# Composite Trotter-QDrift Simulator with Optimized Partitioning

**Dhruva Chayapathy**
Utterwqlnut
Alpharetta High School
`dhruva.chayapathy@gmail.com`

February 22, 2024

## Abstract

The simulation of quantum systems is an extremely important task with many applications in fields like chemistry, optimization, and condensed matter. It is believed quantum computers could excel at this task. Algorithms like the Trotter methods and QDrift algorithm are often used to construct the the complex exponential of the Hamiltonian which can express how the quantum system evolves over time. However, these still require a large amount of gates especially for simulating longer periods of time. And so in this project we look at a composite Trotter-QDrift algorithm which uses both a Trotter channel and QDrift channel, harnessing the benefits of both methods. We find that our method uses much less gates and is more hardware-efficient which is especially important in our NISQ and upcoming ISQ era.

## 1 Introduction

Our project looks to develop a better more hardware-efficient approach to creating the complex exponential of the Hamiltonian, $U(t) = e^{iHt}$. This operator calculates how a quantum system evolves over time and is extremely useful in quantum simulations with applications in fields like chemistry, optimization, and condensed matter. Typical methods to do this are the deterministic Trotter methods and the stochastic QDrift algorithm. Both these methods however have downsides, with the number of gates required for trotter methods being exponential to the number of terms $L$ in the Hamiltonian, and the number of gates required for QDrift largely depending on the spectral weight $\lambda$ of the Hamiltonian. We develop a composite Trotter-Qdrift algorithm which captures the best of these two algorithms through a Hamiltonian partitioning scheme. The Hamiltonian is split into two parts for a Trotter Channel and QDrift Channel where the result is later recombined. We use a genetic algorithm to search for a near-optimal partitioning allowing us to reap the benefits of both algorithms, significantly outperforming both QDrift and Trotter methods.

## 2 Methods

### 2.1 Composite Channels

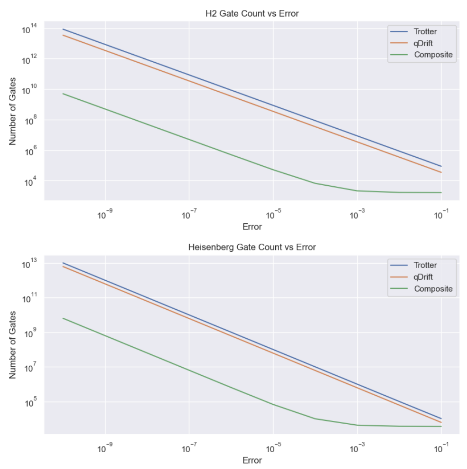
As mentioned before the composite approach uses two channels, the deterministic trotter channel and stochastic qdrift channel. We partition hamiltonian $H$ into two parts $H = A + B$ with

$$A = \sum_i a_i A_i, B = \sum_i b_i B_i$$

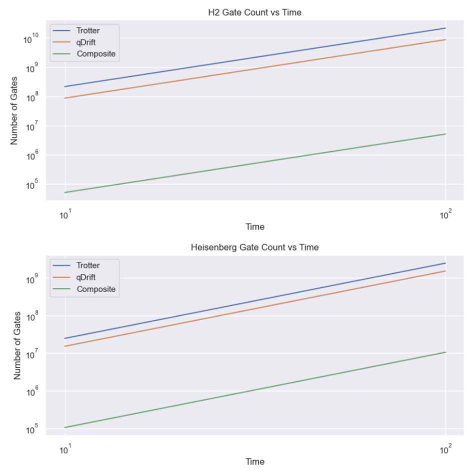
where A is the hamiltonian for the trotter channel $U_A(t)$ and B is the hamiltonian for the QDrift channel $U_B(t; N, M)$. We can define the average channel as

$$U_H(t) = U_A(t) \circ U_B(t; N, M)$$

This average channel is our approximation for the complex exponential of hamiltonian $H$.



(a) Gate Count vs Error



(b) Gate Count vs Time

Figure 1: (a) Scaling of number of gates vs time for Trotter, QDrift, and our Composite method. Time is kept constant at 2 seconds (b) Scaling of number of gates vs time for Trotter, QDrift, and our Composite method. Systemetic error is kept constant at $10^{-3}$

### 2.2 Hamiltonian Partitioning

To partition the hamiltonian $H$ we look to take advantage of each of the channels strengths and weaknesses. To do this we construct an overall cost function: the upper bound on the number of gates required. First look at the error bounds for a deterministic first order trotter and stochastic QDrift.

$$\epsilon \leq \frac{L^2\lambda^2 t^2}{2r} e^{L\lambda t/r}$$

$$\epsilon \leq \frac{2\lambda^2 t^2}{N}$$

The first bound is for the Trotter method and second is for the QDrift algorithm. With this in mind it is important minimize the length of hamiltonian $A$ and minimize the spectral norm $\lambda$ of hamiltonian B. Now lets construct our cost function or the upper bound on the number of gates required for our algorithm. Which is similar to the one obtained in `https://arxiv.org/pdf/2212.05952.pdf`, however slightly different because we use a deterministic partitioning scheme.

$$C_{cost} \leq (L_A + N_B)\frac{t^2}{\epsilon}\left(\sum_{i<j} a_i a_j |||[A_i, A_j]||| + \frac{1}{2}\sum_{ij} a_i b_j |||[A_i, B_j]||| + \frac{2\lambda^2}{N_B}\right)$$

Because the number of ways to partition the hamiltonian grows exponentially $2^L$, We use a genetic algorithm, a heuristic search based off of natural selection, to search for a near-optimal partitioning of the hamiltonian $H$ . Each individual in the algorithm represents a specific partitioning scheme. We chose the cost function above as the fitness score for each individual to evaluate it's partitioning scheme. Each individual has a bit string chromosome eg. 1010 which is defines how the hamiltonian is split into a Trotter or QDrift channels (1 for Trotter 0 for QDrift).

## 3 Results

Now lets look at how our algorithm performed compared to typical Trotter and QDrift methods. We tested these algorithms on an H2 molecular model and Heisenberg model. We first analyzed how the number of gates scaled against error shown in in Figure 1. For both models we see that the Trotter method required the most gates, and the composite method required the least gates over all errors. Our composite model shows superior gate scaling compared to both QDrift and Trotter. We then analyzed how the number of gates scaled against time also shown in Figure 1. We again see that the Trotter method required the most gates, and the composite method required the least gates over all times. For both tasks in each model our composite method significantly outperforms the deterministic Trotter and stochastic QDrift.

Trotter and QDrift each have their own advantages, but also their own disadvantages. Our composite method allows us to split the hamiltonian up favourably for both channels, significantly improving performance and hardware-efficiency. This method can make quantum simulations, or other algorithms like QPE more practical on NISQ and upcoming ISQ computers.

## 4 Overview

Heres an overview of everything I did in this project.

- Implemented a composite Trotter-QDrift algorithm
- Developed a novel genetic algorithm to search for optimal partitioning schemes
- Tested our method against Trotter and QDrift over multiple hamiltonians and tasks
- Found that on all tasks our method scales significantly better than Trotter and QDrift methods