ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Restriced Variance Optimization with Gradient Enhanced Kriging for Variational Monte Carlo

Master Thesis

Yoel Zimmermann

Tuesday 8th July, 2025

Advisors: Prof. Sandeep Sharma (Caltech), Prof. Markus Reiher (ETHZ)
Department of Chemistry and Applied Biosciences, ETH Zürich

Abstract

In this thesis, we explore the application of surrogate-based Restricted Variance Optimization with Gradient-Enhanced Kriging to the variational Monte Carlo problem. The method iteratively constructs a Gaussian process surrogate model of the variational Monte Carlo energy landscape from both energies and gradients of past evaluations. By restricting optimization steps to regions of low predictive variance, the method can improve robustness in noisy, non-convex optimization problems. While cubic scaling of Gaussian processes sets practical limits on the number of evaluations and parameters, the global surrogate enables effective reuse of information across iterations. Better sample efficiency over conventional local optimization techniques is demonstrated for an exactly solvable system with emulated Monte Carlo noise.

Contents

Chapter 1

Introduction

Finding the ground state of a given system is a paradigmatic challenge in quantum many-body physics. While exact diagonalization methods exist for small systems, quantum mechanics's exponential scaling with respect to system size renders them intractable for realistic applications [1, 2]. Even a simple system of N spin-1/2 particles, where each particle can only be in one of two states has a Hilbert space of dimension 2^N. Fortunately, the variational principle provides a window of opportunity: it states that for any trial wavefunction, called ansatz, the expected energy is always greater than or equal to the true ground state energy [3, 1]. Therefore, if we choose an ansatz flexible enough to approximate the true ground state, we can find that state by just minimizing the energy with respect to the ansatz parameters. In practice, computing the expected energy of a given ansatz is also nontrivial as it essentially requires the computation of the energy over all possible states. This is where the Monte Carlo (MC) method, ubiquitous in all areas of computational physics [4], comes into play. Instead of evaluating the entire Hilbert space, one can use random samples of the space to statistically estimate the true energy. In combination with the variational principle, this leads to the variational Monte Carlo (VMC) method. The resulting optimization problem, however, remains highly non-convex, high-dimensional, and plagued by noisy data. Traditional first- and second-order optimization methods can become inefficient or unstable in such settings [5, 2, 6]. The recent success of neural network ansätze [7, 8, 9, 10], where the wave function is parametrized by artificial neural networks which can act as universal function approximators [11], has led to a refreshed interest from the community in new optimization methods.

This thesis explores Gaussian processes (GPs), a method from probabilistic machine learning, to accelerate the optimization in VMC. GPs can act as non-parametric regressors to model unknown functions and offer a natural way for uncertainty quantification through the Bayesian framework in which they are derived [12, 13]. In particular, gradient-enhanced GPs (also called gradient-enhanced Kriging, GEK) incorporate both function values and gradient information in one probabilistic model, which can offer more accurate surrogate models [14, 15, 16]. This is especially true in regimes where data is sparse like in VMC.

Our main contribution is the application of restricted-variance optimization with gradient-enhanced Kriging (RVO-GEK) [17] to the variational quantum state optimization problem. RVO-GEK replaces the local parametric surrogates used in traditional optimization with a smooth, global, and uncertainty-aware Gaussian process model. In contrast to methods that constrain the optimization step based on heuristic step sizes or local curvature information [6], RVO-GEK restricts optimization to regions where the uncertainty of the surrogate remains below a specified threshold. This allows for a more informed exploration of the parameter space while minimizing the risk of making optimization steps into poorly understood regions. While RVO-GEK and GPs more generally have been used in the context of quantum chemistry in recent years – e.g. for molecular dynamics [18], geometry optimization [17, 19, 20, 21], and even as an ansatz [22] – this is the first application to the VMC optimization problem itself to the author's knowledge.

The thesis is structured as follows. In chapter 2, we give an introduction to GPs, starting from linear regression and progressing through the kernel trick to the formal definition of GPs as distributions over function. We then give an extensive review of how GPs can incorporate gradient observations and account for heteroscedastic noise. We then summarize the necessary theoretical background behind the variational principle and VMC, including methods for stochastic sampling and variance reduction. Furthermore, we review different optimization techniques for VMC such as gradient descent, stochastic reconfiguration [23], and the more recent Rayleigh-Gauss-Newton method [6]. Finally, we introduce RVO-GEK and connect it conceptually and mathematically to the existing methods.

In chapter 3, we apply RVO-GEK to the one-dimensional transverse-field Ising model, a well-studied model system [24]. Using a restricted Boltzmann machine ansatz, a type of artificial neural network, we demonstrate how the method performs in practice, including parameter sweeps over kernel hyperparameters and noise levels, and comparisons to existing optimization methods. With the limited computational resources available, we emulate MC noise and use exact solutions as ground truth.

Finally, in chapter 4, we briefly reflect on the strengths and limitations of the method. While the use of full gradient-enhanced Gaussian processes can be prohibitively expensive due to the cubic scaling with the number of observations, approximate methods, and sparse GPs offer promising directions for scaling up to systems of real interest.

Chapter 2

Theory

2.1 Gaussian Processes

2.1.1 Linear Regression and the Kernel Trick

Before we introduce the concept of Gaussian processes (GPs), it makes sense to take a step back. One of the most important concepts in machine learning is regression. The key idea is to learn some relationship between inputs $\boldsymbol{x}$ and outputs $\boldsymbol{y}$ for any given input $\boldsymbol{x}$ based on finite data $(\boldsymbol{x}_i, y_i) \in \mathcal{D}$. The most basic version of regression is linear regression where the relationship is some (affine) linear combination of the inputs:

$$f[w_0, w_1, w_2, \ldots](\boldsymbol{x}) = w_0 + w_1 x_1 + w_2 x_2 + \ldots, \tag{2.1}$$

where the $w_0, w_1, \ldots$ have to be determined through a so-called learning algorithm. The most basic idea is to look at the observed input, output pairs $(\boldsymbol{x}_i, y_i)$ to then determine the distance between our model $f(\boldsymbol{x})$ and the observed data. For this purpose, one defines a loss (objective, optimization goal), most commonly the quadratic loss

$$\mathcal{L}(\boldsymbol{w}) = \sum_{(\boldsymbol{x}_i, y_i)} (f[w_0, w_1, w_2, \ldots](\boldsymbol{x}_i) - y_i)^2. \tag{2.2}$$

This loss is differentiable in w_i (this is why a quadratic loss is used instead of, e.g., a absolute value difference) and yields a convex optimization problem, i.e., there exists a unique solution to the objective

$$\boldsymbol{w}^* = \arg\min_{\boldsymbol{w} \in \mathbb{R}^n} \mathcal{L}(\boldsymbol{w}) \tag{2.3}$$

This minimum can easily be found by solving the equation $\nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{w}) = 0$, which gives the canonical equation

$$X^\top X \boldsymbol{w}^* = X^\top \boldsymbol{y}, \tag{2.4}$$

where X is the $N \times d$ design matrix with rows $\boldsymbol{x}_i^\top$, and $\boldsymbol{y}$ is the vector of all targets y_i. This raises the question of how to learn relationships that are more complex than just linear. One could think of redefining the parametrization in Equation 2.1 to feature some more complex relationship. Then, however, we would lose access to the normal equations that we know how to solve with standard numerical linear algebra packages. What one ends up doing is simply to transform the input features into some higher-dimensional space with a feature map $\varphi(x)$ and then treat the problem as a regular linear regression. One could, e.g., map the input features to a space that includes the squares of all inputs and could effectively perform a quadratic regression, or more generally some polynomial regression. One can similarly also do exponential or logarithmic regression following the same scheme and construct very expressive feature maps. This naturally raises the issue of determining the appropriate level of expressiveness for our model. We can in principle construct feature maps that drive the loss down to zero, i.e., fitting the training data exactly. However, this is rarely of interest. What we are actually interested in is the generalization error, i.e., the expected error or loss on new data generated from the same distribution as the training data. This naturally leads to trade-off of model complexity and model generalization, one of the most important concepts in machine learning. What is often done is to penalize model complexity with some additional regularization term in the loss function, e.g., one could add a $\lambda \|\boldsymbol{w}\|^2$ to the loss function which would drive down the use of features to only the most needed to reduce the loss and to not use unnecessary complexity.

In solving the normal equations, one can observe that we only need the scalar products of individual features for the solution and not the actual input features in the high-dimensional feature space. One can exploit this by defining the so-called kernel, which is the matrix of all possible scalar products between inputs

$$K_{ij} = \langle \varphi(\boldsymbol{x}_i), \varphi(\boldsymbol{x}_j) \rangle. \tag{2.5}$$

One can then rewrite the normal equation and its solution in the dual formulation as

$$f(\boldsymbol{x}) = \sum_{i=1}^{N} \alpha_i K(\boldsymbol{x}_i, \boldsymbol{x}), \tag{2.6}$$

where the α_i are coefficients obtained by solving

$$(K + \lambda I)\boldsymbol{\alpha} = \boldsymbol{y}, \tag{2.7}$$

and where the extra term λI results from the regularization term. Conveniently, instead of constructing the kernel matrix from the perspective of the feature map, one can just define the kernel directly. It only has to obey the properties of the scalar product, i.e., it has to be positive semi-definite and symmetric, which guarantees that this feature map exists by Mercer's theorem [25, 26, 27]. The kernel for polynomial regression can e.g. be constructed as

$$K_{\text{Polynomial}}(\boldsymbol{x}, \boldsymbol{x}') = (\langle \boldsymbol{x}, \boldsymbol{x}' \rangle + c)^d, \tag{2.8}$$

where c is a constant and d is the degree of the polynomial. Now one can also define new kernels such as the radial basis function (RBF) kernel[1]:

$$K_{\text{RBF}}(\boldsymbol{x}, \boldsymbol{x}') = \sigma_f^2 \exp\left(-\frac{\|\boldsymbol{x} - \boldsymbol{x}'\|^2}{2\ell^2}\right), \tag{2.9}$$

where $\sigma_f^2 > 0$ is called signal variance and $\ell > 0$ is the lengthscale, which can be thought of as the distance one has to move in input space before the function can change significantly [25]. We can now ask the reverse question of what feature space that would correspond to. To motivate the answer, consider $x \in \mathbb{R}^1$ and the Taylor decomposition [28]:

$$\sigma_f^2 \exp\left(-\frac{\|x - x'\|^2}{2\ell^2}\right) = \sigma_f^2 e^{-\frac{1}{2\ell^2}x^2 + \frac{1}{\ell^2}xx' - \frac{1}{2\ell^2}x'^2} \tag{2.10}$$

$$= \sigma_f^2 e^{-\frac{1}{2\ell^2}x^2 - \frac{1}{2\ell^2}x'^2} \left(1 \cdot 1 + \sqrt{\frac{1/\ell^2}{1!}}\, x \cdot \sqrt{\frac{1/\ell^2}{1!}}\, x' + \sqrt{\frac{(1/\ell^2)^2}{2!}}\, x^2 \cdot \sqrt{\frac{(1/\ell^2)^2}{2!}}\, x'^2 \right. \tag{2.11}$$

$$\left. + \sqrt{\frac{(1/\ell^2)^3}{3!}}\, x^3 \cdot \sqrt{\frac{(1/\ell^2)^3}{3!}}\, x'^3 + \cdots \right) = \varphi(x)^\top \varphi(x'). \tag{2.12}$$

Therefore,

$$\varphi(x) = \sigma_f^2 e^{-\frac{1}{2\ell^2}x^2} \left[1,\ \sqrt{\frac{1/\ell^2}{1!}}\, x,\ \sqrt{\frac{(1/\ell^2)^2}{2!}}\, x^2,\ \sqrt{\frac{(1/\ell^2)^3}{3!}}\, x^3,\ \cdots\right]^\top. \tag{2.13}$$

The feature space is therefore infinite-dimensional. The fact that we can do linear regression in an infinite-dimensional feature space is why this dual formulation is known as the "Kernel Trick". Of course, we do not actually compute these features and cannot solve Equation 2.4 explicitly. Hence the RBF kernel induces a so-called non-parametric model, as opposed to models with a finite number of parameters w. The class of possible functions is strictly only defined by the two hyperparameters σ_f^2 and ℓ but still incredibly expressive.

What we have done so far is get a line of best fit, i.e., the function that is most likely. There will, however, be some areas in our input space that we

[1] Sometimes also referred to as the squared-exponential (SE) kernel.

have way more data than for other. Especially for expressive models like the one we get using the RBF kernel, the question is how we can quantify the uncertainty of our estimate. For this, a Bayesian treatment is exactly what we need.

2.1.2 The Bayesian View of Regression

Everything we have discussed so far has taken place in the frequentist paradigm, the conventional modus operandi in physics and much of classical machine learning. Here, model parameters are fixed quantities, and uncertainty arises only indirectly from noisy observations. The Bayesian paradigm offers a fundamentally different lens through which we can understand regression. At the heart of Bayesianism lies Bayes' theorem,

$$\text{posterior} = \frac{\text{likelihood} \times \text{prior}}{\text{evidence}}, \qquad p(\boldsymbol{w} \mid \mathcal{D}) = \frac{p(\mathcal{D} \mid \boldsymbol{w})\, p(\boldsymbol{w})}{p(\mathcal{D})}. \tag{2.14}$$

Bayesian regression does not begin with the assumption that there is a "true" set of weights to be discovered, but rather with a prior belief, i.e., a probability distribution over plausible weights, which is then updated after observing the data.

Let us consider Bayesian linear regression, where one assumes that the data $\boldsymbol{y}$ is generated by a noisy linear model

$$\boldsymbol{y} = X\boldsymbol{w} + \boldsymbol{\varepsilon}, \tag{2.15}$$

where the noise is independent and identically distributed (i.i.d.), $\boldsymbol{\varepsilon} \sim \mathcal{N}\left(0, \sigma_n^2 I\right)$. The likelihood of observing the data $\boldsymbol{y}$ under a given set of weights $\boldsymbol{w}$ is then given as

$$p(\boldsymbol{y} \mid X, \boldsymbol{w}) = \mathcal{N}(X\boldsymbol{w}, \sigma_n^2 I). \tag{2.16}$$

If we now additionally impose a Gaussian prior on the weights:

$$p(\boldsymbol{w}) = \mathcal{N}(0, \sigma_p^2 I), \tag{2.17}$$

and combine it with the likelihood, we get the posterior modulo normalization

$$p(\boldsymbol{w} \mid \boldsymbol{y}, X) \propto p(\boldsymbol{y} \mid X, \boldsymbol{w}) \cdot p(\boldsymbol{w}). \tag{2.18}$$

This gives us a full distribution of the weights of our linear model including variance, given the data. To find the most likely weights we can maximize the posterior, which yields the maximum a posteriori (MAP) estimator

$$\boldsymbol{w}_{\text{MAP}} = \arg\max_{\boldsymbol{w}} \left[-\frac{1}{2\sigma_n^2} \|\boldsymbol{y} - X\boldsymbol{w}\|^2 - \frac{1}{2\sigma_p^2} \|\boldsymbol{w}\|^2 \right], \tag{2.19}$$

where a logarithm had been applied to get the exponent from the in density of the normal distribution as it does not change the solution. This leads to the solution

$$w_{\mathrm{MAP}} = \left(X^\top X + \frac{\sigma_n^2}{\sigma_p^2} I \right)^{-1} X^\top \boldsymbol{y}, \tag{2.20}$$

which is nothing more than the regularized linear regression with regularization parameter $\lambda = \frac{\sigma_n^2}{\sigma_p^2}$. What previously looked like a heuristic penalty on large weights now emerges as a natural consequence of the Gaussian prior that pulls the weights to the origin.

This Bayesian perspective reveals an entirely different way to think about prediction: instead of selecting a single best-fit function, we can infer a full distribution over functions, each weighted by how well it explains the data. This idea comes to full fruition in GPs, which generalize Bayesian linear regression from distributions over weights to distributions over functions themselves. Instead of specifying priors on finite-dimensional parameters, we can place priors directly on a function space, defined entirely by a mean and a kernel function and perform inference in a non-parametric, fully Bayesian manner.

The text book definition of GPs is that they are a collection of random variables such that every finite subset is a multivariate Gaussian distribution [12]. Formally,

$$f(\boldsymbol{x}) \sim \mathcal{GP}(\mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')), \tag{2.21}$$

for a given mean function

$$\mu(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})], \tag{2.22}$$

and the kernel (covariance) function specifies how function values at different inputs covary

$$K(\boldsymbol{x}, \boldsymbol{x}') = \mathrm{Cov}\left[f(\boldsymbol{x}), f(\boldsymbol{x}')\right] = \mathbb{E}\left[(f(\boldsymbol{x}) - \mathbb{E}[f(\boldsymbol{x})])(f(\boldsymbol{x}') - \mathbb{E}[f(\boldsymbol{x}')])\right]. \tag{2.23}$$

In particular, this means that for any finite set of inputs $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n$, the corresponding outputs follow

$$[f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_n)]^\top \sim \mathcal{N}(\boldsymbol{\mu}, K), \tag{2.24}$$

where K is essentially the kernel matrix as previously introduced.

2.1.3 Kernel Choice and Function Regularity

Instead of defining basis functions and finding weights, we can again directly define properties of the functions we want to learn through the covariance function. For example, a kernel function that rapidly decreases with distance between input encodes the belief that the function varies rapidly. This idea can be made precise using results from the well-developed theory of stochastic processes, where GPs are a central example. In this framework, functions are modeled as random processes, and concepts such as continuity or differentiability must be defined in a probabilistic sense, typically using notions like mean-square continuity or differentiability, and making statements that hold almost surely with respect to some probability measure [13, 29, 30, 25]. Roughly speaking, however, the sample functions inherit their smoothness and regularity from the kernel. For instance, the RBF kernel yields functions that are continuous and infinitely differentiable, while the Brownian motion kernel $K(\boldsymbol{x}, \boldsymbol{x}') = \min(\boldsymbol{x}, \boldsymbol{x}')$ leads to continuous but nowhere differentiable functions. An available choice between these two extremes is the Matérn kernel [31, 32], which allows for a continuous tuning of the smoothness with an additional parameter $\nu > 0$:

$$K_{\text{Matérn}}(\boldsymbol{x}, \boldsymbol{x}') = \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|}{\ell} \right)^{\nu} K_{\nu} \left(\frac{\sqrt{2\nu}\|\boldsymbol{x} - \boldsymbol{x}'\|}{\ell} \right), \tag{2.25}$$

where K_ν is a modified Bessel function and the gamma function $\Gamma(\nu) = \int_0^\infty t^{\nu-1} e^{-t}\, \mathrm{d}t$ acts as a normalization factor. When $\nu = \frac{1}{2}$, the Matérn kernel produces continuous but nowhere differentiable sample functions. As ν increases, the functions become progressively smoother, and in the limit $\nu \to \infty$, the Matérn kernel recovers the RBF kernel, yielding infinitely differentiable functions. This makes the Matérn kernel particularly useful for spatial statistics [33] or modeling physical processes [25], as it allows one to encode more fine-grained assumptions about function regularity. Sample functions for different choices of kernels are shown on the RHS of Figure 2.1.

2.1.4 Gaussian Process Regression

Most of the time, we are not interested in sampling functions from the GP prior itself, but from the posterior that incorporates information from the training data. For this purpose, consider a set of training data $(x_i, y_i)_{i=1}^n$ and for now assume that the observations were made in a noiseless setting, i.e., $y_i = f(x_i)$. Given an additional number of test points $\{x_i^*\}_{i=1}^{n^*}$, the joint distribution of the outputs at the training and test points, $\boldsymbol{f} = f(X) = [f(x_1), \dots, f(x_n)]$ and $\boldsymbol{f}^* = f(X^*) = [f(x_1^*), \dots, f(x_n^*)]$ under the GP prior is

$$\begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{f}^* \end{bmatrix} \sim \mathcal{N}\left(\boldsymbol{0}, \begin{bmatrix} K(X,X) & K(X,X^*) \\ K(X^*,X) & K(X^*,X^*) \end{bmatrix}\right), \tag{2.26}$$

where $K(X, X^*)$ is the $n \times n^*$ matrix of the covariance function evaluated at all pairs of training with test points and $K(X, X), K(X^*, X)$, and $K(X^*, X^*)$ are defined analogously. Now we want to restrict this distribution from a very general one, fully defined by the covariance function, to one that only

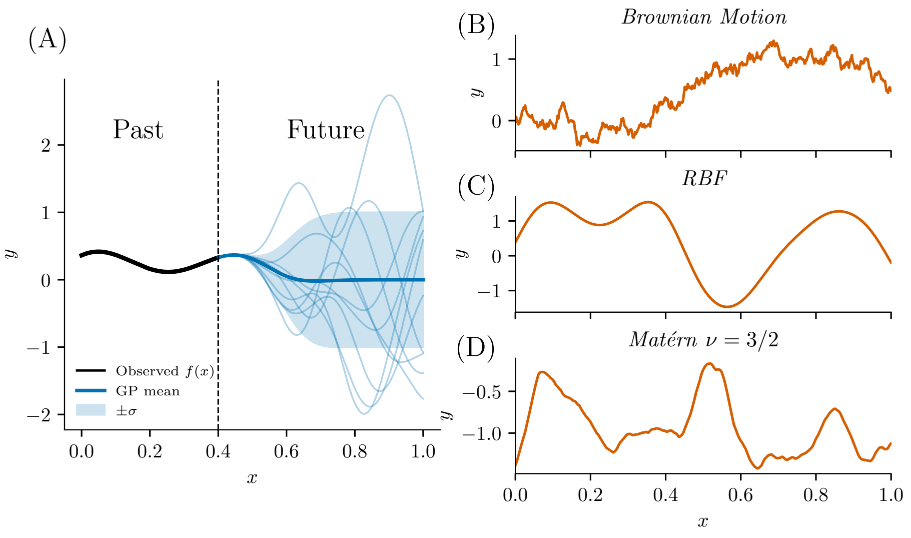

Figure 2.1: Illustration of Gaussian processes (GPs) as stochastic processes, extrapolation behavior and sample functions under different kernels. (A) Extrapolation of a GP conditioned on function values in ob served "past" interval $x \in [0, 0.4]$ (black line). The GP posterior mean (dark blue), standard deviation (shaded), and posterior sample functions (light blue) illustrate the expansion of uncertainty in the unobserved "future" ($x > 0.4$). This fan-like structure illustrates the stochastic nature of GPs. Uncertainty grows with distance, dictated by the particular choice of covariance function. (B) Sample function from a GP with a Brownian motion kernel, producing continuous but nowhere differentiable functions. (C) Sample function from a GP with a RBF kernel. This kernel produces extremely smooth functions with strong correlations over potentially long distances, depending on lengthscale ℓ. (D) Sample function from a Matérn kernel with smoothness parameter $\nu = 3/2$. Compared to the RBF kernel, the Matérn class allows for rougher functions, and offers more flexibility when the underlying function is expected to be less smooth or only a few times differentiable.

gives sample functions that go through the observed y_i. A naïve approach could be to sample from the prior and to only accept the functions that do, similar to the acceptance-rejection method for sampling from general distributions. However, this is intractable in high-dimensional settings, as the probability of randomly drawing a function that satisfies the desired conditions tends to zero. Fortunately, analytic conditionals for the normal distribution are available in closed-form and so one can show [25, 12] that the distribution of $\boldsymbol{f}^*$ conditioned on fixed $\boldsymbol{f} = [y_1, \dots, y_n]$ is given by

$$\begin{aligned} \boldsymbol{f}^* | X^*, X, \boldsymbol{f} \sim \mathcal{N}\big(&K(X^*, X)\, K(X, X)^{-1} \boldsymbol{f}, \\ &K(X^*, X^*) - K(X^*, X)\, K(X, X)^{-1} K(X, X^*)\big)\,. \end{aligned} \tag{2.27}$$

Note that the conditional covariance is simply the prior covariance minus the positive semi-definite term $K(X^*, X)K(X, X)^{-1}K(X, X^*)$, which represents the uncertainty that is resolved by observing the data. This is entirely independent of the actual observed function values and only depends on the input locations, which is a property of the normal distribution [25]. The stochastic process nature of GPs and sample functions from a GP posterior are illustrated on the LHS of Figure 2.1.

2.1.5 Incorporating Observation Noise

As discussed, in most practical scenarios, we do not observe the true function values directly, but rather noisy versions thereof, modeled as $y_i = f(x_i) + \varepsilon_i$, where $\varepsilon_i \sim \mathcal{N}\left(0, \sigma_n^2\right)$. GPs can quite naturally incorporate this noise through modifying the prior over observations by adding a noise term to the covariance, yielding

$$\operatorname{Cov}\left(y_l, y_m\right) = K\left(x_l, x_m\right) + \sigma_n^2 \delta_{l,m} \implies \operatorname{Cov}\left(\boldsymbol{y}\right) = K\left(X, X\right) + \sigma_n^2 I, \tag{2.28}$$

where $\delta_{l,m}$ is the Kronecker delta, which manifests through the assumption that the noise is independent. The assumption that the noise level σ_n^2 is the same for all observations, which leads to the simple identity matrix in the equation, is known as homoscedasticity. While this is a reasonable simplification for most use cases, in this thesis, we will deal with heteroscedastic noise, i.e., each observation will have a different noise level $\varepsilon_i \sim \mathcal{N}(0, \sigma_{n,i}^2)$, which leads to the slightly adapted

$$\operatorname{Cov}\left(\boldsymbol{y}\right) = K\left(X, X\right) + \operatorname{diag}\left(\sigma_{n,1}^2, \ldots, \sigma_{n,n}^2\right), \tag{2.29}$$

where $\operatorname{diag}(\sigma_{n,1}^2, \ldots, \sigma_{n,n}^2)$ denotes a diagonal matrix with the individual noise variances along the diagonal. We can again write the joint prior of the observations and the function values at the test points,

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{f}^* \end{bmatrix} \sim \mathcal{N}\left(\mathbf{0}, \begin{bmatrix} K(X,X) + \operatorname{diag}\left(\sigma_{n,1}^2, \ldots, \sigma_{n,n}^2\right) & K\left(X, X^*\right) \\ K\left(X^*, X\right) & K\left(X^*, X^*\right) \end{bmatrix}\right). \tag{2.30}$$

Analogously to Equation 2.27, we finally arrive at the relevant conditional

$$\begin{aligned} &\boldsymbol{f}^* \mid X, X^*, \boldsymbol{y} \sim \mathcal{N}\left(\boldsymbol{\mu}^*, \Sigma^*\right), \\ &\boldsymbol{\mu}^* = K\left(X^*, X\right)\left[K(X,X) + \operatorname{diag}\left(\sigma_{n,1}^2, \ldots, \sigma_{n,n}^2\right)\right]^{-1} \boldsymbol{y} \\ &\Sigma^* = K\left(X^*, X^*\right) - K\left(X^*, X\right)\left[K(X,X) + \operatorname{diag}\left(\sigma_{n,1}^2, \ldots, \sigma_{n,n}^2\right)\right]^{-1} K\left(X, X^*\right), \end{aligned} \tag{2.31}$$

which constitute the main predictive equations for GPs.

At this point, we make two remarks. First, the noise levels $\sigma_{n,i}^2$ are typically not known in practice and are therefore often treated as additional hyperparameters, along with the kernel's lengthscale ℓ and signal variance σ_f^2. These parameters are then often selected by optimizing a measure of model fit to the data, most commonly using the marginal likelihood, although other methods like cross-validation can also be employed [11]. In this work, we will not explicitly rely on or analyze the marginal likelihood, especially as the noise levels will be known. Second, performing inference with GPs requires the inversion of the training kernel matrix $K(X,X) + \mathrm{diag}\left(\sigma^2_{n,1}, \dots, \sigma^2_{n,n}\right)$. This step is the primary computational bottleneck, as the inversion scales cubically with the number of training points. In practice, direct inversion is avoided in favor of a Cholesky decomposition

$$K(X,X) + \mathrm{diag}\left(\sigma^2_{n,1}, \dots, \sigma^2_{n,n}\right) = LL^\top, \tag{2.32}$$

where L is a lower-triangular matrix, followed by triangular solves

$$\begin{aligned} Lz = \boldsymbol{y}, \quad L^\top \boldsymbol{\alpha} = z \quad &\Rightarrow \quad \boldsymbol{\mu}^* = K(X^*, X)\,\boldsymbol{\alpha}. \\ L\boldsymbol{v} = K(X, X^*) \quad &\Rightarrow \quad \Sigma^* = K(X^*, X^*) - \boldsymbol{v}^\top \boldsymbol{v}. \end{aligned} \tag{2.33}$$

While this still scales as $\mathcal{O}(n^3)$, it is numerically more stable. To further ensure robustness (especially when the kernel matrix is ill-conditioned due to closely spaced inputs), a small constant is added to the diagonal of the matrix before the decomposition.

2.1.6 Gradient-Enhanced Gaussian Processes

In some applications, we not only have access to function values, but also to function gradients. This additional data offers an opportunity to further constrain the posterior and can significantly improve the accuracy of GP models, especially in high-dimensional settings with a limited number of datapoints [34]. This gradient-enhanced GP regression, also called gradient-enhanced Kriging (GEK) exploits the linearity of the covariance function to incorporate derivative information into the GP framework we already built up.

Concretely, if we model the unknown function f as a GP with covariance function $K(x, x')$, then the partial derivatives $\frac{\partial f}{\partial x_i}$ are also GPs. Moreover, the cross-covariances between function values and derivatives, as well as between different derivatives, can be computed analytically by differentiating the kernel function. Specifically, if $f \sim \mathcal{GP}(0, K(x, x'))$, then the covariance between the function value at x and the partial derivative with respect to the i-th input dimension at x' is given by

$$\begin{aligned} \mathrm{Cov}\left(f(x), \frac{\partial f(x')}{\partial x'_i}\right) &= \mathbb{E}\left[(f(x) - \mathbb{E}[f(x)])(\frac{\partial f(x')}{\partial x'_i} - \mathbb{E}[\frac{\partial f(x')}{\partial x'_i}])\right] \\ &= \mathbb{E}\left[(f(x) - \mathbb{E}[f(x)])(\frac{\partial f(x')}{\partial x'_i} - \frac{\partial}{\partial x'_i}\mathbb{E}[f(x')])\right] \\ &= \mathbb{E}\left[\frac{\partial}{\partial x'_i}(f(x) - \mathbb{E}[f(x)])(f(x') - \mathbb{E}[f(x')])\right] \\ &= \frac{\partial K(x, x')}{\partial x'_i}, \end{aligned} \tag{2.34}$$

where we used the definition of the covariance function in Equation 2.23 and the linearity of the expectation operator. Similarly, the covariance between partial derivatives is

$$\operatorname{Cov}\left(\frac{\partial f(x)}{\partial x_i}, \frac{\partial f(x')}{\partial x'_j}\right) = \frac{\partial^2 K(x,x')}{\partial x_i \partial x'_j}. \tag{2.35}$$

To make this more concrete, consider again the RBF kernel. The required derivatives are

$$\frac{\partial K(x,x')}{\partial x'_i} = \sigma_f^2 \exp\left(-\frac{\|x-x'\|^2}{2\ell^2}\right) \cdot \frac{x_i - x'_i}{\ell^2}, \tag{2.36}$$

$$\frac{\partial^2 K(x,x')}{\partial x_i \partial x' j} = \sigma_f^2 \exp\left(-\frac{\|x-x'\|^2}{2\ell^2}\right) \cdot \left[\frac{\delta_{i,j}}{\ell^2} - \frac{(x_i - x'_i)(x_j - x'_j)}{\ell^4}\right]. \tag{2.37}$$

Now suppose we have training data consisting of both function evaluations $(x_i, y_i)_{i=1}^n$ and gradient observations $(x_j, \boldsymbol{g}_j)_{j=1}^m$, where $\boldsymbol{g}_j = \nabla f(x_j) + \boldsymbol{\varepsilon}_j$ with $\boldsymbol{\varepsilon}_j \sim \mathcal{N}(\mathbf{0}, \operatorname{diag}\left(\sigma^2_{g,j,1}, \ldots, \sigma^2_{g,j,d}\right)$ representing gradient noise, heteroscedastic in each component. We can combine all observations into a single vector

$$\boldsymbol{y}_{\text{aug}} = \begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{g}_1 \\ \vdots \\ \boldsymbol{g}_m \end{bmatrix}, \tag{2.38}$$

where $\boldsymbol{y} = [y_1, \ldots, y_n]^\top$ contains the function evaluations and each $\boldsymbol{g}_j \in \mathbb{R}^d$ is the gradient observation at location x_j. An augmented kernel matrix K_{aug} is constructed by evaluating the appropriate kernel derivatives between all pairs of observations. This matrix has the block structure

$$K_{\text{aug}} = \begin{bmatrix} K_{ff} & K_{fg} \\ K_{gf} & K_{gg} \end{bmatrix}, \tag{2.39}$$

where K_{ff} is the standard $n \times n$ kernel matrix between function observations, K_{fg} and K_{gf} contains the cross-covariances between function and gradient observations, and K_{gg} contains the covariances between gradient observations. The noise term is incorporated analogously to before, yielding

$$\operatorname{Cov}(\boldsymbol{y}_{\text{aug}}) = K_{\text{aug}} + \underbrace{\begin{bmatrix} \operatorname{diag}\left(\sigma^2_{n,1}, \ldots, \sigma^2_{n,n}\right) & 0_{n \times md} \\ 0_{md \times n} & \operatorname{diag}\left(\sigma^2_{g,1,1}, \ldots, \sigma^2_{g,m,d}\right) \end{bmatrix}}_{\Sigma_{\text{noise}}}. \tag{2.40}$$

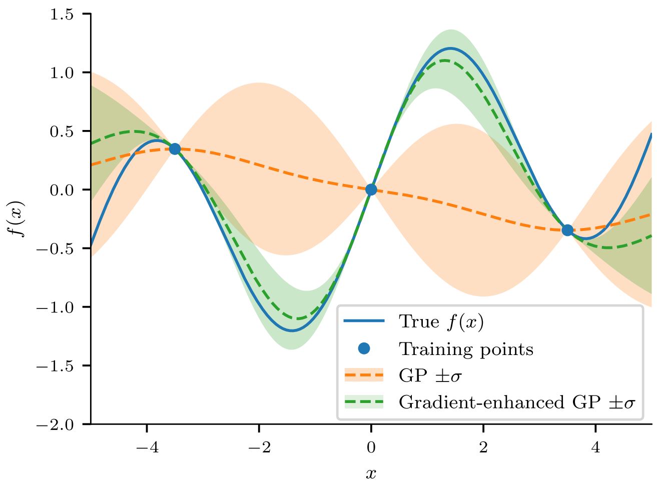

Figure 2.2: Comparison of standard Gaussian process (GP) regression and gradient-enhanced GP regression. The true function $f(x)$ (solid blue line) is approximated using only three training points (blue dots) with zero noise. The standard GP posterior (orange dashed line with orange shaded uncertainty) does not capture the right function behavior and exhibits large uncertainty regions between training points. In contrast, the gradient-enhanced GP (green dashed line with green shaded uncertainty) has a significantly lower posterior uncertainty and provides markedly more accurate predictions throughout the domain.

The predictive equations for gradient-enhanced GPs follow the same form as standard GPs, but now the conditioning data includes both function and gradient observations. For predictions at test points X^*, we obtain

$$\begin{aligned}
&\boldsymbol{f}^* \mid X^*, X_{\text{aug}}, \boldsymbol{y}_{\text{aug}} \sim \mathcal{N}\left(\boldsymbol{\mu}^*, \Sigma^*\right), \\
&\boldsymbol{\mu}^* = K_{\text{aug}}(X^*, X_{\text{aug}}) \left[K_{\text{aug}}(X_{\text{aug}}, X_{\text{aug}}) + \Sigma_{\text{noise}}\right]^{-1} \boldsymbol{y}_{\text{aug}}, \\
&\Sigma^* = K(X^*, X^*) - K_{\text{aug}}(X^*, X_{\text{aug}}) \left[K_{\text{aug}}(X_{\text{aug}}, X_{\text{aug}}) + \Sigma_{\text{noise}}\right]^{-1} K_{\text{aug}}(X_{\text{aug}}, X^*),
\end{aligned} \tag{2.41}$$

where $K_{\text{aug}}(X^*, X_{\text{aug}})$ contains the covariances between test points and all training observations (both function and gradient evaluations). Gradient-enhanced GPs are especially useful in scenarios where gradient information is readily available at negligible additional cost, such as when using automatic differentiation. However, the total computational cost does increases significantly as the effective size of the kernel matrix grows from $n \times n$ to $(n + md) \times (n + md)$, where m is the number of gradient observations and d is the input dimensionality, which leads to a $\mathcal{O}((n + md)^3)$ scaling (matrix inversion or Cholesky decomposition). The stark contrast between a GP trained only on function observations and a GP trained on both function and gradient observations is illustrated in Figure 2.2.

2.2 Variational Monte Carlo

2.2.1 The Variational Principle

Here we briefly summarize the theory of variational Monte Carlo (VMC). The goal of VMC is to find the ground state energy E_0 and wavefunction $|\Psi_0\rangle$ for some quantum Hamiltonian $\hat{H}$, i.e., to find the lowest energy eigenstate of the Schrödinger equation $\hat{H}|\Psi\rangle = E|\Psi\rangle$. The workhorse of any variational method is the variational principle for the energy functional $E[\cdot]$ of a generic wavefunction $|\Psi\rangle$:

$$E[|\Psi\rangle] \equiv \frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle} = \frac{\sum_n |c_n|^2 E_n}{\langle\Psi|\Psi\rangle} \geq \frac{\sum_n |c_n|^2 E_0}{\langle\Psi|\Psi\rangle} = E_0, \tag{2.42}$$

where $c_n = \langle\Psi_n|\Psi\rangle$ and we used the resolution of the identity $\mathbb{1} = \sum_n |\Psi_n\rangle\langle\Psi_n|$. This establishes that

$$E_0 = \min_{|\Psi\rangle\in\mathcal{H}} E[|\Psi\rangle] \qquad |\Psi_0\rangle = \arg\min_{|\Psi\rangle\in\mathcal{H}} E[|\Psi\rangle], \tag{2.43}$$

where $\mathcal{H}$ is the associated Hilbert space. This optimization is clearly intractable for the high- and infinite-dimensional Hilbert spaces of many-body systems. We therefore dramatically reduce the size of the search space by introducing a parametrization (ansatz) $|\Psi(\boldsymbol{\theta})\rangle$, where $\boldsymbol{\theta}$ is a vector of real (or sometimes complex) parameters. The variational problem then becomes:

$$\tilde{E}_0 = \min_{\boldsymbol{\theta}} E[|\Psi(\boldsymbol{\theta})\rangle] \qquad \boldsymbol{\theta}_0 = \arg\min_{\boldsymbol{\theta}} E[|\Psi(\boldsymbol{\theta})\rangle], \tag{2.44}$$

where $\tilde{E}_0 \geq E_0$ is the variational estimate of the ground state energy and similarly $|\Psi(\boldsymbol{\theta}_0)\rangle \approx |\Psi_0\rangle$. Of course, the quality of this approximation critically depends on the choice of ansatz and whether the true ground state lies within or close to the search space manifold.

Because we have effectively turned the functional $E[|\Psi\rangle]$ into a function $E[\boldsymbol{\theta}] \equiv E[|\Psi(\boldsymbol{\theta})\rangle]$, the parameter optimization problem in Equation 2.44 can, in principle, be solved using standard numerical optimization techniques such as gradient descent (a more detailed review of optimization methods is provided in the following section). However, to access the energy $E[\boldsymbol{\theta}]$, we would need to evaluate

$$E[\boldsymbol{\theta}] = \frac{\int \mathrm{d}\xi\, \Psi^*(\xi;\boldsymbol{\theta})\hat{H}\Psi(\xi;\boldsymbol{\theta})}{\int \mathrm{d}\boldsymbol{\xi}\, |\Psi(\boldsymbol{\xi};\boldsymbol{\theta})|^2}, \tag{2.45}$$

where $\Psi(\boldsymbol{\xi};\boldsymbol{\theta}) = \langle\boldsymbol{\xi}|\Psi(\boldsymbol{\theta})\rangle$, with $|\boldsymbol{\xi}\rangle = |\xi_1, \xi_2, \ldots\rangle$ collectively representing all relevant degrees of freedom (positions, spins, etc.), and $\mathrm{d}\xi$ denoting the appropriate measure over these degrees of freedom (integration over continuous variables and summation over discrete ones).

2.2.2 Stochastic Estimation of the Variational Energy

Even for simple ansätze, this integral is still typically intractable for direct numerical evaluation due to the exponential scaling of the space of all configurations with system size. In other words, while the parametrization reduces the optimization problem to a low-dimensional space, we are still burdened by the full Hilbert space when evaluating the energy! The key idea of VMC is to realize that one can rewrite $E[\boldsymbol{\theta}]$ as

$$E[\boldsymbol{\theta}] = \frac{\int \mathrm{d}\xi\, \Psi^*(\xi;\boldsymbol{\theta})\hat{H}\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}{\int \mathrm{d}\boldsymbol{\xi}\, |\Psi(\boldsymbol{\xi};\boldsymbol{\theta})|^2} = \frac{\int \mathrm{d}\xi\, |\Psi^*(\xi;\boldsymbol{\theta})|^2 \frac{\hat{H}\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}{\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}}{\int \mathrm{d}\boldsymbol{\xi}\, |\Psi(\boldsymbol{\xi};\boldsymbol{\theta})|^2} = \mathbb{E}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}\left[E_L(\boldsymbol{\xi})\right], \tag{2.46}$$

where we defined the normalized wavefunction density $\rho_{\boldsymbol{\theta}} = \frac{|\Psi^*(\xi;\boldsymbol{\theta})|^2}{\int \mathrm{d}\boldsymbol{\xi}\,|\Psi(\boldsymbol{\xi};\boldsymbol{\theta})|^2}$ and the local energy $E_L(\boldsymbol{\xi}) = \frac{\hat{H}\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}{\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}$, $\mathbb{E}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}$ denoting the expectation value taken with respect to the probability density $\rho_{\boldsymbol{\theta}}$. In practice, given a finite set of samples $\{\boldsymbol{\xi}_i\}_{i=1}^N$ drawn i.i.d. from $\rho_{\boldsymbol{\theta}}$, the variational energy can be approximated as

$$E[\boldsymbol{\theta}] \approx \frac{1}{N}\sum_{i=1}^{N} E_L(\boldsymbol{\xi}_i), \tag{2.47}$$

with the error scaling as $\mathcal{O}(1/\sqrt{N})$, independent of the dimensionality of the configuration space. This is the notable advantage of Monte Carlo methods over traditional deterministic grid sampling methods such as Simpson's rule, whose error increases exponentially with dimensionality at constant N [35].

2.2.3 Monte Carlo Sampling and Variance Reduction Strategies

However, this stochastic reformulation introduces a new challenge: sampling from the high-dimensional distribution $\rho_{\boldsymbol{\theta}}$ which typically does not have a closed form is non-trivial as standard techniques like inverse transform sampling or the acceptance-rejection method are inapplicable or prohibitively inefficient. That is why in the context of VMC, Monte Carlo sampling is virtually synonymous with Markov Chain Monte Carlo (MCMC), where a Markov chain (a discrete-in-time and memoryless stochastic process) is constructed whose stationary distribution is $\rho_{\boldsymbol{\theta}}$. The most common approach is the Metropolis–Hastings algorithm [36, 37], which generates a sequence of configurations (often called "walker" in this context) by proposing local updates to the current sample and accepting or rejecting these updates based on a detailed balance condition [2]. Concretely, a trajectory of successive configurations $\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_N$ is constructed by iterating over two steps [38]:

1. Starting from an initial configuration $\boldsymbol{\xi}_i$, generated a new candidate configuration $\boldsymbol{\xi}_f$ from a proposal distribution $\alpha(\boldsymbol{\xi}_f|\boldsymbol{\xi}_i)$.

2. Accept the candidate if $\min\left(1, \frac{\rho_{\boldsymbol{\theta}}(\boldsymbol{\xi}_f)\alpha(\boldsymbol{\xi}_i|\boldsymbol{\xi}_f)}{\rho_{\boldsymbol{\theta}}(\boldsymbol{\xi}_i)\alpha(\boldsymbol{\xi}_f|\boldsymbol{\xi}_i)}\right)$ is larger than a randomly generated number between 0 and 1 (uniform distribution), otherwise remain at $\boldsymbol{\xi}_i$ for another iteration.[2]

The algorithm ensures convergence to the probability distribution $\rho_{\boldsymbol{\theta}}$ provided that any configuration $\boldsymbol{\xi}$ is reachable from any initial $\boldsymbol{\xi}_f$ within a finite number of transitions, i.e. the stochastic process must be ergodic. This in turn depends on the choice of proposal distribution. Furthermore, note that asymptotic convergence in the limit of many steps does not guarantee that individual samples are independent of each other. In fact, consecutive samples are by construction correlated with each other, which can lead to prolonged autocorrelation times and thus to statistically biased estimates. This autocorrelation time again is a function of the choice of proposal distribution and can lead to the optimizer getting stuck in "local modes". As an illustrative example, when configurations $\boldsymbol{\xi}$ with small $\rho_{\boldsymbol{\theta}}(\boldsymbol{\xi})$ are proposed with a disproportionately high probability, the process remains in the same configuration. Although the theory puts minimal restrictions on the proposal distribution, the construction of distributions that minimize rejections remain challenging and part of a longstanding line of research [39, 4, 40] that has also led to a manifold of rejection-free alternatives [38, 41, 42]. The former is especially difficult in the context of VMC because the target distribution $\rho_{\boldsymbol{\theta}}$ changes at each iteration as the parameters $\boldsymbol{\theta}$ are updated.

We remark at this point that if the trial wavefunction is an exact eigenstate of the Hamiltonian, the local energy is constant, $E_L(\boldsymbol{\xi}) = E_n$, and the variance of the energy estimator in Equation 2.47 vanishes exactly. Webber and Lindsey recently made this vanishing-variance principle rigorous, proving that both energy and gradient estimators converge to zero variance as the wavefunction approaches an eigenstate [6]. They do also note, however, that away from an eigenstate, the variance can vary drastically, leading to unstable optimization. Following Webber and Lindsey, we mention specifically two strategies for variance reduction that will be used in the experimental part of this thesis and that have been used with great success in recent influential works [9, 8]. The first strategy is to run multiple MCMC samplers at the same time and combine the samples rather than running one sampler for a long time. This allows both vectorizing the code and parallelizing it across hardware, such as multiple cores or even different compute nodes. The second, called parallel tempering, introduces interacting samplers targeting different distributions

$$\rho_{\boldsymbol{\theta},i}(\boldsymbol{\xi}) \propto \rho_{\boldsymbol{\theta}}(\boldsymbol{\xi})^{i/m} \qquad i = 0, 1, \ldots, m. \tag{2.48}$$

Periodically, samplers targeting adjacent distributions $\rho_{\boldsymbol{\theta},i}$ and $\rho_{\boldsymbol{\theta},i+1}$ attempt to exchange their current configurations $\boldsymbol{\xi}_i$ and $\boldsymbol{\xi}_{i+1}$, respectively. These

[2]Note that the prohibitively expensive normalization factor $\int \, \mathrm{d}\boldsymbol{\xi} \, |\Psi(\boldsymbol{\xi};\boldsymbol{\theta})|^2$ in $\rho_{\boldsymbol{\theta}}$ cancels out.

swaps are accepted if the Metropolis-type acceptance probability

$$\min\left(1, \frac{\rho_{\theta,i}(\xi_{i+1})\rho_{\theta,i+1}(\xi_i)}{\rho_{\theta,i}(\xi_i)\rho_{\theta,i+1}(\xi_{i+1})}\right) = \min\left(1, (\frac{\rho_\theta(\xi_i)}{\rho_\theta(\xi_{i+1})})^{1/m}\right) \tag{2.49}$$

is larger than a randomly generated number between 0 and 1.

Samplers with large i, also referred to as "low temperature", closely approximate the target distribution $\rho(\sigma)$, but may lead to the problems described above, like getting stuck in local modes. In contrast, samplers targeting distributions with small i ("high temperature") sample flattened distributions that increase the likelihood of escaping from local modes. The ratio in Equation 2.49 favors swaps when $\rho_\theta(\xi_i) > \rho_\theta(\xi_{i+1})$, allowing high-probability configurations from flatter, high-temperature samples to move into lower-temperature ones. Conversely, low-probability configurations in low-temperature samplers can escape to high-temperature chains, where they are more likely to move freely. These mechanisms help the emergent random process to mix faster, i.e., approach the target distribution faster. Concretely, Webber and Lindsey demonstrate that parallel tempering reduces energy spike recovery times during VMC optimization [6].

2.3 Optimization

Now we give a brief overview of different optimization methods for VMC. The task of minimizing a continuous objective function $f(\boldsymbol{x})$ on some domain $A \subseteq \mathbb{R}^D$ is ubiquitous in virtually all areas of science and engineering. For very small D or A with small volume, one might just evaluate $f(\boldsymbol{x})$ at a number of points on an equidistant (or more carefully designed [43]) grid and pick the minimum "manually", as is, e.g., often done in the optimization of the hyperparameters of machine learning models [25, 44]. In most cases, however, such brute force methods are not feasible. The actual workhorse of continuous optimization is gradient descent and its variants, which is commonly introduced with a simple idea: for any $\boldsymbol{x}$, the gradient $\nabla f(\boldsymbol{x})$ points in the direction of steepest ascent. If we continuously follow the negative gradient, i.e., the direction of steepest descent, we should intuitively eventually end up at an $\boldsymbol{x}$ with $\nabla f(\boldsymbol{x}) = 0$ and therefore a minimum or saddle point. Formally, one iteratively updates the parameters as

$$\boldsymbol{x}_{i+1} = \boldsymbol{x}_i - \eta \nabla f(\boldsymbol{x}_i), \tag{2.50}$$

where η is called the step size (or learning rate), which has to be chosen sufficiently small to not "overstep" the optimum. To gain a deeper understanding of gradient descent and methods built on top of it, we shall reintroduce it from a different angle

For this purpose, we now go back to the VMC optimization problem specifically. Closely following Ref. [6], consider a small parameter update δ to a fixed set of parameters $\boldsymbol{\theta}$ and the so-called intermediate normalization, used, e.g., in perturbation theory [3] and coupled cluster theory [45],

$$|\tilde{\Psi}(\boldsymbol{\theta}+\delta)\rangle = \frac{\langle\Psi(\boldsymbol{\theta})|\Psi(\boldsymbol{\theta})\rangle}{\langle\Psi(\boldsymbol{\theta})|\Psi(\boldsymbol{\theta}+\delta)\rangle}\,|\Psi(\boldsymbol{\theta}+\delta)\rangle \tag{2.51}$$

such that the inner product with $|\Psi(\boldsymbol{\theta})\rangle$ is constant. Now, we Taylor-expand in δ to second order:

$$|\tilde{\Psi}(\boldsymbol{\theta}+\delta)\rangle = |\tilde{\Psi}\rangle + \sum_i \delta_i\,|\tilde{\Psi}_i\rangle + \frac{1}{2}\sum_{i,j}\delta_i\delta_j\,|\tilde{\Psi}_{ij}\rangle + \mathcal{O}\left(|\delta|^3\right), \tag{2.52}$$

with shorthand notation $|\tilde{\Psi}\rangle = |\tilde{\Psi}(\boldsymbol{\theta})\rangle\,, |\tilde{\Psi}_i\rangle = \frac{\partial}{\partial\theta_i}\,|\tilde{\Psi}(\boldsymbol{\theta})\rangle\Big|_{\boldsymbol{\theta}}$, and $|\tilde{\Psi}_{ij}\rangle = \frac{\partial^2}{\partial\theta_i\partial\theta_j}\,|\tilde{\Psi}(\boldsymbol{\theta})\rangle\Big|_{\boldsymbol{\theta}}$. We then define the energy-shifted Hamiltonian

$$\hat{H}' \equiv \hat{H} - E[\tilde{\Psi}], \tag{2.53}$$

such that

$$E[\tilde{\Psi}(\boldsymbol{\theta}+\delta)] - E[\tilde{\Psi}] = \frac{\langle\varphi|\hat{H}'|\varphi\rangle}{\langle\varphi|\varphi\rangle}, \tag{2.54}$$

where we defined $|\varphi\rangle \equiv |\tilde{\Psi}(\boldsymbol{\theta}+\delta)\rangle$ to reduce notational clutter and we used the energy functional as given in Equation 2.45. Inserting the expansion from Equation 2.52 into the numerator on the RHS of Equation 2.54 yields

$$\begin{aligned}\langle\varphi|\hat{H}'|\varphi\rangle = \langle\tilde{\Psi}|\hat{H}'|\tilde{\Psi}\rangle &+ \sum_i \delta_i\left(\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}\rangle + \langle\tilde{\Psi}|\hat{H}'|\tilde{\Psi}_i\rangle\right)\\ &+ \sum_{i,j}\delta_i\delta_j\left[\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}_j\rangle + \frac{1}{2}\langle\tilde{\Psi}_{ij}|\hat{H}'|\tilde{\Psi}\rangle + \frac{1}{2}\langle\tilde{\Psi}|\hat{H}'|\tilde{\Psi}_{ij}\rangle\right] + \mathcal{O}(|\delta|^3).\end{aligned} \tag{2.55}$$

The zeroth-order term vanishes by construction of $\hat{H}'$ and we can exploit the identity $z + z^* = 2\Re(z)$ to simplify to

$$\langle\varphi|\hat{H}'|\varphi\rangle = 2\Re\left(\sum_i \delta_i\,\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}\rangle\right) + \sum_{ij}\delta_i\delta_j\left[\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}_j\rangle + \Re\left(\langle\tilde{\Psi}_{ij}|\hat{H}'|\tilde{\Psi}\rangle\right)\right] + \mathcal{O}(|\delta|^3). \tag{2.56}$$

Analogously, we can derive for the denominator

$$\langle\varphi|\varphi\rangle = \langle\tilde{\Psi}|\tilde{\Psi}\rangle + 2\Re\left(\sum_i \delta_i\,\langle\tilde{\Psi}_i|\tilde{\Psi}\rangle\right) + \sum_{ij}\delta_i\delta_j\left[\langle\tilde{\Psi}_i|\tilde{\Psi}_j\rangle + \Re\left(\langle\tilde{\Psi}_{ij}|\tilde{\Psi}\rangle\right)\right] + \mathcal{O}(|\delta|^3). \tag{2.57}$$

Because the intermediate normalization condition, we have $\langle\tilde{\Psi}|\tilde{\Psi}(\theta+\delta)\rangle = \langle\tilde{\Psi}|\tilde{\Psi}\rangle$ and by differentiating both sides with respect to δ, $\langle\tilde{\Psi}|\tilde{\Psi}_i\rangle = \langle\tilde{\Psi}_i|\tilde{\Psi}\rangle = 0$. Therefore, the first-order correction to the norm vanishes and we get

$$\langle\varphi|\varphi\rangle = \langle\tilde{\Psi}|\tilde{\Psi}\rangle + \mathcal{O}(|\delta|^2), \tag{2.58}$$

and because $\frac{1}{1+x} = 1 + x + \mathcal{O}(x^2)$, the quadratic term only enters the entire expression in $\mathcal{O}(|\delta|^3)$ and can thus be neglected. We finally obtain

$$E[|\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle] - E[|\tilde{\Psi}\rangle] = \boldsymbol{\delta}^\dagger \boldsymbol{g} + \boldsymbol{g}^\dagger \boldsymbol{\delta} + \boldsymbol{\delta}^\dagger H \boldsymbol{\delta} + \Re\left(\boldsymbol{\delta}^T J \boldsymbol{\delta}\right) + \mathcal{O}(|\delta|^3), \tag{2.59}$$

where we defined

$$g_i \equiv \frac{\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}\rangle}{\langle\tilde{\Psi}|\tilde{\Psi}\rangle}, \quad H_{ij} \equiv \frac{\langle\tilde{\Psi}_i|\hat{H}'|\tilde{\Psi}_j\rangle}{\langle\tilde{\Psi}|\tilde{\Psi}\rangle}, \quad J_{ij} \equiv \frac{\langle\tilde{\Psi}_{ij}|\hat{H}'|\tilde{\Psi}\rangle}{\langle\tilde{\Psi}|\tilde{\Psi}\rangle}. \tag{2.60}$$

Now the core idea is that instead of "directly" optimizing a highly complex and non-convex function like $E[\Psi(\boldsymbol{\theta})]$, we optimize a low-order surrogate, e.g., if we only include linear terms in Equation 2.59, we could minimize $E[|\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle] \approx E[|\tilde{\Psi}\rangle] + \boldsymbol{\delta}^\dagger \boldsymbol{g} + \boldsymbol{g}^\dagger \boldsymbol{\delta}$, but this would clearly lead to an unbounded solution. We therefore have to put a constraint on $\boldsymbol{\delta}$ by adding a penalty term to the minimization objective to limit to a region where we can trust the surrogate. This could, e.g., yield

$$\min_{\boldsymbol{\delta}} \boldsymbol{\delta}^\dagger \boldsymbol{g} + \boldsymbol{g}^\dagger \boldsymbol{\delta} + \lambda \|\boldsymbol{\delta}\|^2, \tag{2.61}$$

which has the solution $\boldsymbol{\delta} = \lambda^{-1}\boldsymbol{g}$ for some tunable parameter λ. This is exactly gradient descent as presented in Equation 2.50 with $\eta = \lambda^{-1}$. One could alternatively use the angle between wavefunctions

$$\angle\left(|\tilde{\Psi}(\boldsymbol{\theta})\rangle, |\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle\right) = \arccos\left(\frac{|\langle\tilde{\Psi}(\boldsymbol{\theta})|\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle|}{\||\tilde{\Psi}(\boldsymbol{\theta})\rangle\| \, \||\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle\|}\right) \tag{2.62}$$

for the penalty term. One can approximate this angle as

$$\angle\left(|\tilde{\Psi}(\boldsymbol{\theta})\rangle, |\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle\right)^2 = \boldsymbol{\delta}^\dagger S \boldsymbol{\delta} + \mathcal{O}\left(|\delta|^3\right), \tag{2.63}$$

where $S_{ij} = \frac{\langle\tilde{\Psi}_i|\tilde{\Psi}_j\rangle}{\langle\tilde{\Psi}|\tilde{\Psi}\rangle}$ is called quantum Fisher information matrix, Fubini study metric tensor, or quantum geomtric tensor [6, 23, 46]. The emerging penalty term

$$\lambda \boldsymbol{\delta}^\dagger (S + \varepsilon I) \boldsymbol{\delta}, \tag{2.64}$$

where εI for $\varepsilon > 0$ is added to make the metric tensor positive definite, leads to a modified gradient descent called (quantum) natural gradient descent or stochastic reconfiguration (SR) [6, 23, 8]. The update term then turns into $\boldsymbol{\delta} = -\eta(S+\varepsilon I)^{-1}\boldsymbol{g}$.

Webber and Lindsey then go on to additionally include the first second order correction term in Equation 2.59, $\boldsymbol{\delta}^\dagger H \boldsymbol{\delta}$, to make the surrogate more accurate by including some curvature information. They combine this with the penalty term from SR to obtain the Rayleigh-Gauss-Newton (RGN) method, which has update $\boldsymbol{\delta} = -\left[H + \eta(S+\varepsilon I)\right]^{-1}\boldsymbol{g}$. RGN is closely related to the linear method (LM), where one minimizes an objective for the linearized intermediate-normalized wavefunction $|\tilde{\Psi}(\boldsymbol{\theta}+\boldsymbol{\delta})\rangle = |\tilde{\Psi}\rangle + \sum_i \delta_i |\tilde{\Psi}_i\rangle$:

$$E\left[|\tilde{\Psi}\rangle + \sum_i \delta_i\, |\tilde{\Psi}_i\rangle\right] - E[|\tilde{\Psi}\rangle] = \frac{\boldsymbol{\delta}^\dagger \boldsymbol{g} + \boldsymbol{g}^\dagger \boldsymbol{\delta} + \boldsymbol{\delta}^\dagger H \boldsymbol{\delta}}{1 + \boldsymbol{\delta}^\dagger S \boldsymbol{\delta}}, \tag{2.65}$$

which leads to the generalized eigenvalue problem [6, 47]

$$\begin{pmatrix} 0 & \boldsymbol{g}^\dagger \\ \boldsymbol{g} & H \end{pmatrix} \begin{pmatrix} 1 \\ \boldsymbol{\delta} \end{pmatrix} = \omega \begin{pmatrix} 1 & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} 1 \\ \boldsymbol{\delta} \end{pmatrix}. \tag{2.66}$$

To bridge to the previous section, in VMC we would sample all introduced quantities using [6, 2]

$$\begin{aligned} g_i &= \mathrm{Cov}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}\left[\nu_i(\boldsymbol{\xi}), E_L(\boldsymbol{\xi})\right], \\ S_{ij} &= \mathrm{Cov}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}\left[\nu_i(\boldsymbol{\xi}), \nu_j(\boldsymbol{\xi})\right], \\ H_{ij} &= \mathrm{Cov}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}\left[\nu_i(\boldsymbol{\xi}), E_{L,j}(\boldsymbol{\xi})\right] - g_i\, \mathbb{E}_{\boldsymbol{\xi}\sim\rho_{\boldsymbol{\theta}}}\left[\nu_j(\boldsymbol{\xi})\right] - E[\boldsymbol{\theta}]\, S_{ij}, \end{aligned} \tag{2.67}$$

where $E_{L,i}(\boldsymbol{\xi}) = \frac{\hat{H}\,\partial_{\theta_i}\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}{\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}$ and $\nu_i(\boldsymbol{\xi}) = \frac{\partial_{\theta_i}\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}{\Psi(\boldsymbol{\xi};\boldsymbol{\theta})}$ is called logarithmic derivative. Note that noise accumulates especially in H_{ij} as it involves many multiplications of quantities themselves sampled. As a result, accurate estimation of H_{ij} requires significantly more samples. S and H also often exhibit near-singularities in practice [5], which generally make second-order methods less favorable.

To summarize, all of the methods discussed can be viewed as attempts to minimize a local surrogate model of the energy functional, subject to a constraint that ensures the surrogate remains valid. The differences between them roughly lie in the order of the expansion used (first or second) and the choice of metric or geometry to define the constraint.

2.4 Restricted Variance Optimization with Gradient-Enhanced Kriging

From the previous elaborations, it is relatively straightforward to introduce the core method of this thesis: Restricted Variance Optimization with

Gradient-Enhanced Kriging (RVO-GEK), which was introduced in a similar form by Raggi et al. for molecular geometry optimization [17] and is closely related to Bayesian optimization with gradients [15, 48, 14]. As with the other optimization methods discussed, RVO-GEK can be understood as constructing a surrogate of the objective function and minimizing it while respecting a certain trust condition. RVO-GEK, however, goes beyond these traditional approaches by replacing the fixed, parametric surrogates (e.g., linear or quadratic models) with a non-parametric gradient-enhanced GP model. This also means that instead of just using local information from the current iterate (and potentially the previous iterate, as in momentum-based methods), RVO-GEK is informed by all available data, i.e., the function values and gradients at previous iterations, to yield a smooth surrogate that can captures both local and global curvature more accurately and provides an estimate of its own uncertainty. This means that instead of using a fixed trust region radius or norm constraint, optimization can be restricted to regions where the predictive variance of the surrogate is below a certain threshold $\sigma^2(x) < \varepsilon$. We would like to stress at this point that this surrogate variance is highly informative as it includes both epistemic uncertainty, i,e., uncertainty from lack of data in a given region, and aleatoric uncertainty, i.e., uncertainty in the underlying data (in this case, from VMC) [12].

Concretely, at every iteration, a gradient-enhanced GP is trained on the full set of function values and gradients collected up to and including the current iteration. The known noise levels are incorporated directly into the kernel, as discussed above. The next set of parameters is then found by optimizing the surrogate within the region where the predictive variance remains below the given ε. This optimization is relatively straightforward, as the surrogate is smooth (for kernels like RBF) and analytic derivatives are available. We can therefore use standard optimization methods such as gradient descent or BFGS [49], provided they respect the variance constraint. The specific choice of ε introduces a so-called exploration–exploitation trade-off, commonly encountered in probabilistic machine learning methods [12]: a low ε favors optimization in regions where we are confident that the surrogate is accurate, while a high ε allows exploration of uncertain regions that may contain entirely new areas of lower function value. The full procedure for our use case is summarized in Algorithm 1 and illustrated with a toy example in Figure 2.3.

Algorithm 1 RVO-GEK for Variational Monte Carlo

```
1: Input: Initial parameters $\theta_0$, variance threshold $\varepsilon$, convergence threshold $\delta$, maximum iterations $T$
2: $t \leftarrow 0$
3: while $t < T$ do
4:     Estimate energy $E(\theta_t)$ and gradient $\nabla E(\theta_t)$ using Quantum Monte Carlo
5:     Estimate variances $\sigma_n^2(\theta_t)$ and $\sigma_g^2(\theta_t)$
6:     Train a gradient-enhanced Gaussian Process surrogate model using
7:     $\{E(\theta_i), \nabla E(\theta_i), \sigma_n^2(\theta_i), \sigma_g^2(\theta_i)\}_{i=0}^t$
8:     Solve constrained optimization problem:
           $$\theta_{t+1} = \arg\min_\theta \mu(\theta) \quad \text{s.t.} \quad \sigma^2(\theta) \leq \varepsilon$$
9:     if $\|\theta_{t+1} - \theta_t\| < \delta$ then
10:        break                                      ▷ Convergence reached
11:    end if
12:    $t \leftarrow t + 1$
13: end while
14: Return: Estimated optimal parameters $\theta_t$
```

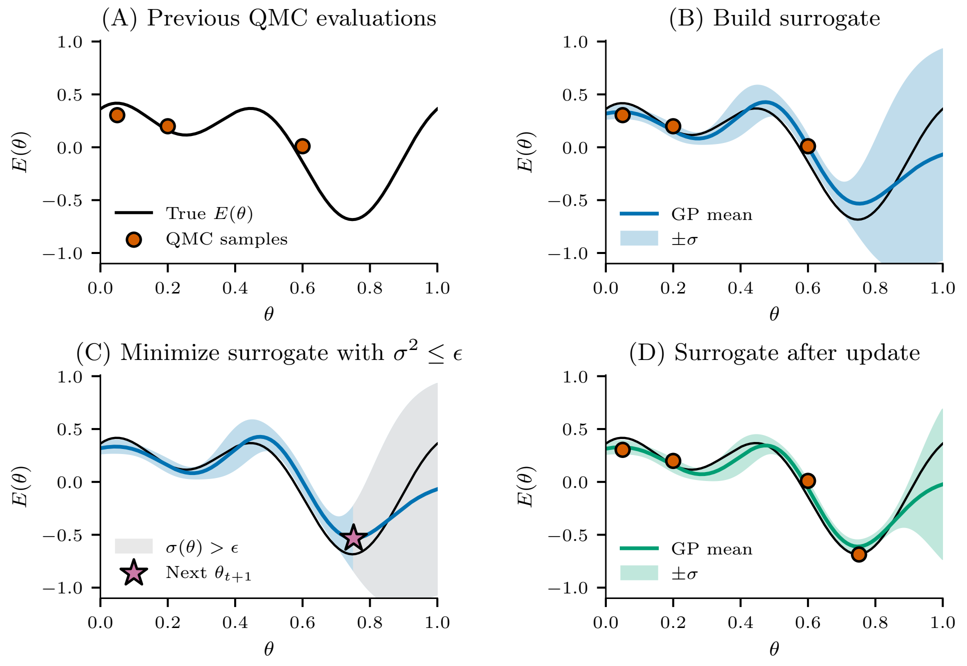

Figure 2.3: Illustration of a RVO-GEK optimization step for minimizing the variational Monte Carlo (VMC) energy as a function of a single variational parameter θ. (A) Initial quantum Monte Carlo (QMC) evaluations of the energy E(θ) (black curve) are shown alongside a set of sampled data points (orange) obtained from previous VMC runs. (B) A Gaussian process (GP) surrogate model is constructed using both energy and gradient information, producing a predictive mean (blue curve) and uncertainty (shaded band, $\pm\sigma$). This surrogate enables efficient exploration of the energy landscape while accounting for both aleatoric and epistemic uncertainty. (C) The acquisition step selects the next candidate parameter θ_{t+1}, shown in magenta, by minimizing the surrogate under a constraint on the model variance, $\sigma^2 \leq \varepsilon$. Regions where uncertainty exceeds this threshold (light gray) are excluded to ensure that the GP surrogate is informative. (D) A new surrogate is constructed with the newly evaluated data point at θ_{t+1}, leading to reduced uncertainty in the surrounding region. The process iterates until convergence is reached.

Chapter 3

Numerical Experiments

For the numerical experiments with our optimization method, we followed Ref. [6] and used the transverse-field Ising (TFI) model on a 1D lattice of d spins with periodic boundary conditions, for which the dimensionless Hamiltonian is given by

$$\hat{H}_{\mathrm{TFI}} = -\sum_{i\sim j} \sigma_i^z \sigma_j^z - h \sum_i \sigma_i^x, \tag{3.1}$$

where σ_i^z and σ_i^x are the respective Pauli operators acting on site i, and the sum $\sum_{i\sim j}$ runs over nearest-neighbor pairs. The first term favors alignment along of spins along the z-axis and the second term introduces quantum fluctuations with a transverse magnetic field of strength h. The model is known to exhibit a quantum phase transition at $h_c = 1$ between an ordered, ferromagnetic phase ($h < 1$) and a disordered, paramagnetic phase ($h > 1$) [24].

As the variational ansatz, we used a restricted Boltzmann machine (RBM), defined as

$$\Psi(\boldsymbol{\xi}; W, \boldsymbol{b}) = \prod_{i=1}^{\alpha} \prod_{\mathcal{T}} \cosh\left(\sum_j w_{ij} (\mathcal{T}\boldsymbol{\xi})_j + b_i\right), \tag{3.2}$$

where $\boldsymbol{\xi} \in \{-1, +1\}^d$ is a spin configuration in the conventional σ^z (computational) basis, w_{ij} and $\boldsymbol{b}_i$ are the variational parameters, α is the number of hidden states, and $\mathcal{T}$ is the set of translation operators on the lattice. An RBM is a type of artifical neural network, which transforms inputs by applying a weight matrix W and bias vector $\boldsymbol{b}$ to project it into a number of hidden units and applying a non-linear function. Most of the time this pattern is repeated in a number of hidden layers to produce some output. The parameters can then be optimized to learn a relationship between inputs and outputs. In this case, there is only one hidden layer with α hidden states h_i. RBMs have been shown to be equivalent to tensor network states under certain conditions [50] and were also used by, inter alia, Caleo and Troyer in their work that spurred the modern development of neural-network-parametrized quantum states [10]. The weights and biases lead to a total of $\alpha(d+1)$ optimizable parameters in our ansatz.

Our numerical implementation of the system, ansatz, MC sampling, and optimization methods that we compare against is based on the implementation by Webber and Lindsey [6, 51] in Python with JAX [52], which enables autoparallelization and just-in-time compilation. We refer to the original work for some additional numerical details. The actual RVO-GEK optimization procedure was implemented from scratch using SciPy [53] and JAX. Although there exist several off-the-shelf libraries for GP regression with support for derivative information, such as SMT [54], GPy [55], and BoTorch [56], we initially did not find a straightforward way to incorporate known heteroscedastic noise levels for both function values and gradients, motivating a custom implementation. We also note that even though there exist frameworks for GPs with complex-valued Kernels [57], we chose to represent the complex parameters using their real and imaginary parts and train a standard real-valued GP. This effectively doubles the number of parameters to $2\alpha(d+1)$.

Due to limitations in the available computational resources within the project's timeframe, we opted to study small systems ($d \leq 12$) for which exact numerical solving is possible. To emulate the effects of MC sampling noise, we add synthetic Gaussian noise to the exact energies and gradients to evaluate the optimization method under realistic conditions. We performed the experiments, unless otherwise noted, with a noise level of $\sigma_n^2 = 1$ and $\sigma_g^2 = 1$ (for all gradient components individually) to test our method in a challenging setting. Furthermore, we used the RBF kernel for all experiments and the number of hidden states was fixed at $\alpha = 5$. All energies are reported per site, i.e., normalized by the number of spins d. The optimization of the surrogate is performed with the *NonlinearConstraint* class in SciPy.

3.1 Analysis of Kernel Parameters

To understand the effect of the RBF kernel parameters lengthscale ℓ and signal variance σ_f^2 on the optimization procedure, we first ran it over a grid of $\sigma_f^2 \in [0.25, 4.0]$ in linear increments and $\ell \in [0.6, 1.8]$ in linear increments. To see potential deviations for different system sizes, we ran this experiment twice, once for $d = 8$ and once for $d = 12$, each with $h = 1.5$ (ferromagnetic) and $\varepsilon = 1$. Each parameter and system setting was evaluated over 10 independent runs. For each setting, the random number generator was re-seeded with a fixed seed (seed = 43) before the first run. The 10 runs

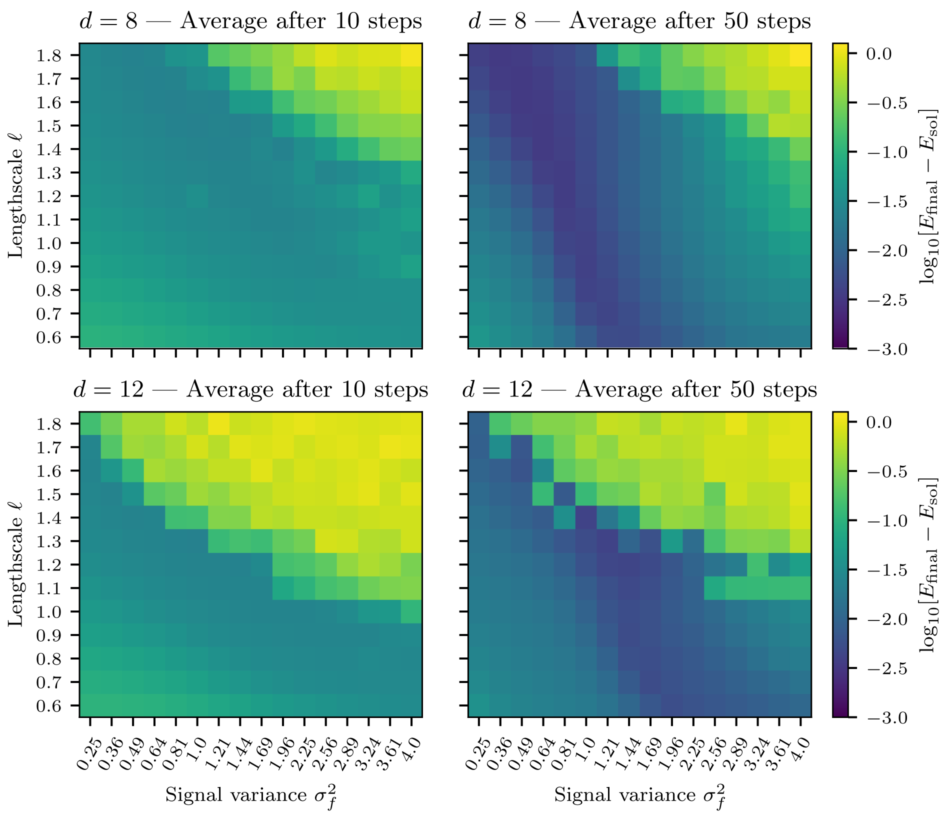

Figure 3.1: Performance of RVO-GEK for different RBF kernel hyperparameters σ_f^2 and ℓ on the $d = 8$ (top row) and $d = 12$ (bottom row) transverse-field Ising model with $h = 1.5$. Each pixel shows the average performance over 10 independent runs. The noise variance was assumed equal for energy and gradients, $\sigma_n^2 = \sigma_g^2 = 1$, and the color indicates the logarithmic error with respect to the exact ground state energy E_{sol}. The left hand side shows the energy after 10 optimization steps, while the right hand side shows the final energy after 50 steps.

then drew from the resulting RNG state in sequence to ensure that runs within each setting were independent but reproducible, and that random samples were comparable across settings. The results are shown in Figure 3.1. Note the logarithmic scale. The results clearly illustrate a dependence of RVO-GEK's performance on the kernel hyperparameters. For both systems, a too large signal variance in combination with a large lengthscale leads to a significantly worse optimization trajectory, even though this is slightly more pronounced in the very early optimization regime ($\sim$ step 10). This is likely due to the fact that a too smooth surrogate can lead to overconfident steps and might not capture the relevant curvature, which would both be detrimental to the overall performance. After 50 steps, in both systems, a relatively wide, but pronounced basin forms with errors close to $10^{-2.5}$. Comparing the small with the larger system, we see that there is a tendency towards smaller lengthscales and signal variances. This is expected, as larger systems

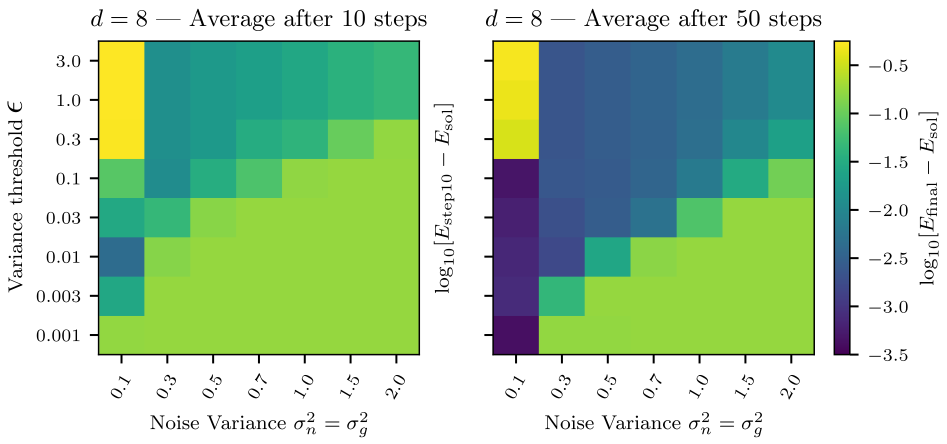

Figure 3.2: Performance of RVO-GEK for different noise levels and variance thresholds on the $d = 8$ transverse-field Ising model with $h = 1.5$. Each pixel shows the average performance over 10 independent runs. The noise variance was assumed equal for energy and gradients, $\sigma_n^2 = \sigma_g^2$, and the color indicates the logarithmic error with respect to the exact ground state energy E_{sol}. The left panel shows the energy after 10 optimization steps, while the right panel shows the final energy after 50 steps. While final performance remains robust across a wide range of settings, too large or too small thresholds ε lead to degrading performance as expected.

exhibit more complex energy landscapes that require a finer resolution in the surrogate model. Overall, while the results suggest that RVO-GEK is fairly robust to the exact choice of kernel parameters in the long run, some tuning of the hyperparameters might be required for rapid convergence. This motivates the exploration of adaptive hyperparameter strategies for setting ℓ and σ_f^2 in future work. Furthermore, we also expect at least some interaction of the kernel parameters with the variance threshold for the optimal overall result.

3.2 Analysis of Variance Threshold

To investigate the relationship between the noise levels σ_n^2, σ_g^2 and the variance threshold ε, we performed another parameter sweep for $\sigma_n^2 = \sigma_g^2 \in [0.1, 2.0]$ in linear increments and $\varepsilon \in [0.001, 3.0]$ in geometric increments for $d = 8$ and $h = 1.5$. The kernel parameters were set to $\sigma_f^2 = 1.0$ and $\ell = 1.2$ using the results from the previous experiments. We then compared the exact solution with the energy after 10 and 50 optimization steps. The results are shown in Figure 3.2. Again, note the logarithmic scale.

We observe a generous corridor of acceptable variance threshold for a given noise level. For example, values between 0.001 and 0.01 for the noise level $\sigma_n^2 = \sigma_g^2 = 0.1$ lead to excellent results for the error almost reaching $10^{-3.5}$ after only 50 steps. Above this corridor, the performance degrades abruptly and significantly as the algorithm becomes too explorative. For higher noise levels, the corridor rises to higher thresholds with a low, exploitative threshold leading to larger errors after 50 steps. Overall, the performance degrades with increasing noise level, as expected, although remarkably slowly with the error going as low as $10^{-1.5}$ for the very high noise level $\sigma_n^2 = \sigma_g^2 = 2.0$. These results indicate that in practice we have substantial flexibility in choosing the variance threshold. Although extreme values should be avoided, a broad range of thresholds leads to fairly good results.

3.3 Comparison With Other Optimization Methods

Finally, we benchmarked RVO-GEK against other popular optimization methods for VMC: gradient descent (GD), stochastic reconfiguration (SR), the linear method (LM), and the Rayleigh-Gauss-Newton (RGN) method as presented in section 2.3. We looked at both $d = 8$ and $d = 12$ with the three regimes $h = 0.5$ (ferromagnetic), $h = 1.0$ (transitional), $h = 1.5$ (paramagnetic). The variance threshold for RVO-GEK was fixed at $\varepsilon = 1$, again using the results from the previous experiment. For the hyperparameters of the other optimizers, such as the learning rate η or regularization parameter ε as defined in Equation 2.64, constant values were chosen that resulted in minimum energy after 50 steps with a manual parameter sweep with $d = 8$ to ensure a fair (if not favorable) comparison against our method. Specifically, we set $\eta_{\text{GD}} = 0.03, \eta_{\text{SR}} = 0.0066, \varepsilon_{\text{SR}} = 0.2, \eta_{\text{LM}} = 0.011, \varepsilon_{\text{LM}} = 0.08, \eta_{\text{RGN}} = 0.0055, \varepsilon_{\text{RGN}} = 0.25$. We note that we found the methods to be highly sensitive to their hyperparameters in these noisy conditions, with most sets of parameters leading to exploding gradients. The results are shown in Figure 3.3.

The results demonstrate a clear advantage of RVO-GEK over the competing optimization methods across all tested regimes, at least in the comparatively early regime of 50 steps. In the ferromagnetic phase, RVO-GEK and GD converge particularly rapidly, with LM following within the 50 steps, while RGN and SR decay rather slowly and do not reach the plateau within 50 steps. In the transitional regime, RVO-GEK maintains fast convergence for both system sizes. SR, LM, RGN are almost indistinguishable (SR performing slightly better), making slow but steady progress. GD performs inconsistently, comparing favorably with RVO-GEK for $d = 8$ while exhibiting significantly higher variance but becoming unstable for $d = 12$ and diverging. In the paramagnetic regime, which presents the most difficult optimization landscape [**Webber**], RVO-GEK again outperforms all other methods in both stability and final energy. SR, LM, and RGN plateau at higher energies, while GD again compares favorably with RVO-GEK for $d = 8$ but diverges for $d = 12$. For $d = 8$ SR performs slightly better than RGN, which in turn is consistently lower in energy than LM. For $d = 12$ RGN now performs best out of the three from step 30 on, with SR slightly higher and LM even going

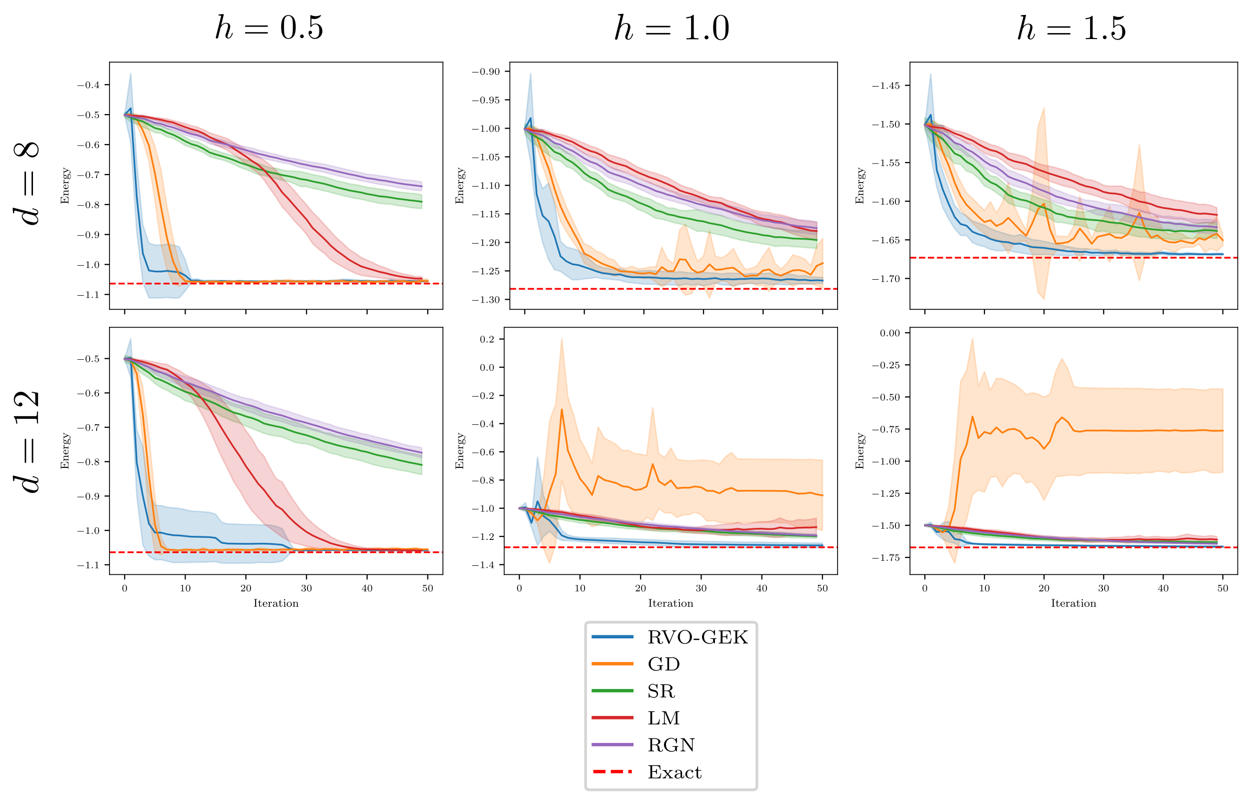

Figure 3.3: Comparison of optimization methods for different system sizes d and transverse field strengths h. Each panel shows the exact energy as a function of the optimization iteration, averaged over 10 different runs, with shaded bands indicating one standard deviation over runs. The red dashed line denotes the exact ground state energy. The new method, RVO-GEK (blue), consistently achieves low energy values across settings and shows rapid convergence, outperforming standard gradient descent (GD, orange), stochastic reconfiguration (SR, green), linear method (LM, red), and the Gauss–Newton method (RGN, purple) in this early optimization regime.

slightly up within the last 10 steps. Overall, RVO-GEK shows consistently strong results across the two system sizes and physical regimes with it being the only method to consistently reach near-exact energies within this setup. However, we do note that RVO-GEK took significantly longer to run, as the outer loop that normally includes the lengthy MC sampling procedure did not have to be performed here. The results indicate that RVO-GEK works well even with very large noise levels, which suggests it may be effective with significantly fewer MC samples in practice.

Chapter 4

Conclusion

In this thesis, we investigated the use of restricted variance optimization with gradient-enhanced Kriging (RVO-GEK) for optimizing variational quantum states in the context of variational Monte Carlo (VMC). To the best of our knowledge, this is the first application of Gaussian Processes (GPs) for VMC optimization directly. We implemented a gradient-enhanced GP that can handle heteroscedastic noise for both function values and gradients, and used it to guide energy minimization of a restricted Boltzmann machine wavefunction for the transverse-field Ising model. Our results demonstrate that RVO-GEK can achieve more sample-efficient convergence than established methods such as gradient descent, stochastic reconfiguration, and even the quasi-second-order Rayleigh-Gauss-Newton method, under high noise conditions.

However, our experiments were restricted to small spin systems with emulated Monte Carlo noise. The applicability of the method to real VMC evaluations and larger (molecular) systems of real interest remains to be tested. Moreover, the cubic scaling with the number of observations and iterations currently presents a strong limitation. Approximate methods such as sparse GPs, random feature expansions, or nested Kriging [58, 59, 60, 61] could help scale the method to longer optimization runs or larger ansätze. Future work should also investigate if using only a constant number of past optimization points instead of all previous points can maintain the observed sample efficiency.

Finally, there is room for improvement on the implementation side. For example, the kernel matrix is currently rebuilt from scratch at each iteration, which introduces unnecessary overhead. Efficient reuse of intermediate computations or incremental (low-rank) updates could significantly reduce total runtime. Overall, this work illustrates the potential of combining machine learning methods with traditional ideas from computational physics. As simulations become more data-driven [62] and optimization-heavy with expressive neural network ansätze, probabilistic learning ideas like the ones presented here could potentially play a larger role.

Bibliography

[1] F. Becca and S. Sorella, *Quantum Monte Carlo Approaches for Correlated Systems*, Nov. 2017. DOI: 10.1017/9781316417041.

[2] B. Rubenstein, "Introduction to the Variational Monte Carlo Method in Quantum Chemistry and Physics," in *Molecular Modeling and Simulation*. Springer Singapore, Dec. 2016, pp. 285–313, ISBN: 9789811025006. DOI: 10.1007/978-981-10-2502-0_10.

[3] J. J. Sakurai and J. Napolitano, *Modern Quantum Mechanics*. Cambridge University Press, Sep. 2020, ISBN: 9781108473224.

[4] J. S. Liu, *Monte Carlo Strategies in Scientific Computing*. Springer Science & Business Media, Jan. 2008, ISBN: 9780387763699.

[5] R. Peng and G. K.-L. Chan, "An Analysis of First- and Second-Order Optimization Algorithms in Variational Monte Carlo," 2025. DOI: 10.48550/ARXIV.2502.19576.

[6] R. J. Webber and M. Lindsey, "Rayleigh-Gauss-Newton optimization with enhanced sampling for variational Monte Carlo," *Physical Review Research*, vol. 4, no. 3, Aug. 2022, ISSN: 2643-1564. DOI: 10.1103/physrevresearch.4.033099.

[7] Y. Song, "Neural Quantum States in Variational Monte Carlo Method: A Brief Summary," 2024. DOI: 10.48550/ARXIV.2406.01017.

[8] C.-Y. Park and M. J. Kastoryano, "Geometry of learning neural quantum states," *Physical Review Research*, vol. 2, no. 2, May 2020, ISSN: 2643-1564. DOI: 10.1103/physrevresearch.2.023232.

[9] K. Choo, G. Carleo, N. Regnault, and T. Neupert, "Symmetries and Many-Body Excitations with Neural-Network Quantum States," *Physical Review Letters*, vol. 121, no. 16, Oct. 2018, ISSN: 0031-9007. DOI: 10.1103/physrevlett.121.167204.

[10] G. Carleo and M. Troyer, "Solving the quantum many-body problem with artificial neural networks," *arXiv*, 2016. DOI: 10.48550/arxiv.1606.02318. eprint: 1606.02318.

[11] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer Verlag, Aug. 2006, ISBN: 9780387310732.

[12] A. Krause and J. Hübotter, "Probabilistic Artificial Intelligence," 2025. DOI: 10.48550/ARXIV.2502.05244.

[13] O. Knill, *Probability Theory And Stochastic Processes With Applications*. Overseas Link Inc, 2009, ISBN: 9788189938406.

[14] H.-S. Chung and J. Alonso, "Using gradients to construct cokriging approximation models for high-dimensional design optimization problems," in *40th AIAA Aerospace Sciences Meeting &amp; Exhibit*, American Institute of Aeronautics and Astronautics, Jan. 2002. DOI: 10.2514/6.2002-317.

[15] J. Wu, M. Poloczek, A. G. Wilson, and P. Frazier, "Bayesian optimization with gradients," in *Advances in Neural Information Processing Systems*, I. Guyon *et al.*, Eds., vol. 30, Curran Associates, Inc., 2017.

[16] M. A. Bouhlel and J. R. R. A. Martins, "Gradient-enhanced kriging for high-dimensional problems," *Engineering with Computers*, vol. 35, no. 1, pp. 157–173, Feb. 2018, ISSN: 0177-0667. DOI: 10.1007/s00366-018-0590-x.

[17] G. Raggi, I. F. Galván, C. L. Ritterhoff, M. Vacher, and R. Lindh, "Restricted-Variance Molecular Geometry Optimization Based on Gradient-Enhanced Kriging," *Journal of Chemical Theory and Computation*, vol. 16, no. 6, pp. 3989–4001, May 2020, ISSN: 1549-9618. DOI: 10.1021/acs.jctc.0c00257.

[18] J. Cui and R. V. Krems, "Gaussian Process Model for Collision Dynamics of Complex Molecules," *Physical Review Letters*, vol. 115, no. 7, Aug. 2015, ISSN: 0031-9007. DOI: 10.1103/physrevlett.115.073202.

[19] A. Denzel and J. Kästner, "Gaussian process regression for geometry optimization," *The Journal of Chemical Physics*, vol. 148, no. 9, Mar. 2018, ISSN: 0021-9606. DOI: 10.1063/1.5017103. [Online]. Available: http://dx.doi.org/10.1063/1.5017103.

[20] R. Meyer and A. W. Hauser, "Geometry optimization using Gaussian process regression in internal coordinate systems," *The Journal of Chemical Physics*, vol. 152, no. 8, Feb. 2020, ISSN: 0021-9606. DOI: 10.1063/1.5144603.

[21] R. Lindh and I. Fdez. Galván, "Molecular structure optimizations with Gaussian process regression," in *Quantum Chemistry in the Age of Machine Learning*. Elsevier, 2023, pp. 391–428, ISBN: 9780323900492. DOI: 10.1016/b978-0-323-90049-2.00017-2.

[22] Y. Rath and G. H. Booth, “Quantum Gaussian process state: A kernel-inspired state with quantum support data,” *Physical Review Research*, vol. 4, no. 2, May 2022, ISSN: 2643-1564. DOI: 10.1103/physrevresearch.4.023126.

[23] J. Stokes, J. Izaac, N. Killoran, and G. Carleo, “Quantum natural gradient,” *Quantum*, vol. 4, p. 269, 2020.

[24] S. Sachdev, *Quantum Phase Transitions*, Apr. 2011. DOI: 10.1017/cbo9780511973765.

[25] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, Nov. 2005. DOI: 10.7551/mitpress/3206.001.0001. [Online]. Available: http://dx.doi.org/10.7551/mitpress/3206.001.0001.

[26] H. König, *Eigenvalue Distribution of Compact Operators*. Birkhäuser, 1986, ISBN: 9783034862806.

[27] B. Ghojogh, A. Ghodsi, F. Karray, and M. Crowley, “Reproducing Kernel Hilbert Space, Mercer’s Theorem, Eigenfunctions, Nyström Method, and Use of Kernels in Machine Learning: Tutorial and Survey,” 2021. DOI: 10.48550/ARXIV.2106.08443.

[28] C.-J. Lin, *Support vector machines and kernel methods: Status and challenges*, Talk at K.U. Leuven Optimization in Engineering Center, Retrieved from https://www.csie.ntu.edu.tw/~cjlin/talks/kuleuven_svm.pdf, Jan. 2013.

[29] M. Seeger, “Gaussian processes for machine learning,” *International journal of neural systems*, vol. 14, no. 02, pp. 69–106, 2004.

[30] R. J. Adler, *The geometry of random fields*. SIAM, 2010.

[31] M. L. Stein, *Interpolation of Spatial Data*. Springer New York, 1999, ISBN: 9781461271666. DOI: 10.1007/978-1-4612-1494-6.

[32] B. Matérn, *Spatial Variation*. Springer New York, 1986, ISBN: 9780387963655. DOI: 10.1007/978-1-4615-7892-5.

[33] E. Porcu, M. Bevilacqua, R. Schaback, and C. J. Oates, “The Matérn Model: A Journey Through Statistics, Numerical Analysis and Machine Learning,” *Statistical Science*, vol. 39, no. 3, Aug. 2024, ISSN: 0883-4237. DOI: 10.1214/24-sts923.

[34] E. Solak, R. Murray-smith, W. Leithead, D. Leith, and C. Rasmussen, “Derivative observations in gaussian process models of dynamic systems,” in *Advances in Neural Information Processing Systems*, S. Becker, S. Thrun, and K. Obermayer, Eds., vol. 15, MIT Press, 2002.

[35] M. H. Kalos and P. A. Whitlock, *Monte Carlo Methods*, Sep. 2008. DOI: 10.1002/9783527626212.

[36] W. K. Hastings, "Monte Carlo sampling methods using Markov chains and their applications," *Biometrika*, vol. 57, no. 1, pp. 97–109, Apr. 1970, ISSN: 1464-3510. DOI: 10.1093/biomet/57.1.97.

[37] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of State Calculations by Fast Computing Machines," *The Journal of Chemical Physics*, vol. 21, no. 6, pp. 1087–1092, Jun. 1953, ISSN: 0021-9606. DOI: 10.1063/1.1699114.

[38] I. Sabzevari and S. Sharma, "Improved Speed and Scaling in Orbital Space Variational Monte Carlo," *Journal of Chemical Theory and Computation*, vol. 14, no. 12, pp. 6276–6286, Nov. 2018, ISSN: 1549-9618. DOI: 10.1021/acs.jctc.8b00780.

[39] P. H. PESKUN, "Optimum Monte-Carlo sampling using Markov chains," *Biometrika*, vol. 60, no. 3, pp. 607–612, 1973, ISSN: 0006-3444. DOI: 10.1093/biomet/60.3.607.

[40] G. Jerfel, S. Wang, C. Fannjiang, K. A. Heller, Y. Ma, and M. I. Jordan, "Variational Refinement for Importance Sampling Using the Forward Kullback-Leibler Divergence," 2021. DOI: 10.48550/ARXIV.2106.15980.

[41] A. Bortz, M. Kalos, and J. Lebowitz, "A new algorithm for Monte Carlo simulation of Ising spin systems," *Journal of Computational Physics*, vol. 17, no. 1, pp. 10–18, Jan. 1975, ISSN: 0021-9991. DOI: 10.1016/0021-9991(75)90060-1.

[42] D. T. Gillespie, "A general method for numerically simulating the stochastic time evolution of coupled chemical reactions," *Journal of Computational Physics*, vol. 22, no. 4, pp. 403–434, Dec. 1976, ISSN: 0021-9991. DOI: 10.1016/0021-9991(76)90041-3.

[43] R. A. Fisher, "Design of Experiments," *BMJ*, vol. 1, no. 3923, pp. 554–554, Mar. 1936, ISSN: 0959-8138. DOI: 10.1136/bmj.1.3923.554-a.

[44] P. Liashchynskyi and P. Liashchynskyi, "Grid Search, Random Search, Genetic Algorithm: A Big Comparison for NAS," 2019. DOI: 10.48550/ARXIV.1912.06059.

[45] Y. A. Aoto, "Geometric interpretation for coupled-cluster theory. A comparison of accuracy with the corresponding configuration interaction model," *The Journal of Chemical Physics*, vol. 157, no. 8, Aug. 2022, ISSN: 0021-9606. DOI: 10.1063/5.0099102.

[46] R. Cheng, "Quantum Geometric Tensor (Fubini-Study Metric) in Simple Quantum System: A pedagogical Introduction," 2010. DOI: 10.48550/ARXIV.1012.1337.

[47] I. Sabzevari, A. Mahajan, and S. Sharma, "An accelerated linear method for optimizing non-linear wavefunctions in variational Monte Carlo," *The Journal of Chemical Physics*, vol. 152, no. 2, Jan. 2020, ISSN: 0021-9606. DOI: 10.1063/1.5125803.

[48] L. Chen, H. Qiu, L. Gao, C. Jiang, and Z. Yang, "Optimization of expensive black-box problems via Gradient-enhanced Kriging," *Computer Methods in Applied Mechanics and Engineering*, vol. 362, p. 112 861, Apr. 2020, ISSN: 0045-7825. DOI: 10.1016/j.cma.2020.112861.

[49] C. G. Broyden, "The convergence of a class of double-rank minimization algorithms 1. general considerations," *IMA Journal of Applied Mathematics*, vol. 6, no. 1, pp. 76–90, 1970.

[50] J. Chen, S. Cheng, H. Xie, L. Wang, and T. Xiang, "Equivalence of restricted boltzmann machines and tensor network states," *Phys. Rev. B*, vol. 97, p. 085 104, 8 Feb. 2018. DOI: 10.1103/PhysRevB.97.085104. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevB.97.085104.

[51] R. J. Webber, *Rjwebber/rgn_optimization*, https://github.com/rjwebber/rgn_optimization, Jun. 2022.

[52] J. Bradbury *et al.*, *JAX: Composable transformations of Python+NumPy programs*, version 0.3.13, 2018. [Online]. Available: http://github.com/jax-ml/jax.

[53] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[54] P. Saves *et al.*, "SMT 2.0: A surrogate modeling toolbox with a focus on hierarchical and mixed variables gaussian processes," *Advances in Engineering Sofware*, vol. 188, p. 103 571, 2024. DOI: https://doi.org/10.1016/j.advengsoft.2023.103571.

[55] GPy, *GPy: A gaussian process framework in python*, http://github.com/SheffieldML/GPy, since 2012.

[56] M. Balandat *et al.*, "BoTorch: A Framework for Efficient Monte-Carlo Bayesian Optimization," in *Advances in Neural Information Processing Systems 33*, 2020.

[57] R. Boloix-Tortosa, J. J. Murillo-Fuentes, F. J. Payan-Somet, and F. Perez-Cruz, "Complex gaussian processes for regression," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, no. 11, pp. 5499–5511, Nov. 2018, ISSN: 2162-2388. DOI: 10.1109/tnnls.2018.2805019.

[58] D. Eriksson, K. Dong, E. Lee, D. Bindel, and A. G. Wilson, "Scaling gaussian process regression with derivatives," *Advances in neural information processing systems*, vol. 31, 2018.

[59] A. Nayebi, A. Munteanu, and M. Poloczek, “A framework for Bayesian optimization in embedded subspaces,” in *Proceedings of the 36th International Conference on Machine Learning*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, PMLR, Sep. 2019, pp. 4752–4761.

[60] J. Quiñonero-Candela and C. E. Rasmussen, “A unifying view of sparse approximate gaussian process regression,” *Journal of Machine Learning Research*, vol. 6, no. 65, pp. 1939–1959, 2005. [Online]. Available: `http://jmlr.org/papers/v6/quinonero-candela05a.html`.

[61] D. Rullière, N. Durrande, F. Bachoc, and C. Chevalier, “Nested Kriging predictions for datasets with a large number of observations,” *Statistics and Computing*, vol. 28, no. 4, pp. 849–867, Jul. 2017, ISSN: 0960-3174. DOI: `10.1007/s11222-017-9766-2`.

[62] D. S. Levine *et al.*, “The Open Molecules 2025 (OMol25) Dataset, Evaluations, and Models,” 2025. DOI: `10.48550/ARXIV.2505.08762`.

Acknowledgment

I would like to thank Prof. Sandeep Sharma for supervising this project and his guidance throughout my time with the group. I also thank Prof. Garnet Chan for facilitating the connection with Prof. Sharma and for a few stimulating discussions. I am also grateful to Prof. Markus Reiher for taking on the internal supervision at ETH.

Furthemore, I would like to thank the whole research group for welcoming me and for their support.

On a personal note, I would like to thank my roommate, Andrew Ivanov, for introducing me to the joy of cycling in the numerous mountains the LA area has to offer.

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every written paper or thesis authored during the course of studies. **In consultation with the supervisor**, one of the following two options must be selected:

[✓] I hereby declare that I authored the work in question independently, i.e. that no one helped me to author it. Suggestions from the supervisor regarding language and content are excepted. I used no generative artificial intelligence technologies[1].

☐ I hereby declare that I authored the work in question independently. In doing so I only used the authorised aids, which included suggestions from the supervisor regarding language and content and generative artificial intelligence technologies. The use of the latter and the respective source declarations proceeded in consultation with the supervisor.

Title of paper or thesis:

Restriced Variance Optimization with Gradient Enhanced Kriging for Variational Monte Carlo

Authored by:
If the work was compiled in a group, the names of all authors are required.

Last name(s):	**First name(s):**
Zimmermann	Yoel

With my signature I confirm the following:
- I have adhered to the rules set out in the Citation Guidelines.
- I have documented all methods, data and processes truthfully and fully.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for originality.

Place, date	**Signature(s)**
Pasadena, 8 July 2025	[illegible]

If the work was compiled in a group, the names of all authors are required. Through their signatures they vouch jointly for the entire content of the written work.

[1] For further information please consult the ETH Zurich websites, e.g. https://ethz.ch/en/the-eth-zurich/education/ai-in-education.html and https://library.ethz.ch/en/researching-and-publishing/scientific-writing-at-eth-zurich.html (subject to change).