# The symbolic simulation and manipulation of quantum circuits

Addison Cugini
*Apple Computer, Cupertino, CA*

Matt Curry
*Department of Physics, University of New Mexico, Albuquerque, NM 87131, USA*

Raymond Wong
*Department of Computer Science, California Polytechnic State University, San Luis Obispo, CA 93407, USA*

Austen Greene and Brian E. Granger*
*Department of Physics, California Polytechnic State University, San Luis Obispo, CA 93407, USA*
(Dated: December 6, 2011)

The simulation of quantum computers and quantum information systems on classical computers is an important part of quantum information science. With such simulations, algorithms can be developed and tested, experiments can be validated and insight can be gained about the foundations of quantum information. We have developed an open source software package for simulating quantum computers symbolically using a computer algebra system, SymPy. The symbolic manipulation of gates and circuits has many advantages over the traditional numerical approach where gates and qubits are represented as large matrices and vectors. In this work, we introduce the software, describe its features through examples and outline the advantages of this approach.

* bgranger@calpoly.edu

## I. INTRODUCTION

## II. SYMPY

SymPy is a Python library for symbolic mathematics. It aims to become a full-featured computer algebra system (CAS) while keeping the code as simple as possible in order to be comprehensible and easily extensible. SymPy is written entirely in Python and does not require any external libraries.

SymPy was started in 2005 by Ondřej Čertík and it now has an active community of over 100 developers contributing to it. SymPy is BSD licensed and anyone can easily contribute new features and algorithms by submitting a GitHub pull request. SymPy has been periodically participating in the Google Summer of Code (25 students in 5 years) and the quantum computing module has started as one such project as well.

```
from __future__ import division
from sympy import *
x, y, z, t = symbols('x y z t')
k, m, n = symbols('k m n', integer=True)
f, g, h = symbols('f g h', cls=Function)
```

```
((x+y)**2 * (x+1)).expand()
```

$$x^3 + 2x^2y + x^2 + xy^2 + 2xy + y^2$$

```
solve(Eq(x**3 + 2*x**2 + 4*x + 8, 0), x)
[-2*I, 2*I, -2]
```

```
limit((sin(x)-x)/x**3, x, 0)
```

$$-\frac{1}{6}$$

```
(1/cos(x)).series(x, 0, 6)
```

$$1 + \frac{1}{2}x^2 + \frac{5}{24}x^4 + \mathcal{O}\left(x^6\right)$$

## III. SYMBOLIC DIRAC NOTATION

## IV. QUBITS

## V. GATES

## VI. DENSITY MATRIX

## VII. ALGORITHMS AND EXAMPLES

### A. Quantum Fourier transform

### B. Grover's Algorithm

### C. Teleportation

### D. Dense coding

### E. Heisenberg limited measurement

### F. Quantum error correction

---

[1] L. K. Grover, Am. J. Phys **69**, 769 (2001).