Simulation-Based Engineering Lab
University of Wisconsin-Madison
Technical Report TR–2018–03

Quantum Annealing for Mobility Studies: Generation of GO/NO-GO Maps via Quantum-Assisted Machine Learning

R. Serban
Department of Mechanical Engineering, University of Wisconsin – Madison

M. Wilson, M. Benedetti, J. Realpe-Gomez, A. Perdomo-Ortiz
NASA Ames Research Center

P. Jayakumar
U.S. Army RDECOM-TARDEC

July 1, 2018

Contents

1 Introduction

1.1 Project Objective

The goal of this project was to explore and provide a proof-of-concept approach to solving ground vehicle mobility-related problems on emerging quantum computing (QC) machines, in particular as embodied in the D-Wave quantum annealer systems. We identified the problem of generating GO/NO-GO-like maps as a suitable target problem, which can be mapped into a problem amenable to QC, taking into consideration current hardware limitations. The GO/NO-GO problem is first cast as a machine learning problem and subsequently solved using quantum annealing, while relying on classical high-performance computing simulations for the generation of the required training set.

The specific aims of this exploratory project were as follows:

(i) Cast the problem of generating GO/NO-GO maps as a machine learning problem;

(ii) Identify appropriate problem variables (including input data and targets) and find a way to handle them on a quantum annealing device;

(iii) Develop a quantum-assisted learning algorithm to train complex generative models capable of answering relevant mobility related questions;

(iv) Devise a strategy for generating the training and validation data sets, using simulation on classical computers with problem representations of appropriate fidelity;

(v) Assess the performance of quantum-assisted machine learning in the context of the GO/NO-GO problem;

(vi) Identify which type of mobility problems can benefit the most from quantum-assisted machine learning.

1.2 Project Outcome

A main result of this highly interdisciplinary project was the identification of some of the constraints that come from integrating the different fields. Mobility simulations are computationally expensive and generate high-dimensional results. This means that a dataset of mobility simulations will contain (i) a relatively small number of simulations, and (ii) a large amount of information for each of the simulations. This regime is the exact opposite of the desired one for two reasons. First, some of the most successful machine learning algorithms are known to perform well when large amount of data is available, in contrast to (i). Second, currently available quantum computers have a small number of qubits and therefore can handle a limited number of variables, in contrast to (ii).

Among the numerous quantum machine learning proposals, we chose an algorithm that can in principle deal with some of the above constraints and, at the same time, be implemented on existing quantum hardware. In particular, the algorithm was designed for quantum annealing hardware. A simplified classical version was used for the preliminary study and the quantum version was used for the final results. Although the quantum algorithm did not excel, it did not perform worse than the classical version. This is promising in the sense that the algorithm was able to cope with noise and errors from the quantum annealer. As more general gate-model quantum computers become available and quantum algorithms evolve, we expect much better outcomes.

The machine learning model used here has a large number of hyperparameters: number of layers, number of units, learning rate, regularization factor, number of samples, and data encoding, to name a few. A thorough exploration of these would require a large number of executions and was beyond the scope and timeline of this work. Although we successfully proposed and implemented a D-wave-assisted alternative for the GO/NO-GO problem, more work is needed to determine whether there could be any advantage in this approach or if it would be better to consider quantum machine learning implementations of other alternative solutions to this problem; e.g., via Bayesian optimization. However, it is likely that implementation of these alternative machine learning techniques will require quantum hardware resources beyond the state-of-the-art currently available devices.

2 Technical Approach

2.1 Ground Vehicle Mobility and GO/NO-GO Maps

While there is no general agreement on the exact definition of ground vehicle mobility, the consensus is that the maximal *speed-made-good* is a good measure for mobility [1]. Herein, speed-made-good is defined as the ratio of the straight-line distance between two points and the time required to travel between them, regardless of the actual path. This measure of mobility depends in a highly-nonlinear fashion on many parameters, among which we mention: terrain topology, profile, and roughness; soil type and properties; meteorological conditions; and, finally, vehicle type, configuration, and characteristics.

Under the definition adopted here, a GO/NO-GO map can thus be obtained by color-coding a particular geographical area as function of the corresponding mobility measure (either a continuous variable such as the speed-made-good or a binary output indicating whether or not the destination point can be reached). Since terrain variability and heterogeneity can be very large even for relatively confined geographical regions, a pragmatic approach consists of discretizing the area in a grid, not necessarily uniform, such that input parameters (e.g., terrain characteristics and soil properties) can be considered constant within a grid cell. Then, the GO/NO-GO map is obtained by estimating a mobility measure for each cell and appropriately color-coding the grid.

2.2 Mobility Maps through Quantum-Assisted Machine Learning

The machine learning aspect. Generation of such a GO/NO-GO map can be based on empirical data or else on simulation results of varying fidelity. However, the former will lack accuracy and predictive power. On the other hand, depending on the level of fidelity adopted for modeling soil and/or vehicles, a simulation-based approach can be prohibitively intensive in terms of computational cost, rendering this approach infeasible for operational purposes. Instead, generation of GO/NO-GO maps can be framed as either a classification problem; i.e., the problem of deciding to which label – GO or NO-GO– a given set of observations (input data; e.g., terrain profile, soil properties, vehicle type) belongs, or even more generally, as an unsupervised learning problem, where the goal is to infer/learn correlations among all variables involved, including both input data and labels.

The quantum computing aspect. Quantum computers are designed to manipulate vectors and tensor products in high-dimensional spaces. As a result, quantum machine learning can potentially

provide exponential speed-ups for a wide range of learning tasks and algorithms [2]. Much of the record-breaking performance of classical machine-learning algorithms regularly reported in the literature pertains to task-specific supervised learning algorithms. On the other hand, unsupervised learning algorithms are more human-like, in principle more general and powerful, but their development has been lagging behind due to the intractability of traditional sampling techniques (e.g. Markov Chain Monte Carlo). Quantum annealing holds the potential to sample more efficiently and from more complex probabilistic models, which can significantly advance the field of unsupervised learning.

The hybrid computing aspect. The overall methodology envisioned and adopted for exploration in this project was therefore a hybrid approach in which quantum computing will be used to train a machine learning model using classical computing-based simulation to generate the training and test data.

2.3 Problem Characteristics

We list here some of the characteristics of the GO/NO-GO problem as defined above that modulated and informed the decisions taken in the course of this investigation:

- Depending on the observations that we choose to include in the machine learning process, the input space dimensionality can be very large.

  While QC can in principle address this *curse of dimensionality*, current hardware capabilities (2048 qubits in the D-Wave 2000Q at NASA Ames), are limited to somewhere around 50 binary problem variables for highly connected models and a few hundred variables for sparsely connected ones. This must account for any binary input data, as well as any digitized continuous variables. Moreover, a large dimensional input space requires a large training set, the generation of which can be itself costly if, for example, fully-resolved granular dynamics is to be used for modeling deformable soil.

- Input data (observations) can be very heterogeneous and span multiple scales.

  Some input variables are discrete by nature (e.g., vehicle type), while others are continuous (e.g., soil modulus of elasticity) and can be very differently scaled.

- The vehicle-ground interaction is highly-nonlinear.

  As detailed below, such complex correlations between input variables require highly connected graphs.

Taking the above under consideration, we limited the mobility problem to involve 5 independent terrain and soil parameters and the mobility metric to represent the maximum achievable speed in a straight-line acceleration maneuver, starting from rest on an incline plane. Besides the longitudinal terrain slope, the other 4 soil parameters considered here (also informed by the particular deformable terrain model employed; see Section 4.2.1) were: particle radius, particle material density, inter-particle cohesion, and inter-particle coefficient of friction. This results in training and test datasets with 5 input parameters and one (continuous) output parameter.

3 Quantum-Assisted Machine Learning

The mobility study comes with a dataset made of continuous variables, time series and other types. To model such complex data it is reasonable to includes latent variables, as commonly done in deep learning. To further improve the model, we aimed at including quantum resources such as quantum bits and quantum interactions. The Quantum-Assisted Helmholtz Machine (QAHM) [3] is a concrete proposal that can handle such a scenario on available quantum computers.

3.1 Model

A generative model is based on a probability distribution $P(\mathbf{v}) = \sum_{\mathbf{u}} P(\mathbf{v}|\mathbf{u})P(\mathbf{u})$, where $\mathbf{v} = \{v_1, \ldots, v_N\}$ are visible variables encoding the data and $\mathbf{u} = \{u_1, \ldots, u_M\}$ are unobserved or hidden variables that serve to capture non-trivial correlations by encoding high-level features. To perform inference and learning on this model, we have to sample from the posterior distribution $P(\mathbf{u}|\mathbf{v})$, which is intractable in general. A standard approach to this problem consists of introducing a distribution $Q(\mathbf{u}|\mathbf{v})$ to approximate the true posterior. When choosing the family of such distributions, one should consider functional forms that are both expressive and tractable. The learning algorithm is then in charge of adjusting $P(\mathbf{v})$ to model the data, and adjusting $Q(\mathbf{u}|\mathbf{v})$ to approximate $P(\mathbf{u}|\mathbf{v})$. The Quantum-Assisted Helmholtz Machine is an architecture of this kind that can also work in synergy with quantum devices.

More formally, consider a dataset $\mathcal{S} = \{\mathbf{v}^1, \ldots, \mathbf{v}^d\}$ with empirical distribution $Q_{\mathcal{S}}(\mathbf{v})$. We seek a generative model $P(\mathbf{v}) = \sum_{\mathbf{u}} P(\mathbf{u}, \mathbf{v})$, where $P(\mathbf{u}, \mathbf{v}) = P(\mathbf{v}|\mathbf{u})P_{QC}(\mathbf{u})$. The prior distribution $P_{QC}(\mathbf{u}) = \langle \mathbf{u}|\rho|\mathbf{u}\rangle$ describes samples obtained from a quantum device, and we assume it can be described by a quantum Gibbs distribution $\rho = e^{-\beta\mathcal{H}}/\mathcal{Z}$, where $\mathcal{H}$ is the Hamiltonian implemented in quantum hardware and $\mathcal{Z}$ is the partition function. In the particular case of quantum annealing hardware, we have

$$\mathcal{H} = \sum_{i<j} J_{ij}\hat{Z}_i\hat{Z}_j + \sum_i h_i\hat{Z}_i + \Gamma\sum_i \hat{X}_i, \tag{1}$$

where $\hat{Z}_i$ and $\hat{X}_i$ denote Pauli matrices in the z and x direction, respectively, while J_{ij}, h_i, and Γ are controllable parameters. The conditional distribution $P(\mathbf{v}|\mathbf{u})$ stochastically translates samples from the quantum computer into samples on the domain of the data. Because $\mathbf{v}$ is sampled indirectly from hardware, it can be any type of data. In our case, it corresponds to a vector of continuous variables.

In practice, all the probabilities involved in this architecture can be thought of as artificial neural networks. From now on, we call probability distribution P the *generator network*, and probability distribution Q the *recognition network*.

3.2 Learning

Ideally, an unsupervised learning algorithm for generative models would maximize the average log-likelihood of the data

$$\mathcal{L} = \sum_{\mathbf{v}} Q_{\mathcal{S}}(\mathbf{v}) \ln P(\mathbf{v}). \tag{2}$$

However, the training of a Helmholtz machine is based on the lower bound

$$\sum_{\mathbf{v}} Q_{\mathcal{S}}(\mathbf{v}) \ln P(\mathbf{v}) \geq \sum_{\mathbf{v},\mathbf{u}} Q_{\mathcal{S}}(\mathbf{v}) Q(\mathbf{u}|\mathbf{v}) \ln \frac{P(\mathbf{u},\mathbf{v})}{Q(\mathbf{u}|\mathbf{v})}, \tag{3}$$

where $Q(\mathbf{u}|\mathbf{v})$ is an auxiliary recognition network that approximates the intractable true posterior $P(\mathbf{u}|\mathbf{v})$. Our hybrid architecture uses a classical neural network for Q, sidestepping the need to sample from a quantum device for each data point and at each iteration of learning.

Recall that $P_{QC}(\mathbf{u})$ is a quantum Gibbs distribution. The term $\ln\langle\mathbf{u}|\rho|\mathbf{u}\rangle$ arising from $\ln P(\mathbf{u},\mathbf{v})$ in Eq. (3) is intractable due to the projection of the Gibbs distribution on the states $|\mathbf{u}\rangle$. We can bound this term further using Jensen's inequality and obtain

$$\ln\langle\mathbf{u}|\rho|\mathbf{u}\rangle \geq \langle\mathbf{u}|\ln\rho|\mathbf{u}.\rangle \tag{4}$$

Combining Eqs. (3) and (4), we get a tractable lower bound to maximize, namely the function

$$\mathcal{G}(\theta_G, \theta_{QC}) = \sum_{\mathbf{v},\mathbf{u}} Q_{\mathcal{S}}(\mathbf{v}) Q(\mathbf{u}|\mathbf{v}) \left[\ln P(\mathbf{v}|\mathbf{u}) + \langle\mathbf{u}|\ln\rho|\mathbf{u}\rangle\right], \tag{5}$$

where θ_G and θ_{QC} denote the parameters of generator network $P(\mathbf{v}|\mathbf{u})$ and quantum state ρ, respectively. In Eq. (5) we neglected terms that do *not* depend on either θ_G or θ_{QC}, as they vanish when computing the gradient of $\mathcal{G}$.

For a successful inference, the recognition network $Q(\mathbf{u}|\mathbf{v})$ has to closely track the true posterior during learning. It is easy to see that the bound in Eq. (3) is tight for $Q(\mathbf{u}|\mathbf{v}) = P(\mathbf{u}|\mathbf{v})$. Unfortunately, the maximization of the lower bound in Eq. (3) with respect to the parameters of the recognition network is often intractable. The wake-sleep algorithm [4] attempts to bring $Q(\mathbf{u}|\mathbf{v})$ closer to the true posterior $P(\mathbf{u}|\mathbf{v})$ by minimizing a more tractable notion of distance. Such distance is the Kullback-Leibler divergence

$$D_{KL}\left[P(\mathbf{u}|\mathbf{v})||Q(\mathbf{u}|\mathbf{v})\right] = \sum_{\mathbf{u}} P(\mathbf{u}|\mathbf{v}) \ln \frac{P(\mathbf{u}|\mathbf{v})}{Q(\mathbf{u}|\mathbf{v})}, \tag{6}$$

averaged over the marginal $P(\mathbf{v})$ to take into account the relevance of each configuration $\mathbf{v}$. In other words, wake-sleep maximizes the function

$$\mathcal{R}(\theta_R) = \sum_{\mathbf{u},\mathbf{v}} P(\mathbf{u},\mathbf{v}) \ln Q(\mathbf{u}|\mathbf{v}), \tag{7}$$

where θ_R denotes, collectively, the parameters of the recognition network $Q(\mathbf{u}|\mathbf{v})$. In Eq. (7) we neglected terms that do not depend on θ_R, as they vanish when computing the gradient of $\mathcal{R}$. The gradient ascent equations have structure $\theta^{(t+1)} = (1-\eta\lambda)\theta^{(t)} + \eta\nabla_\theta\mathcal{F}$, where θ stands for the parameters being updated, η is the learning rate, λ is a regularization factor, and $\mathcal{F}$ stands for either $\mathcal{G}$ or $\mathcal{R}$, accordingly. The type of regularization used here is called weight decay and helps in preventing overfitting by keeping the weights small.

Since $\ln\rho = -\beta\mathcal{H} - \ln\mathcal{Z}$, for the parameters of the quantum distribution $\theta_{QC} = (J_{ij}, h_i)$ we have

$$-\frac{1}{\beta}\frac{\partial\mathcal{G}}{\partial J_{ij}} \quad = \quad \langle u_i u_j\rangle_Q - \langle u_i u_j\rangle_\rho, \tag{8}$$

$$-\frac{1}{\beta}\frac{\partial \mathcal{G}}{\partial h_i} = \langle u_i \rangle_Q - \langle u_i \rangle_\rho, \tag{9}$$

where $\langle\,\rangle_Q$ and $\langle\,\rangle_\rho$ denote expectation values with respect to $Q(\mathbf{u}|\mathbf{v})Q_{\mathcal{S}}(\mathbf{v})$ and $P_{QC}(\mathbf{u}) = \langle \mathbf{u}|\rho|\mathbf{u}\rangle$, respectively. Here we have used the property $\hat{Z}_i|u_i\rangle = u_i|u_i\rangle$.

The generation and recognition networks can be written as deep learning architectures

$$P(\mathbf{v}|\mathbf{u}) = \sum_{\mathbf{u}^1,\ldots,\mathbf{u}^L} P_0(\mathbf{v}|\mathbf{u}^1)P_1(\mathbf{u}^1|\mathbf{u}^2)\cdots P_L(\mathbf{u}^L|\mathbf{u}), \tag{10}$$

$$Q(\mathbf{u}|\mathbf{v}) = \sum_{\mathbf{u}^1,\ldots,\mathbf{u}^L} Q_L(\mathbf{u}|\mathbf{u}^L)\cdots Q_1(\mathbf{u}^2|\mathbf{u}^1)Q_0(\mathbf{u}^1|\mathbf{v}), \tag{11}$$

in terms of L additional sets of hidden variables $\mathbf{u}^1,\ldots,\mathbf{u}^L$ that connect the variables $\mathbf{v} \equiv \mathbf{u}^0$ in the visible layer with $\mathbf{u} \equiv \mathbf{u}^{L+1}$ in the last hidden layer. More specifically, when using Bernoulli variables $u_i^\ell \in \{-1,+1\}$, we have

$$P_\ell(\mathbf{u}^\ell|\mathbf{u}^{\ell+1}) = \prod_i \pi(u_i^\ell|\mathbf{u}^{\ell+1}; A^\ell, a^\ell), \tag{12}$$

$$Q_\ell(\mathbf{u}^\ell|\mathbf{u}^{\ell-1}) = \prod_i \pi(u_i^\ell|\mathbf{u}^{\ell-1}; B^\ell, b^\ell), \tag{13}$$

where

$$\pi(u_i|\mathbf{u}'; C, c) = \left[1 + e^{-2u_i\left(\sum_j C_{ij}u'_j + c_i\right)}\right]^{-1}. \tag{14}$$

The gradients for the generative network are

$$\frac{\partial \mathcal{G}}{\partial A_{ij}^\ell} = \langle u_i^\ell u_j^{\ell+1}\rangle_Q - \langle u_i^\ell\rangle_P \langle u_j^{\ell+1}\rangle_Q, \tag{15}$$

$$\frac{\partial \mathcal{G}}{\partial a_i^\ell} = \langle u_i^\ell\rangle_Q - \langle u_i^\ell\rangle_P, \tag{16}$$

and similarly for the recognition network

$$\frac{\partial \mathcal{R}}{\partial B_{ij}^\ell} = \langle u_i^\ell u_j^{\ell-1}\rangle_P - \langle u_i^\ell\rangle_Q \langle u_j^{\ell-1}\rangle_P, \tag{17}$$

$$\frac{\partial \mathcal{R}}{\partial b_i^\ell} = \langle u_i^\ell\rangle_P - \langle u_i^\ell\rangle_Q. \tag{18}$$

3.3 Implementation

We now discuss our implementation of the QAHM for the D-Wave 2000Q quantum annealer. The annealer implements a noisy version of the programmed Hamiltonian in Eq. (1) defined on a sparse graph of qubit interactions. In particular, the device is designed to exploit quantum tunneling to sample low energy states at transverse field $\Gamma \approx 0$. However, non-trivial non-equilibrium effects may make samples deviate from the corresponding classical Gibbs distribution. This scenario requires some engineering of the QAHM framework.

Following the work in Ref. [5], we use a gray-box model for the quantum annealer so that we can update its parameters without the need to estimate deviations from the Gibbs distribution. This approach relies on the assumption that, despite the deviations, the estimated gradients have a positive projection in the direction of the true gradient. Because of a varying unknown inverse temperature β, the learning rate at which parameters are updated varies too. This should not pose a problem as long as we schedule the learning rate to decrease, which is a general condition for convergence of stochastic approximation algorithms of Robbins-Monro type [6].

Now, we would like to implement a fully connected prior distribution $P_{QC}(\mathbf{u})$ over hidden variables in the deepest layer. This connectivity in not available in hardware, so we map each variable to a subgraph of physical qubits. This way, the additional physical interactions between qubits can effectively encode long-range interactions. This expansion needs not be globally optimal, and can be found efficiently using heuristic techniques. The new dynamics are described by the programmed Hamiltonian

$$\widetilde{\mathcal{H}} = -\frac{1}{2}\sum_{i,j=1}^{N}\sum_{k,l=1}^{Q_i,Q_j} J_{ij}^{(kl)}\hat{Z}_i^{(k)}\hat{Z}_j^{(l)} - \sum_{i=1}^{N}\sum_{k=1}^{Q_i} h_i^{(k)}\hat{Z}_i^{(k)}. \tag{19}$$

Here, N is the number of hidden variables in the deepest layer, equal to the number of subgraphs realized in hardware, Q_i is the number of qubits in subgraph i, $\hat{Z}_i^{(k)}$ is the Pauli matrix in the z-direction for qubit k of subgraph i, $h_i^{(k)}$ is the local field for qubit k of subgraph i, and $J_{ij}^{(kl)}$ is the coupling between qubit k of subgraph i and qubit l of subgraph j. Note that the couplings serve to model both the consistency within subgraphs, when $i = j$, and the correlation among subgraphs, when $i \neq j$. A factor of $1/2$ is required to avoid double counting. The gradients required to learn these parameters are similar to those in Eqs. (8) and (9), and can also be found in Ref. [5].

The model is also equipped with two deterministic functions that map samples back and forth between the two spaces (i.e. logical and qubit spaces). We use the following *replica* and *majority vote* mappings

$$z_i^{(k)} = f(\mathbf{u}, i) = u_i \quad (\text{for } k = 1, \ldots, Q_i), \tag{20}$$

$$u_i = g(\mathbf{z}, i) = \text{sign}\left(\sum_{k=1}^{Q_i} z_i^{(k)}\right). \tag{21}$$

We now summarize how the architecture works. The recognition network samples all the hidden variables $\{\mathbf{u}^l\}_{l=1}^{L}$ using an efficient bottom-up pass. The replica mapping converts the deepest layer of hidden variables $\mathbf{u}^L$ into a higher-dimensional representation $\mathbf{z}$. The quantum computer is never used in the recognition network. In the generator network instead, the quantum device is used to sample $\mathbf{z}$ from a Gibbs-like distribution. Samples are mapped back to the hidden variables $\mathbf{u}^L$ using the majority vote over subgraphs. An efficient top-down pass is used to sample the visible variables $\mathbf{v}$.

A final subtlety to take under consideration is the question of how to provide continuous-valued input and obtain continuous-valued output from the aforementioned networks. One way would be to implement Q_0 in Eq. (11) and P_0 in Eq. (10) as Gaussian distributions. This would introduce additional parameters, such as means and standard deviations, to be learned. On the other hand, when a continuous variable lies within a known range, we can simplify things. When using the recognition network, each variable v_i in the dataset is rescaled to lie in $[-1, +1]$ and interpreted as

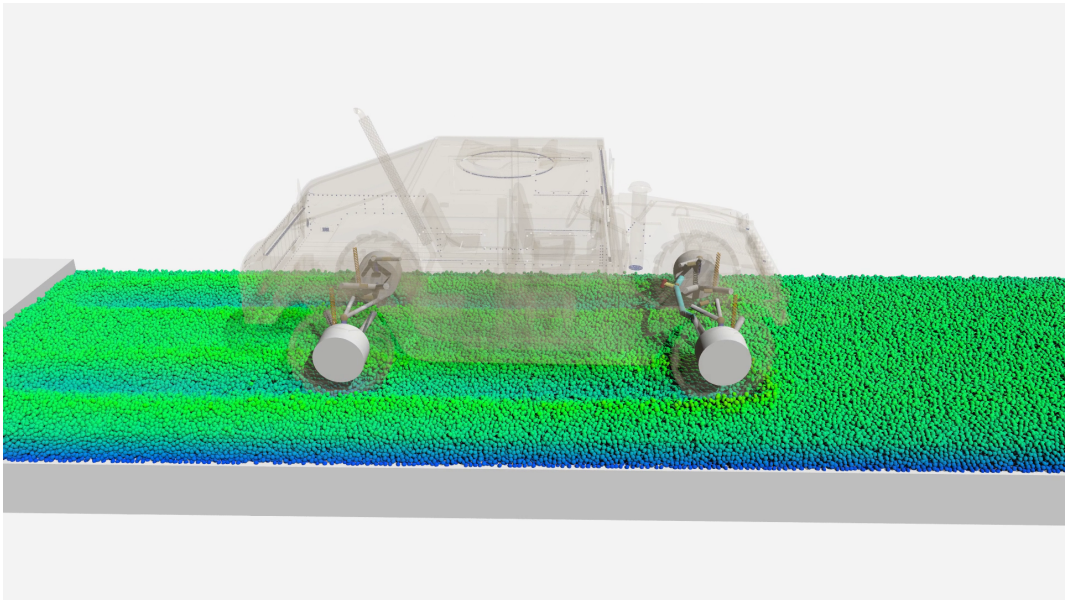


Figure 1: Snapshot from a typical *Chrono* simulation of a vehicle on granular terrain.

the *expected value* of a binary variable in $\{-1, +1\}$. Conversely, the probabilities in output from the generator network are used to compute the expected value of a binary variable, rather than to actually sample the binary variable. Such a value can then be scaled back to the original range of the continuous variable. These techniques allow to use the model without further modifications and without the need of additional parameters.

4 Using Simulation for Mobility Assessment

Generation of the training and test sets were performed using the *Chrono* software suite [7]. *Chrono*'s strength lies in its ability to simulate the dynamics of large multibody systems [8], including discipline-specific support for vehicle modeling and simulation and soil/terrain modeling and simulation. In particular, deformable terrains can be simulated using a fully-resolved Discrete Element Method (DEM) approach, by modeling the soil as a large system of bodies interacting through contact, friction, and cohesion. Alternatively, *Chrono* provides support for more expeditious deformable soil representation, such as the Soil Contact Model (SCM) [9].

The *Chrono*::Vehicle module provides support for template-based modeling and simulation of ground vehicles, both wheeled and tracked, that can be fully and implicitly coupled with the terrain/soil models mentioned above (see Fig. 1). As such, *Chrono* can be used to generate the input data sets required for the learning algorithms considered here: for a given set of parameters, a coupled vehicle-terrain interaction simulation is conducted and results processed to obtain the desired mobility measure (speed-made-good or simply a binary GO/NO-GO decision variable).

4.1 Vehicle Modeling

Chrono::Vehicle [10] is a module of the open-source multi-physics simulation package *Chrono*, aimed

at modeling, simulation, and visualization of wheeled and tracked ground vehicle multi-body systems. Its software architecture and design was dictated by the desire to provide an expeditious and user-friendly mechanism for assembling complex vehicle models, while leveraging the underlying *Chrono* modeling and simulation capabilities, allowing seamless interfacing to other optional *Chrono* modules (e.g., its granular dynamics and fluid-solid interaction capabilities), and providing a modular and expressive API to facilitate its use in third-party applications. Vehicle models are specified as a hierarchy of subsystems, each of which is an instantiation of a predefined subsystem template. Written in C++, *Chrono*::Vehicle is offered as a middleware library.

In this section, we provide an overview of the *Chrono*::Vehicle software design philosophy, its main capabilities and features, and describe the types of ground vehicle mobility simulations it enables.

Chrono::Vehicle provides a comprehensive set of vehicle subsystem templates (tire, suspension, steering mechanism, driveline, sprocket, track shoe, etc.), templates for external systems (powertrain, driver, terrain), and additional utility classes and functions for vehicle visualization, monitoring, and collection of simulation results. As a middleware library, *Chrono*::Vehicle requires the user to provide C++ classes for a concrete instantiation of a particular template. An optional *Chrono* library provides complete sets of such concrete C++ classes for a few ground vehicles, both wheeled and tracked, which can serve as examples for other specific vehicle models. An alternative mechanism for defining concrete instantiation of vehicle system and subsystem templates is based on input specification files in the JSON format. For additional flexibility and to allow integration of third-party software, *Chrono*::Vehicle is designed to permit either monolithic simulations or co-simulation where the vehicle, powertrain, tires, driver, and terrain/soil can be simulated independently and simultaneously.

Chrono::Vehicle currently supports three different classes of tire models: rigid, semi-empirical, and finite element. Rigid tires can be modeled as cylindrical shapes or else as non-deformable triangular meshes. From the second class of tire models, *Chrono*::Vehicle provides templated implementations for Pacejka (89 and 2002), Fiala, Lugre, and TMeasy tire models, all suitable for maneuvers on rigid terrain. Finally, the third class of tire models offered are full finite element representations of the tire. While these models have the potential to be the most accurate due to their detailed physical model of the tire, they are also the most computationally expensive of the tire model currently available in *Chrono*::Vehicle. Using ANCF or Reissner shell elements, these FEA-based tire models can account for simultaneous deformation in tire and soil, for high-fidelity off-road simulations.

Tracked vehicles in *Chrono*::Vehicle are full multibody system models. Templates for both segmented and continuous-band tracks are available, the latter providing options for modeling using 6-DOF bushing elements or else FEA shell elements. Frictional contact interaction, both internal (between vehicle components) and with the terrain, relies on the underlying *Chrono* capabilities and supports both non-smooth (i.e., complementarity-based) and smooth (i.e., penalty-based) contact formulations.

For the studies conducted herein, we used a model of a four-wheel drive off-road vehicle with independent double-wishbone suspension and Pitman arm steering (see Fig. 2).

4.2 Granular Terrain Modeling

Chrono::Vehicle provides several classes of terrain and soil models, of different fidelity and compu-

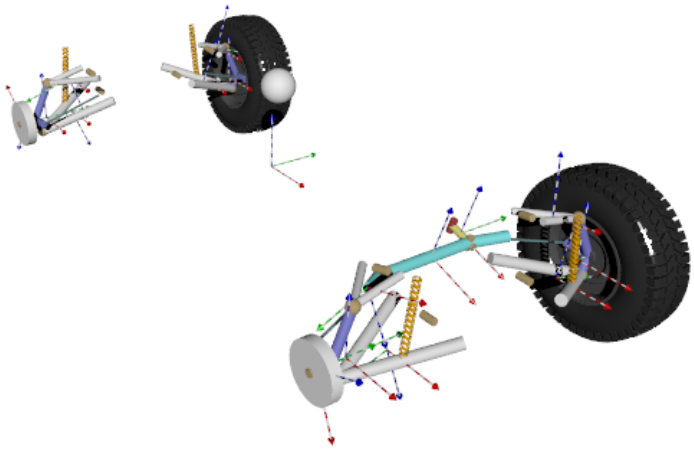


Figure 2: Wheeled vehicle with double wishbone suspensions and Pitman arm steering.

tational complexity, ranging from rigid, to semi-empirical Bekker-Wong type models, to complex physics-based models based on either a granular or finite-element based soil representation. For simple terramechanics simulations, *Chrono*::Vehicle provides a customized implementation of the Soil Contact Model, based on Bekker theory, with extensions to allow non-structured triangular grids and adaptive mesh refinement. Second, *Chrono* provides an FEA continuum soil model based on multiplicative plasticity theory with Drucker-Prager failure criterion and a specialized 9-node brick element which alleviates locking issues with standard brick elements. Finally, leveraging the *Chrono*::Granular module and support for multi-core and distributed parallel computing in *Chrono*, off-road vehicle simulations can be conducted using fully-resolved, granular dynamics-based complex terramechanics, using a Discrete Element Method (DEM) approach. Such simulations can use either of the two methods supported in *Chrono*, namely a penalty-based, compliant-body approach, or a complementarity-based, rigid-body approach.

Two alternative approaches have emerged as viable solutions for large frictional contact problems in granular flow dynamics and quasi-static geomechanics applications, collectively termed herein DEM.

The so-called complementarity method (DEM-C) is generally favored within the multibody dynamics community. In this approach, individual particles in a bulk granular material are modeled as rigid bodies, and non-penetration conditions are written as complementarity equations which, in conjunction with a Coulomb friction law, lead to a Differential Variational Inequality (DVI) form of the Newton-Euler equations of motion. Not limited by stability considerations, DEM-C allows for much larger time integration steps than the alternative penalty-based (DEM-P) solutions, since the latter involve large contact stiffnesses that impose strict stability conditions on all explicit time integration algorithms. However, DEM-C involves a relatively complex and computationally costly solution sequence per time step, since it leads to a mathematical program with complementarity and equality constraints, which must be relaxed to obtain tractable linear complementarity or cone complementarity problems (see below, Section 4.2.1).

More mature and widely adopted within the geomechanics community [11], DEM-P can be

viewed either as a regularization (or smoothing) approach, which relies on a relaxation of the rigid-body assumption, or as a deformable-body approach localized to the points of contact between individual particles in a bulk granular material [12]. In this approach, normal and tangential contact forces are calculated using various laws [13, 14], which are based on the local body deformation at the point of contact. In the contact-normal direction, this local body deformation is defined as the penetration (overlap) of the two quasi-rigid bodies. In the tangential direction, the deformation is defined as the total tangential displacement incurred since the initiation of contact. Once contact forces are known, the time evolution of each body in the system is obtained by integrating the Newton-Euler equations of motion. Since in this approach the contact force-displacement law is derived from the elastic properties of the materials constituting the contacting bodies; i.e., Young's modulus and Poisson's ratio, the DEM-P is capable of resolving statically indeterminate loading conditions that can exist at the particle level in random granular packings. However, due to large contact stiffnesses and the use of explicit time integration [15], the DEM-P approach is limited to very small time integration step-sizes to ensure stability.

A more in depth comparison between the rigid-body (DEM-C) and soft-body (DEM-P) formulations is provided in [16]. Since all granular terrain simulations used herein relied on the DEM-C formulation, we provide in the next section a short overview of the mathematical formulation of this approach, highlighting the quadratic optimization problem with conic constraints that must be solved at each integration time step to obtain contact forces.

4.2.1 Complementarity Formulation for Non-Smooth Frictional Contact Problems

The dynamics of articulated systems composed of rigid and flexible bodies is characterized by a system of index-3 differential algebraic equations [17, 18] routinely solved by commercial packages using techniques that pose the problem using the terms in black font in Eqs. (22a)–(22c):

$$\dot{\mathbf{q}} = \mathbf{L}(\mathbf{q})\mathbf{v} \tag{22a}$$

$$\mathbf{M}(\mathbf{q})\dot{\mathbf{v}} = \mathbf{f}\left(t, \mathbf{q}, \mathbf{v}\right) - \mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}, t)\hat{\lambda} \quad + \sum_{i \in \mathcal{A}(\mathbf{q}, \delta)} \underbrace{\left(\widehat{\gamma}_{i,n}\, \mathbf{D}_{i,n} + \widehat{\gamma}_{i,u}\, \mathbf{D}_{i,u} + \widehat{\gamma}_{i,w}\, \mathbf{D}_{i,w}\right)}_{i^{th}\text{frictional contact force}} \tag{22b}$$

$$\mathbf{0} = \mathbf{g}(\mathbf{q}, t) \tag{22c}$$

$$i \in \mathcal{A}(\mathbf{q}(t)) : \begin{cases} 0 \leq \Phi_i(\mathbf{q}) \;\perp\; \widehat{\gamma}_{i,n} \geq 0 \\ (\widehat{\gamma}_{i,u}, \widehat{\gamma}_{i,w}) = \underset{\sqrt{\bar{\gamma}_{i,u}^2 + \bar{\gamma}_{i,w}^2} \leq \mu_i \widehat{\gamma}_{i,n}}{\operatorname{argmin}} \quad \mathbf{v}^T \cdot \left(\bar{\gamma}_{i,u}\, \mathbf{D}_{i,u} + \bar{\gamma}_{i,w}\, \mathbf{D}_{i,w}\right) \end{cases} . \tag{22d}$$

The differential equations in Eq. (22a) relate the time derivative of the generalized positions $\mathbf{q}$ and velocities $\mathbf{v}$ through a linear transformation defined by $\mathbf{L}(\mathbf{q})$. The force balance in Eq. (22b) ties the inertial forces to the applied and constraint forces, $\mathbf{f}\left(t, \mathbf{q}, \mathbf{v}\right)$ and $-\mathbf{g}_{\mathbf{q}}^{\mathrm{T}}(\mathbf{q}, t)\hat{\lambda}$, respectively. The latter are impressed by bilateral constraints that restrict the relative motion of the rigid or flexible bodies present in the system. These bilateral constraints, which lead to Eq. (22c), can be augmented by unilateral constraints associated with contact/impact phenomena. To that end, the concept of equations of motion is extended to employ differential inclusions [19]. The simplest example is that of a body that interacts with the ground through friction and contact, when the equations of motion become an inclusion $\mathbf{M}\ddot{\mathbf{q}} - \mathbf{f} \in \mathbf{F}(\mathbf{q}, t)$, where $\mathbf{M}$ is the inertia matrix, $\ddot{\mathbf{q}}$ is the body acceleration, $\mathbf{f}$ is the external force, and $\mathbf{F}(\mathbf{q}, t)$ is a set-valued function. The inclusion states that the frictional contact force lies somewhere inside the friction cone, with a value yet to be determined

and controlled by the stick/slip state of the interaction between body and ground. In many-body dynamics, the differential inclusion can be posed as a differential variational inequality problem, which brings along the terms highlighted in red of Eq. (22); see [20–23]. Specifically, the unilateral constraints define a set of contact complementarity conditions $0 \leq \Phi_i(\mathbf{q}) \perp \widehat{\gamma}_{i,n} \geq 0$, which make a simple point: for a potential contact i in the active set, $i \in \mathcal{A}(\mathbf{q}(t))$, either the gap Φ_i between two geometries is zero and consequently the normal contact force $\widehat{\gamma}_{i,n}$ is greater than zero, or vice versa. The last equation poses an optimization problem whose first order Karush-Kuhn-Tucker optimality conditions are equivalent to the Coulomb dry friction model. The frictional contact force associated with contact i leads to a set of generalized forces, shown in red in Eq. (22b), which are obtained using the projectors $\mathbf{D}_{i,n}$, $\mathbf{D}_{i,u}$, and $\mathbf{D}_{i,w}$ [24].

The mixed differential algebraic–differential variational inequality problem in Eq. (22) is discretized to yield the mathematical program with complementarity and equality constraints in Eq. (23). The discretization is based on a symplectic half-implicit Euler method used to advance the state of the system from time $t^{(l)}$ to $t^{(l+1)}$ with a step size Δt. The instantaneous Lagrange multipliers are scaled to become constraint reaction impulses: $\lambda \equiv \Delta t\, \hat{\lambda}$ for bilateral constraints and $\gamma \equiv \Delta t\, \hat{\gamma}_\alpha$, $\alpha \in \{n, u, w\}$, for the normal (n) and two tangential friction (u and w) forces. This discretization yields a first order scheme [25]:

$$\overbrace{\mathbf{q}^{(l+1)}}^{\text{Generalized positions}} = \mathbf{q}^{(l)} + \overbrace{\Delta t}^{\text{Step size}} \underbrace{\mathbf{L}(\mathbf{q}^{(l)})}_{\text{Velocity transformation matrix}} \mathbf{v}^{(l+1)} \tag{23a}$$

$$\mathbf{M}(\overbrace{\mathbf{v}^{(l+1)}}^{\text{Generalized speeds}} - \mathbf{v}^{(l)}) = \underbrace{\Delta t\, \mathbf{f}(t^{(l)}, \mathbf{q}^{(l)}, \mathbf{v}^{(l)})}_{\text{Applied impulse}} - \overbrace{\mathbf{g}_\mathbf{q}^{\mathrm{T}}(\mathbf{q}^{(l)}, t^{(l)})\lambda}^{\text{Reaction impulse}} + \sum_{i \in \mathcal{A}(q^{(l)}, \delta)} \underbrace{(\gamma_{i,n}\, \mathbf{D}_{i,n} + \gamma_{i,u}\, \mathbf{D}_{i,u} + \gamma_{i,w}\, \mathbf{D}_{i,w})}_{\text{Frictional contact reaction impulses}} \tag{23b}$$

$$0 = \underbrace{\frac{1}{\Delta t}\mathbf{g}(\mathbf{q}^{(l)}, t^{(l)})}_{\text{Stabilization term}} + \mathbf{g}_\mathbf{q}^T \mathbf{v}^{(l+1)} + \mathbf{g}_t \tag{23c}$$

$$i \in \mathcal{A}(\mathbf{q}(t), \delta) : \begin{cases} \overbrace{0 \leq \frac{1}{\Delta t}\Phi_i(\mathbf{q}^{(l)})}^{\text{Stabilization term}} + \mathbf{D}_{i,n}^T \mathbf{v}^{(l+1)} \perp \gamma_{i,n} \geq 0 \\ (\gamma_{i,u}, \gamma_{i,w}) = \underset{\sqrt{\bar{\gamma}_{i,u}^2 + \bar{\gamma}_{i,w}^2} \leq \mu_i \gamma_{i,n}}{\operatorname{argmin}} \mathbf{v}^T \cdot (\bar{\gamma}_{i,u}\, \mathbf{D}_{i,u} + \bar{\gamma}_{i,w}\, \mathbf{D}_{i,w}) \ . \end{cases} \tag{23d}$$

A relaxation of the complementarity conditions in Eq. (23d) and further algebraic manipulations [25] yield a cone complementarity problem (CCP) whose solution provides λ and γ. For the N_c contacts in the system, the resulting CCP is posed as follows: find γ_i for $i = 1, \ldots, N_c$ such that $\Upsilon_i \ni \gamma_i \perp (-\mathbf{N}\gamma - \mathbf{r})_i \in \Upsilon_i^\circ$, where $\Upsilon_i = \{[x, y, z]^T \in \mathbb{R}^3 \,|\, \sqrt{y^2 + z^2} \leq \mu_i x\}$ and $\Upsilon_i^\circ = \{[x, y, z]^T \in \mathbb{R}^3 \,|\, x \leq -\mu_i \sqrt{y^2 + z^2}\}$. Here, $\gamma_i = [\gamma_{i,n}, \gamma_{i,u}, \gamma_{i,w}]^T$ represents a triplet of Lagrange multipliers associated with contact i, the first with the normal direction n and two with the friction force components u and w. $\mathbf{D} \equiv [\mathbf{D}_1, \cdots, \mathbf{D}_{N_c}] \in \mathbb{R}^{6n_b \times 3N_c}$ is the generalized contact transformation matrix obtained by splicing the matrices $\mathbf{D}_i \equiv [\mathbf{D}_{i,n}, \mathbf{D}_{i,v}, \mathbf{D}_{i,w}] \in \mathbb{R}^{6n_b \times 3}$, each of which is the contact transformation matrix associated with contact $i \in \{1, \ldots, N_c\}$. $\boldsymbol{N} \equiv \mathbf{D}^T \mathbf{M}^{-1} \mathbf{D} \in \mathbb{R}^{3N_c \times 3N_c}$

is the contact-associated symmetric positive-semidefinite Schur complement matrix, typically very sparse. Finally, $\mathbf{r} \in \mathbb{R}^{3N_c}$ is a constant vector computed once at the beginning of each time step. This CCP represents the first order optimality conditions of a quadratic optimization problem with conic constraints whose solution provides $\gamma = [\gamma_1^T, \ldots, \gamma_{N_c}^T]^T \in \mathbb{R}^{3N_c}$:

$$\begin{aligned} &\min_{\gamma} \frac{1}{2}\gamma^T \mathbf{N}\gamma + \mathbf{r}^T\gamma \\ &\text{subject to } \gamma_i \in \Upsilon_i \text{ for } i = 1, 2, \ldots, N_c \ . \end{aligned} \tag{24}$$

The optimization problem in Eq. (24) is set up and solved in *Chrono* to produce the frictional contact forces [26], which are subsequently used in Eq. (23b) to obtain the new velocities $\mathbf{v}^{(l+1)}$, which are then used in Eq. (23a) to update the generalized positions $\mathbf{q}^{(l+1)}$ and thus conclude the calculation of the new state at $t^{(l+1)}$. The key attribute of a many-body dynamics problem is a small number of bilateral constraints in Eq. (22c), in the tens or maybe hundreds, but a very large number of frictional contact events, which leads to millions of subproblems of the form in Eq. (22d).

4.3 Computational Infrastructure and Simulation Setup

All simulations were conducted on a Cray XC30 system with 12-core Intel® Xeon® E5-2697 v2 processors and a dedicated Cray Aries high-speed network. Independent runs were launched in batches corresponding to the subsets of training points requested by the sequential approach described previously. Each separate run (corresponding to a particular set of the five independent design parameters considered, namely longitudinal terrain slope, particle radius, particle density, inter-particle coefficient of friction, and cohesion pressure) was queued on a single node and used 24 OpenMP threads.

To accelerate time to solution, simulations conducted for this study employed the moving patch feature provided within *Chrono*::Vehicle. In this approach, granular material is simulated only within a sliding window (of user-defined dimensions) centered around and moving with the vehicle. The number of granular material particles remain constant (and relatively low). Particles falling outside the moving patch behind the vehicle are reused by relocating them in front of the vehicle in chunks of spatial dimensions controlled by the user. With this approach, depending on the current particle size, the simulations used herein employed between 48,000 and 480,000 particles.

Each simulation run represented a straight-line acceleration maneuver on longitudinally inclined terrain. The granular material was considered homogeneous and consisting of identical spheres. Particles were initialized in layers (with a number of layers determined dynamically, function of the particle radius) and using a uniform random distribution within each layer obtained with a Poisson disk sampling technique. Following a short granular material settling phase, the vehicle is created above the terrain and allowed to settle. Subsequently, throttle is increased from 0% to 100% over the span of 0.5 s, while the gravitational acceleration vector is rotated by the appropriate angle to model the incline plane. To maintain a straight-line, a path-follower steering controller is used which makes minute adjustments to the vehicle steering input. Several heuristics are implemented to decide completion of the simulated maneuver; tracking running averages of the vehicle forward velocity and acceleration, simulations are stopped when a steady-state maximum velocity is achieved or when the case of the vehicle sliding backward is identified.

During simulation, we record relevant vehicle states for each individual run. In a post-processing stage, information from these output files is collated to generate the incremental training set for

the sequential QML algorithm, as well as statistics for estimating computational performance.

Computational efficiency of *Chrono*::Vehicle simulations is difficult to quantify, as it depends on the complexity of the vehicle, tire, and soil models, on the contact formulation employed, on accuracy levels, and on available hardware. Typical mobility simulations on hard surface, such as double lane change, can be performed in close to real-time for wheeled vehicles (depending on choice of tire model) or in a few minutes for tracked vehicles. Off-road simulation execution times can range from a few hours (for example a tracked vehicle on SCM soil) to several days (in the case of a wheeled vehicle with FEA-based flexible tires on million-body granular terrain).

While not directly relevant to the object of this project, the large number of required simulations provided a good opportunity to evaluate robustness and performance of *Chrono* vehicle simulations on granular terrain (using the DEM-C formulation). More than 500 individual runs were simulated, with different values of the five parameters defined above. It is worth to note that all these simulations were completed successfully and robustly by the *Chrono* software. Additional information pertaining to computational performance is provided in Appendix A.

5 Algorithm Investigations

The purpose of this section is to investigate the performance of the algorithm for learning the distribution of the data, demonstrate the characteristics of the model and the data, and to explore methods for reading predictions from the model. Figure 3 provides a high-level view of the interaction between the concepts outlined in Sections 3 and 4.

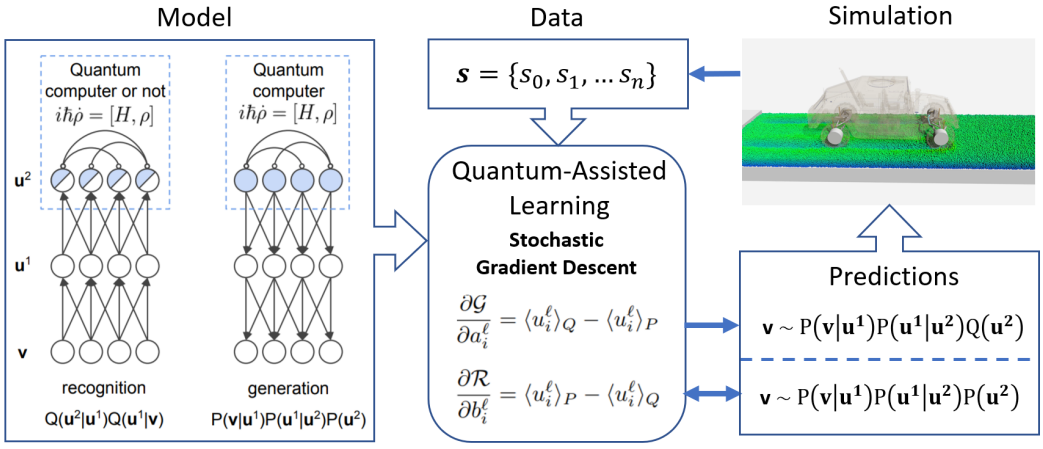


Figure 3: Pipeline for producing simulation data, building a model and learning the distribution of the data and using the generative model to inform future simulations and the quantum assisted learning.

We describe here the implementation of the QAHM on datasets provided through simulations and examine the various features of both the dataset and the model. We highlight and discuss two salient features of the model and draw a strong link between these features and the model effectiveness, using as a performance metric the mean squared error between the predictions of the model and an evaluation set. A natural metric to measure the difference between the learned distribution and the distribution of the data is the log-likelihood in Eq. (2). However, calculating the

distribution of the model $P(\mathbf{v})$ for an arbitrary data point $\mathbf{v}$ is intractable. The bound introduced in Eq. (3) requires estimation of $P(\mathbf{u})$ from samples from the annealer, which is also not feasible. Hence, we select the mean squared error as a measure of the performance of the model predictions against the test and training sets.

Primarily, the classical algorithm is used as a proxy to testing the quantum algorithm for two reasons: (i) the sampling subroutine is entirely classical in both the quantum and classical instances of the algorithm; and (ii) similar features in the predictions are present in both the quantum and classical model predictions so that conclusions drawn about the classical algorithm are also applicable to the quantum algorithm for all the cases discussed here .

The training and test sets contain 320 and 40 points, respectively, sampled from the 6-D parameter space (5 inputs and 1 output). The majority of the networks used for these demonstrations consist of 2 hidden layers comprising 12 nodes each, and a visible 6-node layer. Several experiments were needed in order to choose the hyperparameters. A learning rate $\eta = 0.01$ and a regularization factor $\lambda = 0.0001$ were found to produce best results. The ranges used for the normalization of the input parameters were slope = [0.0, 20.0], radius = [8.0, 18.0], density = [1000.0, 2000.0], friction = [0.6, 1.0], cohesion = [500.0, 1500.0] and velocity = [-5.0, 40.0].

There are systematic errors in the performance of the model for predictions: (i) the error is not evenly distributed; (ii) sampling from the model has high variance; (iii) samples come from a bimodal distribution; and (iv) samples from the model have a smaller range than that of the training and test sets.

5.1 Training and Test Data

There are three key features of the data which are important to these investigations. First, the size of the initial training set is 320 points; although neural network machine learning algorithms are notoriously data intensive, expanding the dataset available was limited by the computational resource required. Second, the points in the training set and the test set were both distributed regularly, using Latin Hypercube Sampling (LHS), over the 5-D input parameter space. Lastly, the distribution of the velocities is roughly bimodal. As shown in Fig. 4 the final velocity of the simulation tends to fall into one of two modes, high ($20 \ldots 34$ m/s) or low ($-5 \ldots 5$ m/s), and is skewed to higher velocities.

In state-of-the-art classical machine learning these characteristics of the data could be accounted for by changing the algorithm. However, in the current early-stage quantum machine learning development, there are few viable options. Therefore, the dataset size limits the ability of the algorithm to effectively train models with larger number of nodes (and hence the number of parameters to be updated during training), which can result in underfitting. It might be easy to conclude from the large mean squared error that the model underfits, however it is argued here that there are other characteristics of the model unrelated to underfitting that are limiting the accuracy of the predictions.

The distribution, size, and skew of the dataset are factors which may influence the algorithm performance. There are characteristics of the results which seem to be correlated with the specifics of this dataset, and are discussed further in the following sections. As mentioned before, in classical settings it is possible to change the hyperparameters and features of the algorithm to account for this, whereas current limitations and constraints in quantum machine learning prevent from making the same adjustments.

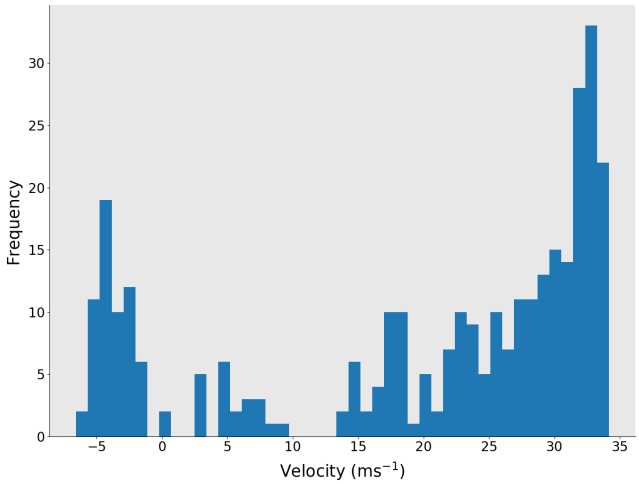


Figure 4: Distribution of the training data which demonstrates its skew and bimodal nature.

5.2 Model Fitting

Initial investigation into optimizing the model performance, primarily by varying the number of nodes in an attempt to correctly fit the model, indicated that the fit of the model was not the limiting factor for the mean squared error between the model predictions and the test set. This can be seen in Fig. 5 where a noise floor around 60 $(\mathrm{m/s})^2$ is observed for both the training and the test sets.

Two features of the data are discussed here: (i) the relative performance of the test and training set; and (ii) the noise floor. First, it is unusual to observe better performance on the test set than on the training set and observe correlation between performance on the training and test sets. Both these observations can be explained as characteristics of the sampling from the space and size of the dataset.

The datasets are small and the distribution of the squared error between the predictions and the data is not uniform. Indeed, the ranges of the performance distributions for the test and training sets are around 200 $(\mathrm{m/s})^2$ and 400 $(\mathrm{m/s})^2$, respectively. As the datasets are small, the large range means that the contributions of high squared error results are significant in the mean. This effect can usually be mitigated by bootstrapping, sampling many times from the distribution to find the true mean, though is not implemented in this case as the contributions are systematic errors that allow the shortcomings of the algorithm to be explored.

The distribution of the squared error for the training set is shown in Fig. 6. This plot indicates that there is a pattern to how the model is inaccurately predicting values: The lower velocities tend to have larger errors; there is a peak in the errors in around the central values; and there is a tendency for the squared error to increase at the extremities of the predictions.

We conclude that the better performance of the test set can be explained by chance, as there are relatively fewer points from the test set that are drawn from the large squared error parts of the distribution. Furthermore, the mean squared error on the test and training sets are correlated because they are both sampled using LHS from the distribution: roughly, any change in performance

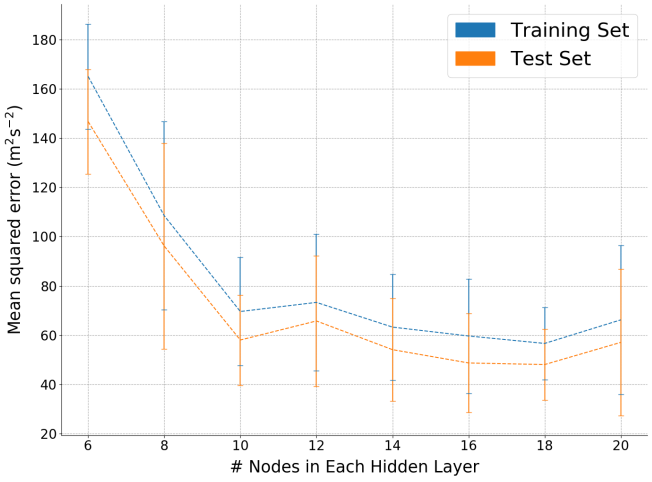


Figure 5: Average performance over 10 training runs of the model for the predictions of the test and training sets across node number. Each model was trained for 5000 epochs with learning rate $\eta = 0.01$ and regularization factor $\lambda = 0.0001$.

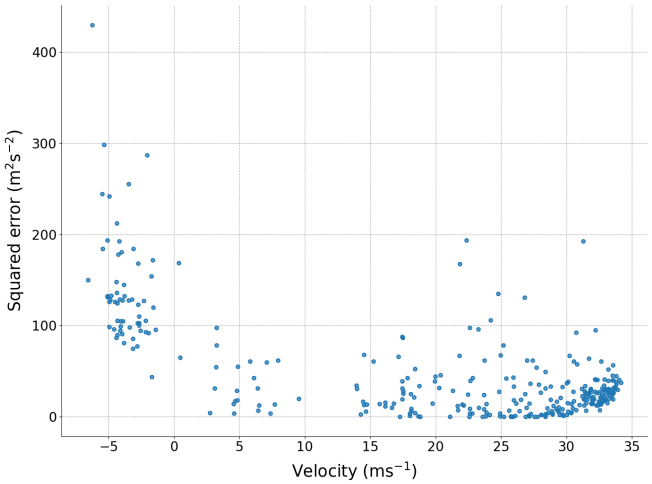


Figure 6: Distribution of the squared error between predictions of the model and the training set data.

of one will be reflected in the other. Occam's Razor applies not only because these hypotheses have a small number of assumptions, but also because there are no reasonable logical counterarguments.

There is no correlation between the number of nodes and the performance of the algorithm. This simple experiment shows that there are other limiting factors which are decreasing the performance of the model. In the next section we investigate different sampling methods and examine the results in an effort to identify other ways to reduce the noise floor.

5.3 Model Sampling

The model is sampled in order to make predictions about the velocity at a given point in the parameter space. Several different sampling procedures were investigated, each of which will be briefly outlined and the key results stated.

Sampling procedure 1. The steps of the first procedure discussed are outlined below:

(i) An input vector from the parameter space is chosen and the associated velocity for this vector is randomized;

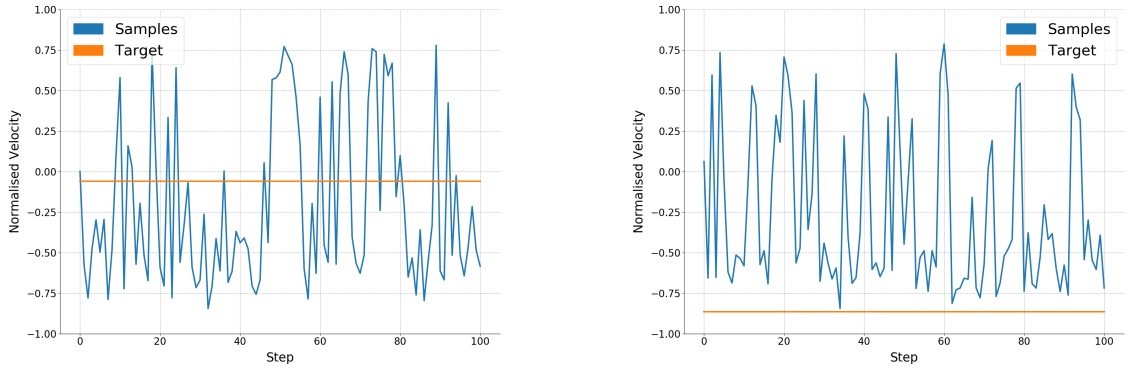
(ii) The activations of the top layer of the recognition network are calculated by a bottom-up pass;

(iii) The activations of this layer are used as inputs to the top layer of the generator network;

(iv) A sample from the generator network is obtained with a top-down pass;

(v) The sampled velocity is used as the new input velocity for the recognition network keeping all other inputs constant;

(vi) Repeat (ii) - (v) until equilibrium reached.



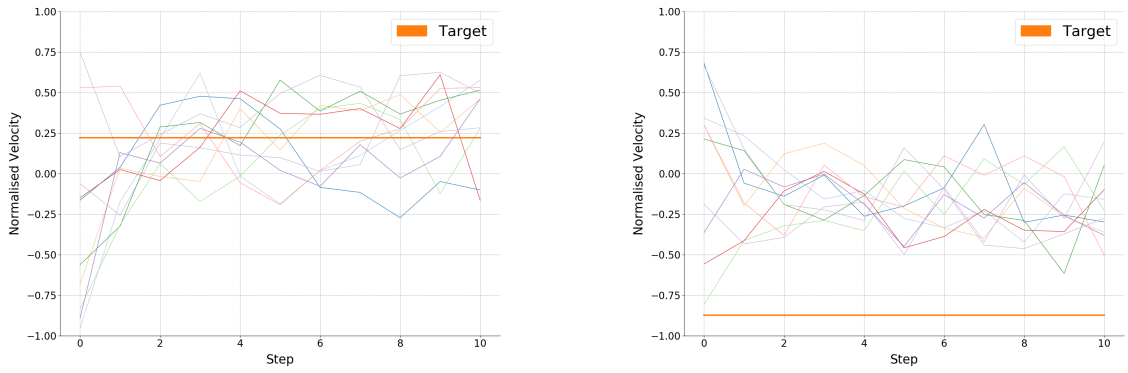
(a) (b)

Figure 7: Variation of velocity samples from generative model over 100 steps for different target velocities.

Each iteration of this subroutine represents a point in a Markov chain evolution of the machine state. The results are shown in Figs. 7a and 7b. We note that the Markov chain evolution does

not convincingly converge because of the high variance of the distribution. It was expected that this would result in a Markov chain evolution of the system where the sampling distribution would thermalise and the velocity parameter reached some stable value. Unfortunately, the variance of the samples is so large that no useful information can be inferred from a single sample.

Sampling procedure 2. The second sampling procedure was adjusted to mitigate the large variance of the samples: at each step of the Markov chain an average of samples is taken instead of one value. Predictably, this squeezes the results with the resulting average between the extremities of samples taken from the device, but does reduce the variance of single samples from the device; see Figs. 8a and 8b.



(a) (b)

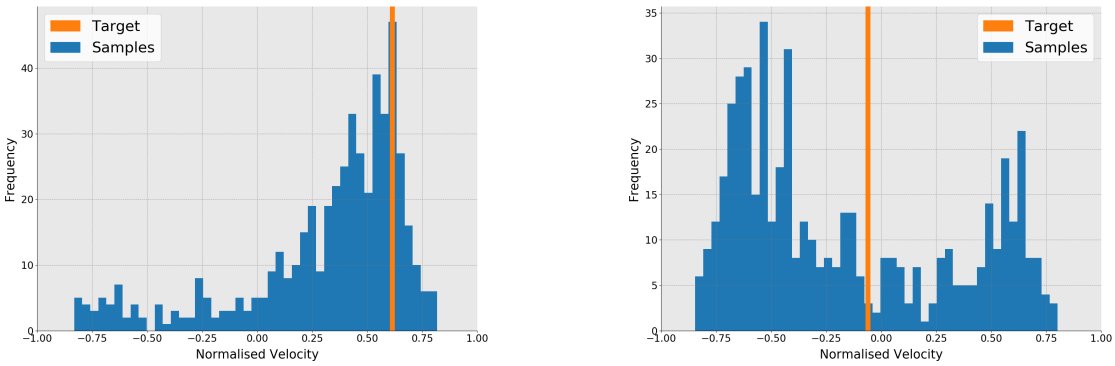
Figure 8: Averaging of samples at each step of the Markov chain squeezes the results and reduces the variance of the final result.

Sampling procedure 3. Third, many samples are taken at a given random velocity and the state of the system is not evolved, a mean of the peaks of the frequency of velocities is taken for a number of batches each at different random input velocities Figs. 9a and 9b. In Fig. 9b the peak frequency of samples is in the lower mode. This phenomena can be observed for target velocities across the whole range: The samples come from a mode that is not close to the target velocity. In the following section this is referred to as mode-switching as the model generates samples from the wrong mode of the distribution, which is particularly observable in the mid-range velocities where no mode exists.

These experiments indicate that the sampling procedure is not the main source of noise in the results and that, while adjusting the sampling to mitigate the core issue is possible, this may lead in further 'squeezing' of the results from the expected predicted values.

5.4 Lessons Learned

The issues outlined in these investigations are mode-switching and squeezing. In this section potential solutions and avenues of research are suggested to understand and mitigate these effects. Mode-switching results in dominant anomalous predictions and squeezing in a small systematic error for the predictions at the extremities of the model.



(a) (b)

Figure 9: 100 samples taken at a random velocity gives a wide distribution, sometimes with a peak at the desired value, in other cases not.

Throughout the investigations several sources of squeezing to the predictions have been highlighted. Although other sources may contribute to the effect, the evidence suggests that fundamental aspects of the training algorithm result in a model which squeezes predictions to some range less than the range of the input data. A potentially avenue of future research is investigating training algorithm adjustments to reduce variance and increase range of the resulting model.

It is possible to say with some confidence that the mode-switching effect is the combined effect of the structure of the data and some learning feature of the algorithm which results in samples with a high variance and with a bias toward regions trained with more data. Potential avenues are detailed in Section 6.

6 Results and Discussion

In this section two core examples of the results are presented. First, we provide a comparison of the learning of the classical and quantum algorithms. Second, we discuss the predictions of the quantum-assisted generative model and respective variations, considering both a quantum-assisted and a classical model. In both cases a 6 x 12 x 12 deep neural network is trained on a 340 point dataset from the 6-D parameter space for 5000 epochs with learning rate $\eta = 0.01$ and regularization factor $\lambda = 0.0001$.

The quantum and classical predictions are comparable in accuracy to the actual values and both sets of predictions have the same characteristics, namely non-uniform mean squared error distribution with a large range and squeezing, as explored in the algorithm investigations of Section 5.

The variations of predictions across the whole space generally followed the expected trend. In the case shown below, velocity is negatively correlated with slope and positively correlated with cohesion. The standard deviation of predictions in a slice of the parameter space shows how the regions of high variance of the predictions tend to be in regions of change, namely between the two modes (large and small velocity).

The first idea to pull from Fig. 10 is that the quantum algorithm is successfully training a model.

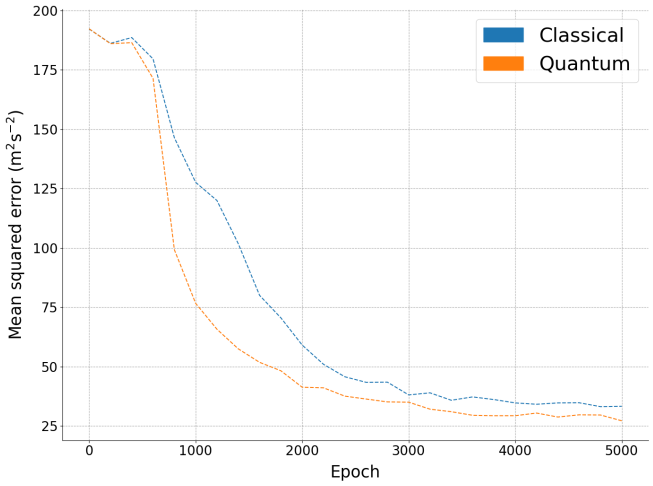


Figure 10: Comparison of the performance of the quantum and classical algorithms.

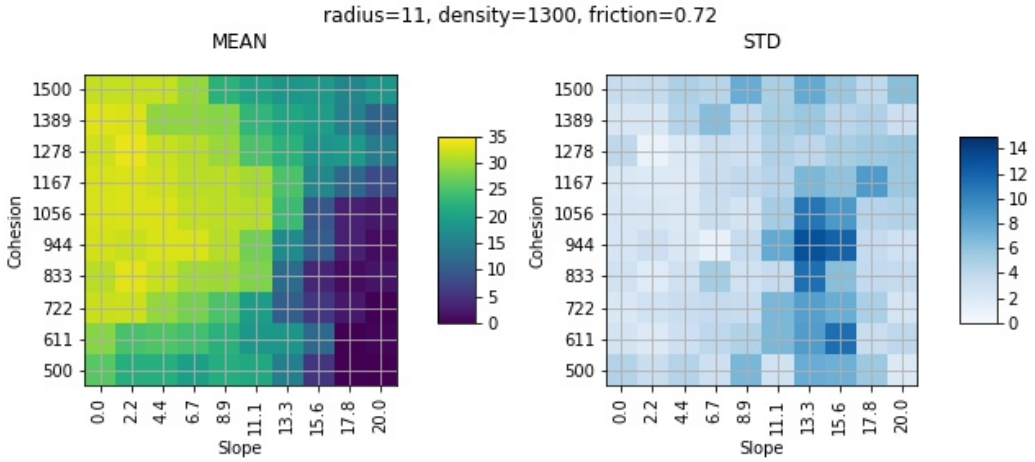


Figure 11: Prediction slice through the parameter space across the slope and cohesion parameters. Regions of high standard deviation are observed in regions of largest gradient in the velocity.

The second promising result is that the classical and quantum algorithms are comparable. Any slight difference can be attributed to the numerical differences incurred when translating between the two algorithms (e.g, a learning rate of 1 in the classical algorithm may not equate to a learning rate of 1 for the quantum algorithm). These ideas are complementary and a first step in establishing the algorithm and laying the groundwork for extensions, variations, and improvements.

Figure 11 confirms intuitive understanding of the data providing evidence that the quantum assisted model is learning the distribution. The variance of predictions also follows an expected pattern: in regions of large gradient of the velocity the predictions have a wider spread.

6.1 Further Work

There are many routes open to expanding this work. As an early stage implementation of sampling from a quantum annealer to assist classical machine learning, it stands as one of the first efforts to address the problems facing near-term implementation of quantum devices: small size of the quantum device, temperature estimation when sampling from the quantum Boltzmann distribution, and slow reprogramming at training epochs. Lessons learned here inform the development of quantum algorithms that will eventually surpass their classical counterparts. It is taken as a given that quantum devices will be used in some way for computation, with implementation as subroutines within hybrid classical-quantum architectures, like the one presented here, a likely scenario. However, as repeatedly demonstrated in this report, the real-world reality does not always conform to a flawless theoretical picture. This report is a collaborative effort across disciplines spanning the cutting edge of classical simulation, machine learning, and quantum computation. It is such collaborations that break the barriers between revolutionary theoretical research and the sandbox of the real-world.

We highlight here three key potential avenues for further exploration.

(i) First, it would be possible to explore other existing classical machine learning architectures where subroutines can be replaced with some quantum algorithm. As there are promising aspects of the results and advantages to the implementation contained within this report, the lessons learned here can be applied to the development of novel and potentially improved hybrid algorithms. Specific to the QAHM algorithm demonstrated, the applications of the generative model could be applied in a wide range of scenarios such as: time-series analysis, compression, fast search of data through example generation, and the effect of dataset size and complexity on the model.

Hardware examples of numerous quantum computing architectures have passed small-scale, proof of principle, tests and are now coming to the the second major hurdle: scalability. For the foreseeable future these different architectures will have various limitations, caveats, and idiosyncrasies that make their ability to solve real-world problems a challenge. However, when solved, quantum computers will have applications that are completely beyond the realm of classical computation. In this vein, future work in this field should aim at answering the question of how quantum devices will be used in real-world problems. This work directly addresses this question and the extensions suggested above follow this theme. For the first, exploring different architectures expands our understanding of the difficulties of integrating quantum devices with classical algorithms. Additionally to this, not only do the different classical algorithms have applications suited to each of them but their quantum-classical hybrid counterparts will have features that need to be matched to specific problems. Finally, any research into quantum-classical machine learning algorithms will inform the development of improved realizations. Understanding the shortcomings of any given algorithm would help choose appropriate problems and form alternative mathematical basis for another algorithm, which does not have the same problems for learning a distribution.

(ii) A second suggested extension relates to expanding real-world demonstrations of quantum devices at work. This may feed into research in the classical domain and also inform the development of this and other quantum algorithms, for example answering the questions:

Where can quantum devices be most useful? What are the advantages of using a quantum device? Are there applications particularly suited to hybrid machine learning algorithms?

For the problem of creating GO/NO-GO maps, the generative model could be exploited by selecting velocities and generating samples from the space to direct training and inform where additional simulations should be performed. This *sequential algorithm* approach is illustrated in Fig. 3 by the interplay between simulation and prediction. This would partly address the problem of the simulation computational cost; the generative capabilities of the model could inform the choice of parameters for simulation, thus reducing the number of required simulations. Concretely, phase 1 of this sequential approach would involve fitting the model to data over the whole space. Results of this learning phase could then direct the choice of points for the next simulations, for example towards regions of large variance in predictions.

Many scientific and engineering applications similar to the problem tackled here are likely to share one of the original constraints, namely the relatively small size of the training set. From an engineering perspective, it would be informative to future uses of this (or similar) algorithms to know how the amount of data affects model performance.

(iii) A third possible research avenue takes a closer look at the particular algorithm employed here. As shown in Section 5, there are errors in the model features. However, this particular QAHM algorithm has several advantages: the classical recognition network can map continuous variables and large data vectors to a small quantum device, the quantum device does not need to be queried for each data point, and the gray-box model outlined in Section 3 does not require that the system temperature be calculated. With these advantages in mind, there are significant incentives to look at variations on the QAHM, both to see if there can be improvements made to the model learned and to determine if alternative implementations can be applied to a wider class of applications.

All of the above has been concerned with the quantum machine learning aspects of the research. However, there are adjustments to the simulation that can be adapted to the requirements peculiar to quantum machine learning. Currently, the data is represented by a continuous 6-D vector. The real-world version of the GO/NO-GO problem invites a higher dimensionality input space, or put another way, using more parameters. Increasing the size of the input parameter space will require an increase in the number of nodes in visible layer of the neural network and therefore the number of parameters being learned. More parameters to learn by the neural network will require more data. It is possible that using lower fidelity simulations with a lower computational cost can increase the size of the dataset available, increasing the size of the possible neural network to be trained. Here it is noted that the disadvantages of using a lower fidelity simulation may be offset by the advantages of learning the space and the generative capabilities of the model. Furthermore, the model may be able to capture the inherent uncertainty in these lower fidelity simulations.

Overall, it is worth noting that the fields of quantum machine learning and multibody system dynamics simulation are at vastly different stages of development. This juxtaposition is likely to be mirrored across many of the potential applications of quantum devices. The future work suggested here aims at bridging the gap between real-world problems and the nascent quantum computing industry, and forms the necessary stepping stones needed to usher-in widespread adoption of these methods.

References

[1] P.W. Haley, M.P. Jurkat, and P.M. Brady. NATO reference mobility model. Technical Report 12503, U.S. Army TARDEC, 1979.

[2] Seth Lloyd, Masoud Mohseni, and Patrick Rebentrost. Quantum algorithms for supervised and unsupervised machine learning. *arXiv:1307.0411*, 2013.

[3] Marcello Benedetti, John Realpe Gmez, and Alejandro Perdomo-Ortiz. Quantum-assisted helmholtz machines: A quantum-classical deep learning framework for industrial datasets in near-term devices. *Quantum Science and Technology*, 2018.

[4] Geoffrey E. Hinton, Peter Dayan, Brendan J. Frey, and Radford M. Neal. The wake-sleep algorithm for unsupervised neural networks. *Science*, 268(5214):1158, 1995.

[5] Marcello Benedetti, John Realpe-Gómez, Rupak Biswas, and Alejandro Perdomo-Ortiz. Quantum-assisted learning of hardware-embedded probabilistic graphical models. *Phys. Rev. X*, 7:041052, Nov 2017.

[6] Laurent Younes. On the convergence of markovian stochastic algorithms with rapidly decreasing ergodicity rates. *Stochastics: An International Journal of Probability and Stochastic Processes*, 65(3-4):177–228, 1999.

[7] Project Chrono. Chrono: An Open Source Framework for the Physics-Based Simulation of Dynamic Systems. `http://www.projectchrono.org`. Accessed: 2015-02-07.

[8] A. Tasora, R. Serban, H. Mazhar, A. Pazouki, D. Melanz, J. Fleischmann, M. Taylor, H. Sugiyama, and D. Negrut. Chrono: An open source multi-physics dynamics engine. In T. Kozubek, editor, *Lecture Notes in Computer Science, in print*, pages –. Springer, 2016.

[9] Rainer Krenn and Gerd Hirzinger. SCM – A soil contact model for multi-body system simulations. In *11th European Regional Conference of the International Society for Terrain-Vehicle Systems, Bremen, Germany*, 2009.

[10] R. Serban, M. Taylor, D. Negrut, and A. Tasora. Chrono::Vehicle Template-Based Ground Vehicle Modeling and Simulation. *Intl. J. Veh. Performance*, in press, 2018.

[11] C. O'Sullivan. Particle-based discrete element modeling: Geomechanics perspective. *Int. J. Geomech.*, 11(6):449–464, 2011.

[12] P. Cundall and O. Strack. A discrete element model for granular assemblies. *Geotechnique*, 29:47–65, 1979.

[13] H. Kruggel-Emden, E. Simsek, S. Rickelt, S. Wirtz, and V. Scherer. Review and extension of normal force models for the discrete element method. *Powder Technology*, 171:157–173, 2007.

[14] H. Kruggel-Emden, S. Wirtz, and V. Scherer. A study of tangential force laws applicable to the discrete element method (DEM) for materials with viscoelastic or plastic behavior. *Chem. Eng. Sci.*, 63:1523–1541, 2008.

[15] K. Samiei, B. Peters, M. Bolten, and A. Frommer. Assessment of the potentials of implicit integration method in discrete element modelling of granular matter. *Computers & Chemical Engineering*, 49:183–193, 2013.

[16] A. Pazouki, M. Kwarta, K. Williams, W. Likos, R. Serban, J. Jayakumar, and D. Negrut. Compliant versus rigid contact: A comparison in the context of granular dynamics. *Phys. Rev. E*, 96, 2017.

[17] E. J. Haug. *Computer-Aided Kinematics and Dynamics of Mechanical Systems Volume-I.* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.

[18] A. A. Shabana. *Dynamics of Multibody Systems.* Cambridge University Press, Cambridge, England, fourth edition, 2013.

[19] A. F. Filippov and F. M. Arscott. *Differential Equations with Discontinuous Righthand Sides: Control Systems*, volume 18. Springer, 1988.

[20] David E. Stewart and Jeffrey C. Trinkle. An implicit time-stepping scheme for rigid-body dynamics with inelastic collisions and Coulomb friction. *International Journal for Numerical Methods in Engineering*, 39:2673–2691, 1996.

[21] M. Anitescu, J. F. Cremer, and F. A. Potra. Formulating 3D contact dynamics problems. *Mechanics of Structures and Machines*, 24(4):405–437, 1996.

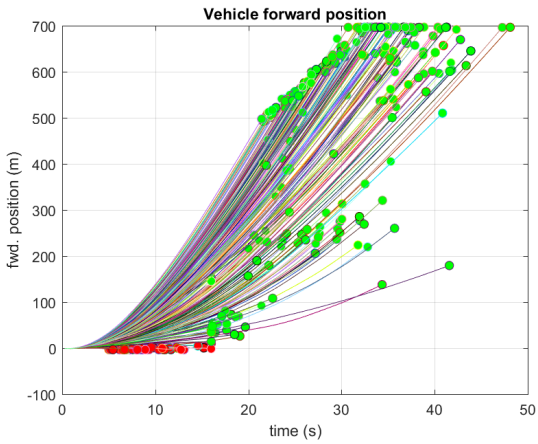
[22] Jeffrey Trinkle, Jong-Shi Pang, Sandra Sudarsky, and Grace Lo. On dynamic multi-rigid-body contact problems with Coulomb friction. *Zeitschrift fur angewandte Mathematik und Mechanik*, 77:267–279, 1997.

[23] Danny M Kaufman and Dinesh K Pai. Geometric numerical integration of inequality constrained, nonsmooth hamiltonian systems. *SIAM Journal on Scientific Computing*, 34(5):A2670–A2703, 2012.

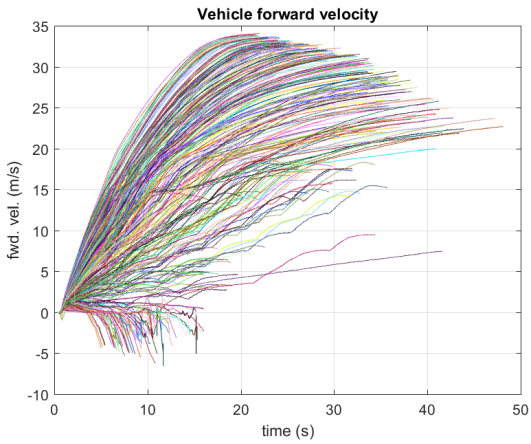
[24] M. Anitescu and A. Tasora. An iterative approach for cone complementarity problems for nonsmooth dynamics. *Computational Optimization and Applications*, 47(2):207–235, 2010.

[25] M. Anitescu. Optimization-based simulation of nonsmooth rigid multibody dynamics. *Mathematical Programming*, 105(1):113–143, 2006.

[26] H. Mazhar, T. Heyn, A. Tasora, and D. Negrut. Using Nesterov's method to accelerate multibody dynamics with friction and contact. *ACM Trans. Graph.*, 34(3):32:1–32:14, 2015.



(a) Final longitudinal vehicle position.



(b) Final forward vehicle speed.

Figure 12: Results of the 320 simulations in the initial training set.

A Performance of Vehicle Simulation on Granular Terrain

Around 500 separate straight-line acceleration maneuvers on inclined granular terrain were simulated to generate various training and test sets for the QML algorithm. In all cases, the vehicle was an off-road wheeled vehicle with independent double-wishbone suspension and Pitman arm steering. These simulations were conducted for different up-slopes (ranging from 0° to 20°), particle radius (ranging from 8 mm to 18 mm) and different values for soil parameters (material density, coefficient of friction, and cohesion pressure). A moving patch method was used, resulting in a number of particles ranging from 48,000 to 480,000.

Typical simulation results, depicting the final longitudinal vehicle position (along the incline) and the maximum achievable vehicle speed, are presented in Fig. 12. Trajectories with a final green marker indicate a GO scenario, while red ones indicate a NO-GO situation (vehicle sliding backward).

Simulations were conducted on a Cray XC30 system with 12-core Intel® Xeon® E5-2697 v2 processors and a dedicated Cray Aries high-speed network, with each independent run using 24 OpenMP threads (parallelization provided by the *Chrono*::Parallel module). To assess computational efficiency and allow comparison of runs with different number of particles and of different time length, the following performance metric was calculated:

$$E = \frac{T_{wc}/T_s}{n_p/1000},$$

where T_{wc} represents the wall-clock time, T_s the simulated time, and n_p the number of particles. In other words, we report the time to simulate 1000 particles for 1 s. Figure 13 shows the value of E for a set of 320 simulation runs (used as the initial QML training set). Values corresponding to NO-GO cases are marked with a red circle. The performance metric range for GO simulations was $E \in [10.81, 16.19]$, with a mean value $\bar{E} = 12.52$.

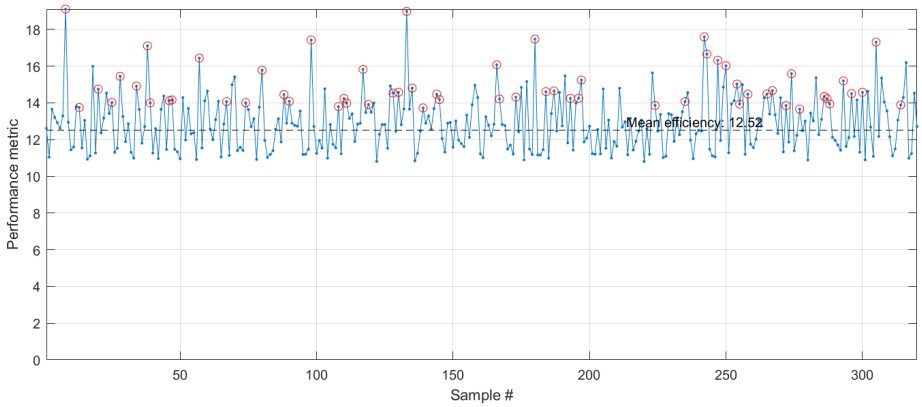


Figure 13: Performance metric of the 320 simulations in the initial training set.