat{\square}} \right\rangle$	$\lvert 0_{\hat{1}}\rangle \otimes \langle 1_{\hat{2}}\rvert$	$\mathrm{Tr}_{\widehat{\square}}(\square)$	$\mathrm{Tr}_{\hat{1}}(\lvert 0_{\hat{1}}, 1_{\hat{2}}\rangle \langle 0_{\hat{1}}, 0_{\hat{2}}\rvert) = \lvert 1_2\rangle \langle 0_2\rvert$
$\square \cdot \left\lvert \square_{\widehat{\square}} \right\rangle$	$\mathcal{O} \cdot \lvert 1_{\hat{2}}\rangle$	$\lVert \square \rVert$	$\lVert\, \lvert 0_{\hat{1}}\rangle \,\rVert = \sqrt{\langle 0_1 \mid 0_1\rangle}$

Table 1. Some objects included in Quantum'Notation' package and examples of their usage.

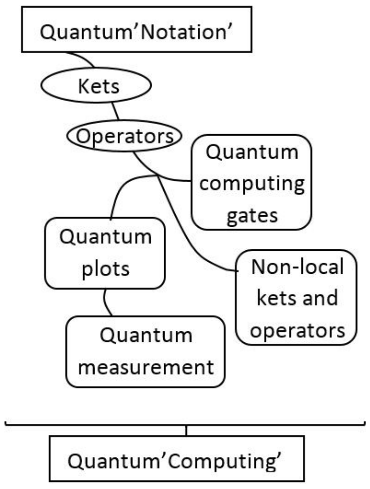


Figure 4. Main sections developed in Quantum'Notation' module integrating the basic aspects in Dirac notation: bra-ket's, operators and operations.

Gates) are included to ease the introduction of the templates for those elements and commands (Figure 5), but still each one of them can be entered with a keyboard shortcut. Table 2 reports a group of representative templates and examples for this module.

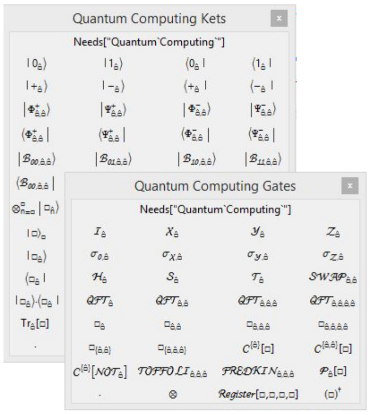
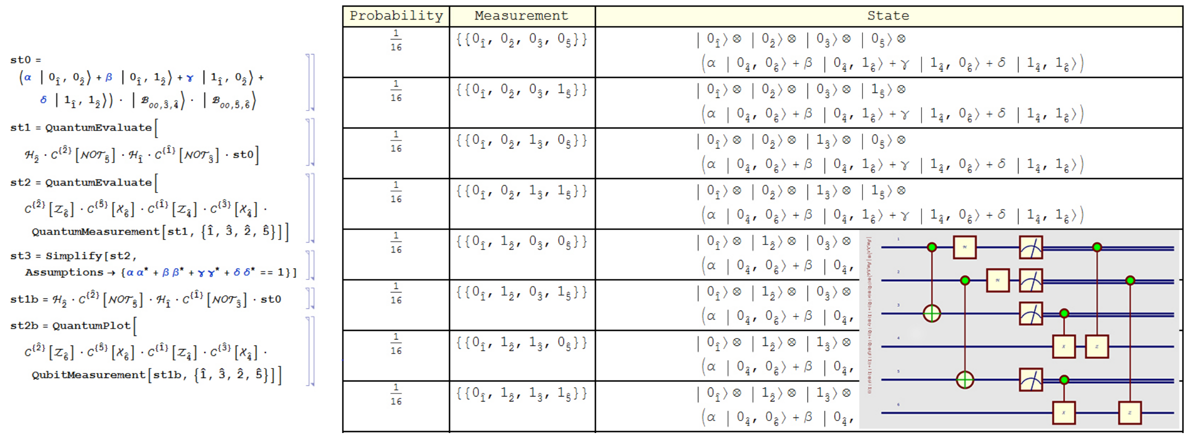


Figure 5. Main Quantum'Computing' palettes showing some commands implemented in the module: computing gates, measurements and plots.

Nevertheless Quantum'Computing' is based on computational basis from Quantum'Notation', the user can hold the calculations to not evolve in the final expressions on that basis, then



```
st0 =
 (α |0_1̂, 0_2̂⟩ + β |0_1̂, 1_2̂⟩ + γ |1_1̂, 0_2̂⟩ +
    δ |1_1̂, 1_2̂⟩) · |𝓑_00,3̂,4̂⟩ · |𝓑_00,5̂,6̂⟩
st1 = QuantumEvaluate[
  𝓗_2̂ · 𝒞^{2̂}[𝒩𝒪𝒯_5̂] · 𝓗_1̂ · 𝒞^{1̂}[𝒩𝒪𝒯_3̂] · st0]
st2 = QuantumEvaluate[
  𝒞^{2̂}[𝒵_6̂] · 𝒞^{5̂}[𝒳_6̂] · 𝒞^{1̂}[𝒵_4̂] · 𝒞^{3̂}[𝒳_4̂] ·
   QuantumMeasurement[st1, {1̂, 3̂, 2̂, 5̂}]]
st3 = Simplify[st2,
  Assumptions → {α α* + β β* + γ γ* + δ δ* == 1}]
st1b = 𝓗_2̂ · 𝒞^{2̂}[𝒩𝒪𝒯_5̂] · 𝓗_1̂ · 𝒞^{1̂}[𝒩𝒪𝒯_3̂] · st0
st2b = QuantumPlot[
  𝒞^{2̂}[𝒵_6̂] · 𝒞^{5̂}[𝒳_6̂] · 𝒞^{1̂}[𝒵_4̂] · 𝒞^{3̂}[𝒳_4̂] ·
   QubitMeasurement[st1b, {1̂, 3̂, 2̂, 5̂}]]
```

Probability	Measurement	State
$\frac{1}{16}$	$\{\{0_{\hat1}, 0_{\hat2}, 0_{\hat3}, 0_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|0_{\hat2}\rangle\otimes\|0_{\hat3}\rangle\otimes\|0_{\hat5}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},1_{\hat6}\rangle+\gamma\,\|1_{\hat4},0_{\hat6}\rangle+\delta\,\|1_{\hat4},1_{\hat6}\rangle)$
$\frac{1}{16}$	$\{\{0_{\hat1}, 0_{\hat2}, 0_{\hat3}, 1_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|0_{\hat2}\rangle\otimes\|0_{\hat3}\rangle\otimes\|1_{\hat5}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},1_{\hat6}\rangle+\gamma\,\|1_{\hat4},0_{\hat6}\rangle+\delta\,\|1_{\hat4},1_{\hat6}\rangle)$
$\frac{1}{16}$	$\{\{0_{\hat1}, 0_{\hat2}, 1_{\hat3}, 0_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|0_{\hat2}\rangle\otimes\|1_{\hat3}\rangle\otimes\|0_{\hat5}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},1_{\hat6}\rangle+\gamma\,\|1_{\hat4},0_{\hat6}\rangle+\delta\,\|1_{\hat4},1_{\hat6}\rangle)$
$\frac{1}{16}$	$\{\{0_{\hat1}, 0_{\hat2}, 1_{\hat3}, 1_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|0_{\hat2}\rangle\otimes\|1_{\hat3}\rangle\otimes\|1_{\hat5}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},1_{\hat6}\rangle+\gamma\,\|1_{\hat4},0_{\hat6}\rangle+\delta\,\|1_{\hat4},1_{\hat6}\rangle)$
$\frac{1}{16}$	$\{\{0_{\hat1}, 1_{\hat2}, 0_{\hat3}, 0_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|1_{\hat2}\rangle\otimes\|0_{\hat3}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},$
$\frac{1}{16}$	$\{\{0_{\hat1}, 1_{\hat2}, 0_{\hat3}, 1_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|1_{\hat2}\rangle\otimes\|0_{\hat3}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},$
$\frac{1}{16}$	$\{\{0_{\hat1}, 1_{\hat2}, 1_{\hat3}, 0_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|1_{\hat2}\rangle\otimes\|1_{\hat3}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},$
$\frac{1}{16}$	$\{\{0_{\hat1}, 1_{\hat2}, 1_{\hat3}, 1_{\hat5}\}\}$	$\|0_{\hat1}\rangle\otimes\|1_{\hat2}\rangle\otimes\|1_{\hat3}\rangle\otimes$ $(\alpha\,\|0_{\hat4},0_{\hat6}\rangle+\beta\,\|0_{\hat4},$

Figure 6. Code and measurement outcomes table in QUANTUM for a Quantum Teleportation algorithm of a 2-qubit state. The right-down inset is the quantum circuit plot obtained with the command QuantumPlot.

being capable to be processed by different commands to produce states, tables or plots. This library includes tensor product templates, norms, states sums for superposition, commutators, anti-commutators and quantum partial transpose as additional elements working together with those in Quantum'Notation'. On those elements, additional operators or gates related with circuit quantum computation are included. They are enriched with commands able to operate on them to generate measurements, probability tables and circuit plots (Figure 6). Table 2 reports a group of representative templates and examples for this module. Expressions after of symbol $\to$ are QUANTUM outputs.

Quantum'Computing' can perform both, simple and complex calculations in Quantum Information Processing or Quantum Computation. As a simple example, Figure 6 shows the code to realize the traditional Quantum Teleportation algorithm [10, 11] for a general 2-qubit state. Process begins defining the state on six qubits (st0) with state to teleport located in qubits 1 and 2. After, a state (st1) is obtained applying the common $\mathcal{H}_i$ and C^iNOT_j gates in the algorithm. Finally, the teleportated state is obtained on the qubits 4 and 6 after of four measurements in the qubits 1, 2, 3 and 5, in the form of a probability table with the command QuantumMeasurement including further corrections using classical communication. Notice once again how almost the same code under alternative commands as QuantumPlot instead of QuantumEvaluate generates alternative outputs considered in the package (as the plot of respective quantum circuit). In similar ways, other more complex algorithms can be developed using the gates included or other defined by the users.

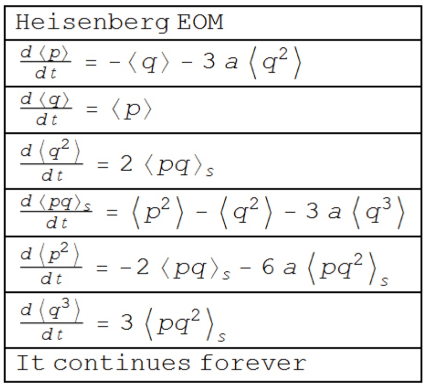
States		Gates	
Template	Example	Template	Example
$\left\|\beta_{ij,\hat\square,\hat\square}\right\rangle$	$\left\|\beta_{10,\hat1,\hat2}\right\rangle = \frac{\|0_1 0_2\rangle - \|1_1 1_2\rangle}{\sqrt2}$	$\sigma_{i,\hat\square}$	$\sigma_{1,\hat2} = \sigma_{x,2} = \mathcal{X}_2$
$\|\pm_{\hat\square}\rangle$	$\|+_{\hat2}\rangle = \frac{\|0_2\rangle + \|1_2\rangle}{\sqrt2}$	$\mathcal{H}_{\hat\square}$	$\mathcal{H}_{\hat3} = \frac{1}{\sqrt2}(\mathcal{X}_2 + \mathcal{Z}_2)$
$\bigotimes_{n=\hat\square}^{\hat\square} \|\square_{\hat n}\rangle$	$\bigotimes_{n=\hat0}^{\hat2} \|n+1_{\hat n}\rangle \to \|1_{\hat0}, 2_{\hat1}, 3_{\hat2}\rangle$	$\mathcal{C}^{\{\hat\square\}}[\mathcal{NOT}_{\hat\square}]$	$\mathcal{C}^{\{\hat1\}}[\mathcal{NOT}_{\hat2}] = C^1NOT_2$
QuantumEvaluate[□]	QuantumEvaluate[$\mathcal{Z}_2$] $\to \|0_{\hat2}\rangle\langle 0_{\hat2}\| - \|1_{\hat2}\rangle\langle 1_{\hat2}\|$	$\mathcal{C}^{\{\hat\square\}}[\square]$	$\mathcal{C}^{\hat1}[\mathcal{H}_{\hat3}] = C^1\mathcal{H}_3$

Table 2. Some objects included in Quantum'Computing' package and examples of their usage.

5. Quantized Hamilton Dynamics Module

The evolution of the expected value of an observable A in the Heisenberg representation is given by the equation of motion (EOM):

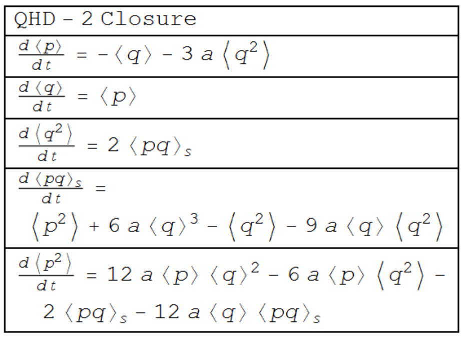
$$i\hbar \frac{d}{dt}\langle A\rangle = \langle [A, H]\rangle \tag{2}$$



Heisenberg EOM
$\frac{d\langle p\rangle}{dt} = -\langle q\rangle - 3a\langle q^2\rangle$
$\frac{d\langle q\rangle}{dt} = \langle p\rangle$
$\frac{d\langle q^2\rangle}{dt} = 2\langle pq\rangle_s$
$\frac{d\langle pq\rangle_s}{dt} = \langle p^2\rangle - \langle q^2\rangle - 3a\langle q^3\rangle$
$\frac{d\langle p^2\rangle}{dt} = -2\langle pq\rangle_s - 6a\langle pq^2\rangle_s$
$\frac{d\langle q^3\rangle}{dt} = 3\langle pq^2\rangle_s$
It continues forever

Figure 7. A particular example of Heisenberg EOM. They are coupled and form an infinite hierarchy of equations.

Consider the averages of momentum, position and their products $\langle p\rangle$, $\langle q\rangle$, $\langle p^2\rangle$, $\langle q^2\rangle$, $\langle pq\rangle$, $\langle p^3\rangle$, $\langle p^2 q\rangle$... The EOMs for their average values are coupled and, in general, form an infinite hierarchy of equations. As an example, the EOMs for the Hamiltonian $H = \frac{1}{2}(p^2 + q^2) + aq^3$ are shown in Figure 7. In order to obtain a finite hierarchy of equations, a closure procedure has to be applied. In the Quantized Hamilton Dynamics approximation [12], the expectation values of higher order expressions are approximated in terms of those other expectation values already included in the hierarchy, for instance, the approximation $\langle ABC\rangle \approx \langle AB\rangle\langle C\rangle + \langle AC\rangle\langle B\rangle + \langle BC\rangle\langle A\rangle - 2\langle A\rangle\langle B\rangle\langle C\rangle$ is one of the approximations used at second order, QHD-2. Figure 8 shows the finite QHD-2 hierarchy obtained in the example.



QHD – 2 Closure
$\frac{d\langle p\rangle}{dt} = -\langle q\rangle - 3a\langle q^2\rangle$
$\frac{d\langle q\rangle}{dt} = \langle p\rangle$
$\frac{d\langle q^2\rangle}{dt} = 2\langle pq\rangle_s$
$\frac{d\langle pq\rangle_s}{dt} = \langle p^2\rangle + 6a\langle q\rangle^3 - \langle q^2\rangle - 9a\langle q\rangle\langle q^2\rangle$
$\frac{d\langle p^2\rangle}{dt} = 12a\langle p\rangle\langle q\rangle^2 - 6a\langle p\rangle\langle q^2\rangle - 2\langle pq\rangle_s - 12a\langle q\rangle\langle pq\rangle_s$

Figure 8. The QHD-2 closure procedure was applied to the EOM of the previous figure, resulting in hierarchy of only five equations.

The third module of QUANTUM has several commands that allow the automatic generation of EOM for arbitrary Hamiltonians, with or without closure (those commands were used to generate Figures 7 and 8). Closures can be applied at orders QHD-1 (which gives classical dynamics for the Hamiltonian), QHD-2 and QHD-3; then the module has commands to include the initial conditions, solve the system of equations, and generate plots of the evolution of the expected values in time, as well as phase plots. The website of QUANTUM [1] includes examples of the application of these commands to several physical systems, including the O-H bond of a water molecule.

6. Conclusions

In this paper we have presented QUANTUM, a Mathematica add-on for calculations in Dirac notation, non-commutative operator algebra, quantum computing and approximations to the Heisenberg equations of motion. It allows the end user to input the calculations directly in the usual notation for Quantum Mechanics, and to obtain results in the same format. This unique characteristic has appealed to students and researchers in more than 20 countries to the best of our knowledge. Actually some of the commands and features of QUANTUM were included by recommendation of those users. Future work is also based on those recommendations, and it includes modules for occupation number notation (Second Quantization and Quantum Field Theory), angular momentum, continuous Hilbert spaces, among others.

Aknowledgements
The support from Tecnológico de Monterrey to develop this research work is acknowledged.

References

[1] Gomez J L 2015 *Website of QUANTUM* (QUANTUM, Tecnológico de Monterrey) Retrieved from: http://homepage.cem.itesm.mx/jose.luis.gomez/quantum/
[2] Nielsen M A and Chuang I L 2000 *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge)
[3] Barnett S M 2009 *Quantum Information* (Oxford University Press, New York)
[4] Jaeger G 2007 *Quantum Information: an overview* (Sringer, New York)
[5] GitHub 2015 *Website of Julia Libraries for Quantum Science and Technology* (Julia, GitHub Inc.) Retrieved from: https://github.com/JuliaQuantum
[6] Wolfram 2015 *QuCalc: A Quantum Computation Package* (QuCalc, Universite de Montreal) Retrieved from: http://library.wolfram.com/infocenter/MathSource/657/
[7] Maplesoft 2015 *Website of Physics package* (Maplesoft, Waterloo Maple Inc.) Retrieved from: http://www.maplesoft.com/support/help/maple/view.aspx?path=Physics
[8] Díaz S 2010 *Diseño de y simulación de Hamiltonianos para Cómputo Cuántico Adiabático* (Master Thesis, Tecnológico de Monterrey, México)
[9] Wolfram 2015 *Wolfram Language and System: documentation center* Retrieved from: http://reference.wolfram.com/language/
[10] Bennett C H, Brassard G, Crépeau C, Jozsa R, Peres A, Wootters W K 1993 *Phys. Rev. Lett.* **70** 1895
[11] Delgado F 2015 *J. Phys.: Conf. Series* **624** 012006
[12] Prezhdo O V and Pereverzev Y V 2000 *J. Chem. Phys.* **113** (16) 6557