Modeling Irregular Small Bodies Gravity Field Via Extreme Learning Machines and Bayesian Optimization

As Published: 10.1016/J.ASR.2020.06.021

Publisher: Elsevier BV

Persistent URL: https://hdl.handle.net/1721.1/134439

Version: Author's final manuscript: final author's manuscript post peer review, without publisher's formatting or copy editing

Modeling Irregular Small Bodies Gravity Field Via Extreme Learning Machines and Bayesian Optimization

Roberto Furfaro[1,*]

University of Arizona, 1127 E. James E. Roger Way, Tucson, Arizona, 85721,USA

Riccardo Barocco[2,*]

University of Arizona, Tucson, AZ, 86721

Richard Linares[3,*]

Massachusetts Institute of Technology, Cambridge, MA 02139

Francesco Topputo[4,*]

Politecnico di Milano, viaLa Masa 34, 20156 Milano, Italy

Vishnu Reddy[5,*]

University of Arizona, 1127 E. James E. Roger Way, Tucson, Arizona, 85721,USA

Jules Simo[6,*]

University of Central Lancashire, Preston, Lancashire, PR1 2HE, UK

Lucille Le Corre[7,*]

Planetary Science Institute, 1700 East Fort Lowell, Suite 106, Tucson, AZ 85719, USA

Abstract

Close proximity operations around small bodies are extremely challenging due to their uncertain dynamical environment. Autonomous guidance and navigation around small bodies require fast and accurate modeling of the gravitational field for potential on-board computation. In this paper, we investigate a

*Corresponding author

Email address: robertof@email.arizona.edu (Roberto Furfaro)

[1]Professor, Department of Systems and Industrial Engineering, Department of Aerospace and Mechanical Engineering

[2]Visiting Graduate Student, Department of Systems and Industrial Engineering

[3]Charles Stark Draper Assistant Professor, Department of Aeronautics and Astronautics

[4]Associate Professor, Dipartimento di Scienze e Tecnologie Aerospaziali,

[5]Associate Professor, Department of Planetary Science, Lunar and Planetary Laboratory

[6]Lecturer, School of Engineering

[7]Research Scientist, Planetary Science Institute

model-based, data-driven approach to compute and predict the gravitational acceleration around irregular small bodies. More specifically, we employ Extreme Learning Machine (ELM) theories to design, train and validate Single-Layer Feedforward Networks (SLFN) capable of learning the relationship between the spacecraft position and the gravitational acceleration. ELM-base neural networks are trained without iterative tuning therefore dramatically reducing the training time. Analysis of performance in constant density models for asteroid 25143 Itokawa and comet 67/P Churyumov-Gerasimenko show that ELM-based SLFN are able learn the desired functional relationship both globally and in selected localized areas near the surface. The latter results in a robust neural algorithm for on-board, real-time calculation of the gravity field needed for guidance and control in close-proximity operations near the asteroid surface.

Keywords: Extreme Learning Machine; Gravity Modeling; Asteroid

1. Introduction

Over the past few years, there has been a strong interest in sending robotic spacecrafts to small bodies in the solar system, including comets (e.g. Rosetta Mission) and Near Earth Asteroids (NEAs, e. g. Hayabusa Mission). The interest in exploring small bodies stems from the fact that they have been minimally processed since the birth of the solar system. Indeed, detailed remote mapping and in-situ sampling of such objects may provide scientists with opportunities to unveil the early history of the solar system (Ciesla and Charnley (2006)). Aside from the extremely valuable contribution that NEA missions would provide to the global understanding of the origin of the Solar System, such robotic missions would help characterize and quantify the amount of extraterrestrial natural resources (Kargel (1994)), as well as help quantifying the risk that such objects may collide with planet Earth (Strange et al. (2013)). The NASA OSIRIS REx Asteroid Sample Return Mission has arrived in proximity of the asteroid Bennu in August 2018 and shortly after the spacecraft approach, initiated the mapping operations that will culminate with the collection of a surface sample to be returned on Earth by 2023. Meanwhile, the Japanese space agency JAXA has been operating Hayabusa 2 around asteroid Ryugu with similar intents. Furthermore, NASA is planning for additional missions to small bodies such as Lucy, who will be launched in 2021 to execute a tour of six Jupiter's Trojans (Levison et al. (2017)), and Psyche, an orbiter mission that will explore the origin of planetary cores by studying the metallic asteroid, 16 Psyche (Oh et al. (2016)).

Any of the currently operating and upcoming asteroid or comet missions requires planning a set of robust close-proximity operations around small bodies. Such operations tend to be extremely challenging and complicated by a number of factors including irregular shape and mass distribution, weak and uncertain gravitational field, accelerations due to outgassing of comets, as well as perturbing accelerations due to solar radiation which in many cases tend to

be of the same order of magnitude of the gravitational acceleration. Due to such factors, the orbital dynamics around small bodies significantly deviates from the ideal Keplerian motion and tends to be highly irregular. In strongly perturbed dynamical systems, the orbital motion tends to be very unstable and many orbits either escape or crash on the small body surface. Each small body exhibits a certain number of orbits that can be considered to be stable over a long period of time (Berry et al. (2013)). Likewise, planning and executing close-proximity operations such as hovering, landing and Touch-And-Go (TAG) (Brillouin (1933)) require an accurate characterization of the dynamical environment including accurate knowledge and representation of the gravitational field. Indeed, precise design of descending trajectories requires accurate knowledge of the gravitational acceleration. Autonomous, closed-loop operations may need an efficient and fast representation of the gravity field near the surface for on-board calculations of the guiding acceleration command and/or timing the impulse.

The classical approach to modeling the gravitational field has been via the spherical harmonics expansion method. The multi-pole expansion has been largely employed to represent the gravitational potential field because of the high accuracy ensured with relatively low computational cost. Indeed, once the gravitational coefficients are measured (e.g. via a radio science orbital campaign during the operational phase), the gravitational acceleration can be easily computed by taking the spatial derivatives of the potential harmonics with the desired degree of precision. However, the mathematical convergence of the series is guaranteed only for points outside the so-called Brillouin Sphere, i.e. a sphere centered at the expansion center and circumscribing the outermost mass element of the small body (Russell and Arora (2012a)). Inside the sphere, the series of harmonics is known to diverge. Whereas the divergence does not constitute a problem for near-spherical small bodies, it causes major issues for bodies exhibiting irregular shape and mass distribution. As a result, the exterior potential is not capable of modeling the gravitational field inside the Brillouin Sphere. Alternatives to the exterior gravity model potential exist, e.g. the mass concentration (mascon) model (Werner and Scheeres (1996a)) and the polyhedral model (Takahashi et al. (2013a)). Whereas the first model tends to be inaccurate, the polyhedral gravity field can accurately describe the gravity field as function of the density (homogeneous or heterogeneous). However, the polyhedron model is very computationally expensive and generally not suitable for ground-based Monte Carlo simulations or for on-board propagation of the spacecraft dynamics. Recently, the so-called interior gravity field expansion has been investigated both from a theoretical (Takahashi et al. (2013a)) and computational (Huang et al. (2006b)) point of view. The method, which is a variant of the exterior gravity harmonic expansion, is computationally inexpensive but tends to be less general and restricted to TAG scenarios because convergence is guaranteed up to one point at the surface and total mapping has not been studied.

Recently, there has been interest in exploring new methods in modeling the gravitational field using a data-driven approach. For example, Yang et al. (Yang

et al. (2020)) developed a computationally fast method to calculate the gravitational acceleration of small irregular bodies using Chebyshev polynomials. The proposed approach relies on an analytical approximation of the polyhedron gravity via polynomial interpolation. Similarly, Gao and Liao (Gao and Liao (2019)) propose a Gaussian Process Regression approach to gravity acceleration modeling. The overall idea is to directly approximate the map between position around the asteroid and the corresponding gravitational acceleration using a learning paradigm rooted in statistical machine learning.

In this paper, we propose a new methodology for modeling the gravity field of an irregular small body for a fast, accurate, and efficient calculation of the gravitational acceleration as function of the relative position around the small body of interest. The methodology is based on a recently developed machine learning approach called Extreme Learning Machines (ELMs) (Huang et al. (2006a)) which employ a Single Layer Feedforward Network (SLFN) to model the non-linear relationship between inputs and outputs. In this case, the goal is to train, both in batch and sequential fashion, a SLFN to represent the relationship between spacecraft position around the small body of interest and the value of the gravitational acceleration (i.e. $g(r)$). We rely on a series of physical models (e.g. polyhedral models) to accurately represent the spatial variation of gravity field around the small body and sample the spacecraft position to generate the training set $r_i, g(r_i)_{i=1}^N$ that specifies the desired functional relationship. Subsequently, we employ the ELM theories to efficiently train the network to learn the desired relationship $g(r)$. Once trained and validated, the ELM-based SLFN is capable of processing a new input position (i.e. locations that have never been seen by the network) and rapidly compute the corresponding gravitational acceleration. The method is particularly suited for situations that require fast and accurate calculations of the gravity field as function of the position (e.g. on-board propagation of spacecraft trajectories during operations near the surface of a highly irregular body (Simplício et al. (2018); Furfaro et al. (2013); Pinson and Lu (2018); Yang et al. (2019)).

This paper is organized as follows. First, the classical methods for computing the gravity field around small bodies (i.e. asteroids and comets) are briefly reviewed. Then, the model-based, data-driven ELM approach is described both theoretically and operationally. The Bayesian optimization approach is discussed as fundamental methodology employed to tune the ELM hyper parameters. Subsequently, the proposed approach is evaluated in modeling the gravity field of the asteroid 25143 Itokawa and the comet 67P/Churyumov-Gerasimenko. High-fidelity polyhedron models are employed to generate the desired training set and to evaluate the performance of the proposed ELM modeling approach for both local and global gravity field determination. The ELM-based gravity acceleration is also employed to compute the closed-loop, energy-optimal acceleration command that may occur during real-time guided landing scenarios. The paper ends with conclusions and a look at future efforts beyond this study.

2. Review of Methods for Modeling the Gravity Field in Small Bodies

2.1. Spherical harmonics expansion

The most common and best documented approach concerns the representation of the gravity potential as derived by an expansion of orthogonal spherical harmonics, hereinafter denoted as SH, for brevity's sake. One common notation for the exterior gravitational potential is:

$$U(r,\phi,\theta) = \frac{GM}{r}\sum_{n=0}^{\infty}\left(\frac{R^{\star}}{r}\right)^{n}\sum_{m=0}^{n}P_{nm}\sin\phi\left[a_{nm}\cos(m\theta)+b_{nm}sin(m\theta)\right] \quad (1)$$

where $R^{\star}$ is the reference radius, P_{nm} is the associated Legendre function of degree n and order m, and a_{nm} and b_{nm} are the spherical harmonics coefficients; the spherical coordinates (r,ϕ,θ) are radius, latitude and longitude respectively. One major advantage of such an approach lies in the possibility of adjusting the complexity of the representation depending on the required accuracy by truncating the series, thus reducing the computational cost of the algorithm. In fact, in most applications, a suitably accurate gravitational model can be achieved by retaining only few relevant terms of the expansion. As analyzed by Werner and Scheeres (1996b), the main flaw of the SH approach is its unstable convergence behaviour inside the Brillouin sphere, which can be identified as the sphere circumscribing the body. Evaluating the gravity potential of an arbitrary object through SH inside craters, valleys and tori, may produce grossly incorrect results, making this approach unreliable for dynamical environment modeling when performing Touch-and-Go (TAG) or landing on irregular bodies. Takahashi and Sheeres (Takahashi et al. (2013b)) recently presented a method to extend the convergence region within the Brillouin sphere by relying on interior spherical harmonics expansion. Although the series convergence issue is brilliantly tackled, the degree of complexity introduced does not make this method appealing for practical implementations of on-board guidance algorithms. It's worth mentioning that another deficiency of the SH approach is the lack of a cheap and systematic check condition to discriminate between points inside and outside the volume of the body.

2.2. Mass concentrations

Mass Concentrations (MASCONS) offer an alternative to model the gravity field of an object. The use of localized finite concentrated masses, distributed in grids over a volume to realize the entire mass of the body, has been around since the first satellite geodesy applications in the early 1970s (Weightman (1967), Russell and Arora (2012b)). Original formulations involved combination of global SH solutions and accurate local MASCONS descriptions. Indeed, it's common practice to use refined mass elements to describe accurately local gravitational effects along a reference path, while the mass elements distribution is coarser anywhere else.

The MASCONS approach converges to the actual gravitational potential of the mass distribution considered, but it is in general less accurate than SH, given the same computational effort. Because of its natural structure, MASCONS is well suited for parallel computation implementations. But despite the simplicity and the attractive convergence properties, an accurate estimate of the gravity force is generally affected by large errors, and requires a very high number of mass elements, as documented in Werner and Scheeres (1996b). Analogously to the SH approach, MASCONS does not provide information on the a point field being inside or outside the body.

2.3. Polyhedron model

In 1996 R.A. Werner et al. presented a method to use polyhedral models in order to determine in closed form the gravitational field of an arbitrary object. A polyhedral model consists of a number of triangular plates that approximate the shape of the celestial body, as accurately as possible, depending on the number of facets considered.

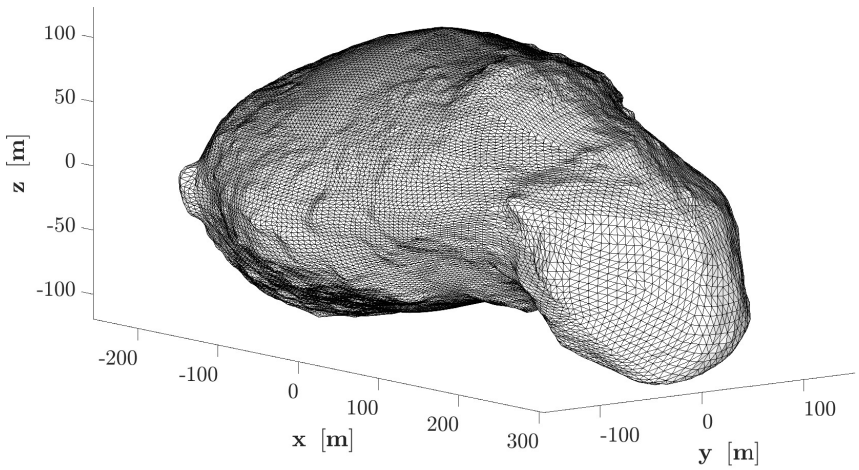


Figure 1: Low resolution polyhedron model of 25143 Itokawa. Low resolution model of asteroid 25143 Itokawa from Gaskell et al. (2008), with 49152 triangular plates.

As discussed in reference Werner and Scheeres (1996b), using the divergence theorem and green's identities, the gravity potential and the attraction can be expressed as a summation of integrals over the number of facets of the target. In the particular case of a polyhedron, the exact analytical solution does not involve numerical integral but can be expressed as a pure summation, including

logarithms and inverse tangents as most complex operations:

$$\boldsymbol{\nabla} U(\boldsymbol{r}) = \sigma G \left(- \sum_{e \in edges} \boldsymbol{E}_e \boldsymbol{r}_e L_e + \sum_{f \in faces} \boldsymbol{F}_f \boldsymbol{r}_f \omega_f \right) \tag{2}$$

Expressions of matrices $\boldsymbol{E}_e$, $\boldsymbol{F}_f$ and scalars L_e and ω_f are formulated in Werner and Scheeres (1996b). This suggests that, although very simple, such a strategy suffers from high computational demands when very accurate shape models, with a number of facets in the order of millions, are considered. The analytical expression (2) is the exact solution of a uniform polyhedron with given density, and the better the actual shape is resembled by the facets, the better Equation (2) approximates the actual field and it is as close as the actual field as the accurate is the shape approximation. Unlike SH approach, the polyhedron method can be used well within the exterior Brilluoin sphere up to the very surface of the target, without incurring undesired large misestimations. In fact, the exact gravity potential solution holds also in proximity of those morphologically peculiar regions such as ridges, valleys and even tori. Contrary to SH and MASCONS, the polyhedron model also provides an effective and cheap way to distinguish points that are inside or outside the target body. In fact, as discussed in reference Werner and Scheeres (1996b) , the discriminant between points inside, on the surface and outside the polyhedron is the value of the Laplacian of the gravity potential $\nabla^2 U$, which is a function of only the same quantities required to compute the local gravity field in the same position:

$$\nabla^2 U(\boldsymbol{r}) = -\sigma G \sum_{f \in faces} \omega_f \tag{3}$$

The tricky and elegant derivation of the closed form expression of the polyhedron gravity potential, attraction, and Laplacian is discussed in detail in reference Werner and Scheeres (1996b).

3. Machine Learning Methodology

In this section, we describe the proposed methodology and approach to model the functional relationship between spacecraft position and gravitational acceleration of small bodies. Importantly, the ELM theories are employed to train a SLFN that quickly and accurately computes the gravitational acceleration exibithed by a specified asteroid or comet. Importantly, the bayesian optimization framework is employed to optimize the network hyperparameters (e.g. number of neurons, regularization parameter) to maximize the network performance.

3.1. Extreme Learning Machines

In the last decades, Extreme Learning Machines (ELM) have been proposed as an innovative method to overcome many of the challenges presented by the conventional machine learning algorithms (Support Vector Machines, Neural

Table 1: Overview of activation functions.

Activation Functions		
Name	Expression	Output range
Step	$h(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases}$	$x = 0 \ \vee \ x = 1$
Sigmoid	$h(x) = \frac{1}{1+e^{-x}}$	$x \in (0, 1)$
Hyperbolic tangent	$h(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$	$x \in (-1, 1)$
Arctangent	$h(x) = \arctan(x)$	$x \in (-\frac{\pi}{2}, \frac{\pi}{2})$
ReLu	$h(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases}$	$x \in [0, \infty)$
Gaussian Function	$h(x) = e^{-x^2}$	$x \in (0, 1]$

Nets, Huang (2015)). It has been shown that ELM can achieve better generalization capabilities and learning scalability without the requirement of human intervention and with much less computational effort, with learning times reduced by factor of thousands if compared with traditional learning algorithms (Huang et al. (2012)). ELM were originally developed for SLFNs, and then extended to "generalized" SLFNs, which may not be neuron-alike (Duan et al. (2016)). The idea that learning can be achieved without iteratively tuning highly-specialized neurons of a network (which is recently confirmed by evidences in the field of biological neural systems) is the base onto which modern ELM theories have been built (Fusi et al. (2016)). It has been proven by Huang et al. (2012) that if a SLFN with tunable hidden nodes' parameters can learn a regression of a target function $\boldsymbol{f}(\boldsymbol{x})$, then, if the hidden nodes' activation functions $h_i(\boldsymbol{x}, \boldsymbol{w}_i, b_i)$, $i = 1 \dots L$ are non-linear piecewise continuous, training of the network does not require tuning of those parameters. Examples of commonly used activation functions are reported in Table 1. This means that input weights $\boldsymbol{w}_i$ and biases b_i of hidden nodes can be assigned randomly, and a SLFN will still maintain the property of universal approximator, as long as the output weights $\boldsymbol{\beta}_i$ are calculated properly. This proposition, which is the cardinal principle of ELM, has been formalized by Huang et al. (2012):

Theorem 1. Given any non-constant piecewise continuous function $h : \mathbb{R} \mapsto$

$\mathbb{R}$, if $\text{Span}\{h(\boldsymbol{x}, \boldsymbol{w}, b) : (\boldsymbol{w}, b) \in \mathbb{R}^d \times \mathbb{R}\}$ is dense in L^2, for any continuous target function f and any function sequence $\{h_i(x) = h(\boldsymbol{x}, \boldsymbol{w}_i, b_i)\}$, $i = 1, \ldots, L$ randomly generated based on any continuous sampling distribution, $\lim_{L \to \inf} \|f - f_L\| = 0$ holds with probability 1 if the output weights β_i are determined by ordinary least square to minimize $\|f(\boldsymbol{x}) - f_L(\boldsymbol{x})\|$.

Expanding to the case of N training set inputs $\{\boldsymbol{x}_j\}$ with $j = 1 \ldots N$, one can write the N output equations of the SLFN as:

$$\boldsymbol{y}_j = \sum_{i=1}^{L} \boldsymbol{\beta_i} h_i(\boldsymbol{x}_j, \boldsymbol{w}_i, b_i) \qquad \text{for} \quad j = 1 \ldots N \tag{4}$$

with $\boldsymbol{\beta}_i \in \mathbb{R}^{m_{out} \times 1}$ and $\boldsymbol{w}_i \in \mathbb{R}^{m_{in} \times 1}$, where m_{in} is the dimension of the input and m_{out} is the size of the output. A training process in which all the data in the training set are presented at the same time to the learner is called *batch learning*. In Figure 2 is depicted the SLFN presented with a batch of N examples, and the connections between input, hidden and output layer.

Let's define the hidden layer matrix of the network as $\boldsymbol{H}$, whose i-th column

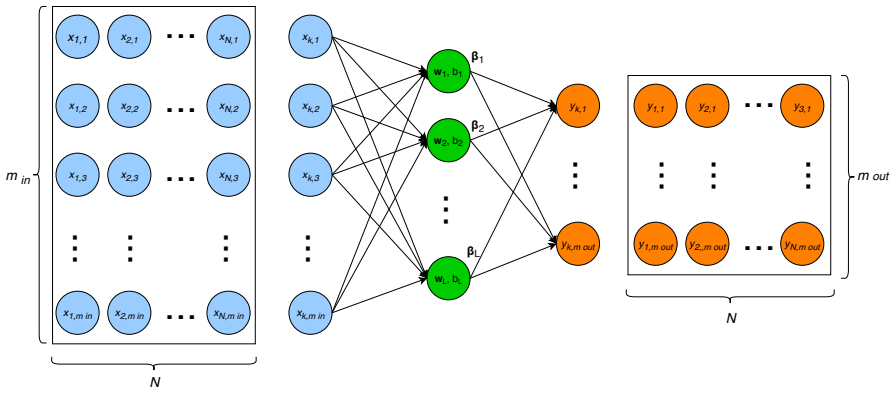


Figure 2: N observations $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}$ are available to the learner include input data ($\boldsymbol{x}_i$ on the left) and output data ($\boldsymbol{y}_i$ on the right) and the parameters of the ELM is adjusted (green in center) to match the data. The output of the learner is, for each of the inputs, expressed by Equation 4.

is the output of the i-th hidden node with respect to the set of inputs $\boldsymbol{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N] \in \mathbb{R}^{m_{in} \times N}$:

$$\boldsymbol{H} = \begin{bmatrix} h(\boldsymbol{x_1}, \boldsymbol{w_1}, b_1) & \ldots & h(\boldsymbol{x_1}, \boldsymbol{w_L}, b_L) \\ \vdots & \ddots & \vdots \\ h(\boldsymbol{x_N}, \boldsymbol{w_1}, b_1) & \ldots & h(\boldsymbol{x_N}, \boldsymbol{w_L}, b_L) \end{bmatrix}, \quad \boldsymbol{H} \in \mathbb{R}^{N \times L} \tag{5}$$

It is possible to write in compact matrix formulation the outputs of the SLFN when the set $\boldsymbol{X}$ is processed through the hidden layer, by gathering $\boldsymbol{y}_i$, $i =$

$1, \dots, N$ as the columns of $\boldsymbol{Y} \in \mathbb{R}^{m_{out} \times N}$

$$\boldsymbol{f_L}(\boldsymbol{X}) = \boldsymbol{Y} = \boldsymbol{H}\boldsymbol{\beta} \quad \text{with} \qquad \boldsymbol{\beta} = \begin{bmatrix} \boldsymbol{\beta_1}^T \\ \vdots \\ \boldsymbol{\beta_L}^T \end{bmatrix}, \ \boldsymbol{\beta} \in \mathbb{R}^{L \times m_{out}} \tag{6}$$

The training algorithm of ELM is aimed at the minimization of the cost functional E representing the training error of the SLFN.

$$E = \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2, \quad \text{with} \quad \boldsymbol{T} = \begin{bmatrix} \boldsymbol{t_1}^T \\ \vdots \\ \boldsymbol{t_N}^T \end{bmatrix}, \ \boldsymbol{T} \in \mathbb{R}^{N \times m_{out}} \tag{7}$$

Where $\boldsymbol{T}$ is the collection of the supervisor responses. According to Theorem 1, and for the most common case in which the number of nodes in the SLFN is lower than the number of data to fit, the trained network is a universal approximator if $\boldsymbol{\beta}$ are assigned according to the Least Square Error (LSE) of the overdetermined linear system:

$$\boldsymbol{H}\boldsymbol{\beta} = \boldsymbol{T} \tag{8}$$

The necessary and sufficient condition to have the solution $\overline{\boldsymbol{\beta}}$ with minimum L_2 norm among all least squares solutions is to evaluate $\boldsymbol{\beta}$ using the Moore-Penrose generalized inverse of the hidden layer matrix $\boldsymbol{H}$ (Serre (2000)), Casella et al. (2007) . Accordingly, the ELM training algorithm for SLFN can be written as:

$$\overline{\boldsymbol{\beta}} = \boldsymbol{H}^{\dagger}\boldsymbol{T} \qquad \boldsymbol{H}^{\dagger} = (\boldsymbol{H}^{\boldsymbol{T}}\boldsymbol{H})^{-1}\boldsymbol{H}^T \tag{9}$$

As suggested by the works of Bartlett (Bartlett (1997), Bartlett (1998)), the choice of $\overline{\boldsymbol{\beta}}$ as the set of smallest output weights satisfying the least squares condition ensures good generalization performance of the network when the number of hidden nodes is high. As it can be observed by considering Equation 9, the ELM training algorithm does not require iterative tuning of any parameter, but it just consists of the expensive evaluation of the pseudo inverse of $\boldsymbol{H}$, usually obtained via Singular Value Decomposition (SVD) with a huge computational complexity $\mathcal{O}(NL^2)$. However, as discussed in Huang et al. (2012), ELM approach allows reduction of the effort required for the training of a SLFN as compared with iterative, gradient-based algorithms with the same accuracy performances, while, at the same time, the capability of prediction on real world data is increased by choosing the smallest output weights $\boldsymbol{\beta}$. This training paradigm is focused on the efficiency in performing well on the training data, but it can lead to *overfitting*, i.e. poor generalization capabilities of the regressor: even if a perfect training with zero error can be accomplished, discrepancies on the "never seen" test set could be unacceptably high. The ELM training algorithm (Equation 9) is then said to be a realization of the *Empirical Risk Minimization (ERM)* (Vapnik (1992)) induction principle, meaning that the training is aimed only at the minimization of a risk functional, in this case the squared norm of the errors, depending only on the known input-response data.

3.2. Regularized Extreme Learning Machines

As opposed to ERM, *Structural Risk Minimization (SRM)* theory is based on the simultaneous minimization of a risk functional considering two terms: a term directly related to the empirical risk, and a further term concerning the complexity of the learning machine (Vapnik (1992)). It's proven in Bartlett (1997) that ELM generalization capabilities can be enhanced if a further term controlling the magnitude of the output weights $\boldsymbol{\beta}$, called *regularization* term, is combined to the empirical risk functional. Huang et al. (2012), provided a detailed demonstration of regularization techniques applied to ELM, and showed the efficiency of the above mentioned approach.
The regularized ELM training algorithm is then aimed at the minimization of the following risk functional:

$$E = \sum_{i=1}^{N} \|\boldsymbol{\xi}_i\|^2 + K \|\boldsymbol{\beta}\|^2 \tag{10}$$

or, equivalently and in compact form:

$$\tilde{E} = \frac{1}{2} C \|\boldsymbol{H}\boldsymbol{\beta} - \boldsymbol{T}\|^2 + \frac{1}{2} \|\boldsymbol{\beta}\|^2 \tag{11}$$

with $C = \frac{1}{K}$ and subject to the training residuals equality constraints:

$$\xi_{i,j} = \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta}_j - t_{i,j} \qquad i = 1, \ldots, N \quad j = 1, \ldots, m_{out} \tag{12}$$

where $\xi_{i,j}$ is the j-th component of the residual error on the i-th training sample. The quantity $\boldsymbol{h}(\boldsymbol{x}_i)$ represents the hidden layer output corresponding to the input $\boldsymbol{x}_i$, $\boldsymbol{\beta}_j$ is the output weights vector linking the hidden layer to the j-th output node, and $t_{i,j}$ is the j-th component of the i-th sample output. The *regularization factor* K is a positive number expressing the importance of the minimization of the empirical risk with respect to the magnitude of the output weights $\boldsymbol{\beta}$.
Minimization of $\tilde{E}$ in Equation 11 is equivalent to solving the optimization problem with lagrangian:

$$L_{\tilde{E}}(\boldsymbol{\beta}, \boldsymbol{\xi}, \boldsymbol{\lambda}) = C\frac{1}{2}\sum_{i=1}^{N} \|\boldsymbol{\xi}_i\|^2 + \frac{1}{2}\|\boldsymbol{\beta}\|^2 - \sum_{i=1}^{N}\sum_{j=1}^{m_{out}} \lambda_{i,j}(\boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta}_j - t_{i,j} + \xi_{i,j}) \tag{13}$$

The quantity $\lambda_{i,j}$ is the Lagrange multiplier associated with the j-th component of the i-th training sample. The optimality conditions are:

$$\frac{\partial L_E}{\partial \boldsymbol{\beta}_j} = 0 \;\rightarrow\; \boldsymbol{\beta}_j = \sum_{i=1}^{N} \lambda_{i,j}\boldsymbol{h}(\boldsymbol{x}_i)^T \;\rightarrow\; \boldsymbol{\beta} = \boldsymbol{H}^T\boldsymbol{\lambda} \tag{14}$$

$$\frac{\partial L_E}{\partial \boldsymbol{\xi}_i} = 0 \;\rightarrow\; \boldsymbol{\lambda}_i = C\boldsymbol{\xi}_i \quad i = 1, \ldots, N \tag{15}$$

$$\frac{\partial L_E}{\partial \boldsymbol{\lambda}_i} = 0 \;\rightarrow\; \boldsymbol{h}(\boldsymbol{x}_i)\boldsymbol{\beta} - \boldsymbol{t}_i^T + \boldsymbol{\xi}_i^T = 0 \quad i = 1, \ldots, N \tag{16}$$

with $\boldsymbol{\lambda}_i = [\lambda_{i,1}, \ldots, \lambda_{i,m_o ut}]^T$ and $\boldsymbol{\lambda} = [\boldsymbol{\lambda}_1, \ldots, \boldsymbol{\lambda}_N]^T$. Based on the number of nodes in the hidden layer and the size of the training sets, different efficient solutions are found to the above mentioned problem:

- The number of training samples is not huge: $N \approx L$
  Substituting Equations 14 and 15 in 16:

$$\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^T\right)\boldsymbol{\lambda} = \boldsymbol{T} \tag{17}$$

  Multiplying by $\boldsymbol{H}^T$ and considering Equation 14, the training algorithm of the regularized ELM can be written as:

$$\boldsymbol{\beta} = \boldsymbol{H}^T\left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}\boldsymbol{H}^T\right)^{-1}\boldsymbol{T} \tag{18}$$

- The number of training samples is huge: $N \gg L$
  From Equations 14 and 15:

$$\boldsymbol{\beta} = C\boldsymbol{H}^T\boldsymbol{\xi} \;\rightarrow\; \boldsymbol{\xi} = \frac{1}{C}(\boldsymbol{H}^T)^{\dagger}\boldsymbol{\beta} \tag{19}$$

  and substituting in Equation 16 and rearranging leads to the expression of the output weights $\boldsymbol{\beta}$:

$$\boldsymbol{\beta} = \left(\frac{\boldsymbol{I}}{C} + \boldsymbol{H}^T\boldsymbol{H}\right)^{-1}\boldsymbol{H}^T\boldsymbol{T} \tag{20}$$

  which corresponds to the solution of m_{out} linear systems of dimension L.

3.3. Large Dataset Management: Online Sequential ELM

When dealing with big training sets with millions of data, ELM tends to become computationally extremely demanding, and most of the time, it becomes practically impossible to rely on the LSE solution (Equation 9) to train a SLFN. Two main causes can be identified.

- The number of elements of the *hidden layer matrix* $\boldsymbol{H}$ is $N \times L$. The memory used by the variable can grow very quickly, reaching orders of magnitude of $\sim 1\,TB$ for networks with $L \sim 10^5$ nodes and $N \sim 10^6$ observations in the dataset.

- The computational cost in terms of number of operations in order to solve the linear system in Equation 20 of dimension L is $\mathcal{O}(L^3)$. As L becomes high enough, the amount of CPU time required grows rapidly, resulting in extremely large training times which makes this approach not practical.

Recently, a variant of ELM has been developed for cases, such as time series predictions, in which not all training observations are available to the learner simultaneously. In such scenarios, the learning machine needs to implement a

modified algorithm in order to be able to learn from training samples presented one-by-one or chunk-by-chunk. Consider a SLFN with L hidden nodes trained using regularized ELM with an initial chunk of N_0 available training samples $\boldsymbol{X}_0 = [\boldsymbol{x}_1 \dots \boldsymbol{x}_{N_0}]$. Equation 20 including the regularization term can be written as:

$$\boldsymbol{\beta}_0 = (\boldsymbol{H}_0^T \boldsymbol{H}_0 + \frac{\boldsymbol{I}}{C}))^{-1} \boldsymbol{H}_0^T \boldsymbol{T}_0 = \boldsymbol{K}_0^{-1} \boldsymbol{H}_0 \boldsymbol{T}_0 \tag{21}$$

with

$$\boldsymbol{H}_0 = \begin{bmatrix} h(\boldsymbol{x_1}, \boldsymbol{w_1}, b_1) & \dots & h(\boldsymbol{x_1}, \boldsymbol{w_L}, b_L) \\ \vdots & \ddots & \vdots \\ h(\boldsymbol{x_{N_0}}, \boldsymbol{w_1}, b_1) & \dots & h(\boldsymbol{x_{N_0}}, \boldsymbol{w_L}, b_L) \end{bmatrix}, \quad \boldsymbol{H}_0 \in \mathbb{R}^{N_0 \times L} \tag{22}$$

$$\boldsymbol{T}_0 = \begin{bmatrix} \boldsymbol{t_1}^T \\ \vdots \\ \boldsymbol{t_{N_0}}^T \end{bmatrix}, \ \boldsymbol{T}_0 \in \mathbb{R}^{N_0 \times m_{out}} \quad \boldsymbol{\beta}_0 = \begin{bmatrix} \boldsymbol{\beta_1}^T \\ \vdots \\ \boldsymbol{\beta_{N_0}}^T \end{bmatrix}, \ \boldsymbol{\beta}_0 \in \mathbb{R}^{L \times m_{out}}$$

Now, assume that another chunk of N_1 observations becomes available to the learner, which now has to minimize the training error considering both the previous and the current chunks of data presented, by suitably evaluating the updated set of output weights $\boldsymbol{\beta}_1$:

$$L_1 = \left\| \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix} \boldsymbol{\beta}_1 - \begin{bmatrix} \boldsymbol{T}_0 \\ \boldsymbol{T}_1 \end{bmatrix} \right\|^2 \tag{23}$$

The LSE solution (Equation 9) can be written as:

$$\boldsymbol{\beta}_1 = \left(\begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix}^T \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix} \right)^{-1} \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix}^T \begin{bmatrix} \boldsymbol{T}_0 \\ \boldsymbol{T}_1 \end{bmatrix} = \boldsymbol{K}_1^{-1} \begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix}^T \begin{bmatrix} \boldsymbol{T}_0 \\ \boldsymbol{T}_1 \end{bmatrix} \tag{24}$$

with $\boldsymbol{K}_1$ satisfying the relation:

$$\boldsymbol{K}_1 = \boldsymbol{H}_0^T \boldsymbol{H}_0 + \boldsymbol{H}_1^T \boldsymbol{H}_1 = \boldsymbol{K}_0 + \boldsymbol{H}_1^T \boldsymbol{H}_1 \tag{25}$$

With some simple rearrangements, equation 25 can be rearranged:

$$\begin{aligned} \boldsymbol{\beta}_1 = \boldsymbol{K}_1^{-1} \left(\begin{bmatrix} \boldsymbol{H}_0 \\ \boldsymbol{H}_1 \end{bmatrix}^T \begin{bmatrix} \boldsymbol{T}_0 \\ \boldsymbol{T}_1 \end{bmatrix} \right) = \boldsymbol{K}_1^{-1} \left(\boldsymbol{H}_0^T \boldsymbol{T}_0 + \boldsymbol{H}_1^T \boldsymbol{T}_1 \right) = \\ = \boldsymbol{K}_1^{-1} \left(\boldsymbol{K}_0 \boldsymbol{K}_0^{-1} \boldsymbol{H}_0^T \boldsymbol{T}_0 + \boldsymbol{H}_1^T \boldsymbol{T}_1 \right) = \boldsymbol{K}_1^{-1} \left(\boldsymbol{K}_0 \boldsymbol{\beta}_0 + \boldsymbol{H}_1^T \boldsymbol{T}_1 \right) = \\ \boldsymbol{\beta}_0 + \boldsymbol{K}_1^{-1} \boldsymbol{H}_1^T \left(\boldsymbol{T}_1 - \boldsymbol{H}_1^T \boldsymbol{\beta}_0 \right) \end{aligned} \tag{26}$$

Since $\boldsymbol{K}_1^{-1}$ is used to evaluate the updated solution $\boldsymbol{\beta}_1$, it is convenient to use an update formula for $\boldsymbol{K}_1^{-1}$ itself, by means of Woodbury's formula (Golub and Van Loan (2012)):

$$\boldsymbol{K}_1^{-1} = \boldsymbol{K}_0^{-1} - \boldsymbol{K}_0^{-1} \boldsymbol{H}_1^T \left(\boldsymbol{I} + \boldsymbol{H}_1 \boldsymbol{K}_0^{-1} \boldsymbol{H}_1^T \right)^{-1} \boldsymbol{H}_1 \boldsymbol{K}_0^{-1} \tag{27}$$

This process can be generalized to the case of sequential learning when the $k+1$ chunk of examples is presented to the SLFN. The update formula for $\boldsymbol{\beta}$, which requires the update for $\boldsymbol{K}^{-1}$ as well, is then given by Equations 29, 28.

$$\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k + \boldsymbol{K}_{k+1}^{-1}\boldsymbol{H}_{k+1}^T\left(\boldsymbol{T}_{k+1} - \boldsymbol{H}_{k+1}^T\boldsymbol{\beta}_k\right) \tag{28}$$

$$\boldsymbol{K}_{k+1}^{-1} = \boldsymbol{K}_k^{-1} - \boldsymbol{K}_k^{-1}\boldsymbol{H}_{k+1}^T\left(\boldsymbol{I} + \boldsymbol{H}_{k+1}\boldsymbol{K}_k^{-1}\boldsymbol{H}_{k+1}^T\right)^{-1}\boldsymbol{H}_{k+1}\boldsymbol{K}_k^{-1} \tag{29}$$

The *innovation* term $\boldsymbol{T}_{k+1} - \boldsymbol{H}_{k+1}^T\boldsymbol{\beta}_k$ in Equation 28 can be seen as the residual when the output weights associated to the previous sequential learning are used: if this term is zero, then the outdated output weights perfectly fit the new chunk of observations, so $\boldsymbol{\beta}_{k+1} = \boldsymbol{\beta}_k$. An important feature of the recursive training is that each time a new chunk of N_{k+1} data is presented, the update of output weights $\boldsymbol{\beta}_{k+1}$ involves only terms evaluated at the current and previous sequential step. This means that the SLFN can be trained sequentially by presenting small chunks of data with $N_k \ll N$, thus reducing the amount of memory required to store big variables such as $\boldsymbol{H}$, and at the same time allowing for memory clearing at each step.

In fact, each time a chunk of training observations is presented to the machine, it is discarded as soon as $\boldsymbol{\beta}_{k+1}$ is evaluated, along with all the quantities related to the previous learning step, except for $\boldsymbol{\beta}_k$ and $\boldsymbol{K}^{-1}$. It is remarked that, comparing the algorithm for R-ELM (Equation 9) and the initial learning phase of OSELM described in Equation 21, it can be observed that when $rank(\boldsymbol{H}_0) = L$, ELM and OSELM can achieve the same learning performances. In order to have $rank(\boldsymbol{H}_0) = L$, the number of initialization data should not be less than the hidden node number ($N_0 \geq L$). Generally, OSELM suffers some drawbacks (Liang et al. (2006)):

- In general OSELM relying on Recursive Least Square Error (RLSE) training algorithm is less accurate than ELM, given the same amount of hidden nodes in the network. Best performances can be obtained if the number of observations in the initial chunk of data is equal to the number of neurons in the SLFN.
- If regression or classification can be performed on the same machine using the same set of training data, in general OSELM will be more computationally expensive than batch learning using classical R-ELM.

3.4. Performance indices

In order to assess the risk associated to predictions by the trained learning machine, several performance indices commonly used in the analysis of regression models have been considered in this study:

Mean Square Error (MSE): This scalar index is used to indicate the expected value of the square of the difference between the output of the

trained SLFN and the target response given by the the supervisor:

$$MSE = \frac{1}{N}\sum_{i=1}^{N} \|y_i - t_i\|^2 \tag{30}$$

where N is the number of elements included in the set considered. Values of MSE close to zero mean that the regressor model is able to fit the data efficiently.

Root Mean Square Error (RMSE): It is the square root of the mean square error and represents an the standard deviation of the residual between the prediction and the observation:

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|y_i - t_i\|^2} = \sqrt{MSE} \tag{31}$$

This quantity is an indication, on average, of how far the prediction of the SLFN is from the real observation.

Normalized Root Mean Square Error (NRMSE): It is a non-dimensional performance index suitable for the comparison of efficiency on sets characterized by targets in a wide magnitude range.

$$NRMSE = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \|y_i - t_i\|^2}}{\sqrt{\frac{1}{N}\sum_{i=1}^{N} \|t_i - \mu_t\|^2}} = \frac{RMSE}{\sigma_t} \tag{32}$$

where μ_t is the mean value of the targets and σ_t is the standard deviation of the targets. NRMSE is suitable to represent and compare prediction performances for very different targets

3.5. Bayesian Optimization

This section is aimed at giving a brief overview of Bayesian optimization, as a tool to perform automatic tuning of complex learning machines hyperparameters, a practice that is often left to be handled by human experience or brute force search. This global optimization approach is well suited for automatic setting of the architecture of the network in order to minimize an objective function $b = b(\boldsymbol{x})$ defined on a bounded subset $\Gamma \in \mathbb{R}^D$. This function usually does not have a simple closed form representation or it is too expensive to evaluate, and it is treated as an unknown deterministic or stochastic *"black-box"* which can be sampled at any point $\boldsymbol{x} \in \Gamma$. The algorithm consists of a sequential search that, at the k-th round, selects a location at which the objective function is queried. This observation, along with the previous ones, is used to infer a belief on the behaviour of the objective function. The process of sampling and update

is iterated until a specified stopping criterion is met: the algorithm returns the final recommendation of $\bar{\boldsymbol{x}}$ which minimizes $b(\boldsymbol{x})$, for $\boldsymbol{x} \in \Gamma$.

$$\bar{\boldsymbol{x}} = \underset{\boldsymbol{x} \in \Gamma}{\operatorname{argmin}}\, b(\boldsymbol{x}) \tag{33}$$

Differently from other optimization algorithms, the Bayesian approach does not rely on information about local derivatives of the objective function, but it is rather aimed at building and refining a model of $b(\boldsymbol{x})$ via Bayesian posterior update based on the entire set of samples acquired during previous sequential decision rounds. Bayesian optimization relies on *Gaussian processes (GP)* regression to fit the objective to the incoming observations, while the choice of the next query point is determined by an *acquisition function* evaluated on the refined model.

A GP is a stochastic process with the property that any finite dimensional collection of N random variables $\{b(\boldsymbol{x}_i), \boldsymbol{x}_i \in \Gamma\}$, $i = 1, \dots, N$, drawn from the GP, has a multivariate gaussian distribution, identified by a *mean* $\boldsymbol{\mu}(\boldsymbol{x}) : \Gamma \to \mathbb{R}^N$ and a *covariance matrix* $\boldsymbol{K} \in \mathbb{R}^{N \times N}$, whose components $k_{i,j}$, $i, j = 1 \dots, N$ map two elements $\boldsymbol{x}_i, \boldsymbol{x}_j \in \Gamma$ into $\mathbb{R}$. This property can be written as:

$$\begin{bmatrix} b(\boldsymbol{x}_1) \\ \vdots \\ b(\boldsymbol{x}_N) \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu(\boldsymbol{x}_1) \\ \vdots \\ \mu(\boldsymbol{x}_N) \end{bmatrix}, \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \right) \tag{34}$$

The notation $\boldsymbol{f} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K})$ is used to state that $\boldsymbol{f}$ has multivariate gaussian distribution. Referring to Equation 34, it is natural to extend the property to collections of random variables with $N \to \infty$: in this case $\boldsymbol{b}$ is seen as an extremely high dimensional vector, drawn from an extremely-high multivariate distribution. The elements of the huge vector $\boldsymbol{b}$ represent the value of the objective variable given the infinite set of inputs $\boldsymbol{x}_i$: intuitively it can be seen that $b(\boldsymbol{x})$ is a function, drawn from the GP characterized by a *mean function* $\mu(\boldsymbol{x}) : \Gamma \mapsto \mathbb{R}$ and a *covariance function* or *kernel* $k(\boldsymbol{x}, \boldsymbol{x}) : \Gamma \times \Gamma \mapsto \mathbb{R}$. Gaussian processes are then the generalization of Gaussian distributions: whereas distributions describe vectorial or scalar random variables, processes governs the properties of functions (Williams and Rasmussen (2006)). Equation 34 can be written in compact form as:

$$b(\boldsymbol{x}) \sim \mathcal{GP}(\mu, k) \tag{35}$$

meaning that it is assumed that the unknown objective b is a Gaussian Process (GP). At the beginning of the routine to find the minimum of $b(\boldsymbol{x})$, there is no empirical information about this function. Before gaining samples, only *a priori* beliefs can be inferred in terms of mean and covariance functions of the so-called *prior* process. Since any real valued $\mu(\boldsymbol{x})$ may give rise to an acceptable GP, it is assumed, without loss of generality, that $\mu_0(\boldsymbol{x}) = 0$ over the entire bounded domain Γ. On the other hand, more care should be taken in the choice of the *kernel* k, which must be such that, given any set of $\{\boldsymbol{x}_i, \dots, \boldsymbol{x}_N\}$, the

covariance matrix $\boldsymbol{K}$ is positive definite, where $\boldsymbol{K}$ is:

$$\boldsymbol{K} = \begin{bmatrix} k(\boldsymbol{x}_1, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_1, \boldsymbol{x}_N) \\ \vdots & \ddots & \vdots \\ k(\boldsymbol{x}_N, \boldsymbol{x}_1) & \cdots & k(\boldsymbol{x}_N, \boldsymbol{x}_N) \end{bmatrix} \tag{36}$$

The other role of the prior covariance k is to express our belief on the correlation (i.e. the smoothness) of the unknown objective function: in practice this means that, as it is reasonable to assume in many applications, the kernel should represent the belief that similar inputs produces similar outputs of the objective function. Among the many kernel functions available in the literature, Matlab built-in Bayesian Optimization routine `bayesopt` uses by default the kernel function automatic relevant determination (ARD) Matèrn 5/2 (Williams and Rasmussen (2006)):

$$k(\boldsymbol{x}_i, \boldsymbol{x}_j, \boldsymbol{\theta}) = \sigma_f^2 \left(1 + \frac{\sqrt{3}\, r}{\sigma_l}\right) exp\left(-\frac{\sqrt{3}\, r}{\sigma_l}\right) \tag{37}$$

parametrized on the components of the set of parameter $\boldsymbol{\theta}$, which for the Matèrn 5/2 case are σ_l and σ_f representing a characteristic length scale and the signal standard deviation, respectively. The quantity r is defined as:

$$r = \sqrt{(\boldsymbol{x}_i - \boldsymbol{x}_j)^T (\boldsymbol{x}_i - \boldsymbol{x}_j)}$$

Once the prior process is assumed, the first samples $\{\boldsymbol{x}_k,\, y_k\}$ of the objective function are acquired: a Bayesian update procedure is implemented to select the process conditioned on the newly arrived sample by adjusting the kernel parameters θ. The Matlab built-in function `fitrgp`, called by `bayesopt` is used to fit the gaussian process to the objective function samples. Assuming that the samples are affected by a noise with variance σ^2, the *posterior* Q, i.e. the distribution conditioned on the evidences $\{\boldsymbol{x}_k,\, y_k\}$, will also be a GP. The mean function of the posterior is interpreted to be the best fit to the data, while the posterior covariance function is representative of uncertainty on the fit of the data.

Among the many acquisition functions that can be found in the literature, the algorithm considered in the case studies presented in this document employs the family of the *expected improvements* $a_{EI}(\boldsymbol{x})$. The expected improvement considers the expected amount of improvement in the objective function, ignoring values of the optimization variables that cause an increase in the objective. Supposing that the best function observed so far is b_{best} evaluated at $\boldsymbol{x}_{best}$, let's define the utility function $u(\boldsymbol{x})$ defined as:

$$u(\boldsymbol{x}) = \max(0, b_{best} - b(\boldsymbol{x})) \tag{38}$$

A reward equal to $b_{best} - b(\boldsymbol{x})$ is then gained only if $b(\boldsymbol{x})$ is less that b_{best},

otherwise it is null. The expected improvement is then defined as $a_{EI}(\boldsymbol{x}, Q_k)$:

$$a_{EI}(\boldsymbol{x}, Q_k) = \mathbb{E}_{Q_k}[u(\boldsymbol{x})|\boldsymbol{x}, \mathcal{D}_k] = \int_{-\infty}^{b_{best}} (b_{best} - b(\boldsymbol{x}))\mathcal{N}(b(\boldsymbol{x}); \mu(x), k(x,x)))db$$
$$= (b_{best} - \mu(x))\phi(b_{best}; \mu, k(\boldsymbol{x}, \boldsymbol{x})) + k(\boldsymbol{x}, \boldsymbol{x})\mathcal{N}(b_{best}; \mu(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}))) \quad (39)$$

where $\boldsymbol{x}_{best}$ is the location corresponding to the minimum mean of the posterior observed so far. The next query point, i.e. the $k+1$ observation location of the Bayesian optimization algorithm is then selected according to:

$$\boldsymbol{x}_{k+1} = \underset{x}{\text{argmax}}\ a_{EI}(\boldsymbol{x}, Q_k) \quad (40)$$

Considering Equation 39, it can be seen that a_{EI} is a combination of two terms: an *exploitative* term, guiding the choice of the next observation where the mean prediction is low and an *explorative* term which make regions with high uncertainty attractive for the next query. This *exploration-exploitation tradeoff* characterizes the family of expected improvement acquisition functions. The process of sampling, prior model updating to a posterior, and acquisition function evaluation is repeated for each sampling round, until the maximum number of observations is reached, or until the difference between consecutive recommendations $\boldsymbol{x}$ becomes smaller than a fixed threshold.

4. Results: Gravity field modeling

In this section, we report the design, training, validation and the performance obtained while applying ELM theories combined with bayesian optimization to the problem of modeling small bodies gravity field. To this end, we consider two test cases, i.e. asteroid Itokawa and comet 67P/Churyumov-Gerasimenko, and we employ high-fidelity polyhedron models to accurately generate training points representative of the functional relationship between position and corresponding gravitational acceleration generated by the small body. Importantly, for both cases, we consider the ability of the trained SLFN to compute the gravitational acceleration as part of a potential on-board, energy-optimal, closed-loop guidance system for landing on the small body surface (Hawkins et al. (2012)).

4.1. ELM-based SLFN for Asteroid Itokawa

In this first case, we focus on asteroid Itokawa and we show the design, training and validation of a SLFN capable of computing the gravitational acceleration for the above mentioned small body. Importantly, we evaluate accuracy and computational training times for both global and localized gravity modeling. The localized version of the forward network has been employed in a simulated guided landing scenarios where the feedback generalized Zero-Effort-Miss/Zero-Effort-Velocity (ZEM/ZEV) algorithm (Guo et al. (2013); Hawkins et al. (2012)) employs the trained SLFN to propagate the trajectory on-board.

The polyhedron model has been employed to generate the training set. More specifically, the shape models, derived by R. Gaskell (Gaskell et al. (2008)) and used to generate the datasets for global and local 25143 Itokawa gravity field, are based on Hayabusa AMICA (Asteroid Multi-Band Imaging Camera) images acquired between September 11th and November 12th, 2005. These shape models are derived from the Implicitly Connected Quadrilateral (ICQ) models, and they have been adjusted to suit an easy management in MATLAB®, by composing a matrix $\boldsymbol{V} \in \mathbb{R}^{N_V \times 3}$ whose rows contain the body frame coordinates of each of the vertices. The row number in $\boldsymbol{V}$ represents the vertex ID. Each shape model is also comprised of a connectivity matrix $\boldsymbol{F} \in \mathbb{R}^{N_F \times 3}$ that represents how the facets are composed, by indicating on each row the ID of the three vertices that, connected, identify the triangular plate. All models are expressed in body fixed reference frame and describe a set of triangular plates that approximate the asteroid surface, with different resolutions available (Gaskell et al. (2008)).

A first attempt to learn the global mapping from the body-fixed Cartesian coordinates to the local gravitational attraction field vector, within a spherical region surrounding the body, was performed using a low-resolution, constant-density polyhedron model with $49,152$ plates. A higher-fidelity polyhedron model employing a higher resolution model with $3,145,728$ triangular faces is also considered to accurately model the gravitational acceleration in a localized portion of the asteroid space. For practical reasons, due to the high computational effort of the Polyhedron algorithmic routine using such a high number of faces and vertices, the landing guidance study is limited to the inside of a cylindrical region broad enough that all the landing trajectories considered are well within it.

4.1.1. Global gravity model for 25143 Itokawa

In order to train a SLFN to learn the functional relationship between positions about 25143 Itokawa and the corresponding gravitational acceleration, a dataset of $N = 10^7$ input-response $\{\boldsymbol{r}_i, \boldsymbol{g}(\boldsymbol{r}_i))\}$, $i = 1, \ldots, N$ pairs has been assembled. The field points $\boldsymbol{r}_i$ have been sampled from a uniform distribution within a sphere containing the entire asteroid, with radius $R_{sphere} = 670\,m$. The general process of evaluating the k-th sample consist of three steps:

Step 1: The k-th field point position $\boldsymbol{r}$ is randomly sampled within the mentioned sphere.

Step 2: The constant density polyhderon gravity $\boldsymbol{g}(\boldsymbol{r})$ is obtained by solving Equation 2

Step 3: Using all the necessary quantities already calculated in the previous step, a test criterion is employed to discard the samples if its location is inside the asteroid surface.
This is executed by evaluating the Laplacian of the gravity potential $U(\boldsymbol{r})$:

$$\nabla^2 U(\boldsymbol{r}) = -\sigma G \sum_{f \in faces} \omega_f \tag{41}$$

As discussed in Werner and Scheeres (1996b) and reported in reference Werner and Scheeres (1996b), whenever the field point lies within the polyhedron, summation in Equation 41 is equal to 4π, whereas on the outside region of the polyhedron, the Laplace's equation holds, and $\nabla^2 U(\boldsymbol{r}) = 0$. Therefore, the following cases can occur:

$$\nabla^2 U(\boldsymbol{r}) = \begin{cases} -4\pi\sigma G, & \text{if } \boldsymbol{r} \text{ inside} \\ 0, & \text{if } \boldsymbol{r} \text{ outside} \end{cases} \tag{42}$$

The shape is initially approximated by a low resolution polyhedron model (Gaskell et al. (2008)). The density of the body is assumed uniform and equal to be $\rho = 1.9 \times 10^3\, kg/m^3$ (Ostro et al. (2004)). Using a single computational node of the University of Arizona Ocelote cluster, with 28 core processors Intel® Xeon® E5-2695, with a speed of 2.3 GHz and 6 GB RAM each, the mean time to get a single sample is $T_{mean} \approx 3.5\, s$. Overall, the CPU time required to build the entire dataset is approximatively $T_{CPU,dataset} \approx 9800\, h$. The observations are randomly partitioned into a training set and a test set, containing respectively 90% and 10% of the whole amount of data.

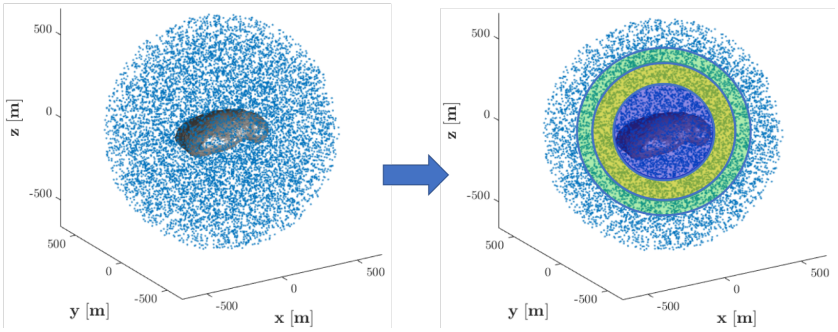


Figure 3: Training points selected for the global modeling of Itokawa gravitational acceleration. The rightmost panel shown the partition strategy employed to implement sequential training via OSELM

Because of the huge number of training points (i.e. 10 million), the SLFN training via regularized ELM batch learning would be extremely expensive both in terms of training time and in terms of required computational resources. Consequently, we developed a strategy to make the computational time manageable. Such strategy consists of a sequential training approach using the OSELM algorithm, for a SLFN with sigmoid hidden nodes. The whole amount of field points in the training set is sub-sequentially partitioned according to their position in concentric spherical shells, each one containing the same number of $N_{shell} = 500,000$ points and with the same data density (see Figure 3). The initial training is performed using $100,000$ samples in the inner shell, which contains points closest to the asteroid surface. Once the initial training is completed, the next chunk of data rendered available to the learner is taken from

the outer shell. This sequential shells strategy continues as more samples are provided, moving outward until the last field points in the outermost shell are used. The network architecture has been chosen according to the best estimated testing accuracy reachable after the initial training set, returned by the Bayesian optimization routine. This choice has been guided by the assumption that generalization capability in the region closest to the surface, where the functional relation $\boldsymbol{g}(\boldsymbol{r})$ show the most irregular behavior (Figure 5), has the strongest impact on the performances over the whole spherical region described above. The bounded variables of the optimization are:

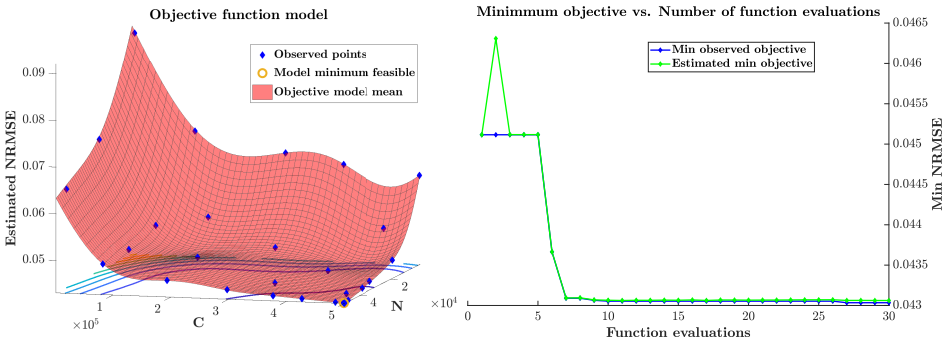
- The number of nodes in the hidden layer: $L \in \left[10^4;\, 5 \times 10^4\right] \in \mathbb{N}$
- The regularization factor: $C \in \left[10^4;\, 5 \times 10^5\right] \in \mathbb{R}$

Among the many performance indices to express the goodness of a regression performed by a learning machine, the objective function of the Bayesian optimization has been set to be the 10-fold cross-validation loss in terms of Normalized Root Mean Square Error (NRMSE), defined in Section 3.4. Since the supervisor response to an input position $\boldsymbol{r}$ in this case is a vector $\boldsymbol{g} \in \mathbb{R}^3$, the definition of NRMSE given in Equation 32, returns three values $NRMSE_{test,j}$, with $j = 1to3$, one for the regression of each of the components. Without favoring the minimization of mispredictions of any of these components, the objective function $b(C, L)$ is considered to be simply the mean of these three values:

$$b(C, L) = NRMSE_{test} = \frac{1}{3} \sum_{j=1}^{3} NRMSE_{test,j} \tag{43}$$

The stopping criterion is fixed in terms of maximum of function evaluations N_{max}, with $N_{max} = 30$. This is shown to be a safe value since the minimum objective function estimation doesn't change significantly after the tenth sampling, as it is possible to observe in Figure 20.
Once the optimizer returns its recommended $\bar{\boldsymbol{x}}$ and the SLFN is trained with the whole dataset, the actual generalization capability can be checked on the test set, which is unseen during the entire optimization process.

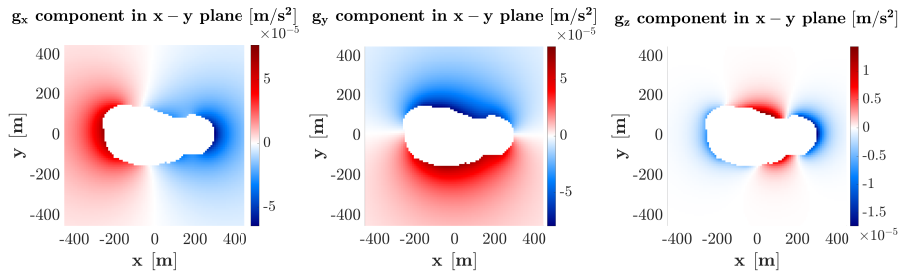
Figure 20 also shows that the actual minimum test NRMSE is found for values of $\bar{C} = 5 \times 10^5$ and $\bar{L} = 5 \times 10^4$ at the very edge of the searching interval. This suggests that further increasing the maximum number of nodes would lead to better performances of the net, without the occurrence of overfitting. However, this would result in a very complex learning machine, whose training would be extremely expensive both in terms of time and in terms of computational resources required, without a significant or worthy improvement on the objective function estimated minimum. Results of training and test regression performed with the SLFN trained with OSELM are reported in Figures 6 and 7, respectively. It can be seen that the learner is able to predict accurately the gravitational acceleration both on the training data and on the never-seen field points of the test set. It can be seen that the learner is able to predict accurately the gravitational acceleration on the training data, with an achieved



(a) Objective model (b) Minimum observed and estimated objective

Figure 4: 25143 Itokawa: Bayesian optimization results. Results of Bayesian optimization of the SLFN for the global gravity modeling of Itokawa. Figure (a): $NRMSE_{test}$ mean prediction returned by the optimizer after 30 evaluations as a function of the regularized ELM hyperparameters, along with the observations and the recommended $\bar{\boldsymbol{x}} = [\bar{C}, \bar{L}]$. Figure (b): the observed and estimated minimum objective function as the number of observations increases.

$NRMSE_{train} = 0.0434$ and $RMSE_{train} = 1.994 \times 10^{-7}\, m/s^2$. Good performances are also observed on the test set samples, with values of $NRMSE_{test} = 0.0440$ and $RMSE_{test} = 2.007 \times 10^{-7}\, m/s^2$, as indicating that the SLFN is able to efficiently generalize.

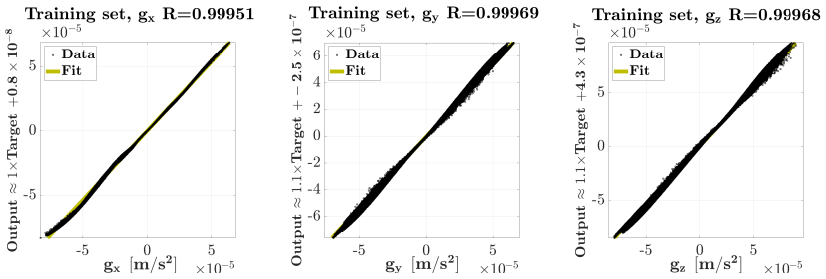


(a) (b) (c)

Figure 5: Gravity attraction field about 25143 Itokawa, Figures (a),(b) and (c) represent the gravity acceleration components g_x, g_y and g_z respectively, in the x-y plane. The value of the gravity attraction is evaluated using the low resolution polyhedron model Gaskell et al. (2008)

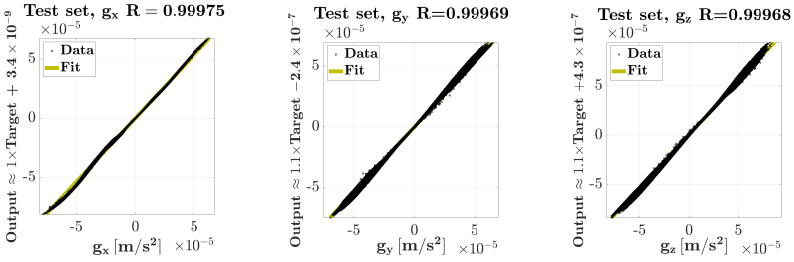
4.1.2. Local gravity model for 25143 Itokawa

In this section, we employ a SLFN to accurately model a local gravitational field in a cylindrical region near the surface based on a high resolution



(a) (b) (c)

Figure 6: 25143 Itokawa Low resolution. Regression plots: training set, Linear regression plot of samples used to train sequentially the learning machine, reporting the target constant density low resolution polyhedron gravity attraction of Itokawa compared with the output of the SLFN with $\bar{L} = 5 \times 10^4$ and $\bar{C} = 5 \times 10^5$. Training samples have been presented in chunks of size $N_{chunk} = 5000$.



(a) (b) (c)

Figure 7: 25143 Itokawa Low resolution. Regression plots: test set, Regression plots on test field points, reporting the target gravity attraction of the constant density low resolution polyhedron compared with the one predicted by the SLFN on Itokawa.

polyhedron and show how the SLFN can be employed in a feedback guidance algorithm for a soft landing operation on 25143 Itokawa. Expressed in principal axes frame, the landing site is chosen as a point with coordinates $\boldsymbol{r} = [10.0m, -40.0m, 112.5m]^T$ on the northern side of the surface of the asteroid. We selected a cylindrical region as the region of interest. The radius and the height of the cylinder, depicted in Figure 4.1.3, are $r_{cyl} = 150\,m$ and $h_{cyl} = 800\,m$, as measured from the $x - y$ plane. A dataset with $N = 10^6$ field points randomly sampled, according to a uniform distribution, inside a cylindrical region is obtained using a high-resolution constant density polyhedron model with $N_{plates} = 3145728$, and a density $\rho = 1.9 \times 10^3\ kg/m^3$. The local attraction $\boldsymbol{g}(\boldsymbol{r})$ is computed executing the routine presented in Section 4.1.1, using MATLAB® parallel pool toolbox in order to reduce the time required to perform the huge number of basic operations involved in Equation 2. Using the same computational resources employed for the global gravity field modeling

case of Section 4.1.1, the mean time to obtain a single sample is $T_{sample} \approx 168\,s$.

Comparing the computational efforts to evaluate the gravity fields of the same target in these two scenarios, it is evident that the accuracy of the shape model strongly affects the complexity of the computation, making on-board, real-time, feed-back guidance requiring the knowledge of the exact polyhedron gravity contribute unfeasible. The total amount of required CPU time for the generation of the entire dataset is $T_{CPU,dataset} \approx 46872\,h$. Note that this particular implementation of the polyhedron model has been done using a legacy platform such as MATLAB coupled with a set of parallel processors. We believe that the training set generation computational time may be further reduced by one order of magnitude if a different coding language is employed (e.g. C++).

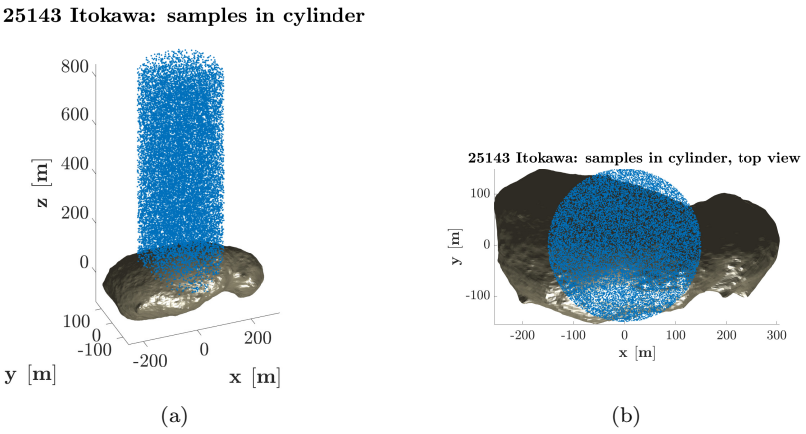


Figure 8: Local field points in a cylinder on 25143 Itokawa. Some of the field points in the region of interest for the modeling of the local gravity field of 25143 Itokawa. Top view of the cylindrical volume on the left.

The learning machine is a SLFN with $L = 50,000$ sigmoid hidden nodes that has been trained using OSELM, which has been designed to learn the relation $\boldsymbol{g} = \boldsymbol{g}(\boldsymbol{r})$ on sequentially presented training data chunks. Similarly to the global gravity modeling case, according to a random partition, 90% of the samples have been chosen to compose the training set, the remaining constituting the set of "never-seen points", or test set. The initial learning phase has been realized with a chunk of randomly selected $N_0 = 50,000$ data among the training set, using the common practice of OSELM to have an initial chunk size equal to the number of hidden units, as reported in Section 3.3. Sequential small chunks presented to the learner had size of $N_{chunk} = 10,000$. Training of the SLFN on the entire training set required $T_{train} \approx 40.34\,h$. Regression performances are reported in Figures 10 and 11, in which are represented the absolute errors in gravity components prediction on the training and test sets.

Accuracy on training set data is achieved with $NRMSE_{train} = 0.0184$ and a $RMSE_{train} = 1.731 \times 10^{-7}\,m/s^2$. Good performances have also been obtained on test set samples, with a $NRMSE_{test} = 0.0183$ and $RMSE_{train} = 1.729 \times$

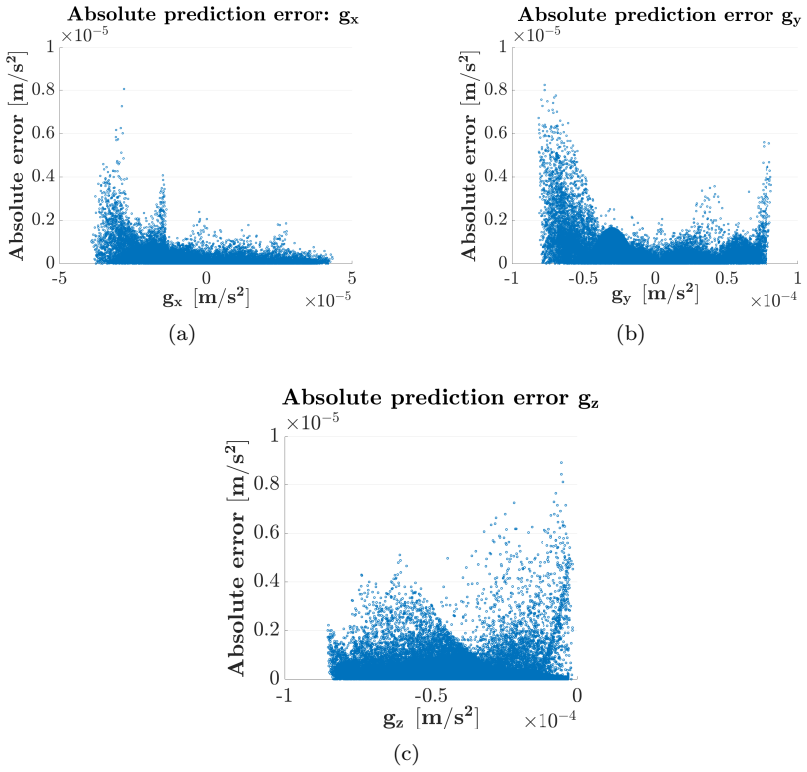


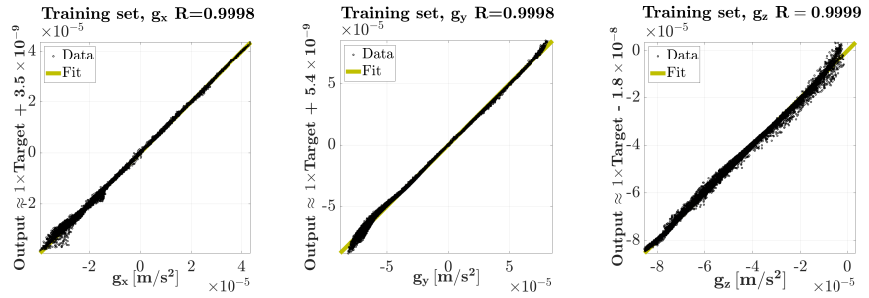
Figure 9: 25143 Itokawa: Absolute prediction errors. Absolute errors on the prediction of the local gravity attraction components, expressed in principal axes frame.

$10^{-7}\, m/s^2$. Results in terms of accuracy are coherent to what observed in the global modeling scenario discussed in Section 4.1.1, showing that the SLFN learned the mapping from $\boldsymbol{r}$ to $\boldsymbol{g}(\boldsymbol{r})$, with satisfying capability to generalize.

4.1.3. Soft Landing on 25143 Itokawa

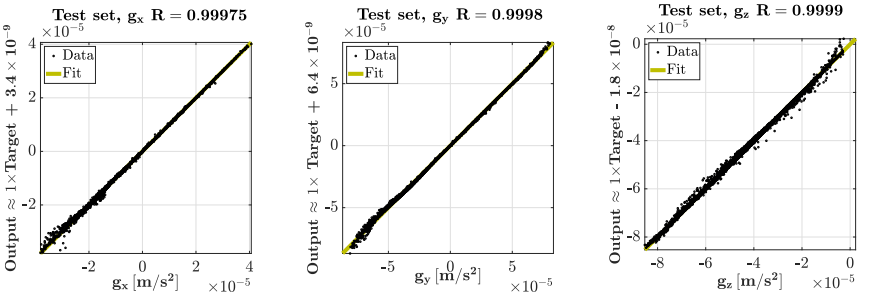
As a practical application of the trained SLFN, we report the results of a simulated soft landing on 25143 Itokawa using an on-line guidance algorithm which exploits the modeling of the local gravitational field performed by the trained SLFN. The motion of the lander is described in a body fixed frame with the axis x, y, z oriented as the minimum, intermediate and maximum axes of inertia, respectively. Because of the short period of time during which the landing takes place, it is assumed that the asteroid spins about a constant axis of rotation, aligned with z.

The equations of motion expressed in the above mentioned asteroid frame can



(a) (b) (c)

Figure 10: 25143 Itokawa High resolution. Regression plots: training set. Linear regression plot of samples used to train sequentially the learning machine, reporting the actual constant density high resolution polyhedron gravity attraction of Itokawa compared with the output of the SLFN.



(a) (b) (c)

Figure 11: 25143 Itokawa High resolution. Regression plots: test set. Linear regression plot on test field points "never seen" during the training, reporting the actual constant density high resolution polyhedron gravity attraction of Itokawa compared with the one predicted by the SLFN.

be written as:

$$\begin{cases} \dot{\boldsymbol{r}} = \boldsymbol{v} \\ \dot{\boldsymbol{v}} = -2\boldsymbol{\omega} \times \boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{\omega} \times \boldsymbol{r} + \boldsymbol{g}(\boldsymbol{r}) + \boldsymbol{a}_c + \boldsymbol{d} \end{cases} \tag{44}$$

where $\boldsymbol{r}$, and $\boldsymbol{v}$ are the lander position and velocity in asteroid centered frame, ω is the asteroid rotation rate. The gravitational acceleration term $\boldsymbol{g}$ is modeled using the constant density polyhedron, while $\boldsymbol{a}_c$ is the control acceleration.
The rotation period of the asteroid, as derived by Kaasalainen et al.(Kaasalainen et al. (2003)) , is assumed to be $T_{rot} = 12.1\,h$.
After neglecting the disturbance term $\boldsymbol{d}$, the Coriolis acceleration $2\boldsymbol{\omega} \times \boldsymbol{v}$, the centripetal term $\boldsymbol{\omega} \times \boldsymbol{\omega} \times \boldsymbol{r}$ and the local gravity field are grouped into a generalized acceleration term $\boldsymbol{g_g}(\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{\omega})$. The equations of motions are then written

as:

$$\begin{cases} \dot{\boldsymbol{r}} = \boldsymbol{v} \\ \dot{\boldsymbol{v}} = \boldsymbol{a}_c + \boldsymbol{g}_g(\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{\omega}) \end{cases} \tag{45}$$

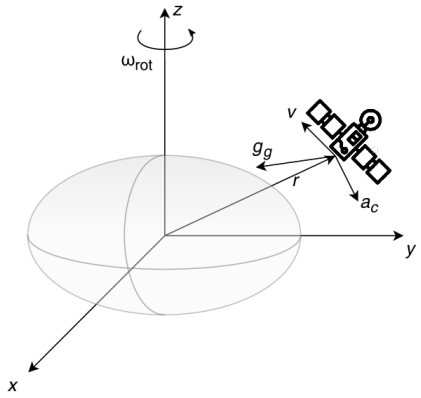


Figure 12: Body frame and relevant quantities of Equation 45 depicted.

A Zero-Effort-Miss/ Zero-Effort-Velocity (ZEM/ZEV) guidance algorithm is adopted to generate in real time the acceleration command that delivers the lander to the desired landing site, in an energy efficient way. As discussed in Hawkins et al. (2012), when the gravity field is not uniform or an explicit function of time, ZEM/ZEV does not represent an optimal guidance solution. When the dynamical environment is highly nonlinear, a modified ZEM/ZEV that directly compensates for the non linear terms at all time instants can be preferable to the usual predictive strategy, as suggested by Battin(Battin (1999)). This sub-optimal guidance algorithm approaches feedback linearization behaviour by using the estimation of the current state and the current gravity attraction to generate the acceleration command $\boldsymbol{a}_c$:

$$\boldsymbol{a}_c = \frac{6[\boldsymbol{r}_f - (\boldsymbol{r} - t_{go}\boldsymbol{v})]}{t_{go}^2} - \frac{2(\boldsymbol{v}_f - \boldsymbol{v})}{t_{go}} - \boldsymbol{g}_{g,p}(\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{\omega}) \tag{46}$$

where t_{go} represents the time-to-go, which is simply the difference between the current instant and the final time; $\boldsymbol{r}_f$ and $\boldsymbol{v}_f$ are the desired final position and velocity respectively. The term $\boldsymbol{g}_{g,p}$ is the predicted generalized acceleration:

$$\boldsymbol{g}_{g,p} = -2\boldsymbol{\omega} \times \boldsymbol{v} + \boldsymbol{\omega} \times \boldsymbol{\omega} \times \boldsymbol{r} + \boldsymbol{g}_p(\boldsymbol{r}) \tag{47}$$

where $\boldsymbol{g}_p$ is the local gravity attraction predicted by the SLFN. It is assumed that $\boldsymbol{r}$, $\boldsymbol{v}$ and body the spin rate $\boldsymbol{\omega}$ are known. In the particular case in which

the target terminal velocity is zero in the body reference frame, this expression becomes just:

$$\boldsymbol{a}_c = -6\frac{\boldsymbol{r}-\boldsymbol{r}_f}{t_{go}^2} - 4\frac{\boldsymbol{v}}{t_{go}} - \boldsymbol{g}_{g,p}(\boldsymbol{r},\boldsymbol{v},\boldsymbol{\omega}) \tag{48}$$

The time of flight for the descent on Itokawa has been fixed to $T = 1800\,s$. The lander, initially located at $\boldsymbol{x}_0 = \{10\,m,\ 20\,m,\ 500\,m\}$ with a velocity $\boldsymbol{v}_0 = \{0.04\,m/s,\ 0.1\,m/s,\ 0\,m/s\}$ had a wet mass of $m_0 = 400\,kg$, and a propulsive system characterized by a specific impulse $I_{sp} = 1500\,s$.

The trajectory has been evaluated using a fixed time step ($T_{step} = 0.5\,s$) fourth-order Runge-Kutta integrator. Figure 13 shows the guided trajectory to the asteroid surface as computed using the polyhedron model as compared with the trajectory computed using the trained SLFN. Figure 14 shows the component of the gravitational acceleration as computed using both methods during the descent phase. Figure 15 reports the histories of the thrust and acceleration components. Gravity acceleration predictions are accurate, especially considering the low absolute values. As expected, the largest deviations occur near the asteroid surface. Importantly, the guided trajectory overlaps and the accuracy of the guided approach is comparable. However, the use of trained SLFN is compatible with on-board implementation. It is estimated that on average, it took on $0.01sec$ to compute each individual instance of the local gravity field during descent using SLFN, versus $168sec$ of CPU time required by the polyhedron model. Indeed, it took about $26hrs$ of CPU time to simulate a single descent trajectory.

4.2. ELM-based SLFN for Comet 67P/Churyumov-Gerasimenko

For the sake of repeatability of the previous approach, a further analysis of gravity modeling and its application in feedback guidance algorithms is reported on the comet 67P/Churyumov-Gerasimenko. For the same reasons cited in the previous section, this case-study is aimed at the modeling an approximation of the gravity field in a localized area. The Hapi region narrow neck between the two lobes of the comet nucleus is designated as one of the most interesting regions to be investigated. This decision does not come from any particular practical mission requirement, but it has been chosen since identified as the region in which the most peculiar gravity field will be experienced, mostly due to the proximity of the two big irregular lobes on the sides. This is also a region in which, because of its morphology, the polyhedron approach is most suitable for gravity modeling. The source of the shape model, built based on images from Rosetta NAVCAM using stereophotoclinometry techniques, is the ESA Rosetta Mission Operation Centre (RMOC, Jorda et al. (2015)).

4.2.1. Local Gravity Model for 67P/Churyumov-Gerasimeko

The body fixed frame used for this study is the Cheops mapping scheme, defined in Jorda (2015). Similarly to the previously reported 25143 Itokawa case, a high resolution polyhedron with $N_{plates} = 4,000,512$ is considered in

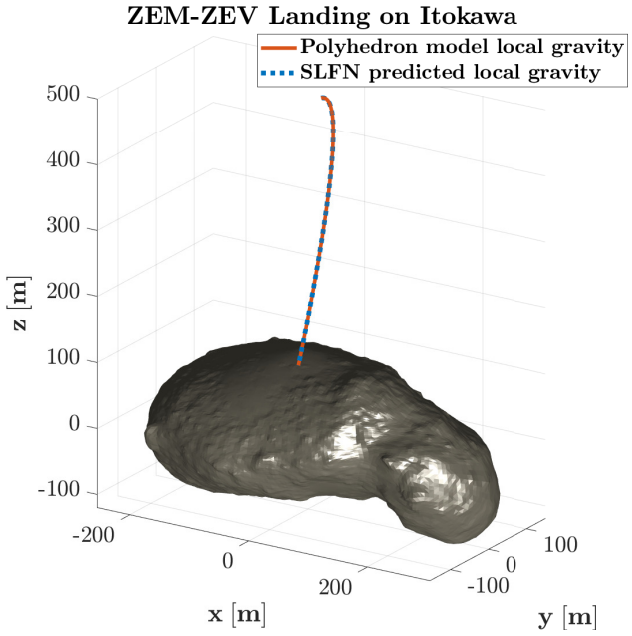


Figure 13: Landing on asteroid 25143 Itokawa: Representation of the descent path. The solid line represents the landing trajectory when the guidance algorithm with compensation of the generalized acceleration is implemented. It can be observed that it is almost coincident with the one computed using the exact constant polyhedron model.

order to generate the $768,000$ samples comprising the training and test sets. These field points are evaluated inside a cylindrical area that encloses the valley above mentioned, as illustrated in Figure 16.

The MATLAB® code deployed on on node of Oceleote cluster as part of the University of Arizona HPC system, takes about $T_{sample} = 200\,s$ to generate a single sample. Employing the same cluster node used for the previous reported study cases, the CPU time needed for the entire dataset generation is $T_{CPU,dataset} \approx 42,560\,h$.

In order to learn the functional map between the position and the correspondent gravitational acceleration, a SLFN with $L = 50,000$ nodes is trained and tested using the computed $768,000$ examples. The choice of the number of sigmoid hidden nodes L and regularization factor C is assigned to an automatic Bayesian optimization routine, explained in Section 4.1.1. Similarly to the results obtained for the global modeling of the gravity field of Itokawa, it is found that the best results in terms of generalization performances are obtained for values of hyperparameters at the boundary of the search interval. Even if lower losses can be reached increasing the number of hidden nodes, it is chosen to limit the search to this case because any improvement would be paid with a drastic increase in the computational effort requested. The SLFN is trained on

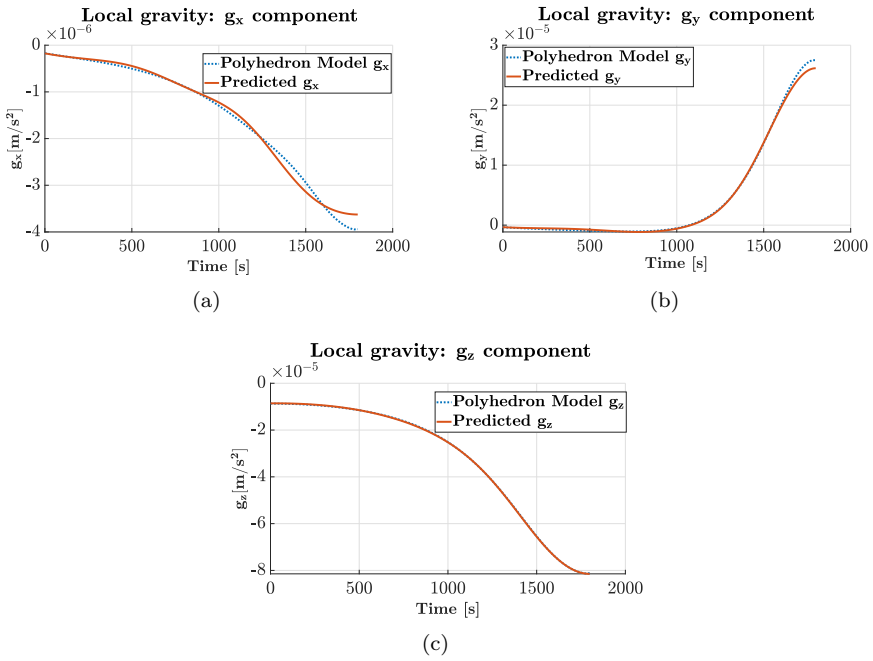


Figure 14: Soft Landing on 25143 Itokawa: gravity acceleration components. Figures (a),(b) and (c) represent the gravity acceleration components experienced during the descent on Itokawa, comparing the ones evaluated using the constant density polyhedron (dotted line) and the prediction from the SLFN (solid line).

90% of the total observations available, using the remaining 10% for testing in order to ensure that the machine is able to generalize. Obtained performances in terms of RMSE and NRMSE are the following:

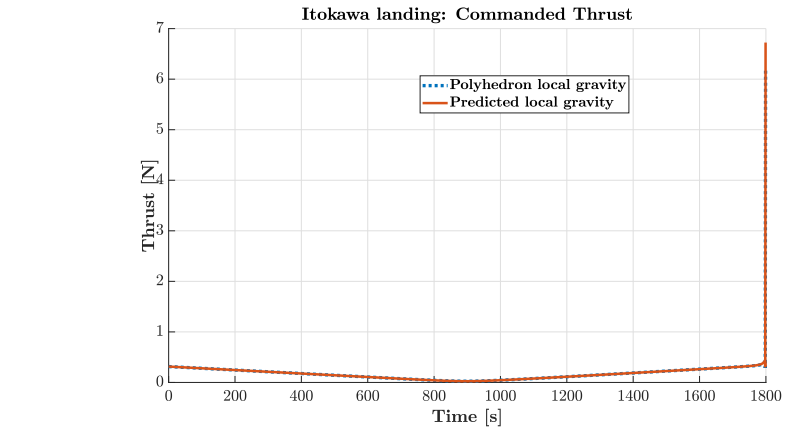
- Accuracy on training data prediction:
  $NRMSE_{train} = 0.057,\ \ RMSE_{train} = 2.2029 \times 10^{-6}\, m/s^2$.
- Accuracy on *"never-seen"* data prediction:
  $NRMSE_{test} = 0.0662, RMSE_{train} = 4.0605 \times 10^{-6}\, m/s^2$.

The total training time has been $T_{train} = 37.25\, h$. Regression analysis reported in Figures 18 and 19 shows that the mapping from the position vector to the gravity acceleration has been efficiently learned.

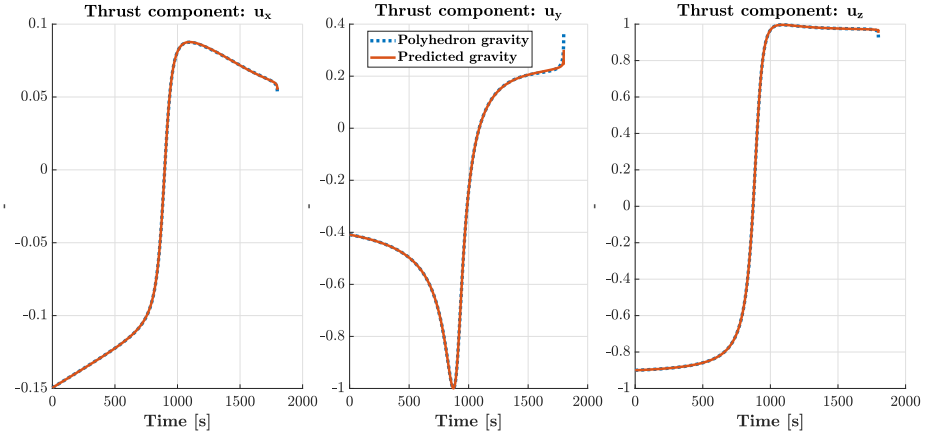
4.2.2. Soft landing on 67P/Churyumov-Gerasimenko

This section reports the results of the case study about a guided soft landing on a 67P/ Churyumov-Gerasimenko where a trained SLFN is employed in the guidance routine. For this kind of operation, the ZEM/ZEV guidance algorithm compensating for the generalized acceleration, briefly presented in Section 4.1.3, is analyzed. The time of flight is fixed to be $T = 2600\, s$, and the simulation

is conducted using a fourth-order Runge-Kutta fixed time step integrator with $T_{step} = 1.3\,s$. The lander mass at the beginning of the landing is considered as $m_0 = 400\,kg$, and the lander is equipped with a propulsive system characterized by a specific impulse $I_{sp} = 1500s$. The landing site is set on a plateau in the Hapi region, with coordinates in the comet nucleus fixed frame set as $\boldsymbol{r}_f = [500\,m,\, -100\,m,\, 390.89\,m]$. The initial position and velocity of the lander are $\boldsymbol{x}_0 = [-250\,m,\, -10\,m,\, 2000\,m]$ and $\boldsymbol{v}_0 = [0.8\,m/s,\, 0.6\,m/s,\, 0.04\,m/s]$ respectively. The equations of motion are Equations 44, and it is still assumed that during the brief landing time interval, nutation and precession of the spin axis are negligible. The comet's spin rate is $\omega = 12.40\,h$ and assumed to be constant (Jorda (2015)). Figure 21 shows the guided trajectory to the comet surface as computed using the polyhedron model as compared with the trajectory computed using the trained SLFN. Figure 22 shows the component of the gravitational acceleration as computed using both methods during the descent phase. Figure 23 reports the histories of the thrust and acceleration components. Gravity acceleration predictions are accurate, especially considering the low absolute values. As expected, the largest deviations occur near the comet surface. Deviations are also experienced in modeling the z-component of the comet gravitational acceleration during the descent. Importantly, guided trajectory overlaps and accuracy of the guided approach is comparable. However, the use of trained SLFN is compatible with on-board implementation. As can be observed by Figure 24, which reports the computational time required to compute the gravitational acceleration from the SLFN given the input position, an approach based on the prediction of the gravity field by a learning machine is computationally cheap enough to be considered viable for an on-board, real-time guidance algorithm.



(a)



(b)

Figure 15: Soft Landing on 25143 Itokawa: commanded thrust comparison. Comet 67P landing: On the top figure is reported the comparison between the magnitude of the commanded thrust. On the bottom one are reported the components, in comet frame, of the thrust vector. The dotted line represents the solution related to the gravity acceleration evaluated using the polyhedron model; the solid one corresponds to the guidance using the SLFN prediction of the gravity components.

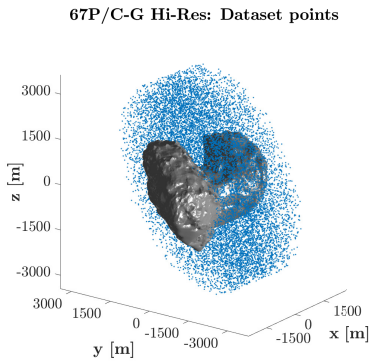
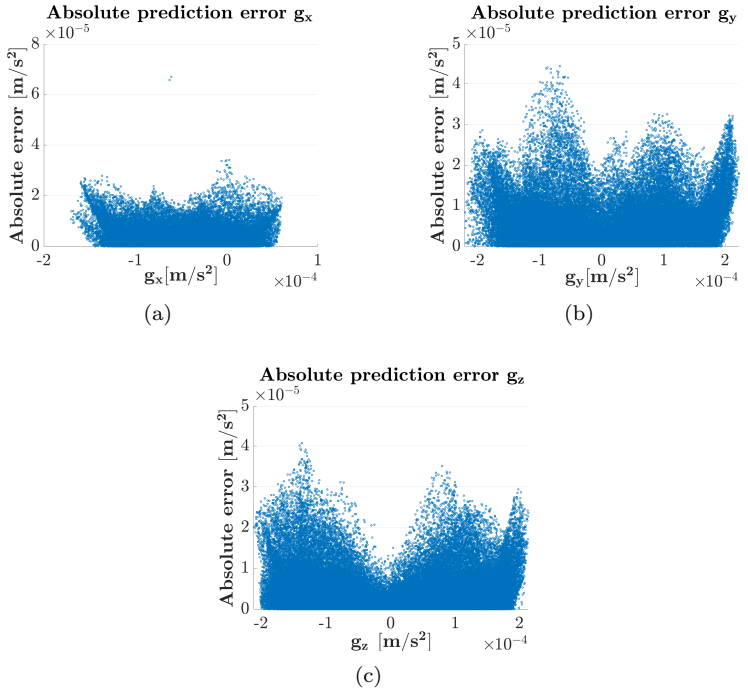
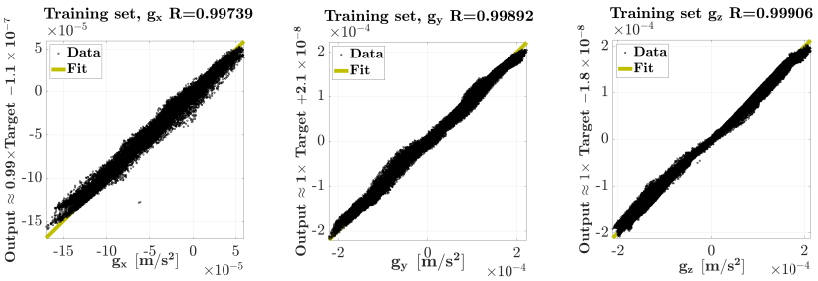


Figure 16: Some of the training set points in the neck region between the lobes of the comet 67P.



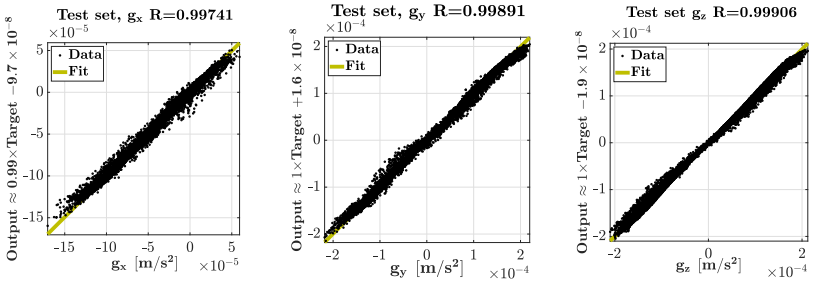
(a) (b) (c)

Figure 17: 67P/C-G: Absolute prediction errors. 67P/ Churyumov-Gerasimenko: Absolute errors on the prediction of the local gravity attraction components, expressed in principal axes frame.



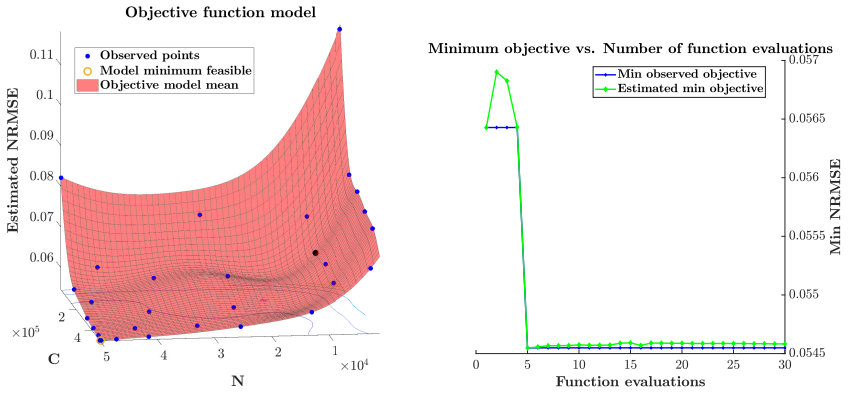
(a) (b) (c)

Figure 18: 67P/C-G High resolution. Regression plots: Training set. 67P/ Churyumov-Gerasimenko: Regression plots for the local 67P gravity acceleration training set.



(a) (b) (c)

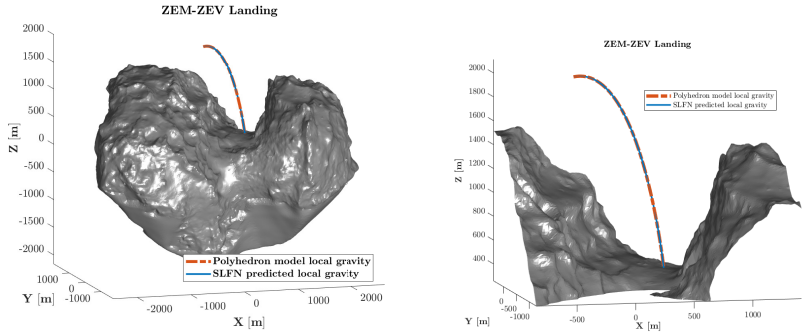
Figure 19: 67P/C-G High resolution. Regression plots: Test set. 67P/ Churyumov-Gerasimenko: Regression plot for the local 67P gravity acceleration testing set.



(a) Objective model

(b) Minimum observed and estimated objective

Figure 20: 67P/C-G: Bayesian optimization results. Figure (a): objective model mean prediction returned by the optimizer after 30 evaluations as a function of the regularized ELM hyperparameters, along with the observations and the recommended $\tilde{\boldsymbol{x}} = \left[\tilde{C}, \tilde{L}\right]$. Figure (b): the observed and estimated minimum objective function as the number of query points increases.



(a) (b)

Figure 21: Soft landing on 67P/ C-G: trajectory. Comet 67P landing trajectory comparison, using ZEM/ZEV guidance algorithm compensating for the local generalized acceleration, comprising the effects due to the comet spinning and the local gravity. The solid line represents the path resulting from the guidance algorithm using the SLFN prediction of the gravity attraction. The dotted line is obtained by compensating for the exact polyhedron gravity acceleration. A more detailed visuaization of the landing site and trajectory is reported in picture on the right.

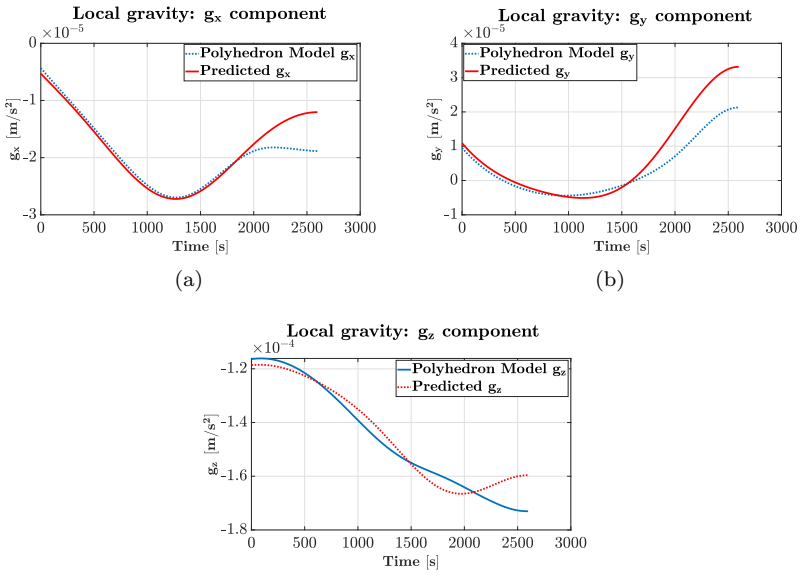
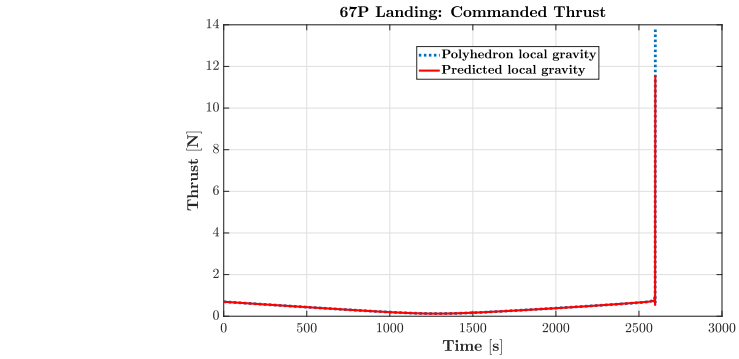
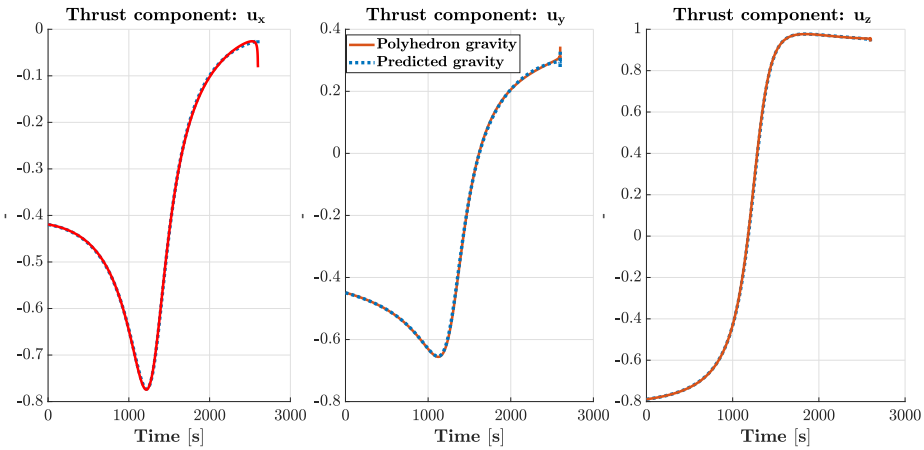


Figure 22: Soft landing on 67P/ C-G: gravity acceleration components. Figures (a),(b) and (c) represent the gravity acceleration components experienced during the descent on 67P, comparing the ones evaluated using the constant density polyhedron (dotted line) and the prediction from the SLFN (solid line). g_x, g_y and g_z respectively,in the x-y plane.



(a)



(b)

Figure 23: Soft landing on 67P/ C-G: commanded thrust comparison. Comet 67P landing: On the top figure is reported the comparison between the magnitude of the commanded thrust. On the bottom one are reported the components, in comet frame, of the thrust vector. As usual, the dotted line represents the solution related to the gravity acceleration evaluated using the polyhedron model, while the solid one is corresponds to the guidance using the SLFN prediction of the gravity component.

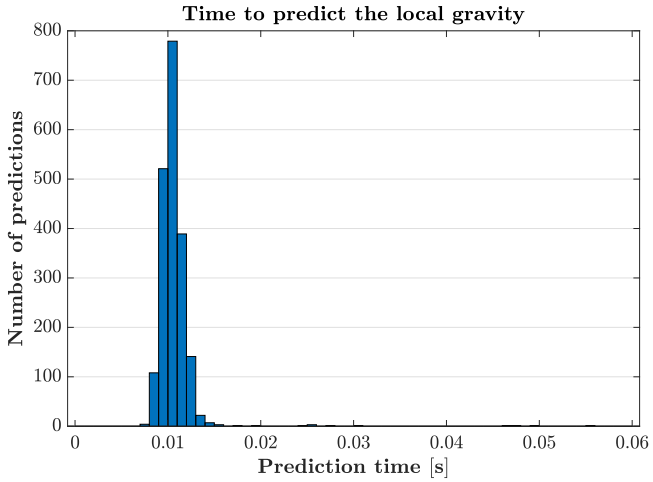


Figure 24: Comet 67P landing: Time required by the SLFN to predict the local gravity, given the input position. The computational effort lies in the normalization of the lander position and the SLFN response evaluation, according to equation 6

5. Conclusions

In this paper, we developed and analyzed a new model-based, data-driven methodology to compute the gravity field of an irregular small body for a fast, accurate and efficient calculation of the gravitational acceleration as function of the relative position around the small body of interest. The proposed approach is based on a recently developed machine learning approach called Extreme Learning Machines (ELM) which employs a Single Layer Feedforward Network (SLFN) to model the non-linear relationship between inputs and outputs. Additionaly, the ELM approach has been embedded in a Bayesian Optimization framework necessary to fine-tune the networks and determine the optimal hyper-parameters. Here, the specific goal is to train, both in batch and sequential fashion, a SLFN to represent the relationship between spacecraft position around the small body of interest and the value of the gravitational acceleration. The approach has been applied to two scenarios comprising the well-known asteroid 25143 Itokawa and the recently explored comet 67/P Churyumov-Gerasimenko. The gravitational field has been computed using computationally expensive polyhedron models. Such models have been employed to generate the training sets necessary to execute the SLFN learning phase. Regression analysis shows that the proposed approach can capture the functional relationship between spacecraft position and gravitational acceleration accurately and efficiently. Overall, we believe we demonstrated that SLFN trained using ELM theories and the Bayesian optimization framework can be successfully employed to accurately approximate the computationally expensive polyhedron gravitational field for small bodies, even when complex and very high resolution models are considered. As shown in the case of gravity modeling about 25143 Itokawa and 67P/ Churyumov-Gerasimenko, the algorithm devised to get the exact polyhedron attraction is computationally expensive and it is neither suitable for on board and real time guidance algorithms nor suitable for applications in the early stage of the mission design to perform Monte Carlo analysis. We have shown that the proposed methodology can obtain fairly accurate results especially considering the low absolute magnitude of the gravitational field encountered around asteroids and comets. Results of the two case studies for guided soft landing on the surface of both asteroid Itokawa and comet 67/P Churyumov-Gerasimenko demonstrated that the trained SLFN are able to quickly and efficiently estimate the gravity field and that, given the input, the time required to calculate the output is in the order of a hundredth of a second. This is an impressive result considering that the time required to solve the exact polyhedron model is thousands of times higher. Consequently, ELM-based SLFN are suitable for on-board implementation thus enabling autonomy for future missions to small bodies.

Reference

Bartlett, P. L., 1997. For valid generalization the size of the weights is more important than the size of the network. In: Advances in neural information

processing systems. pp. 134–140.

Bartlett, P. L., 1998. The sample complexity of pattern classification with neural networks: the size of the weights is more important than the size of the network. IEEE transactions on Information Theory 44 (2), 525–536.

Battin, R. H., 1999. An Introduction to the Mathematics and Methods of Astrodynamics, revised edition. American Institute of Aeronautics and Astronautics.

Berry, K., Sutter, B., May, A., Williams, K., Barbee, B. W., Beckman, M., Williams, B., 2013. Osiris-rex touch-and-go (tag) mission design and analysis. NASA Report.

Brillouin, M., 1933. Équations aux dérivées partielles du 2e ordre. domaines à connexion multiple. fonctions sphériques non antipodes. In: Annales de l'institut Henri Poincaré. Vol. 4. pp. 173–206.

Casella, G., Fienberg, S., Olkin, I., 2007. Matrix Algebra: Theory, Computations, and Applications in Statistics. Springer New York.

Ciesla, F., Charnley, S., 2006. The physics and chemistry of nebular evolution. Meteorites and the early solar system II 943, 209–230.

Duan, L., Bao, M., Miao, J., Xu, Y., Chen, J., 2016. Classification based on multilayer extreme learning machine for motor imagery task from eeg signals. Procedia Computer Science 88, 176–184.

Furfaro, R., Cersosimo, D., Wibben, D. R., 2013. Asteroid precision landing via multiple sliding surfaces guidance techniques. Journal of Guidance, Control, and Dynamics 36 (4), 1075–1092.

Fusi, S., Miller, E. K., Rigotti, M., 2016. Why neurons mix: high dimensionality for higher cognition. Current opinion in neurobiology 37, 66–74.

Gao, A., Liao, W., 2019. Efficient gravity field modeling method for small bodies based on gaussian process regression. Acta Astronautica 157, 73–91.

Gaskell, R., Saito, J., Ishiguro, M., Kubota, T., Hashimoto, T., Hirata, N., Abe, S., Barnouin-Jha, O., Scheeres, D., 2008. Gaskell itokawa shape model v1. 0. NASA Planetary Data System 92.

Golub, G. H., Van Loan, C. F., 2012. Matrix computations. Vol. 3. JHU press.

Guo, Y., Hawkins, M., Wie, B., 2013. Applications of generalized zero-effort-miss/zero-effort-velocity feedback guidance algorithm. Journal of Guidance, Control, and Dynamics 36 (3), 810–820.

Hawkins, M., Guo, Y., Wie, B., 2012. Zem/zev feedback guidance application to fuel-efficient orbital maneuvers around an irregular-shaped asteroid. In: AIAA Guidance, Navigation, and Control Conference. p. 5045.

Huang, G.-B., 2015. What are extreme learning machines? filling the gap between frank rosenblatt's dream and john von neumann's puzzle. Cognitive Computation 7 (3), 263–278.

Huang, G.-B., Chen, L., Siew, C. K., 2006a. Universal approximation using incremental constructive feedforward networks with random hidden nodes. IEEE Trans. Neural Networks 17 (4), 879–892.

Huang, G.-B., Zhou, H., Ding, X., Zhang, R., 2012. Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 42 (2), 513–529.

Huang, G.-B., Zhu, Q.-Y., Siew, C.-K., 2006b. Extreme learning machine: theory and applications. Neurocomputing 70 (1-3), 489–501.

Jorda, L., 2015. Reference frames and mapping schemes of comet 67p. NASA Planetary Data System and ESA Planetary Science Archive.

Jorda, L., Gaskell, R., Hviid, S., et al., 2015. Nasa planetary data system and esa planetary science archive. NASA Report.

Kaasalainen, M., Kwiatkowski, T., Abe, M., Piironen, J., Nakamura, T., Ohba, Y., Dermawan, B., Farnham, T., Colas, F., Lowry, S., et al., 2003. Ccd photometry and model of muses-c target (25143) 1998 sf36. Astronomy & Astrophysics 405 (3), L29–L32.

Kargel, J. S., 1994. Metalliferous asteroids as potential sources of precious metals. Journal of Geophysical Research: Planets 99 (E10), 21129–21141.

Levison, H., Olkin, C., Noll, K., Marchi, S., 2017. Lucy: surveying the diversity of trojans. In: European Planetary Science Congress. Vol. 11.

Liang, N.-Y., Huang, G.-B., Saratchandran, P., Sundararajan, N., 2006. A fast and accurate online sequential learning algorithm for feedforward networks. IEEE Transactions on neural networks 17 (6), 1411–1423.

Oh, D. Y., Goebel, D. M., Elkins-Tanton, L., Polanskey, C., Lord, P., Tilley, S., Snyder, J. S., Carr, G., Collins, S., Lantoine, G., et al., 2016. Psyche: Journey to a metal world. In: 52nd AIAA/SAE/ASEE Joint Propulsion Conference. p. 4541.

Ostro, S. J., Benner, L. A., Nolan, M. C., Magri, C., Giorgini, J. D., Scheeres, D. J., Broschart, S. B., Kaasalainen, M., Vokrouhlicky, D., Chesley, S. R., et al., 2004. Radar observations of asteroid 25143 itokawa (1998 sf36). Meteoritics & Planetary Science 39 (3), 407–424.

Pinson, R. M., Lu, P., 2018. Trajectory design employing convex optimization for landing on irregularly shaped asteroids. Journal of Guidance, Control, and Dynamics 41 (6), 1243–1256.

Russell, R. P., Arora, N., 2012a. Global point mascon models for simple, accurate, and parallel geopotential computation. Journal of Guidance, Control, and Dynamics 35 (5), 1568–1581.

Russell, R. P., Arora, N., 2012b. Global point mascon models for simple, accurate, and parallel geopotential computation. Journal of Guidance, Control, and Dynamics 35 (5), 1568–1581.

Serre, D., 2000. Matrices: Theory and applications. 2002. Graduate texts in mathematics.

Simplício, P., Marcos, A., Joffre, E., Zamaro, M., Silva, N., 2018. Review of guidance techniques for landing on small bodies. Progress in Aerospace Sciences.

Strange, N., Landau, D., McElrath, T., Lantoine, G., Lam, T., 2013. Overview of mission design for nasa asteroid redirect robotic mission concept. et Propulsion Laboratory Report.

Takahashi, Y., Scheeres, D. J., Werner., R. A., 2013a. Surface gravity fields for asteroids and comets. Journal of Guidance, Control, and Dynamics 36 (2), 362–374.

Takahashi, Y., Scheeres, D. J., Werner, R. A., 2013b. Surface gravity fields for asteroids and comets. Journal of guidance, control, and dynamics 36 (2), 362–374.

Vapnik, V., 1992. Principles of risk minimization for learning theory. In: Advances in neural information processing systems. pp. 831–838.

Weightman, J., 1967. Gravity, geodesy and artificial satellites. a unified analytical approach. In: The use of artificial satellites for geodesy. p. 467.

Werner, R. A., Scheeres, D. J., 1996a. Exterior gravitation of a polyhedron derived and compared with harmonic and mascon gravitation representations of asteroid 4769 castalia. Celestial Mechanics and Dynamical Astronomy 65 (3), 313–344.

Werner, R. A., Scheeres, D. J., 1996b. Exterior gravitation of a polyhedron derived and compared with harmonic and mascon gravitation representations of asteroid 4769 castalia. Celestial Mechanics and Dynamical Astronomy 65 (3), 313–344.

Williams, C. K., Rasmussen, C. E., 2006. Gaussian processes for machine learning. Vol. 2. MIT Press Cambridge, MA.

Yang, H., Li, S., Bai, X., 2019. Fast homotopy method for asteroid landing trajectory optimization using approximate initial costates. Journal of Guidance, Control, and Dynamics 42 (3), 585–597.

Yang, H., Li, S., Sun, J., 2020. A fast chebyshev polynomial method for calculating asteroid gravitational fields using space partitioning and cosine sampling. Advances in Space Research 65 (4), 1105–1124.