EIGENVECTOR CONTINUATION: CONVERGENCE AND EMULATORS

By

Avik Sarkar

A DISSERTATION

Submitted to
Michigan State University
in partial fulfillment of the requirements
for the degree of

Physics – Doctor of Philosophy

2022

ABSTRACT

EIGENVECTOR CONTINUATION: CONVERGENCE AND EMULATORS

By

Avik Sarkar

There has been a great interest in the scientific community in using machine learning to build emulators that can accurately predict scientific processes using only a fraction of the time needed for direct calculations. The computational advantage of emulators allows us to study processes that are beyond what is possible with direct calculations. Eigenvector continuation is one such emulation technique that was introduced recently. It is a variational method that finds the extremal eigenvalues and eigenvectors of a Hamiltonian matrix with one or more control parameters. The computational advantage comes from projecting the Hamiltonian onto a much smaller subspace of basis vectors corresponding to eigenvectors at some chosen training values of the control parameters. The method has proven to be very efficient and accurate for interpolating and extrapolating eigenvectors.

In this work, we present a study on the error convergence properties of eigenvector continuation. With the insights we gain from learning the convergence properties, we then propose a self-learning algorithm to efficiently select training eigenvectors for eigenvector continuation. Self-learning is an active-learning process that relies on a fast estimate of the emulator error and a greedy local optimization algorithm that becomes more accurate as the emulator approximation improves. We show that self-learning emulators are highly efficient algorithms that offer both high speed and high accuracy, and it can be applied to any emulator that emulates the solution to a system of constraint equations, such as solutions of algebraic or transcendental equations, linear and nonlinear differential equations, and linear and nonlinear eigenvalue problems.

ACKNOWLEDGMENTS

I am extremely grateful to my academic advisor, Dean Lee, for his guidance and support throughout my time as a graduate student. All this work would not have been possible without him. I am also thankful for the amount of time he spent with his research students and me outside of work and research. I am grateful for all the great experiences I had with him while traveling to Saugatuck beach, visiting the tulip festival in Holland, going apple-picking, and during the long drives to conferences. Through several conversations over lunches and dinners, I had the opportunity to learn various things, including several insights about academic life, which I appreciated a lot. I hope to learn more from him through future research collaborations.

I would like to thank my other PhD guidance committee members, William Lynch, Thomas Parker, Jeffery Schenker, and Vladimir Zelevinsky. I have learned a lot from their excellent questions and feedback on my research.

I am also grateful for all the faculty I had the opportunity to interact with during my time as a graduate student. I would especially like to thank Scott Pratt, Wade Fisher, Thomas Parker, Jeffery Schenker, Alex Brown, and Vladimir Zelevinsky for helping me learn the basics of theoretical physics and mathematics during the first couple of years of my graduate student life.

I would also like to express gratitude to Ning Li, who spent a lot of time with me to help me learn the nuclear lattice Effective Field Theory codes. I also appreciated the research discussions I had with several other people in the lab. In particular, I would like to thank Xilin Zhang. I would also like to thank my group members, Caleb Hicks, Jacob Watkins, Gabriel Given, Zhengrong Qian, and Joseph Bonitati, for all the great discussions.

Finally, I would like to acknowledge the enormous emotional support that I have received from my friends in the lab. I will also miss the company of my roommates Zhen Li and Juan Gu. This work would have been much more difficult without them.

The computational resources for this work were provided by the Institute for Cyber-Enabled Research (iCER) at Michigan State University.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

Introduction

Eigenvector continuation (EC) is a relatively new computational technique [1] that finds the extremal eigenvalues and eigenvectors of a parameter dependent Hamiltonian matrix. It is a variational method where we project the Hamiltonian onto a subspace of basis vectors corresponding to eigenvectors at some chosen training values of the control parameters.

Eigenvector continuation was originally designed for extrapolation of quantum many-body wave functions in extremely large vector spaces where general vector manipulations are not possible. It has been used in our research group to extend quantum Monte Carlo methods to problems with strong sign oscillations [2]. It has been used as a fast emulator for quantum many-body systems [3,4], and as a resummation method for perturbation theory [5]. It is well suited for studying the connections between microscopic nuclear forces and nuclear structure, a topic that has generated much recent interest [6–14]. However, we should also mention that eigenvector continuation is a special case of a larger existing formalism called reduced basis methods, which is itself a special case of an even larger class of methods called model order reduction.

Although it has been shown that eigenvector continuation gives us accurate approximations to our target extremal eigenvalue and eigenvector, its convergence properties have not been studied before. We know that the error in our approximation improves if we take

more training points, but the location of the training points also make a big difference. This dependence of approximation error on the number of training points and their location is very problem specific, and very hard to characterize.

In this work, we present a study of convergence properties of eigenvector continuation, which give us additional insight about how the method works. Using this insight and machine learning, we then present a self-learning algorithm to select optimal training points for eigenvector continuation. We show that this active-learning protocol of self-learning can not only be used with eigenvector continuation emulators, but also with any emulator that tries to emulate the solution to a system of constraint equations. With various examples, we try to demonstrate the widespread applicability of self-learning emulators and hope that they will be useful in other scientific fields using emulators.

This work is organized as follows. In this chapter we go over our notations, basic definitions, and some introduction and background on basic topics that will be relevant for studying eigenvector continuations. We encourage the reader to directly skip to the next chapter if the reader is already familiar with these concepts. All the chapters after this will make no reference to the materials presented in this chapter.

In chapter 2, we introduce the method of eigenvector continuation, and explain all its details. We demonstrate with simple examples how to apply it to any sample problem, and illustrate the advantages of using this method. In chapter 3, we consider a practical problem in physics - the anharmonic oscillator. Standard perturbation theory has zero radius of convergence in the case of the anharmonic oscillator, but we show that eigenvector continuation can still be applied to this problem. The method works perfectly well, and we can extrapolate to any region using this technique.

In chapter 4, motivated by our results from the anharmonic oscillator, we discuss the

error convergence properties of eigenvector continuation. We show how the approximation error in eigenvector continuation depends on the location and number of training points in this problem, and we try to study the asymptotic convergence behaviour as we add more and more training points. This helps us understand more about eigenvector continuation, and we make an important realization about orthogonality among the training eigenvectors.

In chapter 5, we continue our discussion of error convergence and selection of optimal training points for eigenvector continuation. We come up with an iterative self-learning algorithm that uses eigenvector continuation emulator itself to learn where the next optimal training point is. We then show that this algorithm is actually very general and can be applied to a whole class of emulators with a constraint equation. We present several examples to demonstrate that self-learning emulators work very well and that it efficiently learns the optimal locations for training points.

In chapter 6, we give a brief introduction to our research group's Nuclear Lattice Effective Field Theory (EFT) calculations of nuclear observables. This is done the help of Auxiliary-Field Quantum Monte Carlo, but in these calculations we are plagued by a problem called "sign problem", which result from the repulsive nature of the Coulomb interaction. This is a signal-to-noise ratio problem, and with strong sign problems, we cannot extract the ground state energy using this formalism. We then present an algorithm to implement eigenvector continuation in these lattice Monte Carlo codes, called the floating-block eigenvector continuation. We show that eigenvector continuation can help us alleviate some of our sign problems by being able to extrapolate to the target point we are interested in.

Finally we end with conclusions and possible future work. We also provide some details of our numerical calculations in a supplemental materials, which is given at the end of this document.

We start with defining our notations and basic Hamiltonian mechanics. We will not describe all the details, and the reader, if unfamiliar with the topic, is encouraged to look up any quantum mechanics book [15].

1.1 Hamiltonian

In physics, we are often interested in understanding how particles interact with each other, and how they evolve in time. For large particles, we have Newtonian mechanics, which describe the physics of their interactions very well. For small particles we need quantum mechanics. Here we will first describe quantum mechanics using wave mechanics formalism. Following the usual wave mechanics notations, we use the term wavefunction to represent any function in our coordinate space. If we take the Fourier transform of the wavefunction in coordinate space, we get a wavefunction in momentum space.

The wavefunction $\Psi(\mathbf{x}, t)$ can represent a state of particle(s), and in coordinate representation, $|\Psi(\mathbf{x}, t)|^2 d\mathbf{x}$ is the probability of finding the particle(s) in the region $\mathbf{x}$ to $\mathbf{x} + d\mathbf{x}$. This wave function satisfies the Schrödinger equation, which governs how it evolves in time. In one-dimension, this is given by,

$$i\hbar \frac{\partial \Psi}{\partial t} = -\frac{\hbar^2}{2m} \frac{\partial^2 \Psi}{\partial x^2} + V\Psi \tag{1.1}$$

This time-dependent equation is often written in terms of the *Hamiltonian* operator $\hat{H} = \hat{p}^2/2m + V$, where $\hat{p} = (\hbar/i)(\partial/\partial x)$. The corresponding equation is then $\hat{H}\Psi = i\hbar(\partial\Psi/\partial t)$. In multi-dimensions, $(\partial/\partial x)$ becomes multi-dimensional derivatives ∇.

The Hamiltonian is an important concept because any given problem and its interac-

tions can be expressed by its Hamiltonian. For example, for the free particle without any interactions, the Hamiltonian is given by $\hat{H} = \hat{p}^2/2m$.

We solve equation 1.1 by separation of variables, and separating out the time dependence we get,

$$\hat{H}\psi = E\psi \tag{1.2}$$

where $\Psi = \psi e^{-iEt/\hbar}$. We call the solutions to equation 1.2 stationary states because the probability of finding the particle at any point $|\Psi|^2 = |\psi e^{-iEt/\hbar}|^2 = |\psi|^2$ does not change with time.

Since the Hamiltonian $\hat{H}$ is a Hermitian operator, the stationary state solutions to 1.2 form a complete basis. This means that we can represent any given wavefunction as a linear combination of the stationary states. This also allows us to write any operator as a function that takes a given stationary state to some combination of other stationary states. In other words, we can represent any operator as a matrix that acts on a vector written in terms of a basis of stationary states. This is the matrix formalism of quantum mechanics, and using this matrix mechanics we can turn the Schrödinger equation of 1.2 into a matrix equation,

$$H\,|\psi\rangle = E\,|\psi\rangle \tag{1.3}$$

where H is a matrix and $|\psi\rangle$ is a column vector. We are using the Dirac notation of bra-ket, where $|\psi\rangle$ is called a ket and is a column vector, and $\langle\psi|$ is called a bra and is a row vector. Their inner product is written as $\langle\psi|\psi\rangle$, and it is just the multiplication of a row vector and a column vector. We also assume that any given state $|\psi\rangle$ is normalized such that $\langle\psi|\psi\rangle = 1$,

because we can always re-normalize any given ket as $|\psi'\rangle = |\psi\rangle / \sqrt{\langle\psi|\psi\rangle}$, so that $\langle\psi'|\psi'\rangle = 1$.

In this work, we will be looking at the Hamiltonian as a matrix H, and we are interested in solving the eigenvalue problem given in equation 1.3. The eigenvalue E has a physical interpretation as energy of the system, and thus, we will often call the eigenvalue of the Hamiltonian as eigenenergy or just energy. We call the smallest eigenvalue as the ground state energy, and the larger eigenvalues as excited state energies. The eigenvectors corresponding to them are called ground state eigenvector and excited state eigenvectors respectively.

Since eigenvector continuation is a variational technique that gives best results with the ground state, we will often be interested only in the ground state. Therefore, in this work we will often call the ground state energy and the ground state eigenvector of the Hamiltonian as just the eigenenergy (or simply energy) and eigenvector of the Hamiltonian respectively.

1.2 The Harmonic Oscillator

In this section we review the quantum harmonic oscillator that is usually covered in a quantum mechanics class. The harmonic oscillator problem is important for us because in chapter 3, we will consider the anharmonic oscillator, which builds on the harmonic oscillator problem. Here we define the ladder operators and show how we get the eigenenergies for this problem.

The classical harmonic oscillator is a mass m attached to a spring with spring constant k. Its motion is given by Hooke's law,

$$F = -kx = m\frac{d^2x}{dt^2} \tag{1.4}$$

Its potential is given by,

$$V = \frac{1}{2}kx^2 = \frac{1}{2}m\omega^2 x^2 \tag{1.5}$$

where $\omega = \sqrt{k/m}$ is the frequency of oscillation.

Accordingly, the quantum harmonic oscillator Hamiltonian is defined by,

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega^2 x^2 \tag{1.6}$$

To solve the Schrödinger equation with this Hamiltonian, we define the ladder operations as follows,

$$a_\pm = \frac{1}{\sqrt{2\hbar m\omega}}\big(\mp ip + m\omega x\big) \tag{1.7}$$

Here a_+ is called the creation operator and a_- is called the annihilation operator. If we define the eigenstates of the number operator $N = a_+a_-$ as $|n\rangle$, then $a_+ |n\rangle = \sqrt{n+1}\,|n\rangle$ and $a_- |n+1\rangle = \sqrt{n}\,|n-1\rangle$.

With the help of commutation relationship $[x, p] = i\hbar$, these operators allows us to rewrite the Hamiltonian in the following way,

$$H = \hbar\omega\Big(a_+a_- + \frac{1}{2}\Big) \tag{1.8}$$

Since $N |n\rangle = n |n\rangle$, we see that the eigenenergies of the Hamiltonian are of the form,

$$E_n = \hbar\omega\Big(n + \frac{1}{2}\Big) \tag{1.9}$$

and the eigenvectors of the Hamiltonian are the same as the eigenvectors of the Number operator $|n\rangle$. In coordinate representation,

$$|n\rangle = A_n(a_+)^n\psi_0(x), \qquad \text{where } \psi_0(x) = \left(\frac{m\omega}{\pi\hbar}\right)^{1/4} e^{-\frac{m\omega}{2\hbar}x^2} \tag{1.10}$$

where A_n is some normalization constant to make $\langle n|n\rangle = 1$.

1.3 Perturbation Theory and Avoided Level-Crossing

In this work, we consider parameter dependent Hamiltonians of the form $H(c) = H_0 + cH_1$, where H_0 and H_1 are hermitian matrices. The idea here is that H_1 can be seen as a perturbation from H_0, whose solution we know already. We can think of arriving at the target Hamiltonian $H(c_t)$ by setting $c = 0$ and slowly increasing it until $c = c_t$. If the parameter dependent wavefunction and energy are denoted by $\psi(c)$ and $E(c)$, then for small c, we can expand them in a power series,

$$\begin{aligned} \psi(c) &= \psi(0) + c\psi'(0) + \frac{c^2}{2!}\psi''(0) + \cdots \\ E(c) &= E(0) + cE'(0) + \frac{c^2}{2!}E''(0) + \cdots \end{aligned} \tag{1.11}$$

where the prime represents derivative with respect to the parameter c.

In standard perturbation theory, one would put these expansions into the Schrödinger equation and compare the coefficients of equal derivatives. We do not derive the expressions for the results of perturbation theory here. Instead, here we will look at the expansion in equation 1.11 in the complex plane (when we extend c to be complex parameter), and try to understand why at some point the series expansion break down.

If we assume that all eigenvalues of H_0 and H_1 are not degenerate, then as we change the parameter in our Hamiltonian in $H = H_0 + cH_1$, the relative ordering of eigenvalues and their corresponding eigenvectors change. As $c \to \pm\infty$, the eigenvalues and eigenvectors of H are dominated by the eigenvalues and eigenvectors of H_1. When $c \to -\infty$ the largest eigenvalue of H_1 is the smallest eigenvalue of H, and the smallest eigenvalue of H_1 is the largest eigenvalue of H. But when $c \to \infty$ the largest eigenvalue of H_1 is now the largest eigenvalue of H, and the smallest eigenvalue of H_1 becomes the smallest eigenvalue of H. As we move from $c \to -\infty$ to $c \to \infty$, all the eigenvalues of the H invert, and at some point they cross each other. In general, this crossing happens somewhere in the complex plane, and not necessarily on the real axis. If we look at only along the real axis, then we see that different eigenvalues cross each other, but they do not intersect. Instead, we see *avoided level-crossings*.

Corresponding to these avoided level-crossings along the real-axis, there are associated branch point singularities of the levels when they are continued in the complex plane. These very special points are called exceptional points. They have been well studied [16, 17], and we only mention some brief details here.

Consider the avoided level-crossing between two eigenvalues $\epsilon_1(c)$ and $\epsilon_2(c)$. Suppose these eigenvalues meet at λ_c in the complex plane. At this point, the two eigenvalues are equal, i.e., $\epsilon_1(\lambda_c) = \epsilon_2(\lambda_c)$, and their corresponding eigenvectors also converge into a single eigenvector. This makes these points special because at these points there is only one eigenvector corresponding to this repeated eigenvalue, and the dimension of the eigenspace is less than the dimension of the matrix. Therefore, we have a defective matrix, and the Hamiltonian is not fully diagonalizable. The two functions, $\epsilon_1(c)$ and $\epsilon_2(c)$, are the values of one single analytic function on two Riemann sheets, and the two sheets are connected at

the branch point singularity occurring at $c = \lambda_c$.

These branch point singularities of eigenvectors and eigenvalues restrict the area of convergence for simple expansions like that in equation 1.11. This means that when we have an exceptional point at λ_c, standard perturbation theory will fail if we try for a large c, with $|c| > |\lambda_c|$. This concept will be relevant for us when we discuss how eigenvector continuation can extrapolate beyond the region of convergence of perturbation theory using analytic continuation in the complex plane.

1.4 Variational Principle

The variational principle says that for any given state $|\psi\rangle$ and Hamiltonian H, the ground state energy E_{gs} will be less than or equal to $\langle\psi|H|\psi\rangle$, i.e., $E_{gs} \leq \langle\psi|H|\psi\rangle$. This is relevant for us because eigenvector continuation is a variational technique, and we will see that the eigenvector continuation energy approximation for the ground state will always be greater than the actual energy.

The variational principle also highlights a second important observation about eigenvector continuation. Variational principle applies only for the ground state energy of the system. If we want to apply variational principle for excited state, then we need to remove the ground state and look in orthogonal directions to apply it. We have a similar situation with eigenvector continuation. Eigenvector continuation works best for estimating ground state energy and eigenvector (i.e. extremal eigenvector and eigenvalue), however, it can be applied to estimate the excited state energy if we include the ground state and excited state eigenvectors in the training data.

Chapter 2

Eigenvector Continuation

Eigenvector continuation (EC) is a variational method that finds the extremal eigenvalues and eigenvectors of a Hamiltonian matrix that depends on one or more control parameters. It was first introduced in [1, 2], and was originally developed from studying the convergence of the Lanczos algorithm and searching for a set of basis states that would converge more quickly than the Krylov subspace.

The main idea in eigenvector continuation is that as we vary the parameters in the Hamiltonian that we have control over, the extremal eigenvector also varies with the control parameter, but this eigenvector moves in a subspace whose dimensionality is much less than the dimension of the Hilbert space that the eigenvector lives in. This implies that we will capture the essence of the problem if we project the problem onto the smaller vector space that the extremal eigenvector lives in. The estimate that we can get from this projection is much faster to calculate because of the reduction in dimensionality. This speedup boost makes eigenvector continuation a general technique that can be applied to variety of problems.

Typically, we apply eigenvector continuation to a problem by projecting the target Hamiltonian onto a subspace of basis vectors corresponding to eigenvectors at some chosen training values of the control parameters. We then solve the generalized eigenvalue problem

to find the extremal eigenvectors. Effectively, this gives a linear combination of the input eigenvectors that minimizes the eigenvalue of the Hamiltonian at the target value of the parameter. This linear combination serves as a good low-dimensional estimate for our exact eigenvector.

2.1 How to Apply Eigenvector Continuation

Let us consider a one-parameter family of Hamiltonian matrices $H(c) = H_0 + cH_1$, where both H_0 and H_1 are finite-dimensional Hermitian matrices, and c is a parameter that we can control. The extension to the multi-parameter case follows naturally, and will be discussed after our discussion of the one-parameter case.

For the given Hamiltonian family $H(c)$, we are interested in finding the ground state eigenvector $|v(c_t)\rangle$ (=extremal eigenvector) and eigenvalue $E(c_t)$ for some target parameter value $c = c_t$. As noted in Ref. [1], eigenvector continuation can also be extended to excited states by including excited state eigenvectors at the training points. For now we will focus on ground state calculations in this analysis.

The first big advantage of eigenvector continuation that we will illustrate is that it allows us to extrapolate to a region where we cannot access by direct calculation. Our next chapter is entirely dedicated to illustrate this point with the help of an example on anharmonic oscillator. For now, suppose that for the given c_t that we are interested in, $H(c_t)$ cannot be diagonalized directly with regular computational techniques in a reasonable amount of time. We can only calculate its eigenvalues and eigenvectors for small c. We want to use this information about the eigenvectors for small c to find the eigenvector at large c_t. Let us calculate the exact eigenvectors for k small values of c, where we can compute the eigenvector

of the Hamiltonian matrix directly. We will call these k points as our training points since we are training eigenvector continuation to give an output based on these training eigenvectors. We denote these points by $c_i = \{c_1, \cdots, c_k\}$, and the ground state eigenvector at those corresponding points by $|v(c_1)\rangle, \cdots, |v(c_k)\rangle$. We are interested in finding $|v(c_t)\rangle$ and $E(c_t)$, where $H|v(c_t)\rangle = E(c_t)|v(c_t)\rangle$.

We define the norm matrix N as,

$$N_{ij} = \langle v(c_i)|v(c_j)\rangle \tag{2.1}$$

where $|v(c_i)\rangle$ and $|v(c_j)\rangle$ are both taken from the training eigenvector set $|v(c_1)\rangle, \cdots, |v(c_k)\rangle$.

We project the target Hamiltonian $H(c_t)$ on to the subspace spanned by the training eigenvectors $|v(c_1)\rangle, \cdots, |v(c_k)\rangle$, and define the projected matrix $\tilde{H}$ as,

$$\tilde{H}_{ij}(c_t) = \langle v(c_i)|H(c_t)|v(c_j)\rangle \tag{2.2}$$

Note that we are using the target Hamiltonian with parameter $c = c_t$, but we are projecting onto eigenvectors with parameter $c = c_i$.

After calculating these matrices, we solve the generalized eigenvalue problem for $\tilde{H}$ and N. A generalized eigenvalue problem for these two matrices is the problem of finding a (nonzero) vector $|w(c_k)\rangle$ that satisfies,

$$\tilde{H}|w(c_t)\rangle = \tilde{E}N|w(c_t)\rangle \tag{2.3}$$

Then, our eigenvector continuation estimate of energy is $E_{EC} = \tilde{E}$ and EC estimate of

the eigenvector is $|v\rangle_{EC} = T_V\,|w(c_t)\rangle$, where T_V is the matrix of training vectors,

$$T_V = \Big[\,|v(c_1)\rangle \quad \cdots \quad |v(c_k)\rangle\,\Big] \tag{2.4}$$

Here $|v(c_i)\rangle$ corresponds to the eigenvector at $c = c_i$, with all elements written in a column. The eigenvector $|w(c_t)\rangle$ we get after we solve the generalized eigenvalue problem is the eigenvector we want, but it is in the projected low-dimensional space. We get our eigenvector in our original Hilbert space, we need to multiply by the matrix of training vectors T_V.

We will address the question of how to choose our training points $c_i = \{c_1, \cdots, c_k\}$ later. Different choices of c_i gives us different results, and the accuracy of our eigenvector continuation approximation depends on our choice of c_i, and we will study this dependence when we study the convergence properties of eigenvector continuation. Only after we study the convergence properties, we will be in a position to discuss strategies to optimally choose our training points.

In theory, this is all we need to perform eigenvector continuation. However, in practice, we often ortho-normalize our training vectors by Gram-Schmidt algorithm. This makes the norm matrix N_{ij} the identity operator, and equation 2.3 just becomes a normal eigenvalue problem. Note that when we orthogonalize our training vectors, we also need to use a orthonormal set of eigenvectors in the training vector matrix in 2.4. Orthogonalization speeds up our calculation especially for large number of training vectors, and it is critical for our understanding of convergence of eigenvector continuation. We will describe the convergence in more details after two chapters.

Before we move to an example, let us summarize the main idea of eigenvector continuation. H is a n dimensional matrix, where n can be a very large number, and $\tilde{H}$ is a k

dimensional projection of the H matrix (at the target parameter), onto the space of k training points, and k is much usually smaller than n. The training eigenvectors are the exact eigenvectors of the Hamiltonian at different parameter values, and often far away from the target parameter value. By solving for the eigenvalue of $\tilde{H}$ instead of H, we have reduced the dimensionality of the problem, and the diagonalization is much faster. This illustrates the second big advantage of eigenvector continuation - huge computational speed up factor. Later we will see that instead of using eigenvector continuation for extrapolation, we can use EC just for this computational speed boost and use it to build fast and accurate emulators that can interpolate in between training points.

For now, we will focus on extrapolation. In the next chapter, we will show how this extrapolation works with a practical example - anharmonic oscillator. In the anharmonic oscillator problem, standard perturbation theory diverges and we cannot get any results beyond a certain value of the parameter. But as we will soon show, eigenvector continuation is able to extrapolate to a region where perturbation theory fails. However, we will first take a look at a simple problem using random matrices to demonstrate how we apply eigenvector continuation to any problem. Suppose we have a Hamiltonian $H = H_0 + cH_1$, where H_0 and H_1 are random 10-dimensional Hermitian matrices, given by,

$$H_0 = \begin{bmatrix} 9.87 & 9.00 & 10.14 & 9.86 & 10.62 & 9.94 & 10.01 & 10.11 & 9.90 & 9.04 \\ 9.00 & 10.11 & 8.93 & 9.58 & 9.38 & 9.15 & 10.77 & 9.70 & 9.71 & 7.90 \\ 10.14 & 8.93 & 9.88 & 9.87 & 10.85 & 9.86 & 10.11 & 8.78 & 9.78 & 10.55 \\ 9.86 & 9.58 & 9.87 & 9.93 & 9.82 & 10.35 & 10.06 & 9.91 & 9.47 & 10.81 \\ 10.62 & 9.38 & 10.85 & 9.82 & 10.15 & 9.74 & 9.63 & 11.29 & 10.03 & 10.68 \\ 9.94 & 9.15 & 9.86 & 10.35 & 9.74 & 8.08 & 9.67 & 9.00 & 11.16 & 10.52 \\ 10.01 & 10.77 & 10.11 & 10.06 & 9.63 & 9.67 & 9.41 & 8.54 & 10.73 & 10.15 \\ 10.11 & 9.70 & 8.78 & 9.91 & 11.29 & 9.00 & 8.54 & 8.74 & 10.40 & 10.04 \\ 9.90 & 9.71 & 9.78 & 9.47 & 10.03 & 11.16 & 10.73 & 10.40 & 11.20 & 8.96 \\ 9.04 & 7.90 & 10.55 & 10.81 & 10.68 & 10.52 & 10.15 & 10.04 & 8.96 & 12.27 \end{bmatrix}$$

$$H_1 = \begin{bmatrix} 0.15 & 0.04 & 0.13 & 0.10 & 0.15 & 0.09 & 0.07 & 0.13 & 0.13 & 0.08 \\ 0.04 & 0.21 & 0.10 & 0.09 & 0.08 & 0.14 & 0.14 & 0.06 & 0.10 & 0.09 \\ 0.13 & 0.10 & 0.03 & 0.08 & 0.06 & 0.04 & 0.12 & 0.08 & 0.07 & 0.07 \\ 0.10 & 0.09 & 0.08 & 0.11 & 0.13 & 0.11 & 0.06 & 0.10 & 0.10 & 0.12 \\ 0.15 & 0.08 & 0.06 & 0.13 & 0.13 & 0.02 & 0.04 & 0.11 & 0.11 & 0.04 \\ 0.09 & 0.14 & 0.04 & 0.11 & 0.02 & 0.10 & 0.11 & 0.09 & 0.10 & 0.13 \\ 0.07 & 0.14 & 0.12 & 0.06 & 0.04 & 0.11 & 0.09 & 0.13 & 0.05 & 0.08 \\ 0.13 & 0.06 & 0.08 & 0.10 & 0.11 & 0.09 & 0.13 & 0.07 & 0.13 & 0.13 \\ 0.13 & 0.10 & 0.07 & 0.10 & 0.11 & 0.10 & 0.05 & 0.13 & 0.05 & 0.12 \\ 0.08 & 0.09 & 0.07 & 0.12 & 0.04 & 0.13 & 0.08 & 0.13 & 0.12 & 0.04 \end{bmatrix}$$

Suppose we have the exact ground state eigenvector $|\psi(c)\rangle$ at $c = 50$ and $c = 100$, but we are interested in the eigenvector at $c_t = 1000$. For this simple example, we can calculate the

eigenvector directly by othrogonalization of the Hamiltonian, but for demonstration purposes we will show that we can find an estimate of the ground state eigenvector at $c = 1000$ with eigenvector continuation, using only two training eigenvectors.

The ground state eigenvectors at $c = 50$ and $c = 100$ are,

$$|\psi(c=50)\rangle = \begin{bmatrix} 0.31 & 0.08 & -0.44 & 0.06 & -0.04 & -0.46 & 0.31 & -0.46 & 0.24 & 0.36 \end{bmatrix}^T$$

$$|\psi(c=100)\rangle = \begin{bmatrix} 0.31 & 0.06 & -0.48 & 0.07 & -0.07 & -0.42 & 0.39 & -0.43 & 0.22 & 0.32 \end{bmatrix}^T$$

So our training vector matrix T_V is a 10×2 matrix, where the two columns are the two eigenvectors above. However, before projecting $H(c_t)$ onto these training vectors, we first orthonormalize them. The orthogonalized T_V matrix is,

$$T_V = \begin{bmatrix} 0.31 & 0.08 & -0.44 & 0.06 & -0.04 & -0.46 & 0.31 & -0.46 & 0.24 & 0.36 \\ 0.01 & 0.22 & 0.38 & -0.12 & 0.24 & -0.31 & -0.70 & -0.16 & 0.20 & 0.30 \end{bmatrix}^T$$

Now the norm matrix N is just the 2×2 identity matrix, and the projected Hamiltonian matrix $\tilde{H}$ is given by,

$$\tilde{H} = T_V^\dagger H T_V = \begin{bmatrix} -124.24 & 27.09 \\ 27.09 & -9.39 \end{bmatrix}$$

The eigenvalue of this matrix $\tilde{H}$ is -130.31, and its corresponding eigenvector is $\begin{bmatrix} 0.98 & -0.22 \end{bmatrix}^T$. When projected back to our 10-dimensional space, we get our eigenvector continuation estimate of eigenvector.

$$|\psi(c_t)\rangle_{EC} = \Big[0.30\ 0.03\ -0.51\ 0.08\ -0.09\ -0.38\ 0.45\ -0.41\ 0.19\ 0.29\Big]^T$$

The exact eigenvalue at $c_t = 1000$ is -130.56 and the exact eigenvector is

$$|\psi(c_t)\rangle = \Big[0.31\ 0.05\ -0.52\ 0.11\ -0.13\ -0.39\ 0.44\ -0.39\ 0.22\ 0.23\Big]^T$$

The difference in eigenvalue is 0.25, and the norm of the difference of the two eigenvectors is about 0.08. Thus, with only two training points, eigenvector continuation gave us an eigenvalue estimate with 99.8% accuracy, and an eigenvector estimate with 92% accuracy. With more training points, the estimate becomes better. If we add another training point at $c = 150$, then with three training points, the eigenvector continuation estimate of eigenvalue is 99.99% accurate, and the EC estimate of eigenvector is 99% accurate. Note that to find the precision, we need to calculate the exact result at the target point and directly compare with the eigenvector continuation approximation. Calculation of the exact result can be hard or impossible in some problems, and in those cases the precision of our approximation cannot be calculated directly. For now, we discuss problems where we can calculate the exact result and directly compare our approximations. Later, we will address this issue when we discuss self-learning eigenvector continuation.

Eigenvector continuation reduced the problem of diagonalizing a 10-dimensional matrix to diagonalizing a 2-dimensional matrix. For this particular example, we achieved a speed-up factor of more than 7 times, while losing $< 0.2\%$ accuracy. This shows why eigenvector continuation can be such a efficient tool at making approximations. We can achieve even more accurate estimates with more training points, but it will take a little longer time. There is a trade-off between accuracy and computational speed. However, eigenvector continuation

will always be faster than direct diagonalization of the target matrix.

Note that in our eigenvector continuation estimate, the accuracy in eigenvalue is higher than accuracy in eigenvector. Furthermore, the estimate of eigenvalue is greater than the actual eigenvalue. This is because eigenvector continuation is a variational technique, and works by trying to minimize the eigenvalue. In the next section, we describe the variational approach of this method, and gain deeper understanding of how eigenvector continuation works.

2.2 Variational Principle in Eigenvector Continuation

Although it might not seem like it, eigenvector continuation is a variational technique, and as such we see several properties in EC as seen in other variational methods. For example, the eigenvector continuation estimate of ground state energy will always be greater than the actual ground state energy of the problem. We mentioned before that the EC estimate of the eigenvector is a linear combination of the training eigenvectors. Let us see why this is true, and it will be clear why eigenvector continuation is a variational technique.

Consider again the T_V, H and $\tilde{H}$ matrix in equations 2.1, 2.2 and 2.4. This time let us write the eigenvector $|w(c_t)\rangle$ as a column vector $|w(c_t)\rangle = [a_1 \cdots a_k]^T$. Also, for simplification, let us orthonormalize our training vectors, so that the norm matrix becomes an identity matrix, i.e., $N = T_V^\dagger T_V = T_V T_V^\dagger = I$. Now we are interested in solving the

generalized eigenvalue problem given in equation 2.3,

$$\tilde{H} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix} = \tilde{E} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix}$$

$$\Rightarrow \Big[|v_1\rangle \cdots |v_k\rangle \Big]^T_{k\times n} H_{n\times n} \Big[|v_1\rangle \cdots |v_k\rangle \Big]_{n\times k} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix}_{k\times 1} = \tilde{E} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix}_{k\times 1}$$

Multiplying both sides by T_V and using orthonormality of T_V matrix, we have

$$\Rightarrow H_{n\times n} \Big[|v_1\rangle \cdots |v_k\rangle \Big]_{n\times k} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix}_{k\times 1} = \tilde{E} \Big[|v_1\rangle \cdots |v_k\rangle \Big]_{k\times n} \begin{bmatrix} a_1 \\ \dots \\ a_k \end{bmatrix}_{k\times 1}$$

$$\Rightarrow H_{n\times n} \Big[a_1 |v_1\rangle + \cdots + a_k |v_k\rangle \Big]_{n\times 1} = \tilde{E} \Big[a_1 |v_1\rangle + \cdots + a_k |v_k\rangle \Big]_{n\times 1} \tag{2.5}$$

To improve clarity, we have shown the size of the vectors in subscript.

We are interested in the lowest eigenvalue, and as equation 2.5 shows, the eigenvector continuation algorithm picks a linear combination of the training vectors that minimizes the eigenvalue $\tilde{E}$. The eigenvector $[a_1 \cdots a_k]^T$ of the projected Hamiltonian $\tilde{H}$ is in fact the coefficients of the linear combination of the training eigenvectors.

Now the variational approach of eigenvector continuation is clear, because the method is varying the coefficients $[a_1 \cdots a_k]^T$ to minimize the energy. We do not see it directly be-

cause we are not minimizing anything directly. However, when we are looking for the lowest eigenvalue, the method chooses the coefficients so that it can get the smallest eigenvalue as possible. In this way, eigenvector continuation chooses a linear combination of the training eigenvectors such that the eigenvalue is minimized. This variational principle is also the reason why eigenvector continuation works best for the ground state eigenvector and eigenvalue, i.e., extremal eigenvector and eigenvalue.

Now let us look into why a linear combination of training points can approximate the target eigenvector so well, even when we are extrapolating far away in the parameter space. This brings us to our next topic - analytic continuation.

2.3 Analytic Continuation

We can understand why eigenvector continuation can extrapolate so well by seeing it as an analytic continuation in the complex plane. By looking at series expansions in the complex plane, we can also see how eigenvector continuation compares to perturbation theory. Let us first see how analytic continuation can be used to extend perturbation theory, and then we will discuss why this is relevant to eigenvector continuation. The ideas presented in this section are not directly used while applying eigenvector continuation to a practical problem. Here we only present theoretical ideas that can help understand how eigenvector continuation works.

Suppose we have a parameter dependent Hamiltonian $H(c)$, where H is Hermitian for real c. We are interested in calculating the ground state eigenvector of $H(c)$ for some real c, but we can also look at the dependence of the eigenvector as we vary c in the complex plane. In the end we will be interested in a point on the real axis, but this extension to

complex plane has several benefits, and that one that will be obvious soon is that it allows us to visualize the effective range of perturbation theory. For the typical Hamiltonian form that we will consider, $H(c) = H_0 + cH_1$ with Hermitian H_0 and H_1, one sees several avoided level-crossings that correspond to singular points on the complex plane. These points limit the radius of convergence of any power series expansion of the eigenvector involved in the level-crossing.

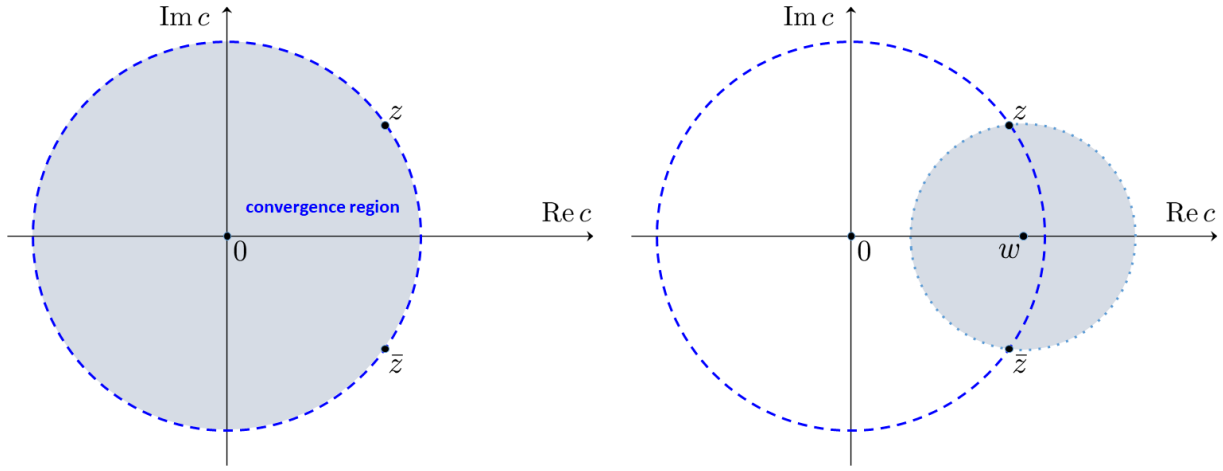
Suppose we want to expand the ground state eigenvector $|\psi(c)\rangle$ around $c = 0$. Then we can write,

$$|\psi(c)\rangle = \sum_{n=0}^{\infty} |\psi^{(n)}(0)\rangle \frac{c^n}{n!} \tag{2.6}$$

where $|\psi^{(n)}(0)\rangle$ denotes the n^{th} derivative of the eigenvector with respect to the parameter c, and evaluated at $c = 0$. Note that this derivative is another eigenvector, and is in general a multidimensional vector. Now suppose there is a singularity at z and $\bar{z}$. The situation shown in figure 2.1. The singularity will restrict the radius of convergence of equation 2.6 to the circle shown in picture 2.1a. Equation 2.6 is how standard perturbation theory works, and we generally perturb from $c = 0$. So if we were to use perturbation theory in our problem, we would only be able to get results for any c within the radius of convergence. If we move outside the circle shown in 2.1a, perturbation theory breaks down.

Now we can also expand in a power series centered at another point on the real axis, say w. This would be corresponding to doing perturbation theory, but starting our perturbation from $c = w$. For this series we can write,

$$|\psi(c)\rangle = \sum_{n=0}^{\infty} |\psi^{(n)}(w)\rangle \frac{(c-w)^n}{n!} \tag{2.7}$$



(a) Expansion around $c = 0$ (b) Expansion around $c = w$

Figure 2.1: Radius of convergence of power series expansion at any point is limited by the singular points z and $\bar{z}$.

As figure 2.1b shows, the circle within which this series converges is little different to the previous circle where perturbation theory from $c = 0$ converged. We have some overlap with the previous circle, but we also have covered some region outside our circle centered at $c = 0$. In the region where they overlap, both of our expansions work, and we can write an equation for the n^{th} derivative of $|\psi(c)\rangle$ at $c = w$ as a series,

$$|\psi^{(n)}(w)\rangle = \sum_{m=0}^{\infty} |\psi^{(n+m)}(0)\rangle \frac{w^m}{m!} \tag{2.8}$$

This follows from equation 2.6. Now we can substitute this equation in equation 2.7 and get,

$$|\psi^{(n)}(w)\rangle = \sum_{n=0}^{\infty} \sum_{m=0}^{\infty} |\psi^{(n+m)}(0)\rangle \frac{w^m (c-w)^n}{n!m!} \tag{2.9}$$

This equation is now applicable anywhere in the radius of convergence of the series in equation 2.7, which is the circle shown in figure 2.1b. However, it only has derivatives at $c = 0$, which can all be calculated at the origin. Thus, using only the information at $c = 0$,

we can calculate the eigenvector at any point in the circle shown in figure 2.1b. This means that we have analytically continued from the circle in figure 2.1a to the circle in figure 2.1b, and went beyond the standard perturbation theory radius of convergence. This new region that we can access by analytic continuation is shown in figure 2.2.

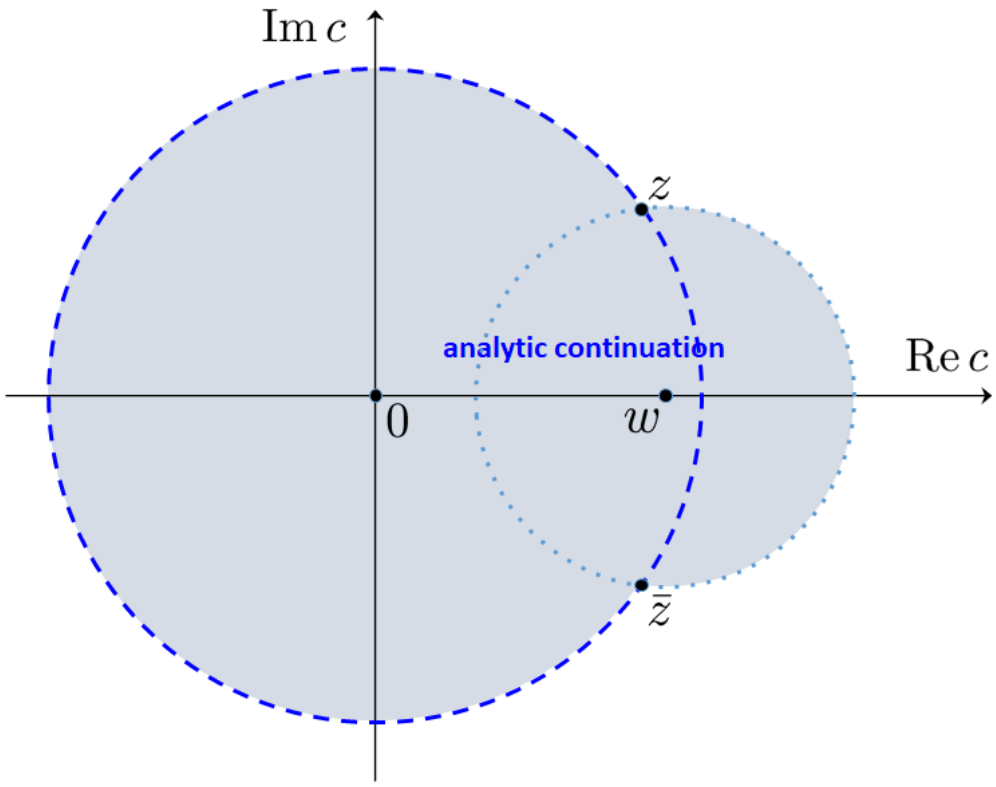


Figure 2.2: Analytic continuation allows us to extend standard perturbation theory radius of convergence.

So we have seen that we can extend perturbation theory by analytic continuation. This is possible because of the smoothness in the parameter of the Hamiltonian, and complex analysis tells us that the information about the far away points from origin, is actually present near the origin. We can write the eigenvector at a far away point as a linear combination of the derivatives of the eigenvector at origin. Now, any given training vector can also be written as a sum of derivatives of the eigenvector at origin, and consequently, we can combine them to write the eigenvector at a far away point as a linear combination of our training

eigenvectors. As we have seen in our previous section, eigenvector continuation gives us a linear combination of our input training vectors, and thus it now makes sense why eigenvector continuation works. The important thing is that for any given value of the parameter c, there exists a linear combination of training points that can approximate the eigenvector at that point $|\psi(c)\rangle$. As long as this linear combination exists, eigenvector continuation can do a variational approach to find the linear combination that approximates the energy the best.

In practical application, when we perform eigenvector continuation, we are dealing with finite dimensional matrices and we have a finite number of training points. So we don't have an infinite number of terms in our series and as such we are making truncated approximations in equations 2.6, 2.7, 2.8, and 2.9. This makes our linear sum not *exactly* equal to the actual target point eigenvector. However, the main point of this technique is that for numerical purposes, we don't need infinite or large number of training points. We get a very good approximation with a few training points.

This relation to analytic continuation is the reason why the technique was named eigenvector continuation. Let us emphasize once again that in a practical calculation, we are not calculating any series expansions. But these ideas of analytic continuation help us explain why eigenvector continuation works so well in extrapolating. We will come back to this in the next chapter, where we study the problem of anharmonic oscillator. In that context, we will demonstrate the power of analytic continuation, and how eigenvector continuation allows us to calculate values that perturbation theory cannot give.

2.4 Interpolation and Emulators using Eigenvector Continuation

In this section we discuss the second big advantage of using eigenvector continuation - huge computational speedup boost. As we mentioned before, by projecting the full size Hamiltonian H onto the subspace of training vectors, which is substantially smaller than the dimension of the H, we reduce the dimensionality of the problem by possibly orders of magnitude. This means that even when we can solve a problem using conventional means, eigenvector continuation allows us to get accurate approximations of the result potentially orders of magnitude faster.

This brings us to the next point - interpolation. So far we have seen eigenvector continuation as an extrapolation technique, especially when we see it as an analytic continuation. However, the reasoning that we used with series expansions before also works when we are inside the base circle where perturbation theory works. Within the circle where the perturbation theory works, the eigenvector at target coupling that we are interested in can also be written as a linear combination of training eigenvectors (technically an infinite number of training eigenvectors, but we get an approximation by truncating it). Another way to put it is that we can analytically continue to the circle we are already in - it is the trivial continuation. So, eigenvector continuation can be used at any target coupling, whether we are extrapolating to a far away point in the parameter space, or we are interpolating between training points in the parameter space.

We should mention that when eigenvector continuation was introduced as a general technique, the interpolation aspect of the method gained more popularity than the extrapolation aspect. The reason is because we can use eigenvector continuation as a emulator, by

giving it the exact data at particular training points and estimating (or emulating) the exact result everywhere else. This can be interpolation or extrapolation depending on where our training data is in the parameter space, however the accuracy of our eigenvector continuation emulator is greatly increased if our training points are spread out and we are interpolating points in between them.

Let us illustrate this interpolation idea with an example. We will go back to the same example that we used for extrapolation - the one with 10-dimensional random matrices. With the same H_0 and H_1, we have $H = H_0 + cH_1$. But this time, instead of being interested in the ground state eigenvector at $c = 1000$, we want an approximate ground state eigenvector for all c between $-700 \leq c \leq 700$. That is a lot of points. In our case, we discretized the c space into 14000 points, and that means we want the eigenvector at 14000 points. If we perform direct diagonalization, we need to diagonalize 14000 times, which takes some time to perform numerically. Instead we can perform eigenvector continuation everywhere, using the exact eigenvectors obtained from direct diagonalization at a fixed set of training values, as training vectors. We only need to calculate the exact result a couple of times, and performing eigenvector continuation over the rest of the points is much faster. Figure 2.3 shows the error in our eigenvector continuation emulator across the whole c parameter space with six training eigenvectors. The error in eigenvalue is just the difference between the EC emulated eigenvalue and the actual eigenvalue, whereas the error in eigenvector is the norm of the difference between the EC emulated eigenvector and the actual eigenvector. The training eigenvectors are located at points where the log of difference of the errors is -35. Technically the error should be $-\infty$, but we have limitation of numerical precision, and we manually set the error at these points to e^{-35}.

As figure 2.3 shows, eigenvector continuation does an excellent job at interpolating

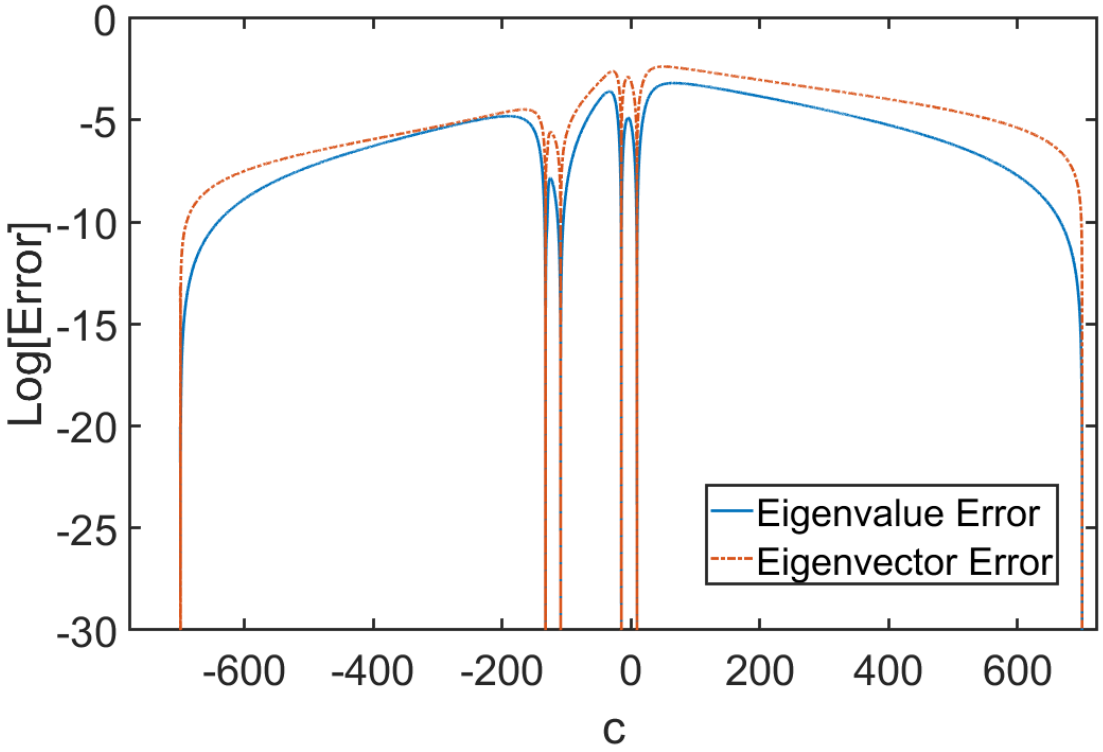


Figure 2.3: Interpolation with eigenvector continuation - we try to find approximate eigenvalue problem solution everywhere between $-700 \leq c \leq 700$ using only six training eigenvectors.

everywhere across our domain $-700 \leq c \leq 700$. However, this plot also raises several questions like, how many training points do we need to get a reasonable approximation, and where should we take our training points? We will try to answer these questions in chapters 4 and 5. For this particular example, we actually chose our training points carefully to get good results with only six training points, and we will explain the details behind choosing training points later.

Chapter 3

Anharmonic Oscillator

In this section we present our first results of eigenvector continuation applied to a practical physics problem - the anharmonic oscillator. The anharmonic oscillator has been well studied in physics, and it has several applications in nuclear and condensed matter physics. We will start with motivation of studying anharmonicity, and we will mention some of the literature work done in this field. We discuss several things about how we can apply eigenvector continuation to this system, and how it helps solve the zero radius of convergence of perturbation theory. We will also start looking at the error convergence properties of eigenvector continuation, and see how they scale with increasing number of training points. We will derive some bounds for the error, however we will come back to error convergence in more details in the next chapter.

To start things with, we have the simple quantum harmonic oscillator, which is studied in a typical quantum mechanics course. A Hamiltonian with a harmonic oscillator potential is given by,

$$H(c) = H = \frac{p^2}{2m} + \frac{m\omega^2 x^2}{2} \tag{3.1}$$

where ω is the fundamental frequency of the harmonic oscillator. We will assume the reader is familiar with this problem, and a small introduction is given in the first chapter.

However, in studying vibrations occurring in several places of nuclear and condensed matter physics, one often needs to go beyond the simple harmonic oscillator approximation. For example, anharmonic effects can be important in systems incorporating light atoms with large vibrational amplitudes, systems with weak bonding such as hydrogen bonding, and systems at high temperatures.

The quantum anharmonic oscillator has been studied extensively during the past few decades [18–23]. Several studies involving asymptotic expansions and perturbation theory, using techniques from the WKB method, have been done. In particular, it is known that in the case of the quartic anharmonic oscillator, perturbation theory has zero radius of convergence. As a matter of fact, one of the reasons to consider such a system is because this model has a well-defined but divergent perturbation series. Furthermore, the singularity structure has been analyzed before [24], and there is a singularity that lies near the negative imaginary axis.

Apart from being useful in modeling vibrations, the quartic anharmonic oscillator (henceforth referred to as just anharmonic oscillator) is also a simple model field theory in one-dimensional space-time. This field theory is defined by the Hamiltonian,

$$H = \frac{1}{2}\dot{\phi}^2 + \frac{1}{2}m^2\phi^2 + \lambda\phi^4 \tag{3.2}$$

with commutation relations $[\phi, \dot{\phi}] = i$.

Since this field-theory model involves no space dimensions, there are no asymptotic states and particle scattering. Moreover, it is a one-dimensional model, which means that it describes a universe which has just one oscillating point. Nonetheless, this model has some value in studying, since it may give us some idea of the underlying complexity of a more

realistic field theory. This model has been well studied as well [25, 26].

3.1 Anharmonic Oscillator Hamiltonian

If we add a perturbation of x^4 term to the Hamiltonian in equation 3.1, then we get the Hamiltonian for anharmonic oscillator,

$$H(c) = \frac{p^2}{2m} + \frac{m\omega^2 x^2}{2} + cx^4 \tag{3.3}$$

where c is the strength of the perturbation.

Since we consider the cx^4 term as a perturbation, we employ the annihilation and creator operators of the standard harmonic oscillator, which are given by,

$$\begin{aligned} a &= \sqrt{\frac{m\omega}{2\hbar}}(x + \frac{ip}{m\omega}) \\ a^\dagger &= \sqrt{\frac{m\omega}{2\hbar}}(x - \frac{ip}{m\omega}) \end{aligned} \tag{3.4}$$

They act on the n^{th} eigenstate, represented by $|n\rangle$, by $a|n\rangle = \sqrt{n}\,|n-1\rangle$ and $a^\dagger = \sqrt{n+1}\,|n+1\rangle$. From this we can write $x = \sqrt{\frac{\hbar}{2m\omega}}(a + a^\dagger)$.

We choose the eigenstates of the harmonic oscillator to be our basis states. In this basis, our perturbation Hamiltonian $h' = x^4$, has matrix elements given by,

$$\begin{aligned} \langle i|h'|i\rangle &= \Big(\frac{\hbar}{2m\omega}\Big)^2 6i^2 + 6i + 3 \\ \langle i-2|h'|i\rangle &= \Big(\frac{\hbar}{2m\omega}\Big)^2 (4i-2)\sqrt{i(i-1)} \\ \langle i+2|h'|i\rangle &= \Big(\frac{\hbar}{2m\omega}\Big)^2 (4i+6)\sqrt{(i+1)(i+2)} \end{aligned} \tag{3.5}$$

$$\langle i-4|h'|i\rangle = \Big(\frac{\hbar}{2m\omega}\Big)^2 \sqrt{i(i-1)(i-2)(i-3)}$$
$$\langle i+4|h'|i\rangle = \Big(\frac{\hbar}{2m\omega}\Big)^2 \sqrt{(i+1)(i+2)(i+3)(i+4)}$$
$$\langle i|h|i\rangle = \hbar\omega(i+\frac{1}{2}) \tag{3.6}$$

where $h = \frac{p^2}{2m} + \frac{m\omega^2 x^2}{2}$ is the harmonic oscillator Hamiltonian. Our entire Hamiltonian is $H = h + ch'$, and for simplicity we choose $\hbar\omega = 1$ and $m\omega^2 = 1$. All the numerical results in this work are with these choices of m and ω.

The x^4 term dominates the potential $V = \frac{m\omega^2 x^2}{2} + cx^4$ for large x, and with c being the coefficient of x^4, we immediately notice that for $c < 0$ the potential $V \to -\infty$ as $x \to \pm\infty$, and thus there can be no bound states. However, for $c \geq 0$ there can be bound states. This means that if we try perturbation theory at $c = 0$, it should diverge because there is a singularity right on the negative x-axis that prevents any kind of series expansion at $c = 0$. And we will indeed show that perturbation theory diverges for even small values of c, however even in this situation eigenvector continuation manages to extrapolate correctly.

3.2 Eigenvector Continuation Applied to the Anharmonic Oscillator

If we try to think of eigenvector continuation as an analytic continuation, it still makes no sense to analytically continue if the starting circle has zero radius of convergence, and one would guess that eigenvector continuation should also fail. Yes, in theory we do have zero radius of convergence, but the anharmonic oscillator problem also has an infinite dimensional basis. The basis eigenstates $|n\rangle$ go from $n = 0$ to $n = \infty$. In a practical calculation, we

work with finite basis, and this truncation causes the singular point to be not exactly at the origin, but a little bit away from origin. As we include more basis points, the singularity comes closer to the origin, and converges to origin as our matrices dimension $N \to \infty$. In our calculation and results, we take $N = 200$.

So eigenvector continuation does have a small circle from where we can analytically continue. This might make it look like that perturbation theory also will converge in that small circle, however, as we take higher orders in perturbation theory, we need information further and further away from origin. To see this, let us just write a few derivatives of any function $f(c)$ at $c = 0$,

$$f'(x) = \frac{-f(x-h) + f(x+h)}{2h}$$

$$f''(x) = \frac{-f(x-2h) + 16f(x-h) - 30f(x) + 16f(x+h) - f(x+2h)}{12h^2}$$

$$f'''(x) = \frac{f(x-3h) - 8f(x-2h) + 14f(x-h) - 10f(x) + f(x+h) + 8f(x+2h) - f(x+3h)}{4h^3}$$

Here we are using two-sided derivatives. As we can see for higher derivatives we need points further away from the point where derivatives is being calculated ($f(x \pm 2h)$ for second derivative, $f(x \pm 3h)$ for third derivative, and so on). What this means is that in that small circle, perturbation theory will converge for a few orders, but as we calculate higher order terms, the series starts to diverge because eventually higher order terms need information outside the circle. No matter where we calculate using perturbation theory at the origin, it will eventually diverge as we go to higher order. We show this result in figure 3.1.

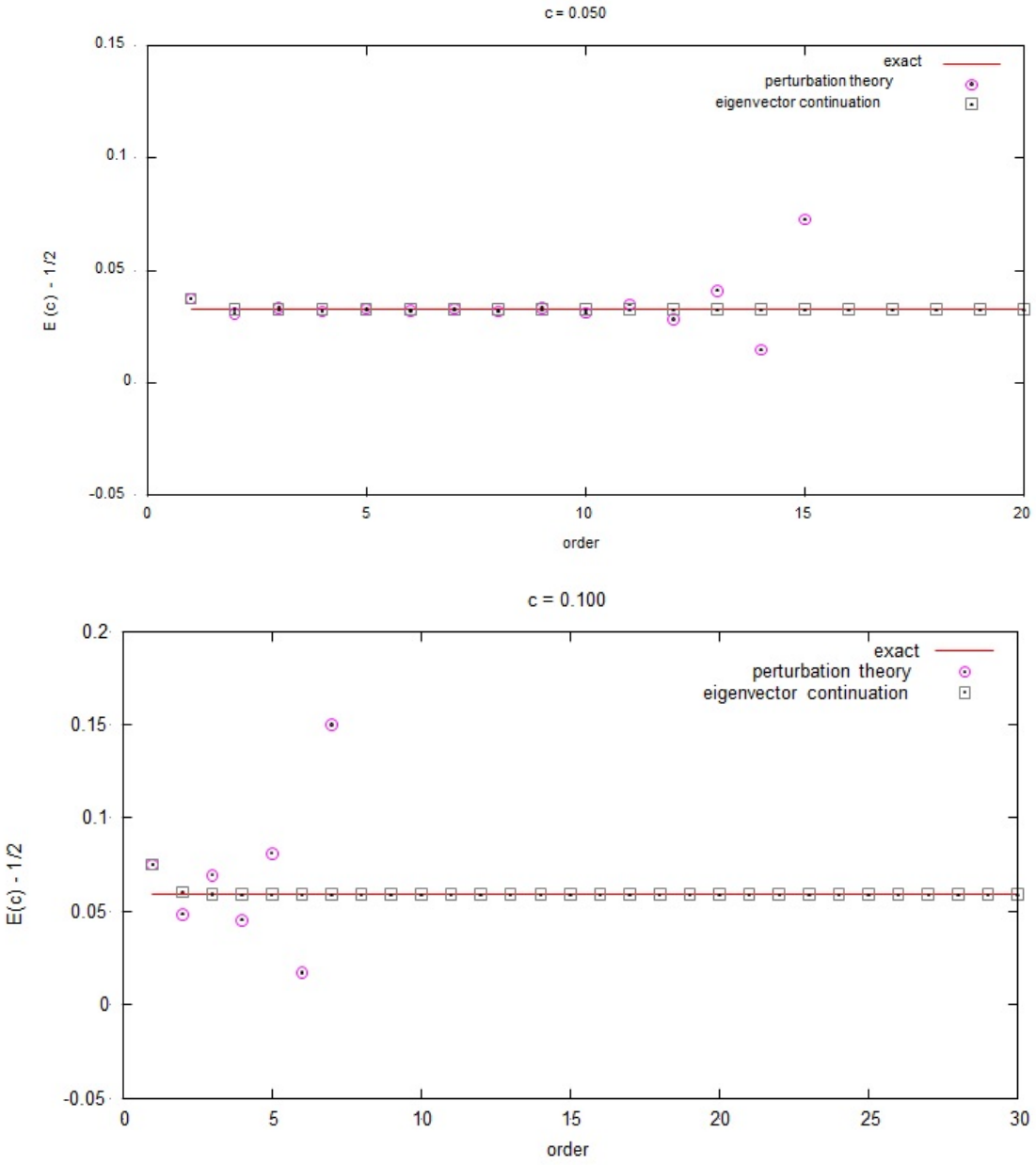


Figure 3.1: Convergence of ground state energy with different methods for $c_t = 0.05$ and 0.1. Perturbation theory fails, but eigenvector continuation does not. Notice that perturbation theory works for a few orders, and then diverges eventually.

Now we will explain how eigenvector continuation results were calculated as shown in figure 3.1. We have already described the basics of eigenvector continuation and how it is applied in chapter 2. Here we mention how we choose our training points. In order to make a fair comparison with perturbation theory, we take the same training eigenvectors

as perturbation theory. Perturbation theory tries to expand in a series around the point of perturbation, and tries to approximate by truncating the series up to a certain point. We have seen this in previous section in equation 2.6, and we write a similar expansion for perturbation theory at $c = 0$, but this time truncated to order N,

$$|\psi(c)\rangle_{PT} = \sum_{n=0}^{N} |\psi^{(n)}(0)\rangle \frac{c^n}{n!} \tag{3.7}$$

where $|\psi^{(n)}(0)\rangle$ denotes the n^{th} derivative of the eigenvector with respect to the parameter c, and evaluated at $c = 0$.

So the N^{th} order perturbation theory tries to approximate the actual eigenvector as a linear combination of the N derivatives of the ground state eigenvector $\{|\psi^{(0)}(0)\rangle, \cdots, |\psi^{(N)}(0)\rangle\}$. However, the coefficients for this linear sum are fixed. If we performed eigenvector continuation with the same set of derivatives $\{|\psi^{(0)}(0)\rangle, \cdots, |\psi^{(N)}(0)\rangle\}$, then eigenvector continuation will also try to estimate the exact target eigenvector by a linear combination of those same N derivatives of the ground state eigenvector at $c = 0$. This time however, the coefficients are not fixed, and eigenvector continuation tries to vary these coefficients to get the best approximation. From this perspective, it seems natural that eigenvector continuation will outperform the standard perturbation theory.

This is exactly how we do for our eigenvector continuation calculation. For N^{th} order EC calculation, we use the N derivatives of the ground state eigenvector at $c = 0$, $\{|\psi^{(0)}(0)\rangle, \cdots, |\psi^{(N)}(0)\rangle\}$ as our N training eigenvectors. As we mentioned before, higher order derivatives of the eigenvector at the origin have information of the eigenvector near the origin, with higher the order of derivative, the further away information we have. So our choice of training vectors $\{|\psi^{(0)}(c)\rangle, \cdots, |\psi^{(N)}(c)\rangle\}$ is equivalent to taking N training

vectors close to the origin. However, the derivatives are a systematic way to adding more training vectors, rather than randomly choosing training points near origin, and we stick to choosing higher order derivatives as our training vectors for eigenvector continuation. This idea of choosing training points will be important later when we discuss the convergence of eigenvector continuation in next chapter.

We have established through figure 3.1 that eigenvector continuation works and is able to converge to the correct result, whereas perturbation theory fails. We show in figure 3.2 that eigenvector continuation is also able to extrapolate to points far away from the origin. This might be surprising since our original circle where we can analytically continue from is abysmally small. However the idea of single analytic continuation can be extended to several analytic continuations, drawing bigger and bigger circles each time. This idea of multiple analytic continuations is illustrated in figure 3.3.

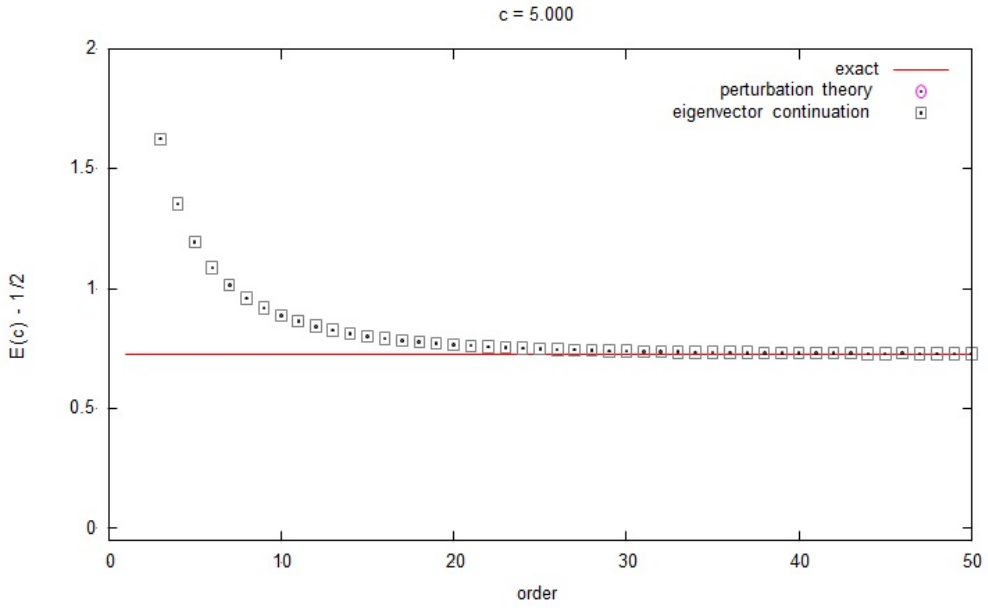


Figure 3.2: Eigenvector continuation converges for $c_t = 5$, a point far away from origin. At this point perturbation theory diverges right away.

To keep this section shorter, and since it is not directly related to the results we are

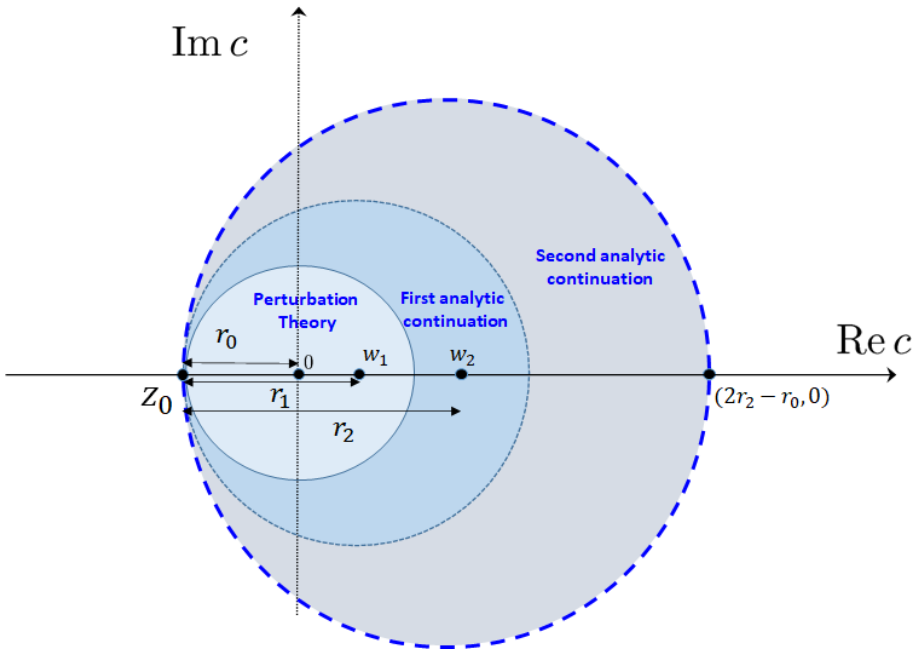


Figure 3.3: Even though we have a singularity z_0 in the negative c axis, we can analytically continue to any point on the positive c axis with the help of multiple analytic continuations.

presenting, we describe in the supplemental materials, how we computationally calculate the derivatives of the ground state eigenvector.

Although figures 3.1 and 3.2 show how the eigenvector continuation method converges as we add more training points, we learn more details when we look at it in a log graph. The first thing we observe is that for the first few orders there is an exponential convergence (which is the best type of convergence we want). This is shown in figure 3.4. However, as we go higher in order of eigenvector continuation, we notice that the convergence accelerates after some time (see figure 3.5). This can be understood intuitively by the fact that as we perform analytic continuation and go further away from our singularities, we can perform the next analytic continuation with larger radii, which accelerates the convergence rate.

We can compare the error convergence of eigenvector continuation to other simple methods, like direct diagonalization of the Hamiltonian matrix. Now direct diagonalization of a large matrix can be difficult, however we can employ a truncation trick to simply the process.

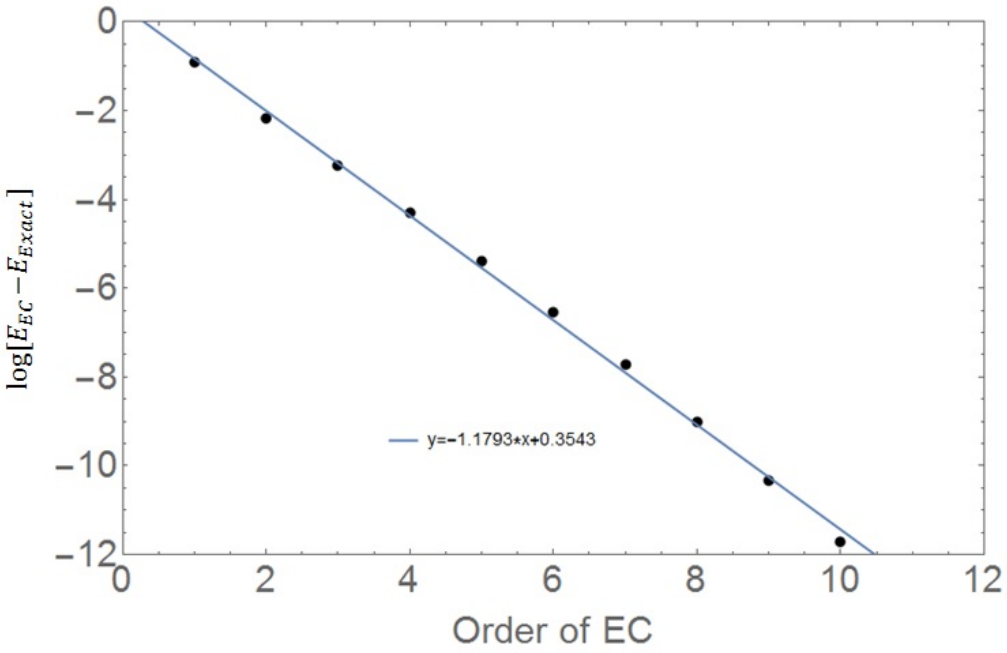


Figure 3.4: Exponential convergence in eigenvector continuation.

Our Hamiltonian matrix might be a huge n-dimensional matrix, but we simply truncate the matrix to the lowest N states, and diagonalize a $N \times N$ Hamiltonian matrix, where N can be a small number. As we increase N and it approaches the dimension of the original Hamiltonian matrix, the eigenvalue solution of the truncated matrix will converge exponentially to the eigenvalue solution of the actual matrix. However, the numerical complexity grows higher as we increase the dimension. Furthermore, if our original unperturbed Hamiltonian is diagonal and we add a non-diagonal perturbation matrix, the numerical complexity increases with strength of the perturbation c. So, this direct method may not work with high dimensions of matrices and high perturbation strength. However, for small enough dimensions and perturbation strength, we can compare how the diagonalization method works versus the eigenvector continuation. As we mentioned earlier, our current full dimension of Hamiltonian matrix is 200.

We can now compare how the convergence of direct orthogonalization of truncated

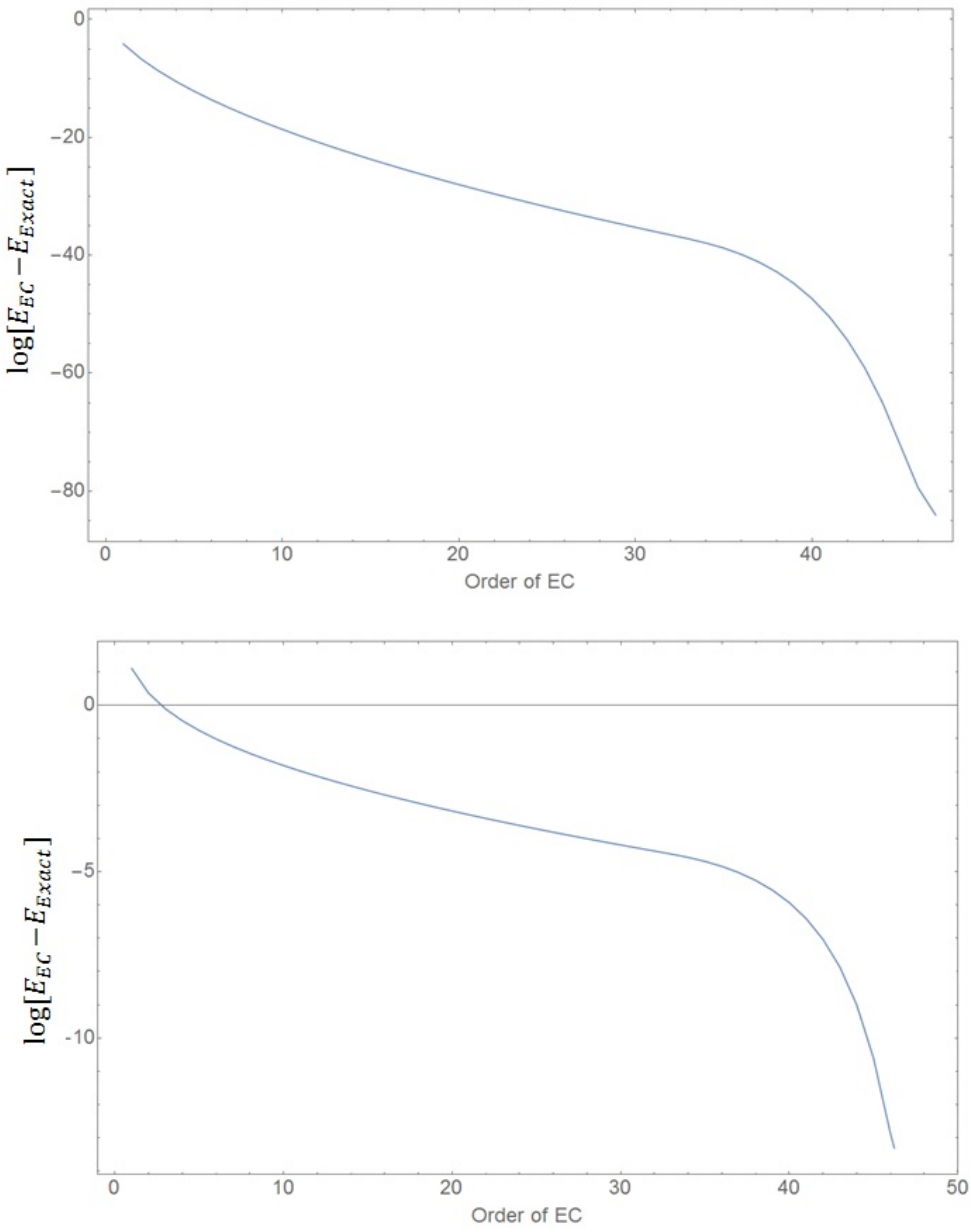


Figure 3.5: Convergence of the eigenvector continuation method with c = 0.1 (top) and 5 (bottom). The convergence accelerates as we analytically continue further away from our point of singularity because for each successive analytic continuation we can draw bigger circles where our series will converge.

Hamiltonian depends on truncation size N to the convergence of eigenvector continuation as a function of number of training points N. Figure (3.6) shows the result. Here we have also included the case where we perform eigenvector continuation with training vectors randomly

chosen at small c. These eigenvectors contain more information of the subspace than the case where we choose training vectors from perturbative expansion, because they span a larger space along the c axis. This is why we see this kind of eigenvector continuation (with training points near origin) being better than perturbative EC (with training vectors as derivatives of eigenvector at $c = 0$).

We should also clarify a small detail in choosing training eigenvectors for eigenvector continuation, when we use eigenvectors of the Hamiltonian at small perturbation strengths as training eigenvectors. As we mentioned before, the eigenvector continuation method needs to "learn" the low-dimensional subspace through the different training eigenvectors that we choose for the method. When we choose our training eigenvectors to be the exact eigenvectors at K small parameter values, say $\{c_1, \cdots, c_K\}$, we obtain a basis $\{|\psi(c_1)\rangle, \cdots, |\psi(c_K)\rangle\}$, and the method projects the Hamiltonian onto the subspace spanned by this basis to get an approximate result. To get better results, we perform a Gram-Schmidt orthogonalization on that set of K eigenvectors $\{|\psi(c_1)\rangle, \cdots, |\psi(c_K)\rangle\}$, and remove (if any) vectors that are pointing in almost the same direction. The removal is justified by the fact that such vectors do not give much additional information about the subspace. This is often the case if we choose our c_i very close together and the total Hamiltonian and its eigenvectors do not change much within the small variation of c_i.

We will now try to mathematically estimate the asymptotic error convergence with eigenvector continuation and analytic continuation. But before we begin, we remind ourselves that the singularity that causes perturbation theory to diverge is due to the avoided level crossings of the eigenvectors corresponding to different eigenvalues. These level crossings for the even parity states of anharmonic oscillator is shown in figure 3.7. As we can see, several different states cross with each other. Although it looks like the different lines intersect with

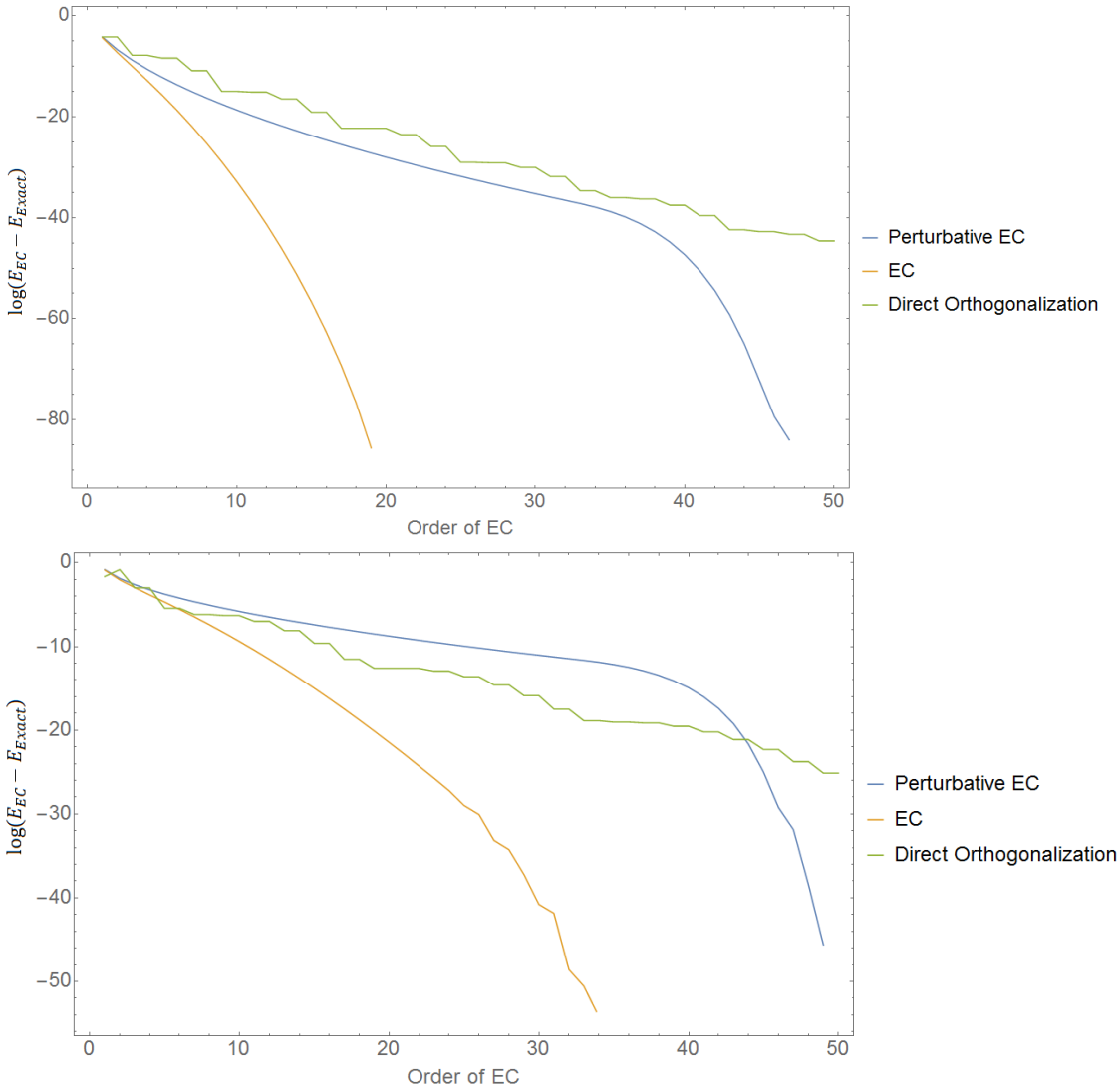


Figure 3.6: Comparison of error convergence for direct orthogonalization of truncated Hamiltonian with dimension N, and eigenvector continuation with order N. Perturbative EC refers to the case where we take the derivatives of the eigenvector at $c = 0$ as the training points. The other EC refers to the case where we random select training points at small c. For the top figure, $c_t = 0.1$ and for the bottom figure $c_t = 0.1$. We consistently see eigenvector continuation outperforming direct orthogonalization.

each other, actually all the crossings are the avoided-level crossing, and they do not intersect.

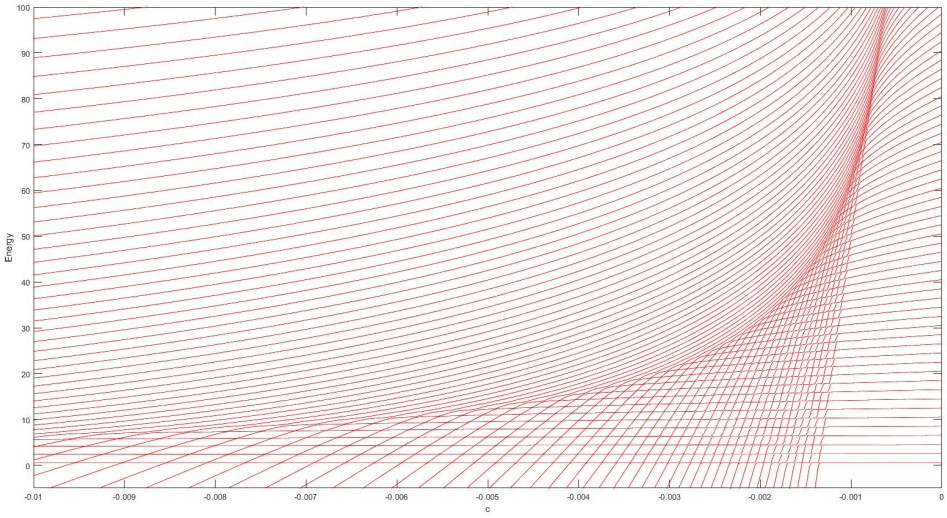


Figure 3.7: Level crossing for even states along negative real axis of c.

3.3 Error Bound for General Analytic Continuation

Consider a general case where z_0 and $\bar{z}_0$ are the nearest singularities to the origin. Let w_1 be a real number such that $|w_1| < |z_0|$, and let z_1 and $\bar{z}_1$ be the nearest singularities to the point w_1. Let c be a real number such that $|c - w_1| < |z_1 - w_1|$.

We start with the convergent series

$$|\psi(c)\rangle = \sum_{n_1=0}^{\infty} |\psi^{(n_1)}(w_1)\rangle \frac{(c-w_1)^{n_1}}{n_1!}. \tag{3.8}$$

Since z_1 and $\bar{z}_1$ are the nearest singularities to the point w_1, they sit at the convergence radius for this series expansion. We therefore know that for any positive ρ_1 such $\rho_1 < |z_1 - w_1|$, there exists a finite positive number $B_1(\rho_1)$ that gives an upper bound,

$$\left| |\psi^{(n_1)}(w_1)\rangle \right| \frac{\rho_1^{n_1}}{n_1!} < B_1(\rho_1), \tag{3.9}$$

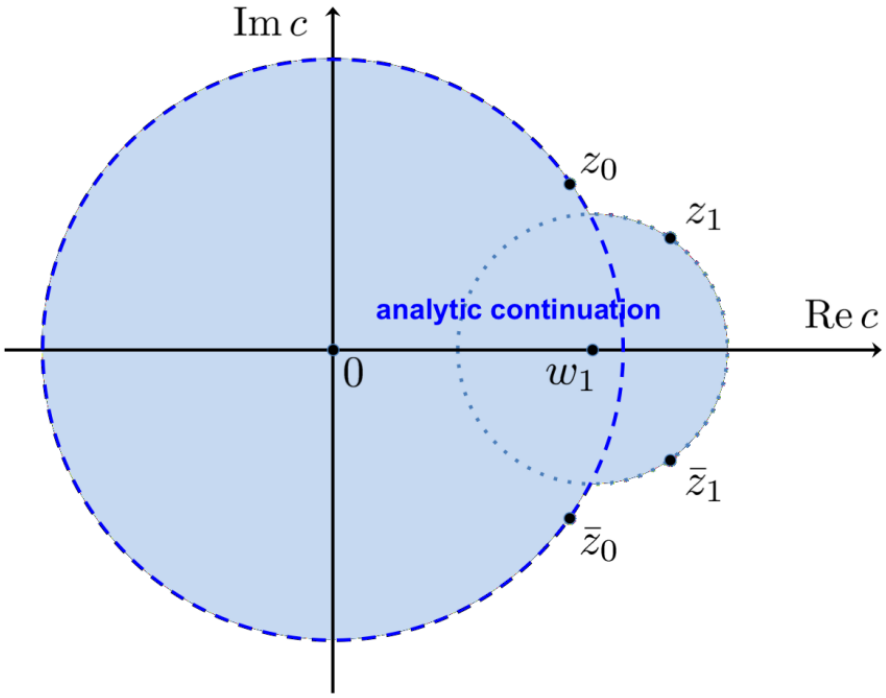


Figure 3.8: Analytic continuation of a general wave function $|\psi(c)\rangle$ with singularities at z_0, z_1 and their complex conjugates.

for all n_1. Let the remainder $|R_1(c, w_1)\rangle_{N_1}$ be defined as

$$|R_1(c, w_1)\rangle_{N_1} = |\psi(c)\rangle - [|\psi(c, w_1)\rangle]_{N_1}\,, \tag{3.10}$$

where

$$[|\psi(c, w_1)\rangle]_{N_1} = \sum_{n_1=0}^{N_1} |\psi^{(n_1)}(w_1)\rangle \frac{(c - w_1)^{n_1}}{n_1!}. \tag{3.11}$$

We then have

$$|R_1(c, w_1)\rangle_{N_1} = \sum_{n_1=N_1+1}^{\infty} |\psi^{(n_1)}(w_1)\rangle \frac{(c - w_1)^{n_1}}{n_1!}, \tag{3.12}$$

which satisfies the bound

$$\begin{aligned}\left||R_1(c,w_1)\rangle_{N_1}\right| &< \sum_{n_1=N_1+1}^{\infty} \left||\psi^{(n_1)}(w_1)\rangle\right| \frac{|c-w_1|^{n_1}}{n_1!} \\ &< \sum_{n_1=N_1+1}^{\infty} B_1(\rho_1) \left(\frac{|c-w_1|}{\rho_1}\right)^{n_1} \\ &< B_1(\rho_1)\frac{\rho_1}{\rho_1-|c-w_1|}\left(\frac{|c-w_1|}{\rho_1}\right)^{N_1+1}, \end{aligned} \tag{3.13}$$

for any positive ρ_1 with $\rho_1 < |z_1 - w_1|$.

We now consider the finite sum,

$$[|\psi(c,w_1)\rangle]_{N_1} = \sum_{n_1=0}^{N_1} |\psi^{(n_1)}(w_1)\rangle \frac{(c-w_1)^{n_1}}{n_1!}. \tag{3.14}$$

We can write $|\psi^{(n_1)}(w_1)\rangle$ as the convergent series

$$|\psi^{(n_1)}(w_1)\rangle = \sum_{n_0=0}^{\infty} |\psi^{(n_0+n_1)}(0)\rangle \frac{w_1^{n_0}}{n_0!}. \tag{3.15}$$

Since z_0 and $\bar{z}_0$ are the nearest singularities to the origin, they sit at the radius of convergence for this series expansion. Therefore we know that for any positive ρ_0 with $\rho_0 < |z_0|$, there exists a finite positive number $B_0(\rho_0)$ such that

$$\left||\psi^{(n_0+n_1)}(0)\rangle\right| \frac{\rho_0^{n_0}}{n_0!} < B_0(\rho_0) \tag{3.16}$$

for all n_0 and for all n_1 from 0 to N_1. There also exists a finite positive number $\tilde{B}_0(\rho_0)$ such

that

$$\sum_{n_1=0}^{N_1} \left| |\psi^{(n_0+n_1)}(0)\rangle \right| \frac{\rho_0^{n_0}|c-w_1|^{n_1}}{n_0!n_1!} < \tilde{B}_0(\rho_0). \tag{3.17}$$

Let the remainder $|R_0(c,w_0,w_1)\rangle_{N_0,N_1}$ be defined as

$$|R_0(c,w_0,w_1)\rangle_{N_0,N_1} = [|\psi(c,w_1)\rangle]_{N_1} - [|\psi(c,w_0,w_1)\rangle]_{N_0,N_1}\,, \tag{3.18}$$

where

$$[|\psi(c,w_0,w_1)\rangle]_{N_0,N_1} = \sum_{n_1=0}^{N_1}\sum_{n_0=0}^{N_0} |\psi^{(n_0+n_1)}(0)\rangle \frac{w_1^{n_0}(c-w_1)^{n_1}}{n_0!\,n_1!}. \tag{3.19}$$

The remainder $|R_0(c,w_0,w_1)\rangle_{N_0,N_1}$ is then

$$|R_0(c,w_0,w_1)\rangle_{N_0,N_1} = \sum_{n_1=0}^{N_1}\sum_{n_0=N_0+1}^{\infty} |\psi^{(n_0+n_1)}(0)\rangle \frac{w_1^{n_0}(c-w_1)^{n_1}}{n_0!\,n_1!}, \tag{3.20}$$

and this satisfies the bound

$$\begin{aligned} \left| \left| R_0(c,w_0,w_1)\rangle_{N_0,N_1} \right| \right. &< \sum_{n_1=0}^{N_1}\sum_{n_0=N_0+1}^{\infty} \left| |\psi^{(n_0+n_1)}(0)\rangle \right| \frac{|w_1|^{n_0}|c-w_1|^{n_1}}{n_0!\,n_1!}, \\ &< \sum_{n_0=N_0+1}^{\infty} \tilde{B}_0(\rho_0) \left(\frac{|w_1|}{\rho_0}\right)^{n_0} \\ &< \tilde{B}_0(\rho_0)\frac{\rho_0}{\rho_0-|w_1|}\left(\frac{|w_1|}{\rho_0}\right)^{N_0+1}. \end{aligned} \tag{3.21}$$

Putting everything together, we have the result

$$\begin{aligned} \left| |\psi(c)\rangle - [|\psi(c,w_0,w_1)\rangle]_{N_0,N_1} \right| &< \tilde{B}_0(\rho_0)\frac{\rho_0}{\rho_0-|w_1|}\left(\frac{|w_1|}{\rho_0}\right)^{N_0+1} \\ &\quad + B_1(\rho_1)\frac{\rho_1}{\rho_1-|c-w_1|}\left(\frac{|c-w_1|}{\rho_1}\right)^{N_1+1}, \end{aligned} \tag{3.22}$$

where

$$[|\psi(c,w_0,w_1)\rangle]_{N_0,N_1} = \sum_{n_1=0}^{N_1}\sum_{n_0=0}^{N_0} |\psi^{(n_0+n_1)}(0)\rangle \frac{w_1^{n_0}(c-w_1)^{n_1}}{n_0!\,n_1!}. \tag{3.23}$$

If we now choose $\rho_0 = |z_0| - \epsilon_0$ for small positive ϵ_0 and choose $\rho_1 = |z_1 - w_1| - \epsilon_1$ for small positive ϵ_1, then we have the asymptotic bounds

$$\Big||\psi(c)\rangle - [|\psi(c,w_0,w_1)\rangle]_{N_0,N_1}\Big| = O\left(\frac{|w_1|}{|z_0|-\epsilon_0}\right)^{N_0+1} + O\left(\frac{|c-w_1|}{|z_1-w_1|-\epsilon_1}\right)^{N_1+1}, \tag{3.24}$$

in the limits $N_1 \to \infty$ followed by $N_0 \to \infty$.

If we were to analytically continue two times (at center w_1 and w_2), then the asymptotic error bound would be given by,

$$\Big||\psi(c)\rangle - [|\psi(c,w_0,w_1)\rangle]_{N_0,N_1}\Big| = O\left(\frac{|w_1|}{|z_0|-\epsilon_0}\right)^{N_0+1} \tag{3.25}$$

$$+O\left(\frac{|w_2-w_1|}{|z_1-w_1|-\epsilon_1}\right)^{N_1+1} + O\left(\frac{|c-w_2|}{|z_2-w_2|-\epsilon_1}\right)^{N_2+1} \tag{3.26}$$

Similarly, we could write an equation for n analytic continuations.

3.4 Error Bound for the Anharmonic Oscillator

Now, let us consider the case of anharmonic oscillator.

$$H = \frac{p^2}{2m} + \frac{m\omega^2 x^2}{2} + cx^4 \tag{3.27}$$

If we truncate our matrix dimension at some value N_{max}, then this would correspond to analyzing the anharmonic oscillator up to a certain maximum distance r_{max} from the

origin, which would go as $r_{max} \sim \sqrt{N_{max}}$. Since x^4 dominates x^2, if we go large enough x, the potential becomes negative for negative c and no bound state can exist (since $V \to -\infty$ as $x \to \infty$). Thus, we will have a singularity when c is such that,

$$\frac{m\omega^2 r_{max}^2}{2} + c r_{max}^4 \approx 0$$

Thus, the dependence of the point of singularity on N_{max} goes like $c \sim -1/N_{max}$.

When we truncate our Hilbert space at N_{max}, the matrix is finite-dimensional and Hermitian, so the eigenvector will be analytic on the real axis (real part of c). So, the singularity must appear on the complex plane, with non-zero $\mathrm{Im}(c)$. It was shown in [24] that the singularity lies close to the negative imaginary axis. We have tried to verify this by looking at the level crossing of the lowest two even parity states. This is shown in the figure 3.9, where the difference between the two lowest even parity state eigenvalues is shown over the complex plane near origin. We can see some nice avoided level crossings that we mentioned in figure 3.7.

Let z_0 be the real part of point of singularity, which would be situated along the negative real axis. Let it be at a distance of r_0 from the origin, which is related to the dimension of our Hamiltonian matrix N_{max}, by $r_0 \sim 1/N_{max}$. Then, we can perform the analytic continuation as described earlier. To extend the reach of the process, we can perform multiple analytic continuation. Although the actual point of singularity is at some point further than $|z_0|$, we will perform the following analysis assuming that the singularity is at z_0. This choice actually limits our analytic continuation, as our radius of convergence would be a little larger than what we treat here, but we will show that even with moving the singularity a little closer to the origin, there still exists analytic continuation to any point on the real axis.

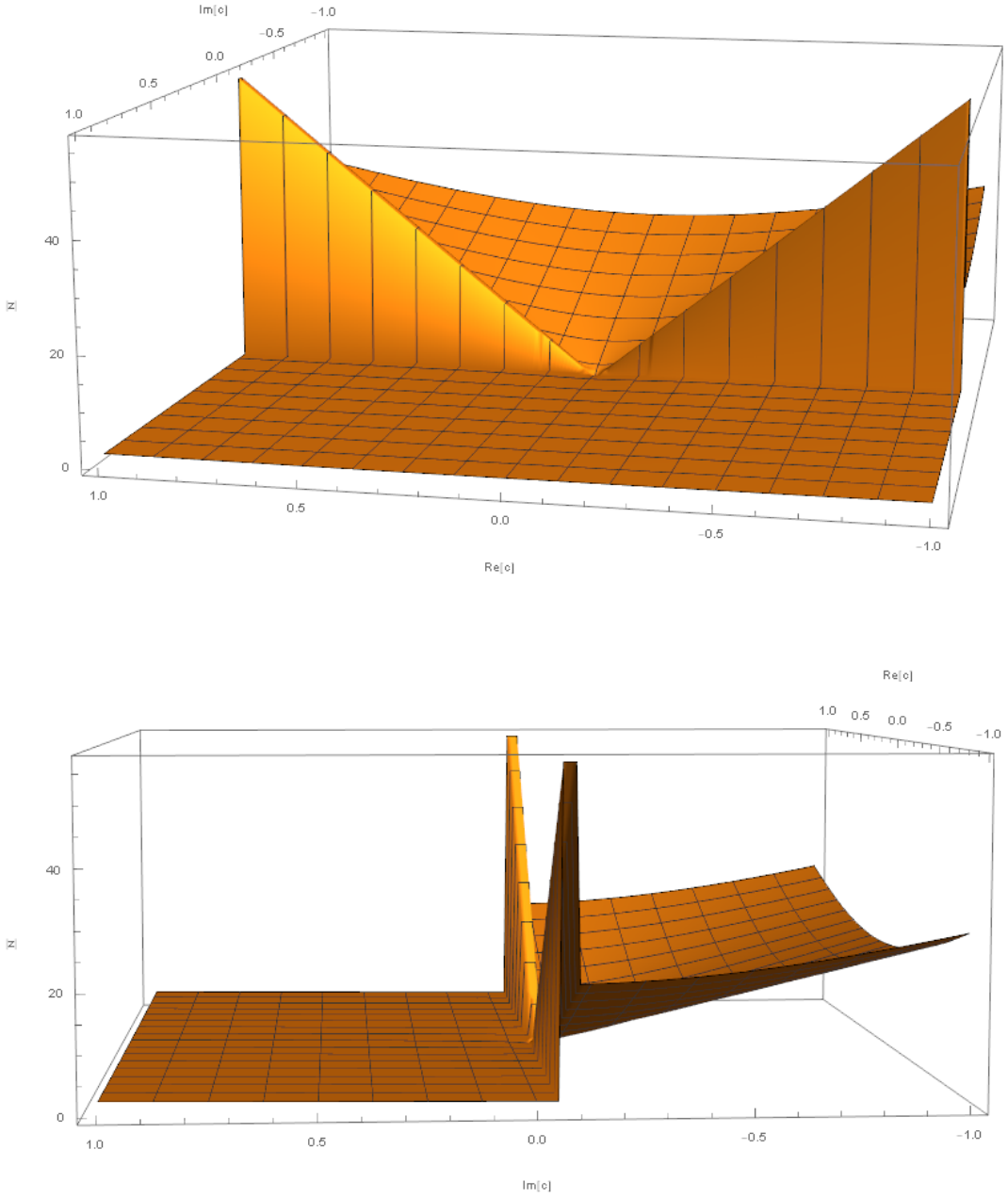


Figure 3.9: Difference of absolute values of lowest two even parity state eigenvalues.

For the first continuation, let us choose $w_1 = \alpha r_0 - r_0$ with $\alpha > 1$, and so w_1 lies along positive real axis (see figure 3.10). Then, we can analytically continue to a circle of radius $r_1 = \alpha r_0$. Now, we can perform the analytic continuation again, but this time the expansion

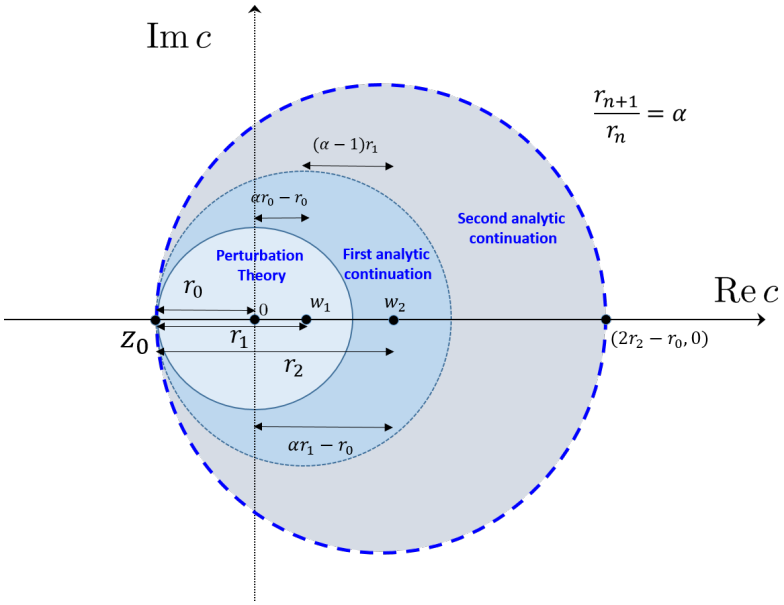


Figure 3.10: Multiple analytic continuation of the wave function $|\psi(c)\rangle$ for the anharmonic oscillator.

would be around w_1, instead of the origin. Let us choose $w_2 = \alpha r_1 - r_0$ (from the origin and along the real axis), so the radius of analytic continuation would be $r_2 = \alpha r_1 = (\alpha)^2 r_0$. Analytically continuing again, by taking the center at $w_3 = \alpha r_2 - r_0$ (expanding around w_2), this time we get the radius of convergence to be $r_3 = (\alpha)^3 r_0$. Repeating this process, we find that when we do the analytic continuation for the n^{th} time, we can analytically continue to a circle of radius $r_n = (\alpha)^n r_0$. The center of the circle is at $c = \alpha r_n - r_0$ on the real axis, and the rightmost point on the real axis during the n^{th} iteration is $2r_n - r_0$. Each time the next center of circle is $(\alpha - 1)r_n$ away from the previous center of the circle.

In the case of anharmonic oscillator we have $z_0 = z_1$, and thus using equation (3.24),

we can find the asymptotic error for one analytic expansion,

$$\begin{aligned}\Big||\psi(c)\rangle - [|\psi(c,w_0,w_1)\rangle]_{N_0,N_1}\Big| &= O\left(\frac{|w_1|}{|z_0|-\epsilon_0}\right)^{N_0+1} + O\left(\frac{|c-w_1|}{|z_0-w_1|-\epsilon_1}\right)^{N_1+1} \\ &= O\left(\frac{(\alpha-1)r_0}{r_0-\epsilon_0}\right)^{N_0+1} + O\left(\frac{(\alpha-1)r_1}{r_1-\epsilon_1}\right)^{N_1+1} \\ &\sim O\,(\alpha-1)^{N_0+1} + O\,(\alpha)^{N_1+1} \end{aligned} \tag{3.28}$$

Now, for two analytic continuations, then the asymptotic error would be,

$$\begin{aligned}\Big||\psi(c)\rangle - [|\psi(c,w_0,w_1,w_2)\rangle]_{N_0,N_1,N_2}\Big| &= O\left(\frac{|w_1|}{|z_0|-\epsilon_0}\right)^{N_0+1} + O\left(\frac{|w_2-w_1|}{|z_0-w_1|-\epsilon_1}\right)^{N_1+1} \\ &\quad + O\left(\frac{|c-w_2|}{|z_0-w_2|-\epsilon_1}\right)^{N_2+1}\end{aligned}$$

$$\begin{aligned}\Rightarrow \Big||\psi(c)\rangle - [|\psi(c,w_0,w_1,w_2)\rangle]_{N_0,N_1,N_2}\Big| &= O\left(\frac{(\alpha-1)r_0}{r_0-\epsilon_0}\right)^{N_0+1} + O\left(\frac{(\alpha-1)r_1}{r_1-\epsilon_1}\right)^{N_1+1} \\ &\quad + O\left(\frac{(\alpha-1)r_2}{r_2-\epsilon_1}\right)^{N_1+1} \\ &\sim O\,(\alpha-1)^{N_0+1} + O\,(\alpha-1)^{N_1+1} + O\,(\alpha-1)^{N_2+1}\end{aligned} \tag{3.29}$$

Continuing in this fashion, we find the asymptotic error for n analytic continuations is,

$$\begin{aligned}\Big||\psi(c)\rangle - [|\psi(c,w_0,w_1,...,w_n)\rangle]_{N_0,...,N_n}\Big| &\sim O\,(\alpha-1)^{N_0+1} + O\,(\alpha-1)^{N_1+1} \\ &\quad + ... + O\,(\alpha-1)^{N_n+1}\end{aligned} \tag{3.30}$$

From equation (3.30), it is clear that as long as $1 < \alpha < 2$, we let our order of eigenvector

continuation go to infinity and we choose $N_0, N_1, ..., N_n \to \infty$, the error will go to zero (the sum of N_i is the order of EC). Thus, this analytic continuation method always converges, even though perturbation theory diverges.

However, note that infinite $N_0, N_1, ..., N_n$ would correspond to infinite order of eigenvector continuation. Let us see what we can do in finite order of EC. Suppose we are interested in reaching near the point $c = 1$ on the real axis and ask ourselves the question how many analytic continuations do we need to reach that point. We find that the number of iterations n should be such that,

$$\begin{aligned} r_{n+1} - r_0 &\sim 1 \\ \Rightarrow (\alpha)^{n+1} r_0 - r_0 &\sim 1 \end{aligned} \tag{3.31}$$

Substituting the relation between r_0 and N_{max} (recall that N_{max} was our dimension of Hamiltonian matrix), we have

$$(\alpha)^{n+1} - 1 \sim N_{max} \tag{3.32}$$

and thus,

$$n \sim \frac{\ln(N_{max} + 1)}{\ln(\alpha)} - 1 \tag{3.33}$$

So, n should be an integer larger than the right hand side of equation (3.33).

Now suppose that we are interested in analytically continuing till $c = 1$ with a given order of eigenvector continuation, say M. We still need at least n separate analytic continuations, where n is given by equation (3.33). If each of those expansion series (like equation (3.8)) have N_i terms, then we have $\sum_{i=0}^{n} N_i = M$. With this constraint, we ask the question "What is the optimal choice of α to minimize error for given M, N_{max} and $c = 1$?"

With equation (3.30) in mind, we try to minimize,

$$\sum_{i=1}^{n}(\alpha-1)^{N_i+1}, \tag{3.34}$$

subject to the constraint,

$$\sum_{i=1}^{n} N_i = M \tag{3.35}$$

where n is the number of analytic continuation, given by equation (3.33).

This minimization does not entirely make sense since equation (3.30) is only defined up to order of magnitude and we are missing some constant factors in each sum term, which in principle could be anything. However, the solution of this minimization does give us an idea of what ideal α could look like. Utilizing symmetry in individual terms, we find that N_i are all equal with $N_i = M/n$, and thus we minimize $n(\alpha-1)^{M/n}$, with n given by (3.33). We solve the problem numerically and list our results for different M and N_{max}

N_{max}	M	Number of analytic continuations (n)	Optimum α
10	5	2	1.8171
10	10	4	1.6154
50	10	4	1.6154
50	20	6	1.5449
50	30	8	1.4646
50	40	9	1.4497
50	50	10	1.4297
100	10	8	1.6699
100	25	10	1.5213
100	50	12	1.4262
100	75	13	1.3905
100	100	13	1.3905

Table 3.1: The optimal choice for the number of analytic continuations and the corresponding α (ratio of radii while performing analytic continuations) required to minimize approximation error.

The results suggest that about $\alpha = 1.5$ seems to be a good ratio of radii between

iterations. To find an exact value for optimal α, we would need to define equation (3.30) more concretely, but we do not have an accurate knowledge of the wave-functions. However, our main goal in this derivation is to show that eigenvector continuation with given order does make sense, and the results do converge to the actual numbers as we increase our order of EC and we have indeed shown that.

Note that the choice of optimal α has no effect on our numerical calculation of eigenvector continuation results. We merely present a discussion to better understand the convergence of eigenvector continuation. Our eigenvector continuation method does not have α as a free parameter and we have no control over it. For a given order of EC, the only choice that we have is our selection of training eigenvectors for the method, and it is unclear how that indirectly chooses some α (and therefore n) for us.

Chapter 4

Convergence of Eigenvector Continuation

So far we have seen that eigenvector continuation works brilliantly to simplify our calculations, and allowing us to extrapolate and interpolate in the parameters of the Hamiltonian. One natural question that arises next is how to optimize our given eigenvector continuation calculation. We have a trade-off in accuracy of the emulator and the computational time it takes for emulation, and we want to get the best results with a given number of training points. So we want to know where should we take our training points to get best performance, and what order eigenvector continuation should be perform, i.e., how many training vectors we need. However, to answer these questions, we need to first understand the convergence properties of eigenvector continuation as we add more training points and as we vary the location of training points. In this chapter, we will discuss how eigenvector continuation converges as a function of number of training points, and we will gain additional insight into how eigenvector continuation works. In the next chapter we will come back to where we should take training points for best results.

The answer to what we are seeking are intuitively somewhat clear. Eigenvector continuation method learns the low-dimensional subspace in which the target eigenvector lives through the different training eigenvectors that we choose for the method. With a fixed number of training points, we want our training points to be further apart so that we have

more information about the eigenspace, and this will make eigenvector continuation learn the manifold more. And it is also clear that the more training eigenvectors we choose, the better our EC approximation will be.

No. of terms	Values of c_i	c_t	EC ground state energy	Exact Energy
3	0.01, 0.02, 0.03	0.4	0.672645280	0.668772604
3	0.02, 0.04, 0.06	0.4	0.670458737	0.668772604
3	0.03, 0.06, 0.09	0.4	0.669562096	0.668772604
4	0.01, 0.02, 0.03, 0.04	0.4	0.669503683	0.668772604
4	0.02, 0.04, 0.06, 0.08	0.4	0.668956488	0.668772604
5	0.01, 0.02, $\cdots$, 0.05	0.4	0.668898346	0.668772604
5	0.02, 0.04, $\cdots$, 0.10	0.4	0.668788797	0.668772604
10	0.01, 0.02, $\cdots$, 0.10	0.4	0.668772607	0.668772604
3	0.01, 0.02, 0.03	1	0.842860074	0.803770651
3	0.02, 0.04, 0.06	1	0.827563722	0.803770651
3	0.03, 0.06, 0.09	1	0.819258718	0.803770651
4	0.01, 0.02, 0.03, 0.04	1	0.817399692	0.803770651
4	0.02, 0.04, 0.06, 0.08	1	0.809913282	0.803770651
5	0.01, 0.02, $\cdots$, 0.05	1	0.808359378	0.803770651
5	0.02, 0.04, $\cdots$, 0.10	1	0.805211425	0.803770651
10	0.01, 0.02, $\cdots$, 0.10	1	0.803778746	0.803770651

Table 4.1: Eigenvector continuation in anharmonic oscillator. We calculate the exact eigenvectors at location c_i and use them as training eigenvectors for EC, to estimate the eigenvector at c_t. We see that EC approximation becomes better with more training eigenvectors, and with more spaced apart training points.

However, the error dependence of eigenvector continuation on the location of training points is non-trivial and extremely problem specific. It is hard to make general comments about the convergence behavior as we move our training points around. So it is very hard to answer for a general Hamiltonian where we should choose our training points. As to how many training points we need, it depends on how complicated our problem is, and how much accuracy we want. Let us look at this point through an example. Table 4.1 shows how eigenvector continuation performs as we vary the location and number of training points in the problem of the anharmonic oscillator.

And we have seen in the case of the anharmonic oscillator that as we add more training points, the error convergence looks exponential (see figure 3.4). Table 4.1 shows the data for that graph.

No. of terms	Values of c_i	EC energy	Exact Energy	Difference from Exact Energy
2	0.01, 0.02	0.917673992	0.803770651	0.113903341
3	0.01, 0.02, 0.03	0.842860074	0.803770651	0.039089423
4	0.01, 0.02, $\cdots$, 0.04	0.817399692	0.803770651	0.013629041
5	0.01, 0.02, $\cdots$, 0.05	0.808359378	0.803770651	0.004588727
6	0.01, 0.02, $\cdots$, 0.06	0.805234149	0.803770651	0.001463498
7	0.01, 0.02, $\cdots$, 0.07	0.804209584	0.803770651	0.000438933
8	0.01, 0.02, $\cdots$, 0.08	0.803894175	0.803770651	0.000123524
9	0.01, 0.02, $\cdots$, 0.09	0.803803266	0.803770651	3.26149E-05
10	0.01, 0.02, $\cdots$, 0.10	0.803778746	0.803770651	8.09511E-06
12	0.01, 0.02, $\cdots$, 0.12	0.803771231	0.803770651	5.7948E-07

Table 4.2: Energy Error with eigenvector continuation in the anharmonic oscillator at $c_t = 1$. It converges exponentially as we increase the order of the method (see figure 3.4).

So the eigenvector continuation error depends on the number of training points and the location of the training points. As we mentioned in the beginning of this chapter, there are two questions - how many training points we need, and where to choose our training points. It is hard to answer them together because the error convergence depends on both of them in a complicated way. We simplify the problem and study them separately. We study the error convergence with respect to number of training points, and separately consider the question of how to choose our training points, when we have a fixed number of them. We consider the latter question in the next chapter. It is hard to answer the question of where to choose our N training points for any given problem, but we can come up with an algorithm to iteratively choose the best location of the next training point, and we can start at one or two random training points and repeat the algorithm as many times as we want.

In this chapter, we will consider the answer to the former question - the error dependence

on the number of training points. However, since the error varies so much as we vary the location of a training point, it makes no sense to study the error convergence as we add another training point, because the error depends on where we add the next training point. So if we want to study the error convergence with respect to number of training points, we need a systematic way to add our next training point. And we have seen one systematic way in the anharmonic oscillator problem in the previous chapter. We were taking the derivatives of the eigenvector at $c = 0$ and using them as our training eigenvectors. Now higher order derivatives of the eigenvector at origin are well defined, and it makes sense to ask questions like how does my eigenvector continuation approximation improve as we add one more higher order derivative to the training set. This systematic way is also great way to extend to large number of training points, because we want to know the asymptotic error convergence of the method.

To study this idea of adding higher order derivatives to the training set mathematically, we introduce a more generalized form of eigenvector continuation, and we call it vector continuation (VC). We generalize the problem from being interested in eigenvector of a particular n-dimensional Hamiltonian with a variable parameter c, to being interested in a vector living in n-dimensional vector space, parameterized by c. Our n-dimensional vector could be the eigenvector of a n-dimensional Hamiltonian, and in that case we go back from vector continuation to eigenvector continuation. Vector continuation allows us to redefine our problem more mathematically, and this generalization help us understand eigenvector continuation's convergence properties in a general sense. After we derive our results, we will show that vector continuation when applied to our usual problem of finding eigenvectors of a parameterized Hamiltonian gives us the same result as that of eigenvector continuation.

4.1 Vector Continuation

Consider a n-dimensional vector space V parameterized by c. We first consider c to be one-dimensional parameter. The ideas presented here can be extended to a vector space parameterized by multi-dimensional c. Let us denote any vector in this space by $|v(c)\rangle$. We assume that this vector space varies smoothly as we vary c. Since this is a smooth space in c, we can apply all our tools from analysis.

Suppose we are interested at knowing the vector $|v(c_t)\rangle$ at a particular value of the parameter c_t, but we only have the information of the vector at specified training points $c_i = \{c_1, \cdots, c_k\}$. Let us denote the training vectors as $|v(c_0)\rangle, \cdots, |v(c_k)\rangle$.

Like eigenvector continuation, in vector continuation we approximate $|v(c_t)\rangle$ as a linear combination of vectors $|v(c_0)\rangle, \cdots, |v(c_N)\rangle$ at training points $c = c_0, \cdots, c_N$. The difference is that in vector continuation we construct the best approximation by projecting $|v(c_t)\rangle$ onto the subspace spanned by the training point vectors. While this sounds similar to eigenvector continuation, the way we implement it in our code is different. Vector continuation is a simpler process than the variational calculation used in eigenvector continuation. However, we should emphasize that since it requires knowledge of the target eigenvector, vector continuation should be viewed as a diagnostic tool to study error convergence, rather than a practical method for determining $|v(c_t)\rangle$. We will show that eigenvector continuation and vector continuation have identical convergence properties, and so it suffices to understand the convergence properties of vector continuation. As the name suggests, vector continuation can also be generalized to any smooth vector path $|v(c)\rangle$ without reference to Hamiltonian matrices or eigenvectors.

We want to understand the asymptotic convergence of eigenvector and vector contin-

uation at large orders. As we mentioned earlier, we need a systematic way to add more training points. To do this we consider a sequence of training points $c_0, \cdots, c_N$ with a well-defined limit point c_{lim} for large N that is some distance away from the target point c_t. Without loss of generality we can redefine c_{lim} so that our limit point corresponds to $c = 0$. In the limit where the training points accumulate around $c = 0$, we can replace our training vectors $|v(c_0)\rangle, \cdots, |v(c_N)\rangle$ with the derivatives of $|v(c)\rangle$ at $c = 0$, which we write as $|v^{(0)}(0)\rangle, \cdots, |v^{(N)}(0)\rangle$. These derivative vectors approximately span the same $N + 1$-dimensional subspace as our original training vectors. We explained the reasoning behind this in the previous chapter. And there is an important detail here that we should mention. Although we are taking the limit of large N, the values of N we take are always vastly smaller than the number of dimensions of our linear space.

Our choice of training points provides a good definition for the convergence properties at large orders. It can be viewed as the worst possible choice of training points for best convergence since all of our training points are clustered in one area, and we need to extrapolate to a target point located somewhere else. When the training points are spread apart and not clustered, the convergence is generally much faster, especially if the training points surround the target point from all sides. We have seen this in the example of anharmonic oscillator. However, as we mentioned before, the convergence for this more general case is highly dependent on the positions of the training points, and it is difficult to make a clean definition of asymptotic convergence. Therefore, for now, with our choice of training points, we try to learn the worst possible error convergence for large N. We have already seen the convergence is faster in the one parameter case of the anharmonic oscillator when the training points are spread out, and later we will show that convergence for the multi-parameter case, it is much faster if there exists a smooth curve connecting some subset of the training

points to the target point.

We will now compare the error convergence of vector continuation, eigenvector continuation, and perturbation theory. First we define what are our three different approximations to the target eigenvector $|v(c_t)\rangle$ in the three different methods are. For the given target eigenvector $|v(c_t)\rangle$, we denote the order-N eigenvector continuation approximation as $|v(c_t)\rangle_N^{\mathrm{EC}}$, the order-$N$ vector continuation approximation as $|v(c_t)\rangle_N^{\mathrm{VC}}$, and the N^{th} order perturbative theory approximation as $|v(c_t)\rangle_N^{\mathrm{PT}}$. For eigenvector continuation to make sense, in the following we assume that the vector $|v(c)\rangle$ is actually the eigenvector of some Hamiltonian $H(c)$. As we will soon see, vector continuation does not care about $H(c)$ and works with only $|v(c_t)\rangle$ (assuming it is given somehow), whereas eigenvector continuation works with the Hamiltonian $H(c)$.

All three methods work by approximating the target eigenvector as a linear combination of the training eigenvectors. Therefore, to make a fair comparison, we use the same set of training vectors $\{|v^{(0)}(0)\rangle, \cdots, |v^{(N)}(0)\rangle\}$ for eigenvector continuation and vector continuation.

So we have our set of training derivative vectors $|v^{(0)}(0)\rangle, \cdots, |v^{(N)}(0)\rangle$, and we perform Gram-Schmidt orthogonalization to get a set of orthonormal vectors, which we call $|w^{(0)}(0)\rangle, \cdots, |w^{(N)}(0)\rangle$. Here we are assuming that the derivative vectors are linearly independent, and therefore our orthonormal set also has size N. This assumption is generally true for almost all of the practical problems we encounter in physics. With this orthonormal basis $|w^{(0)}(0)\rangle, \cdots, |w^{(N)}(0)\rangle$, we define our vector continuation approximation $|v(c_t)\rangle_N^{\mathrm{VC}}$ as

$$|v(c_t)\rangle_N^{\mathrm{VC}} = \sum_{n=0}^{N} \langle w^{(n)}(0)|v(c_t)\rangle \, |w^{(n)}(0)\rangle \, . \tag{4.1}$$

Notice that we have just projected the target eigenvector onto the individual (orthonormal) training eigenvectors. Therefore, to perform this we need the information about the target eigenvector $|v(c_t)\rangle$. As a reminder, vector continuation is not a practical method to extrapolate to a target location. It is only a diagnostic tool to compare and understand error convergence.

In the same orthonormal basis of training vectors, we can also write the eigenvector continuation approximation $|v(c_t)\rangle_N^{\mathrm{EC}}$ as

$$|v(c_t)\rangle_N^{\mathrm{EC}} = \sum_{n=0}^{N} a(c_t, n, N)\, |w^{(n)}(0)\rangle\,, \tag{4.2}$$

where the coefficients $a(c_t, n, N)$ are found by minimizing the expectation value of $H(c_t)$.

And we can consider perturbation theory (PT) around the point $c = 0$. If the nearest branch point to $c = 0$ is z_0, then the series expansion

$$|v(c_t)\rangle = \sum_{n=0}^{\infty} |v^{(n)}(0)\rangle\, \frac{c_t^n}{n!} \tag{4.3}$$

will converge for $|c_t| < |z_0|$ and diverge for $|c_t| > |z_0|$. We define the perturbation theory approximation $|v(c_t)\rangle_N^{\mathrm{PT}}$ as the partial series truncated at order $n = N$,

$$|v(c_t)\rangle_N^{\mathrm{PT}} = \sum_{n=0}^{N} |v^{(n)}(0)\rangle\, \frac{c_t^n}{n!}. \tag{4.4}$$

In this analysis we have assumed that the radius of convergence is greater than zero. This follows from the fact that $H(c)$ is a finite-dimensional Hermitian matrix. And we have made similar observation with the case of anharmonic oscillator. For the following discussion, we assume c_t is within our radius of convergence of perturbation theory, and thus equation 4.4

is well defined and converges.

We define the error in these three approximations to $|v(c_t)\rangle$ by computing the norm of the residual vector, which is the difference of exact vector and its approximation. When we look at the norm of the residual vector as a function N, we can determine whether the approximations are converging or not.

Now we will compare the three different approximations in equations 4.1, 4.2 and 4.4 through some examples. The first example that we consider, which we call Model 1, is a one parameter family of eigenvectors of a Hamiltonian matrix of the form $H(c) = H_0 + cH_1$, where H_0 and H_1 are Hermitian matrices. We choose H_0 to be a diagonal matrix, while H_1 has both diagonal and off-diagonal elements. We consider three different situations with different H_0 and H_1, and we call them Models 1A, 1B, and 1C. In all three cases, the dimension of H matrix is 800.

In Model 1A, we take H_0 to be the matrix with elements $H_0(n, n) = n$ for $n = 1, \cdots, 800$. We choose H_1 to be $H_1(n, n) = n$ for $n = 1, \cdots, 800$ and

$$H_1(n + 2, n) = H_1(n, n + 2) = 1, \tag{4.5}$$

for $n = 1, \cdots, 798$. As we mentioned before, avoided level crossing in the complex plane is the cause of singularities, which limit the radius of convergence of perturbation theory. Since we want to look at a target point which is within the radius of convergence of perturbation theory, it is important to know where the singularity for this Hamiltonian is in the complex plane. For this model, the nearest branch point to the origin is located at $c = -0.559 \pm 0.497i$.

In Model 1B, we take H_0 to be the matrix with elements $H_0(n, n) = 2n$ for $n =$

$1, \cdots, 800$. We choose H_1 to be $H_1(n,n) = n$ for $n = 1, \cdots, 800$ and

$$H_1(n+2,n) = H_1(n,n+2) = 1, \tag{4.6}$$

for $n = 1, \cdots, 798$. This time the nearest branch point to the origin is located at $c = -1.422 \pm 0.503i$.

In Model 1C, we take H_0 to be the matrix with elements $H_0(n,n) = 100n$ for $n = 1, \cdots, 800$. We choose H_1 to be $H_1(1,1) = -100$, $H_1(2,2) = -200$, and take $H_1(n,n) = -75n$ for $n = 3, \cdots, 800$. For the off-diagonal entries we set

$$H_1(n+1,n) = H_1(n,n+1) = 1, \tag{4.7}$$

for $n = 1, \cdots, 799$. In this model, the nearest branch point to the origin is located at $c = 0.907 \pm 0.255i$.

With these three different sets of matrices defining three different problems, we try to approximate the eigenvector $|v(c_t)\rangle$ with our three approximation techniques in each of the problems. For all three model calculations, the target value of the parameter is $c_t = 0.2$, which is within the radius of convergence of perturbation theory for all three cases. In Fig. 4.1 we plot the logarithm of the error for all three cases of Model 1.

From figure 4.1 we can make two important observations. The first observation is that eigenvector and vector continuation converge faster than perturbation theory. The second point is that eigenvector and vector continuation have nearly identical errors at each order. We should also mention that there is nothing special about these matrix models, and we find similar results to these in all other matrix examples where perturbation theory is convergent.

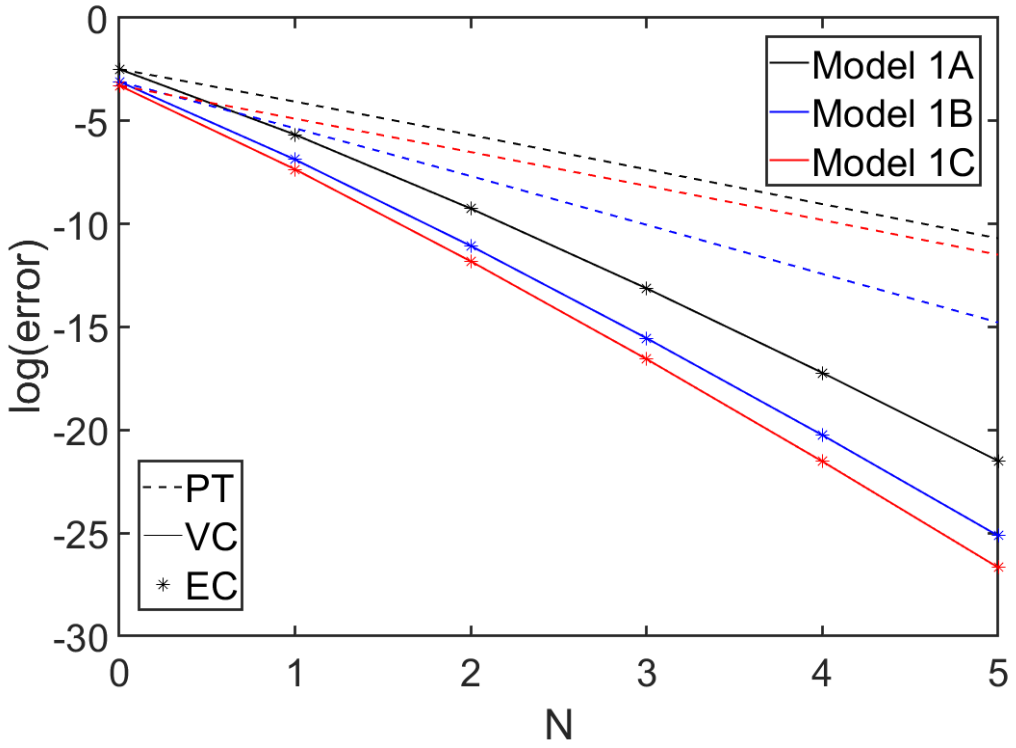


Figure 4.1: Logarithm of the error versus order N for eigenvector continuation (asterisks), vector continuation (solid lines), and perturbation theory (dashed lines). The three different colors (black, blue and red) correspond with Models 1A, 1B, and 1C respectively.

So this shows that eigenvector continuation performs better than perturbation theory even where perturbation theory works, and we have much less error at same order of calculation. The second point is even more interesting - even though we calculated eigenvector continuation and vector continuation approximations very differently (see equations 4.1 and 4.2), they have the same convergence properties. This tell us more about how eigenvector continuation works as a method and how well it converges. Let us try to understand why we get similar convergence results with eigenvector continuation and vector continuation.

4.2 Asymptotic Convergence of Eigenvector Continuation and Vector Continuation

Let us now try to show that eigenvector continuation and vector continuation have identical convergence properties. Consider vector continuation at order N. Let $V^N(0)$ be the subspace spanned by $|w^{(0)}(0)\rangle, \cdots |w^{(N)}(0)\rangle$, and let $V_\perp^N(0)$ be the orthogonal complement to $V^N(0)$. Now equation 4.1 tells us that there is no error at all in the coefficients of $|w^{(0)}(0)\rangle, \cdots |w^{(N)}(0)\rangle$, and the residual vector for $|v(c_t)\rangle_N^{\mathrm{VC}}$ lies entirely in $V_\perp^N(0)$.

Next we consider eigenvector continuation at order N. In this case we project $H(c_t)$ onto $V^N(0)$ and find the resulting ground state by minimizing the eigenvalue. In this way we have effectively turned off all matrix elements of $H(c_t)$ that involve vectors in $V_\perp^N(0)$. If we now turn on these matrix elements as a perturbation, then we get a first-order correction to the wave function from transition matrix elements connecting $V^N(0)$ with $V_\perp^N(0)$. However, this correction to the wave function that lies in $V_\perp^N(0)$, and our wave function in $V^N(0)$ is not affected. The corrections to the coefficients of $|w^{(0)}(0)\rangle, \cdots |w^{(N)}(0)\rangle$ will appear at second order in perturbation theory, since this involves transitions from $V^N(0)$ to $V_\perp^N(0)$ and then returning back from $V_\perp^N(0)$ to $V^N(0)$. If the norm of the residual vector for eigenvector continuation is $O(\epsilon)$, then eigenvector continuation and vector continuation will differ at $O(\epsilon^2)$. This proves that eigenvector continuation and vector continuation have identical convergence properties in the limit of large N.

Now let us study the asymptotic convergence properties by looking at the last term in our approximations in equations 4.1, 4.2 and 4.4. The idea here is that we can understand how quickly these series converge by looking at last term in the series, and observing how quickly that is converging in the limit of large N. The norm of the last term of all the

corresponding series approximations are,

$$L_N^{\mathrm{VC}}(c_t) = \left| \langle w^{(N)}(0) | v(c_t) \rangle \right|, \tag{4.8}$$

$$L_N^{\mathrm{EC}}(c_t) = \left| a(c_t, N, N) \right|, \tag{4.9}$$

$$L_N^{\mathrm{PT}}(c_t) = \left\| |v^{(N)}(0)\rangle \frac{c_t^N}{N!} \right\|. \tag{4.10}$$

Again in equation 4.8, we need to knowledge of the exact eigenvector $|v(c_t)\rangle$. If we don't have that information, and we are within the radius of convergence of perturbation theory, then we can expand $|v(c_t)\rangle$ perturbatively, similar to equation 4.3. Then we have,

$$L_N^{\mathrm{VC}}(c_t) = \left| \sum_{n=0}^{\infty} \langle w^{(N)}(0) | v^{(n)}(0) \rangle \frac{c_t^n}{n!} \right|. \tag{4.11}$$

However, by definition $|w^{(N)}(0)\rangle$ is orthogonal to $\{|v^{(0)}(0)\rangle, \cdots, |v^{(N-1)}(0)\rangle\}$ and thus we can write,

$$L_N^{\mathrm{VC}}(c_t) = \left| \sum_{n=N}^{\infty} \langle w^{(N)}(0) | v^{(n)}(0) \rangle \frac{c_t^n}{n!} \right|. \tag{4.12}$$

So even though we do not know $|v(c_t)\rangle$, we can find $L_N^{\mathrm{VC}}(c_t)$ perturbatively, order by order. In this series expression, we will call the first term in the series for $n = N$ as the leading order (LO) approximation. We will call the partial series up to $n = N+1$ the next-to-leading order (NLO) approximation, and so on. The $\mathrm{N}^k\mathrm{LO}$ approximation is similarly defined by,

$$L_N^{\mathrm{VC,N}^k\mathrm{LO}}(c_t) = \left| \sum_{n=N}^{N+k} \langle w^{(N)}(0) | v^{(n)}(0) \rangle \frac{c_t^n}{n!} \right|, \tag{4.13}$$

and at leading order we have

$$L_N^{\mathrm{VC,LO}}(c_t) = \left| \langle w^{(N)}(0) | v^{(N)}(0) \rangle \, \frac{c_t^N}{N!} \right| . \tag{4.14}$$

Now if we look at equations 4.14 and 4.10, then we notice that they are similar, except for $\left| \langle w^{(N)}(0) | v^{(N)}(0) \rangle \right|$ instead of $|v^{(N)}(0)\rangle$. However, notice that $\left| \langle w^{(N)}(0) | v^{(N)}(0) \rangle \right|$ is smaller than the norm of $|v^{(N)}(0)\rangle$ because, in general, $|v^{(N)}(0)\rangle$ is not orthogonal to the other derivative vectors
$\{|v^{(0)}(0)\rangle, \cdots, |v^{(N-1)}(0)\rangle\}$. This explains why vector continuation is converging more rapidly than perturbation theory. Perturbation theory must deal with constructive and destructive interference between non-orthogonal vectors at different orders of perturbation theory, a phenomenon that we call differential folding. Depending on the problem, differential folding can be a very large effect, and it is the reason why perturbation theory converges more slowly than vector and eigenvector continuation.

Another way to put it is that we know that a Taylor expansion like

$$f(x) = \sum_{n=0}^{\infty} f^{(n)}(0) \frac{x^n}{n!} \tag{4.15}$$

converges as we take the limit to infinity. However, for an approximation with a fixed number of terms in the series (like in equation 4.4),

$$\tilde{f}(x) = \sum_{n=0}^{N} f^{(n)}(0) \frac{x^n}{n!} \tag{4.16}$$

the choice of Taylor series coefficients $x^n/n!$ for the derivatives is not optimal. Our intuition of Fourier analysis tells us that we get a much better approximation when we make a sum of orthogonal terms, and that is indeed the case. Vector continuation uses this orthogonality,

and thus is better than perturbation theory.

Now let us use the last terms in the three series to study the asymptotic convergence. We look at the convergence ratio obtained by taking the ratio of the last term in the series at two widely separated orders N' and N, with $N > N'$. We define the convergence ratios as follows,

$$\mu^{\mathrm{VC}}(c_t) = \left| L_N^{\mathrm{VC}}(c_t) / L_{N'}^{\mathrm{VC}}(c_t) \right|^{1/(N-N')}, \tag{4.17}$$

$$\mu^{\mathrm{EC}}(c_t) = \left| L_N^{\mathrm{EC}}(c_t) / L_{N'}^{\mathrm{EC}}(c_t) \right|^{1/(N-N')}, \tag{4.18}$$

$$\mu^{\mathrm{PT}}(c_t) = \left| L_N^{\mathrm{PT}}(c_t) / L_{N'}^{\mathrm{PT}}(c_t) \right|^{1/(N-N')}. \tag{4.19}$$

These definitions are motivated from the ratio test of convergence for a series. Intuitively, μ is ratio at which consecutive terms in the series converge (or diverge) asymptotically. However, note that our definitions with finite orders will have a small dependence on N' and N. However, for notational convenience, we omit writing the explicit dependence on N and N'. Also, when it is clear from the context, we omit writing the dependence on c_t as well. We also note that these convergence ratio functions will not be very smooth. Occasionally when the numerator vanishes, we will have cusps and it will blow up whenever the denominator vanishes. Fortunately these special points occur at only a few isolated values of c_t, and we eliminate cusps or divergences at any particular value of c_t by changing N or N'. Overall the functions in Eq. (4.17), (4.18), and (4.19) provide a useful measure of the convergence properties of the three methods.

Now let us look at how these three convergence ratios compare with an example. We look at a new practical problem, and we call this example Model 2. In this model, we consider a system of spin-half fermions with attractive zero-range interactions in three dimensions.

As we vary the interaction strength between the fermions, we observe some very interesting physics. At weak coupling this many-body system acts like a Bardeen-Cooper-Schrieffer (BCS) superfluid, but at strong coupling it behaves like a Bose-Einstein condensate (BEC) [27, 28]. In between the BCS and BEC regimes, there is a smooth crossover region that contains a scale-invariant point called the unitary limit. At this point the scattering length diverges, and the two-body system has a zero energy resonance. There have been various experimental studies of BCS-BEC crossover and the unitary limit using trapped ultra-cold Fermi gases of alkali atoms [29–32].

Here we use eigenvector continuation to study the crossover transition for two spin-up and two spin-down fermions in an $L = 4$ periodic cubic lattice as detailed in Ref. [33]. The corresponding Hamiltonian has $4^9 = 262144$ dimensions, and we use projection operators to remove all unphysical states without the proper antisymmetrization. With 4 particles, $L = 4$, and 3 dimensions, one would expect 4^{12} possible configurations, but we measure all the particles with respect to one particle, i.e., we have 3 particles that can move around with respect to the 4^{th} particle. This reduces the configuration space to $4^9 = 262144$ dimensional. Without eigenvector continuation, we will have to diagonalize a 262144-dimensional Hamiltonian matrix at the target coupling, which takes a long time, and even more time as our coupling becomes larger. eigenvector continuation gives us a huge speed boost here.

Let $\boldsymbol{n}$ denote the spatial lattice points on a three dimensional L^3 periodic lattice. We use lattice units where all the physical quantities are multiplied by powers of the spatial lattice spacing to make the combination dimensionless. Let the lattice annihilation operators for the spin-up and spin-down fermions be $a_\uparrow(\boldsymbol{n})$ and $a_\downarrow(\boldsymbol{n})$ respectively. Our free non-relativistic lattice Hamiltonian is given by,

$$H_{\text{free}} = \frac{3}{m} \sum_{i=\uparrow,\downarrow} \sum_{\boldsymbol{n}} a_i^\dagger(\boldsymbol{n}) a_i(\boldsymbol{n})$$
$$-\frac{1}{2m} \sum_{l=1,2,3} \sum_{i=\uparrow,\downarrow} \sum_{\boldsymbol{n}} a_i^\dagger(\boldsymbol{n}) \Big[a_i(\boldsymbol{n}+\hat{l}) + a_i(\boldsymbol{n}-\hat{l}) \Big]. \tag{4.20}$$

We add to the free Hamiltonian an on-site contact interaction,

$$H_{\text{free}} + C \sum_{\boldsymbol{n}} \rho_\uparrow(\boldsymbol{n}) \rho_\downarrow(\boldsymbol{n}), \tag{4.21}$$

where

$$\rho_\uparrow(\boldsymbol{n}) = a_\uparrow^\dagger(\boldsymbol{n}) a_\uparrow(\boldsymbol{n}), \tag{4.22}$$

$$\rho_\downarrow(\boldsymbol{n}) = a_\downarrow^\dagger(\boldsymbol{n}) a_\downarrow(\boldsymbol{n}). \tag{4.23}$$

So the different particles feel an attraction (or repulsion depending on sign of C) only when they are on the same site. If the attraction is strong enough, then it becomes a condensate.

Our control parameter c corresponds to the product of the particle mass m and interaction coupling C. Negative coupling indicates attractive interaction, whereas positive coupling would mean repulsive interaction. In terms of our control parameter c, the value $c = -3.957$ corresponds with the unitary limit. The larger negative values of c correspond to the strong-coupling BEC phase, and smaller negative values of c correspond to the weak-coupling BCS phase. The point $c = 0$ corresponds to a non-interacting system with special symmetries and degeneracies. We choose the training vectors at a more general point on the weak-coupling BCS side at $c = -0.4695$.

Now if we compare the convergence of the three series in this model, then we see similar results as shown in figure 4.1. So let us now look at how the convergence ratios for the three

methods compare for this model. In Fig. 4.2 we show the convergence ratios μ^{VC}, μ^{EC}, and μ^{PT} (as defined in equations 4.17, 4.18 and 4.19) versus c_t for $N = 10$ and $N' = 0$. Note that c_t is negative, meaning we are looking in the attractive coupling regime. Smaller μ corresponds to faster convergence since by definition convergence ratio μ is how small the next term in the series is becoming compared to the previous term. We see from the graph that μ^{VC} and μ^{EC} remain well below μ^{PT}, indicating the faster convergence of vector and eigenvector continuation compared to perturbation theory.

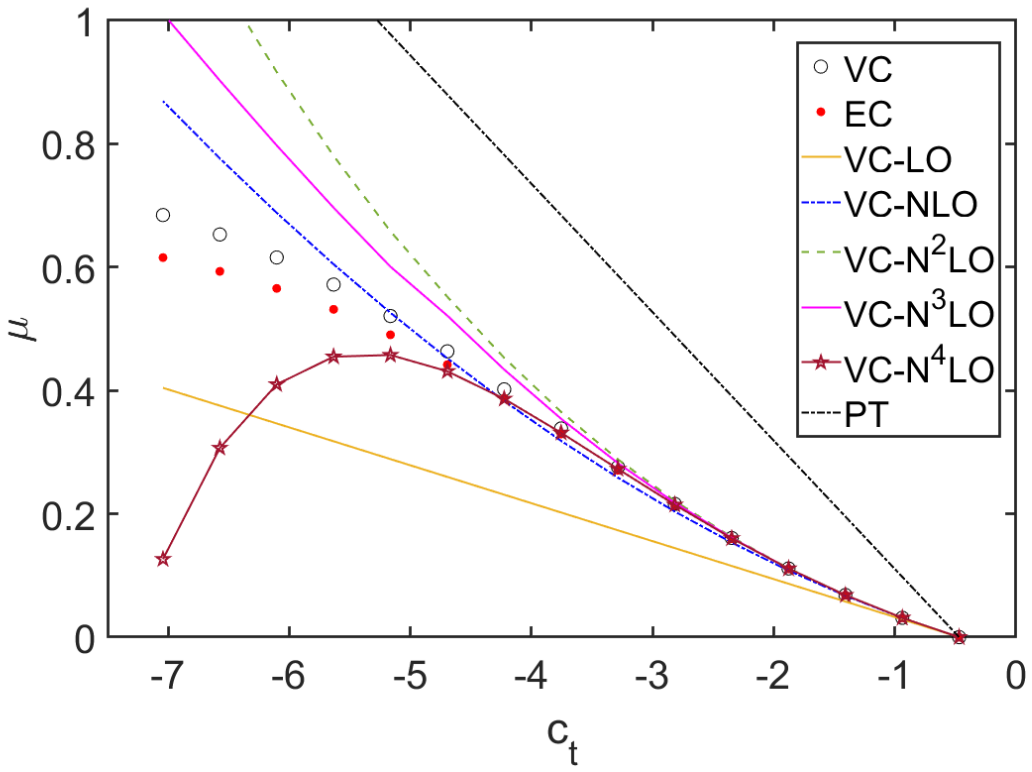


Figure 4.2: Comparison of the convergence ratios $\mu^{\mathrm{VC}}(c_t)$, $\mu^{\mathrm{EC}}(c_t)$, and $\mu^{\mathrm{PT}}(c_t)$ for Model 2 with $N = 10$ and $N' = 0$. The training vectors for all cases are evaluated on the weak-coupling BCS side at $c = -0.4695$. The unitary limit value corresponds to $c = -3.957$.

As we cross into the strong-coupling BEC side, perturbation theory diverges, as indicated by the convergence ratio μ^{PT} exceeding 1. However vector and eigenvector continuation both converge even at very strong coupling far on BEC side, as indicated by μ^{VC} and μ^{EC}

both remaining well below 1. Furthermore, the vector and eigenvector continuation results are in close agreement with each other, with only a slight difference when the convergence is slower. We also plot the LO, NLO, N^{2}LO, N^{3}LO, and N^{4}LO approximations to μ^{VC} as defined in Eq. (4.13). We can observe that the expansion of μ^{VC} converges for c_t within the radius of convergence of perturbation theory. Our results also show that the entire BEC-BCS crossover region can be estimated very well by variational methods like eigenvector continuation.

We also note that figure 4.2 shows the convergence of the three series as a function of the parameter c. Smaller μ corresponds to faster convergence, and we see that the convergence near the origin is faster than the convergence for far away points on the c axis.

To summarize our results so far, we have seen that within the radius of convergence of perturbation theory, eigenvector continuation and vector continuation have similar convergence properties, and they converge faster than the standard perturbation theory. The main reason behind this is because of the orthonormal expansion that vector continuation employs, and somehow eigenvector continuation also learns the orthogonal space correctly and chooses linear combinations that makes best use of the orthonormal expansion. On the other hand, perturbative expansion is sub-optimal because the different non-orthogonal components in different orders of perturbation theory have constructive and destructive interference. Now let us look at how the convergence looks like beyond the radius of convergence of perturbation theory. We have already seen it a little bit in the strong coupling BEC regime in figure 4.2, but let us now look in detail for a general case.

4.3 Convergence of Eigenvector Continuation outside Radius of Convergence of Perturbation Theory

Outside the radius of convergence of perturbation theory, we can estimate the convergence ratio using extrapolation methods. If there are no branch points nearby, then the convergence ratio function can be extrapolated using standard methods such as Padé approximants [5] or conformal mapping [34, 35].

To illustrate this, we consider another example, which we call Model 3. In this model, we again consider a Hamiltonian of the form $H = H_0 + cH_1$, and we take H_0 to be a 500×500 diagonal matrix with entries $H_0(n, n) = 100n$ for $n = 1, \cdots, 500$. H_1 is a 500×500 matrix with nonzero entries as follows:

$$H_1(1, 1) = 40, \tag{4.24}$$

$$H_1(2, 2) = -80, \tag{4.25}$$

$$H_1(3, 3) = -180, \tag{4.26}$$

$$H_1(4, 4) = -260, \tag{4.27}$$

$$H_1(5, 5) = -320, \tag{4.28}$$

$$H_1(6, 6) = -335, \tag{4.29}$$

for $n = 1, \cdots, 499$:

$$H_1(n + 1, n) = H_1(n, n + 1) = 2, \tag{4.30}$$

for $n = 1, \cdots, 498$:

$$H_1(n + 2, n) = H_1(n, n + 2) = 5, \tag{4.31}$$

for $n = 1, \cdots, 497$:

$$H_1(n+3, n) = H_1(n, n+3) = 5, \tag{4.32}$$

for $n = 7, \cdots, 500$:

$$H_1(n, n) = 50n. \tag{4.33}$$

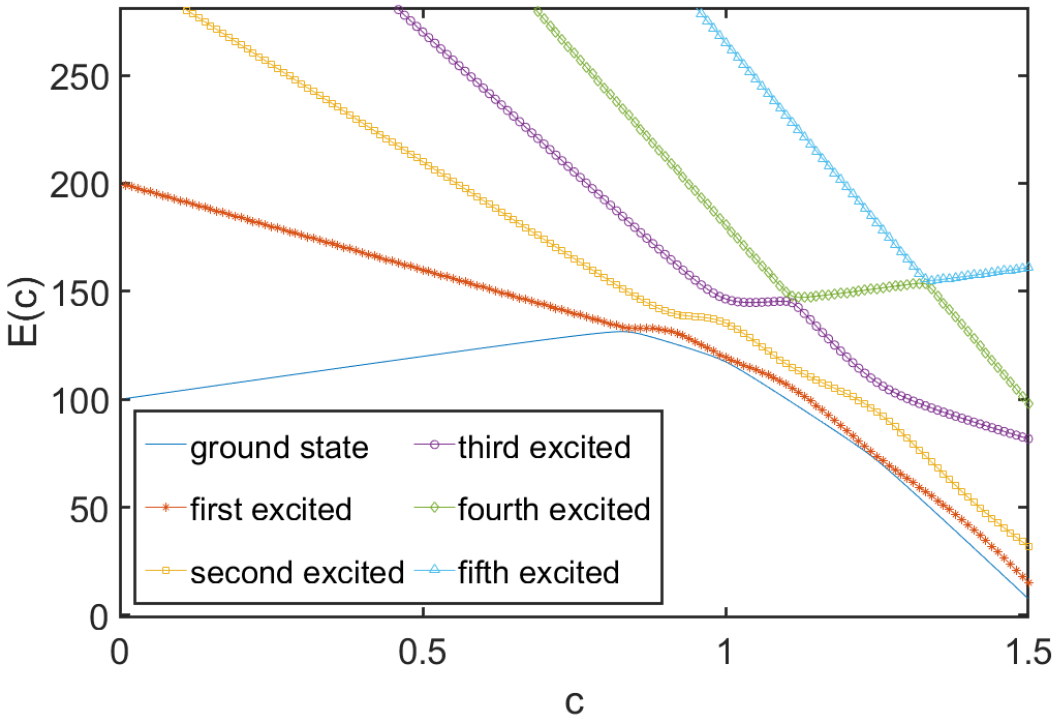


Figure 4.3: The lowest six energies of Model 3 as a function of c.

Model 3 was chosen in such a way that perturbation theory will break down due to several sharp avoided level crossings in the complex plane. In Fig. 4.3 we show the energies of the lowest six energies as a function of the control parameter c. The closest branch point to $c = 0$ occurs very close to the real axis near $c = 0.84$. The first avoided level crossing near $c = 0.84$ can be seen in the figure.

For the three different methods, we can look in model 3 at how the convergence of the three series and the convergence ratios compare. We again find similar results as figure 4.1

when we compare the convergence of the three different series. And a comparison of the convergence ratios μ^{VC}, μ^{EC}, and μ^{PT} is shown in figure 4.4, as a function of c_t for $N = 20$ and $N' = 0$. In the limit $|N - N'| \to \infty$, the convergence ratio μ^{PT} will cross the value 1 for c_t near 0.84, indicating the divergence of perturbation theory. In our calculation, we don't see this because of finite $|N - N'|$, but we have verified that as we increase $|N - N'|$ further, the convergence ratio μ^{PT} goes closer to 1 near $c_t = 0.84$. On the other hand, we observe that μ^{VC} and μ^{EC} are in close agreement with each other and remain below 1 near $c_t = 0.84$, indicating that they still converge even though perturbation theory does not.

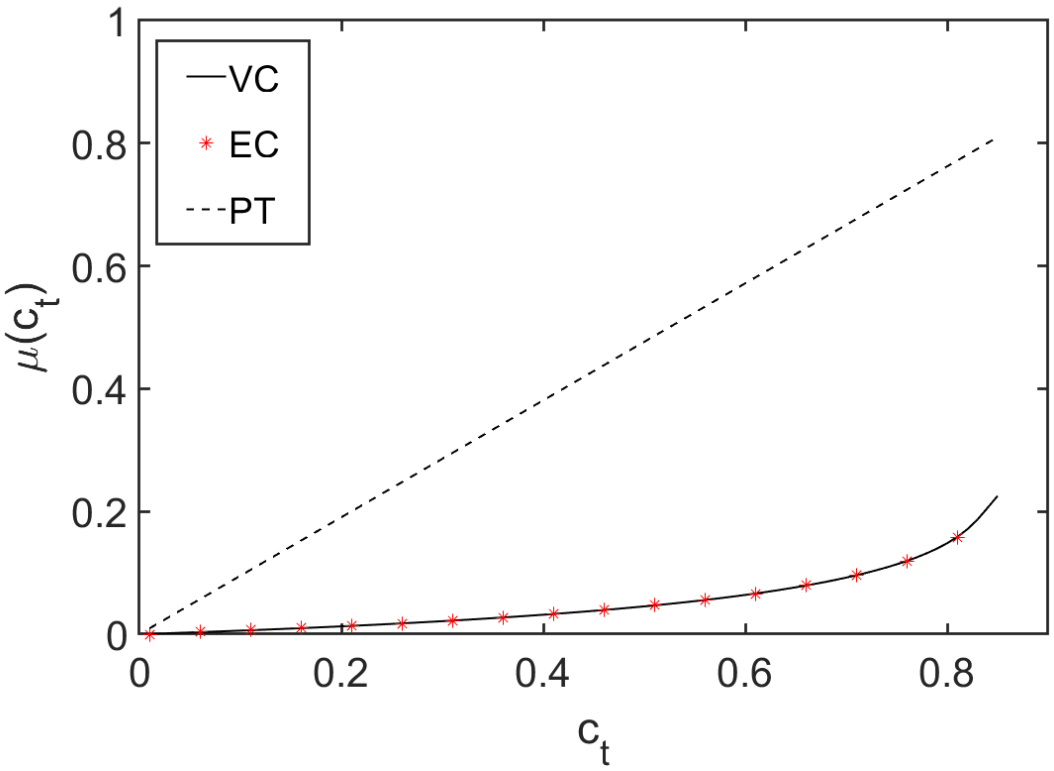


Figure 4.4: Comparison of the convergence ratios μ^{VC}, μ^{EC}, and μ^{PT} versus c_t for Model 3 with $N = 20$ and $N' = 0$.

In Fig. 4.5 we plot μ^{VC} and the LO, NLO, N^{2}LO, and N^{3}LO approximations to μ^{VC} for $N = 20$ and $N' = 0$. Once again we see that the series expansion in equation 4.12 converges

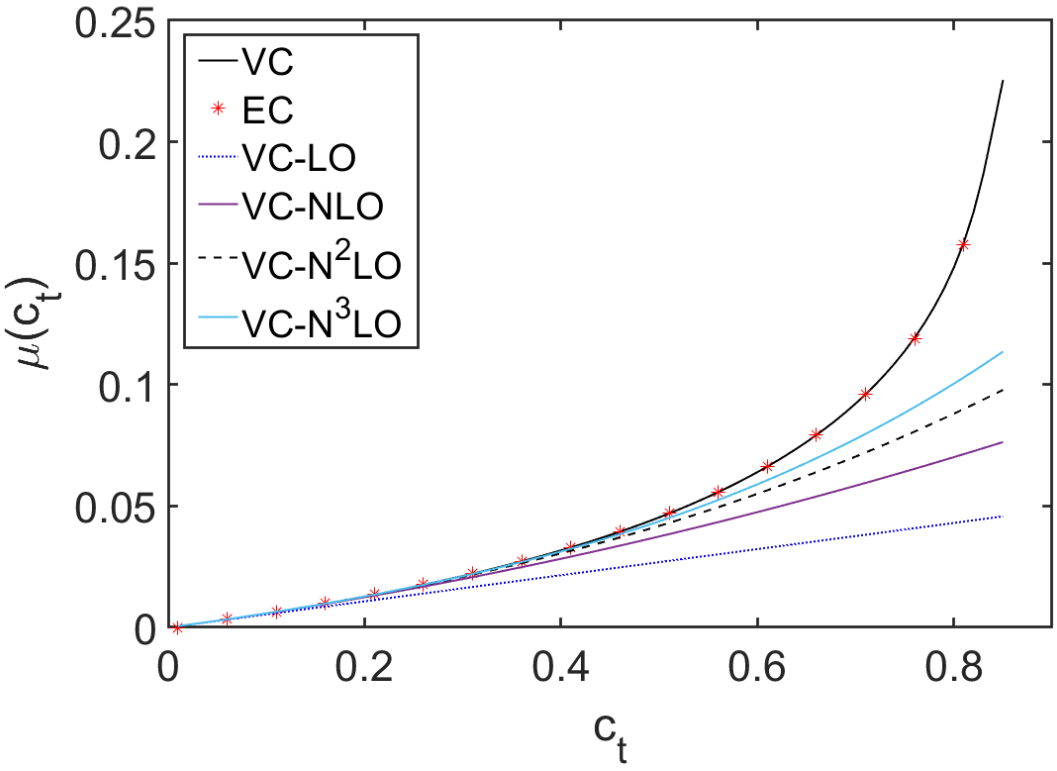


Figure 4.5: Plots of the convergence ratios $\mu^{\rm VC}$, $\mu^{\rm EC}$, and the LO, NLO, N^{2}LO, and N^{3}LO approximations to $\mu^{\rm VC}$ versus c_t for Model 3 with $N = 20$ and $N' = 0$.

within the radius of convergence of perturbation theory, which for this example corresponds to $c = 0.84$.

Now in this model 3, we can easily perform eigenvector and vector continuation on the negative c axis because there are no branch points there. We show this extrapolation in figure 4.6, where we plot $\mu^{\rm VC}$ and $\mu^{\rm EC}$ for $N = 20$ and $N' = 0$, and we are going to parameter values beyond the range of perturbation theory (which is $|c| < 0.84$). We also show the (1,1) and (2,2) Padé approximations to $\mu^{\rm VC}$. We see that the Padé approximations describe the shape of $\mu^{\rm VC}$ quite well since there are no nearby branch points.

If we want to extrapolate along the positive c axis, then we have to deal with the branch point near $c = 0.84$. Near these singular points, the slope of the convergence ratio function

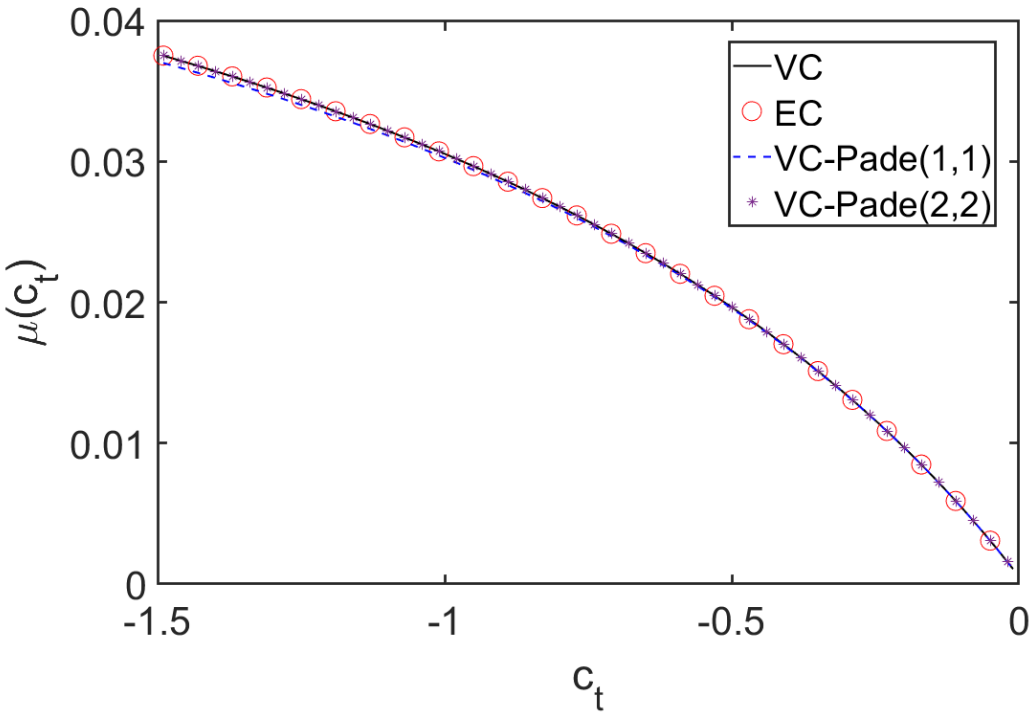


Figure 4.6: Plots of the convergence ratios μ^{VC}, μ^{EC}, and the Padé approximations (1,1) and (2,2) to μ^{VC} versus c_t for Model 3 with $N = 20$ and $N' = 0$.

will be more than that predicted by Padé approximants or conformal mapping because the Riemann surface of the ground state eigenvector is entwined with the Riemann surface of the first excited state eigenvector. If the branch point is very close to the real axis, then we have an avoided level crossing or Landau-Zener transition where the wave functions of the ground state and first excited state interchange as we pass by the branch point. What this means is as we pass by the branch point, the ground state becomes the excited state, and the excited state becomes the ground state. If keep looking only at the ground state, then its properties will change quickly as we cross the branch point. However, if we add in the information from the first excited state, then we have a way to study the ground state convergence beyond the level-crossing - just look at the convergence of the first excited state and extrapolate it across the level-crossing. As we pass the avoided level-crossing, the excited state has become

the ground state, and its convergence properties are same as that of the ground state.

We therefore have a way to predict the convergence ratio of the ground state beyond the branch point by looking at the convergence ratio of the excited state. We define the convergence ratios for the first excited state μ_1^{VC} and μ_1^{EC} in the same manner as μ^{VC} and μ^{EC}, except that we replace the target ground state $|v(c_t)\rangle$ with the first excited state $|v_1(c_t)\rangle$. However, note that we are still using the same orthonormal basis states $|w^{(n)}(0)\rangle$ associated with the ground state at $c = 0$. To calculate the eigenvector continuation approximation of the first excited state, $|v_1(c_t)\rangle_N^{\mathrm{EC}}$, we want to use a subspace that includes derivatives of the ground state $|v^{(0)}(0)\rangle \cdots |v^{(N)}(0)\rangle$ and also derivatives of the first excited state $|v_1^{(0)}(0)\rangle \cdots |v_1^{(N)}(0)\rangle$. Therefore at each training point location, we take the ground state and the excited state eigenvectors as the training eigenvectors. If we have n training point locations, then we have $2n$ number of training points for eigenvector continuation. This increases the numerical complexity and the error in the estimate, but eigenvector continuation can be done for excited states in this way.

In Fig. 4.7 we show μ^{VC}, μ_1^{VC}, μ^{EC}, μ_1^{EC}, and the N^{3}LO approximations to μ^{VC} and μ_1^{VC} for $N = 20$ and $N' = 0$. Notice the approximate vertical and horizontal reflection symmetries near the branch point. For $c_t < 0.84$ the increase in the ground-state convergence ratio mirrors the decrease in the excited-state convergence ratio. Also the increase in the ground-state convergence ratio for $c_t > 0.84$ is reflected in the decrease in the excited-state convergence ratio for $c_t < 0.84$.

To summarize, we can learn a great deal of information about the convergence of vector and eigenvector continuation just from series expansions around $c = 0$. Near the branch point, we know that μ^{VC} (and μ^{EC}) for ground state and excited state, cross near the midpoint between N^{k}LO approximations to μ^{VC} and μ_1^{VC} for any k. And the N^{k}LO ap-

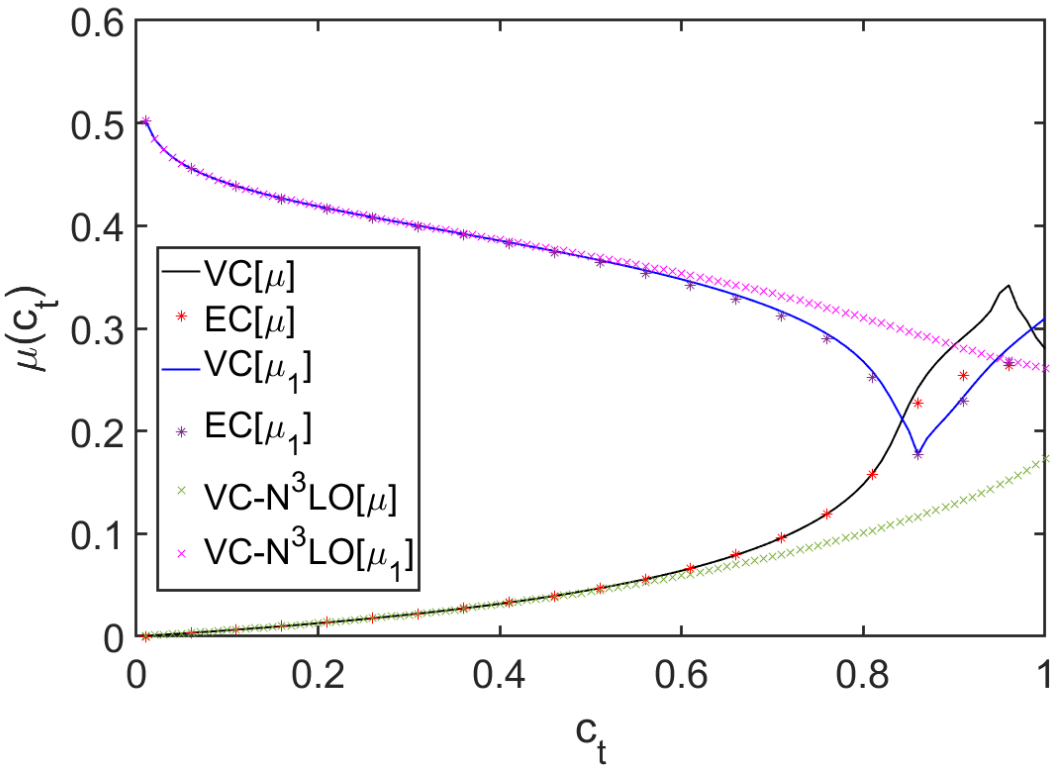


Figure 4.7: Plots of the convergence ratios $\mu^{\rm VC}$, $\mu_1^{\rm VC}$, $\mu^{\rm EC}$, $\mu_1^{\rm EC}$, and the N^{3}LO approximations to $\mu^{\rm VC}$ and $\mu_1^{\rm VC}$ versus c_t for Model 3 with $N = 20$ and $N' = 0$.

proximations to $\mu^{\rm VC}$ and $\mu_1^{\rm VC}$ are calculated entirely from perturbation theory at $c = 0$. Also, the location of the nearby branch point can be deduced from the convergence radius of the series expansions. So all the information about error convergence, even beyond branch point, is present at the origin. While there are limits to how far we can go in c_t with these convergence ratio predictions, it is clear that we can predict the convergence ratios inside and to some extent outside the radius of convergence, from the derivatives of the eigenvectors near $c = 0$. It is quite fascinating how far we can extrapolate beyond the radius of convergence of perturbation theory, given that we are using the same derivatives that perturbation theory utilizes. And the fact that these derivatives of the eigenvectors near $c = 0$ are being used to predict convergence of vector and eigenvector continuation, which are completely

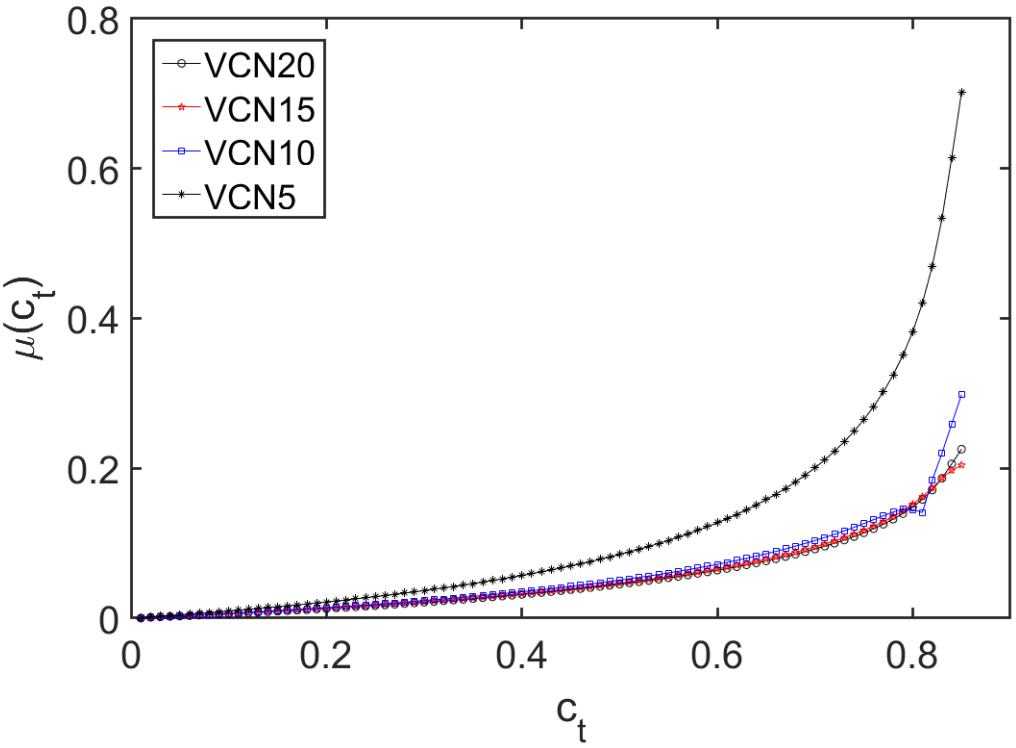


Figure 4.8: Plots of the convergence ratio μ^{VC} versus c_t for Model 3 with $N' = 0$ and $N = 5, 10, 15, 20$.

non-perturbative calculations.

Before we move on to the next topic, let us quickly mention the dependence of our convergence ratio results on N' and N. So far all our calculations were done with $N = 20$ and $N' = 0$. The convergence ratio μ^{VC} provides information about the rate of convergence of vector continuation in the limit of many training vectors, so ideally we want a large $N - N'$. In Fig. 4.8 we plot μ^{VC} for Model 3 for $N' = 0$ and $N = 5, 10, 15, 20$. We see that the convergence ratio is approaching a common ratio in the limit of large N.

4.4 Multi-parameter Eigenvector Continuation

Now let us try to extend our discussion of eigenvector continuation to the case where we have $D > 1$ parameters in the Hamiltonian. If we again want to take the derivatives of the eigenvector at some point, the multi-parameter case is exactly equivalent to the one parameter case if change our one-dimensional derivatives to directional derivatives $(\boldsymbol{c}_t - \boldsymbol{c})\cdot\boldsymbol{\nabla}$ that act upon $|v(\boldsymbol{c})\rangle$. Here $\boldsymbol{c}$ is the limit point of the training data, which we can redefine to be at the origin, and $\boldsymbol{c}_t$ is the target point where we are interested in finding the eigenvector at. In other words, we can just redefine our parameters so that we can work with only one effective parameter.

But if we now consider the more difficult problem of convergence at all target points at some fixed distance from $\boldsymbol{c}$, then we cannot reduce the problem to one dimension. For this problem, if we want eigenvector continuation estimate of the target eigenvector, then we need to include all $(k+D-1)!/[k!(D-1)!]$ partial derivatives at order k. But for the training eigenvectors, we need to include all the derivatives of order less than k as well, and therefore we can conclude that a D parameter calculation with N_D training vectors is equivalent to a one parameter calculation with $N_1 + 1$ training vectors with

$$N_D = \frac{(N_1 + D)!}{N_1! D!}. \tag{4.34}$$

We will now test this idea in multi-parameter eigenvector continuation through a numerical example. Let H_0 be a diagonal matrix with elements $H_0(n, n) = n^{0.1}$ with $n = 1, \cdots, 500$. Let H_1, H_2, and H_3 be three different random 500×500 matrices with each matrix element sampled from a normal distribution with zero mean and standard deviation 10^{-3}. We will call this random matrices example as Model 4.

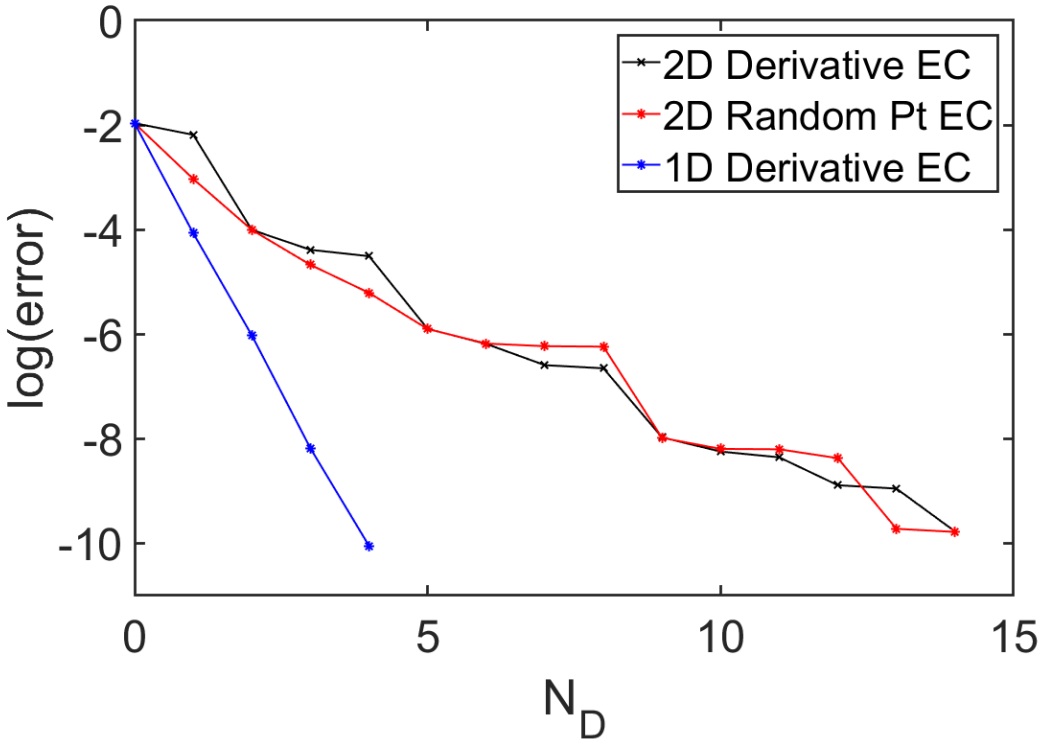


Figure 4.9: Comparison of the logarithm of the eigenvector continuation error versus N_D for $D = 1$ and $D = 2$ dimensions.

First we consider the convergence of eigenvector continuation for the two parameter Hamiltonian given by,

$$H(c_1, c_2) = H_0 + c_1 H_1 + c_2 H_2. \tag{4.35}$$

We take the training points to be in the neighborhood of the origin $\boldsymbol{c} = \boldsymbol{0}$. For the target point we pick a random point $\boldsymbol{c}_t = (-1.636, 1.150)$. Our convergence results are shown in figure 4.9. We first consider the one parameter case labelled as 1D Derivative EC. This corresponds to taking directional derivatives $[\boldsymbol{c}_t \cdot \boldsymbol{\nabla}]^k$ acting on $|v(\boldsymbol{c})\rangle$ at $\boldsymbol{c} = \boldsymbol{0}$ for $k = 0, \cdots, N_1$, for a total $N_1 + 1$ training vectors. We plot the logarithm of the error versus N_1.

We now compare this with the eigenvector continuation results using $N_2 + 1$ training

vectors for the two dimensional case. These training vectors correspond to the $N_2 + 1$ lowest-order partial derivatives $\nabla_1^{k1}\nabla_2^{k2}$ acting on $|v(\boldsymbol{c})\rangle$ at $\boldsymbol{c} = \mathbf{0}$. This convergence data is labelled as 2D Derivative EC in figure 4.9. From the error equivalence formula $N_2 = (N_1 + 2)!/(N_1!2!) - 1$, we have

$$\begin{aligned} N_1 = 0 &\rightarrow N_2 = 0, \\ N_1 = 1 &\rightarrow N_2 = 2, \\ N_1 = 2 &\rightarrow N_2 = 5, \\ N_1 = 3 &\rightarrow N_2 = 9, \\ N_1 = 4 &\rightarrow N_2 = 14, \\ N_1 = 5 &\rightarrow N_2 = 20. \end{aligned} \tag{4.36}$$

As we see in Fig. 4.9, these predictions work quite well. The errors for N_1 are approximately equal to the errors for the corresponding value of N_2. We can compare these results with the results we obtain when, instead of using partial derivatives, we simply take training vectors at $N_2 + 1$ random points in the neighborhood of the origin. This data is labelled as 2D Random EC. As we can see, this data agrees well with the 2D Derivative EC data. This is exactly what we expected, since the partial derivatives are well approximated by finite differences of $|v(\boldsymbol{c})\rangle$ at points in the neighborhood of the origin.

Now in Model 4, we now consider the convergence of eigenvector continuation for the three parameter Hamiltonian

$$H(c_1, c_2, c_3) = H_0 + c_1 H_1 + c_2 H_2 + c_3 H_3. \tag{4.37}$$

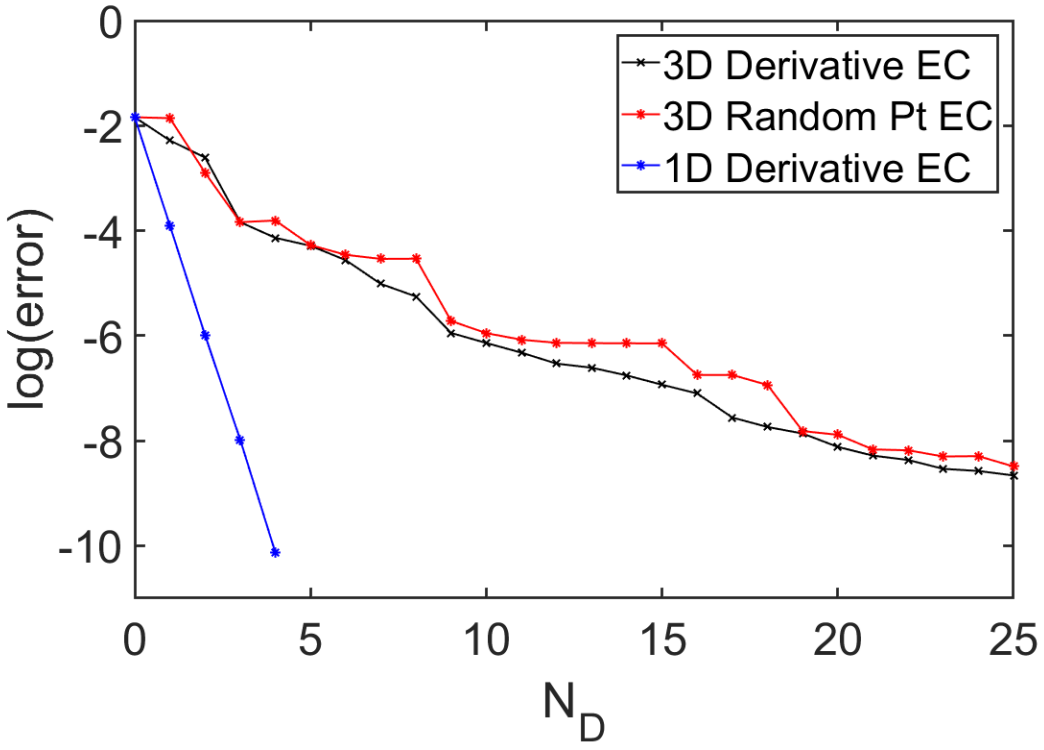


Figure 4.10: Comparison of the logarithm of the eigenvector continuation error versus N_D for $D = 1$ and $D = 3$ dimensions.

with random target point $\boldsymbol{c}_t = (-1.034, -1.065, 1.341)$. In this case we have the error equivalence formula $N_3 = (N_1 + 3)!/(N_1!3!) - 1$ and therefore

$$\begin{aligned} N_1 = 0 &\rightarrow N_3 = 0, \\ N_1 = 1 &\rightarrow N_3 = 3, \\ N_1 = 2 &\rightarrow N_3 = 9, \\ N_1 = 3 &\rightarrow N_3 = 19. \end{aligned} \tag{4.38}$$

Our error convergence results for this case is shown in figure 4.10. As we see in the figure, these predictions also work quite well. The errors for N_1, labelled as 1D Derivative EC, are approximately equal to the errors for the corresponding value of N_3, labelled as 3D

Derivative EC. We also show the results we obtain when we take training vectors at $N_3 + 1$ random points in the neighborhood of the origin. This data is labelled as 3D Random EC, and the errors match quite well with the 3D Derivative EC results.

4.5 Error Dependence in Eigenvector Continuation on Location of Training Points

As we have mentioned before, the error in eigenvector continuation depends not only on the order of calculation, but also on the location of training points. And since this dependence is very problem specific, it is very hard to answer in general, which training point locations are optimal for any given problem. Nevertheless, in this section we discuss some ideas about this dependence, and how to choose training points more efficiently in a multi-parameter case.

As we already noted, the multi-parameter case reduces to the one parameter case if we select training points along a straight line passing through the target point. This probably is the best strategy for choosing locations for training points if we are interested only at one point, or points along one line.

If the training points lie along a smooth curved path that passes through the target point, then we expect convergence faster than the general case but slower than the straight line example, due to the curvature of the path and the increased path length. We illustrate this with the two parameter example for Model 4,

$$H(c_1, c_2) = H_0 + c_1 H_1 + c_2 H_2. \tag{4.39}$$

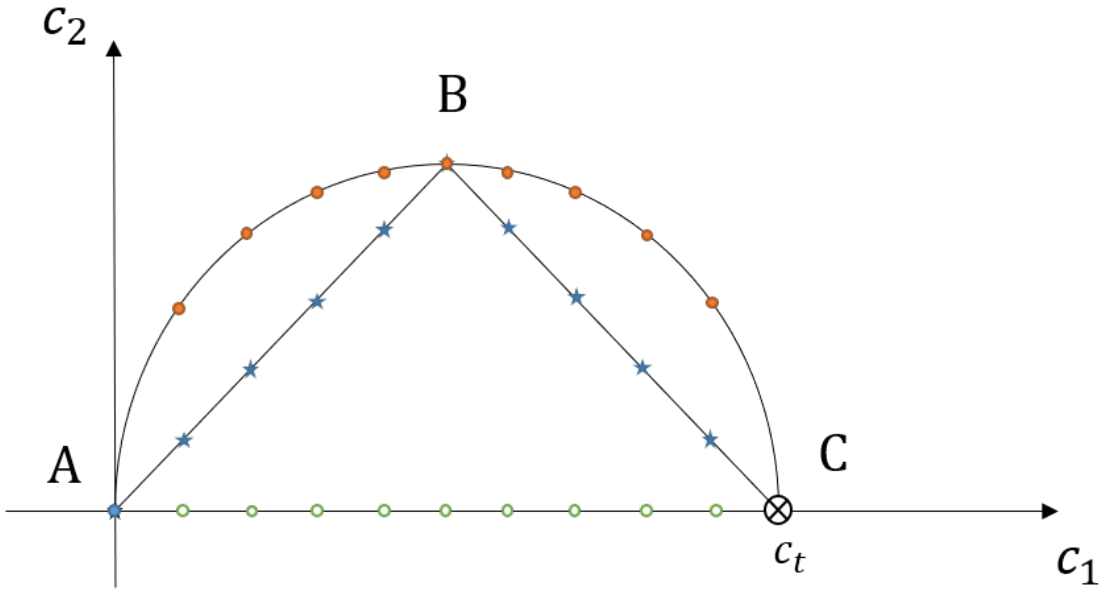


Figure 4.11: We show different choices of path for the eigenvector continuation training vectors. The direct straight line is in green, semicircle is in orange, isosceles right triangle is in blue.

Suppose that our target point, C, is located at $\boldsymbol{c}_t = (2, 0)$. We first consider training points evenly spaced along a straight line from the origin, which we label as point A, to the target point C. This is shown in Fig. 4.11. In Fig. 4.12 we plot the logarithm of the error versus N, where the number of training points is given by $N + 1$, and this data is labelled as 'Straight Line EC'.

Now we consider training points along a semicircular arc ABC, as shown in Fig. 4.11. The convergence for this case is displayed in Fig. 4.12 , and this data is labelled as 'Circle EC'. The convergence is slower than for the straight line example but faster than the general two parameter convergence that we saw in Fig. 4.9. This is in agreement with what we predicted.

Next we consider points along a right isosceles triangle ABC as shown in Fig. 4.11. In this example the path has a discontinuity at B. The convergence for this case is displayed

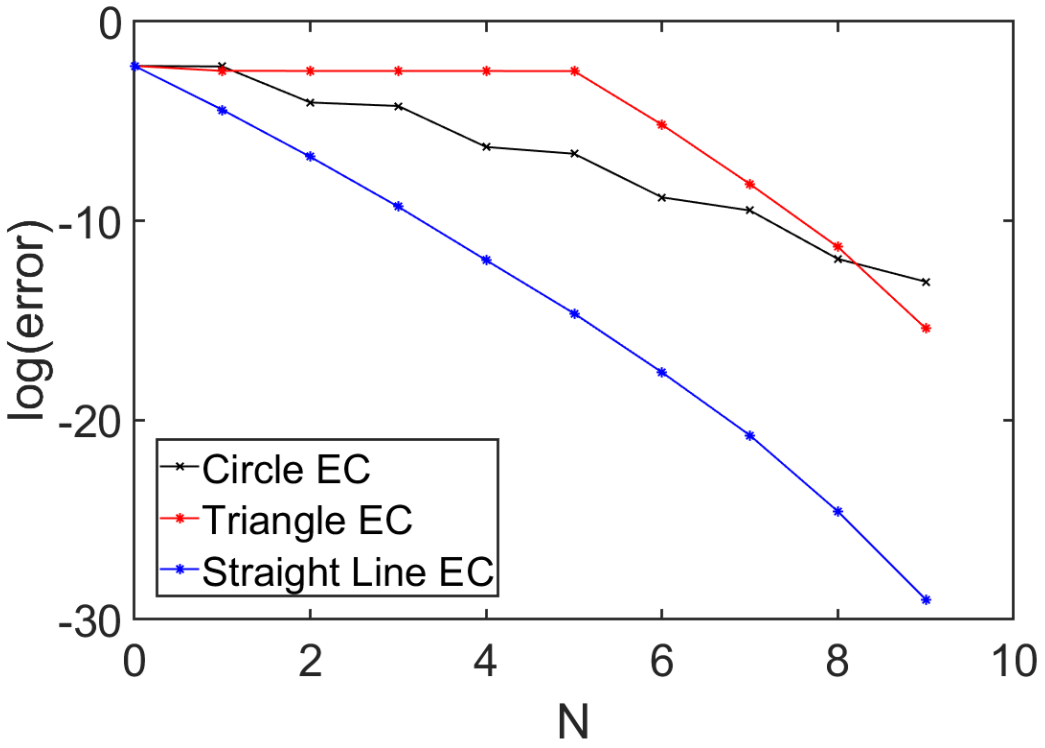


Figure 4.12: Comparison of the logarithm of the error versus order N for different paths for the eigenvector continuation training vectors. The number of training points corresponds to $N + 1$.

in Fig. 4.12 , and this data is labelled as 'Triangle EC'. As one might expect, there is almost no reduction in error as we take training points along the line segment AB, which does not pass through the target point. After we reach B and turn the corner to C, we see that the convergence rate is similar to the one parameter case again. When we are taking training points along the line segment AB, it is as if that eigenvector continuation does not get to see another dimension in the parameter space, but once we turn the corner, all of a sudden the dimension space of training vectors of EC has increased, and now it gives a much better estimate. This shows that the existence of a smooth path connecting the training points and the target is very important for the convergence rate of eigenvector continuation.

It is also worth mentioning how our example with equally spaced points on the straight

line compares with our analysis using derivative vectors. If the training points are not too far apart, then the convergence is similar to that obtained by replacing the training vectors with $N+1$ derivative vectors evaluated at the centroid of the training data. If the training points are spaced far apart, however, then the training points far away from the target point are mostly unhelpful, and the convergence is mostly dominated by the training points closest to the target point. In this situation, although we still call this N^{th} order eigenvector continuation calculation, only a few points in the N training points are actually contributing to the calculation. This is another intricate balance of choosing our training point location and number of training points. A few training points near the target point have much more information than many training points far away from the target point.

The difficulty in optimizing our eigenvector continuation calculations with training points spaced apart, is that for any given target point, we often don't know the optimal region where we should take training points. In our next chapter, we try to answer this question of choosing optimal locations for training points, by using the emulator itself to learn where it should choose its training points. We come up with Self-learning Emulator algorithm, which is an iterative algorithm to choose the best location for the next training point.

Chapter 5

Self-learning Emulators

As we mentioned before, eigenvector continuation builds on the idea that as we vary the parameter(s) in our Hamiltonian, the ground state eigenvector varies in a subspace that has much lower dimensionality than the full dimension of the linear space that the Hamiltonian lives in. And we can learn this subspace in which the target eigenvector lives in through training eigenvectors, taken at certain training points in the parameter space. In this chapter we continue our discussion of how to choose optimal locations for our training points for eigenvector continuation so that we can efficiently learn the subspace in which the target eigenvector lives in.

We also discuss in this chapter about the second advantage of eigenvector continuation that we mentioned in the second chapter - computational advantage of EC and its use in emulator. The dimensionality reduction that eigenvector continuation offers, greatly accelerates the numerical calculation of any problem, and provides us a huge computational speed-up. This speed-up can sometimes be several orders of magnitude faster, and we can use eigenvector continuation to quickly get an estimate at any point in the parameter space, where other conventional methods would take a long time to calculate the exact result. This is justified by that the fact that we can get more than 99.99% accuracy in a fraction of the time we need to calculate the exact result. This is the main idea of an emulator - we can

estimate (emulate) the exact result at any point very quickly using an emulator, rather than performing a long exact calculation.

This observation that eigenvector continuation can function as an accurate emulator for quantum systems, was quite recent [3], and this was followed by a number of new developments and applications [4, 36–41].

The interest in emulators across many scientific disciplines has been long standing one. There has been great interest in using machine learning tools to build efficient emulators that predict scientific processes beyond what is possible with direct calculations [42–45]. A common problem in all these cases is that the emulator must be trained using a large amounts of training data, which is often hard to collect since the required computations are difficult and expensive. It is the same case with eigenvector continuation emulator. We need training eigenvectors for eigenvector continuation calculations, and the more training points we have, the better our EC approximation. However, we need to perform exact calculations at these training points, which can be computationally difficult. Ideally, we want the best results with as few training points as possible. And this brings us back to the problem of optimizing our choice of number of training points, and our choice of their location.

In this chapter, we build on our discussion in the previous chapter, and first show how we could come up with a simple algorithm to choose the locations of our training points based on orthogonality of the training eigenvectors. We then refine this idea to formulate another algorithm to select training points for eigenvector continuation without any prior information about the exact eigenvector at any point. The key idea here is that we can use the eigenvector continuation emulator itself to find the next optimal training point. It turns out that this self-learning algorithm has general applicability, and it can be applied to a whole class of emulators that satisfy some constraint equation. In the following sections, we

will discuss and illustrate this self-learning emulator with several examples.

5.1 Finding Optimal Training Point Locations for Eigenvector Continuation

In the previous chapter, through the orthonormal expansions of vector continuation we have gained a key insight in convergence of eigenvector continuation. The eigenvector continuation method works by projecting the target Hamiltonian onto the subspace spanned by the training eigenvectors, and the higher the effective dimensionality of this subspace spanned by the training eigenvectors, the better our eigenvector continuation estimated will be. The effective dimensionality of this subspace is determined by how orthogonal the training vectors are to each other. If all the training eigenvectors are pointing in almost same direction, the information content in the subspace spanned by these training eigenvectors is low because all the additional eigenvectors after the first eigenvector, are not providing any additional information of the subspace in the target eigenvector lives in. On the other hand, if all the training eigenvectors are orthonormal to each other, then the subspace spanned by these training eigenvectors has the highest possible information content about the subspace in the target eigenvector lives in.

With that in mind, we can come up with an iterative algorithm for selecting the location of next training point in eigenvector continuation. We start with one or two training eigenvectors at a random values of the parameter c_0, and then iteratively add more training eigenvectors. At each iteration, we scan through all the eigenvectors $|\psi(c)\rangle$, and choose the next training location such that the training eigenvectors are as orthogonal as possible. This can be done by selecting the next eigenvector $|\psi(c_i)\rangle$ at location c_i in the i^{th} iteration, such

that the multi-dimensional volume made by the vectors $\{|\psi(c_0)\rangle, \cdots |\psi(c_i)\rangle\}$ is maximized. If we arrange all the training vectors in a matrix form (by putting them as columns),

$$V = \begin{bmatrix} |v(c_1)\rangle & \cdots & |v(c_k)\rangle \end{bmatrix} \tag{5.1}$$

then we can calculate the $k-$dimensional volume formed by these vectors by,

$$volume = Det\{V^T \times V\} \tag{5.2}$$

Since our eigenvectors are normalized to 1, if all of the eigenvectors are orthonormal, then the volume is 1. If they are not orthogonal, then it will be less than 1, and we can use this volume as a measure of how orthogonal our training eigenvectors are. If at any iteration, after scanning through all eigenvectors $|\psi(c)\rangle$, the addition of the new eigenvector reduces the volume of the training eigenvectors significantly, then we know that new eigenvector is not giving us a lot of new information about the subspace in which the target eigenvector lives. This would means that we already have enough training points, and additional training points do not increase the accuracy of the eigenvector continuation estimation by much.

So, in this algorithm, we start with one or two random training points, and keep adding more training points iteratively, until the volumes become less than a specified amount. Note that we are not removing any training points that we already selected earlier, but in a practical application this can be done if we want to further optimize our choice of training points. Let us now show with an example what the results look like. In fact, we have already shown the result with this algorithm in chapter 2, when we presented an eigenvector continuation emulator with 10 dimensional random matrices. We reproduce the plot and

show it in figure 5.1 for convenience.

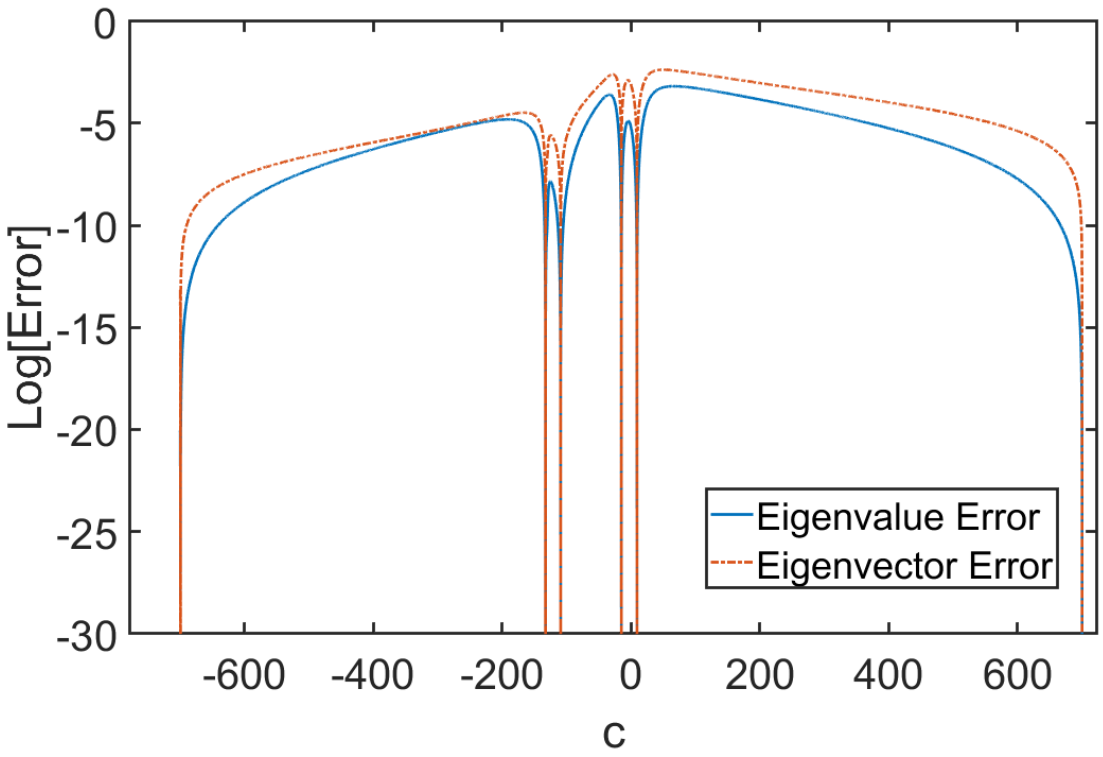


Figure 5.1: Eigenvector continuation emulator - we emulate approximate eigenvalue problem solution everywhere between $-700 \leq c \leq 700$ using six training eigenvectors. The training eigenvectors are chosen such that the volume formed by these vectors is maximized. The final volume is less than 0.02.

In this example, we consider Hamiltonian of the form $H = H_0 + cH_1$, with H_0, H_1 being random Hermitian matrices given in chapter 2. We start with two random training points at $c = -693.4$ and $c = -115$, and add more training points according to the algorithm above. We stop when the volume is less than 0.02. Now, we may ask why the training points are chosen in weird clusters of two. We ideally want far away training points to maximize the volume formed by the training eigenvectors, however this algorithm is picking nearby training points. This can be explained once we look at the plot of avoided-level crossings in figure 5.2.

As we can see, there are several avoided level crossings near $c = -120$ and $c = 0$. As we

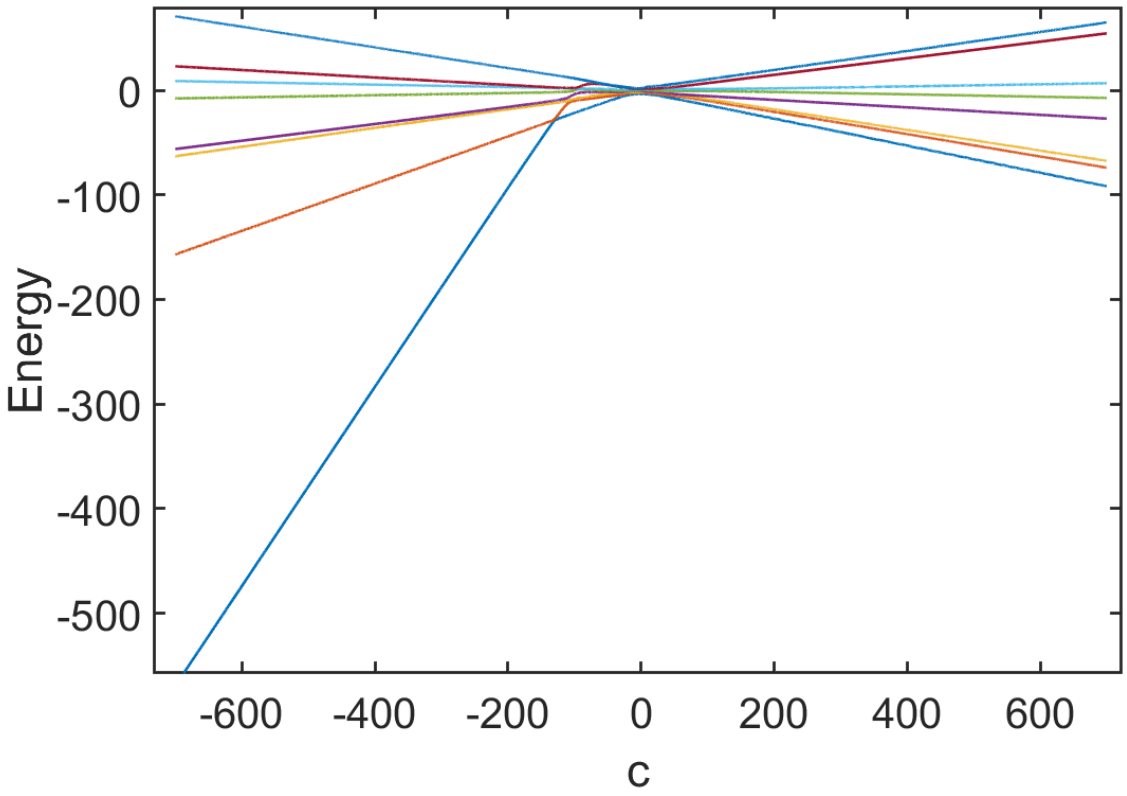


Figure 5.2: Energies for different excited states for the random matrix example given in chapter 2. After an avoided level crossing, the excited eigenvector changes with another eigenvector, and if we want to capture this information we need training points on both sides of the avoided level crossing. This explains our choice of training points in figure 5.1.

go across these level crossings different eigenvectors corresponding to different eigenvalues exchange with each other. eigenvector continuation cannot capture this information with training points on only one side of the level-crossing, because after the level-crossing the eigenvector has changed completely. So we need training points on both sides of the level-crossing, and if there are several nearby avoided level crossings, then we need to take several training points there.

The key idea here is that we want our training eigenvectors to be spaced far apart from each other so that they are pointing in different directions in the subspace in which the target eigenvector lives. While we might be tempted to randomly choose training points far

away from each other, this notion of 'far away' is defined by the avoided level crossings of our problem, and if the crossings are close in the parameter space, we need to take training points that are close to each other. This might look counter intuitive, we are actually taking eigenvectors which greatly differ from each other.

So, for an ideal eigenvector continuation calculation, we want our training points spaced out such that we capture all the information of these avoided level crossings. However, the problem lies in the fact that for a large Hamiltonian matrix, it is not easy to compute the location of all these avoided level crossings. Thus, we have no apriori information about where we should select our training points. But our algorithm of trying to optimize the volume formed by our training eigenvectors does lead us to correct locations for efficient eigenvector continuation calculation.

We can improve this algorithm further by trying to remove some training points when the volume becomes too low, and then adding new training points until the volume becomes small again. We can find out which training eigenvector increases the volume the most upon removal, and remove that vector.We can repeat this process again and again, until we reach an equilibrium point. This will remove the dependence of the algorithm on the random initial training point.

Although this algorithm works very well in finding optimal locations for our training points, it has one massive flaw - we need to compute the exact eigenvector everywhere to be able to compare and choose the eigenvector which maximizes the volume of our set of training eigenvectors. The whole idea behind using an emulator is that direct calculations can be computationally expensive, and we can get a quick approximate using an emulator. If we have to find the exact result everywhere to find the optimal location of our training point, then it beats the point of using an emulator. Nonetheless, this idea of using the volume

formed by the training eigenvectors as a measure of orthogonality between the vectors is still very useful. In any eigenvector continuation calculation, we can use this volume measure to know when we have added too many training points, and to remove some of the training eigenvectors which give the least information about our target subspace. The volume of the eigenvectors helps us find out how many training points we need, and gives us a sense of required dimensionality of the projected space. It is also an important concept that can be used to deal with eigenvector continuation using noisy matrices - a work done by Caleb Hicks in our research group. We will not describe the noise correction work here.

We now have a clear goal of what we want. We want training points as orthogonal to each other as possible, and again we want to add training points iteratively, learning more about the target subspace with each additional training eigenvector. But now we want to find where to take our next training point without actually calculating the exact eigenvector anywhere.

Now we can make use of the fact that eigenvector continuation tries to give us an approximate solution to the eigenvalue problem of the target Hamiltonian $H(c_t)$. If at the target parameter c_t, the exact energy and exact eigenvector are denoted by $E(c_t)$ and $|\psi(c_t)\rangle$ respectively, then they must satisfy,

$$H(c_t)\,|\psi(c_t)\rangle = E(c_t)\,|\psi(c_t)\rangle \tag{5.3}$$

If eigenvector continuation gives us an approximation of energy $\tilde{E}(c_t)$ and approximation to eigenvector by $|\psi(c_t)\rangle_{EC}$, then we can immediately test how good these approximations are, by substituting $\tilde{E}(c_t)$ and $|\psi(c_t)\rangle_{EC}$ in equation 5.3, and seeing how well the equation is satisfied. But we have to define a way to measure how "well" the constraint equation

is satisfied, and this can be done in many ways. For instance, we can use the residual $H(c_t)\,|\psi(c_t)\rangle_{EC} - \tilde{E}(c_t)\,|\psi(c_t)\rangle_{EC}$ as a measure of how well the eigenvalue equation is satisfied. If the eigenvector continuation approximation is very close to the exact results, then the EC approximation should satisfy eigenvalue equation very well, and we should have a very small residual. However, if the EC approximation at some point is bad, then we will have a large residual.

Note that with a given number of training eigenvectors, we do not need to calculate the exact eigenvector anywhere to calculate the eigenvector continuation approximation over the entire parameter space. As we mentioned earlier, calculating the exact result can be computationally expensive, and we only have to do it a few times when we are calculating the exact eigenvector at a training point. In each iteration, after we use eigenvector continuation to emulate over all parameter space, we find the point where the residual (as defined above) is highest, and we calculate the exact eigenvector once at that point. We include that eigenvector in our set of training eigenvectors for our next iteration.

That is all we need to do to find our next optimal training point. The key idea in this self-learning eigenvector continuation algorithm is how we use the constraint equation, namely eigenvalue problem, to quickly determine where our approximation error is largest. The eigenvector at this point has lowest overlap with the linear span of our current set of eigenvectors, and we choose that eigenvector as our next training eigenvector. The reason behind this choice is that among all possible eigenvectors, this eigenvector will have the largest component along the direction orthogonal to the subspace spanned by our current set of eigenvectors, and thus will make the volume formed by the new set of eigenvectors the largest.

Now notice that even if we were not solving an eigenvalue problem, and we wanted

to solve a different constrained equation, we can still use this idea of using the constraint equation to find how good any given approximation to the actual solution is. Thus this idea of self-learning can be used with any emulator, as long as we have a constraint equation that the exact solution satisfies. In the next section, we describe in detail this self-learning emulator algorithm for the general case. We then show its applicability in a variety of examples.

5.2 Self-learning Emulator

Although we described the self-learning algorithm with the example of eigenvector continuation emulator, in this section we will again begin from scratch and describe the detailed algorithm for general use. We begin this section with an introduction to emulators, and what properties we want in a good emulator. We state the training problem that emulators face, and then we explain the self-learning algorithm and show how it helps emulators overcome this problem. This problem of choosing optimal training data is a machine learning problem at its heart, and along the way we will review some machine learning literature as well.

An emulator is a form of data modelling where we build fundamental relationships between a collection of observed input variables and some desired output variables. It is a fast-to-evaluate statistical approximation of a detailed mathematical model. It is useful because the detailed calculation can be slow or relatively expensive, and emulators can be orders of magnitude faster to evaluate. Some examples of usage of emulators include searching for exotic particles in high-energy physics [46], detecting likelihood of disease progression in a patient [47], and emulating eigenvectors of a Hamiltonian in nuclear physics. Here we are interested in the last problem, but different kinds of emulators are in use in several different

disciplines of science.

A good emulator should be detailed enough to capture salient aspects of the mathematical problem at hand, but a highly detailed model can make it computationally expensive, hindering its usability. An emulator is designed using some training points where the exact data is known, and the emulator should faithfully reproduce the exact solution at those training points. Everywhere else, the emulator provides an approximate to the exact result, and this approximation error decreases with the number of training points as either a power law for piecewise continuous emulators or exponentially fast for smooth function emulators. However, more training points means more computational time to emulate, and we want the best possible emulator with least number of training points. This means we want to optimize the location of a fixed number of training points to get better performance. This can be quite a challenge since finding the exact result anywhere is computationally expensive, and each time we want to evaluate the performance of the emulator with a set of training points, we need to solve for the exact solution at those training points. We cannot exhaustively search for all set of training eigenvectors to find the optimal solution.

The problem at hand is about optimally selecting a subset of a large dataset that we want to choose as training points. If we use machine learning to solve this problem, then it becomes an active-learning problem. Active learning (also called "query learning," or sometimes "optimal experimental design" in the statistics literature) is a subfield of machine learning that deals with choosing an optimal subset from a large amount of data. The key hypothesis is that, if the learning algorithm is allowed to choose the data from which it learns it will perform better with less training. It has been used in a variety of fields [48–50] and is often combined with other machine learning methods like neural networks.

Now we introduce an active learning protocol called self-learning emulation that relies

on a fast estimate of the emulator error. It is a greedy local optimization algorithm and it becomes progressively more accurate as the emulator improves. It provides a potential solution to the problem of selecting optimal training data for emulators, when the objective of the emulator is to solve a system of constraint equations over some domain of control parameters. As we will show, self-learning emulators are highly efficient algorithms that speed up our calculations by several orders of magnitude or more. The gain in computational speed is achieved by using the emulator itself to estimate the error.

We note that the self-learning emulators we discuss here are qualitatively different from other machine learning algorithms that model the solutions using gradient descent optimization of some chosen loss function. Although these gradient descent optimization methods are highly parallelizable and very fast, they usually suffer from critical slowing down with respect to error convergence and cannot achieve arbitrarily high accuracy in polynomial computing time.

We can summarize the self-learning algorithm as follows. It is an iterative algorithm to choose training points, and we start with a few random training points. We use the constraint equation and the emulator to find over our parameter space a fast estimate of error in emulation. We then find the point where the emulation error is largest, and then choose our next training point there. And we iterate this algorithm to find and add more training points. We will now describe in details on how to get this fast error estimate.

We should also mention that this idea of using the emulator to find additional training points has been applied before. In [51], along with the usual estimate of the actual result, the emulator also gave an estimate of error in emulation, and the active learning algorithm was based on picking the next training point where the emulated error was the largest. While this idea is similar to ours, our method of calculating estimated error is different, and we

achieve much better results with self-learning emulation.

5.2.1 Constraint equations and error estimates

Suppose we have a set of simultaneous constraint equations $G_i(\mathbf{x}, \mathbf{c}) = 0$ in variables $\mathbf{x} = \{x_j\}$, and control parameters $\mathbf{c} = \{c_k\}$, which vary over some domain $\mathbf{D}$. Let us denote the exact solutions as $\mathbf{x}(\mathbf{c})$. We assume that we have an emulator which uses the exact solutions for some set of training points $\{\mathbf{c}^{(i)}\}$ and constructs an approximate solution $\tilde{\mathbf{x}}(\mathbf{c})$ for all $\mathbf{c} \in \mathbf{D}$.

We define the error or loss function as the norm $\|\Delta\mathbf{x}(\mathbf{c})\|$ of the residual vector $\Delta\mathbf{x}(\mathbf{c}) = \mathbf{x}(\mathbf{c}) - \tilde{\mathbf{x}}(\mathbf{c})$. We want to choose a set of training points for our emulator such that the peak value of the error function over the domain $\mathbf{D}$ is minimized. And we will choose this set of training points iteratively such that the peak error goes down with every iteration. We want to choose the point where the error function $\Delta\mathbf{x}(\mathbf{c})$ is largest as our next training point. However, we cannot calculate the error function $\Delta\mathbf{x}(\mathbf{c})$ without calculating the exact solution $\mathbf{x}(\mathbf{c})$ there, and as we have discussed before, calculating the exact solution is slow and computationally expensive, and we cannot find it for all $\mathbf{c} \in \mathbf{D}$. Thus, in each iteration, we want to find an estimate of the error function, which should be much faster to calculate than calculating the exact solution. We will call this error estimate as *fast error estimate* or simply *fast estimate*.

Since the error function will vary over many orders of magnitude, it is more convenient to work with the natural logarithm of the error function, $\log\|\Delta\mathbf{x}(\mathbf{c})\|$. The emulator will reproduce the exact solution at the training points $\{\mathbf{c}^{(i)}\}$, and the logarithm of the error function will become negative infinity. Therefore, the logarithm of the error function will become a rapidly varying function of $\mathbf{c}$ as we include more training points.

Let us consider the case where $\Delta\mathbf{x}(\mathbf{c})$ is small enough that we can accurately expand the constraint equations as

$$G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c}) + \Delta\mathbf{x}(\mathbf{c})\cdot\nabla_{\mathbf{x}} G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c}) \approx 0. \tag{5.4}$$

If the number of degrees of freedom is small, we can solve the linear inversion problem for $\Delta\mathbf{x}(\mathbf{c})$. The solution for $\Delta\mathbf{x}(\mathbf{c})$ will be a fast estimate for the error. This estimate is just what we would expect from the multivariate form of the Newton-Raphson method.

However, for most cases of interest, there will be many degrees of freedom and the matrix inversion required to solve for $\Delta\mathbf{x}(\mathbf{c})$ will be too slow or computationally impossible for our self-learning emulator training process. We therefore choose another non-negative functional $F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}]$ as a substitute for $\|\Delta\mathbf{x}(\mathbf{c})\|$. The only essential requirement we impose on $F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}]$ is that it is linearly proportional to $\|\Delta\mathbf{x}(\mathbf{c})\|$ in the limit $\|\Delta\mathbf{x}(\mathbf{c})\| \to 0$. This allows us to write the logarithm of the error as

$$\log\|\Delta\mathbf{x}(\mathbf{c})\| = \log F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}] + A + B(\mathbf{c}), \tag{5.5}$$

where A is a constant and the average of $B(\mathbf{c})$ over the domain $\mathbf{D}$ is zero.

Since $F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}]$ is linearly proportional to $\|\Delta\mathbf{x}(\mathbf{c})\|$ in the limit $\|\Delta\mathbf{x}(\mathbf{c})\| \to 0$, the function $\log F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}]$ will have the same steep hills and valleys as the function $\log\|\Delta\mathbf{x}(\mathbf{c})\|$ as we include more training points. In the limit of large number of training points, we can neglect the much smaller variation of $B(\mathbf{c})$ over the domain $\mathbf{D}$. We can therefore approximate the logarithm of the error as $\log F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}),\mathbf{c})\}] + A$. Now the unknown constant A is irrelevant if we are comparing the logarithm of the error for different

points $\mathbf{c}$. Nevertheless, we can also quickly estimate A simply by taking several random samples of $\mathbf{c}$ and computing the average value of the difference between $\log\|\Delta\mathbf{x}(\mathbf{c})\|$ and $\log F[\{G_i(\tilde{\mathbf{x}}(\mathbf{c}), \mathbf{c})\}]$. Later we will show that we can even refine this estimate further using machine learning to approximate the function $B(\mathbf{c})$.

A greedy algorithm is an approach for solving a problem by selecting the best option available at the moment. It doesn't worry whether the current best result will bring the overall optimal result. The algorithm never reverses the earlier decision even if the choice is wrong. It works in a top-down approach. And the self-learning emulator training program is a greedy algorithm since in each iteration, we search over parameter space to find the point $\mathbf{c}$ where the logarithm of the error is greatest at the moment. We then add this point to the training set and repeat the whole process.

In this manner we have constructed a fast emulator that becomes more and more accurate as more training points are added and provides a reliable estimate of the emulator error. It should be emphasized that the self-learning emulation is just an algorithm to learn the best training points for the emulator, and it does not change the process of emulation itself. Thus it can be used with any emulator with a constraint and reproduces the exact solution at the training points. This could be a simple method such as polynomial interpolation or a Gaussian process, or a more involved method such as neural networks or eigenvector continuation. We retain all the beneficial properties of the emulator such as its computational speed advantage, parallelizablilty, ease of application, etc. It can be applied to any system of constraints such as solutions of algebraic or transcendental equations, linear and nonlinear differential equations, and linear and nonlinear eigenvalue problems.

We now show how this algorithm works in several examples.

5.2.2 Natural Cubic Spline Emulator

In mathematics, a spline is a special function defined piecewise by polynomials. If we have a given set of data points $(x_0, y_0), \cdots, (x_n, y_n)$, then a spline interpolator tries to fill in the data between the given data points by piecewise polynomial functions. The data points, where different piecewise polynomial functions converge from different sides, are called *knots*. A common spline is the natural cubic spline, which uses piecewise cubic polynomials to interpolate in between the data points. The word 'natural' refers to the fact that these polynomials are joined C^2 smoothly. In the one dimensional case, this simply means that the first and second derivatives at the knots are equal from both sides. We describe in the supplemental materials how from a given data set $(x_0, y_0), \cdots, (x_n, y_n)$, we can form a set of n cubic polynomials that approximate the data in between the data points.

Let us now consider some natural cubic spline emulators and apply self-learning algorithm to it. In the first example we consider, Model 1, we are interested in finding the smallest root of an n^{th} order polynomial of the form $p(x) = c_n x^n + \cdots + c_1 x + c_0$, where n is odd and the coefficients are real. Since n is odd, the polynomial crosses zero at least once if $c_n \neq 0$, and so we can uniquely define the lowest real root. We will consider the case where all of the coefficients are fixed and only the coefficient c_{n-1} is varied. Let x be the exact lowest real root, and this will be a function of c_{n-1}. We are interested in finding $x(c_{n-1})$ as c_{n-1} varies over some domain D, but instead of finding polynomial roots at every point, we want to find the exact root only at a few points, and we want to use that data to build a cubic spline emulator that will approximate the smallest root everywhere else. If the order of polynomial n is greater than 4, there is no algebraic solution, and the roots must be solved numerically. Now a direct numerical root calculation will take a long time, and we want to

get a quick approximation using splines, which will be faster.

Our goal with spline emulator is to have lowest peak emulation error. We want to optimize our choice of location for the training points that the emulator will work with, but we don't know the emulation error anywhere without calculating the exact root at that point. So we perform self-learning algorithm to find the best set of points.

Let the cubic spline approximation be $\tilde{x}(c_{n-1})$ for any given c_{n-1}. The logarithm of the error function is then $\log|\Delta x(c_{n-1})|$ where $\Delta x(c_{n-1}) = x(c_{n-1}) - \tilde{x}(c_{n-1})$. We want to come up with a fast error estimate that we can use to guess where the error is highest. For our fast error estimate $F[\tilde{x}(z,c),c]$, we need some function that is linearly proportional to the actual error $\|\Delta x(z,c)\|$ in the limit $\|\Delta x(z,c)\| \to 0$. There are many good choices one can make, and here we choose,

$$|\Delta x(c_{n-1})| \approx \frac{|p(\tilde{x}(c_{n-1}))|}{\sqrt{|p'(\tilde{x}(c_{n-1}))|^2 + \epsilon^2}}, \tag{5.6}$$

where we have included a small regulator ϵ to avoid divergences when the derivative p' vanishes. This estimate $|\Delta x(c_{n-1})|$ is again motivated by the Newton-Raphson method. We use the right-hand side of equation 5.6 for our error estimate.

For our example, we take a fifth-order polynomial, $n = 5$, with coefficients $c_5 = 5, c_3 = 3, c_2 = 2, c_1 = 1, c_0 = 1$. We vary c_4 in the region $0 \leq c_4 \leq 100$. We also set $\epsilon = 1$ as defined in equation 5.6. We start with four training points for c_{n-1}, with two on the boundary and two in random points in between. In Fig. 5.3 we show results for the estimate of the logarithm of the error, and how the training process occurs. At our training points, we manually set the error to be exp(-25). For each iteration, the self-learning algorithm is choosing a new training point that corresponds to the location of the maximum of estimated

error function from the previous iteration.

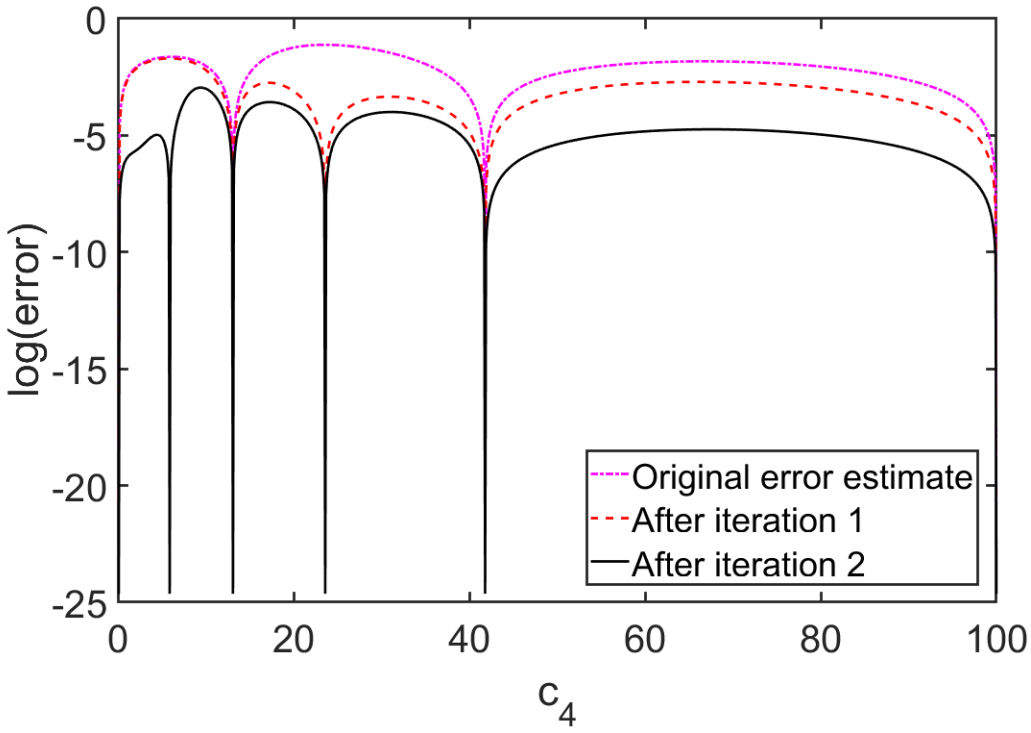


Figure 5.3: Estimates of the logarithm of the error for the cubic spline self-learning emulator, which finds the lowest real root of the fifth order polynomial $p(x)$ in Model 1. We show results after iteration 0, 1, and 2.

In Fig. 5.4 we show that there is excellent agreement between the logarithm of the error for the actual error and the error estimate after 10 iterations. We note that the error has dropped significantly for all values of c_4 from the original error estimate in Fig. 5.3, indicating that the self-learning emulator is functioning as intended. The fact that more training points are needed for smaller values of c_4 shows that the training process is not simply adding more training points at random, but is instead uniformly improving the emulator performance across the entire domain.

We also note that in this case the error estimate is matching well with the actual error. Therefore both A and $B(\mathbf{c})$ as defined in equation 5.5 are negligible for this example.

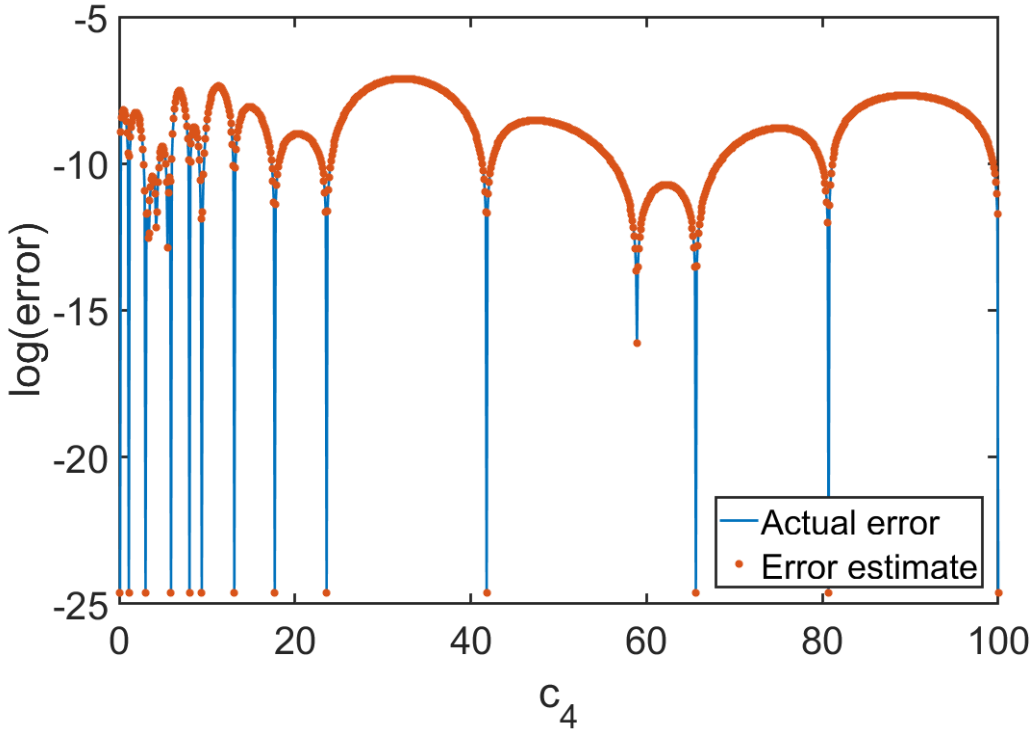


Figure 5.4: Logarithm of the actual error and error estimate for the cubic spline self-learning emulator in Model 1 after 10 iterations.

For this Model 1, it is relatively easy to numerically compute the roots of a 5^{th} order polynomial, but the spline emulator is still faster. And we find that our self-learning protocol provides a factor of about 200 times greater computational speed over tuning the cubic spline emulator using direct calculations of the error. Let us consider a slightly more challenging problem for us to numerically solve. After that we will discuss and define the speed-up factor for our emulator more precisely.

We again want to use a natural cubic spline emulator, but this time we are interested in solving a little harder problem. In this example, Model 2, we want to find the lowest real solution of a transcendental equation given by,

$$p(x) = c_5 x^5 + c_4 x^4 \sin(10x) + c_3 x^3 + c_2 x^2 + c_1 x + c_0 = 0, \tag{5.7}$$

where all the coefficients c_i are real. We again fix coefficients $c_5 = c_3 = c_2 = c_1 = c_0 = 1$, and we vary the coefficient c_4 in the region $-1 \leq c_4 \leq 2$. We are interested in the smallest real x that satisfies equation 5.7, and for this choice of coefficients, we know that a real solution for x always exist for real c_4, however the dependence of the solution on c_4 is not trivial and has discontinuities with respect to parameter c_4.

We apply the self-learning algorithm to train the natural cubic spline emulator for this problem. We start with three training points for c_4, two on the boundary and one in the interior. We denote the cubic spline approximation by $\tilde{x}(c_4)$ for all values of c_4. The logarithm of the error function is then $\log|\Delta x(c_4)|$ where $\Delta x(c_4) = x(c_4) - \tilde{x}(c_4)$. We can again estimate $|\Delta x(c_4)|$ using the Newton-Raphson method, and we use the right-hand side of equation 5.6 for our error estimate with $\epsilon = 1$.

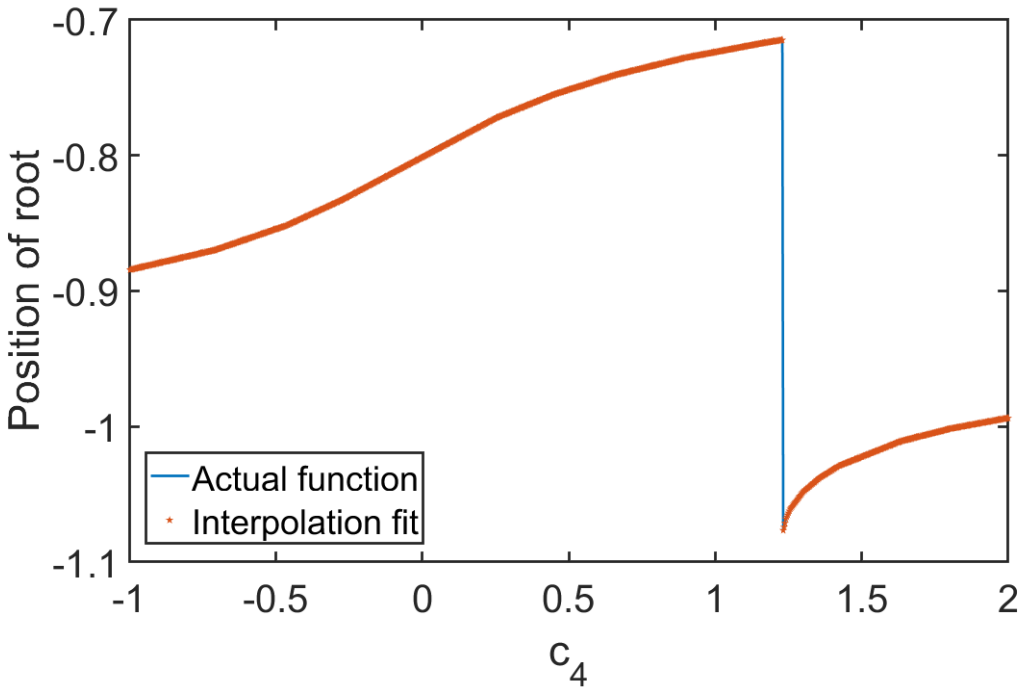


Figure 5.5: Plot of the lowest real solution to Eq. (5.7) versus c_4 in Model 2. The self-learning emulator needs to take significantly more training points near the discontinuity at $c_4 \approx 1.232$.

The smallest solution to the equation is shown in Fig. 5.5. We also show how well our spline emulator does in approximating the function everywhere. We notice that the dependence of the solution on variable c_4 is non-trivial, and there is a discontinuity at $c_4 \approx 1.232$.

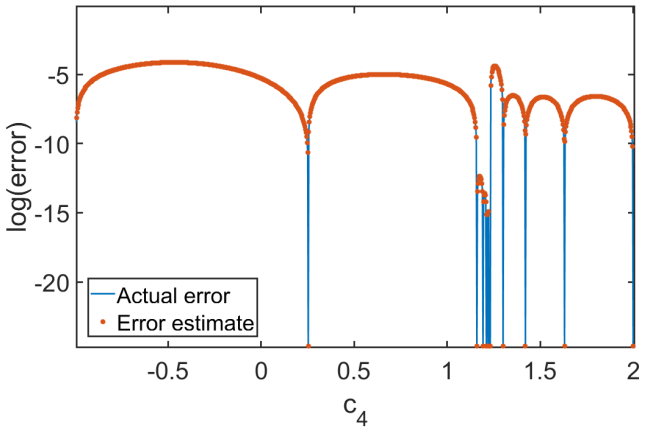


Figure 5.6: Logarithm of the actual error and error estimate for the cubic spline self-learning emulator in Model 2 after 10 iterations.

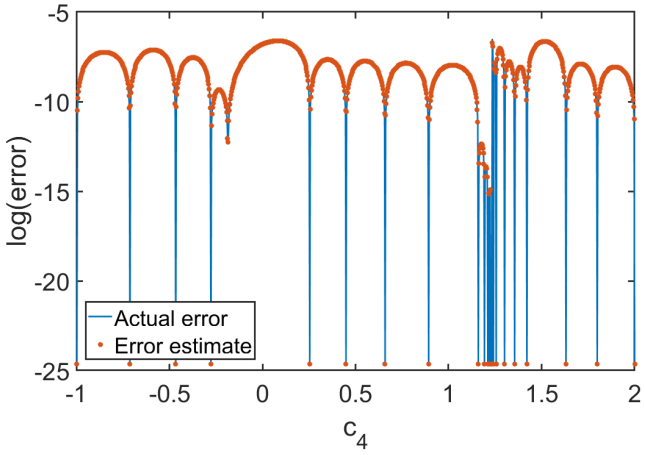


Figure 5.7: Logarithm of the actual error and error estimate for the cubic spline self-learning emulator in Model 2 after 20 iterations.

Figures 5.6 and 5.7 show the logarithm of the error estimate and actual error after 10 and 20 iterations of self-learning algorithm respectively. The fact that more training points are needed near $c_4 \approx 1.2$ shows that the training process is not simply adding more training

points at random, but is instead choosing training points which add the most information to the emulation process. The self-learning emulator took significantly more training points near the discontinuity to accurately emulate it.

Also notice that the peak error has dropped from figure 5.6 to 5.7, as we add another 10 training points. This shows that if we keep continuing in this manner, we can reach arbitrary precision with our emulators. Again our error estimates are matching well with the actual error, and thus both A and $B(\mathbf{c})$ as defined in equation 5.5 are negligible for Model 2.

Now let us comment on the error scaling in spline emulators, and how much speed-up factor do we get from using this spline emulator. In the limit of large number of training points, N, the error for the spline interpolation for a smooth function scales as $O(N^{-4})$ [52]. This is because the error of the cubic interpolation scales as the fourth power of the interval between training points. However, this is only true when the function is smooth, and in the limit that N is large. For Model 2, however, the exact solution has a jump discontinuity, and so the power law scaling is slower. Numerically, we find that the error is approximately $O(N^{-2.2})$. We see this in figure 5.8, where the slope of the graph is -2.2.

Using an Intel i7-9750H processor, evaluating the exact solution using standard root finding methods for one value of c_4 requires about 10^{-1} s of computational time. In comparison, it takes about 10^{-6} s to estimate the solution for any c_4, using spline interpolation with 23 training points. So the raw emulator speedup factor is therefore $s_{\text{raw}} \sim 10^5$. Let M be the number of evaluations of needed, i.e., number of points in our domain except for training points. Let N_ϵ be the number of emulator training points needed to achieve error tolerance ϵ. The overall computational speedup factor for the self-learning emulator can then be estimated by the minimum of M/N_ϵ and raw emulator speed-up factor. If the fast error estimate were not used, then N_ϵ would be replaced by the number of evaluations needed to

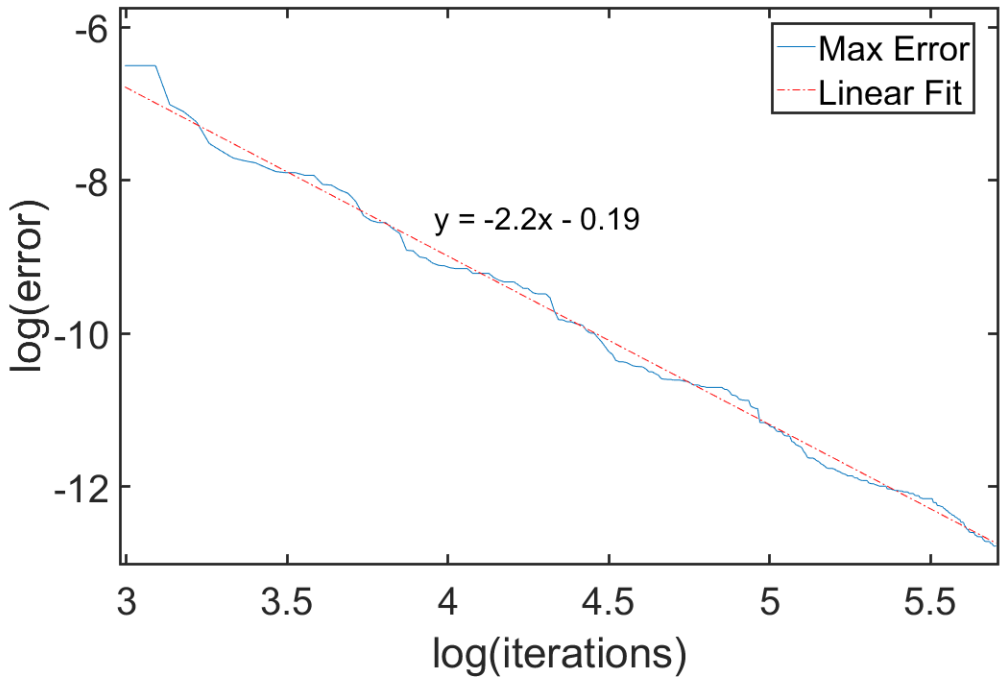


Figure 5.8: Natural spline emulator error scaling for Model 1. We plot the logarithm of the error versus the logarithm of the number of iterations.

train the emulator to the desired error tolerance ϵ, which is generally much larger than N_ϵ.

5.2.3 Reduced Basis Emulator

We now turn our attention to solving differential equations with emulators. In our next example, Model 3, we will emulate the solution of an ordinary differential equation with one variable z and one control parameter c. We consider a family of differential equations $Lx(z) = 0$, where

$$L = \frac{1}{(1+2z)^2}\frac{d^2}{dz^2} - \frac{2}{(1+2z)^3}\frac{d}{dz} + c^2 e^{2c}, \tag{5.8}$$

and c is a real parameter. Our boundary conditions are $x(z=0,c) = 0$ and $\partial_z x(z=0,c) = 1$ for all c. We consider the region $0 \leq z \leq 1$, and $0 \leq c \leq 1$. The exact solution is given by

$x(z,c) = \frac{1}{ce^c} \sin[ce^c(z+z^2)]$.

For this problem, we consider two different emulators. The first is the natural spline emulator, which we have already seen. We can use splines to perform interpolations and extrapolations in c for each value of z. The second emulator is a reduced basis emulator, which uses high-fidelity solutions of the differential equation for several training values of c and solves the constraint equations approximately using subspace projection. Reduced basis (RB) emulators have been proven useful for solving computationally-intensive parameterized partial differential equations [53–57]. We also tried to emulate the solution to this differential equation using neural networks, and Gaussian Process (GP) emulators, but they did not perform very well, and we omit their results.

We again want a fast error estimate $F[\tilde{x}(z,c),c]$ that is linearly proportional to the actual error $\|\Delta x(z,c)\|$ in the limit $\|\Delta x(z,c)\| \to 0$. There are many ways to choose this, but we again choose an expression that we can relate to 5.4. We define our error estimate as,

$$F[\tilde{x}(z,c),c] = \left\| \frac{L\tilde{x}(z,c)}{\sqrt{\left(\frac{d}{dz}L\tilde{x}(z,c)\right)^2 + \epsilon^2}} \right\|_1 , \tag{5.9}$$

where we have again included a small regulator ϵ to avoid divergences. Here we are using the L_1 norm, which is the integral over z of the absolute value. Note that this fast error estimate can be used by both the spline emulator and reduced basis emulator. In fact, for solving this differential equation, this fast estimate can be used with any emulator.

We initialize with two training points at the boundaries and one in the interior. Figure 5.9 shows the actual error and estimated error after 20 iterations of the self-learning spline emulator. On the other hand, we found that the reduced basis emulator performs very well

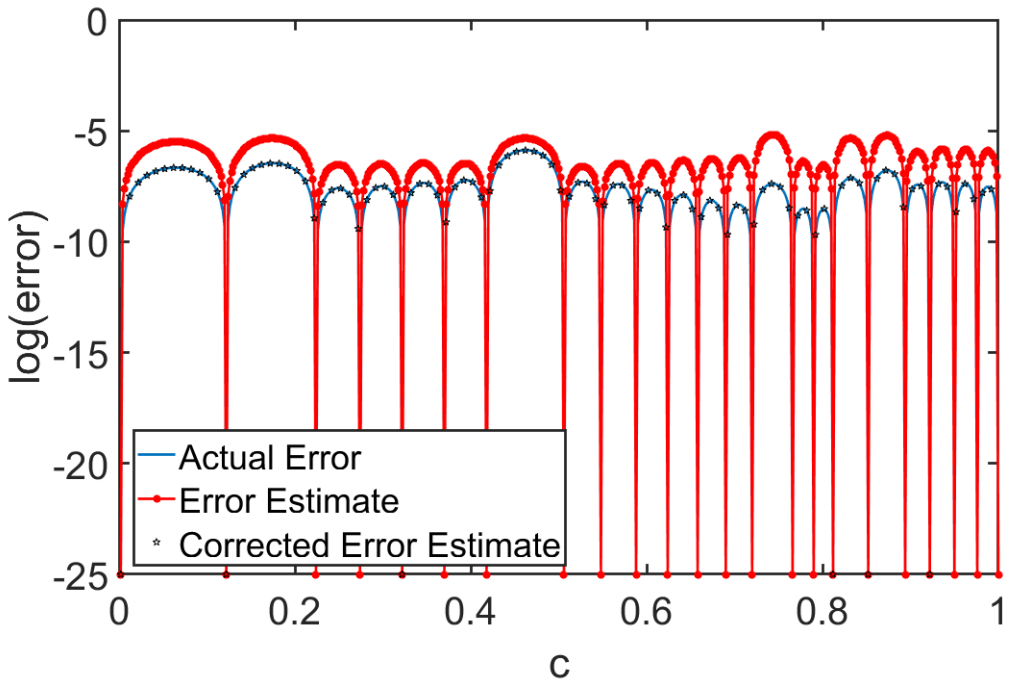


Figure 5.9: Logarithm of the actual error, error estimate, and corrected error estimate for the natural spline emulator with self-learning in Model 3 after 20 iterations.

for this problem, and therefore in order to show more details of the performance before reaching the limits of machine precision, we extend the domain to the wider interval of $0 \leq c \leq 2$ for the RB emulator. Fig. 5.10 shows the actual error and estimated error after 10 iterations of the self-learning RB algorithm.

In both cases the difference between the actual error and estimate error is a slowly-varying function of c as predicted. We also note that the exact solution $x(z,c) = \frac{1}{ce^c}\sin[ce^c(z+z^2)]$ oscillates more rapidly with increasing c, and the emulators therefore need more training points for larger c. This again shows self-learning algorithm is selecting training eigenvectors such that the emulator performance across the entire domain is uniformly improving.

In this example, there is a difference between the error estimate and the actual error, indicating that $A + B(c)$ in equation 5.5 is non-zero. So we cannot tell the actual error, just from knowing the error estimate. However, in these situations we can estimate the difference

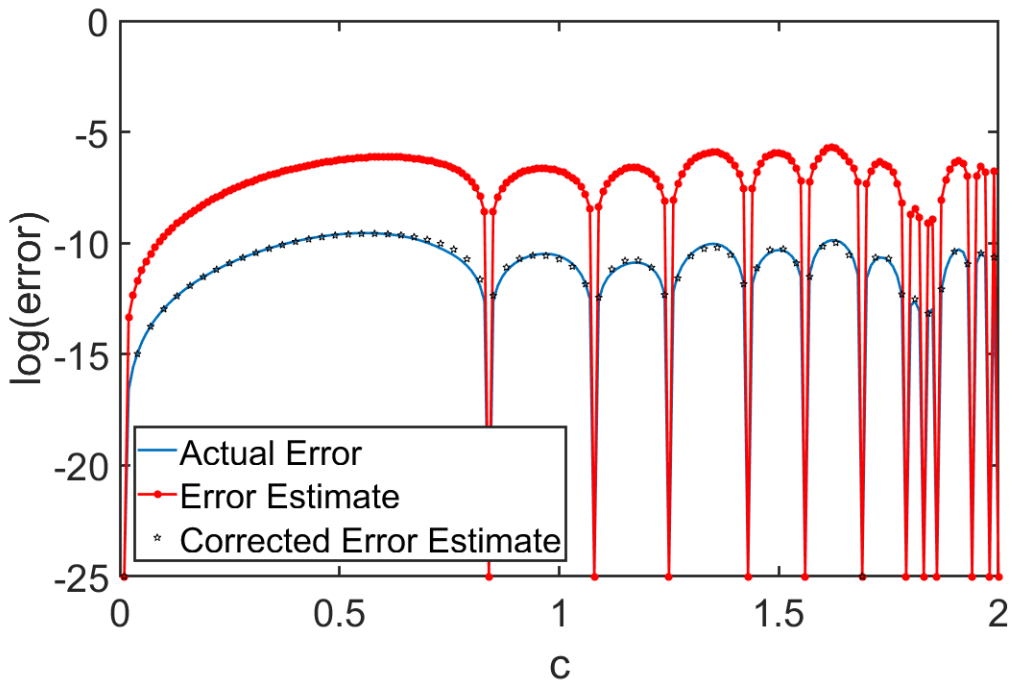


Figure 5.10: Logarithm of the actual error, error estimate, and corrected error estimate for the reduced basis emulator with self-learning in Model 3 after 10 iterations.

between the error estimate and the actual error by constructing a Gaussian Process (GP) emulator for the difference function $A + B(c)$. We train the GP by computing $A + B(c)$ at the midpoints in between the emulator training points. We have performed this correction for both the spline and RB emulators, and the results are shown in Figs. 5.9 and 5.10. We see that the corrected error estimate is in excellent agreement with the actual error.

In this Model 3, the solution is smoothly varying function. So we should expect $O(N^{-4})$ error scaling with spline emulator. However it seems that we have not yet reached the asymptotic scaling large N limit, and the error scaling is approximately $O(N^{-1.88})$. This can be seen in figure 5.11. In contrast, the reduced basis emulator has exponentially fast error scaling. This is because the reduced basis emulator is itself a smooth function. We can view the addition of training points as matching more derivatives of the smooth emulator to derivatives of the smooth exact solution. The error scaling is therefore similar to the error

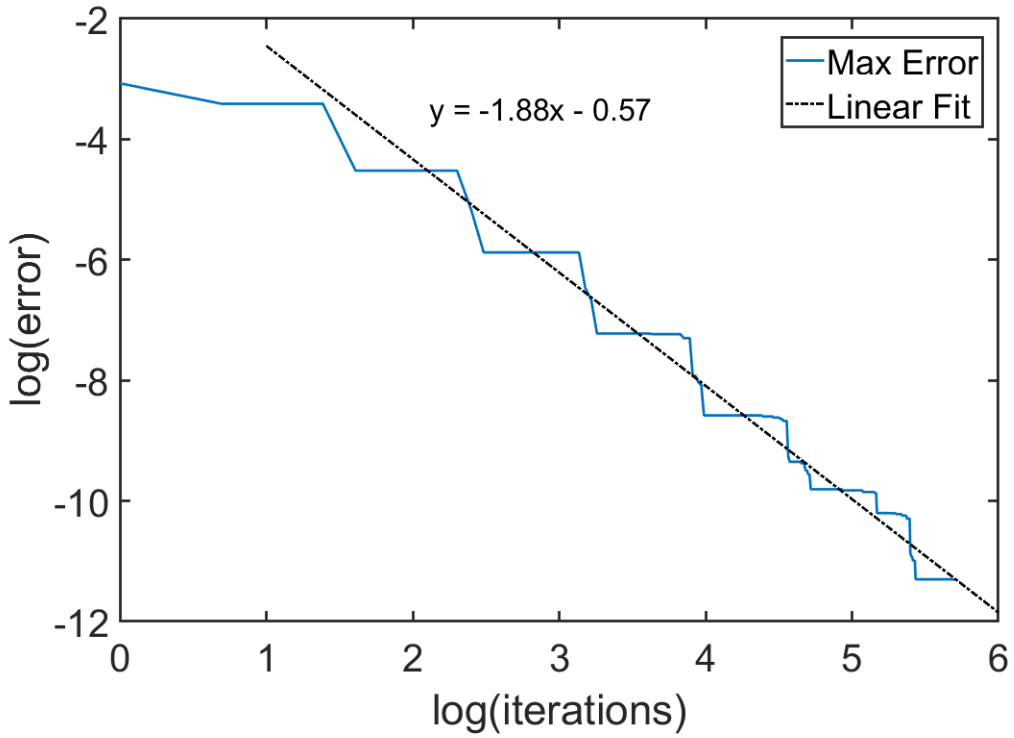


Figure 5.11: Natural spline emulator error scaling for Model 2. We plot the logarithm of the error versus the logarithm of the number of iterations.

scaling of a convergent power series. Figure 5.12 shows the error scaling for the reduced basis emulator for Model 3. We see that the error scaling is $O(e^{-2.66N})$, for N above 10 training points.

On a single Intel i7-9750H processor, numerically solving the differential equation for one value of c takes about 7×10^{-2} s. In contrast the spline emulator requires about 1.7×10^{-3} s for 23 training points, and the RB emulator takes about 5.5×10^{-4} s for 13 training points. Therefore the spline emulator has a raw speedup factor of $s_{\text{raw}} \sim 40$, while the RB emulator has a raw speedup factor of $s_{\text{raw}} \sim 130$. Given the somewhat comparable values for s_{raw} and the exponential scaling of the error for the RB emulator, we conclude that the RB emulator significantly outperforms the spline emulator for this example.

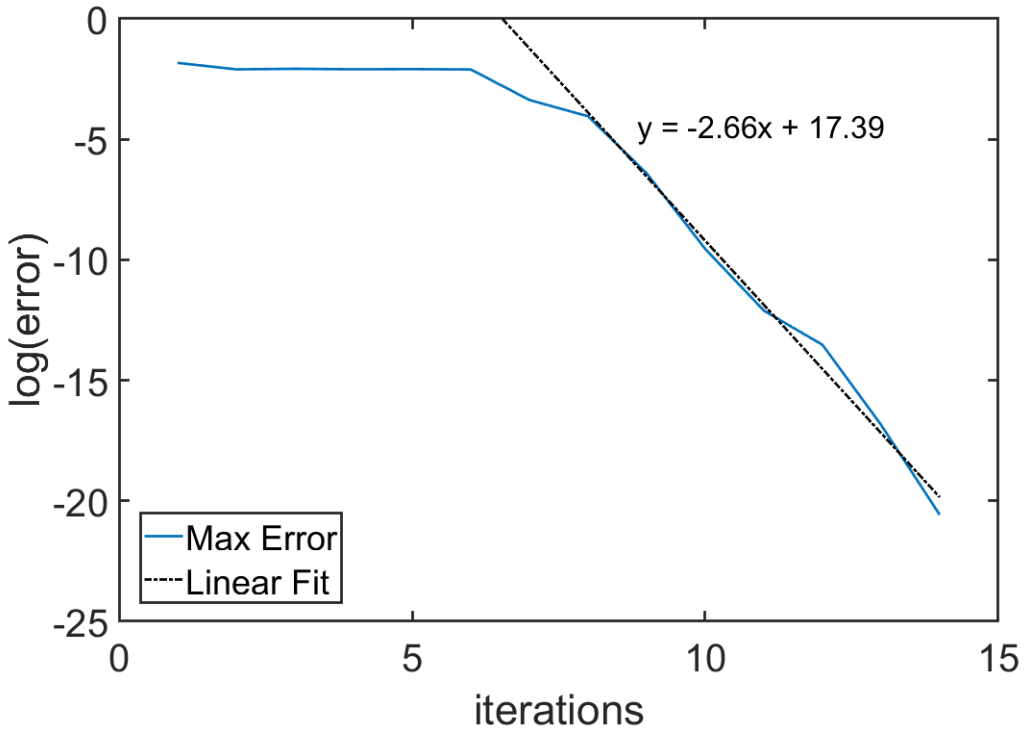


Figure 5.12: Reduced basis method error scaling for Model 2. We plot the logarithm of the error versus the number of iterations.

5.3 Self-learning Eigenvector Continuation

We return to our discussion of eigenvector continuation and show that self-learning eigenvector continuation works very well in selecting optimal training points for us. Suppose we are interested in finding the ground state eigenvector $|v(\mathbf{c})\rangle$, and eigenenergy $E(c)$ of the Hamiltonian $H(\mathbf{c})$. Let the eigenvector continuation estimate of eigenvector be $|\tilde{v}(\mathbf{c})\rangle$, and the estimate of eigenenergy be $\tilde{E}$.

The logarithm of the error is $\log\||\Delta v(\mathbf{c})\rangle\|$, where $|\Delta v(\mathbf{c})\rangle = |v(\mathbf{c})\rangle - |\tilde{v}(\mathbf{c})\rangle$. Computing the error directly will be computationally too expensive for large systems, and so we will instead work with a fast error estimate $F[\tilde{v}(\mathbf{c}), H(\mathbf{c})]$. Earlier we had mentioned that $|H(c)\,|\tilde{v}(c)\rangle - \tilde{E}(c)\,|\tilde{v}(c)\rangle\,|$ could work as a fast error estimate. Here we show that we can

come up with a better error estimate,

$$F[\tilde{v}(\mathbf{c}), H(\mathbf{c})] = \sqrt{\frac{\langle \tilde{v}(\mathbf{c})|[H(\mathbf{c}) - \tilde{E}(\mathbf{c})]^2|\tilde{v}(\mathbf{c})\rangle}{\langle \tilde{v}(\mathbf{c})|[H(\mathbf{c})]^2|\tilde{v}(\mathbf{c})\rangle}}. \tag{5.10}$$

This $F[\tilde{v}(\mathbf{c}), H(\mathbf{c})]$ is proportional to the square root of the variance of the Hamiltonian, but it will be linearly proportional to $\||\Delta v(\mathbf{c})\rangle\|$ in the limit $\||\Delta v(\mathbf{c})\rangle\| \to 0$. Thus, $\log F[\tilde{v}(\mathbf{c}), H(\mathbf{c})]$ can be used as a fast error estimate.

We will now present a geometrical picture of eigenvector continuation error, as well as some additional insight into why we choose the error estimate given in equation 5.10). Suppose we know the eigenvectors at M different training points, $\{\mathbf{c}^{(1)}, \cdots, \mathbf{c}^{(M)}\}$. We label the set of M training eigenvectors as $S_M = \{|v(\mathbf{c}^{(1)})\rangle, \cdots, |v(\mathbf{c}^{(M)})\rangle\}$. Let us define the norm matrix $\mathcal{N}(S_M)$ as

$$\begin{bmatrix} \langle v(\mathbf{c}^{(1)})|v(\mathbf{c}^{(1)})\rangle & \cdots & \langle v(\mathbf{c}^{(1)})|v(\mathbf{c}^{(M)})\rangle \\ \vdots & \ddots & \vdots \\ \langle v(\mathbf{c}^{(M)})|v(\mathbf{c}^{(1)})\rangle & \cdots & \langle v(\mathbf{c}^{(M)})|v(\mathbf{c}^{(M)})\rangle \end{bmatrix}, \tag{5.11}$$

and let $\Omega^2(S_M)$ be the determinant of $\mathcal{N}(S_M)$. Then $\Omega^2(S_M)$ corresponds to the square of the volume of the M-dimensional parallelopiped defined by the vectors in the set S_M. If all the eigenvectors are normalized, then the maximum possible volume is 1, which is attained when all the eigenvectors are orthogonal.

Let us now consider selecting the next training point, $\mathbf{c}_{M+1}$. Let P be the projection operator onto the linear span of S_M, and let Q be the orthogonal complement so that $Q = 1 - P$. Suppose we now expand our training set S_M by adding another training vector

$|v(\mathbf{c})\rangle$ to form S_{M+1}. Let us define the perpendicular projection vector $|v_\perp(\mathbf{c})\rangle$ as

$$|v_\perp(\mathbf{c})\rangle = Q\,|v(\mathbf{c})\rangle\,. \tag{5.12}$$

Since $\Omega^2(S_M)$ is the squared volume of the parallelopiped defined by the vectors in S_M and $\Omega^2(S_{M+1})$ is the squared volume of the parallelopiped defined by the vectors in S_{M+1}, it follows that the ratio $\Omega^2(S_{M+1})$ to $\Omega^2(S_M)$ is given by the squared norm of $|v_\perp(\mathbf{c})\rangle$,

$$\frac{\Omega^2(S_{M+1})}{\Omega^2(S_M)} = \langle v_\perp(\mathbf{c})|v_\perp(\mathbf{c})\rangle\,. \tag{5.13}$$

Let us define the projections of H onto P and Q subspaces as

$$H^P(\mathbf{c}) = PH(\mathbf{c})P, \qquad H^Q(\mathbf{c}) = QH(\mathbf{c})Q. \tag{5.14}$$

The eigenvector continuation approximation is just the approximation of $|v(\mathbf{c})\rangle$ by some eigenvector of $H^P(\mathbf{c})$, which we denote as $|v^P(\mathbf{c})\rangle$. This is because eigenvector continuation just gives us the eigenvector of the Hamiltonian projected on to the space of training eigenvectors, which is the P subspace. Let the corresponding energy be labelled $E^P(\mathbf{c})$ so that

$$H^P(\mathbf{c})\,|v^P(\mathbf{c})\rangle = E^P(\mathbf{c})\,|v^P(\mathbf{c})\rangle\,. \tag{5.15}$$

We also label the eigenvectors of $H^Q(\mathbf{c})$ contained in the orthogonal complement Q as,

$$H^Q(\mathbf{c})\,|v_j^Q(\mathbf{c})\rangle = E^Q(\mathbf{c})\,|v_j^Q(\mathbf{c})\rangle\,. \tag{5.16}$$

When the difference between the exact eigenvector and the eigenvector continuation estimate of the exact eigenvector is small, we can use first order perturbation theory to write

$$|v(\mathbf{c})\rangle \approx |v^P(\mathbf{c})\rangle + \sum_j \frac{\langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{E^P(\mathbf{c}) - E_j^Q(\mathbf{c})} |v_j^Q(\mathbf{c})\rangle \,. \tag{5.17}$$

To first order in perturbation theory, the residual vector is just $|v_\perp(\mathbf{c})\rangle \approx |v(\mathbf{c})\rangle - |v^P(\mathbf{c})\rangle$. We therefore have

$$|v_\perp(\mathbf{c})\rangle \approx \sum_j \frac{\langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{E^P(\mathbf{c}) - E_j^Q(\mathbf{c})} |v_j^Q(\mathbf{c})\rangle \tag{5.18}$$

If we now combine with Eq. (5.13), we get

$$\frac{\Omega^2(S_{M+1})}{\Omega^2(S_M)} = \||v_\perp(\mathbf{c})\rangle\|^2 = \sum_j \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle \langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{[E^P(\mathbf{c}) - E_j^Q(\mathbf{c})]^2}. \tag{5.19}$$

We can now connect this result with the fast error estimate function in equation 5.10. The second part of the equation gives an expression for the error term $\||v_\perp(\mathbf{c})\rangle\|$ using first-order perturbation theory, and the first part of the equation is a geometric interpretation of the error term as the ratio of the squared volumes, $\Omega^2(S_{M+1})$ to $\Omega^2(S_M)$. Taking the logarithm of the square root in equation 5.19, we get

$$\log\||v_\perp(\mathbf{c})\rangle\| = \frac{1}{2}\log \sum_j \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle \langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{[E^P(\mathbf{c}) - E_j^Q(\mathbf{c})]^2}. \tag{5.20}$$

The term in the numerator,

$$\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle \langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle \,, \tag{5.21}$$

will go to zero at each of the training points, causing large variations in the logarithm of the error as we add more and more training points. In contrast, the term in the denominator, $[E^P(\mathbf{c}) - E_j^Q(\mathbf{c})]^2$, will be smooth as a function of c. Similarly, $\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle$ will also be a smooth function of $\mathbf{c}$. We can write

$$\frac{1}{2}\log\sum_j \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle\,\langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{[E^P(\mathbf{c}) - E_j^Q(\mathbf{c})]^2} =$$
$$\frac{1}{2}\log\sum_j \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle\,\langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle} + A + B(\mathbf{c}), \tag{5.22}$$

where A is a constant and $B(\mathbf{c})$ averages to zero over the entire domain of $\mathbf{c}$. While the function $B(\mathbf{c})$ is unknown, it will be dominated by the large variations in the logarithm of the error as more and more training points are added. We note that

$$\begin{aligned}
&\sum_j \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})|v_j^Q(\mathbf{c})\rangle\,\langle v_j^Q(\mathbf{c})|H(\mathbf{c})|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle} \\
&\quad= \frac{\langle v^P(\mathbf{c})|H(\mathbf{c})(1-P)(1-P)H(\mathbf{c})|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle} \\
&\quad= \frac{\langle v^P(\mathbf{c})|[H(\mathbf{c}) - H^P(\mathbf{c})]^2|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle} \\
&\quad= \frac{\langle v^P(\mathbf{c})|[H(\mathbf{c}) - E^P(\mathbf{c})]^2|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle}.
\end{aligned} \tag{5.23}$$

We therefore arrive at the variance error estimate used in equation 5.10,

$$\log\||v_\perp(\mathbf{c})\rangle\| = \frac{1}{2}\log\frac{\langle v^P(\mathbf{c})|[H(\mathbf{c}) - E^P(\mathbf{c})]^2|v^P(\mathbf{c})\rangle}{\langle v^P(\mathbf{c})|[H(\mathbf{c})]^2|v^P(\mathbf{c})\rangle} + A + B(\mathbf{c}). \tag{5.24}$$

Thus, with the fast error estimate of equation 5.10, we apply self-learning to eigenvector continuation emulator. We consider the ground state of a system of four distinguishable par-

ticles with equal masses on a three-dimensional lattice with zero-range interactions. We will call this example Model 4. We use lattice units where physical quantities are multiplied by powers of the spatial lattice spacing to make the combinations dimensionless. Furthermore, we set the particles masses to equal 1 in lattice units. We label the particles as $1, 2, 3, 4$ and take the control parameters to be the six possible pairwise interactions, c_{ij}, with $i < j$. The lattice volume is a periodic cube of size $L^3 = 4^3$, and the corresponding Hamiltonian is a linear space with $262,144$ dimensions. This model can be viewed as a generalization of the four two-component fermions with zero-range interactions that we considered in chapter 4, and considered in Ref. [33, 39], or the Bose-Hubbard model considered in Ref. [1].

Let $\mathbf{n}$ denote the spatial lattice points on our three dimensional L^3 periodic lattice. Let the lattice annihilation and creation operators for particle i be written as $a_i(\mathbf{n})$ and $a_i^\dagger(\mathbf{n})$ respectively. The free non-relativistic lattice Hamiltonian has the form

$$\begin{aligned} H_{\text{free}} = & \frac{3}{m} \sum_{i=1,2,3,4} \sum_{\mathbf{n}} a_i^\dagger(\mathbf{n}) a_i(\mathbf{n}) \\ & - \frac{1}{2m} \sum_{i=1,2,3,4} \sum_{\hat{\mathbf{l}}=\hat{\mathbf{1}},\hat{\mathbf{2}},\hat{\mathbf{3}}} \sum_{\mathbf{n}} a_i^\dagger(\mathbf{n}) \Big[a_i(\mathbf{n}+\hat{\mathbf{l}}) + a_i(\mathbf{n}-\hat{\mathbf{l}}) \Big]. \end{aligned} \tag{5.25}$$

which is the same as the one we have seen in chapter 4. However, instead of being fermions, they are distinguishable particles, and all of them can be on one site. We add to the free Hamiltonian single-site contact interactions, and the resulting Hamiltonian then has the form

$$H = H_{\text{free}} + \sum_{i<j} \sum_{\mathbf{n}} c_{ij} \rho_i(\mathbf{n}) \rho_j(\mathbf{n}), \tag{5.26}$$

where $\rho_i(\mathbf{n})$ is the density operator for particle i,

$$\rho_i(\mathbf{n}) = a_i^\dagger(\mathbf{n}) a_i(\mathbf{n}). \tag{5.27}$$

For calculations discussed in this work, we use a basis of position eigenstates on the lattice.

We would like to study the appearance of interesting structures such as particle clustering [58,59] in the ground state wave function as a function of the six coupling parameters c_{ij}. As noted in Ref. [58], we can determine the formation of particle clusters by measuring the expectation values of products of local density operators. For example, $\rho_{ij}(\mathbf{n}) = \rho_i(\mathbf{n})\rho_j(\mathbf{n})$ can serve as an indicator of two-particle clusters, $\rho_{ijk}(\mathbf{n}) = \rho_i(\mathbf{n})\rho_j(\mathbf{n})\rho_k(\mathbf{n})$ for three-particle clusters, and $\rho_{ijkl}(\mathbf{n}) = \rho_i(\mathbf{n})\rho_j(\mathbf{n})\rho_k(\mathbf{n})\rho_l(\mathbf{n})$ for a four-particle cluster.

Such detailed multi-parameter studies are very difficult due to the number of repeated calculations necessary. However, we now show that self-learning emulation with eigenvector continuation can make such studies fairly straightforward.

Since it is difficult to visualize data for all six parameters, we first present results corresponding to one two-dimensional slice. We set $c_{14} = c_{23} = c_{24} = c_{34} = -2.3475$ and use eigenvector continuation as an emulator to find the ground state as a function of c_{12} and c_{13} over a square domain where each coefficient ranges from -5 to 5. We initialize the self-learning emulator with one random training point for some value of c_{12} and c_{13}. When searching for new training points, we use the method of simulated annealing [60] with an energy functional given by $-\log F[\tilde{v}(\mathbf{c}), H(\mathbf{c})]$.

In figure 5.13 we show the logarithm of the error and error estimate obtained after 40 iterations. In (a) we show the logarithm of the actual error, and in (b) we show the logarithm of the estimated error. As predicted in equation 5.5, we see that the two plots are

approximately the same up to a constant offset A, with $A \approx -2.3$. The peak value of the actual error is $\|\langle|\Delta v(\mathbf{c})\rangle\| = 2 \times 10^{-5}$. From the figure we see that the local maxima of the error reside along an approximately flat horizontal surface. The flatness of this surface indicates that our self-learning emulator is performing as intended, with the training algorithm removing the peak error at each iteration. This is a highly nontrivial result, since the actual error is never calculated in the training protocol. We note that the distribution of training points is far from uniform. The region near the line $c_{12} + c_{13} = -1$ has a higher density of training points, indicating that the ground state wave function has a more complicated dependence on c_{12} and c_{13} in that location.

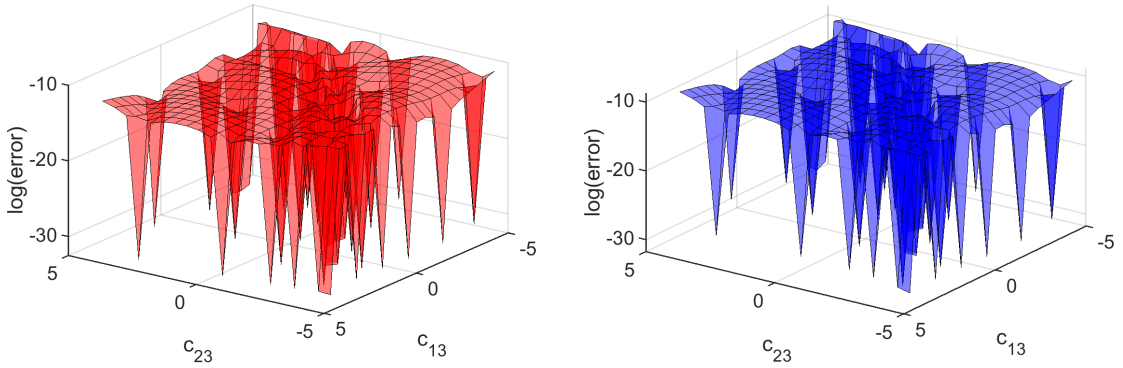


Figure 5.13: Logarithm of the error in Model 2 after 40 iterations using self-learning EC. In (a) we show the logarithm of the actual error (red), and in (b) we show the logarithm of the estimated error (blue).

With the help of self-learning algorithm, we can now measure short-range correlations between pairs of particle in the ground state wave function for all values of c_{12} and c_{13}. In figure 5.14 we show the short-range correlations for pairs of particles 1 with 2 and 1 with 3. The correlation function ρ_{12} measures the probability that particles 1 and 2 occupy the same lattice site, and the correlation function ρ_{13} measures the probability that particles 1 and 3 occupy the same lattice site. We see that ρ_{12} is close to zero when c_{12} is positive and

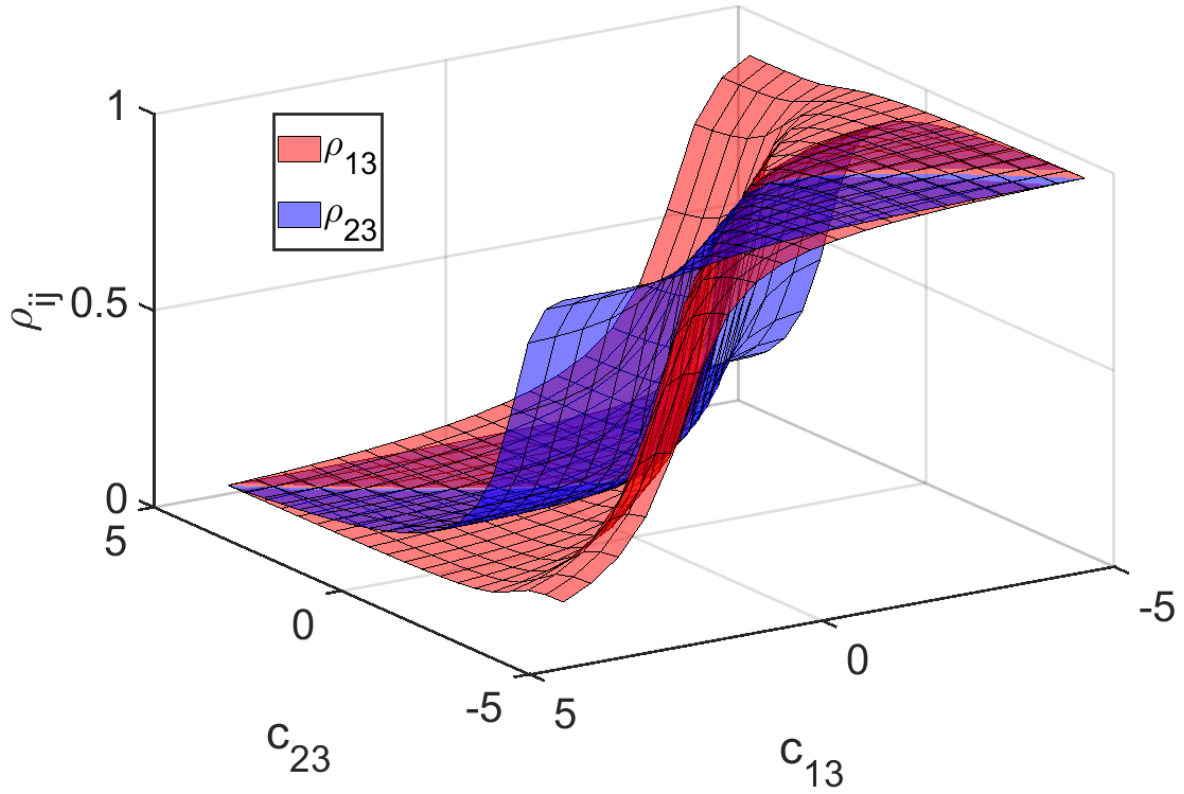


Figure 5.14: Plot of the two-particle short-range correlations in Model 2. ρ_{13} (red) measures the probability that particles 1 and 3 occupy the same lattice site, and the correlation function ρ_{23} (blue) measures the probability that particles 2 and 3 occupy the same lattice site.

rises to a peak of 1 when c_{12} is negative and increasing in magnitude. Similarly, ρ_{13} is close to zero when c_{13} is positive and rises to a peak of 1 when c_{13} is negative and increasing in magnitude. This is consistent with what we should expect - positive (repulsive) coupling should make the particles go away from each other, and negative (attractive) coupling should bring the particles together on the same lattice site. The change in clustering probability is most prominent near the line $c_{12} + c_{13} = -1$, and this is why our self-learning emulator selects much of its training points on this line. An examination of the other short-range correlation functions, $\rho_{14}, \rho_{23}, \rho_{24}, \rho_{34}$, shows that for all negative values of c_{12} and c_{13} in our domain, the four-body system remains a compact bound state. However, when c_{12} or

c_{13} are positive, we induce a repulsive spatial separation between the corresponding particles within the bound state wave function.

For the eigenvector continuation emulator in Model 4, we again expect exponential error scaling because both the emulator and exact solution are both smoothly varying functions. In figure 5.15 we show the error scaling for the eigenvector continuation emulator in Model 4. We see that the error scaling is $O(e^{-0.27N})$. Using an Intel i7-9750H processor, we found that direct calculation of the eigenvector and eigenvalue takes about 1.95 s, whereas eigenvector continuation emulation with 41 training points can be done in 0.013 s. This corresponds to a raw speedup factor of $s_{\text{raw}} \sim 150$.

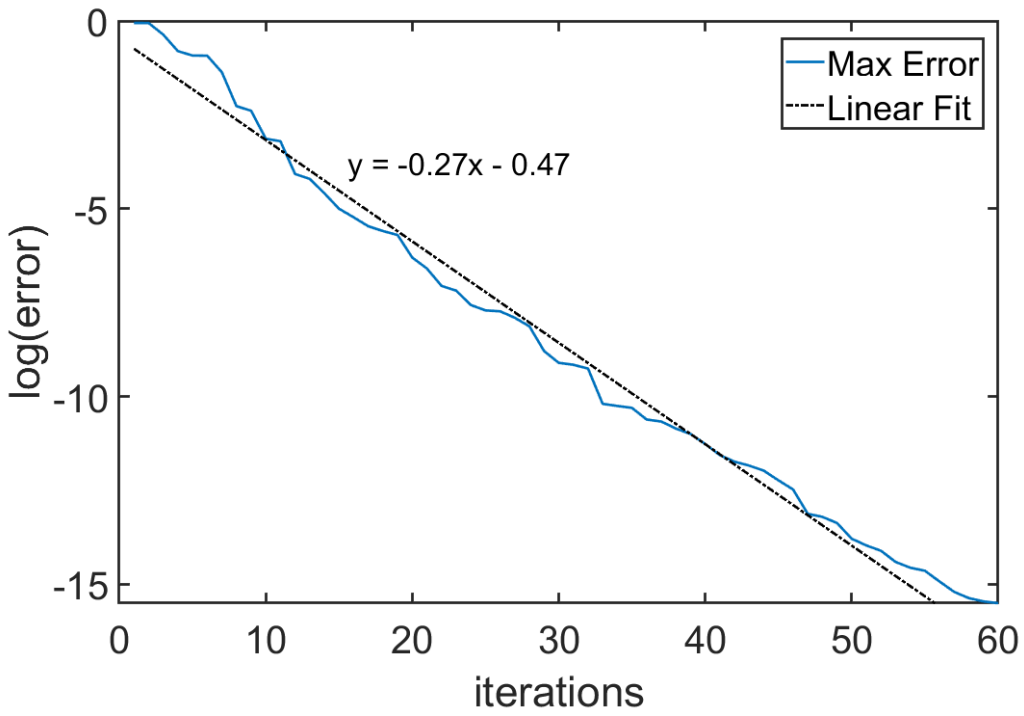


Figure 5.15: Eigenvector continuation emulator error scaling for Model 3. We plot the logarithm of the error versus the number of iterations.

After studying the case where we vary two parameters, we now consider the case where we vary each of the six control parameters c_{ij} in Model 4. We restrict the range for each

of these coefficients from $-5 \leq c_{ij} \leq 0$. We again use simulated annealing with the energy functional $-\frac{1}{2}\log F[\tilde{v}(\mathbf{c}), H(\mathbf{c})]$ to find the optimal training points. After learning the eigenvector manifold with self-learning eigenvector continuation, we calculate the exact error at a few random points in order determine the unknown constant A in equation 5.5 and determine the logarithm of the error. After 80 iterations, the peak value of the error is $\||\Delta v(\mathbf{c})\rangle\| = 4 \times 10^{-3}$.

This highlights the main result of this chapter. We are now able to choose our training points efficiently, and can perform multi-dimensional eigenvector continuation with ease. In our numerical calculation with the six parameter case, we take 10 points along each parameter c_{ij}, so our 6-dimensional space has a total of 10^6 points. We want eigenvector continuation emulator to emulate over all these points, and for that we select only about 80 training points. Through self-learning, our emulator picks 80 training points out a million, such that it can emulate over all the million points with accuracy less than 4×10^{-3} everywhere. This is definitely a non-trivial result that will be hard to achieve without self-learning. Through this example, we have shown how we can answer the question of how to pick our training points optimally.

Chapter 6

Nuclear Lattice EFT and Floating-block Method for Eigenvector Continuation

In this chapter, we consider the nuclear interaction on a lattice, and describe how we can apply eigenvector continuation to our Nuclear Lattice Effective Field Theory (EFT) calculations. We do not provide an introduction to this subject here. The interested reader should look here [61]. We only describe our floating-block algorithm and present preliminary results.

Our Hamiltonian depends on several parameters such as two-body and three-body interaction strength, Coulomb interaction strength, coupling for one pion exchange potential, etc. Our calculations are affected by the Monte-Carlo sign problem, which affects us more when the couplings are larger, and this limits the maximum value of the couplings with which we can perform our Monte-Carlo calculations. Our goal is to use eigenvector continuation to extrapolate from a coupling region where the sign problem is minimal and we can perform accurate calculations, to a coupling region where the sign problem dominates.

To demonstrate the eigenvector continuation calculation, we consider the simple problem when we set all interaction coupling to zero, except the leading order SU(4) contact interaction. Our Hamiltonian has then only one control parameter, which we will denote by c. For quick calculations, we consider Helium-4 nucleus with Lattice size $L = 4$.

This project is work in progress. We have tested the algorithm in the case where we only vary the leading order SU(4) contact interaction. In future, we wish to test the extrapolation in other parameters, like Coulomb interaction, strength of one-pion exchange potential, etc.

6.1 Floating Block Method

In Monte-Carlo Lattice EFT, we do not have access to the eigenvectors directly, however we can calculate the matrix elements. Our main tool here is Euclidean Time Evolution, which is implemented with auxiliary-field Monte Carlo. We start with an initial state and propagate it over several time steps. In each of the time steps we multiply the current state with the transfer matrix, which is given by $\exp(-H(c)\delta t)$. Our initial state will approach the ground state as the number of time steps goes to infinity. With proper normalization, we can easily calculate any Hamiltonian matrix element by,

$$\frac{\langle\psi_{initial}|e^{-H(c)\delta t}|\cdots|H(c)|\cdots|e^{-H(c)\delta t}|\psi_{initial}\rangle}{\langle\psi_{initial}|e^{-H(c)\delta t}|\cdots|e^{-H(c)\delta t}|\psi_{initial}\rangle} \tag{6.1}$$

In practice, we perform our calculations with a finite number of time steps, and then extrapolate to infinite time. We will denote the total number of time steps in our calculations by Lt. We simplify the notation above to write the Hamiltonian matrix elements as,

$$H_{ij}(c_t) = \frac{\langle\psi_{init}|c_i|\cdots|c_i|H(c_t)|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle} \tag{6.2}$$

where $|c_i|$ represents a time step where we multiply by transfer matrix $e^{-H(c_i)\delta t}$, and there $Lt/2$ time steps of $|c_i|$ and $|c_j|$ each, with $H(c_t)$ inserted in the middle. Note that since there are equal time steps of the same coupling in numerator and denominator, the energy terms

cancel out.

Calculating the norm matrix element is non-trivial because if we remove $H(c_t)$ from the middle in the above expression, we just get 1.

$$\frac{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle} = 1 \tag{6.3}$$

So, we reorder the couplings, and move a block of c_j time steps from right to left, to get

$$\frac{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|c_i|\cdots|c_i|c_j|\cdots|c_j|c_j|\cdots|c_j|\psi_{init}\rangle} \tag{6.4}$$

Now, in the numerator we have four blocks of $Lt/4$ time steps with couplings $|c_i|$ and $|c_j|$ interleaved with each other. If we insert $\sum_k |\psi_k\rangle\langle\psi_k| = 1$ in between time steps where the couplings are different, and assume that $Lt/4$ is large enough such that after propagating through $Lt/4$ time steps, only the ground state survives, then we can write equation 6.4 as,

$$\begin{aligned}&\frac{\langle\psi_{init}|c_i|\cdots|c_i|\sum_{k1}|\psi_{k1}\rangle\langle\psi_{k1}|c_j|\cdots|c_j|\sum_{k2}|\psi_{k2}\rangle\langle\psi_{k2}|c_i|\cdots|c_i|\sum_{k3}|\psi_{k3}\rangle\langle\psi_{k3}|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|\sum_{k4}|\psi_{k4}\rangle\langle\psi_{k4}|c_j|\cdots|c_j|\psi_{init}\rangle}\\ =&\frac{\langle\psi_{init}|\psi(c_i)\rangle\langle\psi(c_i)|\psi(c_j)\rangle\langle\psi(c_j)|\psi(c_i)\rangle\langle\psi(c_i)|\psi(c_j)\rangle\langle\psi(c_i)|\psi_{init}\rangle}{\langle\psi_{init}|\psi(c_i)\rangle\langle\psi(c_i)|\psi(c_j)\rangle\langle c_j|\psi_{init}\rangle}\end{aligned} \tag{6.5}$$

where $\psi(c_i)$ is the ground state eigenvector of the Hamiltonian $H(c_i)$.

We can simplify this equation to write equation 6.4 as,

$$\begin{aligned}&\frac{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|c_i|\cdots|c_i|c_j|\cdots|c_j|c_j|\cdots|c_j|\psi_{init}\rangle} = |\langle\psi(c_i)|\psi(c_j)\rangle|^2\\ \Rightarrow N_{ij} =& \Big(\frac{\langle\psi_{init}|c_i|\cdots|c_i|c_j|\cdots|c_j|c_i|\cdots|c_i|c_j|\cdots|c_j|\psi_{init}\rangle}{\langle\psi_{init}|c_i|\cdots|c_i|c_i|\cdots|c_i|c_j|\cdots|c_j|c_j|\cdots|c_j|\psi_{init}\rangle}\Big)^{\frac{1}{2}}\end{aligned} \tag{6.6}$$

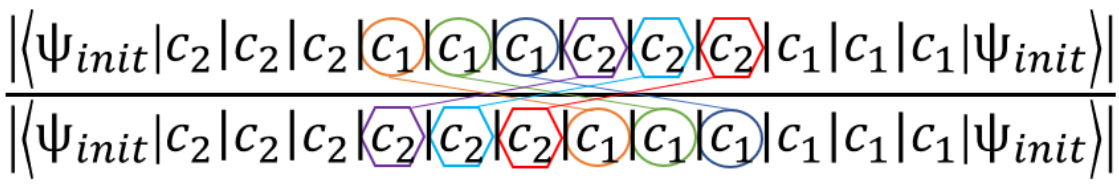


Figure 6.1: We move the auxiliary-fields of different time steps in the numerator so that we have the same auxiliary-field for time steps in numerator and denominator with same coupling. This figure shows the auxiliary-fields are moved around for $Lt = 40$. The different time steps in numerator and denominator with same color have same auxiliary-field.

This gives us a way to calculate the norm matrix. Although equation 6.6 can be used to calculate the norm matrix, we can further reduce the statistical error with another trick. Usually, in auxiliary-field Monte Carlo, in each time step we sample the auxiliary-field and use it for the Hamiltonian in both the numerator and the denominator. In equation 6.6, even with different couplings in the same time step, the numerator and denominator has the same auxiliary-field in the Hamiltonian. However, if we were to move the auxiliary-fields in the numerator such that the same auxiliary-field goes with the same coupling in denominator and numerator, then the only difference between numerator and denominator comes only from the commutator of the transfer matrices with different coupling Hamiltonian. This reduces the uncertainty, and gives us less error. Figure 6.1 demonstrates the idea behind moving the auxiliary-fields of different time steps to reduce the uncertainty in the norm matrix elements with $Lt = 12$. We performed all our Monte-Carlo calculations using this trick.

With both the norm matrix and the Hamiltonian matrix, we can now perform the eigenvector continuation calculation. We tested the method and show the results in the next section.

6.2 Results

We consider the Helium-4 nucleus with Lattice size $L = 4$, and we set all interaction coupling to zero, except the leading order SU(4) contact interaction. In the following, coupling (c or c_t) refers to the interaction strength of the leading order SU(4) contact interaction. We compare the energy we get from eigenvector continuation to the Expectation energy, which we define by,

$$\text{Expectation energy} = \frac{\langle \psi_{init}|c_t|\cdots|c_t|H(c_t)|c_t|\cdots|c_t|\psi_{init}\rangle}{\langle \psi_{init}|c_t|\cdots|c_t|c_t|\cdots|c_t|\psi_{init}\rangle} \tag{6.7}$$

We perform eigenvector continuation calculation for a fixed $Lt = 40$ over several target coupling points. The results are listed in table 6.1. The error in EC is not listed here. We should note that as we perform higher order eigenvector continuation, the EC energy will be less than expectation energy, and closer to the actual $Lt \to \infty$ energy. This is because for $Lt \neq \infty$, the ground state eigenvectors we get and use as training point for eigenvector continuation are contaminated with excited states, and expected energy uses this contaminated ground state to gives us an energy which is different from the actual ground state. On the other hand, when eigenvector continuation chooses a linear combination of these states, it cancels out some of the excited states and reaches closer to the true ground state. However, in our practical calculations we are noise limited in how high order eigenvector continuation we can perform. For the calculation shown in table 6.1, 4^{th} order EC did not improve the result by much because of noise.

Table 6.2 shows the result when we fix the target coupling $c_t = -5.1\text{e}{-7}$ and vary Lt.

From the results of table 6.2, we can perform infinite-time extrapolation to get the eigenvector continuation energy at $Lt \to \infty$. Fitting an exponential curve, we get $Lt \to \infty$

c_t	Expectation Energy	Order 2 EC Energy	Order 3 EC Energy
$-4e-7$	-15.499 ± 0.012	-15.601	-15.666
$-5.1e-7$	-30.274 ± 0.020	-30.260	-30.213
$-6e-7$	-43.124 ± 0.015	-42.922	-42.925

Table 6.1: Eigenvector continuation calculation for He-4 with $Lt = 40$.

Lt	Expectation Energy	Order 2 EC Energy	Order 3 EC Energy
40	-30.274 ± 0.020	-30.260	-30.213
80	-45.588 ± 0.015	-45.357	-45.458
120	-51.237 ± 0.018	-50.852	-51.042
160	-53.213 ± 0.023	-52.866	-53.084

Table 6.2: Eigenvector continuation calculation for He-4 with $c_t = -5.1e-7$.

energy to be -54.34, whereas the expectation energy calculated for a large $Lt = 640$ is -54.269 ± 0.026.

Chapter 7

Conclusions and Outlook

In this work, we have introduced the method of eigenvector continuation, and showed how can be applied to various problems. It provides a great computational speed up, and this can be used to make emulators that provide a fast estimate of the result at any point in our parameter space. Apart from the computational advantage, eigenvector continuation can also be used to extrapolate our data to regions where standard perturbation theory does not work.

This work presented a first study of error convergence for eigenvector continuation. We have seen that eigenvector continuation converges faster than perturbation theory. This is because the series expansion of the wave function in perturbation theory exhibits an effect called differential folding, the interference among non-orthogonal terms at different orders. eigenvector continuation avoids this problem and does not diverge for any value of the control parameter. The error in eigenvector continuation approximation depends on the number and location of training points. While adding new training eigenvectors, it is the component of the new training eigenvector along the orthogonal direction of the subspace spanned by the current set of eigenvectors that contributes to accelerating the convergence of the method.

We have also shown that self-learning emulators can be used to select training data for any emulator that emulates the solution to a set of constraint equations. The self-learning

method is an active learning protocol that relies on a fast estimate of the emulator error and a greedy local optimization algorithm that becomes progressively more accurate as the emulator improves. The computational advantage of using self-learning can be as large as the speed-up factor of the emulator itself, which can be several order of magnitude or more. Using only 80 training points found by the self-learning algorithm, we find that we can emulate data at 10^6 points in a six-dimensional space.

There are several things that can be done in the near future. The error convergence for the anharmonic oscillator still has not been characterized completely. It has been hard to characterize error convergence when the singular point is very close to the origin, and goes even closer to origin as we increase our matrix dimensions. Perhaps with a bit of work, we can explain the weird convergence behaviour we have observed so far.

The connection between eigenvector continuation and Model Order Reduction methods needs to be explored in more details. While eigenvector continuation itself is definitely a special case of a reduced basis methods, we need to look into the Model Order Reduction method literature to see the connections between the two methods, and gain more insights into eigenvector continuation. While this has been done to some extent in [53], there is still additional work to be done. This may also allow us to improve our computations in some way.

The work on Floating-Block eigenvector continuation method is still ongoing. We have presented some preliminary data that shows that the method works when we extrapolate in the SU(4) contact interaction coupling. Now we are trying to implement the method and test it for the alpha particle (He-4 nucleus) with leading order SU(4) contact interaction, and one-pion exchange potential. With only SU(4) contact interactions, there is no sign problem, but once we introduce the one-pion exchange potential, strong sign oscillations dominate and

we cannot calculate our results directly any more. Instead, we now turn on the one-pion exchange potential as a perturbation, and this allows us to calculate ground state energies with small strength of one-pion exchange potential. The idea here is then to extrapolate from these small strength data points to the target coupling that we are interested in, using eigenvector continuation. If we are successful, then this technique would allow us to employ eigenvector continuation in all our lattice Effective Field Theory calculations, and we can compute the properties of heavier nuclei more efficiently.

Chapter 8

Supplemental Materials

8.1 Anharmonic Oscillator

8.1.1 Calculation of derivatives of eigenvector through perturbative expansion

We start with Hamiltonian where some subspace is completely diagonalized. Let $H(E)$ be the Hamiltonian matrix

$$H(E) = \begin{bmatrix} 0 & 0 & \cdots & \langle 1| H |A_1\rangle & \langle 1| H |A_2\rangle & \cdots \\ 0 & \lambda_2 & \cdots & \langle 2| H |A_1\rangle & \langle 2| H |A_2\rangle & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ \langle A_1| H |1\rangle & \langle A_1| H |2\rangle & \cdots & E \cdot \lambda_{A_1} & \langle A_1| H |A_2\rangle & \cdots \\ \langle A_2| H |1\rangle & \langle A_2| H |2\rangle & \cdots & \langle A_2| H |A_1\rangle & E \cdot \lambda_{A_2} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \end{bmatrix} \tag{8.1}$$

We have included a scaling factor E to the diagonal entries of the vectors not in the

original subspace. We will regard the diagonal matrix

$$D(E) = \begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & \cdots \\ 0 & \lambda_2 & \cdots & 0 & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ 0 & 0 & \cdots & E \cdot \lambda_{A_1} & 0 & \cdots \\ 0 & 0 & \cdots & 0 & E \cdot \lambda_{A_2} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \end{bmatrix} \tag{8.2}$$

as the unperturbed Hamiltonian matrix.

For convenience we have set the first diagonal entry of the unperturbed Hamiltonian to zero, $\lambda_1^{(0)} = 0$. We consider the limit $E \to +\infty$.

$$\begin{bmatrix} 0 & 0 & \cdots & \langle 1| H |A_1\rangle & \langle 1| H |A_2\rangle & \cdots \\ 0 & \lambda_2 & \cdots & \langle 2| H |A_1\rangle & \langle 2| H |A_2\rangle & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \\ \langle A_1| H |1\rangle & \langle A_1| H |2\rangle & \cdots & E \cdot \lambda_{A_1} & \langle A_1| H |A_2\rangle & \cdots \\ \langle A_2| H |1\rangle & \langle A_2| H |2\rangle & \cdots & \langle A_2| H |A_1\rangle & E \cdot \lambda_{A_2} & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots \end{bmatrix} \times$$

$$\begin{bmatrix} \frac{c_1^{(0)}}{E^0} + \frac{c_1^{(1)}}{E^1} + \frac{c_1^{(2)}}{E^2} + \cdots \\ \frac{c_2^{(0)}}{E^0} + \frac{c_2^{(1)}}{E^1} + \frac{c_2^{(2)}}{E^2} + \cdots \\ \vdots \\ \frac{c_{A_1}^{(0)}}{E^0} + \frac{c_{A_1}^{(1)}}{E^1} + \frac{c_{A_1}^{(2)}}{E^2} + \cdots \\ \frac{c_{A_2}^{(0)}}{E^0} + \frac{c_{A_2}^{(1)}}{E^1} + \frac{c_{A_2}^{(2)}}{E^2} + \cdots \\ \vdots \end{bmatrix} = \left(\frac{\lambda_1^{(1)}}{E} + \frac{\lambda_1^{(2)}}{E^2} + \cdots \right) \begin{bmatrix} \frac{c_1^{(0)}}{E^0} + \frac{c_1^{(1)}}{E^1} + \frac{c_1^{(2)}}{E^2} + \cdots \\ \frac{c_2^{(0)}}{E^0} + \frac{c_2^{(1)}}{E^1} + \frac{c_2^{(2)}}{E^2} + \cdots \\ \vdots \\ \frac{c_{A_1}^{(0)}}{E^0} + \frac{c_{A_1}^{(1)}}{E^1} + \frac{c_{A_1}^{(2)}}{E^2} + \cdots \\ \frac{c_{A_2}^{(0)}}{E^0} + \frac{c_{A_2}^{(1)}}{E^1} + \frac{c_{A_2}^{(2)}}{E^2} + \cdots \\ \vdots \end{bmatrix}$$

We start with $c_1^{(0)} = 1$, $c_j^{(0)} = 0$ for $j > 1$, and $c_{A_i}^{(0)} = 0$ for all i. Furthermore, since the normalization is arbitrary, we also take $c_1^{(n)} = 0$ for all $n > 1$. In row 1 at order n we have

$$\sum_i \langle 1| H |A_i\rangle c_{A_i}^{(n)} = \sum_{1 \le n' \le n} \lambda_1^{(n')} c_1^{(n-n')}. \tag{8.3}$$

Since $c_1^{(0)} = 1$ and $c_1^{(n)} = 0$ for all $n > 1$, we can rewrite this as

$$\lambda_1^{(n)} = \sum_i \langle 1| H |A_i\rangle c_{A_i}^{(n)} \tag{8.4}$$

In row $j > 1$ at order n we have

$$\lambda_j c_j^{(n)} + \sum_i \langle j| H |A_i\rangle c_{A_i}^{(n)} = \sum_{1 \le n' \le n} \lambda_1^{(n')} c_j^{(n-n')} \tag{8.5}$$

In row A_i at order n we have

$$\lambda_{A_i} c_{A_i}^{(n+1)} + \sum_j \langle A_i| H |j\rangle c_j^{(n)} + \sum_{k \ne i} \langle A_i| H |A_k\rangle c_{A_k}^{(n)} = \sum_{1 \le n' \le n} \lambda_1^{(n')} c_{A_i}^{(n-n')} \tag{8.6}$$

We solve these equations recursively. From Eq. (8.6) we have,

$$c_{A_i}^{(n)} = -\lambda_{A_i}^{-1} \sum_j \langle A_i| H |j\rangle c_j^{(n-1)} - \lambda_{A_i}^{-1} \sum_{k\neq i} \langle A_i| H |A_k\rangle c_{A_k}^{(n-1)} + \lambda_{A_i}^{-1} \sum_{1\leq n'\leq n-1} \lambda_1^{(n')} c_{A_i}^{(n-1-n')} \tag{8.7}$$

From Eq. (8.4), we get

$$\lambda_1^{(n)} = \sum_i \langle 1| H |A_i\rangle c_{A_i}^{(n)} \tag{8.8}$$

And from Eq. (8.5) we have for $j > 1$,

$$c_j^{(n)} = -\lambda_j^{-1} \sum_i \langle j| H |A_i\rangle c_{A_i}^{(n)} + \lambda_j^{-1} \sum_{1\leq n'\leq n} \lambda_1^{(n')} c_j^{(n-n')} \tag{8.9}$$

For applications where the initial subspace is very large, it is impractical to fully diagonalize the subspace. In that case we include the lowest energy subspace-projected eigenvectors $|j\rangle$ with the same quantum numbers as $|1\rangle$ and extrapolate to the limit where all subspace-projected eigenvectors $|j\rangle$ are included.

8.1.2 Simplified recursion relation

We let

$$L = -\sum_i \sum_j |A_i\rangle \lambda_{A_i}^{-1} \langle A_i| H |j\rangle \langle j| - \sum_i \sum_{k\neq i} |A_i\rangle \lambda_{A_i}^{-1} \langle A_i| H |A_k\rangle \langle A_k| , \tag{8.10}$$

$$S = -\sum_{j\neq 1} \sum_i |j\rangle \lambda_j^{-1} \langle j| H |A_i\rangle \langle A_i| , \tag{8.11}$$

$$D = \sum_{j \neq 1} |j\rangle \, \lambda_j^{-1} \, \langle j| + \sum_i |A_i\rangle \, \lambda_{A_i}^{-1} \, \langle A_i| \,, \tag{8.12}$$

$$\langle r_L| = \sum_i \langle 1| \, H \, |A_i\rangle \, \langle A_i| \,. \tag{8.13}$$

We also define

$$\left|\psi_S^{(n)}\right\rangle = \sum_j c_j^{(n)} \, |j\rangle \,, \tag{8.14}$$

$$\left|\psi_L^{(n)}\right\rangle = \sum_i c_{A_i}^{(n)} \, |A_i\rangle \,. \tag{8.15}$$

We can then write the recurrence relation as

$$\left|\psi_L^{(n)}\right\rangle = L \left|\psi_L^{(n-1)}\right\rangle + L \left|\psi_S^{(n-1)}\right\rangle + D \sum_{1 \leq n' \leq n-1} \lambda_1^{(n')} \left|\psi_L^{(n-1-n')}\right\rangle, \tag{8.16}$$

$$\lambda_1^{(n)} = \left\langle r_L \left|\psi_L^{(n)}\right\rangle \right., \tag{8.17}$$

$$\left|\psi_S^{(n)}\right\rangle = S \left|\psi_L^{(n-1)}\right\rangle + D \sum_{1 \leq n' \leq n} \lambda_1^{(n')} \left|\psi_S^{(n-n')}\right\rangle. \tag{8.18}$$

At first order we have

$$\left|\psi_L^{(1)}\right\rangle = L \left|\psi_L^{(0)}\right\rangle + L \left|\psi_S^{(0)}\right\rangle, \tag{8.19}$$

$$\lambda_1^{(1)} = \left\langle r_L \left| L \left|\psi_L^{(0)}\right\rangle \right. \right., \tag{8.20}$$

$$\left|\psi_S^{(1)}\right\rangle = S \left|\psi_L^{(0)}\right\rangle + \left\langle r_L \left| L \left|\psi_L^{(0)}\right\rangle \right. D \left|\psi_S^{(0)}\right\rangle. \tag{8.21}$$

At second order we have

$$\begin{aligned}\left|\psi_L^{(2)}\right\rangle =& L\left|\psi_L^{(1)}\right\rangle + L\left|\psi_S^{(1)}\right\rangle + \lambda_1^{(1)} D\left|\psi_L^{(1)}\right\rangle, \\ =& LL\left|\psi_L^{(0)}\right\rangle + (LL+LS)\left|\psi_L^{(0)}\right\rangle + LS\left|\psi_S^{(1)}\right\rangle \\ &+ \left\langle r_L\left|L\left|\psi_L^{(0)}\right\rangle\right.\left[LD\left|\psi_S^{(0)}\right\rangle + DL\left|\psi_L^{(0)}\right\rangle + DL\left|\psi_S^{(0)}\right\rangle\right], \end{aligned} \tag{8.22}$$

$$\begin{aligned}\lambda_1^{(2)} =& \left\langle r_L\left|L\left|\psi_L^{(2)}\right\rangle\right.\right.\left|\psi_L^{(2)}\right\rangle \\ =& \left\langle r_L\left|LLL\left|\psi_L^{(0)}\right\rangle\right.\right. + \left\langle r_L\left|(LLL+LLS)\left|\psi_L^{(0)}\right\rangle\right.\right. + \left\langle r_L\left|LLS\left|\psi_S^{(0)}\right\rangle\right.\right. \\ &+ \left\langle r_L\left|L\left|\psi_L^{(0)}\right\rangle\right.\right.\left[\left\langle r_L\left|LLD\left|\psi_S^{(0)}\right\rangle\right.\right. + \left\langle r_L\left|LDL\left|\psi_L^{(0)}\right\rangle\right.\right.\right] \\ &+ \left\langle r_L\left|L\left|\psi_L^{(0)}\right\rangle\right.\right.\left\langle r_L\left|LDL\left|\psi_S^{(0)}\right\rangle\right.\right., \end{aligned} \tag{8.23}$$

$$\left|\psi_S^{(2)}\right\rangle = S\left|\psi_L^{(1)}\right\rangle + \lambda_1^{(1)} D\left|\psi_S^{(1)}\right\rangle + \lambda_1^{(2)} D\left|\psi_S^{(0)}\right\rangle. \tag{8.24}$$

8.1.3 Use in Eigenvector Continuation

We now use the perturbative corrections at each order n as our basis vectors for eigenvector continuation. The norm matrix elements are

$$N_{n,n'} = \sum_j c_j^{(n)*} c_j^{(n')} + \sum_i c_{A_i}^{(n)*} c_{A_i}^{(n')}. \tag{8.25}$$

while the Hamiltonian matrix elements are

$$
\begin{aligned}
H_{n,n'}(E) = & \sum_{j\neq 1} c_j^{(n)*} \lambda_j c_j^{(n')} + E \sum_i c_{A_i}^{(n)*} \lambda_{A_i} c_{A_i}^{(n')} \\
& + \sum_j \sum_i c_j^{(n)*} \langle j| H |A_i\rangle c_{A_i}^{(n')} + \sum_i \sum_j c_{A_i}^{(n)*} \langle A_i| H |j\rangle c_j^{(n')} \\
& + \sum_i \sum_{k\neq i} c_{A_i}^{(n)*} \langle A_i| H |A_k\rangle c_{A_k}^{(n')}. \qquad (8.26)
\end{aligned}
$$

For many applications of interest, the number of basis vectors $|A_i\rangle$ will be extremely large, so large that standard vector operators are not possible and storage of the entries $c_{A_i}^{(n)}$ require more computer memory than available. In that case we cannot use the recursion relations directly. Instead, we need to completely unroll the recursion relations into their individual terms. The individual expressions can be evaluated by one of two options. The first option is exact calculation. This however becomes impractical rather quickly even with massively parallel computing. The more practical approach is to use stochastic sampling of the summation terms as suggested in the original stochastic error correction paper [62].

For our case of the anharmonic oscillator, we have a single parameter λ which we set to 0 after shifting the Hamiltonian by λI. The elements $\langle n|H|A_i\rangle$ and $\langle A_i|H|n\rangle$ are particularly simple as $n = 1$ only. We have only one c_1 as well, which is set to $c_1^{(0)} = 1$ and $c_1^{(n)} = 0$ for $n > 1$.

8.2 Natural Cubic Splines

In this section, we show the algorithm for numerically computing Natural Cubic Splines. This can also be found in any book on splines [63], and even on Wikipedia.

Our input is a set of data C containing $n+1$ points.

Our output is a set of splines, which is composed of n 5-tuples.

Algorithm:

1. Create new array a of size $n+1$ and for $i = 0, \ldots, n$ set $a_i = y_i$.

2. Create new arrays b and d each of size n.

3. Create new array h of size n and for $i = 0, \ldots, n-1$ set $h_i = x_{i+1} - x_i$.

4. Create new array α of size n and for $i = 1, \ldots, n-1$ set
   $\alpha_i = \dfrac{3}{h_i}(a_{i+1} - a_i) - \dfrac{3}{h_{i-1}}(a_i - a_{i-1})$.

5. Create new arrays c, l, μ, and z each of size $n+1$.

6. Set $l_0 = 1, \mu_0 = z_0 = 0$.

7. For $i = 1, \ldots, n-1$

   (a) Set $l_i = 2(x_{i+1} - x_{i-1}) - h_{i-1}\mu_{i-1}$.

   (b) Set $\mu_i = \dfrac{h_i}{l_i}$.

   (c) Set $z_i = \dfrac{\alpha_i - h_{i-1}z_{i-1}}{l_i}$.

8. Set $l_n = 1; z_n = c_n = 0$.

9. For $j = n-1, n-2, \ldots, 0$.

   (a) Set $c_j = z_j - \mu_j c_{j+1}$.

   (b) Set $b_j = \dfrac{a_{j+1} - a_j}{h_j} - \dfrac{h_j\left(c_{j+1} + 2c_j\right)}{3}$.

   (c) Set $d_j = \dfrac{c_{j+1} - c_j}{3h_j}$.

10. Create new set Splines and populate it with n splines S

11. For $i = 0, \ldots, n-1$

    (a) Set $S_{i,a} = a_i$.

    (b) Set $S_{i,b} = b_i$.

    (c) Set $S_{i,c} = c_i$.

    (d) Set $S_{i,d} = d_i$.

    (e) Set $S_{i,x} = x_i$.

12. The n output splines S are in the form:

$$S_j(x) = a_j + b_j\left(x - x_j\right) + c_j\left(x - x_j\right)^2 + d_j\left(x - x_j\right)^3$$

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] D. Frame, R. He, I. Ipsen, D. Lee, D. Lee, and E. Rrapaj. Eigenvector continuation with subspace learning. *Phys. Rev. Lett.*, 121(3):032501, 2018.

[2] D. K. Frame. *Ab Initio Simulations of Light Nuclear Systems Using Eigenvector Continuation and Auxiliary Field Monte Carlo*. PhD thesis, Michigan State University, 2019.

[3] S. König, A. Ekström, K. Hebeler, D. Lee, and A. Schwenk. Eigenvector Continuation as an Efficient and Accurate Emulator for Uncertainty Quantification. 2019.

[4] A. Ekström and G. Hagen. Global sensitivity analysis of bulk properties of an atomic nucleus. *Phys. Rev. Lett.*, 123(25):252501, 2019.

[5] P. Demol, T. Duguet, A. Ekström, M. Frosini, K. Hebeler, S. König, D. Lee, A. Schwenk, V. Somà, and A. Tichai. Improved many-body expansions from eigenvector continuation. *Phys. Rev.*, C101:041302(R), 2020.

[6] A. Ekström, G. R. Jansen, K. A. Wendt, G. Hagen, T. Papenbrock, B. D. Carlsson, C. Forssén, M. Hjorth-Jensen, P. Navrátil, and W. Nazarewicz. Accurate nuclear radii and binding energies from a chiral interaction. *Phys. Rev.*, C91(5):051301, 2015.

[7] S. Elhatisari et al. Nuclear binding near a quantum phase transition. *Phys. Rev. Lett.*, 117(13):132501, 2016.

[8] V. Lapoux, V. Somà, C. Barbieri, H. Hergert, J. D. Holt, and S. R. Stroberg. Radii and Binding Energies in Oxygen Isotopes: A Challenge for Nuclear Forces. *Phys. Rev. Lett.*, 117(5):052501, 2016.

[9] M. Piarulli et al. Light-nuclei spectra from chiral dynamics. *Phys. Rev. Lett.*, 120(5):052503, 2018.

[10] B.-N. Lu, N. Li, S. Elhatisari, D. Lee, E. Epelbaum, and U.-G. Meißner. Essential elements for nuclear binding. *Phys. Lett.*, B797:134863, 2019.

[11] S. Binder et al. Few-nucleon and many-nucleon systems with semilocal coordinate-space regularized chiral nucleon-nucleon forces. *Phys. Rev.*, C98(1):014002, 2018.

[12] V. Somà, P. Navrátil, F. Raimondi, C. Barbieri, and T. Duguet. Novel chiral Hamiltonian and observables in light and medium-mass nuclei. *Phys. Rev.*, C101(1):014318, 2020.

[13] S. Gandolfi, D. Lonardoni, A. Lovato, and M. Piarulli. Atomic nuclei from quantum Monte Carlo calculations with chiral EFT interactions. 2020.

[14] I. Tews, Z. Davoudi, A. Ekström, J. D. Holt, and J. E. Lynn. New Ideas in Constraining Nuclear Forces. 2020.

[15] J. J. Sakurai and J. Napolitano. *Modern Quantum Mechanics*. Cambridge University Press, 2021.

[16] W. D. Heiss and A. L. Sannino. Avoided level crossing and exceptional points. *Journal of Physics A: Mathematical and General*, 23(7):1167–1178, apr 1990.

[17] T. Kato. *Perturbation Theory for Linear Operators*. Springer, 1995.

[18] E. J. Weniger. Very accurate summation for the infinite coupling limit of the perturbation series expansions of anharmonic oscillators. *Phys. Lett. A*, 156:169–174, Jun, 1991.

[19] E. J. Weniger. The summation of the ordinary and renormalized perturbation series for the ground state energy of the quartic, sextic, and octic anharmonic oscillators using nonlinear sequence transformations. *J. Math. Phys.*, 34, 571, 1993.

[20] E. J. Weniger. A convergent renormalized strong coupling perturbation expansion for the ground state energy of the quartic, sextic, and octic anharmonic oscillator. *Ann. Phys.*, 246:133–165, Feb, 1996.

[21] G. Bozzolo and A. Plastino. Generalized anharmonic oscillator: A simple variational approach. *Phys. Rev. D*, 24, Dec, 1981.

[22] N. Bessis and G. Bessis. Open perturbation and the riccati equation: Algebraic determination of the quartic anharmonic oscillator energies and eigenfunctions. *J. Math. Phys.*, 38, Jul, 1997.

[23] F. M. Fernández, Q. Ma, and R. H. Tipping. Tight upper and lower bounds for energy eigenvalues of the schrödinger equation. *Phys. Rev. A*, 39, Feb, 1989.

[24] C. M. Bender and T. T. Wu. Anharmonic oscillator. *Phys. Rev.*, 184, 1231, Aug,1969.

[25] C. M. Bender and T. T. Wu. Large-order behavior of perturbation theory. *Phys. Rev. Lett.*, 27, 461, August 1971.

[26] C. M. Bender and T. T. Wu. Anharmonic oscillator. ii. a study of perturbation theory in large order. *Phys. Rev. D*, 7, 1620, March 1973.

[27] A. J. Leggett. *Modern Trends in the Theory of Condensed Matter. Proceedings of the XVIth Karpacz Winter School of Theoretical Physics, Karpacz, Poland, 1980*. Springer-Verlag, Berlin, 1980.

[28] P. Nozieres and S. Schmitt-Rink. Bose condensation in an attractive fermion gas: from weak to strong coupling superconductivity. *J. Low Temp. Phys.*, 59:195–211, 1985.

[29] K. M. O'Hara, S. L. Hemmer, M. E. Gehm, S. R. Granade, and J. E. Thomas. Observation of a strongly interacting degenerate fermi gas of atoms. *Science*, 298:2179–2182, 2002.

[30] S. Gupta, Z. Hadzibabic, M. W. Zwierlein, C. A. Stan, K. Dieckmann, C. H. Schunck, E. G. M. van Kempen, B. J. Verhaar, and W. Ketterle. Radio-frequency spectroscopy of ultracold fermions. *Science*, 300:1723–1726, 2003.

[31] C. A. Regal and D. S. Jin. Measurement of positive and negative scattering lengths in a fermi gas of atoms. *Phys. Rev. Lett.*, 90:230404, 2003.

[32] M. J. H. Ku, A. T. Sommer, L. W. Cheuk, and M. W. Zwierlein. Revealing the Superfluid Lambda Transition in the Universal Thermodynamics of a Unitary Fermi Gas. *Science*, 335:563–, February 2012.

[33] S. Bour, X. Li, D. Lee, U.-G. Meißner, and L. Mitas. Precision benchmark calculations for four particles at unitarity. *Phys. Rev.*, A83:063619, 2011.

[34] M. Kompaniets. Prediction of the higher-order terms based on Borel resummation with conformal mapping. *J. Phys. Conf. Ser.*, 762(1):012075, 2016.

[35] K. Van Houcke, F. Werner, and R. Rossi. High-precision numerical solution of the Fermi polaron problem and large-order behavior of its diagrammatic series. *Phys. Rev.*, 101(4):045134, Jan 2020.

[36] R. J. Furnstahl, A. J. Garcia, P. J. Millican, and X. Zhang. Efficient emulators for scattering using eigenvector continuation. *Phys. Lett. B*, 809:135719, 2020.

[37] D. Bai and Z. Ren. Generalizing the calculable R-matrix theory and eigenvector continuation to the incoming wave boundary condition. *Phys. Rev. C*, 103(1):014612, 2021.

[38] S. Wesolowski, I. Svensson, A. Ekström, C. Forssén, R. J. Furnstahl, J. A. Melendez, and D. R. Phillips. Fast & rigorous constraints on chiral three-nucleon forces from few-body observables. 4 2021.

[39] A. Sarkar and D. Lee. Convergence of eigenvector continuation. *Phys. Rev. Lett.*, 126:032501, Jan 2021.

[40] S. Yoshida and N. Shimizu. A new workflow of shell-model calculations with the emulator and preprocessing using eigenvector continuation, and shell-model code ShellModel.jl. 5 2021.

[41] J. A. Melendez, C. Drischler, A. J. Garcia, R. J. Furnstahl, and X. Zhang. Fast & accurate emulation of two-body scattering observables without wave functions. 6 2021.

[42] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová. Machine learning and the physical sciences. *Rev. Mod. Phys.*, 91:045002, Dec 2019.

[43] J. J. Thiagarajan, B. Venkatesh, R. Anirudh, P.-T. Bremer, J. Gaffney, G. Anderson, and B. Spears. Designing accurate emulators for scientific processes using calibration-driven deep models. *Nature Communications*, 11:5622, November 2020.

[44] M. F. Kasim, D. Watson-Parris, L. Deaconu, S. Oliver, P. Hatfield, D. H. Froula, G. Gregori, M. Jarvis, S. Khatiwala, J. Korenaga, J. Topp-Mugglestone, E. Viezzer, and S. M. Vinko. Building high accuracy emulators for scientific simulations with deep neural architecture search. *arXiv e-prints*, page arXiv:2001.08055, January 2020.

[45] P. Bedaque et al. Report from the A.I. For Nuclear Physics Workshop. *Eur. Phys. J. A*, 57(3):100, 2021.

[46] P. Baldi, S. P., and Whiteson. Searching for exotic particles in high-energy physics with deep learning. *Nat. Commun.*, July 2014.

[47] E. A, R. A, R. B, K. V, D. M, C. K, C. C, C. G, T. S, and D. J. A guide to deep learning in healthcare. *Nat Med.*, Jan 2019.

[48] B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

[49] D. A. Cohn, Z. Ghahramani, and M. I. Jordan. Active learning with statistical models. *Journal of artificial intelligence research*, 4:129–145, 1996.

[50] D. Cohn, L. Atlas, and R. Ladner. Improving generalization with active learning. *Machine learning*, 15(2):201–221, 1994.

[51] D. J. C. MacKay. Information-based objective functions for active data selection. *MIT Press Direct*, 4, 1992.

[52] J. H. Ahlberg, E. N. Nilson, and J. H. Walsh. Theory of splines and their applications. *Academic Press*, 1967.

[53] E. Bonilla, P. Giuliani, K. Godbey, and D. Lee. Training and Projecting: A Reduced Basis Method Emulator for Many-Body Physics. 3 2022.

[54] J. A. Melendez, C. Drischler, R. J. Furnstahl, A. J. Garcia, and X. Zhang. Model reduction methods for nuclear emulators. 3 2022.

[55] A. Quarteroni, A. Manzoni, and F. Negri. *Reduced Basis Methods for Partial Differential Equations*. Springer, 2016.

[56] A. Quarteroni, G. Rozza, and A. Manzoni. Certified reduced basis approximation for parametrized partial differential equations and applications. *J.Math.Industry*, 1, Jun 2011.

[57] S. E. Field, C. R. Galley, F. Herrmann, J. S. Hesthaven, E. Ochsner, and M. Tiglio. Reduced basis catalogs for gravitational wave templates. *Phys. Rev. Lett.*, 106:221102, Jun 2011.

[58] S. Elhatisari, E. Epelbaum, H. Krebs, T. A. Lähde, D. Lee, N. Li, B.-n. Lu, U.-G. Meißner, and G. Rupak. Ab initio Calculations of the Isotopic Dependence of Nuclear Clustering. *Phys. Rev. Lett.*, 119(22):222505, 2017.

[59] M. Freer, H. Horiuchi, Y. Kanada-En'yo, D. Lee, and U.-G. Meißner. Microscopic Clustering in Light Nuclei. *Rev. Mod. Phys.*, 90(3):035004, 2018.

[60] M. Pincus. Letter to the editor—a monte carlo method for the approximate solution of certain types of constrained optimization problems. *Operations Research*, 18(6):1225–1228, 1970.

[61] A. Quarteroni, A. Manzoni, and F. Negri. *Nuclear Lattice Effective Field Theory - An Introduction*. Springer, 2019.

[62] D. Lee, N. Salwen, and M. Windoloski. Introduction to stochastic error correction methods. *Phy. Lett. B*, 502:329–337, 2001.

[63] J. H. Ahlberg, E. N. Nilson, and J. L. Walsh. *The Theory of Splines and Their Applications*. ACADEMIC PRESS, 1967.