Physics-Informed Neural Networks for Gravity Field Modeling

by

J. R. Martin

B.S. Physics & Astronomy, University of North Carolina at Chapel Hill, 2018

B.A. Music, University of North Carolina at Chapel Hill, 2018

M.S. Aerospace Engineering Sciences, University of Colorado Boulder, 2021

A thesis submitted to the

Faculty of the Graduate School of the

University of Colorado in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

Department of Aerospace Engineering Sciences

2023

Committee Members:

Hanspeter Schaub, Chair

Jay McMahon

Daniel Scheeres

Zachary Sunberg

Roberto Furfaro

Martin, J. R. (Ph.D., Aerospace Engineering Sciences)

Physics-Informed Neural Networks for Gravity Field Modeling

Thesis directed by Prof. Hanspeter Schaub

Gravity is among the most ubiquitous forces within astrodynamics. The motion of every planet, asteroid, and spacecraft is intrinsically influenced by the gravitational forces of objects both near and far. Despite this, no universal model of this force exists. Rather, dynamicists must choose between many different gravity models that each carry their own unique advantages and drawbacks. For example, some models are fast to compute but lack analytic rigor; some achieve high accuracy but come with computational penalties, and others still are built with intrinsic assumptions or have limited operational validity. To combat these challenges, this thesis proposes the Physics-Informed Neural Network gravity model, or PINN-GM, which shifts attention away from analytic approaches and towards data-driven models. Specifically, the PINN-GM leverages recent advances in the field of Scientific Machine Learning to blend the power of neural networks with dynamical systems theory to produce high-fidelity models of complex dynamical systems without sacrificing analytic rigor. Through multiple iterations of development, the PINN-GM now offers high-accuracy, fast execution times, data efficiency, global validity, and exact differentiability. Taken together, these attributes make the PINN-GM well-suited to assist astrodynamicists in a variety of applications including reinforcement learning, periodic orbit discovery, and orbit determination.

Dedication

To my teachers.

Acknowledgements

They say it takes a village...

No matter what I say, there are simply too many people to thank, and one page will not do it justice. Thanks to Dr. Schaub for giving me the opportunity to pivot into a new field. Thanks to my committee for the many conversations, both professional and personal on my journey through academia. Thanks to my brother who taught me how to climb (have we made it to the top yet?). Thanks to all the AVS folk, new and old, for the memories, advice, and donuts. Thanks to the Burroughs and Martin families for celebrating the highs and supporting the lows. Minor thanks to my cat Callie, whose incecent meows encourage me to go to the lab every morning. Most importantly, thanks to Melissa — for everything.

Contents

Chapter

Tables

Table

Figures

Figure

Chapter 1

Introduction

Astrodynamics is the study of objects' motion through space. Be it for spacecraft or celestial bodies, the goal is to better understand the underlying forces acting on these systems and characterize or leverage the resulting motion. Despite the simplicity of the overarching theme, astrodynamics is full of a diverse array of academic topics, spanning trajectory optimization, formation flying, state estimation, control theory, attitude dynamics, and even remote sensing. Amidst the wide range of problems, one thing ties them all together: the force of gravity.

Consider the fact that nearly all problems in astrodynamics begin with equations of motion of the form:

$$\ddot{\boldsymbol{r}} = -\nabla U(\boldsymbol{r}) + \boldsymbol{a}_{\mathrm{d}} \tag{1.1}$$

where U is a model of the gravitational potential, and $\boldsymbol{a}_{\mathrm{d}}$ are the remaining accelerations. The presentation of this differential equation alone highlights the significance of the gravitational force. In nearly all cases, gravity serves as the dominant force and the remaining perturbations are secondary. Yet despite the critical role that gravity plays in all astrodynamics problems, no universal model of the force exists. Instead, dynamicists are left to choose between a variety of analytic or numerical approximations, referred to as gravity models, each with their own corresponding advantages and drawbacks.

The simplest gravity model assumes that all objects can be represented as perfect point masses whose gravitational potential takes the form

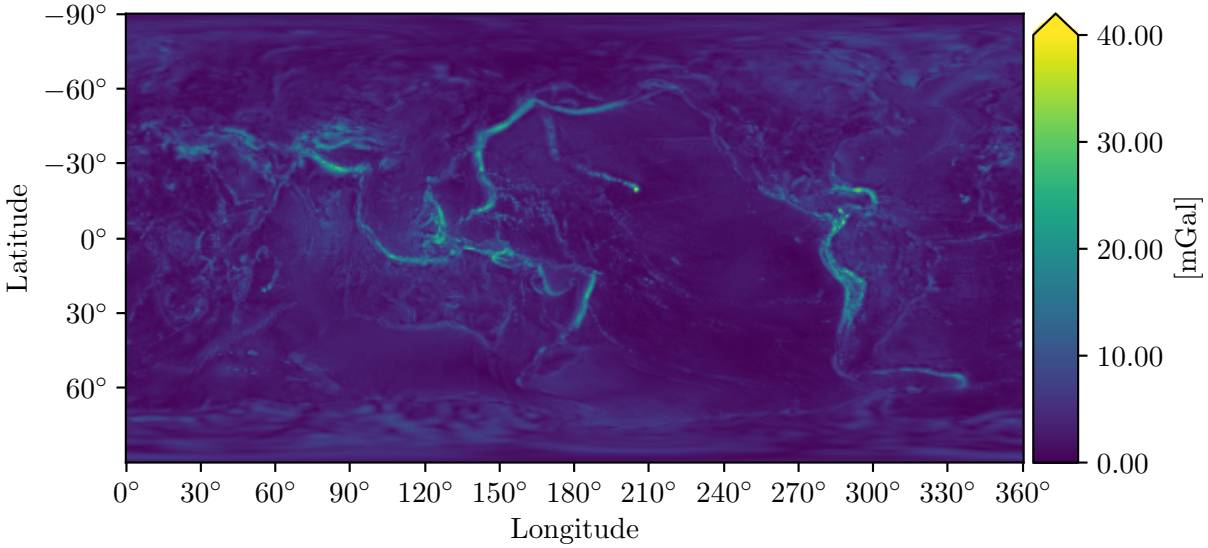
$$U(\boldsymbol{r}) = -\frac{\mu}{r} \tag{1.2}$$

where μ is the gravitational parameter of the body and r is the distance from the body. While this choice may be sufficient for first-order analyses and proof-of-concept work, it becomes considerably less reliable when transitioning into real mission scenarios. In these settings, higher-fidelity models are required, and dynamicists are forced to choose between a plethora of different models, including but not limited to spherical harmonics (2), ellipsoidal harmonics (3), mascons (4), polyhedral models (5), extreme learning machines (6), GeodesyNets (7), and so on.

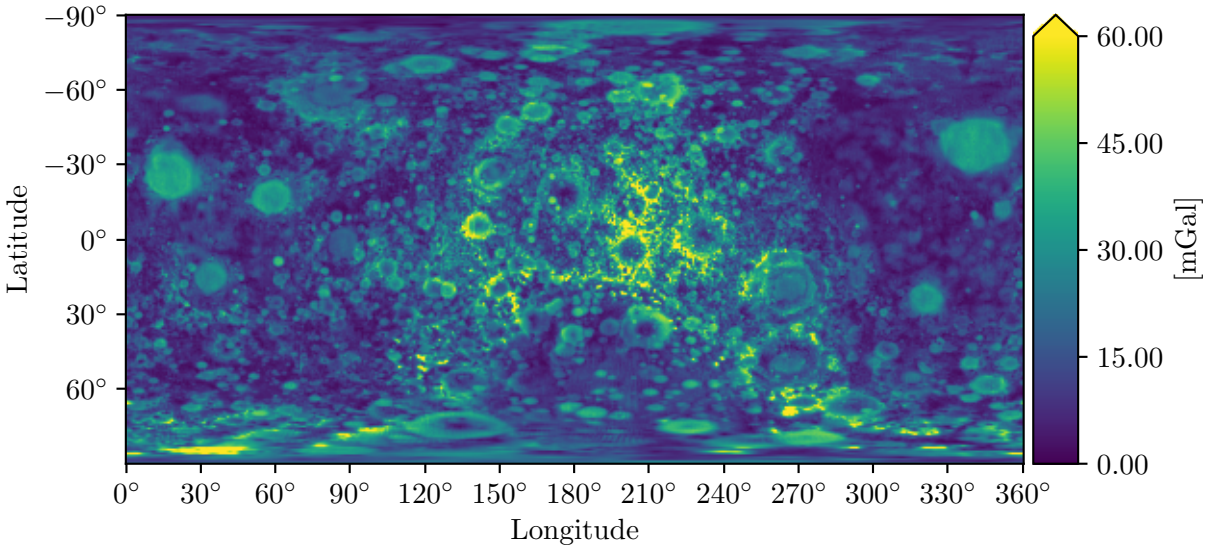
Each of these models have tradeoffs, but the one thing they all recognize is that gravity fields are far more complex than a point mass approximation. Every celestial body comes with asymmetries and heterogeneous densities, or have surfaces peppered with craters, mountains, mineral deposits, and other geologic features that each contribute their own unique perturbation to the total gravity field. One need only look at the woefully textured surfaces of the Earth and Moon shown in Figure 1.1 as evidence. To treat these bodies as perfect spheres, or even oblate ellipsoids, is a gross over-simplification of the true system. If dynamicists and spacecraft neglect these higher-order gravitational features, the consequences can range from the steady drift away from reference trajectories, to missing entry corridors, or in worst cases, unexpectedly transitioning orbiters into intercepting probes. To avoid these scenarios, it is imperative that dynamicists have efficient and accurate models to represent these gravitational perturbations to high-accuracy.

There exists a long history and rich body of literature dedicated to the construction of high-fidelity gravity models. The majority of this literature explores ways to represent gravity fields analytically, leveraging clever mathematical tricks and basis functions to guarantee convergence. While these models offer compelling physical interpretations and desireable mathematical properties, many come with operational limitations, assumptions, or computational overhead. While these analytic models have provided meaningful advances to the gravity modeling community, their lingering drawbacks highlight a fundamental question facing the dynamics community: Are there ways to construct universal models of complex dynamical forces free of these shortcomings?

To answer this question, one must first acknowledge that dynamics models are notoriously difficult to construct. Ideally, these models should be accurate, compact, computationally efficient,



(a) Earth Perturbations



(b) Moon Perturbations

Figure 1.1: Gravitational Perturbations for Earth and Moon

and physically-compliant; however, these objectives are often in natural competition with one another. In many cases, high-accuracy models come with many parameters, large memory footprints, and high computational cost. Compact models come with less overhead, but typically lose accuracy or cut corners in their analytic rigor. It remains an open question if there even exist models for which these compromises do not need to be made. For the case of gravity field modeling, there has yet to be a model that is flexible enough to represent the most dominant perturbations to high-accuracy while maintaining small memory footprints and fast execution times. If such a model did exist, the applications would be far reaching — expanding on-board capabilities for spacecraft guidance, navigation, and control, improving ground-based simulation fidelity for trajectory design and mission planning, as well as unexplored improvements for planetary scientists and geophysicists who use these models for other purposes.

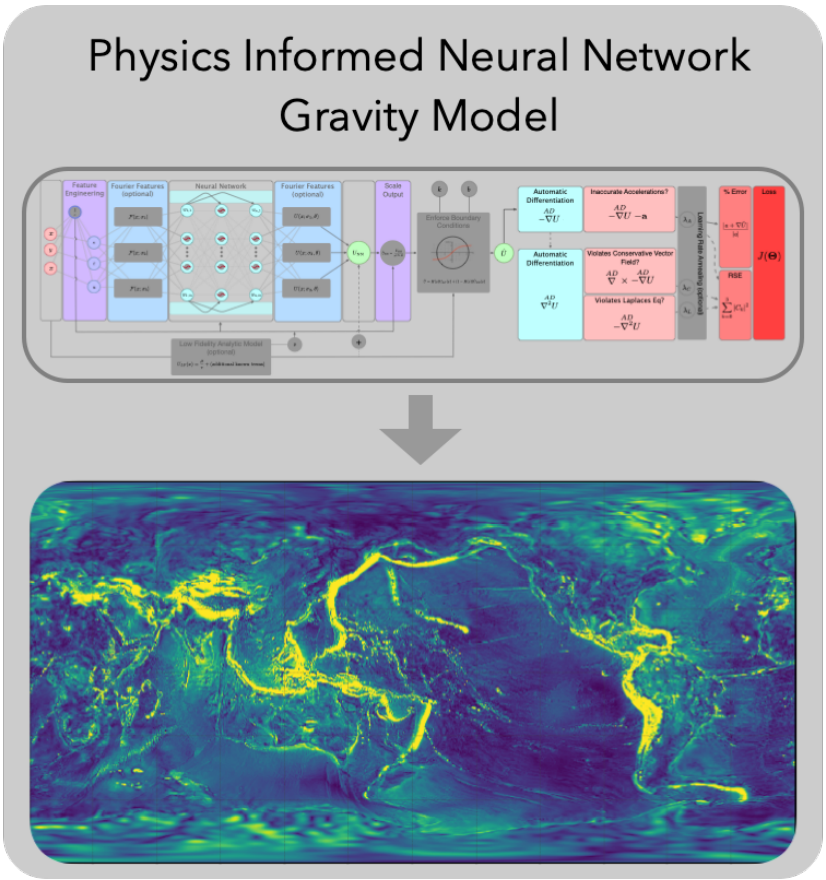


Figure 1.2: Using Scientific Machine Learning models to solve the gravity modeling problem.

This thesis aims to take a step towards the construction of a universal, high-fidelity gravity model. By shifting the focus away from purely analytic approaches, this work instead introduces a class of gravity model that leverages novel techniques within the emerging field of Scientific Machine Learning, or SciML. SciML is subset of artificial intelligence that seeks to construct high-fidelity models of complex dynamical systems in a data-driven manner (8). There exists a wide variety of candidate tools and function approximators in this branch machine learning, but one of the most popular is the Physics-Informed Neural Network (PINN).

PINNs are neural networks specifically designed to learn solutions to differential equations (9). While this can technically be accomplished with traditional neural networks — sampling values of the dynamical system at various positions and points in time and regressing a neural network to that data — PINNs provide a more satisfying and powerful framework. Specifically, PINNs introduce an augmented loss function which simultaneously penalizes mis-modeling of the training data while also including additional terms which penalize the model when it violates differential equations governing the true system. This simple change yields a surprising amount of utility, granting dynamicists a powerful and flexible way to represent complex dynamical phenomena while ensuring that the learned solution is intrinsically complaint with the underlying physics. Not only does this help quell concerns about the learned solution validity, but these changes also produce sizable improvements in the model's accuracy, generalizability, and data efficiency.

This thesis explores how Physics-Informed Neural Networks can be leveraged to solve the gravity modeling problem. Rather than prescribing analytic basis functions which come with their own unique challenges, the following chapters demonstrate how PINNs can learn more compact representations of the gravitational potential while maintaining the desirable physical properties of their analytic predecessors. After multiple iterations of development, the PINN-GM now produces gravity models that offer high-accuracy, fast execution times, data efficiency, global validity, and exact differentiability. Taken together, these attributes make the PINN-GM well-suited to assist astrodynamicists in a variety of applications including reinforcement learning, periodic orbit discovery, and orbit determination.

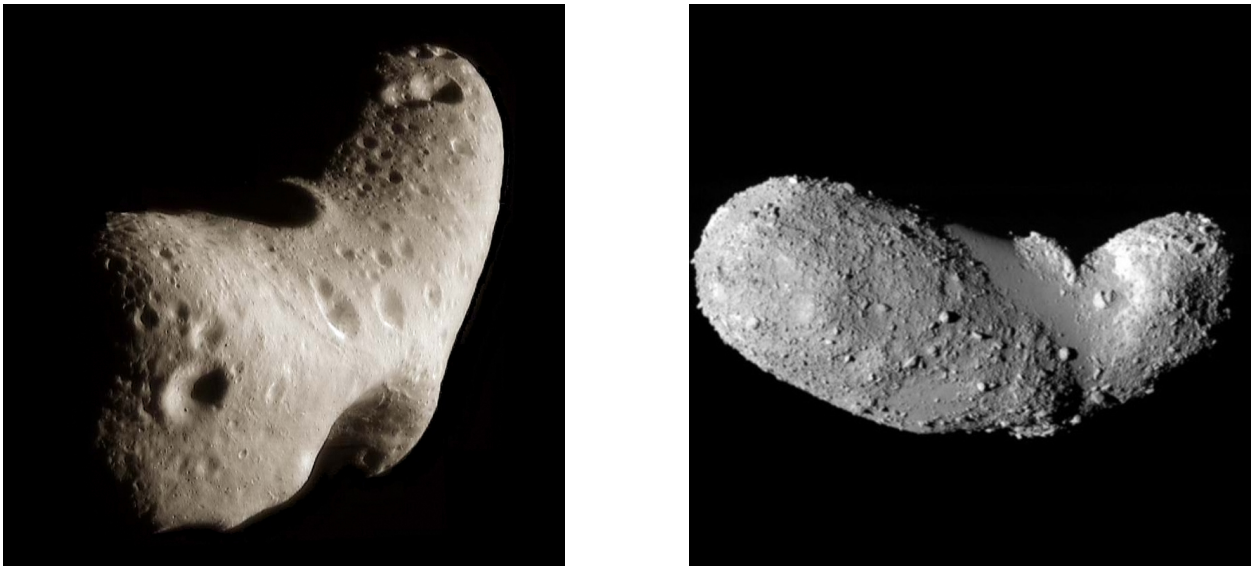
1.1 Significance

As discussed, gravity field modeling is among the most fundamental problems in astrodynamics. In nearly all space-based dynamical systems, gravity plays a key role in the guidance, navigation, and control of the object in question. Consider the most common problems in astrodynamics: the two body problem, three body problem, formation flying, rendezvous, proximity operations and docking, trajectory design, etc. For each of these topics, the gravity model can entirely change the nature of the design space and solutions discovered. Despite this, dynamicists often rely on the point mass gravity model or low-fidelity alternatives, as the transition to a higher-fidelity model risks the problem reaching an level of untenable complexity.

The use of low-fidelity gravity models, however, is becoming unsustainable. As near-Earth, cis-lunar, and deep space missions continue to increase in complexity, there grows a simultaneous need for more powerful dynamics models to assist spacecraft in achieving their goals. The recent decades spent exploring asteroids and comets, or small-bodies, provides a compelling example of this phenomenon. For the earliest missions in the 1980s like Giotto (10) and Galileo (11), the goal was simply to flyby the small-body of study (Halley's comet and the asteroid Gaspra respectively). These missions did not require extremely precise gravity models, only models accurate enough to get the spacecraft in the vicinity of the object to capture images before continuing on into deep space. Later, these missions began requiring greater precision, as designers sought to place spacecraft in orbit around these bodies (NEAR Shoemaker, 1996 (12)) or to conduct multiple flybys (Deep Space 1, 1998 (13)). Once it was demonstrated that spacecraft could enter orbit, priority was then shifted to collecting material (Stardust, 1999 (14)), purposefully impacting the object (Deep Impact, 2005 (15)), even landing on the body's surface (Rosetta, 2004 (16); Hayabusa, 2003 (17)). These increasingly ambitious goals continue to present day today with missions like Hayabusa2 (18) and OSIRIS-REx (19) attempting sample return for the asteroids Ryugu and Bennu respectively.

It is not difficult to extrapolate these trends and recognize that missions will continue to increase in complexity for the foreseeable future, and dynamicists will require increasingly precise

dynamics models to accommodate these ambitions. Of these forces, accurately capturing the effects of the asteroid's gravity field is particularly important. These bodies can have exotic geometries and density profiles (see Figure 1.3), which, when left unaccounted for, can place spacecraft on trajectories that may collide with the body. While avoiding catastrophic failure is the highest priority, high-fidelity gravity models can also be valuable for precision touchdowns. Consider the recent OSIRIS-REx mission whose flagship Touch-And-Go phase had originally been designed to manuever the spacecraft into 25 meter landing zone. After discovering that the asteroid was covered in boulders, the largest landing site available extended only a mere eight meters (20). In these moments, precise knowledge of the gravity field is essential to ensure mission safety during touchdown.



(a) 433-Eros (21)

(b) 25413-Itokawa (22)

Figure 1.3: Asteroids often have complex geometries leading to irregular gravity fields.

Beyond gravity models' application to small-body missions, there are also other academic communities that leverage high-fidelity representations of gravity fields. Specifically, gravity modeling has played a key role for problems in geophysics, ocean and climate science, and planetary science. Gravity models are commonly used to better understand water resources across the globe (23),

variations in sea level (24), and glacial melt (25). These studies play a critical role in predicting droughts (26) and floods (27). Similarly, gravity field models are used to better understand the interior of the Earth (28) and the motion of the tectonic plates (29). The latter is critical for applications in GNSS, which in-turn help determine elevations, reference frames, and coordinate systems used by other disciplines (30).

This work proposes a novel gravity model which seeks to provide utility to each of these fields; however the lessons learned through the development of this model extend beyond the space science community. Specifically, this work fundamentally explores how machine learning can be used to construct high-fidelity dynamics models; investigating how to systemically discover better basis functions and coordinate descriptions of complex dynamical phenomena. In principle, the lessons gathered here can benefit communities which seek to explore data-driven alternatives to otherwise cumbersome analytic models. The focus on gravity modeling offers a productive test bench for the scientific machine learning community at large, as these systems are often time-invariant and have well-studied pre-existing models. From this, SciML practitioners can use this problem to benchmark new machine learning techniques without introducing additional overhead of more complex dynamical systems.

1.2 History of Gravity Models

The search for high-fidelity gravity models predates the age of spacecraft. While the fundamentals of gravity were first characterized by Newton in the late 17th century, it was only in the early 1900s that scientists began constructing higher-fidelity models. Rapp provides a detailed history of these developments for Earth's gravity models in his review paper (31) which is summarized here for convenience. Spanning fromm 1901 to 1958, scientists first recognized that the Earth's gravity field deviates from that of a point mass approximation. To account for this, they first crafted gravity models that were the superposition of gravity variations in latitude (zonal perturbations), and later extended these models to include variations in longitude — the precursor to the spherical harmonic gravity model used today. To estimate the parameters of these models,

geodesists averaged ground-based gravity measurements in bins of fixed degree (30x30, 10x10, 5x5, and 1x1) and regressed the corresponding coefficients.

With the launch of Sputnik and other early satellites, geodesists like Kaula made first use of satellite data to estimate select coefficient in the 1960s. Likewise institutions like the Smithsonian Astrophysical Observatory, the Applied Physics Laboratory, and the Naval Weapons Laboratory computed their own gravity models from satellite data referred to as SE, APL, and NWL models respectively. Through the 1960s and into the 1970s, additional techniques were proposed to blend terrestrial and satellite data achieving higher degree models with greater precision. In the 1970s, Goddard Space Flight Center introduced their own gravity model GEM, which began with GEM-I in 1972 (32) but continued to be updated until GEM-10C which reached degree 180 in 1978. In the 1980s, the introduction of satellite laser tracking provided higher quality and greater quantity of satellite data to update these models. Missions like Lageos and TOPEX/POSEIDON data were used with the later generations of GEM to form models like GEM-L2 (33) and GEM-T1 (34) respectively. Similarly, researchers at the Center for Space Research at the University of Texas at Austin began developing their TEG models which utilized tracking data from multiple satellites. In the 1990s, GSFC and UT Austin's Center for Space Research joined forces and produced the JGM models 1-3 (35; 36). Most recently, the Earth Gravity Models (EGM) from the National Geospatial-Intelligence Agency provide some of the highest-fidelity models. Among the most well-known is EGM96 (37) which reached a spherical harmonic solution spanning degree 360. In 2008, NGA introduce the EGM2008 model which reached degree 2190 and is currently the highest fidelity spherical harmonic model for Earth publicly available (38).

The study of the Moon's gravity field followed a similar trend, albeit at a slower rate due to limited data. Among the first gravity models of the lunar field came as a result of the Apollo program. Studying the motion of the spacecraft which orbited the moon, researchers realized that the spacecraft orbital motion was perturbed over certain region of the lunar surface. These perturbations were later characterized as mass concentrations in the lunar surface, dubbed mascons, which could effectively be modeled as the superposition of multiple point mass elements embedded

on, or just below, the surface of the Moon (39). Analyses of this mascons continued through the late 20th century with missions like the Clementine mission (40), Lunar Reconnaissance Orbiter (41), and the Gravity Recovery and Interior Laboratory (GRAIL) (42) continuing to take measurements of the Moon's topology and gravity field to help construct both higher fidelity mascon models as well as spherical harmonic models.

Beyond the study of large planetary bodies and moons, small-body exploration has also fueled considerable work for gravity field modeling. As discussed, small-body gravity fields can be notoriously difficult to represent, particularly given their irregular geometry and unknown density profiles. Originally researchers continued to model these fields with spherical harmonics; however, as noted by Brillouin in 1933, the spherical harmonic model begins to diverge when the spacecraft enter the sphere which bounds all mass elements (43). This divergence make the spherical harmonic model particularly risky for proximity or landing operations.

In 1997, Werner and Scheeres introduced the polyhedral gravity model which offers a non-diverging solution to represent the gravity fields of these bodies (5). If a polyhedron shape model of the small-body in question exists, then there exist a way to compute the potential of that body analytically under the assumption of constant density. Such model offers a solution that remains stable down to the surface, making it a popular and favorable choice for many small body missions. Notably, this model does come with relatively high computational cost, and the constant density assumption has been shown to be invalid in some cases as demonstrated for the asteroid Bennu from recent OSIRIS-REx data (44).

Recently dynamicists have begun exploring alternative ways to represent the gravity fields of these exotic bodies, relying less on analytic approximations of the system, and instead on more data-driven models. These efforts have included the use of Gaussian processes (45), extreme learning machines (6), and neural networks (46). These models each offer computational advantages over some of their analytic counterparts, but they also lack the analytic niceties of past approaches. To date, none of these models demonstrate that they satisfy important differential properties, and relatively little work has gone into carefully validating these models. In many cases, the learned

solution is only tested in local regions or within the bounds of the training data. These models can also come with large memory footprints using tens or hundreds of thousands of parameters, and many require large quantities of training data to regress which can be difficult to acquire in practice.

1.3 Gravity Missions

Multiple missions have been flown with the goal of refining gravity field models for the Earth and Moon. For Earth, the first of these missions was the Challenging Minisatellite Payload (CHAMP), launched in 2000 by GFZ Potsdam. CHAMP was intended to conduct a variety of atmospheric and ionospheric research in addition to better characterize Earth's gravity and magnetic fields over the span of five years (47). Following CHAMP, was the Gravity Recovery and Climate Experiment (GRACE) launched in 2002 which leveraged two twin satellites and their relative distance to produce more precise measurements of the gravity field (48). Using this distance the mission was able to recover detailed measurements of water distribution (49), glacial ice melt (25), and contribute to state of the art knowledge on climate change (23). Following GRACE was the Gravity Field and Steady-State Ocean Circulation Explorer (GOCE) which used a highly sensitive gradiometer to estimate temporal variations the gravity field (50). After GOCE and the end of GRACE, GRACE Follow On (GRACE-FO) was launched which continued the efforts of the original GRACE mission to acquire time varying gravity field estimates (51). Between these missions, the Gravity Recovery and Interior Laboratory (GRAIL) was also launched in 2011 which sought to measure the gravity field of the Moon using a similar mission concept to GRACE (52).

1.4 Goal of Thesis

This thesis presents the first comprehensive document that discusses how to design, train, and validate physics-informed neural network gravity models (PINN-GM). Explicitly, this thesis answers questions including: How does the design of the PINN-GM impact its performance? How can input features be constructed maximize the parametric capacity of the model while avoiding

numerical instability? Are there advantages to incorporating preexisting models into the system, and if so, how can this be accomplished? What role does the quality and distribution of training data play in the model performance? These questions, among others, are investigated through multiple generations of development outlined in the coming chapters.

Moreover, this thesis also provides a number of new ways to characterize gravity model performance. Rather than randomly sampling test points and computing the corresponding acceleration error, this thesis introduces additional measures to provide a more complete depiction of model performance. These measures include evaluating total integration error during trajectory propagation, testing model accuracy across a range of altitudes within and beyond the training data, measuring model compactness as a function accuracy versus total parameters, and testing the models with varying data distributions and quality. These metrics become useful in not only evaluating the PINN-GM performance with respect to past models, but also provide a framework for guiding the development of future PINN-GM generations.

Through these investigations, the PINN-GM is shown to be the first data-driven model that learns powerful new basis functions of complex gravitational systems while enforcing differential constraints on the learned solution. These constraints provide the PINN-GM greater modeling accuracy and data-efficiency than their analytic and data-driven predecessors while maintaining small memory footprints and fast runtimes. Taken together, these attributes grant the PINN-GM wide utility for the astrodynamics community at large as shown through three case study applications. The first of these applications demonstrates how the PINN-GM's high-accuracy and fast evaluation speeds can improve the quality of autonomous spacecraft agents trained through reinforcement learning. The second application uses the differentiability of PINN-GM to discover candidate periodic orbits around complex gravitational bodies in arbitrary element sets. The third and final application explores how the PINN-GMs can be directly integrated into orbit determination pipelines and filters to estimate complex gravity fields in both an online and offline fashion in less time than current approaches.

In summary, the physics-informed neural network gravity model is a novel way to construct

high-fidelity models of the gravity fields for both large and small celestial bodies in a manner that is free of many pitfalls of former approaches. This thesis provides an overview of the generational improvements of this model and highlights candidate applications within the broader astrodynamics community.

Chapter 2

Review of the Gravity Modeling Problem

2.1 Overview

There exist two families of gravity models: analytic and numerical. Analytic models take a variety of forms, but at their core share the fact that they are derived from first principles, offering solutions to fundamental differential equations like Laplace's equation. In contrast, numerical models are constructed in a data-driven manner — regressing functions capable of interpolating between measured data. Each family of model has its own corresponding advantages and drawbacks, and the choice of which model to use is often dictated by the application. The following chapter details the available gravity models and their respective pros and cons.

2.2 Analytic Models

2.2.1 Spherical Harmonics

Early in the 1900s, it was proposed that spherical harmonic basis functions could be superimposed to produce a high-fidelity estimate of the gravitational potential (43).

$$U(r) = \frac{\mu}{r} \sum_{l=0}^{l} \sum_{m=0}^{l} \left(\frac{R}{r}\right)^{l} P_{l,m}\left(\sin\phi\right) \left[C_{l,m}\cos(m\lambda) + S_{l,m}\sin(m\lambda)\right] \tag{2.1}$$

Equation (2.1) is referred to as the spherical harmonic gravity model where r is the radius to the field point, μ is the gravitational parameter of the body, R is the circumscribing radius of the body, l is the degree of the model, m is the order of the model, $C_{l,m}$ and $S_{l,m}$ are the Stokes coefficients, λ is the longitude, ϕ is the latitude, and $P_{l,m}$ are the associated Legendre polynomials (2).

The spherical harmonic gravity model is the primary model of choice to represent the gravity fields of large planetary bodies like the Earth (53), Moon (54), and Mars (55). One of the most compelling advantages of the spherical harmonic gravity model is how it can compactly capture the large gravitational perturbation of planetary oblateness. Because the Earth and these other large bodies rotate about their axes, a centrifugal acceleration is produced which flattens the body from a sphere into an oblate ellipsoid (56). This flattening or excess of mass near the equator is referred to as planetary oblateness, and its presence has sizable effects on spacecraft trajectories. It is therefore important that this perturbation is accurately captured in a gravity model. The spherical harmonic gravity model makes this simple, needing only one harmonic in the expansion to capture this oblateness ($C_{2,0}$, or its alternative form $J_2 = -C_{2,0}$).

While spherical harmonics are effective at representing planetary oblateness and other global scale perturbations, they struggle to model the remaining and more discrete gravitational perturbations like mountain ranges, tectonic plate boundaries, and craters. Discontinuities are notoriously difficult to represent using periodic basis functions, often requiring hundreds of thousands of harmonics being superimposed to overcome the 3D equivalent of Gibbs phenomena (57). Not only must these harmonics be superimposed, they must also be regressed, and the high-frequency signals become increasingly difficult to detect due to the $(R/r)^l$ term in Equation (2.1). Taken together, these conditions results in memory inefficient models that become challenging to keep on-board spacecraft. In addition, these high-fidelity models come with a high computational cost. High-degree spherical harmonics models require reevaluating all of the associated Legendre polynomials at each field point. This calculation is recursive and therefore no trivial way to parallelize these computations exist. This leads to an unavoidable $\mathcal{O}(n^2)$ computational complexity which can severely limit both ground-based simulation and algorithms flown on-board (58) — see Appendix E and F for implementation details.

Beyond the computational inefficiencies, spherical harmonics also has operational limitations. The derivation of this model requires that all mass elements exist within a sphere of fixed radius (the Brillouin radius). If dynamicists require a potential or acceleration estimate within this sphere, the

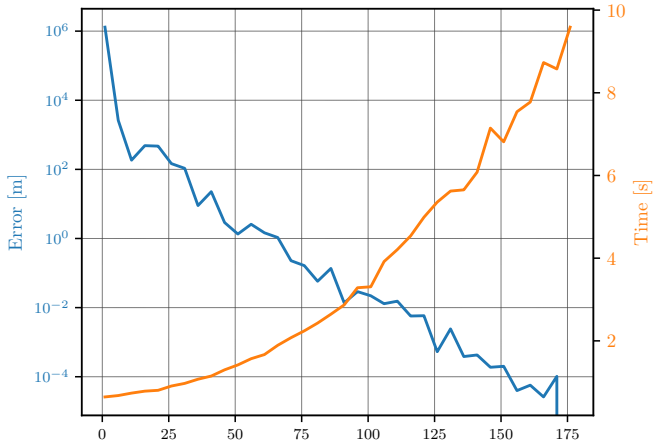


Figure 2.1: Compute time and final error associated with simulating a spacecraft at 600km altitude orbiting for four hours real-time using Pines' formulation as a function of spherical harmonic degree.

model can diverge thanks to the $(R/r)^l$ term in the expansion. While such effects are negligible for near-spherical planets or moons, they can become problematic in small-body settings. For asteroids or comets that exhibit highly non-spherical geometries like Eros or Itokawa, these effects can lead to large errors in the predicted accelerations and potentially risk the safety of a spacecraft (5).

2.2.2 Ellipsoidal Harmonics

The ellipsoidal harmonic model follows a similar approach to that of spherical harmonics but uses ellipsoidal harmonic basis functions instead (3). At a glance the potential can be represented as

$$V = \sum_{n=0}^{\infty} \sum_{p=1}^{2n+1} \alpha_n^p E_n^p(\lambda_1) E_n^p(\lambda_2) E_n^p(\lambda_2) \tag{2.2}$$

where α_n^p are constants, E_n^p are Lamé functions of the first kind, λ_i are the ellipsoidal coordinates, n is the degree, and p are the eigenvalues. This choice allows for a tighter bounding ellipsoid about the body than spherical harmonics, minimizing the region in which the model could potentially diverge. Despite this, the divergence remains a possibility inside the bounding ellipsoid and the model still suffers from the same challenges in representing discontinuity with periodic basis functions.

2.2.3 Interior Spherical Harmonics / Bessel Models

The interior spherical harmonic model inverts the classical spherical harmonic formulation and can model a local region whose boundary intersects only one point on the surface of the body (59). Explicitly, rather than computing the potential by using the law of cosine and factoring out $1/r$ through

$$U^e(r,\phi,\lambda) = G\int_M \frac{1}{r}\frac{1}{\sqrt{1-2(\frac{\rho}{r})\cos S + \frac{\rho}{r}^2}}dm \tag{2.3}$$

the interior spherical harmonic model instead factors out ρ

$$U^i(r,\phi,\lambda) = G\int_M \frac{1}{\rho}\frac{1}{\sqrt{1-2(\frac{r}{\rho})\cos S + \frac{r}{\rho}^2}}dm \tag{2.4}$$

This model maintains stability down to that single point making it valuable for precise landing operations; however, the solution becomes invalid on any other point on the surface and outside of the corresponding local sphere. The interior spherical Bessel gravity model expands on this theme, using bessel functions rather than spherical harmonics, and is able to achieve a wider region of validity. However, this comes at the expense of cumbersome analytics and retains some of the challenges of using harmonic basis functions to capture discontinuity (60).

2.2.4 Mascons

Mascon gravity models distribute a set of point mass elements within a body whose sum can form a global representation of the gravity field (61). Unfortunately, the accelerations can grow inaccurate at field points near the individual mascons (4). Hybrid mascon models offer a slightly more robust alternative to the pure mascon approach by representing each mascon with a low fidelity spherical harmonic model, but this incorporates additional complexity in regression and remains prone to both challenges of the mascon and spherical harmonic models (62).

2.2.5 Polyhedral Model

The polyhedral gravity model provides an alternative to the spherical harmonic model in these settings, offering a solution that maintains validity down to the surface of any body regardless of

shape. This stability makes the polyhedral model the primary gravity model for exploration about small-bodies such as asteroids and comets. If a shape model of the body is available (a collection of triangular facets and vertices which captures the geometry of the object), then the associated gravitational potential of that geometry can be computed under the assumption of constant density through:

$$\nabla U = -G\sigma \sum_{e \in \text{edges}} \mathbf{E}_e \cdot \mathbf{r}_e \cdot L_e + G\sigma \sum_{f \in \text{facets}} \mathbf{F}_f \cdot \mathbf{r}_f \cdot \omega_f \tag{2.5}$$

where G is the gravitational constant, σ is the density of the body, $\mathbf{E}_e$ is an edge dyad, $\mathbf{r}_e$ is the position vector between the center of the edge and the field point, L_e is an analog to the potential contribution by the edge, $\mathbf{F}_f$ is the face normal dyad, $\mathbf{r}_f$ is the distance between the face normal and the field point, and ω_f is an analog to the potential contribution by the face (5).

The polyhedral gravity model circumvents the numerical divergence within the Brillouin sphere, making it an extremely popular choice for small-body operations. Multiple works use this model to characterize the dynamical environment around small-bodies, and design corresponding trajectories (63; 64; 65; 66). Moreover, this model remains under active development, with researchers developing models which incorporate uncertainty in the shape (67) and more efficient evaluation strategies (68). However, this model does with some disadvantages. Foremost, this gravity model can be computationally expensive. High-resolution shape models have hundreds of thousands of facets and vertices which must be individually looped over to compute the acceleration at a single field point. When computing many accelerations or propagating trajectories, this computational burden can lead to excessively long runtimes. While this model can take advantage of parallelization more easily than spherical harmonics, such computational capabilities are not available on-board spacecraft — intrinsically limiting this model to ground-based simulation.

Additionally the polyhedral gravity model assumes that researchers know a density profile for the body in question. Most commonly the density is assumed to be constant, but literature shows that such assumption is not necessarily valid (69; 44). In addition, the polyhedral gravity model requires that a shape model of the body already exists. While these models can be acquired

in-situ, the process is non-trivial and time-consuming in practice (70; 71; 72).

2.2.6 Heterogeneous Polyhedral

An alternative formulation of the polyhedral model proposes treating each facet of the shape model as a tetrahedron connected to the center of the shape. The tetrahedra can be sliced into multiple parts, and different densities assigned to each slice based on some candidate density distribution (73). This allows for an analytic approximation of a heterogeneous density body, but continues to suffer from assumptions made about the candidate density distribution.

2.3 Machine Learning Models

As an alternative to the analytic models, recent efforts explore the use of machine learning to learn representations of complex gravity fields in a data-driven manner. In principle, these machine learning models can learn high-accuracy gravity fields without making assumptions about the body in question, while maintaining low computational cost.

2.3.1 Neural Networks

Neural networks a series of learned, non-linear transformations that map data from an input space to a desired output space by minimizing a prescribed loss function such as mean squared error:

$$\mathcal{J}(\boldsymbol{\Theta}) = \frac{1}{N_f} \sum_{i=1}^{N_f} |y_i - \hat{y}(x_i|\boldsymbol{\Theta})_i|^2 \tag{2.6}$$

where y_i is the true output, $\hat{y}_i(x_i|\boldsymbol{\Theta})$ is predicted output by the artificial neural network given the vector of trainable parameters $\boldsymbol{\Theta}$ which includes the weights, $\mathbf{w}$, and biases, $\mathbf{b}$, of the network, and N_f is the total number of points used to train the network.

Commonly these networks are constructed as a series of densely connected hidden layers with N nodes per layer expressed as:

$$h_i^{(k)} = \sigma\left(w_{ij}^{(k-1)} h_j^{(k-1)} + b_i\right) \quad k \in \{1, \ldots, k_{\max}\} \tag{2.7}$$

where $h^{(k)}$ is the k-th hidden layer, i is the node in the layer, w_{ij} are the weights connecting the hidden layers, b_i are the biases attached to the nodes in the layer, and σ is the non-linear transformation (typically sigmoid, hyperbolic tangent, or rectified linear unit). Note that $h^{(0)} = x$, and $h^{(k_{\max})} = \hat{y}$.

Neural network are trained by iteratively updating the weights and biases of the model to minimize Eq. (2.6) such that:

$$\mathbf{w}^* = \arg\min_{\mathbf{w}\in\mathbf{\Theta}}(\mathcal{J}(\mathbf{w})); \quad \mathbf{b}^* = \arg\min_{\mathbf{b}\in\mathbf{\Theta}}(\mathcal{J}(\mathbf{b})) \tag{2.8}$$

which can be solved using a gradient descent algorithm like Adam or SGD (74; 75)

$$\mathbf{\Theta}^{m+1} = \mathbf{\Theta}^m - \eta\nabla_{\mathbf{\Theta}^m}\mathcal{J}^m(\mathbf{\Theta}), \tag{2.9}$$

where η is the learning rate and m is the training iteration, and the gradient is taken using automatic differentiation.

Neural networks demonstrate considerable potential in their ability to regress highly accurate models of complex phenomena across a wide variety of scientific domains. Despite this much criticism exists regarding these models, as their overparameterization can lend itself to overfitting leading to unreliable models outside of the bounds of the original trained data. Moreover, these models are often described as black-box regressors which are difficult to interpret or provide any analytic guarantees. Despite their drawbacks, neural networks have been proposed as a candidate solution to the gravity modeling problem. By training on position and acceleration data, literature has shown that neural networks are capable of representing complex gravitational phenomena (46).

Automatic Differentiation

Automatic differentiation is method to compute the exact derivative of an algorithm with respect to any input. This is done by constructing a computational graph and using either a forward

or backward form of chain rule such as:

$$\begin{aligned}\frac{\partial y}{\partial x} &= \frac{\partial y}{\partial w_{n-1}}\frac{\partial w_{n-1}}{\partial x} \\ &= \frac{\partial y}{\partial w_{n-1}}(\frac{\partial w_{n-1}}{\partial w_{n-2}}\frac{\partial w_{n-2}}{\partial x}) \\ &= \dots\end{aligned}$$

where x is the input to some arbitrary algorithm, w_i are the sequence of intermediate calculations performed to produce the final output, y.

Because all algorithms are constructed from elementary functions with known derivatives, the partials of each intermediate expressions can always be computed alongside the original calculation. This property ensures that partial of the output y with respect to any input x can be computed automatically. Automatic differentiation is best known for its application within deep learning, where it commonly applied in the stochastic gradient descent algorithms used to train neural networks (76). Specifically, Equation 2.9 is where automatic differentiation is applied. The gradient of the prescribed loss function with respect to the weights and biases of a network can be computed automatically and applied in the descent to move the network's parameters in a direction that will minimize that loss function.

2.3.2 Extreme Learning Machines

Seeing as neural networks often receive warranted criticism due to their overparameterization, alternative models have been developed to alleviate some of these concerns. Specifically, an alternative class of machine learning model has been proposed called Extreme Learning Machines, or ELMs. ELMs are single layer neural networks which are fit by randomly initializing the weights from the inputs to the hidden layer, and then solving for the weights to the output layer using a least squares approach to minimize some quadratic cost function such as:

$$L(\theta) = \frac{1}{N}\sum_{i=0}^{N}|\hat{\boldsymbol{y}}_i(\boldsymbol{x}_i|\theta) - \boldsymbol{y}_i|^2 \tag{2.10}$$

where $\hat{y}_i(\boldsymbol{x_i}|\theta)$ is the machine learning model prediction at input $\boldsymbol{x}_i$ with trainable model parameters θ (77). The ELM is advantageous over other methods because it can be trained in a single iteration using least squares as opposed to neural networks which require many iterations of stochastic gradient descent and backpropagation. ELMs of sufficiently large width have also been proven as universal function approximators (78). ELMs were recently shown to successfully model the gravity field of the asteroid Itokawa in for an landing scenario, and have demonstrated fast evaluation speeds with relatively high accuracy (6). That said, despite the model's simplicity, ELMs do introduce different challenges. Namely, ELMs require taking large matrix inversions to perform the least squares fit, intrinsically limiting the amount of data that can be used to update the model at any given point in time.

2.3.3 Gaussian Processes

In addition to ELMs, Gaussian processes have also been proposed as candidate solutions to the gravity modeling problem (45). Gaussian processes are fit by specifying a prior distribution over functions, and updating that prior based on observed data. This requires the user to specify some kernel function which provides a metric of similarity between data, and computing a covariance matrix between all data points using that function. Once the covariance matrix is computed, it can be inverted and used evaluate the mean and covariance of the learned function at a test point.

Using Gaussian processes for gravity modeling is advantageous because it provides a probabilistic estimate of the uncertainty in the model's prediction; however the model does not scale well to large datasets. Specifically, the Gaussian process is characterized by a covariance matrix built from the training data. This covariance matrix scales as $\mathcal{O}(n^2)$ where n is the size of the training data. This scaling makes it computationally challenging for Gaussian processes to leverage large quantities of data, as the size of the covariance matrix and complexity of corresponding inversion grows rapidly with small sets of training data. As an example, in Reference (45) a Gaussian process was regressed over a mere 3,600 data points resulting in a model covariance matrix of over 12,960,000 parameters. Because this model size grows rapidly, researchers must carefully

select which training data to leverage, as these models inevitably cannot model the entire testing domain. Aside from this fact, the study does demonstrate that these models, once fit, can achieve fast evaluation times and relatively high-accuracy predictions near the training data.

2.3.4 GeodesyNet

In 2021, Reference (7) proposed the use of neural density fields to learn density maps for small bodies. This research takes inspiration from recent machine learning developments in Neural Radiance Fields (NeRFs) which are used to construct 3D shape models from relatively sparse image data (79). Once trained, the density predictions can be integrated numerically to produce corresponding gravitational potentials and accelerations. This work demonstrated promising results – achieving competitive acceleration accuracies and the ability to account for heterogeneous densities; however, the models are relatively large (in excess of 90,000 parameters), the numerical integration necessary to compute accelerations is computationally expensive, and the models continue to require large training datasets.

2.3.5 Physics-Informed Neural Networks

One of the disadvantages of traditional machine learning models is that they lack an analytic foundation. There are not guarantees that the model intrinsically satisfies some a differential equation or will perform accurately over the entire problem domain. When neural networks, extreme learning machines, or GeodesyNets are trained to represent a gravity field, researchers cannot ensure that these models are intrinsically compliant with the underlying physics like Laplace's equation or conservative vector field properties.

In 2019, Raissi et. al. recognized this problem and suggested that neural networks models do not need to be agnostic of the physics which govern the function they are attempting to represent (9). Instead, Raissi argues that networks **can** be trained specifically to ensure that learned representations obey underlying differential equation with the introduction of the original Physics-Informed Neural Network (PINN). PINNs inject differential equations and boundary conditions

into the cost function of traditional neural networks, using automatic differentiation to ensure that these solution learned naturally satisfy these constraints.

As an example, consider the following arbitrary differential equation:

$$f''(x) + f'(x) + f(x) = 0, \tag{2.11}$$

Assume there exist measurements of x as well as the corresponding values of $f(x)$. A traditional neural network can use these observations as training data to learn a mapping from $x \to \hat{f}(x\|\mathbf{\Theta})$ by minimizing the cost function $J(x\|\mathbf{\Theta}) = \|f(x) - \hat{f}(x\|\mathbf{\Theta})\|^2$. The risk of training the network with this particular loss function is that the network does not know that the mapping, $\hat{f}(x\|\mathbf{\Theta})$ must also satisfy Eq. (2.11). PINNs change this paradigm by inserting the original differential equation into the cost function:

$$J(\mathbf{\Theta}) = \frac{1}{N_f}\left(\sum_{i=1}^{N_f}\left\|f(x_i) - \hat{f}(x_i\|\mathbf{\Theta})\right\|^2 + \left\|\hat{f}''(x_i\|\mathbf{\Theta}) + \hat{f}'(x_i\|\mathbf{\Theta}) + \hat{f}(x_i\|\mathbf{\Theta})\right\|^2\right), \tag{2.12}$$

where the derivatives of the network $\hat{f}(x_i\|\mathbf{\Theta})$ are taken with automatic differentiation. This cost function, while similar to Eq. (2.6), not only penalizes erroneous values of $\hat{f}(x\|\mathbf{\Theta})$, but also penalizes when the learned function violates the differential form of the problem. This extra term serves as a form of regularization in the training process which can lead to improved solutions that conveniently also satisfy important physics properties. Much work has gone into the study and utility of these PINNs. Recent work has shown that these models often offer advantages in data efficiency and modeling accuracy over their traditional counterparts.

Chapter 3

Physics-Informed Neural Network Gravity Models

This thesis explores the use of the Physics-Informed Neural Network (PINN) to regress a form of the gravitational potential. The PINN gravity model has undergone multiple generations of development — each stage introducing additional sophistication to improve model accuracy, robustness, and data efficiency. This chapter highlights the chronological developments of each generation and the corresponding performance at each stage.

3.1 Generation I

Past gravity modeling efforts have demonstrated that machine learning can be used to regress models of a gravity field (6; 45). This can be accomplished by collecting estimates of spacecraft position, $\boldsymbol{x}$, and acceleration, $\boldsymbol{a}$, and using this as a training data set to fit a neural network or extreme learning machine by minimizing the following cost function

$$J_{00}(\Theta) = \frac{1}{N_f} \sum_{i=1}^{N_f} |\boldsymbol{a}_i - f(\boldsymbol{x}_i|\Theta)|^2 \tag{3.1}$$

where $f(\boldsymbol{x}|\Theta)$ is the learned mapping of the acceleration vector parameterized by the weights and biases of the network Θ (i.e. $f(\boldsymbol{x}|\Theta) = \hat{\boldsymbol{a}}$).

These former efforts, while important contributions to the literature, are not especially satisfying as they are agnostic of the physics of the system they are trying to represent. Unlike the spherical harmonic or polyhedral gravity models, the solutions learned by the neural networks had no knowledge that the gravitational accelerations it is representing are the byproduct of a more

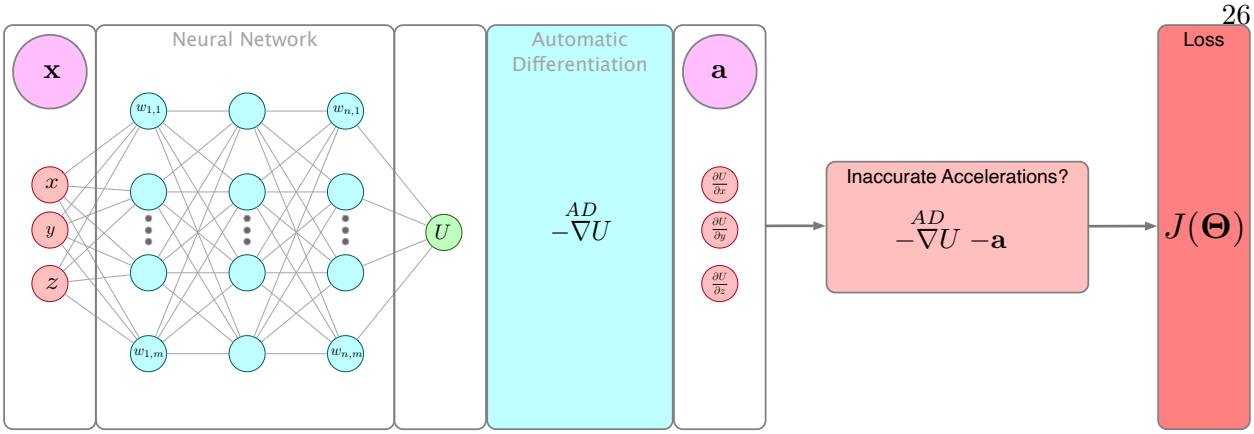


Figure 3.1: PINN-GM Generation I

fundamental scalar potential function. Moreover, these solutions are not aware that the gravitational force is conservative, nor that the behavior at high-altitudes needs to tend towards zero. Until recently, researchers did not have a way to include this knowledge in the construction of these machine learning models. However, in 2019, Raissi et. al. introduced the physics-informed neural network (PINN) which offered the first glance on how this could be accomplished (9).

As discussed in Chapter 2, PINNs are machine learning models that leverage an augmented loss function. These loss functions not only penalize modeling inaccuracies between the predicted and true value, as is the case in traditional machine learning, but they also penalize violations of the underlying differential equations of the system they are attempting to model. In the case of the gravity field modeling problem, physics implies that gravitational accelerations are really byproducts of a more fundamental scalar potential through:

$$\mathbf{a} = -\nabla U \tag{3.2}$$

PINNs can leverage this differential equation directly within the cost function:

$$J_{\mathrm{A}}(\mathbf{\Theta}) = \frac{1}{N_f} \sum_{i=1}^{N_f} \left\| \mathbf{a}_i + \overset{\mathrm{AD}}{\nabla} \hat{U}(r_i \| \mathbf{\Theta}) \right\|^2, \tag{3.3}$$

where $\mathbf{a}_i$ is the estimated acceleration at position $\mathbf{r}_i$, $\hat{U}$ is the learned potential function, and $\overset{\mathrm{AD}}{\nabla}$ is the gradient operator applied via automatic differentiation.

Equation 3.3 is a physics-informed loss function. Rather than simply minimizing the difference between a predicted acceleration and the true acceleration, Eq. 3.3 enforces that the predicted acceleration must be related to the gravitational potential. In this way, the neural network is not directly representing the acceleration vector, but instead is learning the scalar potential function, $\hat{U}$, and enforcing that its negative gradient matches the measured acceleration.

One of the advantages of this approach is the improved data efficiency. When training a traditional neural network learns the acceleration vector directly, the three components of vector are effectively treated as distinct and individual features for the network to learn. In fact, these components are, by construction, orthogonal such that there is no observable relationship between them. As such, a traditional neural network must learn a basis set that can solve three distinct problems simultaneously. The PINN gravity model, in contrast, only needs to learn a scalar potential function; leveraging all three of the acceleration components to regress a single feature. As a consequence, PINNs can make more efficient use of the same amount of training data, both decreasing training times and helping alleviate dynamicists from needing to collect large datasets.

It should be noted that additional physics-informed constraints can be added to the loss function. For example, the cost function could also include a penalty for:

(1) Mis-modeling the potential function itself, $U(r) - \hat{U}(r\|\boldsymbol{\Theta}) = 0$

(2) Violating the boundary condition $U(\mathbf{r}) = 0$ as $\|r\| \to \infty$

(3) Not satisfying conservative vector field properties like $\nabla^2 \hat{U} = 0$ and $\nabla \times \hat{U} = 0$.

This first generation PINN-GM, or PINN-GM-I, purposefully omits these additional physics-informed constraints for two reasons. Optional constraint (1) is omitted because the domain of the potential is order of magnitudes larger than the domain of the corresponding accelerations. These differing scales pose numerical challenges for learning, and can introduce excessively large gradients during backpropagation. Additionally, this choice reflects the more common data circumstances where a high-fidelity potential of the body in question does not exist and therefore cannot be used.

Constraints (2) and (3) are omitted because, in the early stages of this work, they tended to drive the PINN solution for $\hat{U}$ towards zero or simply slow convergence. This is attributed to the multi-objective optimization that gradient descent algorithms struggle to accommodate (80). Explicitly, the constraints of a conservative vector field can be satisfied if the network identifies $U = 0$ as the optimal solution. This can force the network away from the non-zero solution guided by the constraint $\boldsymbol{a} + \nabla U = 0$. In theory, these competing objectives can be better balanced through dynamically adjusting learning rates for each of the objective costs (81). This strategy is explored in later generations of the PINN-GM.

Finally, it is worth stressing that the presented PINN gravity model formulation **does not require** a preexisting gravity model to produce training data. PINNs can be trained in-situ, relying on estimates of accelerations either via finite differencing of relative velocities or more advanced filtering techniques. This chapter chooses to focus primarily on the gravity modeling problem, not the gravity estimation problem. As such, existing high-fidelity models are used as training data for this chapter, but Chapter 6 will provide a more in-depth discussion of how PINNs can be trained in-situ without a pre-existing gravity model.

3.1.1 Network Architecture and Hyperparameters

All networks trained in the upcoming experiments share an architecture of eight densely connected hidden layers with N nodes per layer. The choice of eight hidden layers offered a desirable balance between network capacity and reasonable training times. As discussed, the networks' training data consists of position and acceleration pairs generated from preexisting high-fidelity gravity models. For all networks, the position data are preprocessed via a min-max transformation fit to each component of $\boldsymbol{r}$ into the bounds of $[-1, 1]$. This normalization ensures that the inputs exist in the favorable, non-linear part of the network's activation function, as well as assist with numerical stability during training. In addition, the acceleration data output is also normalized via a min-max transformation. Note, that for the traditional neural networks, the min-max transformation can be applied for each individual component of the acceleration vector. For PINNs,

however, all components must be scaled simultaneously to the dimensions are physically compliant for the learned potential function.

For PINN-GM-I, the hyperparameters that have the largest effect on performance are the learning rate and mini-batch size. If the batch size is too small, the gradient descent algorithm can move in directions other than the local minimum resulting in longer training times. Unfortunately small batches are often unavoidable when the available GPU does not have sufficient VRAM to store the entire dataset on the device. In this case, smaller and more cautious learning rates should be used to ensure the optimizer does not quickly lead the network to a suboptimal minimum that later becomes challenging to escape. Conversely, if a large batch size can be used, larger learning rates are encouraged as they yield shorter training durations that are more likely to descend in the direction of the true gradient of the cost function.

Independent of initial learning rate magnitude, it can be advantageous to slowly decrease the learning rate towards the end of training to prevent the weights from oscillating above the cost function's local minimum (82). To achieve this, an exponential decay is applied to the learning rate:

$$\eta_i = \begin{cases} \eta_0 & i < i_0 \\ \eta_0 * \text{pow}\left(\alpha, -\frac{i-i_0}{\sigma}\right) & i \geq i_0 \end{cases}, \tag{3.4}$$

where η_i is the learning rate at epoch i, i_0 is the reference epoch after which the decay begins, σ is the scale factor, α is the decay rate, and η_0 is the initial learning rate.

An additional critical hyperparameter is the activation function. When training a PINN, it is important that the activation function selected has a sufficiently high order of continuity. If the activation function does not have smooth high-order derivatives, and if gradients of the network are taken using automatic differentiation to enforce the physics constraints, the cost function will no longer be well-behaved for gradient descent. As such, it is recommended to avoid using the popular rectified linear unit (ReLU) or leaky ReLU and instead opt for functions with infinite orders of continuity like hyperbolic tangent or the gaussian exponential linear unit (GELU). The remaining default hyperparameters for the PINN-GM-I are shown in Table 3.1 and any custom

hyperparameters are shown in Table 3.2.

Each network is trained for 100,000 epochs using the Adam optimizer in Tensorflow 2.4 [1] on a NVIDIA RTX 2060 graphics card. This training duration ensured the validation loss consistently plateaued, indicating that learning had terminated. All network weights are initialized using the Xavier uniform initialization scheme detailed in Reference (83).

Table 3.1: Shared hyperparameters for the traditional and physics-informed neural networks trained in this work

Hyperparameter	Value	Hyperparameter	Value
Learning Rate, η_0	0.005	Initializer	Glorot Uniform
Patience Epoch Start, i_0	25000	Epochs	100000
LR Scheduler Decay Rate, α	0.5	$\boldsymbol{x}$ transform	MinMax
Scale Factor, σ	25000	Activation	GELU
Optimizer	Adam	Number of Layers	8

3.1.2 Spherical Harmonic Performance

Spherical harmonics are the defacto gravity model for large planetary bodies. The most accurate spherical harmonic gravity model for the Earth is the Earth Gravitational Model 2008 (EGM-2008) which contains spherical harmonics that extend to degree and order of 2,160 — totalling over 4,000,000 parameters (53). While spherical harmonics are especially convenient at representing global scale gravitational features with low degree harmonics, their initial efficiency does not persist at higher degrees.

To demonstrate, consider the prominent gravitational features that remain after removing the point mass, planetary oblateness, and planetary obliquity accelerations (i.e. the accelerations produced by harmonics above degree and order 2). To view these features, a variable δa is introduced:

$$\delta a(\boldsymbol{r}) = \left\| -\nabla U^{\mathrm{SH}}_{\mathrm{Truth}}(\boldsymbol{r}) - \left(-\nabla U^{\mathrm{SH}}_{2}(\boldsymbol{r})\right) \right\|, \tag{3.5}$$

[1] https://www.tensorflow.org/

Table 3.2: Unique hyperparameters for the traditional and physics-informed neural networks trained in this work

Network Type	Nodes Per Layer (N)	Model Parameters	Batch Size	$\mathbf{a}$ Transform
Traditional	10	843	262144	MinMax
Traditional	20	3083	262144	MinMax
Traditional	40	11763	262144	MinMax
Traditional	80	45923	262144	MinMax
PINN	10	820	262144	Uniform
PINN	20	3040	262144	Uniform
PINN	40	11680	262144	Uniform
PINN	80	45760	131072	Uniform

where $U^{\text{SH}}_{\text{Truth}}$ is constructed using EGM-2008 expanded to degree $l = 1,000$, U^{SH}_{2} is that same model expanded to only degree and order 2, and the gradient of the potential is taken using Pines' algorithm to avoid singularities at the poles (84). Figure 3.2 shows these perturbations computed at the Brillouin sphere of Earth. This figure verifies that the perturbations are discontinuous features in the crust like mountain ranges such as the Himalayas and Andes, tectonic subduction zones as best seen in the Pacific, and hotspots scattered across the globe. These findings align with intuition as the accelerations are directly proportional to the gradient of the potential ($\boldsymbol{a} = -\nabla U$) and large displacements in landmass generate large changes in the potential.

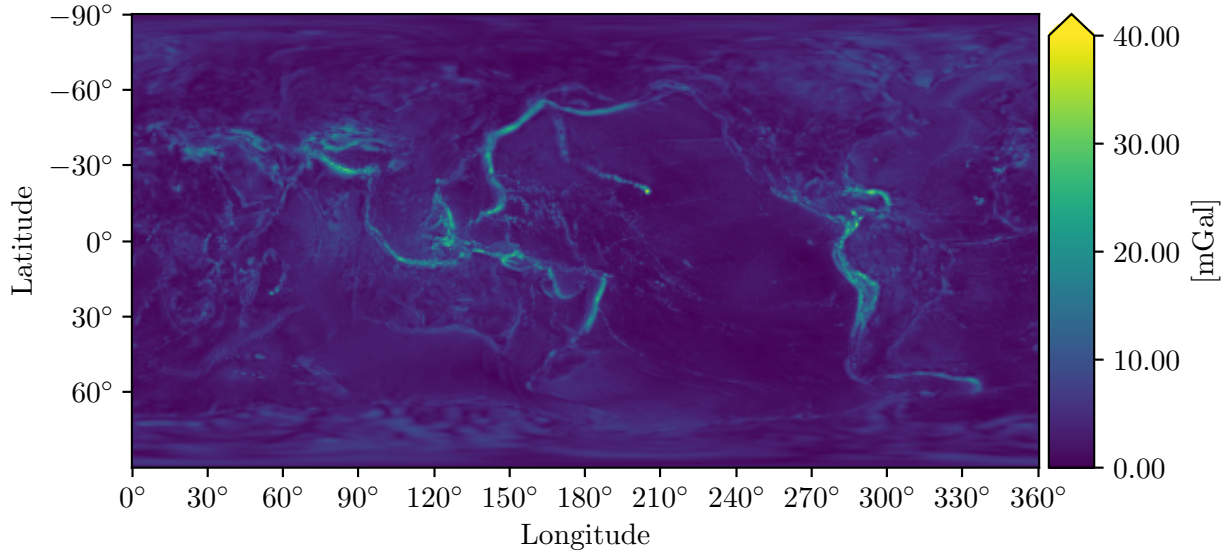


Figure 3.2: Map of δa at the Earth's Brillouin sphere

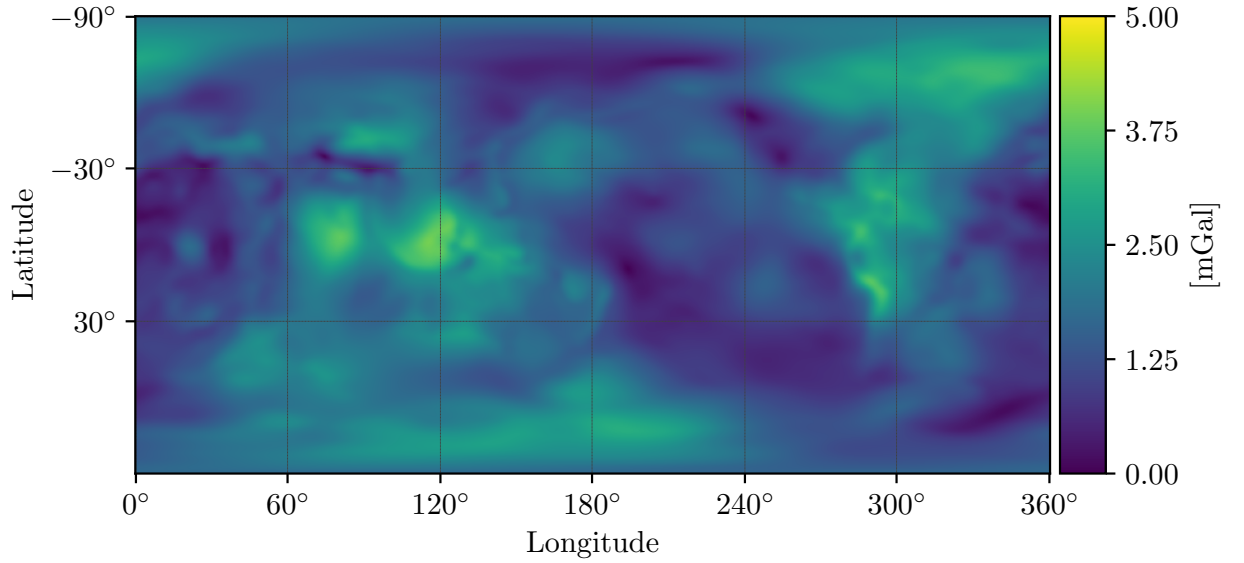


Figure 3.3: Map of δa of the Earth at a LEO altitude (approximately 420 km)

These perturbations are especially important features to capture with a gravity model, second only to planetary oblateness. It is therefore reasonable to question how efficiently do spherical harmonic models represent these perturbations. Specifically, when low-degree spherical harmonic models are used, how much error exists in these important parts of the gravity field? To investigate, a mean root-squared error (MRSE) metric is introduced:

$$\text{MRSE}(\mathcal{A}) = \frac{1}{N}\sum_{i=1}^{N_f} \delta a_p(\boldsymbol{r}_i) \quad \boldsymbol{r_i} \in \mathcal{A}, \tag{3.6}$$

where $\mathcal{A}$ is the set of positions for which the gravity field is evaluated, i.e. field points, N_f is the total number of field points in set $\mathcal{A}$, and δa_p is a generalization of Eq. (3.5) used to measure the corresponding acceleration error of a low-fidelity spherical harmonic model through:

$$\delta a_p(\boldsymbol{r_i}) = \| - \nabla U^{\text{SH}}_{\text{Truth}}(\boldsymbol{r}_i) + \nabla U^{\text{model}}_{p}(\boldsymbol{r}_i)\|, \tag{3.7}$$

where p represents the maximum number of parameters / coefficients used in the gravity field model being evaluated.

To characterize the error of low-fidelity spherical harmonic models, the MRSE metric is applied to three datasets: $\mathcal{A}, \mathcal{F}$, and $\mathcal{C}$. The $\mathcal{A}$ dataset includes $N_f = 250,000$ field points distributed

in a Fibonacci grid at the Brillouin sphere of Earth. The Fibonacci grid is chosen to ensure a near isotropic distribution of data about the Brillouin sphere, thereby avoiding the clustering of data at the poles were the set to be distributed uniformly in latitude and longitude (85). The second set, $\mathcal{F}$, is a subset of $\mathcal{A}$ that only includes data points within the dominant gravitational perturbations. Specifically, $\mathcal{F}$ is generated by selecting the field points within $\mathcal{A}$ whose acceleration exceed 2 standard deviation of the mean acceleration such that $\mathcal{F} : \{\|\delta a(\boldsymbol{r}_i) - \delta\bar{a}\| > 2\sigma_{\boldsymbol{a}}(\mathcal{A})\}$. The third and final set, $\mathcal{C}$ is the compliment of set $\mathcal{F}$ ($\mathcal{C} : \mathcal{A}/\mathcal{F}$) representing the background of the gravity field.

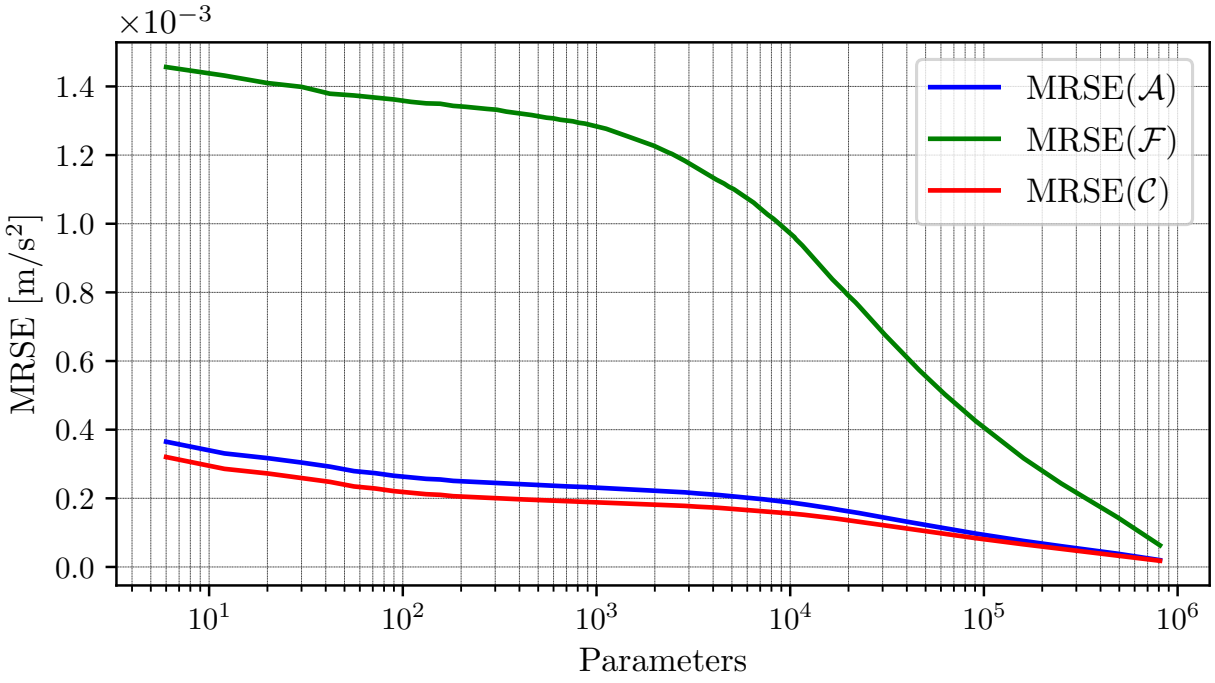


Figure 3.4: Plot of MRSE($\mathcal{A}$), MRSE($\mathcal{F}$), and MRSE($\mathcal{C}$) as a function of total parameters, p, in used the spherical harmonic gravity model where $p = l(l+1)$

The MRSE metric is applied to sets $\mathcal{A}, \mathcal{F}$, and $\mathcal{C}$ and presented in Figure 3.4 as a function of total parameters used in the spherical harmonic representation. This figure quantifies the relationship between spherical harmonic model size and model accuracy. The blue MRSE curve for $\mathcal{A}$ tells a deceptively simple story: the more parameters used in a spherical harmonic model, the better the average accuracy across the Brillouin sphere. However, if the data is separated between the dynamically significant features $\mathcal{F}$ versus the less significant features $\mathcal{C}$, a much more interest-

ing result emerges. Specifically, the error within the dominant gravitational features, $\mathcal{F}$, is over 10 times greater than its background counterpart in the low-parameter regime. In fact, it takes nearly 10,000 parameters, or a spherical harmonic model of degree 110 or larger, before that gap in performance begins to decrease at an appreciable rate.

The discrepancy between the modeling error of $\mathcal{A}$ and $\mathcal{F}$ showcases how the spherical harmonic representation struggles to capture perturbations in order of dynamical significance. While the initial efficiency of representing Earth's oblateness is undeniable, the convenience does not extend into the next most important perturbations. This is because spherical harmonics prioritize fitting prescribed geometries onto a system in which those geometries are not naturally present. As a consequence, the spherical harmonic model must superimpose many high-order frequencies / harmonics before capturing these perturbations.

To a dynamicist, this implies that the next most important perturbations beyond J_2 require, at the minimum, a spherical harmonic model that exceeds degree $l = 110$ if they aspire to incorporate the dynamic effects of the Earth's high-order perturbations into their application. In some circumstances this may not be a problem. When a sufficiently high-fidelity model exists and the researcher is not computationally limited, spherical harmonics will eventually converge even over the discontinuous features. On-board spacecraft, however, computational resources may be limited or a high-fidelity spherical harmonic model may not exist for the body in question. In these conditions, operations over short time scales and near large surface features could be negatively affected by spherical harmonics inability to efficiently represent these perturbations.

Moreover, the results shown in Figure 3.4 motivate why this research turns to learned neural network gravity representations as an alternative to spherical harmonics. The spherical harmonics basis is inherently limited in resolving discontinuous perturbations while retaining a compact model size. The perturbations present on the Earth require small-wavelength harmonics which are only present in high-degree expansions. Neural networks, in contrast, do not have prescribed basis functions and corresponding characteristic wavelengths. There is no inherent minimum of 10,000 parameters needed to represent a specific mountain range or other discontinuity. Rather, the

neural networks learn a convenient basis that represents the most important perturbations of the field independent of their geometry or scale. In principle, neural network gravity models can therefore yield more compact representations that achieve comparable, if not greater, accuracy than traditional spherical harmonics.

3.1.3 Representational Compactness

The following experiment aims to determine if the PINN-GM-I can produce equally accurate gravity models using fewer parameters than its spherical harmonic counterpart. For this experiment, model parameters, p, refer to the total number of coefficients used in a spherical harmonic model or the number of trainable weights and biases of a network model.

The experiment begins by training both traditional and physics-informed neural networks on 5,000,000 position / acceleration vector pairs which are drawn randomly from a uniform distribution in altitude (0-420 km), latitude, and longitude. A total of eight networks are trained, each with varying model capacity as presented in Table 3.2. Once trained, the MRSE metric (Eq. (3.6)) is used to evaluate the performance of each network using the same Fibonacci grid data as was used to generate Figure 3.4 — $\boldsymbol{r}_i \in \{\mathcal{A}, \mathcal{F}, \mathcal{C}\}$. The Fibonacci test samples provide a entirely decoupled dataset from the training data to ensure a fair evaluation of the network performance. The MRSE for the networks is juxtaposed with the MRSE of the spherical harmonic model and presented in Figure 3.5.

Figure 3.5 demonstrates that there exists a wide range in which the neural networks generate more accurate models of the gravity field using fewer parameters compared to their spherical harmonic counterparts. This range spans between $1,000$ to $50,000$ parameters, or between spherical harmonic degree $l = 30$ and $l = 225$. This is best seen in the $\mathcal{F}$ dataset, where the PINNs can produce models of equal accuracy using an order of magnitude fewer parameters than spherical harmonics. Traditional neural networks, also produce more compact representations, albeit less than the PINN model of similar size.

That said, there are conditions where the PINNs' compactness advantage becomes less appar-

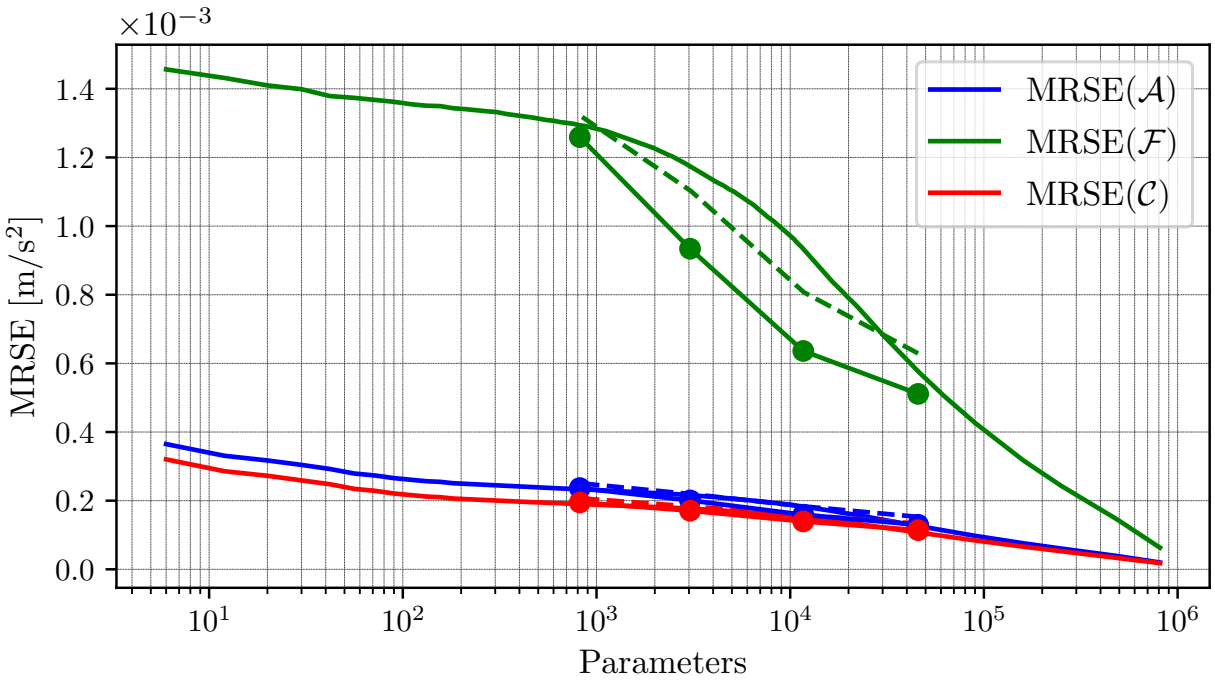


Figure 3.5: Plot of MRSE as a function of total model parameters, p. Solid lines represent the spherical harmonic representation. Dashed lines represent traditional neural networks. The lines with circle markers represent the physics-informed neural networks

ent. Specifically, note the neural networks with the smallest and largest capacities ($N = \{10, 80\}$). These particular networks have a less pronounced compactness advantage which is assumed to be the result of two factors. In the case of $N = 10$, the networks do not have a sufficiently high modeling capacity to represent the non-linear perturbations of the gravity field. With a mere $p \approx 1,000$ trainable parameters, these small networks do not have the parametric flexibility necessary to capture the discontinuous high-order perturbations. The $N = 80$ case, in contrast, has such a large network capacity that it manages to model these features quite well — so well in-fact that the network begins to overfit to the training data and suffer when tested on new data. This can be remedied with additional training data, or more clever model design as will be shown in later sections.

The compactness advantage presented here suggest that the machine learning models are able to learn a set of basis functions that are substantially more efficient at representing the high-order perturbations of Earth's gravity field. In the case of the traditional neural networks, these basis functions and the intermediate non-linear transformations used to generate them can be directly plotted as shown in Figure 3.6.

Figure 3.6 can be interpreted as follows: The top row represents the first layer in the network — i.e the normalized cartesian position vectors inputs. Because these images are generated at the Brillouin sphere, the x and y components of the position vectors grow small at the poles and large at the equator, whereas the z component grows linearly from the south to north pole. The second row represents weighted, linear combinations of the first three inputs that are then passed through the gaussian exponential linear unit (GELU) activation function. This row corresponds to the outputs of the first hidden layer of the network. This non-linear transformation is repeated for each intermediate hidden layer until the 9th row of the figure (the 8th hidden layer) which represents the learned basis functions. These penultimate functions are then combined linearly without the GELU transformation to produce the three predicted acceleration components at the Brillouin sphere of Earth (Figure 3.7).

Notably, as the inputs propagate deeper into the network, the corresponding outputs of the

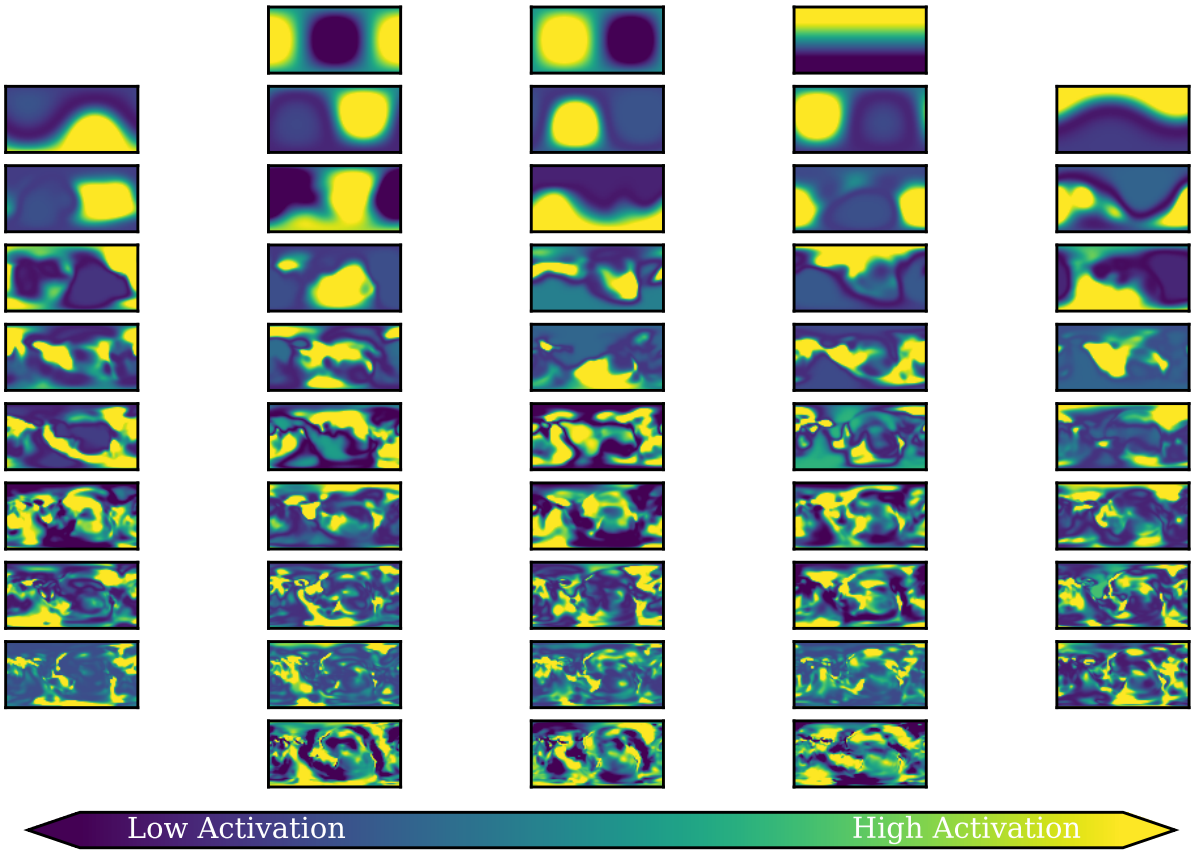


Figure 3.6: Subset of the intermediate transformations and resulting basis function of the $N = 40$ traditional neural network. Each row corresponds with a single layer of the network in order of input (top) to output (bottom). The individual plots represent the normalized and dimensionless output of a particular node's activation function when evaluated across the Earth's Brillouin sphere

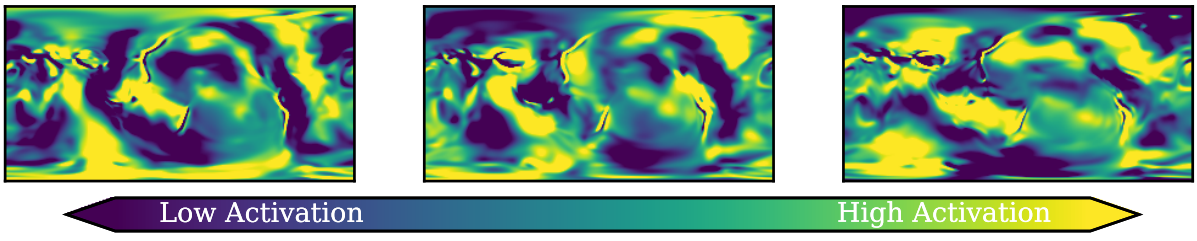
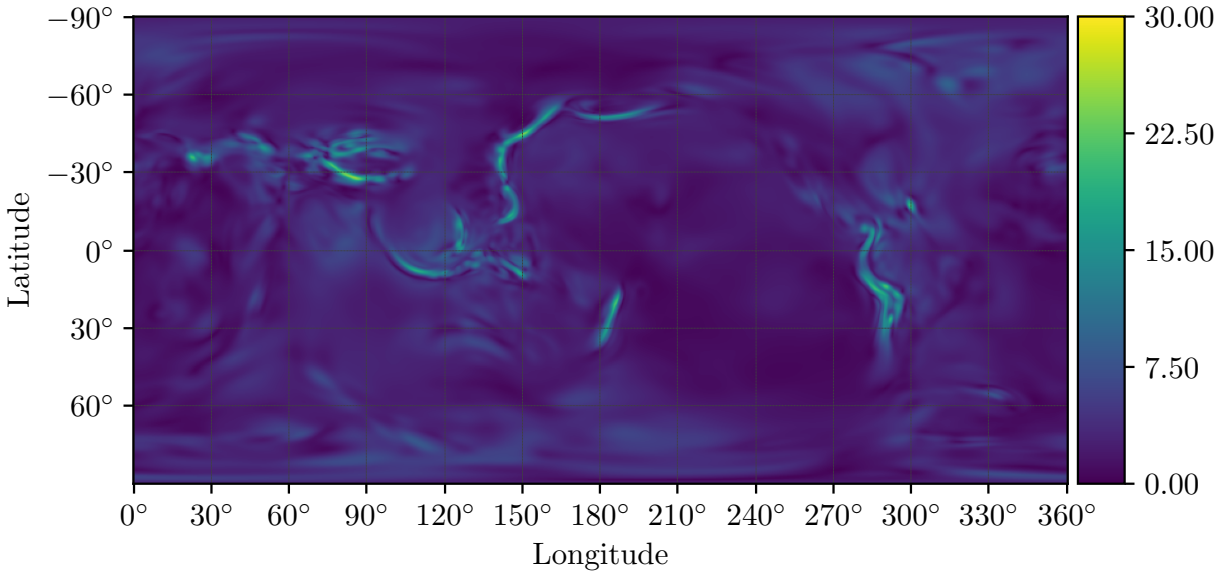
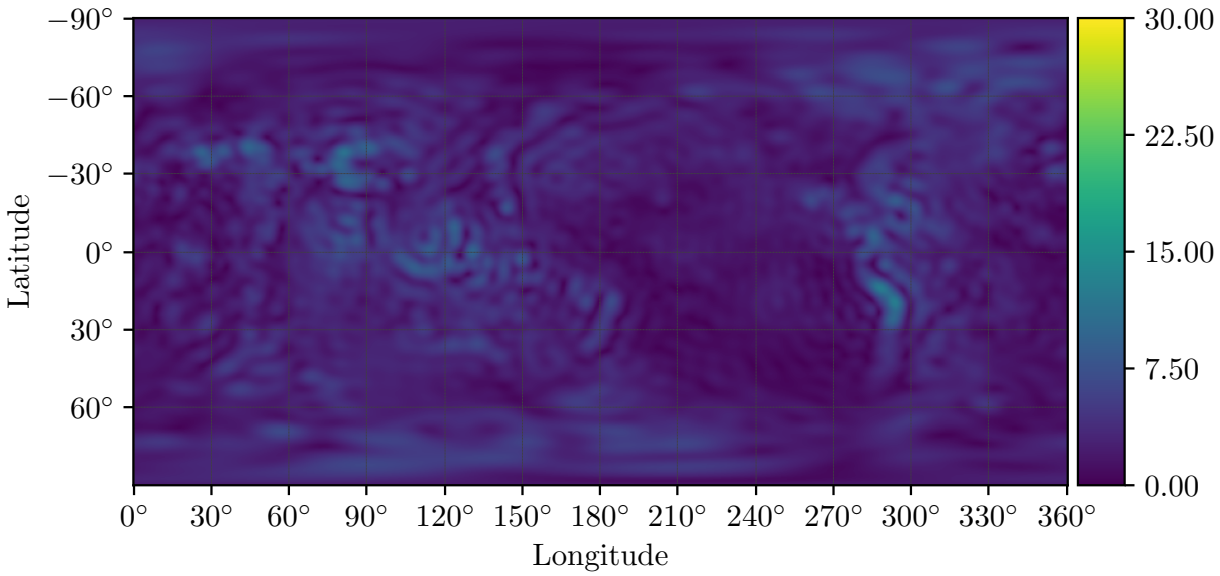


Figure 3.7: Zoomed in final layer of Figure 3.6 representing the predicted cartesian components of the acceleration vectors plotted at the Brillouin sphere



(a) PINN $N = 20 \Leftrightarrow p = 3,040$



(b) Spherical harmonics $l = 55 \Leftrightarrow p = 3,080$

Figure 3.8: Gravity model of Earth using (a) the neural network representation and (b) the spherical harmonic representation given approximately the same number of free parameters ($p \approx 3,000$)

hidden layers grow increasingly complex. The earlier layers activate over broad regions across the entire Brillouin sphere, whereas the deeper layers activate over more localized features. This highlights how the shallower layers in networks tend to resolve high-level, abstracted feature spaces while the deeper layers begin conforming to the specific perturbations of the body in question.

Figure 3.8 visually demonstrates the difference between modeling Earth's discontinuous perturbations using low parameter spherical harmonic model and the more flexible neural network representation. Figure 3.8a shows how the network representation is able to generate a sensible and accurate basis set capable of representing the most prominent perturbations. Conversely, spherical harmonics prescribe oscillatory basis functions which are not amenable to modeling discontinuous mountain ranges and subduction zones as shown in Figure 3.8b. In fact, the spherical harmonic basis can leave unintended wave patterns that obfuscate the important perturbations. Figure 3.8 thereby highlights one of the important takeaways of this research: Astrodynamicsts do not need to apply a one-size-fits-all basis to every gravity field (i.e. spherical harmonics); instead neural network provide astrodynamicists with the choice of generating unique basis functions that are maximally efficient for their specific gravity modeling problem.

3.1.4 Generalization

The prior experiment focused on the PINN-GM-I's accuracy at the Brillouin sphere where the gravitational perturbations are most prominent. While the surface of the body offers the most complex dynamics due to the unattenuated perturbations, most spacecraft operate at higher altitudes where dynamics tend to simplify. As shown in Figure 3.3, the perturbations tend to decrease in magnitude and span larger spacial scales as altitude increases. This is a function of the $(R/r)^l$ term within Eq. 2.1 which rapidly reduces the contribution of high-degree / small wavelength harmonics at larger radii. A second experiment is proposed which investigates how well the PINN-GM-I generalizes to these higher altitudes, as well as studies the role of the distribution of training data on network performance.

Uniform Distribution

The first part of this experiment trains the traditional and PINN gravity models with 5,000,000 position and acceleration pairs distributed uniformly between 0-420 kilometers. This represents the most complete and optimistic set of training data to benchmark an upper-bound on performance. The model performance is measured using the same latitude and longitude of the 250,000 data Fibonacci grid detailed in Sect. 3.1.3, however the grid is evaluated at altitudes varied between 0 to 500 km in 10 km increments. These grids maintain the same $\mathcal{A}$ and $\mathcal{F}$ sets, simply evaluated at different altitudes which will henceforth be referred to as $\mathcal{A}_m$ and $\mathcal{F}_m$, where $m \in [0, 500]$ km. The MRSE metric is then applied to these altitude specific sets, converted into an equivalent spherical harmonic model degree, and plotted in Figure 3.9.

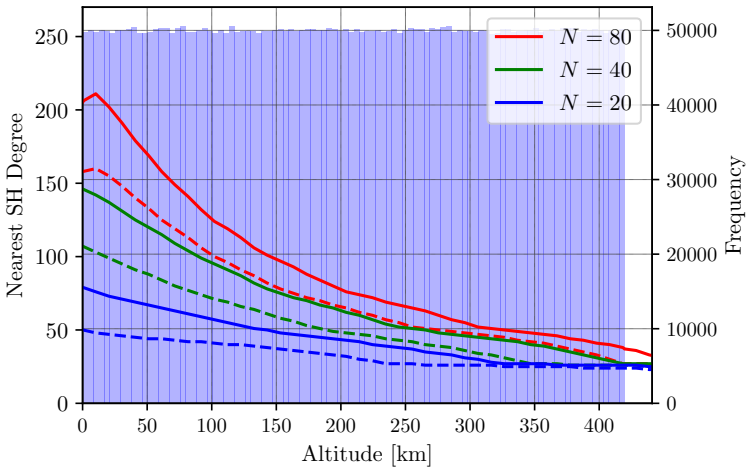
Figure 3.9 demonstrates that the networks' performance decays as a function of altitude. On average, the networks outperform their corresponding spherical harmonic equivalent for the first 100 km, but beyond 100 km, spherical harmonics become the more compact representation. Across all tested altitudes, the networks remain more accurate than degree and order 25 spherical harmonic model suggesting the networks are well-behaved within the domain of the training data and remain a viable option for gravity field modeling up to a LEO altitude.

Non-Uniform Distributions

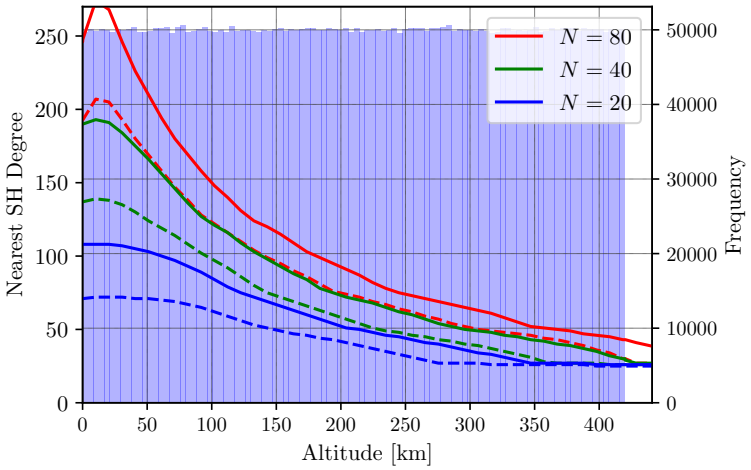
The second part of the experiment investigates the PINN-GM-I performance when the training data is not distributed uniformly in altitude. In more realistic mission circumstances, spacecraft will not be able to fly close to the surface easily and samples must be collected primarily from orbit. To reflect these circumstances, two new datasets of 1,000,000 position / acceleration pairs are drawn from the following exponential distribution:

$$\mathbb{E}(x, x_0, \beta) = \exp\left\{-\frac{\|x - x_0\|}{\beta}\right\}, \tag{3.8}$$

where x_0 is the reference altitude of 420 km, and β is the scaling parameter. The first dataset sets $\beta = 10$ km and simulates a data distribution collected by a spacecraft that begins in a high



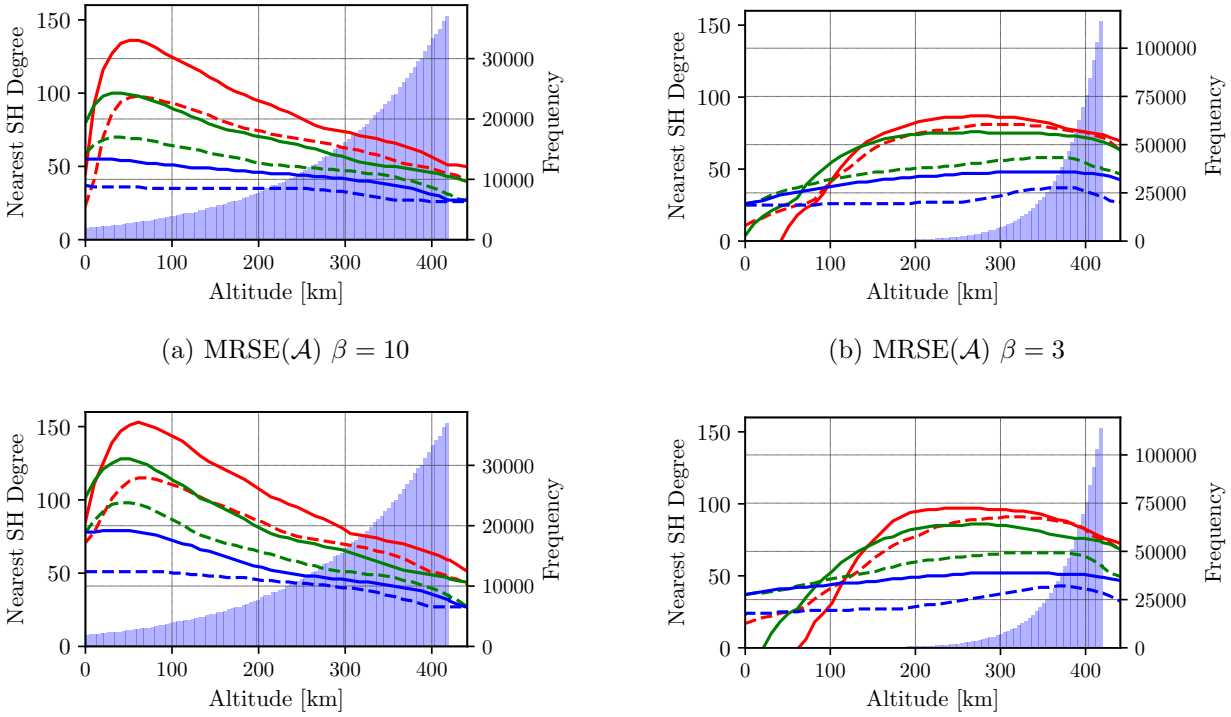
(a) MRSE($\mathcal{A}_m$)



(b) MRSE($\mathcal{F}_m$)

Figure 3.9: MRSE of $\mathcal{A}_m$ (top) and $\mathcal{F}_m$ (bottom) for the traditional (dashed) and physic-informed (solid) neural networks converted into the equivalent spherical harmonic degree as function of altitude. The blue histogram represents the training data distribution

altitude orbit before gradually deorbiting. As such, the majority of the data would come from an operational orbit regime with sparser measurements closer to the surface of the body. The second distribution sets $\beta = 3$ km and represents a satellite in an eccentric orbit that remains at — and collects data from — an operational orbit altitude. There may exist infrequent measurements near the surface but virtually all of the data are collected at altitudes greater than 200 km. Note that to prevent sampling from inside the Brillouin sphere, the distribution for both datasets is restricted to $x \in [0, 420]$ km. The results of the $N = \{20, 40, 80\}$ traditional networks and PINNs are shown in Figure 3.10.



(a) MRSE($\mathcal{A}$) $\beta = 10$

(b) MRSE($\mathcal{A}$) $\beta = 3$

(c) MRSE($\mathcal{F}$) $\beta = 10$

(d) MRSE($\mathcal{F}$) $\beta = 3$

Figure 3.10: Training data distribution and equivalent spherical harmonic degree at varying altitudes for the $\beta = 3$ and $\beta = 10$ datasets. Solid lines represent the PINNs and dashed lines represent the traditional neural networks

Discussion

Figures 3.9 and 3.10 demonstrate that both the traditional neural network and PINN gravity models are most productive when exposed to low-altitude samples, even if infrequently. Figures 3.10b and 3.10d show that the networks struggle to predict accurate accelerations at the Brillouin sphere given no data, but with a mere 2,000 data collected near the surface, the networks achieve substantially better performance at lower altitudes as shown in Figures 3.10a and 3.10c.

This sensitivity to low-altitude sample highlights an intrinsic bias of the early generations of the PINN-GM. Specifically, these neural network models are trained to minimize a cost function that compares the squared difference between the true and predicted accelerations. This inadvertently biases the network to accurately model low-altitude samples first, because low-altitudes are where the accelerations vectors are the largest. At these low-altitudes, even small relative modeling errors will contribute more to the cost function than large relative errors at high altitudes. By consequence, the early generation PINN-GMs will always prioritize these samples, even if they are relatively small fraction of the total training dataset.

Despite this bias, it is worth noting that the physics-informed networks consistently outperform their traditional networks counterparts again highlighting their greater data efficiency. There are minor exceptions, however, in the low-altitude regime of the $\beta = 3$ distribution. In these regimes, the networks are fully extrapolating beyond the bounds of their original training data, so it is generally not advisable to use either the traditional or PINN gravity models.

These generalization results suggest that the PINN-GM-I is most advantageous when used near the surface of a body. For Earth-based spacecraft operations, this is not a common operating regime as, at these altitudes, atmospheric drag alone would produce greater dynamic uncertainty than the high-order gravity perturbations. For bodies with very thin or non-existent atmospheres however (e.g. the Moon or small-bodies), lower orbit altitudes are feasible and present a viable use-case for the network representations. Later PINN-GM generations explore ways to remedy this altitude dependence as will be shown in Chapter 3.3.

3.1.5 Computational Speed

The last experiment for the PINN-GM-I investigates how quickly the trained network models can be executed as compared to other popular gravity representations. The neural network gravity models are written in Tensorflow 2.4 and executed on an NVIDIA RTX 2060 GPU and on the Ryzen 3400G for the GPU and CPU cases respectively. Figure 3.11 shows the execution time required to evaluate the accelerations of 10,000 randomly distributed position data for gravity models with increasing numbers of parameters. The PINN performance is compared first to the spherical harmonic representation, and then to the polyhedral gravity model. The spherical harmonics representations are each generated from the EGM-2008 model with different truncation degrees. The polyhedral models tested use increasingly degraded shape models of 433-Eros generated using Blender[2] . The two analytic representations are written in Python, just-in-time compiled using Numba[3] , and executed on a Ryzen 3400G CPU.

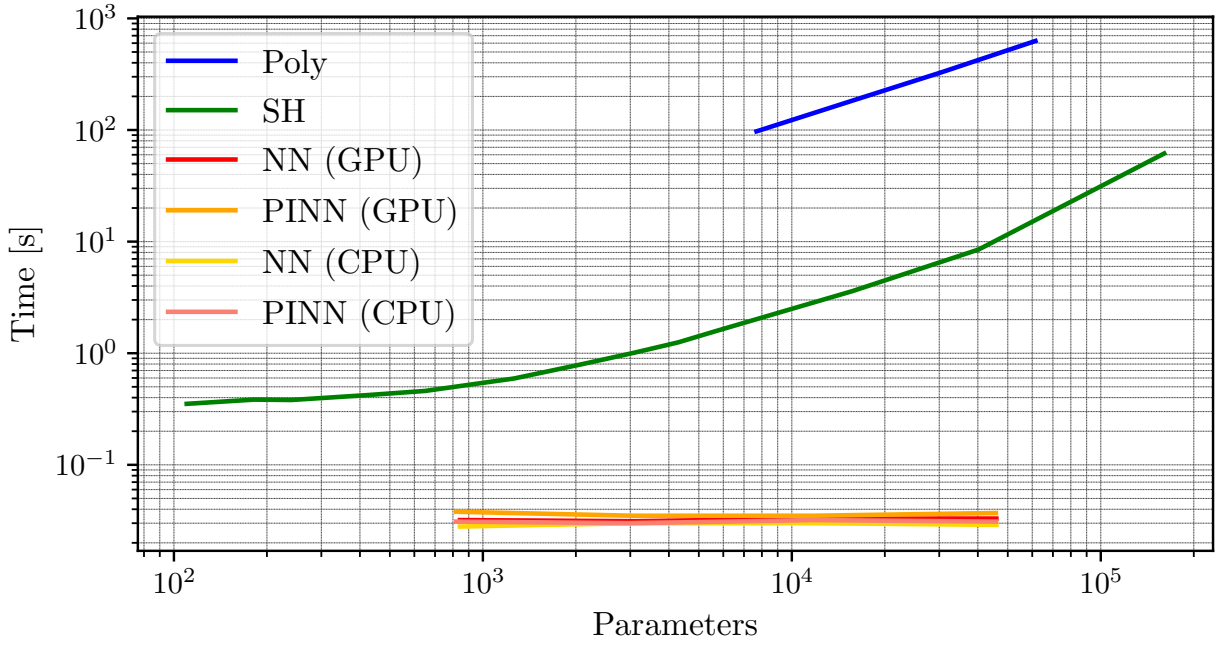


Figure 3.11: Total evaluation time to evaluate 10,000 random data using the various gravity models

[2] https://www.blender.org/
[3] https://numba.pydata.org/

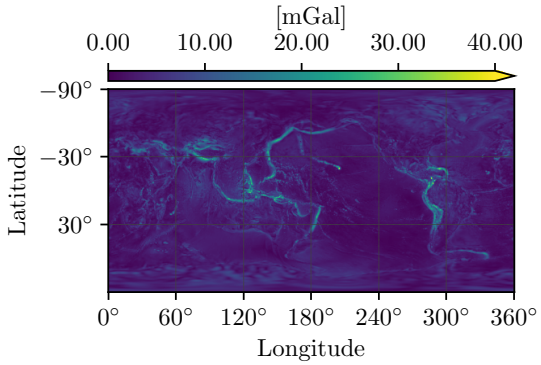
Figure 3.11 demonstrates that the spherical harmonics representation is relatively efficient to evaluate at low degree, but it also has the steepest gradient as more parameters are included in the model. This verifies spherical harmonics $\mathcal{O}(n^2)$ complexity. The polyhedral representation is by far the most time consuming to evaluate — taking nearly two orders-of-magnitude longer than that of a spherical harmonic model equipped with the same number of parameters. The neural network representation run either on the GPU or CPU is considerably more efficient than both of these representations — with performance that is nearly an order of magnitude more efficient than the lowest spherical harmonic model tested ($l = 10$), independent to the number of parameters by these networks. These results are encouraging both for use within simulation and on-board spacecraft, producing high-fidelity estimates of the gravity field that can be used for trajectory design or on-board control purposes.

3.1.6 Network Performance Applied to the Moon's Gravity Field

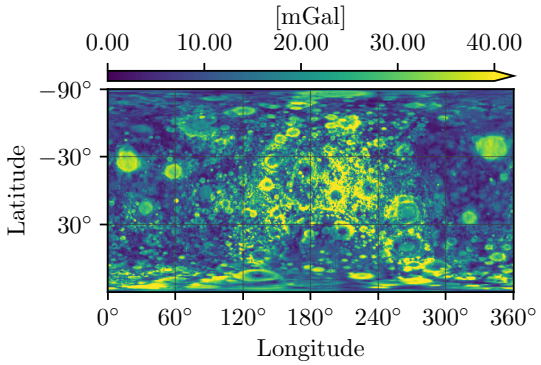
The experiments presented thus far are focused specifically on the Earth, but the conclusions on model compactness and generalization are not necessarily universal to any celestial body whose gravity field is traditionally represented by spherical harmonics. To demonstrate this, the prior experiments are repeated for a body with characteristically different perturbations: the Moon.

The Moon offers an interesting point of comparison to the Earth, as the gravitational perturbations of the Moon are substantially more frequent and of larger magnitude. The Earth's perturbations are typically generated by large, infrequent, and localized geologic structures (mountains, tectonic plate boundaries, etc). The Moon's perturbations, in contrast, are generated by craters and associated mascons which cover most of its surface. As will be shown in upcoming experiments, the complexity of the Moon's gravity field makes efficient modeling of such perturbations more challenging for the PINN-GM-I.

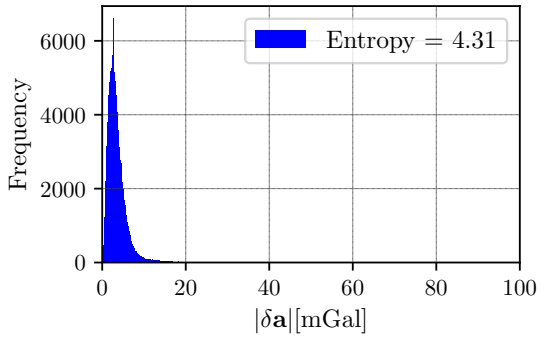
Figure 3.12 aims to contrast the differences between the two bodies and their gravity fields. Qualitatively, note how much simpler the Earth's gravity field is compared to that of the Moon. The craters on the Moon not only form a near random-surface topology, but also they intermingle



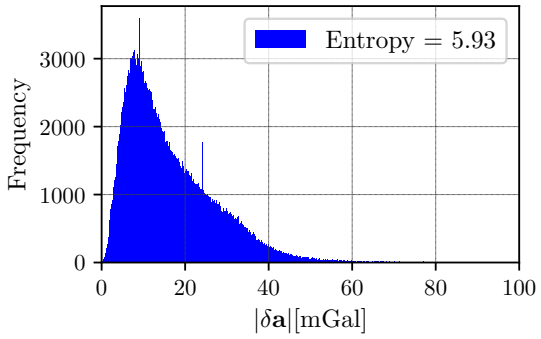
(a) Brillouin Sphere of Earth



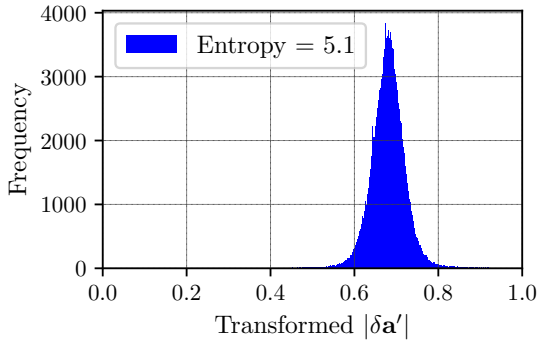
(b) Brillouin Sphere of Moon



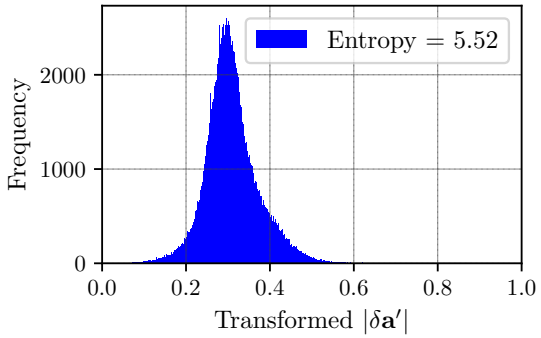
(c) Earth Perturbing Magnitudes



(d) Moon Perturbing Magnitudes



(e) Earth Perturbing Magnitudes (Scaled)



(f) Moon Perturbing Magnitudes (Scaled)

Figure 3.12: Contrasting gravity field and acceleration distributions of the Earth and Moon

some of the strongest perturbations with the weakest. The Earth's perturbations, in comparison, are much more well-behaved. The complexity of the two fields is quantified by estimating the Shannon entropy of the acceleration distributions shown in Figures 3.12c and 3.12d. For the Earth, the field entropy is 4.31 nats (unit of information expressed in base e) whereas the entropy for the Moon is 5.93 nats. This suggests that the Moon's gravity field contains approximately five times more information on average than that of Earth's gravity field — a significantly more challenging modeling task for the neural networks. These distributions are preprocessed before using them as training data for networks, so Figures 3.12e and 3.12f show how the distributions change once having applied the uniform min-max transformation to the acceleration vectors as detailed in Sect. 3.1.1. After the preprocessing, the Moon's gravity field still contains more information compared to the Earth, albeit by only 50%. This suggests in both the raw and preprocessed form, the Moon's gravity field is the more challenging modeling task.

Compactness

The PINN-GM-I model accuracy is evaluated as a function of parameters and compared to the spherical harmonic gravity model in Figure 3.13. The figure replicates the same experiment of Sect. 3.1.3, evaluating error on just the dominate gravitational perturbations, $\mathcal{F}$, as well as the average and background data ($\mathcal{A}$ and $\mathcal{C}$). Instead of training the networks from field points drawn uniformly between 0-420 km altitude, the training data spans between a 0-50 km altitude. This narrower training domain attempts to reflect the fact that spacecraft can orbit at lower altitudes around bodies like the Moon given its very thin atmosphere. In addition, the true potential, $U^{\text{SH}}_{\text{Truth}}$ is generated using the lunar GRGM1200A gravity field model (54).

The spherical harmonic gravity model is considerably more efficient at representing the Moon's gravity field than that of the Earth as shown in Figure 3.14. For the Earth, nearly 10,000 spherical harmonic coefficients are required to begin converging on the feature set, $\mathcal{F}$, whereas the Moon requires mere 300 spherical harmonic coefficients. This suggests that long wavelength harmonics play a greater role in representing the Moon's gravitational features, which is surpris-

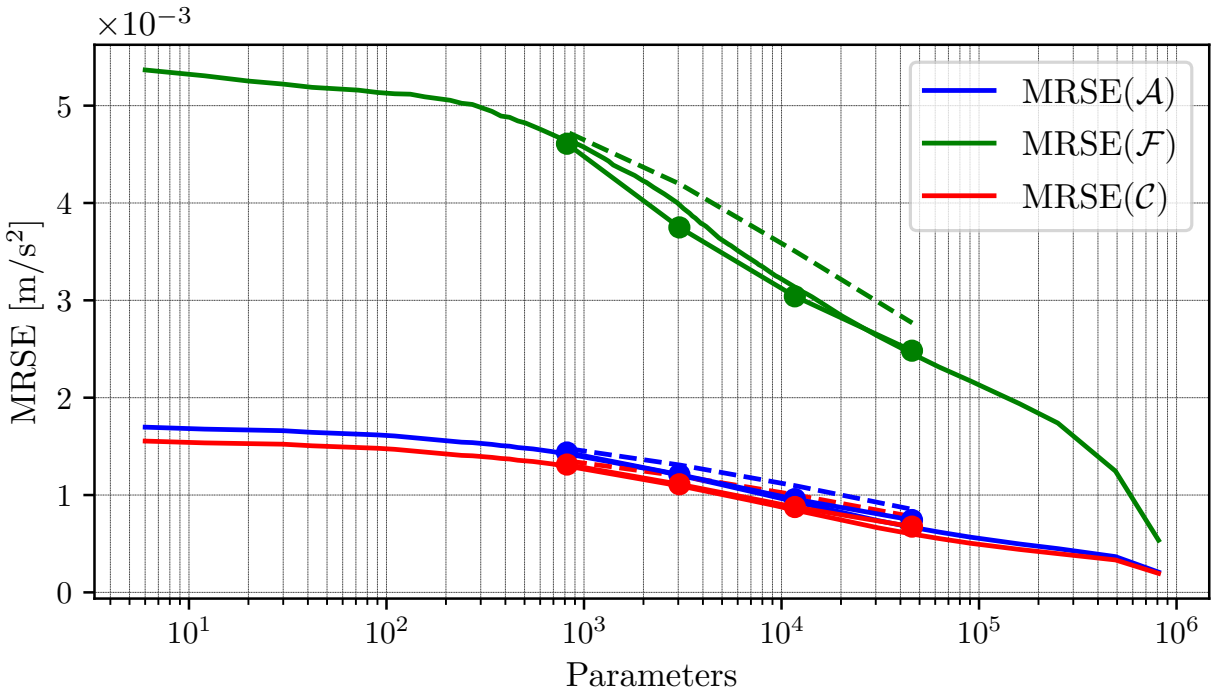
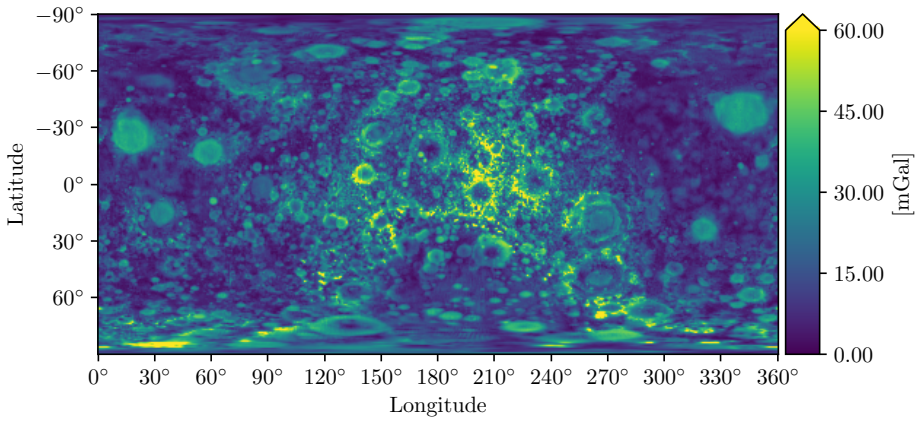
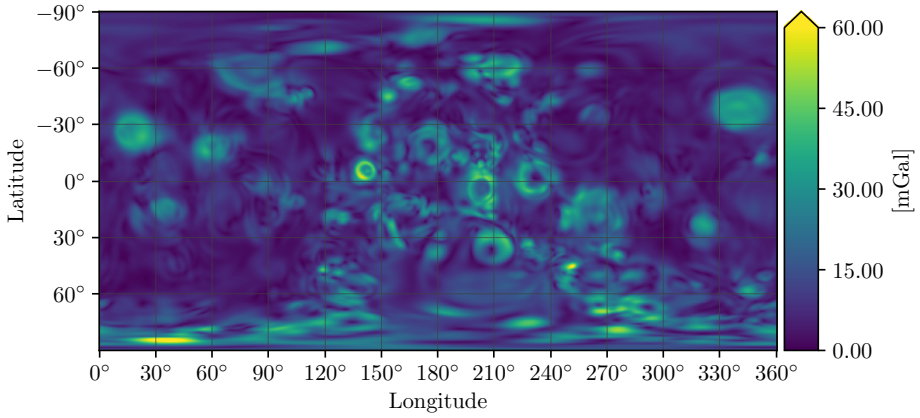


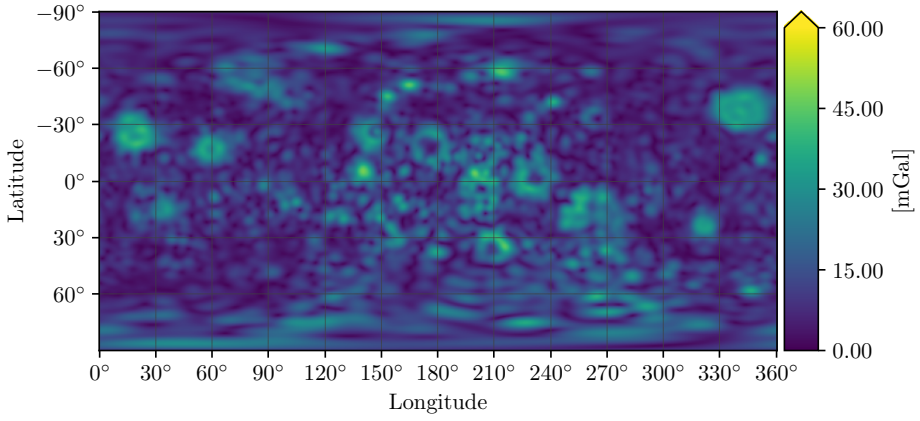
Figure 3.13: Plot of MRSE as a function of total model parameters for the Moon. Dashed lines represent traditional neural networks. The lines with circle markers represent the physics-informed neural networks



(a) True Field



(b) PINN $N = 20 \Leftrightarrow p = 3,040$



(c) SH $l = 55 \Leftrightarrow p = 3,080$

Figure 3.14: Gravity model of Moon using (a) the full $l = 1,000$ spherical harmonic model, (b) the PINN representation with $p = 3,040$ and (c) the low fidelity $l = 55 \Leftrightarrow p = 3080$ spherical harmonic representation

ing given the sharp discontinuous features present in the Moon's field. The efficiency of these low-degree harmonics is attributed to the near isotropic nature of the perturbations. Even if the low-degree harmonics cannot efficiently reduce the error of a single discontinuous feature like a crater, they do contribute a small amount to all of the craters covering the entire surface. This collective contribution generates a measurable reduction in error seen within the feature set, $\mathcal{F}$.

That said, the PINN-GM-I still offers a slight advantage over spherical harmonics in terms of model compactness, producing comparable gravitational estimates using approximately 30% fewer parameters than spherical harmonics at best. The relatively small performance gain is an interesting result attributed to two factors: First is the aforementioned efficiency of the long-wavelengths in the spherical harmonic representation. The analytic model is simply more effective for the Moon than it was for the Earth, so the networks are competing with a more productive representation.

The second contribution to the performance discrepancy is the greater complexity / higher entropy of the Moon's gravity field. The networks are being tasked to generate a distribution that contains more information than was present in the Earth's gravity field — i.e. the networks have to work harder. It remains possible that greater network performance exists, but to witness that performance the networks would likely require additional training data, additional feature engineering, and longer training times to compensate for this more complex problem.

Future work is required to determine if there is a more fundamental relationship between the entropy of a gravity field and the capacity for a successful neural network gravity model. Because environments with high gravitational entropy will generate accelerations that appear unstructured or random, it is possible that all basis sets (learned or analytic) will struggle equally to model these types of fields.

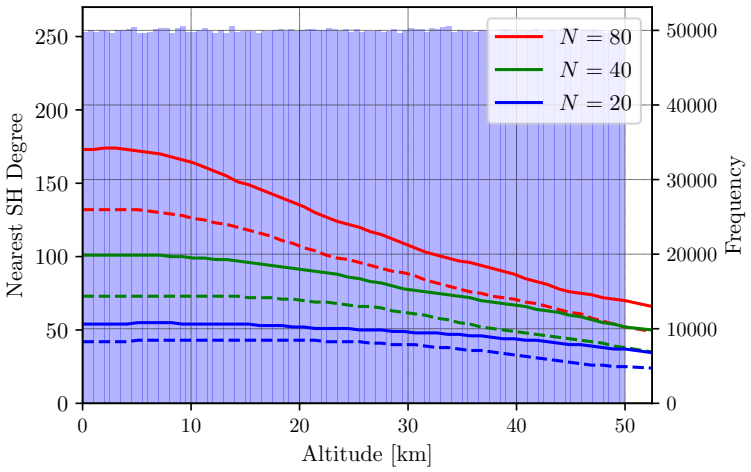
Generalization

Following the same procedure as presented in Sect. 3.1.4, the PINN-GM-I performance is tested at varying altitudes to determine how these networks generalize to different orbits. The altitudes tested vary between 0-55 km and the MRSE for each dataset of $\mathcal{A}_m, \mathcal{F}_m$ and $\mathcal{C}_m$ is shown

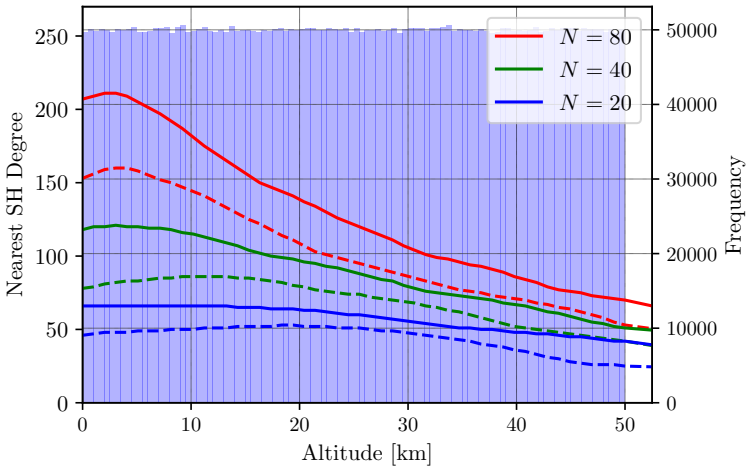
in Figure 3.15.

The general performance of the PINN-GM-I for the Moon closely mimics that of the Earth networks. Both the traditional and physics-informed neural networks tend to have greatest accuracy at low-altitudes, again highlighting the intrinsic bias of these models to low-altitude samples. That said, there is a less pronounced performance peak for the PINN-GMs representing the Moon than those of the Earth. This difference attributed to the Moon's high-entropy gravity field. The field's greater complexity ultimately acts as a type of regularization that helps prevent the networks from overfitting to the training data. The perturbations are so diverse and ever-present that the networks do not develop an overconfidence in their learned solution. Analogous effects are seen when using small mini-batch sizes to train neural networks. These small batches can cause a noisier gradient descent which takes longer to converge but assists the optimizer in exploring the cost landscape and ultimately can produce more robust solutions (86). The difference here is instead of decreasing the batch size, the complexity of the cost landscape is increased to achieve a similar effect.

In summary, the PINN-GM-I demonstrates some advantages over spherical harmonics when representing the gravity fields of large celestial bodies like the Earth and Moon. Among the most compelling of these advantages is the rapid executability of these models over their analytic counterparts, as well as their smaller model sizes and memory footprints.



(a) MRSE($\mathcal{A}_m$)



(b) MRSE($\mathcal{F}_m$)

Figure 3.15: MRSE of $\mathcal{A}_m$ (top) and $\mathcal{F}_m$ (bottom) for the traditional (dashed) and physic-informed (solid) neural networks converted into the equivalent spherical harmonic degree as function of altitude for the Moon. The blue histogram represents the training data distribution

3.2 Generation II

While the results of the PINN-GM-I provide an encouraging start for the use of PINNs to represent gravity fields of large planetary bodies, there remain open questions regarding the capabilities and limits of the PINN gravity model particularly in the context of small-body exploration. For example, are these network representations equally robust when modeling the gravity field of non-spherical celestial bodies? Do the networks continue to generate accurate potentials and accelerations once inside of the Brillouin sphere? What training data conditions are necessary to achieve a robust model, and how sensitive are these networks to noise in the training data?

In addition, little work has been done discussing how these networks can be designed and tuned for increased performance. In Reference (87), only a single physics constraint is used in the cost function ($\boldsymbol{a} = -\nabla U$), however other constraints can be introduced. Do such constraints assist in generalization or does the multi-objective nature of the additional constraints impede the learning process? Does the densely connected architecture of the network preclude the ability of the networks to learn abstract symmetries and conservation laws as suggested in Reference (81)?

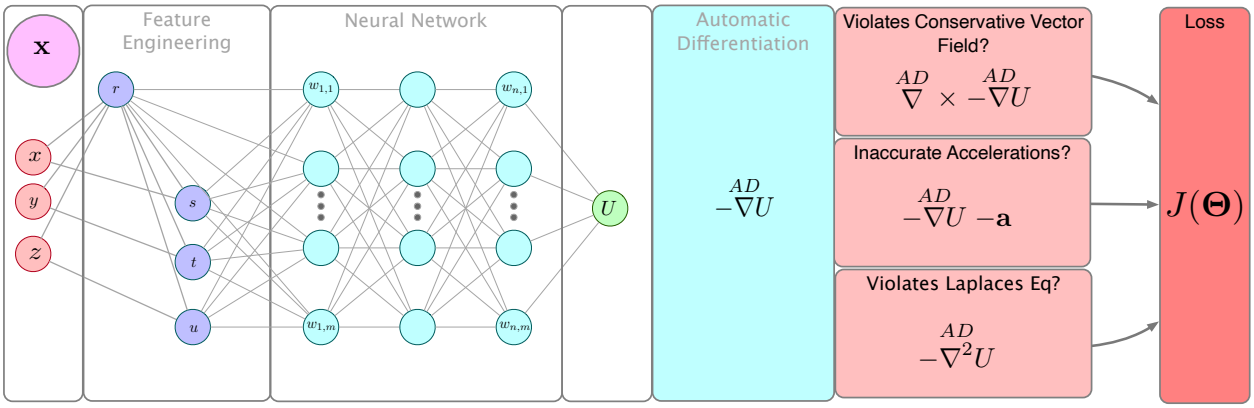


Figure 3.16: PINN-GM Generation II (1)

This section introduces the PINN-GM generation II, or PINN-GM-II, which proposes a collection of new design modifications to the original PINN-GM that aim to addresses these questions. Specifically, this section focuses on small-body gravity field modeling, exploring how the PINN-GM can be designed to produce high-fidelity gravity field models for the asteroid 433-Eros. The model

design changes of the PINN-GM-II and their corresponding effect on performance are detailed below.

3.2.1 Normalization

In many machine learning applications, the data used to train neural networks are normalized to ensure that the weights and biases learned by the network are not of exceedingly large magnitude, with the goal of avoiding numerical issues (88). In the first generation PINN-GM, the training data is normalized via a min-max transform on each acceleration component such that all training data existed between -1 and 1. While effective, this approach does not actually guarantee numerical stability for the network output. Normalizing the position and acceleration to [-1,1] provides no guarantees regarding the range of the potential, the actual output of the neural network. To ensure that the numerical stability of the network output is prioritized, PINN-GM-II introduces a different non-dimensionalization through:

$$\boldsymbol{x}^\star = \frac{\boldsymbol{x} - x_0}{x_s} \tag{3.9}$$

$$U^\star = \frac{U - U_0}{U_s} \tag{3.10}$$

$$\boldsymbol{a}^\star = \frac{x_s}{U_s}\frac{dU}{d\boldsymbol{x}} \tag{3.11}$$

$$\tag{3.12}$$

where x_0 corresponds to the minimum value of all position components and x_s corresponds to the range of the position vector. U_0 and U_s are the minimum and range of the potential.

3.2.2 Feature Engineering

In PINN-GM-I, the networks' inputs are position data represented in cartesian coordinates. While this approach is not inherently problematic, the domain of each unscaled feature (x, y, z) extends from $(-\infty, \infty)$, forcing the network to accommodate a substantially wider feature space than if represented in spherical coordinates. PINN-GM-II changes this construct by instead transforming the PINN inputs into a 4D spherical coordinate set of r, s, t, and u before passing them

through the network. These variables are defined as:

$$r = \sqrt{x^2 + y^2 + z^2}$$
$$s = \frac{x}{r}$$
$$t = \frac{y}{r}$$
$$u = \frac{z}{r}$$

where s, t, and u represent the tangents of angles between the field points and each of the cartesian unit vectors $\hat{x}$, $\hat{y}$, and $\hat{z}$ (84).

This 4D spherical coordinate set is chosen over the more traditional 3D set (r, θ, and ϕ) for multiple reasons. First, the values of s, t, and u naturally exist between the desirable bounds of $[-1, 1]$ whereas the θ and ϕ coordinates of the 3D set have discontinuity due to angle wrapping. Second, the 4D coordinate set avoids a singularity in the gradient of the potential at the poles. This can be seen when evaluating the 3D spherical gradient at $\phi = -90^{\text{o}}$ and 90^{o} as seen in:

$$\nabla U = \frac{\partial U}{\partial r} + \frac{1}{r}\frac{\partial U}{\partial \phi} + \frac{1}{r\cos\phi}\frac{\partial U}{\partial \theta} \tag{3.13}$$

This feature engineering that converts the cartesian position into the 4D spherical coordinates is performed within the official Tensorflow graph as a prepended layer to the neural network model. By integrating this transformation into the model, this means the potential can still be automatically differentiated with respect to the cartesian inputs. This construct provides the best of both coordinate sets: a neural network that trains on a significantly reduced feature space (increasing network convergence and accuracy) while maintaining the derivatives in a convenient cartesian basis.

3.2.3 Additional PINN Constraints

In addition to modifying the input features, PINN-GM-II introduces a modified loss function which include additional physics-informed constraints to improve modeling accuracy and robust-

ness. Explicitly, the loss function proposed for the PINN-GM-I only penalized errors in the predicted acceleration (as denoted by the subscript A in Eq. 3.3). While effective, this loss function fails to account for other dynamical properties that must be observed by the gravity model. For example, the force of gravity is a conservative force — therefore the scalar potential learned by the network must also obey these additional physics properties:

$$\nabla^2 U = 0 \qquad \text{Laplacian (L)} \tag{3.14}$$

$$\nabla \times \boldsymbol{a} = 0 \qquad \text{Curl (C)} \tag{3.15}$$

These conservative vector field properties can be embedded into the original PINN loss function through:

$$J_{\text{ALC}}(\Theta) = \frac{1}{N_f} \sum_{i=1}^{N_f} \left| \boldsymbol{a}_i + \overset{\text{AD}}{\nabla} f(\boldsymbol{x}_i|\Theta) \right|^2 + \left| \overset{\text{AD}}{\nabla}{}^2 f(\boldsymbol{x}_i|\Theta) \right|^2 + \left| \overset{\text{AD}}{\nabla} \times \overset{\text{AD}}{\nabla} f(\boldsymbol{x}_i|\Theta) \right|^2 \tag{3.16}$$

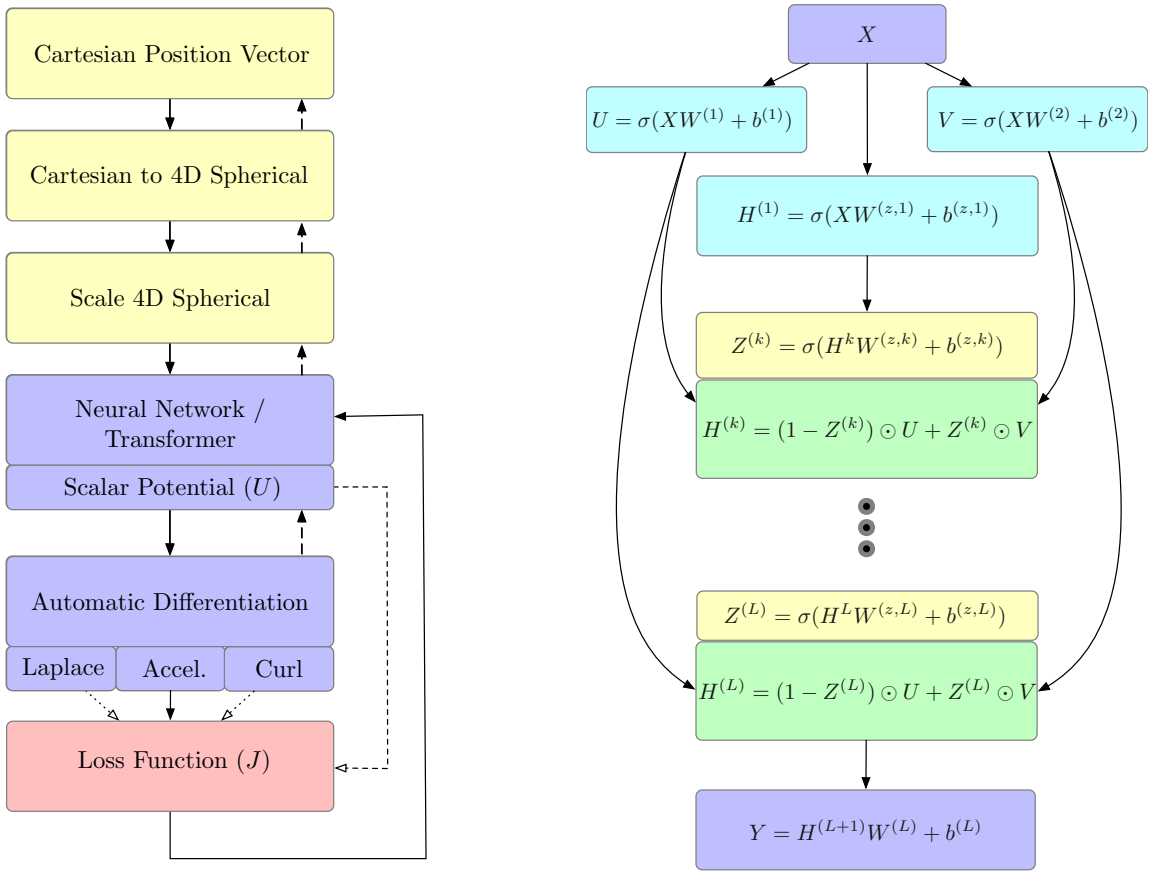
In addition, if a model of the gravitational potential exists, the loss function can be extended to penalize those modeling errors (subscript P):

$$J_{\text{APLC}}(\Theta) = J_{\text{ALC}}(\Theta) + \frac{1}{N_f} \sum_{i=1}^{N_f} |U_i - f(\boldsymbol{x}_i|\Theta)|^2 \tag{3.17}$$

These additions to the cost function should act as a form of regularization during training which can help improve robustness, particularly in the presence of noise.

3.2.4 Modified Network Architectures

Beyond incorporating more physics constraints into the network loss function, the PINN-GM-II also introduces an alternative network architecture. In Reference (81), it is observed that the success of many modern machine learning models rely on embedding symmetry groups or invariances into the network architectures (be it translational invariances in convolutional networks or temporal invariances in recurrent neural networks). Embedding these properties into the network architectures for physical processes can be challenging as it assumes preexisting knowledge of these symmetry groups which may not be present. For this reason, it is common to rely only on fully connected networks in modeling physical processes as opposed to more advanced architectures.



(a) Modified PINN Gravity Model (v2)

(b) Transformer-Inspired Architecture

Figure 3.17: Modified Network Structure

Reference (81) implies that this choice may not be sufficient, as there does not yet exist a proof of the convergence of fully connected Physics-Informed Neural Networks. By consequence, researchers can only hope that fully connected networks provide sufficient modeling capacity and flexibility to represent the solution to the corresponding differential equation. Reference (81) supplement this claim by introducing how a relatively simple change in network architecture can significantly improve the accuracy of the learned model in a collection of archetypal PINN problems. This work adopts the proposed architecture, by extending the fully connected network with two encoders, which project the inputs into a high-dimensional space. In addition, the hidden states are enhanced with residual connections from prior layers through multiplicative, element-wise interactions as shown in Figure 3.17b. These modifications introduce relatively minor changes to the memory and computational footprint of the fully connected network but offer appreciable gains in model accuracy. This architecture change will be referred to as the Physics-Informed Transformer (PIT) for the remainder of this section.

3.2.5 Performance

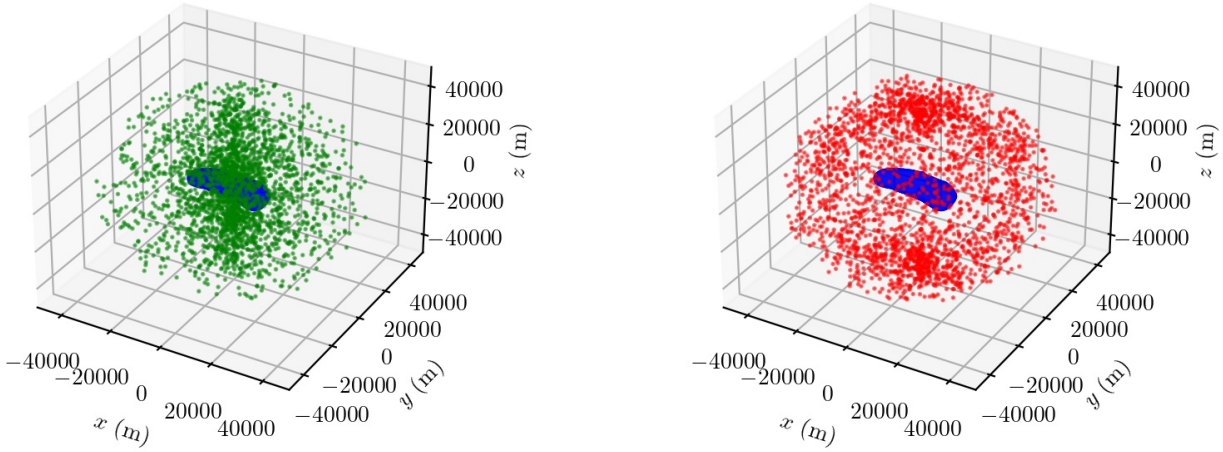
To investigate the effect of these design changes on performance, a collection of experiments are introduced. These experiments are performed by sequentially adding the various design changes introduced above and characterizing the model performance at each intermediate stage.

To begin, baseline metrics are established to assess how well the PINN gravity model is able to learn the gravity field of asteroid 433-Eros from position and acceleration estimates. These experiments are designed assuming that a pre-existing high-fidelity gravity model exists (be it spherical harmonics, polyhedral, mascon, or other) from which perfect or noisy measurements can be drawn and used as training data to generate the PINN representation. For these experiments, the polyhedral representation is assumed to be ground truth and makes use of the Glaskell[4] shape model. Three unique datasets are proposed to characterize the effect of data quantity and distribution on model performance.

[4] https://arcnav.psi.edu/urn:nasa:pds:gaskell.ast-eros.shape-model

Uniform Distribution $0 - 3R$ (r)

The first training distribution samples data uniformly from the surface of the asteroid to a radius equal to three times the maximum radius of the asteroid, R (Figure 3.19a). This sampling represents the best case scenario — where the network is trained on data that spans the entire problem domain (within, at, and above the Brillouin sphere).



(a) r training distribution: Full coverage

(b) $r^\star$ training distribution: Safe altitudes

Figure 3.18: Training Distributions Asteroid 433-Eros

Uniform Distribution $2R - 3R$ $(r^\star)$

The second training distribution reflects more realistic circumstances — where samples are collected uniformly, but only from a range of safe altitudes above the asteroid (2-3 times the maximum radius). Such distribution is shown in Figure 3.19b. These altitudes are generally favorable for estimating low-degree spherical harmonic models or collect images that can be used to resolve a shape model for use in a polyhedral gravity model.

Hybrid Distribution $2R - 3R$ plus Sparse Low-Altitude Samples ($\bar{r}$)

The final training distribution forms a hybrid set from the prior two distributions. The majority of the samples are collected from the $2R-3R$ ($r^\star$) distribution, but 10% are drawn between the surface and $2R$. These sparser samples could represent two potential phenomena: active asteroid ejecta or advance mission concepts like gravity poppers. As recently found in the OSIRIS-REx radio science experiment, some asteroids eject particles which can be tracked using optical measurements and their trajectories estimated to be used in gravity field recovery (89; 90). These particles greatly enhanced the resolution of the original radio science returns of OSIRIS-REx and has consequently motivated a new mission technology called gravity poppers (91). Gravity poppers are a technology demonstration which deploy artificial, uncontrolled probes from a mothercraft onto the surface of a small-body. Those probes then hop off of the surface of the asteroid, and the resulting dynamics are used to assist in estimating the gravity field of the body.

Experimental Setup

These experiments begin by training PINN-GM-IIs with only the new normalization and feature engineering modifications applied. 5,000 training data are sampled from each of the aforementioned distributions and used to train a corresponding PINN gravity model for 7,500 epochs. This process is then repeated with artificial error added to the acceleration measurements to generate a more representative, noisy dataset as might be produced by an orbit determination process using dynamic model compensation. These noisy estimates are generated by adding an error vector with a random orientation and magnitude equal to 20% of the true acceleration vector magnitude. The error is therefore non-gaussian where the larger the true acceleration, the larger the corresponding error. The remaining hyperparameters used to train these networks are listed in Table 3.3.

After training each model, 20,000 test samples are selected randomly from the surface of the body to an altitude corresponding to a $3R$ radius. The residuals between the true and predicted accelerations at the test locations are plotted alongside their moving average. In addition, two

Table 3.3: Nominal Hyperparameters

Parameter	Value
Activation	GELU
Batch Size	5,000
Optimizer	Adam
Epochs	7,500

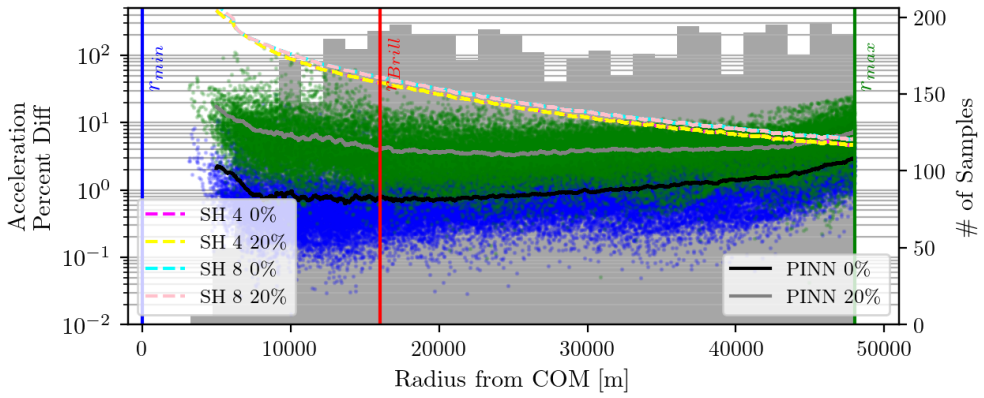
Parameter	Value
Hidden Layers	8
Nodes Per Layer	20
Weight Initialization	Glorot Normal
Learning Rate	0.002

spherical harmonic models (one of degree and order 4, the other of degree and order 8) are fit to the same 5,000 training data using least squares regression and are tested on the same 20,000 test samples. The spherical harmonic model residuals and average error are also plotted to serve as a comparison point. These curves are presented over the corresponding training data distribution histograms in Figure 3.19.

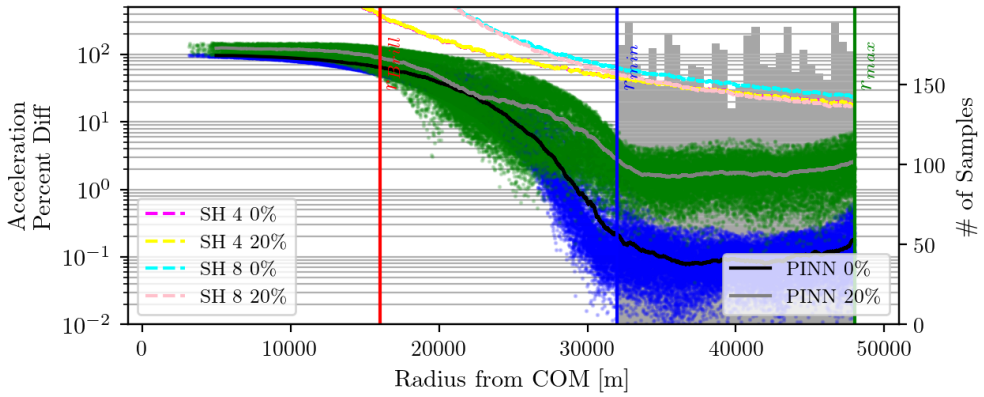
Baseline Results

In the best case scenario (r without noise), the PINN models produce accelerations with less than 3% average error all the way down to the surface — an encouraging feat given that the network was exposed to only 5,000 data points. When the 20% acceleration error is introduced into the estimates, the PINN's error also increases, but remains below an average of 10%. The regressed spherical harmonics models, in contrast, never achieve better than 5% average error — independent of amount of error in the estimates — and can be seen diverging as they approach lower altitudes.

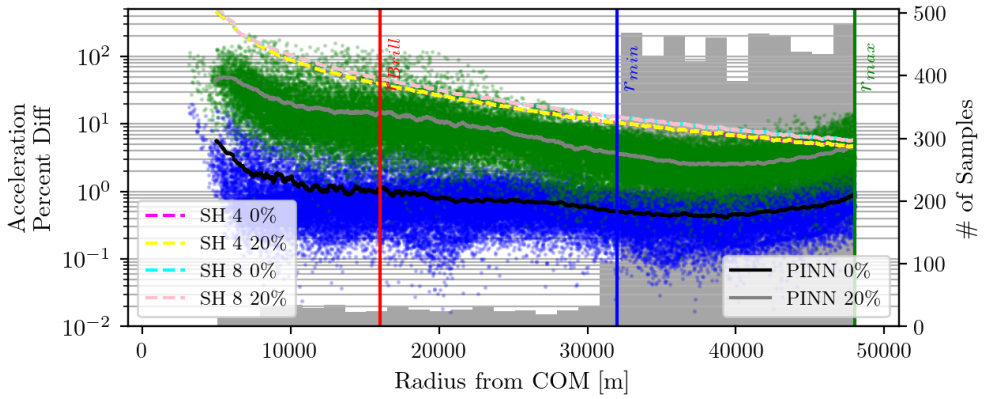
In the case of training PINNs on $r^\star$, the learned representations achieve $< 0.1\%$ average error within the training domain with perfect measurements and $< 3\%$ when the measurements are noisy. However, the PINNs do suffer from extrapolation error beyond these bounds. While this behavior is undesirable, it is not unique to the PINN models. The regressed spherical harmonic models also suffer from greater extrapolation error. Moreover, even within the training domain, spherical harmonics models generate higher error than the PINNs. For both PINNs and spherical harmonics,



(a) Training distribution r and corresponding model errors.



(b) Training distribution $r^\star$ and corresponding model errors.



(c) Training distribution $\bar{r}$ and corresponding model errors.

Figure 3.19: Acceleration residuals, $|\boldsymbol{a} - \hat{\boldsymbol{a}}|/|\boldsymbol{a}| \times 100$, their moving averages, and training data distribution as a function of radius. Blue scatter plots correspond to error of the PINN trained on perfect measurements, and green corresponds to the PINN trained on noisy measurements. Dashed lines represent the error of the spherical harmonic models fit on the same data. The gray histogram represents the radial distribution of the training data.

these results suggest that generalizable models require training samples from lower altitudes if they are to be used in close proximity to the body.

To investigate how many low altitude samples are necessary, the $\bar{r}$ distribution supplements the $r^\star$ distribution with 500 additional samples collected between the asteroid surface and $2R$ radius. By including this relatively small set of measurements, the PINN modeling accuracy stabilizes significantly. When no error is included within the training data, the average error of the PINN remains $< 1\%$ for the majority of altitudes. Only at the surface of the body does the PINN model achieve an average modeling error of 5%. When error is added to the acceleration training data, the modeling accuracy decreases, but the total error remains below 10% for the majority of altitudes and only reaches a maximum average modeling error of 40% when evaluated at the surface — both sizable improvements in accuracy compared to the networks trained only on $\bar{r}$. This experiment is particularly encouraging as it implies that PINNs are quite efficient at extracting information about the gravity field when given access to sparse low-altitude samples. Such findings align with the findings of PINN-GM-I and provide strong evidence in favor of concepts like gravity poppers. Given how little data is needed to achieve generalizable models, there may not be a need for many gravity poppers or even poppers that can produce diverse trajectories.

Architecture and Loss Experiment

To evaluate how additional physics constraints and modified network architecture improve model accuracy, another experiment is performed which trains the PINNs and PITs with different cost functions on increasingly sparse datasets ($N = \{2500, 1250, 625\}$) comprised of position / acceleration pairs that are sampled randomly from $0 - 3R$. Once trained, the modeling error of these PINN and PIT representations are then evaluated on separate test datasets comprised of 20,000 randomly distributed position / acceleration pairs. The first test set contains 20,000 positions / acceleration pairs sampled outside of the Brillouin sphere and within $3R$ of the body (exterior), the second test set is sampled between the surface of the asteroid and the Brillouin sphere (interior), and finally the third set samples positions / accelerations from 20,000 randomly

selected facets of the polyhedral shape model (surface). These three test dataset regions are shown in Figure 3.20. Each network model is trained using only one of the cost functions (J_A, J_{AP}, J_{ALC}, J_{APLC} and a non-physics-informed cost function, denoted J_{00} or

$$J_{00} = \frac{1}{N}\sum_{i=0}^{N} |\boldsymbol{a}_i - \hat{\boldsymbol{a}}_i| \tag{3.18}$$

In addition, each of the models is trained on datasets with increasing levels of error added to the acceleration data. The erroneous acceleration estimates are generated in the same manner as presented in Sect. 3.2.5. The percent errors tested are 0%, 10%, and 20% of the true acceleration magnitude. Three spherical harmonic gravity models (corresponding to degree $l = \{4, 8, 16\}$) are also regressed on each training dataset using least squares and evaluated using the same test datasets.

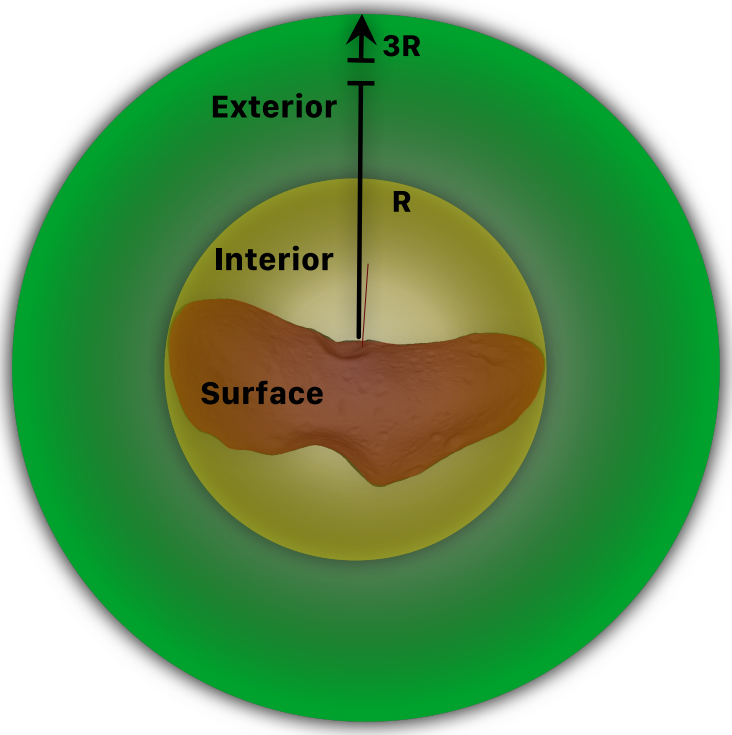


Figure 3.20: Eros Data Distributions

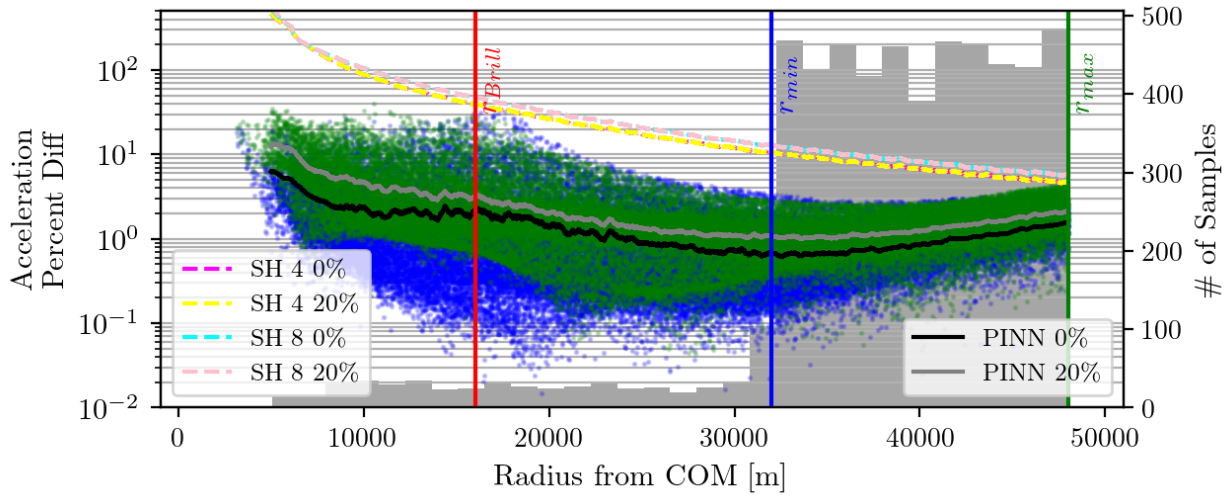


Figure 3.21: Acceleration residuals as a function of training distribution with J_{ALC} demonstrate how additional physics constraints help desensitize the model to noise in the training data

The performance of the PINNs, PITs, and spherical harmonics gravity models are plotted as a function of amount of training data, N, and error in the training data (0%, 10%, and 20%). The corresponding results for each test dataset (exterior, interior, and surface) are plotted in Figure 3.22, Figure 3.23, and Figure 3.24 respectively. Note that the model errors are quantified as acceleration percent error distributions, and they are represented as box-and-whisker plots. The box represents the 1st and 3rd quartiles (25% to 75%) of the percent error distribution, i.e. 50% of data lies within the box. The line within the box corresponds to the median or 2nd quartile. The lines extending beyond the box correspond to $1.5 \times IQR$ where IQR is the interquartile range, and the points beyond those lines correspond with outliers. Also, Figure 3.22, Figure 3.23, and Figure 3.24 cluster the PINN, spherical harmonics (SH), and PIT models into different color groups. As such, the rows within each cluster of the legend correspond to the models in the figure plotted from left to right.

Figure 3.22, 3.23, and 3.24 demonstrate a number of interesting behaviors of the PINN and PIT gravity models. First, the additional constraints of the cost function act as a form of regularization to the regression. When there exists sufficient data ($N = 2500$) with little-to-no noise, the multiple-constraint PINNs (ALC, APLC) tend to perform worse than their simpler counterparts (00, A, AP). However, when the noise in the dataset increases, or the number of samples decrease, the multiple-constraint networks are more robust. These networks verify that the additional constraints minimize the risk of overfitting to noise or sparse data and are a more trustworthy formulation in uncertain or novel environments. This can also be seen by comparing Figures 3.19 and 3.21 and noting how the PINN ALC network trained on the noisy $\bar{r}$ distribution is functionally desensitized to the noise.

The second interesting finding from Figures 3.22, 3.23, and 3.24 is the consistent performance improvement of the PITs over the PINNs. In both the low- and high-sample regimes, the transformer-inspired architecture can offer as high as an order of magnitude reduction in error despite being trained with the same data as best illustrated in the low-to-no noise cases of Figure 3.22 and 3.23. It is worth noting, however, that the transformers are also more prone to overfitting than the PINNs due to their greater modeling capacity — as seen when using the simpler physics con-

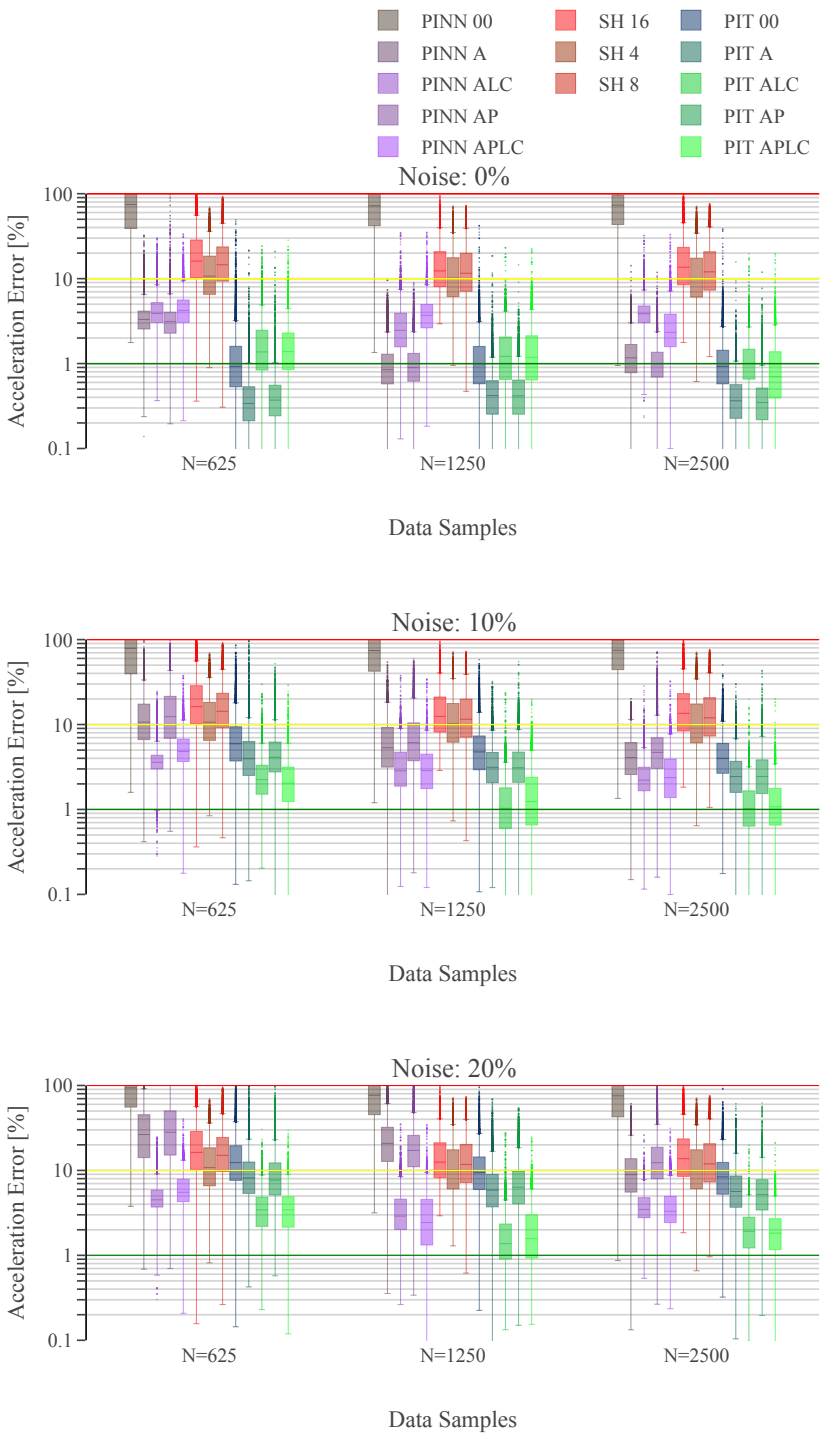


Figure 3.22: Model error **outside** the Brillouin sphere (exterior) as a function of amount of training data and noise added to the training data.

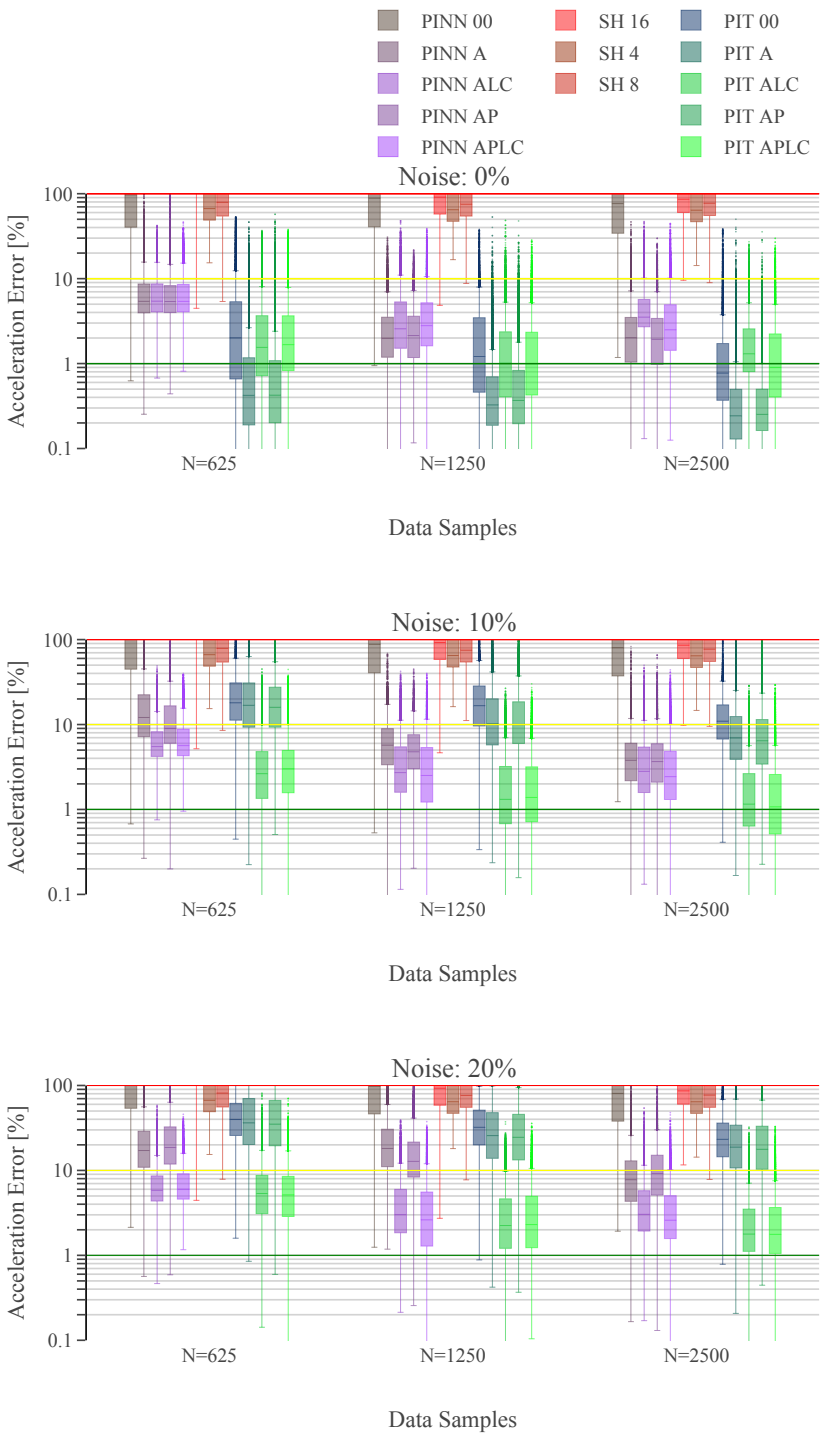


Figure 3.23: Model error **inside** the Brillouin sphere (interior) as a function of amount of training data and noise added to the training data.

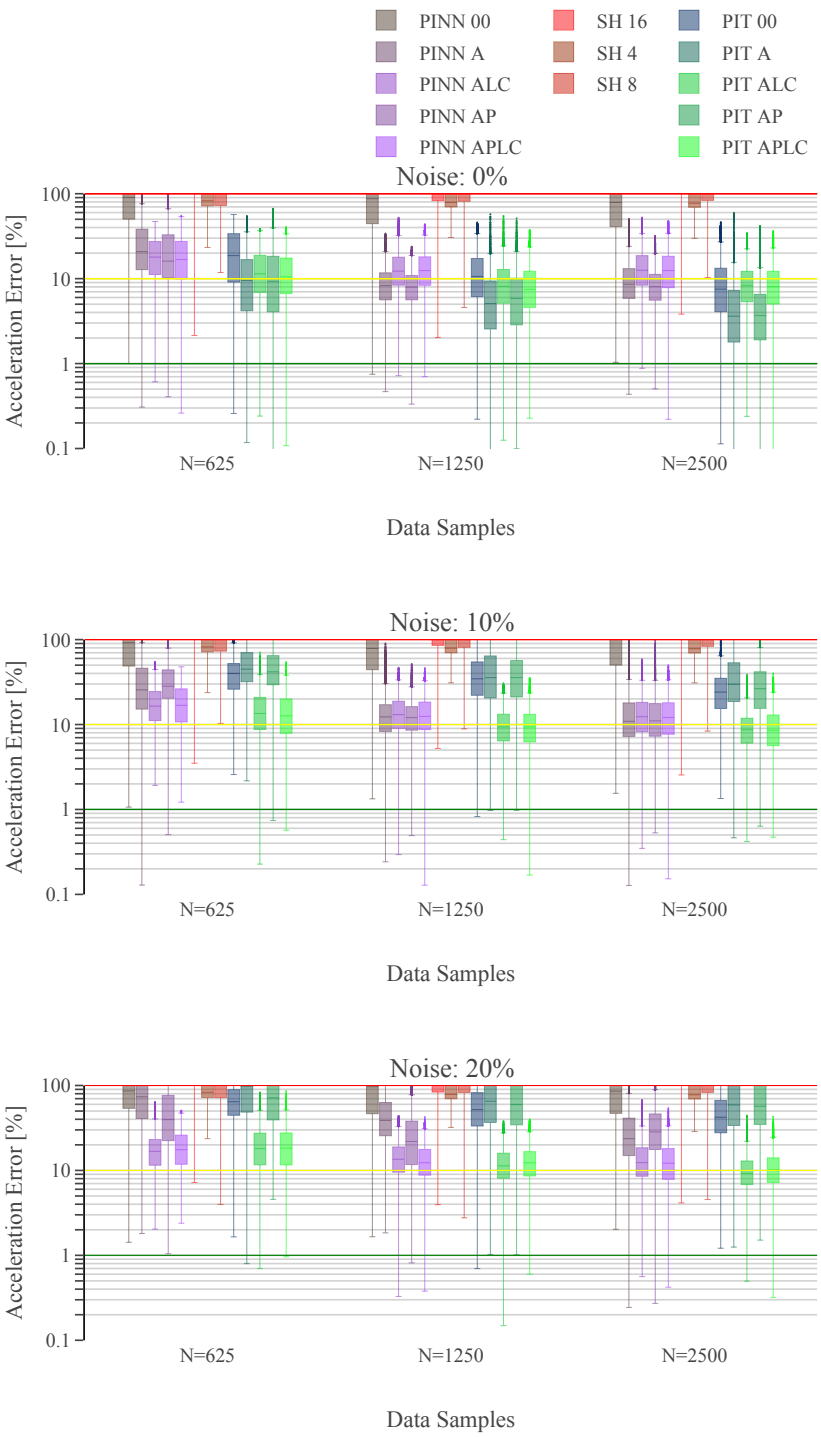


Figure 3.24: Model error **at the surface** of the asteroid (surface) as a function of amount of training data and noise added to the training data.

straints (00, A, AP) on the 10% and 20% noise cases in Figures 3.23 and 3.24. In the 0% noise case, this modeling flexibility could be considered a strength as the fewer physics constraints allow the model to converge more quickly than their more regularized variants. However, in more practical applications like when the field is not known a priori, it is recommended to use the additional constraints to minimize the risk of overfitting.

Finally, both the PINNs and PITs offer sizable performance gains over their spherical harmonic predecessor not only in the interior and surface distributions for which spherical harmonics will diverge, but also the exterior distribution where spherical harmonics has often had the analytical high-ground. This consistently worse performance of the spherical harmonic representation is a function of the small sample size, the random distribution of these samples, and the low altitude data — each capable of posing numerical challenges in the least squares estimate of the spherical harmonic coefficients.

Together the PINN-GM-II demonstrates favorable performance when applied to small-body environments. With improved feature engineering, additional loss terms, and improved network architecture, the PINN-GM-II achieves considerably better sample efficiency than its predecessor. Moreover, the PINN-GM-II demonstrates robustness to noisy data, and is successful at regressing low-altitude regimes with relatively few samples.

3.3 Generation III

The third and most recent generation of PINN gravity model (PINN-GM-III) introduces a collection of additional design modifications to address various pain points left unaddressed by past generations, while also increasing model accuracy and robustness. A visual depiction of these changes are shown in dark gray in Figure 3.25 and can be juxtaposed with the previous PINN-GM-II in Figure 3.16. All modifications are detailed in the sections below.

Model	Parameters	Training Data	Avg. Error [%]	Valid Globally
GP (45)	12,960,000	3,600	1.5%	✗
NNs (92)	1,575,936	800,000	0.35%	✗
ELMs (6)	100,000	768,000	1-10%	✗
GeodesyNet (7)	91,125	500,000	0.36%	✗
PINN-GM-III	3,048	4,096	0.20%	✓

Table 3.4: Machine Learning Gravity Model Statistics – See Appendix D

3.3.1 Preprocessing Efforts

PINN-GM-III continues on the journey to find the most productive normalization of the PINN-GM inputs. As discussed previously, one of the most common practices is to normalize the data such that they exist within the bounds $[-1, 1]$ as many activation functions exhibit their greatest non-linearity in this regime. Non-linearities are what provides neural networks their powerful approximation capabilities, and if the inputs saturate the activation functions then the network's modeling capacity is reduced. In addition, normalization also improves numerical stability during training, and decreases the risk of ill-conditioned matrix operations if the training values are too large or small.

Traditionally, the inputs and outputs of neural networks are normalized in a manner agnostic of one another, e.g. if the inputs are x and the outputs are y then normalization yields $[x_{\min}, x_{\max}] \rightarrow [-1, 1]$ and $[y_{\min}, y_{\max}] \rightarrow [-1, 1]$. In PINNs, however, the inputs and outputs may share units, and this decoupled normalization can produce non-compliant physics. For the gravity field modeling

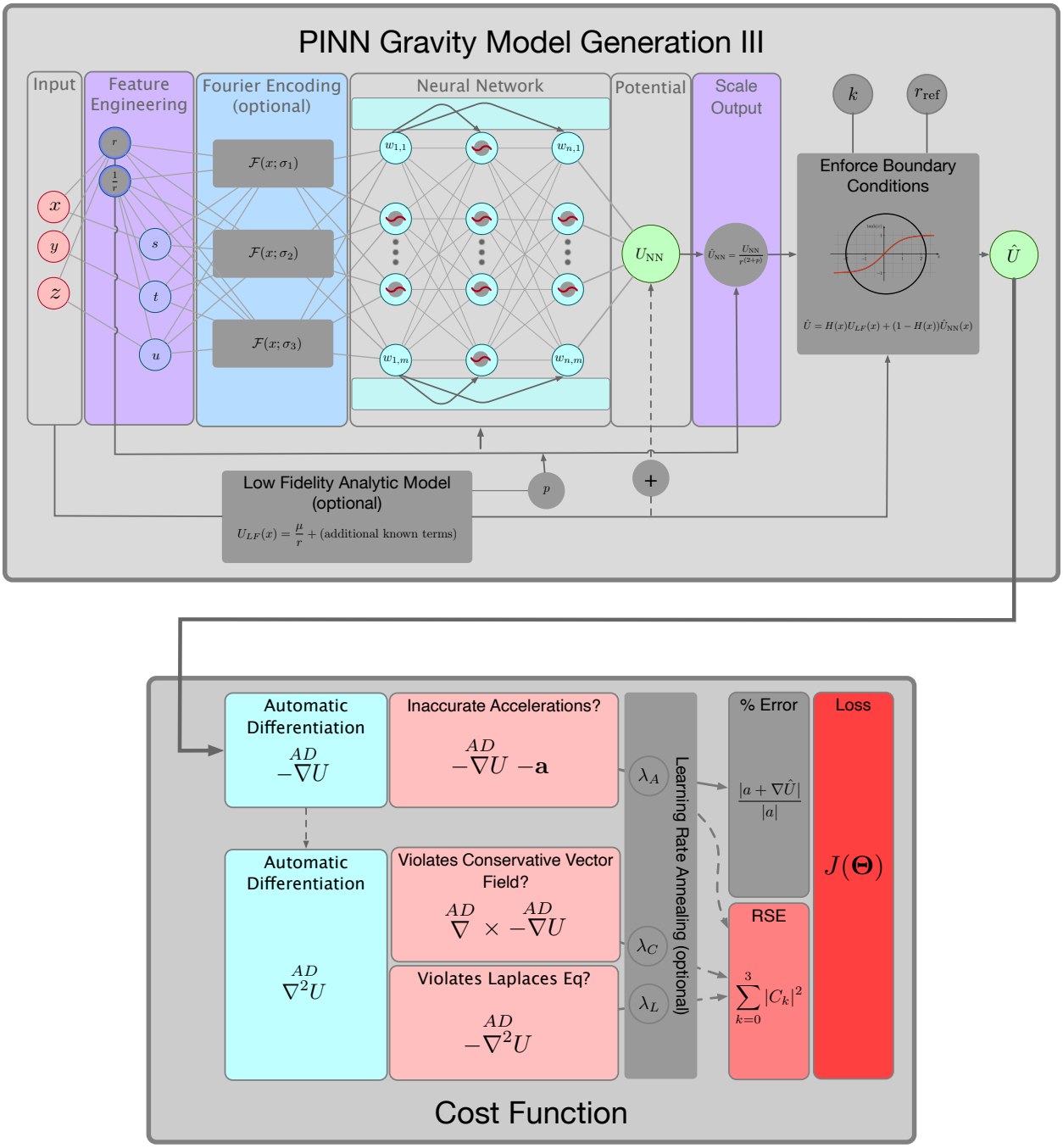


Figure 3.25: PINN-GM Generation III with new modifications contained in dark gray boxes

problem, it is therefore important to normalize the inputs and outputs in a dimensionally-informed manner. In PINN-GM-III this is accomplished by normalizing the position and potential by the characteristic length $x^\star$ equal to the planet radius R and maximum potential value in the training data $U^\star$ respectively. Using these characteristic scalars, a time constant can be computed and used in conjunction with $x^\star$ to non-dimensionalize the accelerations. Explicitly this manifests through:

$$x = \frac{\bar{x}}{x^\star}, \quad U = \frac{\bar{U}}{U^\star}, \quad a = \frac{\bar{a}}{a^\star} \tag{3.19}$$

where $x^\star$, $U^\star$, and $a^\star$ are the non-dimensionalization constants defined as:

$$x^\star = R \tag{3.20}$$

$$U^\star = \max_i(\bar{U}_i - \bar{U}_{\mathrm{LF},i}) \tag{3.21}$$

$$t^\star = \sqrt{\frac{x^{\star 2}}{U^\star}} \tag{3.22}$$

$$a^\star = \frac{x^\star}{t^{\star 2}} \tag{3.23}$$

where R is the maximum radius of the celestial body, $\bar{U}_i$ is the true gravitational potential at the training datum at $\boldsymbol{x}_i$, and U_{LF} is any low-fidelity potential contributions already accounted for within the PINN-GM (discussed in Section 3.3.5).

With the network inputs and outputs non-dimensionalized, one additional preprocessing step is introduced to ensure that all inputs remain bounded. Like with PINN-GM-II, the non-dimensionalized Cartesian position coordinates are converted into their 4D spherical coordinate description of (r, s, t, u) where r is the radius, and s, t, and u are the sine of the angle between the field point and each Cartesian axes defined as x/r, y/r, and z/r (84). This conversion ensures s, t, and u remain in the numerically desirable range of $[-1, 1]$. The radial coordinate, however, is not guaranteed these same conveniences. For field points that exist at infinite radii from the body, the r coordinate can scale from $[0, \infty)$ which can pose numerical challenges to the network. To circumvent these difficulties, two proxy values of the radius are introduced, r_{clip} and $(1/r)_{\mathrm{clip}}$. r_{clip} is the radial component that is clipped for values greater than one, and $(1/r)_{\mathrm{clip}}$ is the reciprocal of r that is also clipped when its value exceeds one. This ensures that information about the radial

coordinate are always captured, but ensure that they never exceed the bounds of $[0, 1]$ and remain numerically stable for learning.

3.3.2 Modified Loss Function to Account for high altitude Samples

The loss function for the original PINN-GM is a root mean squared (RMS) error metric:

$$\mathcal{L}_{\text{RMS}}(\theta) = \sqrt{\frac{1}{N}\sum_{i=0}^{N}\left|-\nabla\hat{U}(\boldsymbol{x}_i|\theta) - \boldsymbol{a}_i\right|^2} \tag{3.24}$$

This loss function is especially common in supervised machine learning regression problems, and it encourages models to minimize the most flagrant residuals between the true and predicted values. For the PINN-GM, this means minimizing the difference between the differentiated network potential, $\nabla\hat{U}(\boldsymbol{x}_i|\theta)$, and the true acceleration, $\boldsymbol{a}_i$, to satisfy the differential equation $-\nabla U = \boldsymbol{a}$. Despite its popularity, this loss function comes with unexpected disadvantages for the gravity modeling problem as shown in earlier sections.

Specifically, gravitational accelerations produced closer to a celestial body have considerably larger magnitudes than the accelerations produced at high altitudes. As a consequence, any small relative errors in low-altitude predictions will appear disproportionately large compared to any high altitude errors. Therefore gravity models trained with this cost function always prioritize low-altitude samples over high altitude samples, even if the high altitude samples are more erroneous in a relative sense. This was demonstrated by both the PINN-GM-I and the PINN-GM-II, for which both models prioritized the accurate modeling of the low-altitude samples, even when the majority of data existed at high altitudes.

To combat this design flaw, PINN-GM-III adopts a new mean percent error loss function:

$$\mathcal{L}_{\%}(\theta) = \frac{1}{N}\sum_{i=0}^{N}\frac{|-\nabla\hat{U}(\boldsymbol{x}_i|\theta) - \boldsymbol{a}_i|}{|\boldsymbol{a}_i|} \tag{3.25}$$

By using a percent error loss instead, the PINN-GM-III is no longer biased by the absolute magnitudes of the acceleration vectors, but instead seeks to minimize relative errors. This choice ensures that all samples, regardless of altitude, are contributing equally to the loss function and

are accurately modeled by the network. If a user has applications which are especially sensitive to low-altitude accelerations — such as a landing or touch-and-go operation — Equation (3.25) can optionally be augmented with Equation (3.24).

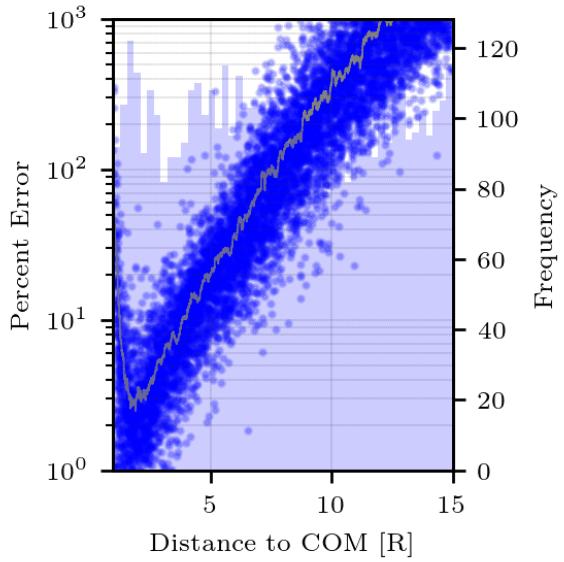
To illustrate how the network loss function affects model performance, a test is proposed which trains two PINN-GMs on 5,000 position / acceleration training data pairs distributed from 0-15 Earth radii above the Earth's surface. One PINN-GM is trained using the original RMS loss function and the other is trained on with the percent error loss function. Once trained, each network is evaluated on a set of 10,000 randomly distributed test points within the 0-15R domain, and their resulting RMS and percent error values[5] are reported as a function of altitude in Figure 3.26.

Figure 3.26a confirms that the networks trained with the RMS loss function disproportionally favor low-altitude field points (0-2R) at the expense high altitude field points (5-15R) despite being trained across the entire 0-15R domain. In contrast, the PINN-GM model trained with the percent error loss function (Figure 3.26c) prioritizes accurate modeling at all altitudes and conveniently produces lower RMS values for high altitude samples as shown in Figure 3.26d.

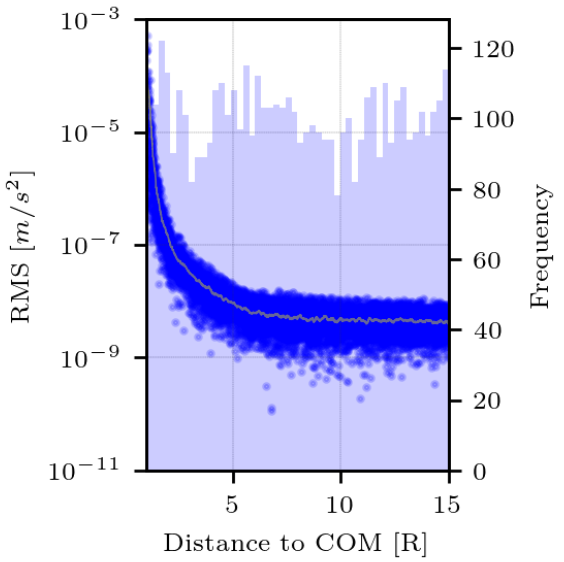
3.3.3 Improve Numerics by Learning a Proxy to the Potential

All gravitational potentials decay at high altitudes according to a power law of the form $1/r^p$. As discussed, this decay poses challenges when using a RMS loss function, but it also introduces other numerical difficulties for neural networks. Consider that the largest gravitational potential represented by the network is non-dimensionalized to magnitude 1 from Equations (3.19) and (3.21). For field points at a sufficiently high altitude, the neural network will need to compute potentials which decay to values to less than or equal to machine precision (i.e. $U(r \to r_{\text{crit}}) \leq \epsilon_{\text{machine}}$). Representing these vastly different numerical scales with the same matrix operations (neural network) is undesirable and can lead numerical instability during training and inference. As a consequence, this instability unnecessarily caps the maximum altitude for which the model is

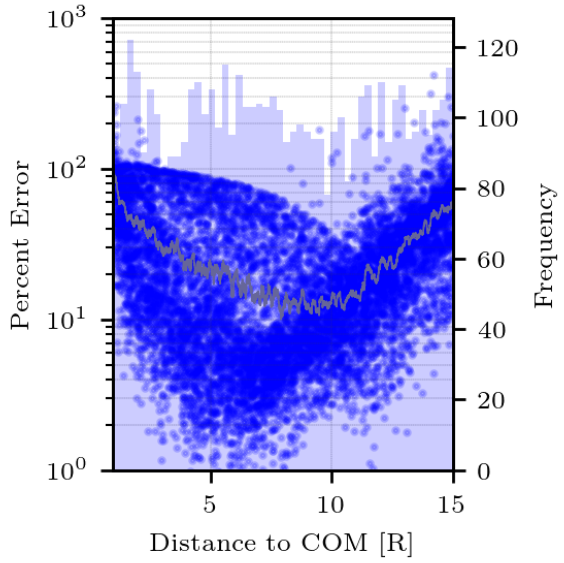
[5] Note that the errors reported are only of acceleration contributions above a degree and order 2 spherical harmonic model—i.e. how accurately the network is captures all remaining gravitational perturbations beyond $C_{2,0}$ and $C_{2,2}$. This avoids these contributions obfuscating the modeling error of the more discontinuous features and follows the procedure found in Section 3.1.2.



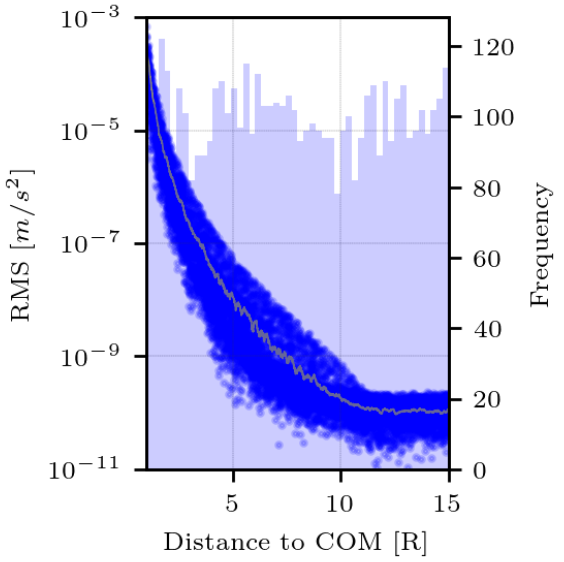
(a) Percent Error With $\mathcal{L}_{\text{RMS}}$ Function



(b) RMS Error With $\mathcal{L}_{\text{RMS}}$ Function



(c) Percent Error With $\mathcal{L}_{\%}$ Function



(d) RMS Error With $\mathcal{L}_{\%}$ Function

Figure 3.26: Different loss function change network performance at high and low altitudes. Blue points are the individual errors of the test data, the blue histogram is the distribution of training data, and the gray line is the average test error within a sliding window of 100 points.

viable.

PINN-GM-III addresses this phenomenon by learning a more numerically favorable representation of the potential that can be later transformed into the proper order of magnitude. Explicitly, PINN-GM-III proposes learning an altitude-invariant proxy to the potential, U_{NN}, defined as

$$U_{\text{NN}} = Ur^p \tag{3.26}$$

where U is the true non-dimensionalized potential function, r is the non-dimensionalized radius of the field point, and p is defined through

$$p = \begin{cases} l_r + 1 & l_r = 0 \vee 1 \\ l_r + 2 & l_r \geq 2 \end{cases} \tag{3.27}$$

where l_r is the maximum spherical harmonic degree included within the optional analytic part of the model, U_{LF} (Section 3.3.5).

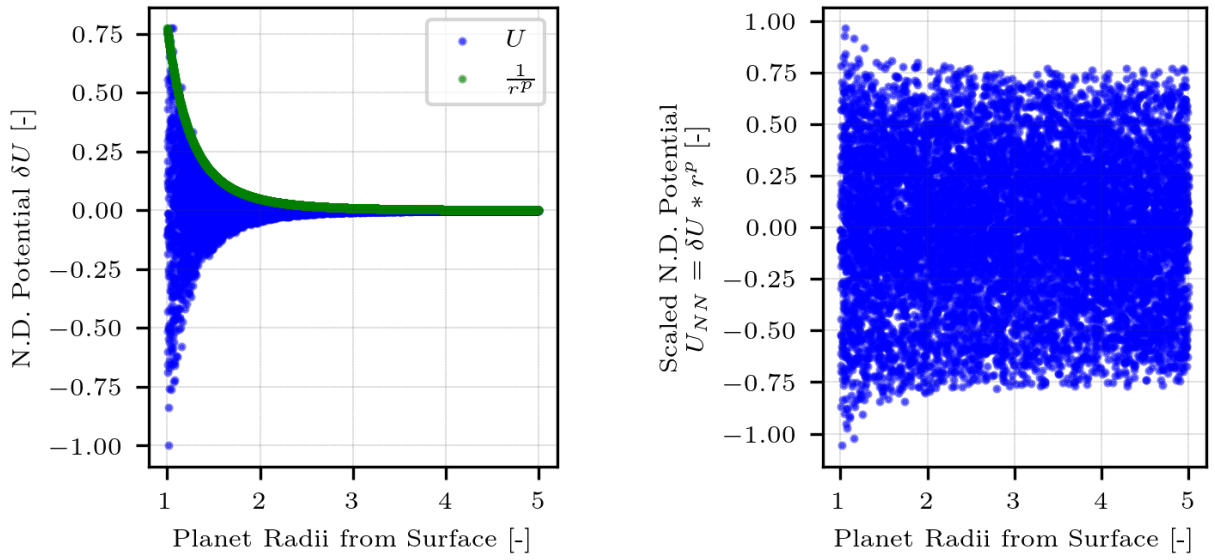
U_{NN} is designed to consistently span the domain of [-1,1] regardless of altitude. Unlike the true potential, U_{NN} has considerably lower risk of introducing numerical error and prematurely capping accuracy of the model at high altitudes. After the network produces its prediction for the proxy potential, it is then transformed into the true potential distribution later in the model through

$$\hat{U}_{\text{NN}} = \frac{U_{\text{NN}}}{r^p} \tag{3.28}$$

This ensures that the numerics of the network inference remain well-conditioned, while still allowing the model to produce the correct value / order of magnitude of the potential.

Figure 3.27 visualizes the difference between the true potential and the proxy potential as a function of altitude and Figure 3.28 shows the corresponding effect on model error. Specifically Figure 3.27a shows 10,000 values of the Earth potential[6] distributed between 0-5R (radii) from the Earth's surface. The perturbations are seen decaying at a rate of $1/r^4$ consistent with Equa-

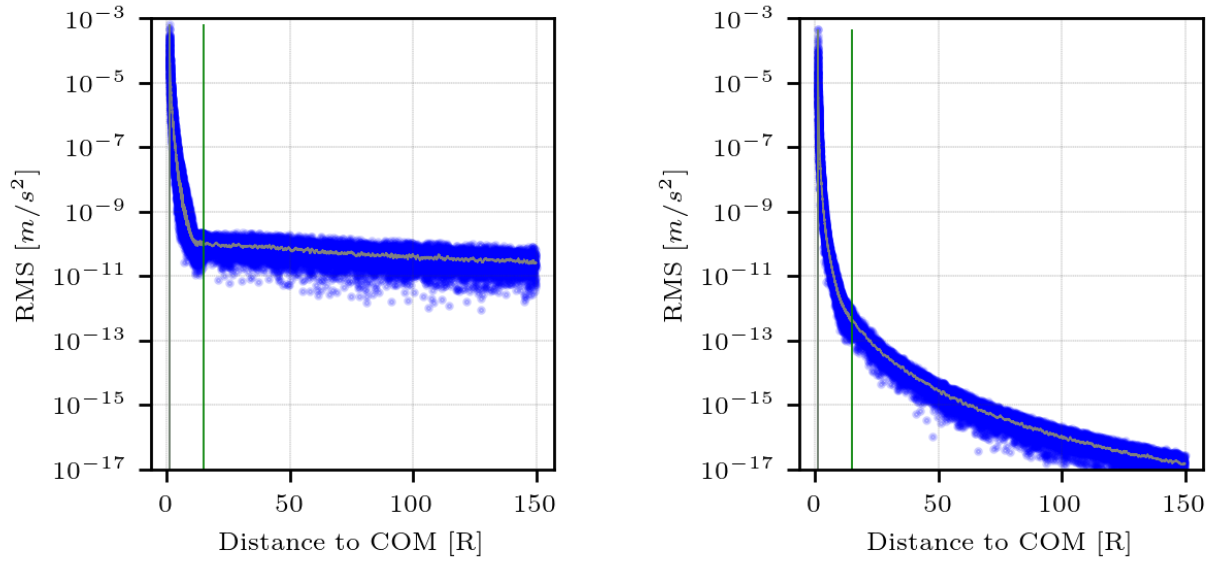
[6] Note that these values exclude contributions of a spherical harmonic model of degree and order 2 (i.e. $\delta U = U - U_{\text{LF}}$ where U_{LF} is a degree $l = 2$ spherical harmonic model).



(a) True potential U

(b) Proxy potential, U_{NN}

Figure 3.27: The true potential U (left) quickly decays to numerically unfavorable values whereas versus the proxy potential U_{NN} (right) remains numerically well-conditioned between $[-1, 1]$.



(a) Network error **without** the proxy potential

(b) Network error **with** the proxy potential

Figure 3.28: Effect of learning a potential proxy on the network inference error.

tion (3.27). In contrast, the proxy potential distribution is shown in Figure 3.27b. Note how the

values consistently exists in the more numerically favorable domain of $[-1, 1]$ regardless of altitude.

The utility of this scaling within the PINN-GM-III is shown with an experiment where two PINN-GM are trained. Both models are trained with 100 samples randomly distributed from 0-15R from the Earth's surface. One network is trained to learn the potential directly, without the scaling proposed in Equation (3.26) (Figure 3.28a). The other PINN-GM is trained to learn the potential proxy which is later divided by r^p to transform it into the correct order of magnitude (Figure 3.28b). The RMS error of each model then is evaluated on 10,000 test points that span from 0-150R.

Figure 3.28a clearly shows that PINN-GMs trained to learn the potential directly hits a numerical barrier at approximately $r = 12$R. Beyond this critical altitude, the model numerics become poorly conditioned and are unable to represent these very small values — prematurely limiting the range of altitudes in which this model can be utilized. In contrast, when the PINN-GM is trained to learn a proxy potential which is later scaled to the correct order of magnitude through Equation (3.28) there is no longer a critical altitude or numerical floor that cannot be surpassed.

3.3.4 Enforcing Boundary Conditions to Avoid Extrapolation Error

While PINNs are most commonly designed to satisfy physics through their cost function, there are also other ways to enforce compliance through the design of the machine learning model itself. Reference (93) notes how machine learning models can be designed to seamlessly transition into known boundary conditions through the use of Heaviside-like functions.

The PINN-GM-III proposes the following design modification to enforce relevant boundary constraints:

$$\hat{U}(r) = \big(1 - H(r)\big)\hat{U}_{\mathrm{NN}}(r) + H(r)U_{\mathrm{BC}}(r) \tag{3.29}$$

where $\hat{U}_{\mathrm{NN}}$ is the predicted gravitational potential, U_{BC} is the potential at the boundary condition,

and $H(r)$ is a Heaviside-inspired function defined as

$$H(r) = \frac{1 + \tanh(k(r - r_{\text{ref}}))}{2} \tag{3.30}$$

where r is the radius of the field point, r_{ref} is a reference radius, and k is a smoothing parameter to control for a more continuous or discrete transition. Note that both r_{ref} and k can be a user-prescribed or learned parameter.

Equation (3.30) enforces that the PINN-GM must smoothly transition into a known boundary condition past the reference altitude r_{ref}. In this way, the machine learning model can leverage the modeling flexibility of the neural network to represent complex regions of the gravity field near the body for which the boundary condition is irrelevant, but then smoothly decrease the network's responsibility in the limit as the model approaches the boundary.

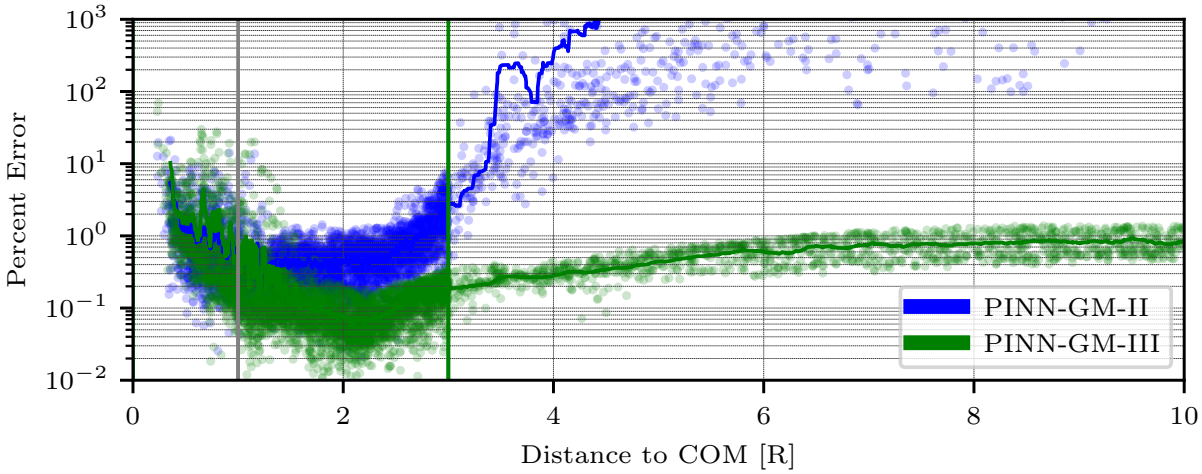
There are multiple ways in which the boundary condition can be enforced. In the limit of $r \to \infty$, the potential decays to zero as discussed in Section 3.3.3. However, setting $U_{\text{BC}} = 0$ and $r_{\text{ref}} = \infty$ in Equation (3.29) is not practical as it demands the neural network must learn a model of the potential for the entire domain $r \in [0, \infty)$. A more useful choice is to use a lower fidelity model in the limit, where higher accuracy is not necessary. PINN-GM-III accomplishes this by taking insights from the spherical harmonic gravity model and recognizing that high frequency components of the gravitational potential decay to zero more quickly than the point mass contribution at high altitudes. I.e.

$$U_{\text{BC}}(r) = U_{LF} = \frac{\mu}{r} + \sum_{l=0}^{n} \sum_{m=0}^{l} \frac{\mu}{r} \left(\frac{R}{r}\right)^{l} (\ldots) \to 0 \tag{3.31}$$

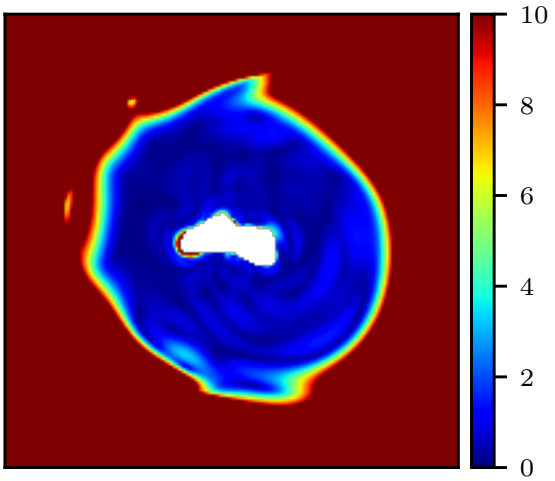
as $r \to \infty$.

This observation implies that $U_{\text{BC}}(r)$ can be set to $\frac{\mu}{r}$ assuming $r \gg R$. As such, the PINN-GM-III sets $U_{\text{BC}} = \frac{\mu}{r} + f(r)$ in Equation (3.29), where $f(r)$ are any higher order terms in the spherical harmonic gravity model that the user knows a priori and wishes to leave as part of the boundary condition.

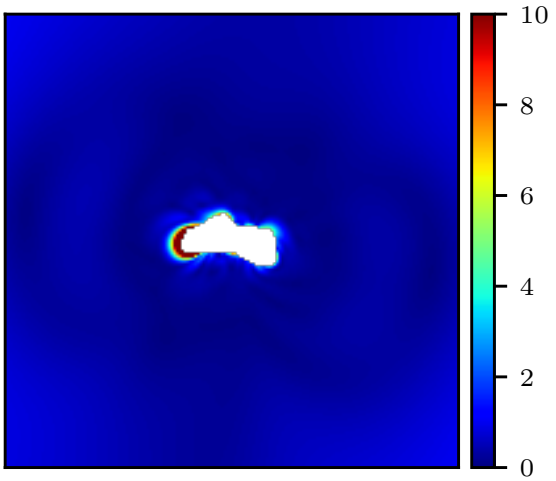
To illustrate the effect of enforcing these boundary conditions, an experiment is performed where a PINN-GM-II and a PINN-GM-III are trained. PINN-GM-II does not contain any infor-



(a) PINN-GM-II and III performance inside (0R-3R) and outside (3R-10R) the training domain.



(b) PINN-GM-II % Error



(c) PINN-GM-III % Error

Figure 3.29: Top Row: Percent error of PINN-GM inside (0-3R) and outside (3R-10R) of the training domain. Gray vertical line is Brillouin radius. Green vertical line is maximum bounds of training data. Bottom Row: % Error of PINN-GM in XY plane.

mation about the boundary condition in its model design, whereas PINN-GM-III incorporates the proposed transition captured by Equation (3.29). Each network contains eight hidden layers with 10 nodes each, and both models are trained on the same 1,000 data points spanning 0-3R about the asteroid Eros. The average errors of both models are reported as a function of altitude from 0-10R in Figure 3.29a, and the XY Cartesian plane cross sections are show in Figures 3.29b and 3.29c.

Figure 3.29 shows that when Equation (3.29) is not included in the model design, the PINN-GM diverges when tested outside of the bounds of its training data. In contrast, when Equation (3.29) is included, PINN-GM-III is able to maintain low errors even at high altitudes for which there is no training data. This stark contrast in model behavior demonstrates the advantage of enforcing boundary conditions in the model design. PINN-GM-III is able to use the neural network to learn a rich gravity model representation within the bounds of the training data, but as soon as the model exits those bounds it is guaranteed to perform no worse than a point mass approximation or a low-fidelity spherical harmonic model. This is a powerful development for these machine learning gravity models as it showcases that these models can be reliably used beyond the bounds of their training data and will always perform as well or better than their low-fidelity counterpart.

3.3.5 Leveraging Preexisting Gravity Information into PINN-GM Solution

In addition to enforcing the boundary condition through the model design, another design choice enabled by the PINN-GM-III is the ability to fuse an a priori gravity model with the neural network solution. For example, most large celestial bodies exhibit planetary oblateness which is succinctly captured with the $C_{2,0}$ spherical harmonic coefficient as discussed in Chapter 2. Rather than requiring the network to relearn this prominent and easily observable perturbation, this information can be directly embedded into the network model. In this way, the PINN-GM-III can predict accelerations by leveraging a low-fidelity, first-order analytic model with a network responsible for capturing high-order perturbations through:

$$\hat{U}(r) = \big(1 - H(r)\big)\big(U_{\text{LF}}(r) + U_{\text{NN}}(r)\big) + H(r)\big(U_{\text{LF}}(r)\big) \tag{3.32}$$

where U_{LF} refers to the known, low-fidelity analytic model such as $U_{\text{LF}} = \frac{\mu}{r} + U_{J_2}$.

Algorithm 1: PINN-GM-III algorithm

Collect training data $(\boldsymbol{x}, \boldsymbol{a})$ from:
    (a) a pre-existing model
    (b) online state estimates (94) Non-dimensionalize the training data `// Sec. 3.3.1`
Convert to spherical coordinates (r_{clip}, 1/r, s, t, u) `// Sec. 3.3.1`
(Optional) Encode inputs into Fourier space `// App. C`
Propagate through the trained neural network Output proxy potential `// Sec. 3.3.3`
Scale proxy potential into true potential `// Sec. 3.3.3`
Enforce boundary conditions on the network potential `// Sec. 3.3.4`
`// Autodifferentiate (AD) potential to produce acceleration` $\hat{\boldsymbol{a}}$
if Training then
    Add acceleration percent error to the loss `// Sec. 3.3.2`
    (Optional) Compute jacobian for $\hat{\boldsymbol{a}}$ *via AD (Optional) Perform learning rate annealing* `// App. B`
    (Optional) Add annealed $\nabla^2 \hat{U}$ *term to the loss function* Compute gradients of loss function Update network parameters

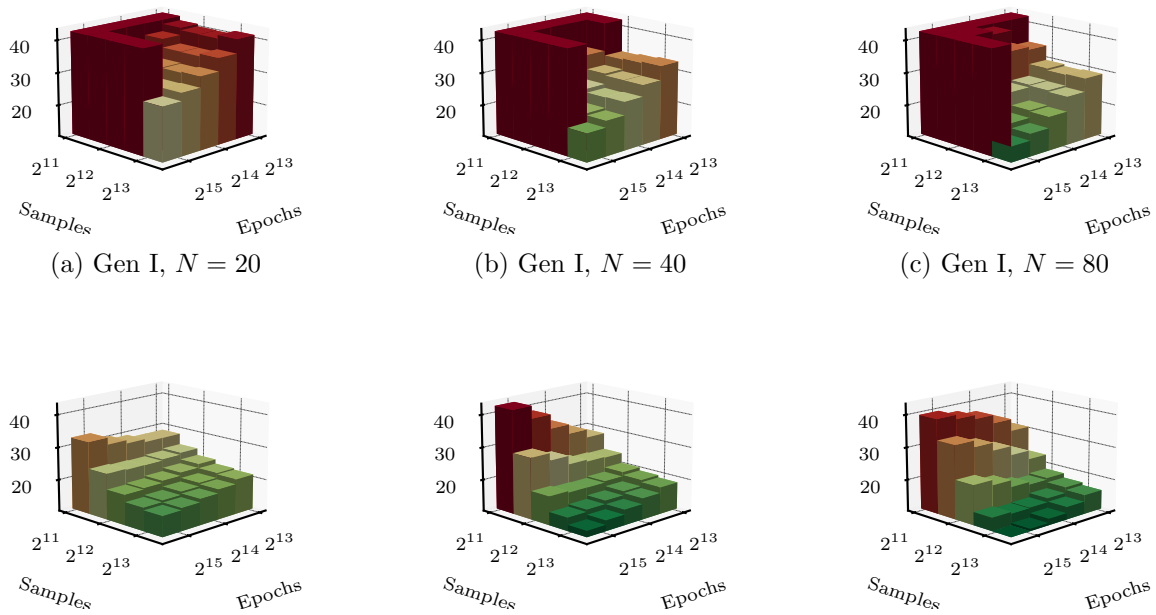
3.4 PINN-GM-III Performance: Comparative Study

In this section, the performance of the PINN-GM-III is compared to that of its two model predecessors, PINN-GM-I and PINN-GM-II. The models are trained in the environments of their original papers, PINN-GM-I trained to represent the gravity field of the Earth, and PINN-GM-II trained to represent the asteroid 433-Eros. In both cases, the models' performance are evaluated across various key hyperparameter configurations including the number of training data, the number of training epochs, and the total model capacity. For the sake of legibility, PINN-GM-I, PINN-GM-II, and PINN-GM-III will be referred to as PINN-I, PINN-II, and PINN-III respectively.

Three model capacities are compared in each experiment. The model capacity is defined by the number of nodes per hidden layer, $N_{\text{nodes}} = \{20, 40, 80\}$, for which there are a total of 8 hidden layers. Each model is trained 25 times, each with a different number of training data and training epochs drawn from $N_{\text{data}} = \left\{2^{10}, 2^{11}, 2^{12}, 2^{13}, 2^{14}\right\}$ and $N_{\text{epochs}} = \left\{2^{12}, 2^{13}, 2^{14}, 2^{15}, 2^{16}\right\}$ respectively. Remaining details regarding the PINN-III's training are found in Appendix A.

3.4.1 Generation I Versus Generation III

The first experiment investigates the performance of the PINN-III as compared to the original PINN-I. Each network is trained on data generated from Earth's high-fidelity static gravity field model: Earth Gravity Model 2008 or EGM 2008 (95). This model extends to degree and order 2,160 but this experiment is limited to degree and order 1,000 (∼1 million parameters) to ensure the fidelity is sufficiently high to capture the high-frequency perturbations.



(a) Gen I, $N = 20$ (b) Gen I, $N = 40$ (c) Gen I, $N = 80$

(d) Gen III, $N = 20$ (e) Gen III, $N = 40$ (f) Gen III, $N = 80$

Figure 3.30: Average acceleration percent error defined in Equation (3.33) of PINN-GM-I (top) and PINN-GM-III (bottom) trained on Earth.

The training data for each network are distributed randomly from the Earth's radius to an approximately LEO altitude of 420 km. Once the field points are selected, their corresponding accelerations are computed using EGM2008 and together form the training data. The remaining hyperparameters for the PINN-III network are included in Table A.1 and the default hyperparameters for PINN-I are found in Table 3.1.

The performance of each network is evaluated using a mean percent error metric computed on

a separate test set of 50,000 randomly distributed field points within the same bounds as the training data. Note that the percent error metric purposefully excludes the acceleration contributions from the Earth's point mass and planetary oblateness through:

$$\mathcal{P} = \frac{1}{N}\sum_{i=0}^{N}\frac{|(\hat{\boldsymbol{a}}_i - \boldsymbol{a}_{i,2}) - (\boldsymbol{a}_i - \boldsymbol{a}_{i,2})|}{|\boldsymbol{a}_i - \boldsymbol{a}_{i,2}|} * 100 \tag{3.33}$$

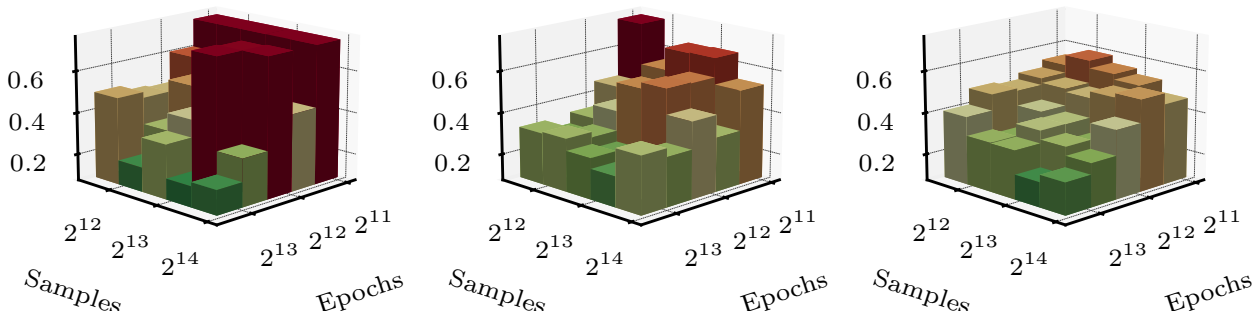
where $\hat{\boldsymbol{a}}$ are the network accelerations produced via $-\nabla U$ and $\boldsymbol{a}_{i,2}$ are the accelerations produced by the point mass and planetary oblateness terms of the spherical harmonic gravity model. The omission of $\boldsymbol{a}_2$ from the percent error makes it easier to assess how well the PINN-GM is capturing the remaining high-order perturbations like mountain ranges and tectonic plates.

Figure 3.30 shows that there is a considerable increase in performance between PINN-I and PINN-III. For the low capacity cases, where the number of nodes per layer is 20 (model size of ≈3,000 parameters), PINN-I consistently produces low-accuracy predictions ($\geq$ 35% error). In the low-data cases, some PINN-I even diverge. In contrast, PINN-III consistently converges to a solution with errors less than 30% even in the low-data cases. Similar findings manifest for the higher capacity cases of $N = \{40, 80\}$, with PINN-III consistently outperforming its predecessor across all data sizes and training durations.

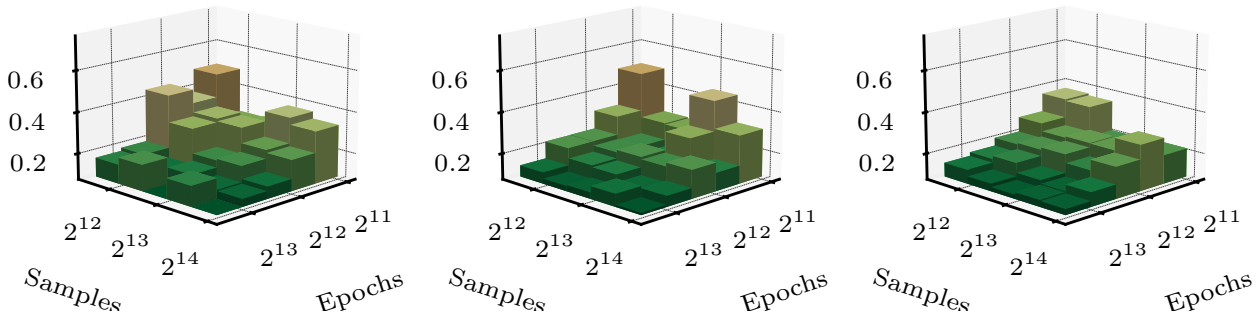
Notably, these results not only imply that the PINN-III is able to achieve considerably lower error than PINN-I, but also that it can be achieved with less data, less training time, and smaller models. Consider the fact that the 20 node PINN-III model trained with $N_{\text{data}} = 1024$ and $N_{\text{epochs}} = 4096$ (Figure 3.30d smallest sample, fewest epochs) has a near identical error to the 80 node PINN-I model with $N_{\text{data}} = 32,768$ and $N_{\text{epochs}} = 8192$ (Figure 3.30a most samples, second fewest epochs). **This suggests that a PINN-III that is 94% smaller, trained with 97% less training data, and for 50% fewer epochs is able to achieve the same accuracy as PINN-I.**

One additional noteworthy result from Figure 3.30 is that high capacity PINN-III models can experience overfitting. When there is insufficient data ($N_{\text{data}} = 1024$), the error begins to increase for the longer training durations as best exhibited in Figures 3.30e and 3.30f. While the

overfitting of PINN-III remains less problematic than what is seen with the PINN-I, this can be trivially remedied by adding a small amount of dropout to each layer, incorporating the Laplacian and curl physics constraints into the cost function, or simply by adding more training data. The dropout and additional constraints are purposefully not included in this experiment as they act as regularizing terms which can obfuscate the maximum performance of these networks in the low-data conditions.



(a) Gen II, $N = 20$ (b) Gen II, $N = 40$ (c) Gen II, $N = 80$



(d) Gen III, $N = 20$ (e) Gen III, $N = 40$ (f) Gen III, $N = 80$

Figure 3.31: Average acceleration percent error of PINN-II (top) and PINN-III (bottom) trained on Eros.

3.4.2 Generation II versus Generation III

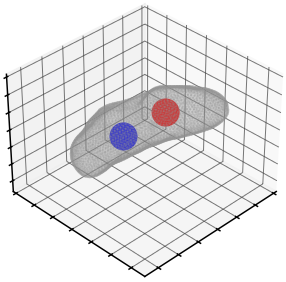
The second experiment investigates the improvements of the PINN-III over its more recent predecessor PINN-II. In this experiment, the testing environment transitions to the asteroid 433-Eros for which the PINN-II was originally developed. Rather than using a spherical harmonic gravity model, the truth accelerations are computed using a polyhedral model based on a low-resolution shape model of Eros comprised of approximately 11,000 total facets and vertices. The training data are distributed uniformly between the surface of the asteroid and a maximum radius equal to three radii from the center of mass.

Figure 3.31 illustrates the acceleration error of PINN-II and PINN-III. Note that the percent error reported is of the full acceleration vector ($|\hat{\boldsymbol{a}} - \boldsymbol{a}|/|\boldsymbol{a}|$) rather than with the point mass and oblateness terms removed. While PINN-II is more robust than PINN-I, converging to a reasonable solution in all cases tested, PINN-III continues to outperform it. While both PINN-II and III maintain an error of $< 1.5\%$, PINN-III consistently remains below $< 0.5\%$ improving performance by a factor of 3. PINN-III is also able to achieve better performance with less data, training time, and smaller model sizes. The smallest models of PINN-III produced errors comparable to the majority of hyperparameters tested with the 80 node PINN-IIs.

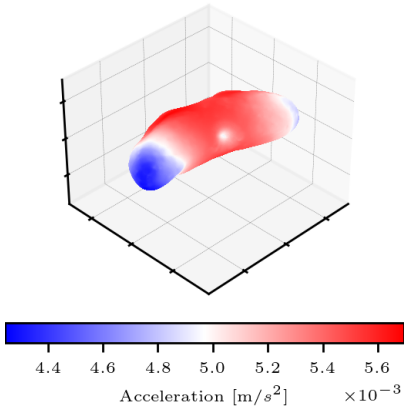
3.5 PINN-GM-III Performance: Heterogeneous Density Asteroid

One challenge modeling gravity fields of small-bodies is the uncertainty surrounding the asteroid's or comet's internal density distribution. Recent work has shown that the constant density assumption often employed by the polyhedral gravity model is not always valid (44). Some asteroids may contain over and under dense regions within their interior, or may have been formed by two asteroids merging together, each with different densities. Unfortunately the internal structure of these small bodies is extremely difficult to measure, as it requires directly probing the interior of the object. As a less intrusive alternative, dynamicists currently make informed assumptions about these distributions based on the asteroid's gravity field and shape and compare the measured

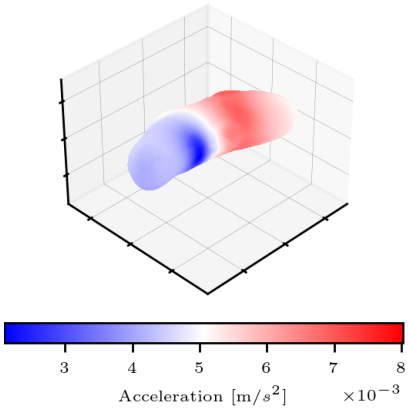
spherical harmonic coefficients against those that would be generated by their assumed profile (73). At best, this process leaves researchers with heuristic assessments of the body's density profile which can be used in heterogeneous forms of the polyhedral model (73). However, it remains common practice to simply proceed with a constant density assumption. This section explores the ramifications of this choice and highlights how the PINN-GM can bypass many of the consequences of this practice.



(a) Induced heterogeneity



(b) Gravity (assumed constant density)



(c) Gravity (with heterogeneity)

Figure 3.32: Asteroid with a mass heterogeneity (top) and the corresponding surface gravity field when constant density is assumed versus the true field (bottom left vs. bottom right respectively).

For this experiment, a small mass heterogeneity is introduced to the asteroid 433-Eros. In one hemisphere, a mass element is added to the interior of the body, and in the other hemisphere, a mass element is removed. Each mass element contains 10% of the total mass of the asteroid,

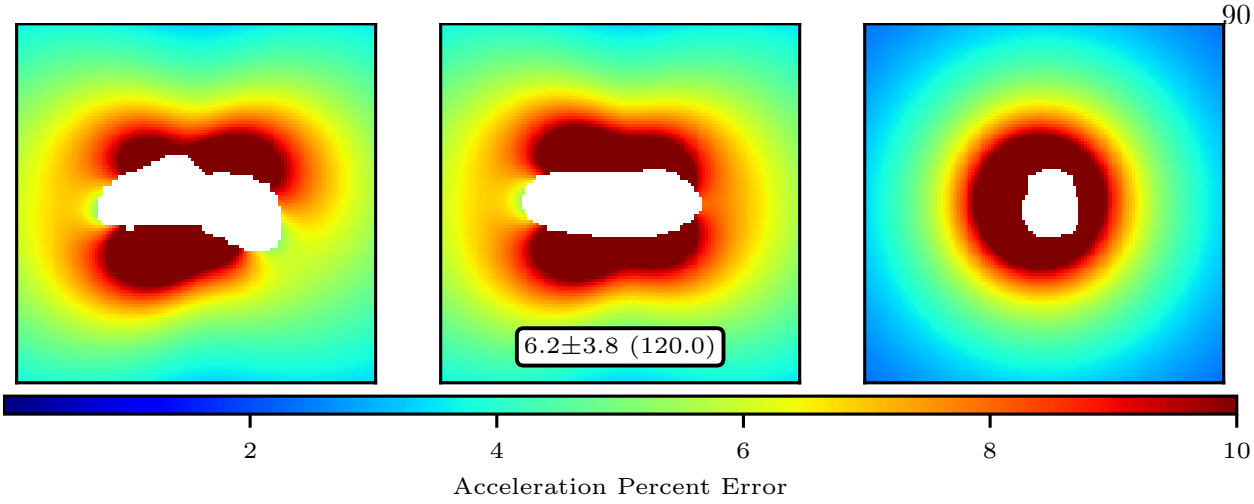


Figure 3.33: Constant density polyhedral acceleration error reported as $\mu \pm \sigma$ (max).

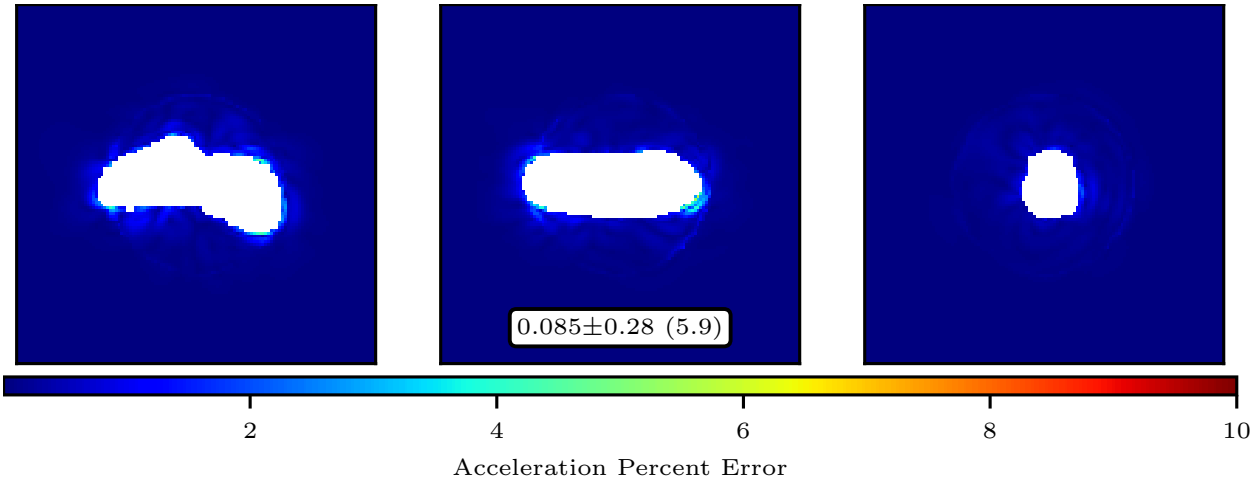
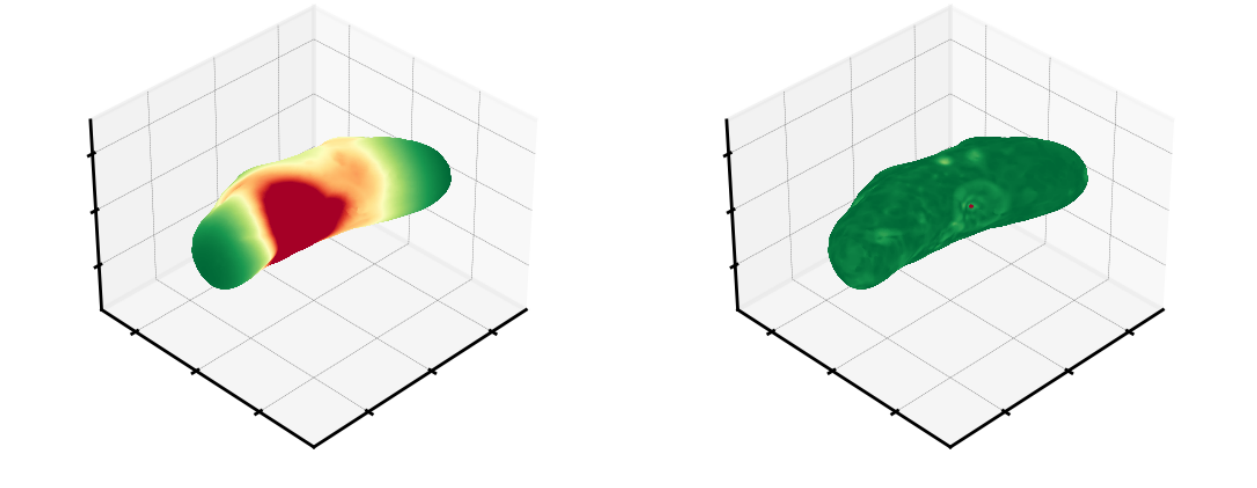


Figure 3.34: PINN-GM-III acceleration error reported as $\mu \pm \sigma$ (max).

and they are symmetrically displaced along the x-axis by 0.33R (see Figure 3.32a). This choice is meant to emulate a scenario where an two asteroids with different densities may have merged into one. The choice to make each mass element 10% of the total mass is motivated based on literature with similar candidate density distributions (96).

This small heterogeneity can have large implications for the asteroid's gravity field. Figure 3.32b depicts the field under the assumption of constant density, whereas Figure 3.32c shows the true, heterogeneous density gravity field. From these figures, it is apparent that the constant density assumption can lead to vastly different gravitational signatures and, by association, space-

craft trajectories.



(a) Constant σ polyhedral model error

(b) PINN-GM-III error

Figure 3.35: Surface error of the constant density polyhedral gravity model and PINN-GM-III.

Figure 3.33 illustrates the acceleration error of a constant density polyhedral model on the XY, XZ, and YZ planes out to $3R$ and reports the average, standard deviation, and maximum acceleration error. The constant density assumption becomes increasingly invalid near the surface, producing maximum errors in excess of 100%. Errors of these magnitudes introduced considerable risk for missions that aspire to land or operate in close proximity to the asteroid.

As an alternative, a PINN-GM-III is trained to represent gravity field of the heterogeneous density asteroid. Explicitly, the training data for the PINN-GM is the superset of two datasets: the first dataset contains data exclusively sampled on the perimeter of the shape model where each data point's position corresponds with one facet of the polyhedral shape model (200,700 facets). The second dataset is one million data points spanning from the surface of the asteroid out to 10 radii. While experiments in Section 3.4.2 studied data-sparse performance of the PINN-GM-III, this experiment purposefully studies performance in a data rich environment, when ample training

samples exists.

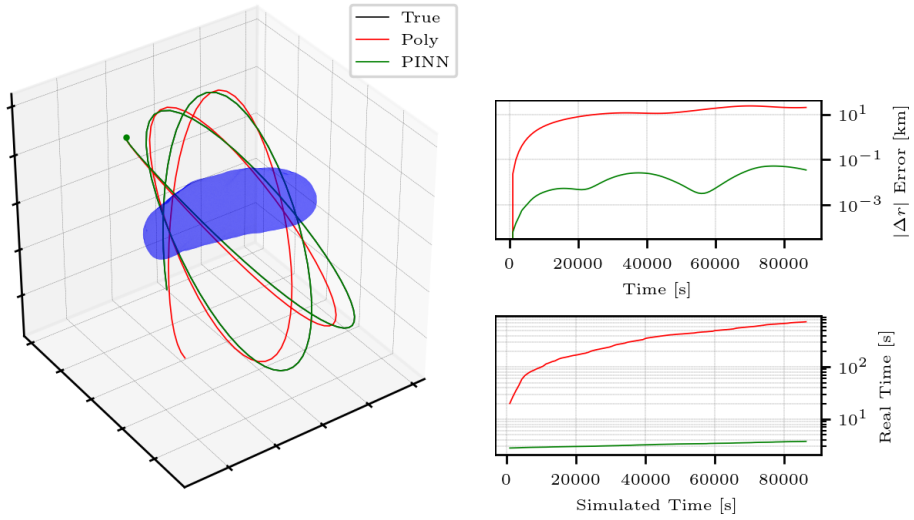


Figure 3.36: Propagation speed and error of the constant density polyhedral model and PINN-GM-III.

The performance of the trained PINN-GM is visualized in Figures 3.34 and 3.35. The average and maximum error of the PINN-GM is nearly two orders of magnitude lower than the constant density polyhedral model at both high altitudes and at the surface. Note that the PINN-GM-III only contains approximately 3,000 parameters in contrast to the polyhedral model, which includes over 200,000 facets and 100,000 vertices. These results demonstrate that the PINN-GM-III is able to achieve an order of magnitude reduction in error using two orders of magnitude fewer parameters in the model.

Not only is the PINN-GM-III more accurate and compact, it is also more computationally efficient. This is demonstrated in Figure 3.36 in which a set of initial conditions is propagated once using the constant density polyhedral model, and once with the trained PINN-GM-III. The figure shows that propagating the spacecraft for one simulated day requires approximately 30 minutes in real time, whereas the PINN-GM-III requires a mere three seconds — a 600x speedup. Moreover, Figure 3.36 also highlights the risk of making the constant density assumption when orbiting exotically shaped small bodies. After one day of propagating the state under a constant density polyhedral gravity model, the trajectory evolves to have position errors norms in excess of

20 km from the true trajectory. The PINN-GM, in contrast, makes no assumptions and produces errors norms of < 0.1km.

Chapter 4

Application I: Reinforcement Learning

Reinforcement learning is framework used to train decision making agents to perform some complex behavior. This can manifest as teaching cars how to drive themselves (97), building recommender systems that suggest which netflix show to watch next (98), or even training chess bots how to outperform international grandmasters (99). At it's core, reinforcement learning is performed by having an agent repeatedly take actions within some environment and receive a reward signal indicating if the action was good or bad. For example, if a chess bot performs a move which puts their opponent in check, that would receive a positive reward. In contrast, if the bot performs a move that jeopardizes their own king's safety, that would receive negative reward. By continuously trying different moves and estimating the associated long-term reward, the reinforcement learning agent builds an history of experiences from which it can learn more sophisticated and desirable behavior.

Reinforcement learning is extremely general, and there exist a wide range of possible applications which has garnered the attention of many scientific and engineering communities (100; 101). In many cases, these communities aspire to train agents to learn more complex behaviors or solutions that outperform current state-of-the-art methods. The field of astrodynamics is no exception. In the past few years, multiple researchers have begun to explore how reinforcement learning can be used to produce more flexible and robust autonomous spacecraft for deep space exploration. These efforts include investigating how spacecraft can design more robust guidance algorithms capable of handling environmental uncertainty (102; 103), identifying candidate transfers trajecto-