Limits of Machine Learning Approach on Improving Orbit Prediction Accuracy using Support Vector Machine

Hao Peng
Postdoctoral Associate, Rutgers, The State University of New Jersey, NJ, USA
AstroH.Peng@gmail.com

Xiaoli Bai
Assistant Professor, Rutgers, The State University of New Jersey, NJ, USA
xiaoli.bai@rutgers.edu

ABSTRACT

With the assumption that a machine learning (ML) approach can learn the underlying pattern of the orbit prediction errors from historical data, in this paper, the support vector machine (SVM) is used to improve the accuracy of orbit prediction of resident space object (RSO) in the future. We first show that the trained SVM model can capture the relationship between the chosen learning variables and the target orbit prediction error with both good average and individual performances. Then through a series of experiments, we show that the performance can be further improved with more training data, until adequate data is provided. Moreover, the correction capability of the trained SVM model is limited to the future horizon and its generalization capability will be reduced greatly if the orbit is predicted too far in the future. At the last part of the paper, the effect of model and measurement errors are investigated, including an idealistic case without any error. The results show that the residual errors after the ML-modification will increase as the measurement error in the system increases, but the trained SVM still shows good capability to improve the orbit prediction accuracy. Some insights for future studies are also provided in the paper.

1. INTRODUCTION

The orbit determination and prediction is a classical problem that can at least dating back to Gauss, who introduced a method to determine the classical orbital element of an object from three observations of it. This method is the basis of modern orbit determination methods, and it is still used as an initial orbital determination (IOD) method. The capability of making accurate orbit predictions is the crucial point to applications, such as the collision avoidance. However, the orbit determination is not perfect and has intrinsic errors depends on various aspects, including the measurement error, the estimation error, and even the error in recording or transmitting the data. Fortunately, when we have a large dataset with implicit errors, we would expect that the error information itself can be compensated through using some machine learning methods.

Rather than trying to extract error distribution information for collision avoidance analysis, some researches directly focus on improving the orbit prediction accuracy. Levit and Marshall have proposed a method to improve orbit prediction accuracy based only on the public two-line element (TLE) catalog [1]. The TLE data of an object are used as a very long series of pseudo-observations and are fitted by least square estimator under a high accuracy model. Their method is tested with some precision orbital ephemeris data from the international laser ranging service. Rivera and Bai have demonstrated the capability of generalized linear learning model to improve the orbit prediction based on the analysis of the consistency of the TLE data [2]. The bias of the prediction can be removed, as most above researches are capable of, but the standard deviation or the periodicity of the error cannot be removed. Hao and Bai have also demonstrated using machine learning (ML) approach to improve the orbit prediction accuracy [3] and using data mining techniques to recover area-to-mass ratio information from historical data [4].

Although there are many factors that cannot be included in a theoretical dynamic model, the information of them is implicitly contained in the historical data of the resident space object (RSO), such as the estimation, the measurement

data, the historical prediction error, and many other seemingly irrelevant variables. Nowadays, the machine learning (ML) method has made great progress, which makes it possible to learning from huge amount of data. The ML method has shown great capability in many areas[5, 6]. Actually, there are already some researches in the aerospace area [7, 8, 9, 10]. Hartikainen et al. combined mechanistic principles with non-parametric data-driven techniques to build a model for the orbit prediction [9]. Their method arises from the data mining area, and can extract useful informations purely from historical data. Many data mining techniques share the same background with the machine learning. Sharma and Cutler have presented a learning approach to do orbit determination, involving distribution regression and the multiple-spacecraft scenario [9]. Their test shows that it is possible to estimate vary significantly orbital parameters, which is supposed to be superior to conventional methods. However, the capability of the ML method has not been fully used in the aerospace area, especially the orbit prediction problem that this paper is dealing with.

We have developed a machine learning (ML) approach to improve the orbit prediction accuracy of RSOs, based on both the TLE catalog and a physics-based simulated catalog. The support vector machine method is chosen as the specific ML algorithm in this paper. The results show that the performance can be greatly enhanced based on purely historical data. However, most results of the ML approach cannot be easily interpreted, if it could be understood indeed. In the practice, the SVM is usually treated as a black-box. In this paper, we are not trying to open the black-box and make interpretations of the trained model. Instead, we explore some limits of the ML approach on the orbit prediction problem. It is helpful to carry out a detailed study to analyze the limits of the proposed ML approach, specifically, using the SVM regression model.

In this paper, as a follow-up study of our previous contributions [3], we address the limits of the ML approach in improving orbit prediction accuracy. The assumptions and limitations of the ML approach is first discussed from a theoretical point of view. Then, by a series of numerical experiments in a simulated RSO catalog, we address four critical questions:

- Could the trained ML model actually learn the model of the orbit prediction error?
- What is the effect of the size of the training data on improving the performance of the trained ML model?
- How far in the future could a trained ML model improve the orbit prediction accuracy?
- What is the effect of the model and measurement errors on the performance of the trained ML model?

The remaining part of the paper is organized as following. The simulation environment and the details about the ML approach are presented in Sec. 2. The background assumptions of from both point views of estimation and machine learning are discussed in Sec. 3. Then, numerical results and discussions to above four questions are presented in Sec. 4. At last, conclusions and further insights are given in Sec. 5.

2. SIMULATION BACKGROUNDS AND MACHINE LEARNING APPROACH

In this section, the detailed implementation of the simulated RSO catalog and the ML approach are presented. The choice of learning and target variables are also presented in detail in the last subsection.

2.1. Simulation Environments

The framework of the ML approach based on a simulated RSO catalog is demonstrated in Fig. 1. In the first stage, as shown by the gray box, we simulate the true orbit, the measurement, the estimation, and the orbit prediction of chosen RSOs. Then in the second stage, as shown by the orange box, the proposed ML approach is demonstrated.

The full dynamical model of a RSO in the Earth orbit can be expressed as

$$\begin{aligned}\boldsymbol{a}(t,\boldsymbol{X}) = \boldsymbol{a}_{\text{newtonian}}(t,\boldsymbol{X}) + \boldsymbol{a}_{\text{harmonic}}(t,\boldsymbol{X}) + \boldsymbol{a}_{\text{3B}}(t,\boldsymbol{X}) \\ + \boldsymbol{a}_{\text{drag}}(t,\boldsymbol{X},\boldsymbol{\xi}_1) + \boldsymbol{a}_{\text{SRP}}(t,\boldsymbol{X},\boldsymbol{\xi}_2) + \boldsymbol{a}_{\text{other}}(\boldsymbol{\xi}_3),\end{aligned} \tag{1}$$

where $\boldsymbol{a}(\cdot)$ with different subscripts represent different accelerations of the RSO, t represents the time, $\boldsymbol{X}$ represents the state of the RSO, and $\boldsymbol{\xi}_i$ $(i = 1, 2, 3)$ represent other relevant parameters.

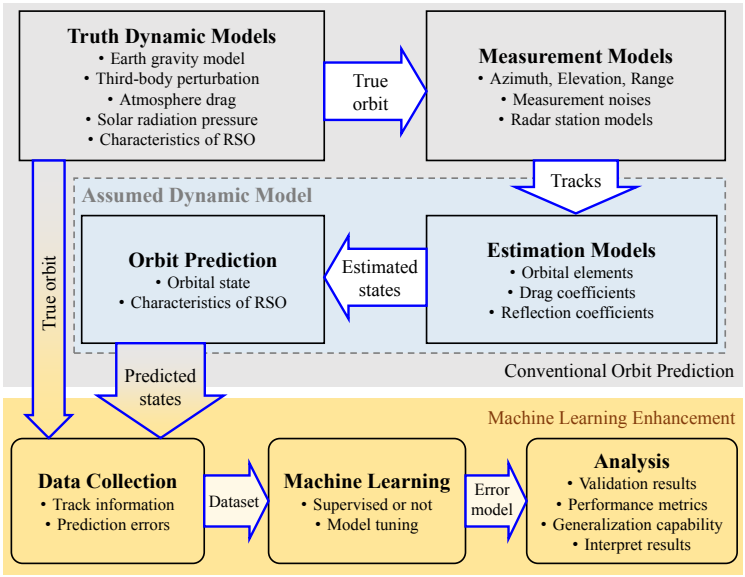


Fig. 1. Framework of the machine learning approach based on a simulated RSO catalog. [3]

The simulated "truth" dynamic models in this paper is expected to include the major factors that could contribute to the orbit prediction error. The setup of the truth model is summarized in Tab. 1. The Newtonian gravitational force is added with an Earth gravitational constant of $3.986 \times 10^{14} \mathrm{m}^3/\mathrm{s}^2$. The non-spherical effect of the Earth gravity is modeled using spherical harmonic functions, with coefficients provided by the EIGEN-6S model [11], truncated with degree/order 40×40 as the truth gravity. Third-body perturbations of all major solar system bodies are considered, including the Sun, all the planets, the Pluto, and the Moon. The position of these bodies are provided by DE430 data file from the JPL [12]. The DTM2000 model is used to approximate the atmosphere, where the Marshall Solar Activity Future Estimate Solar (MSAFE) data from NASA is used to provide solar activity information, which has significant effect on the density and the speed of the atmosphere. The solar radiation pressure is calculated with the reference value as $4.56 \times 10^{-6}\ \mathrm{N/m}^2$ at 1 AU (149,597,870.0 km) from the Sun. And the effect of the penumbra and eclipse are considered. During the generation of true orbit, a spherical RSO with a constant area-to-mass ratio of 0.05 is assumed, and the drag coefficient C_d and single-parameter reflection coefficient C_r are assumed to be constant. These models are implemented using the Orekit, which is a low level space dynamics library written in Java [13].

Tab. 1. Parameters of the "truth" model used to generate orbits and measurements, and the assumed model used in the estimation and prediction.

Parameters	Truth model	Assumed model
Earth Shape	WGS84	WGS84
Harmonics Gravity Field	40×40	10×10
Third-Body Purterbation	Sun + Solar Planets	Sun + Jupiter
	+ Puluto + the Moon	+ the Moon
Atomosphere Model	DTM2000	NRLMSISE-00
Solar Activity	MSAFE	$(F_{10.7}, K_p) = (150.0, 3.0)$

Measurements of a RSO is simulated to be obtained using ground-based radars. The radar station is modeled as a topocentric frame centered at a given geodetic point location defined on the WGS84 Earth ellipsoid. Three stations are used in this study, with their parameters summarized in Tab. 2. The RSO is visible to a station only if the range is less than the maximum range and the elevation is within the feasible elevation range. The station will generate discrete measurements, including the azimuth α, the elevation η, and the range ρ, at each step of the measurement gap, when the target RSO is visible to the ground stations. We note that the measurement gap (60s in this paper) is usually smaller then the orbit propagation step (automatically chosen by integrator to ensure a position tolerance of 0.1 m in

this paper). A series of consecutive measurements are organized as a track, and one track could be combinations of measurements collected from different stations if they could all detect the RSO. In brief, one track starts when it is visible to any station, and end when no stations can detect it.

The measurement errors are simulated as normal distributions with zero biases and standard deviations of σ_α, σ_η, and σ_ρ for the azimuth, elevation and range respectively, as summarized in Tab. 2. We remark that a great advantage of using a simulated RSO catalog is that we can vary these standard deviations to study the performance of the ML approach with respect to the measurement error.

Tab. 2. Ground-based radar stations modeled in the paper [14, 3].

Station	Eglin, FL	Clear, AK	Kaena Point, HI
Latitude [deg]	30.57	64.29	21.57
Longitude [deg]	-86.21	-149.19	-158.27
Altitude [m]	34.7	213.3	300.2
Max Range ρ [m]	13210	4910	6380
Elevation η [deg]	1–90	1–90	0–90
σ_ρ [m]	32.1	62.5	92.5
σ_α [deg]	0.0154	0.0791	0.0224
σ_η [deg]	0.0147	0.024	0.0139

The estimation process is conventional. The batch Least Square (LS) estimator [15] is used to estimate the state of the RSO at the beginning of each track. An assumed dynamic model, different from the "truth" model, is used by the LS estimator. The assumed model is set up with the spherical harmonic gravity model of degree/order 10×10, and with the third-body perturbations including only three major resources, the Sun, the Moon and the Jupiter, and the different NRLMSISE-00 atmosphere model, as summarized in Tab. 1. In this paper, all tracks in past 12 hours are used as the input of the LS estimator, which is chosen through a trail-and-error procedure. The orbit state $\boldsymbol{X}$ and drag coefficient parameter C_d will be finally generated by the LS estimator. We note that although the batch least square estimator can also generate the covariance information for collision risk assessment tasks, but only the orbital state is used in this study, because the covariance information is not available for existing catalogs such as the two-line element (TLE) catalog [16].

After obtaining estimations for all the tracks, the prediction process is straightforward. Using the same assumed dynamic model, the RSO is propagated to a desired future epoch, and then the prediction error can be generated by comparing with the recorded true orbit. Since the assumed dynamic model, the measurement process, and the estimation process will all introduce errors, the resulting prediction error can grow quickly to a meaningless magnitude as the propagation time increases. Therefore, we set a maximum prediction duration of $\Delta t_{\max} = 7$ days to prediction simulations in this paper, which is also long enough for the surveillance and collision avoidance for LEO RSOs.

Finally, we define some notations that will be used throughout the following paper. We use the symbol $\boldsymbol{X}(t)$ to denote the state of the RSO at time t, without expressing it in a coordinate frame. The state $\boldsymbol{X}(t)$ can be expressed in the classical orbital element (COE) form as ${}^{\texttt{COE}}\boldsymbol{X}(t) = [a, e, i, \omega, \Omega, \nu]^\mathsf{T}$, or in the Earth-centered inertial (ECI) frame as ${}^{\texttt{ECI}}\boldsymbol{X}(t) = [X, Y, Z, V_X, V_Y, V_Z]^\mathsf{T}$. The difference $\delta\boldsymbol{X}(t)$ between two states $\boldsymbol{X}_1(t)$ and $\boldsymbol{X}_2(t)$ at the same epoch t will be expressed in the RSW frame as ${}^{\texttt{RSW}}\delta\boldsymbol{X}(t) = [\delta x, \delta y, \delta z, \delta v_x, \delta v_y, \delta v_z]^\mathsf{T}$ in this paper, where x-axis (radial) is the radial direction, the y-axis (along-track) is perpendicular to the x-axis in the orbital plane and points to the inertial velocity direction, and the z axis (cross-track) is along the angular momentum direction [17]. The above symbols without any modifier indicate they are true value, the value of the true orbit, such as the true state $\boldsymbol{X}(t_i)$. While, the hat over a symbol is used to indicate that it is an estimated value, such as the estimated state $\hat{\boldsymbol{X}}(t)$ or drag coefficient $\hat{C}_d$. We use an additional time variable after a semicolon to indicate this value is based on a previous estimate, such as $(t_j; t_i)$ in $\hat{\boldsymbol{X}}(t_j; t_i)$ indicates this state is predicted at t_j based on $\hat{\boldsymbol{X}}(t_i)$.

In order to investigate the limits of the previously proposed ML approach, the RSO ENVISAT is chosen and fixed as the study object. ENVISAT has a hemispherical retroreflector array with a radius of 20 cm, and it is one of the objects tracked by the International Laser Ranging Service (ILRS) using the laser ranging technique. General information and parameters about the ENVISAT is summarized in Tab. 3. ENVISAT is also tracked by the TLE catalog. The initial

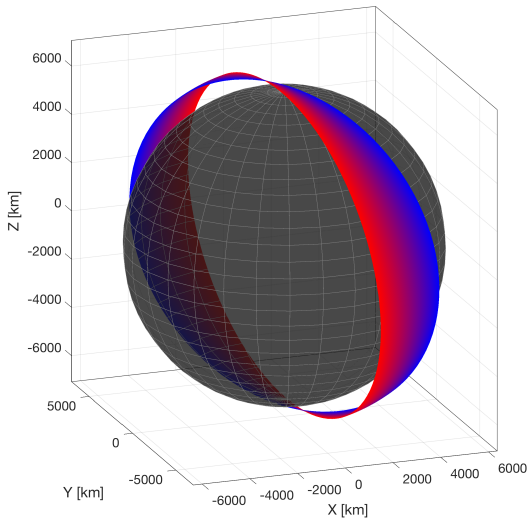
state of the orbit of ENVISAT is extracted from the following TLE set using the standard SGP4 model:

```
0 ENVISAT
1 27386U 02009A   17238.87724141 +.00000000 +00000-0 +13804-4 0  9992
2 27386 098.2254 281.1813 0001340 081.3267 278.8065 14.37901343811233
```

And Tab. 4 demonstrates the orbit of the simulated RSO in the simulated environment for 4 weeks, using the above initial state. The orbit is colored by red at the beginning, and is seen to gradually change to blue at the end. This is expected as that the orbit processes within the simulated time interval, due to the perturbation forces. We note that the framework of the ML approach can be expanded to other RSOs as well.

Tab. 3. Parameters of ENVISAT from ILRS.

Parameters	Values
Name	ENVISAT
NORAD ID	27386
Orbit	Sun-Synchronous Orbit (SSO)
Launch Date	1 March 2002
Altitude	$\sim$ 796 km
Period	$\sim$ 100 minutes
Weight	$\sim$ 8211 kg
Inclination	$\sim$ 98.54 deg
Eccentricity	$\sim$ 0.001165



Tab. 4. figure
Orbit of ENVISAT in 4 weeks based on one specific TLE set.

2.2. Support Vector Machine

The support vector machine (SVM) method is a machine learning algorithm that can be used for both classification and regression problems. One strength of the SVM method is that they are nonparametric techniques, so we do not need to specify the basis functions in prior. The SVM regression can handle nonlinear problems since it relies on kernel functions. Moreover, the SVM method has universal approximation capability with various kernels including the Gaussian kernel [18], which makes it even more suitable to approximate the unknown model of the orbit prediction error. In this paper, the svm regression method in MATLAB is used. The concept of the SVM is briefly reviewed below, and the details are referred to the references.

Suppose the input data is $\boldsymbol{x} \in \mathbb{R}^m$, and the corresponding target is $y \in \mathbb{R}$. Given a set of data point $(\boldsymbol{x}, y)$, the ε-SVM regression method aims to find a function $f(x)$ that has at most ε deviation from the actual obtained targets for all the training data [19, 20].

For the linear case, assume the desired function takes the form

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\mathsf{T}}\boldsymbol{x} + b, \tag{2}$$

where $\boldsymbol{\omega} \in \mathbb{R}^m$ is the variable, $b \in \mathbb{R}$ is the bias. Then the training problem is to find the flattest function in the space $\mathcal{L}$, where the flatness of the function is represented by $\|\boldsymbol{\omega}\|^2$. The training problem is casted as a convex optimization problem to minimize the cost function

$$\min J(\boldsymbol{\omega}) = \frac{1}{2}\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{\omega} + C\sum_{n=1}(\xi_n + \xi_n^*), \tag{3}$$

subject to the constraints, $\forall n$,

$$\begin{aligned} y_n - (\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{x}_n - b) &\leq \varepsilon + \xi_n, \\ (\boldsymbol{\omega}^{\mathsf{T}}\boldsymbol{x}_n - b) - y_n &\leq \varepsilon + \xi_n^*, \\ \xi_n, \xi_n^* &\geq 0, \end{aligned} \tag{4}$$

where C is usually referred to as the box constraint, and the slack variables ξ_n and ξ_n^* are introduced to make the margin soft. This is the so called primary problem. By introducing dual variables α_n and α_n^* for each data set $\boldsymbol{x}_n$, its dual problem is obtained as

$$\min L(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_n - \alpha_n^*)(\alpha_j - \alpha_j^*)\boldsymbol{x}_i^{\mathsf{T}}\boldsymbol{x}_i + \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N}(\alpha_i^* - \alpha_i), \tag{5}$$

subject to the constraints

$$\begin{aligned} \sum_{n=1}^{N}(\alpha_n - \alpha_n^*) &= 0, \\ \forall n\colon 0 \leq \alpha_n \leq C, \quad 0 &\leq \alpha_n^* \leq C. \end{aligned} \tag{6}$$

And the KKT conditions are, $\forall n$,

$$\begin{aligned} \alpha_n(\varepsilon + \xi_n - y_n + f(\boldsymbol{x})) &= 0, \\ \alpha_n^*(\varepsilon + \xi_n^* + y_n - f(\boldsymbol{x})) &= 0, \\ \xi_n(C - \alpha_n) &= 0, \\ \xi_n^*(C - \alpha_n^*) &= 0. \end{aligned} \tag{7}$$

Then, the function $f(\boldsymbol{x})$ can be expressed as a linear combination of training data

$$f(\boldsymbol{x}) = \sum_{n=1}^{N}(\alpha_n - \alpha_n^*)\boldsymbol{x}_n^{\mathsf{T}}\boldsymbol{x} + b, \tag{8}$$

where $\boldsymbol{\omega}$ in Eq. (2) is replaced by $\sum_{n=1}^{N}(\alpha_n - \alpha_n^*)\boldsymbol{x}_n$. This property of the SVM makes it possible to deal with nonlinear regressions via kernels.

Substituting the inner product $\boldsymbol{x}_i^{\mathsf{T}}\boldsymbol{x}_j$ in above equations by the kernel $k(\cdot,\cdot)$, the optimization problem is reformulated to minimize the new cost function in the feature space indicated by the kernel,

$$\min L(\boldsymbol{\alpha}) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)k(\boldsymbol{x}_i, \boldsymbol{x}_j) + \varepsilon\sum_{i=1}^{N}(\alpha_i + \alpha_i^*) - \sum_{i=1} y_i(\alpha_i - \alpha_i^*), \tag{9}$$

The constraints and the KKT conditions are the same. These conditions indicate that all observations strictly inside the epsilon tube have Lagrange multipliers $\alpha_n = 0$ and $\alpha_n^* = 0$. Observations with nonzero Lagrange multipliers are called support vectors.

Finally, the function used to predict new values are

$$f(\boldsymbol{x}) = \sum_{n=1}^{N}(\alpha_n - \alpha_n^*)k(\boldsymbol{x}_n, \boldsymbol{x}) + b, \tag{10}$$

where α_i and α_i^* are the dual variables solved from the dual problem [19, 20]. As shown in Eq. (10), when using the kernel technique, the coefficient $\boldsymbol{\omega}$ in Eq. (2) will not be provide explicitly. However, when given a new testing data $\boldsymbol{x}_{\text{test}}$, according to Eq. (10), we only need the kernel $k(\cdot,\cdot)$ and the corresponding dual variables $\boldsymbol{\alpha}$ and $\boldsymbol{\alpha}^*$ to generate the prediction $f(\boldsymbol{x}_{\text{test}})$ of the trained SVM model.

In this paper, the parameters of the SVM model are chosen based on a series of trail-and-error processes. The finally results are

- The ε-margin is fixed as 1 km with, which means ideally we only tolerate a deviation of at most 1km away from the true value of the target y_n.
- The box constraint C is chosen as the default value in MATLAB, which is the interquartile of all the target variable y_n. This parameter reflects the cost to tolerate a data point out of the ε-margin. So the larger C is, the more data points accurate will be fitted within the margin. However, too large C will dramatically increase the computation time, and it is possible to lead to overfitting.
- The gap tolerance between the primary and the dual problem is used as a stop criteria, which is chosen as 10^{-2} based on a trail-and-error process.
- The KKT tolerance is used as another stop criteria, which is chosen as 10^{1} based on a trail-and-error process to compromise between the computation time and regression accuracy.

We remark that the design and optimization of the SVM model has drawn great attention in the field of machine learning, but in this paper, we will focus on its performance on the orbit prediction problem, with a fixed set of parameters, and without going into the detailed implementations of the algorithm.

2.3. Learning and Target Variables For Orbit Prediction Problem

The concept of the proposed ML approach to directly modify the orbit prediction is illustrated in Fig. 2. After the ground station generate an estimate of the target RSO, its conventional orbit prediction is processed, which generates the predicted state at a future epoch. Then the trained ML model, the SVM model in this paper, will be applied to this prediction, and generates an ML-modified state, which is expected to be closer to the true state. Importantly, as shown in Fig. 2, the ML approach does not modify the estimate or improve dynamic models, whereas, it directly works on the orbit prediction. This particular feature makes the ML approach different from conventional dynamical approaches. Because of this, the ML approach is not restricted to specific assumed model used by the RSO catalog, but can be extended to different catalogs.

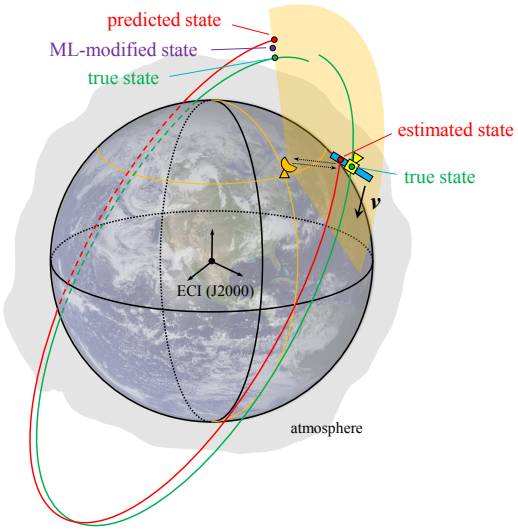


Fig. 2. Illustration of machine learning approach to directly improve the orbit prediction.

As we have discussed in previous sections, enough learning variables is a necessary condition to capture the underlying pattern between the learning and target variables. It is usually easy to determine and choose the target variables, but it is not straightforward at all to determine which variables are related to the target variables.

First, we consider the prediction duration Δt, the estimated state $\hat{\boldsymbol{X}}(t_i)$ at the current epoch t_i (expressed in both COE and ECI forms), the measurements $\boldsymbol{\theta}(t_i)$ at the maximal elevation on the i-th track, the estimated drag coefficient $\hat{C}_d$. It is intuitive to include these parameters, for that their errors will be directly propagated to the prediction results. Second, another sets of significant variables are considered, the predicted state $\hat{\boldsymbol{X}}(t_j; t_i)$ at the future epoch t_j, based

on current epoch t_i. In the practice, this information is available to the RSO catalog, with a simply propagation from t_i to t_j. Third, the information of the track should be considered. Because intuitively, a larger elevation angle and a smaller range will lead to more accurate measurement. One potential drawback is that this information is not available in all catalogs.

The choose of learning variables is based on a trail-and-error process, so the result is not meant to be optimal in any sense. At a specific epoch t_i, the learning variables include:

- Prediction duration $\Delta t = t_j - t_i$ to the future epoch t_j ($t_i < t_j$).
- Estimated state $\hat{\boldsymbol{X}}(t_i)$ at the current epoch t_i, expressed as both ${}^{\texttt{COE}}\boldsymbol{X}(t_i)$ in the COE form and ${}^{\texttt{ECI}}\boldsymbol{X}(t_i)$ in the ECI form. These two forms are related through a nonlinear transformation. Although theoretically they provide the same information, they different for the numerical SVM algorithm.
- Estimated drag coefficient $\hat{C}_d(t_i)$ at the current epoch t_i.
- Maximal measured elevation in the current track η, and the corresponding range ρ and azimuth α at that epoch, denoted by $\boldsymbol{\theta}(t_i) = [\rho, \alpha, \eta]^{\mathsf{T}}$. We note that $\boldsymbol{\theta}(t_i)$ usually occurs at the middle of the track starting at t_i, rather than at the exact epoch t_i.
- Predicted state $\hat{\boldsymbol{X}}(t_j; t_i)$ at the current epoch t_j, based on $\hat{\boldsymbol{X}}(t_i)$, which are also expressed as both ${}^{\texttt{COE}}\hat{\boldsymbol{X}}(t_j; t_i)$ and ${}^{\texttt{ECI}}\hat{\boldsymbol{X}}(t_j; t_i)$.

For simplicity, we denote all learning variables by $\boldsymbol{\Lambda}(t_i)$. The target variables are:

- True predicted error $\boldsymbol{e}(t_j; t_i)$ at the future epoch t_j, based on current estimate $\hat{\boldsymbol{X}}(t_i)$, expressed in the RSW frame as ${}^{\texttt{RSW}}\boldsymbol{e}(t_j; t_i) = [e_x, e_y, e_z, e_{vx}, e_{vy}, e_{vz}]^{\mathsf{T}}$.

With the chosen learning and target variables, the illustration of the training and testing process of the ML approach is demonstrated in Fig. 3. During the collection of training data, each estimated state is propagated to the epoch of all following estimates with $\Delta t < \Delta t_{\max}$, then the learning variable $\boldsymbol{\Gamma}(t_i)$ and the true prediction error $\boldsymbol{e}(t_j; t_i)$ are collected as a data point $(\boldsymbol{\Lambda}(t_i), \boldsymbol{e}(t_j; t_i))$, which corresponds to the set $(\boldsymbol{x}, y)$ in Sec. 2.2. The whole dataset will be used to train a SVM model to approximate $\boldsymbol{e}(t_j; t_i)$ based on $\boldsymbol{\Lambda}(t_i)$.

When the SVM model has been trained with desired accuracy, it can generate the ML-predicted error $\hat{\boldsymbol{e}}_{\mathrm{ML}}(t_j; t_i)$, as shown in Fig. 3. In an ideal situation, $\hat{\boldsymbol{e}}_{\mathrm{ML}}(t_j; t_i)$ should equal to $\boldsymbol{e}(t_j; t_i)$, and thus the residual error $\boldsymbol{e}_{\mathrm{res}}(t_j; t_i) = \boldsymbol{e}(t_j; t_i) - \hat{\boldsymbol{e}}_{\mathrm{ML}}(t_j; t_i)$ should be zero. In the practice, $\boldsymbol{e}_{\mathrm{res}}(t_j; t_i)$ could never be zero due to various factors such as the randomness in the system, so the statistical properties of $\boldsymbol{e}_{\mathrm{res}}(t_j; t_i)$ will be analyzed to evaluate the performance of the trained SVM model.

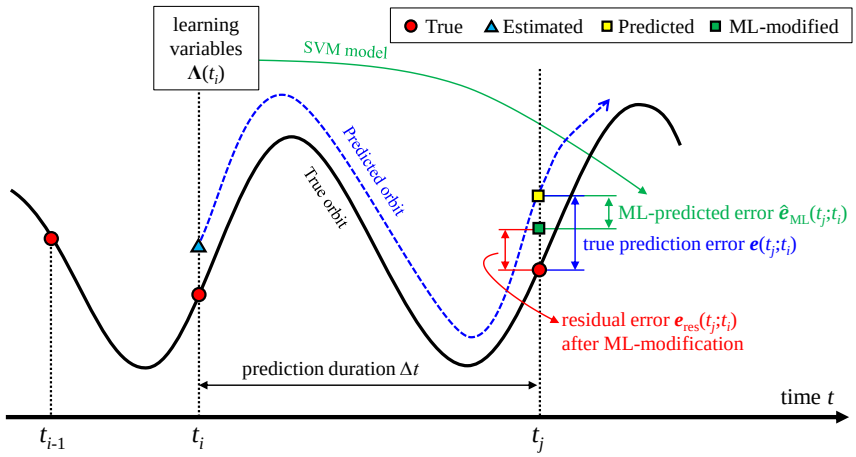


Fig. 3. Illustration of learning and target variables.

In the following discussion, we will use these learning variables to train the SVM and investigate its limits. We remark that current results are based on the simulated RSO catalog and the chosen RSO of ENVISAT, so the choice

of learning variables can be different in other environments. However, similar procedures can be carried out to apply the ML approach on a different catalog.

3. BACKGROUND ASSUMPTIONS AND INTRINSIC LIMITATIONS

In this section, we provide some insights on the background assumptions of the orbit estimation problem and the machine learning problem.

3.1. Assumptions in Orbit Estimation and Prediction

The assumed model used for estimation and prediction is one of the fundamental assumptions. Although in this paper we can directly control the differences between the assumed and the true models, in reality, the only possible way to evaluate the accuracy of the assumed model is to compare the resulted estimate. If the assumed model is not good, we should not expect good accuracies over long time duration, and also the ML approach should not be expected to miraculously make up the model error.

The least square (LS) estimator assumes that the errors are unbiased Gaussian noises and obey the independent identical distribution (IID) rule. Although these assumptions are taken in most areas, they cannot be easily verified. For example, the ranging information depends on the echo from the RSO, so if the RSO shows a smaller cross section at a specific attitude, the corresponding echo will be weaker, and thus the actually generated ranging information will have a larger variance, which means the IID assumption is broken in this example.

Another implicit assumption is that the information of orbital state is not overwhelmed by the random noise in the system. This assumption can be verified if the orbital prediction accuracy could be practically improved by the ML approach. However, a potential problem is that the randomness can be learned by the ML approach, rather than the real information or real patterns, as we will discuss in the next section.

3.2. Assumptions in Machine Learning

The machine learning is basically a data driven process, so the data must contain information, and then the ML algorithm could possibly learn some knowledge. After learning some knowledge, the next step is to generalize these knowledge to a new instance to test whether the knowledge is universal. Mathematically, the learning process is a nonlinear curve-fitting process, while generalization is the interpolation and extrapolation of the input data [21].

Another assumptions for ML algorithms are that the data points are independent and identically distributed (IID), although the probability distribution is unknown in priori. And when the trained model is tested on the testing data, one more assumption that the two dataset are correlated is actually introduced, which is intuitively because the orbit prediction is carried in the same system for the same RSO.

Apart from above assumptions carried by almost any ML algorithm, we would like to emphasis two truth about the ML study:

Bias-Variance Dilemma Most supervised learning algorithms suffer from the the fact that the small bias and the small variance conflict [21, pg.29]. Usually, the best generalization performance is achieved by balancing the bias and variance. Intuitively, bias is caused by the choice of wrong or insufficient basis functions, while the variance is caused by the finite number of training samples [21, pg.30]. We remark that, here, the concept of bias and variance is defined on the generalization result of a trained ML model, but in Sec. 4 the bias and standard deviations is defined on the residual error after the modification of the ML approach, which are different concepts, so there is not contradiction.

No-Free-Lunch Theorem The no-free-lunch theorem says that no search algorithm is better than another in locating an extremum of a cost function when averaged over the set of all possible discrete functions [21, pg.44]. The direct consequence is that no one ML method is better than the others for all problems. For the orbital prediction problem, it implies that it is possible that a successful ML approach that specially designed for RSOs in LEO could fail when generalized to other very different RSOs.

At last, we summarize some facts about the proposed ML approach,

- The data contain randomness and errors;

- The relationship between learning variables and targets are unknown to the ML algorithm;
- Randomness in the system cannot be completely eliminated;

and also some machine learning assumptions used throughout the paper:

- The prediction error follows a statistically modeled that can be modeled;
- The data size is large enough to capture the underlying relationship between the learning and target variables;
- The relationship contained in the data will not be overwhelmed by the randomness, outliers, and errors;

Most time, these assumptions cannot be easily verified before carrying out a practical ML model. However, there are many methods to examine and validate the result or performance of the trained ML model. We note that this paper is not devoted to the research of the ML theory or algorithm, so, more precise discussion are referred to references [21, 20, 5, 22] and the references therein.

3.3. Other Assumptions and Derived Requirements

Some assumptions are difficult to summarize. For example, the initialization of the weights of the training data for the SVM model is made under the assumption that all data points have equal importance. However, this could also be treated as a specific design of the ML approach. For another example, the stop criteria of gap tolerance and KKT tolerance are also based on the implicit assumption that the global minimum is not far from the current position. So, a compromise must be made between strictly obeying various assumptions and boldly exploring all possibilities.

Starting from all above assumptions, we can derive some requirements for the dataset used for the ML approach.

- The learning variables must be practically available;
- The learning variables must be able to capture the information of orbit prediction error;
- The size of the data must be large enough.

4. NUMERICAL PERFORMANCE OF THE ML APPROACH

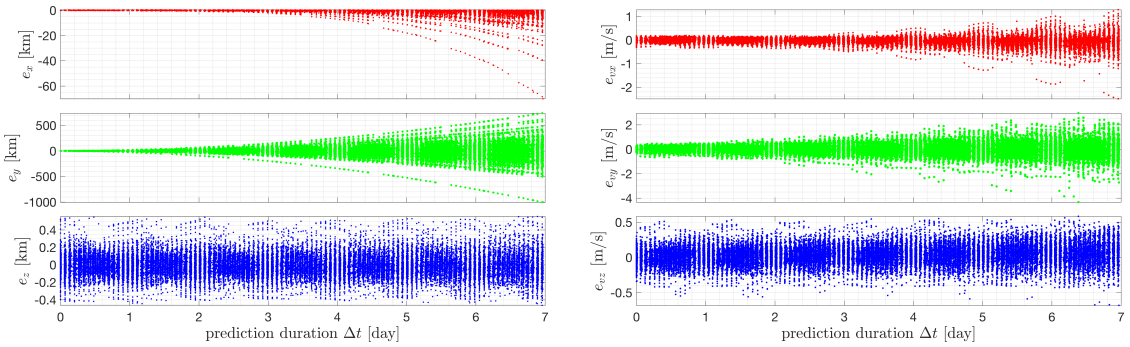
In this section, numerical results are demonstrated in four subsections, and four critical limits of the proposed ML approach are explored along with the demonstrations, including 1) the learning capability of the SVM model; 2) the effect of training data size; 3) the maximum prediction capability; and 4) the effect of the noise and randomness in the system.

The numerical integrator is chosen to be the Dormand-Prince 8(5,3) method for all the propagations in this paper. The absolute tolerance on each position component in the ECI frame is set as 0.1 m, and the maximum propagation step size is restricted to 90 seconds. The standard deviations of the observations generated by the ground stations are provided in Tab. 2. The ε-SVM regression function in MATLAB is used to establish the SVM model, where the parameters are summarized in Sec. 2.2.

4.1. Training Results

In this subsection, we elaborate the definitions and conventions used in the following paper. The introduction is kept as simple and self-complete as possible, and detailed definitions and discussions are available in our previous study [3].

The RSO's orbit is simulated for 4 weeks, and relative data are collected for further analysis. The data in the first 3 weeks is used to train the SVM model, and the remaining data in week 4 is used for testing. We note that the orbit prediction in the testing data must be based on an earlier estimate in the training data. For example, an orbit prediction at day 25 (in week 4) based on the estimate at day 23 (in week) cannot be used as the testing data, because, physically, the estimate at day 23 has not happened and thus learning variables are not available. All the simulation of the generalization problem to future epochs should satisfy this constraint, so we will not mention it again in the following paper.



(a) velocity error components

(b) velocity error components

Fig. 4. Different components of true orbit prediction errors of ENVISAT.

Fig. 4 demonstrates different components of all the true prediction errors $\boldsymbol{e}$. The horizontal axis represents the prediction interval Δt, and the vertical axis represents position components of $\boldsymbol{e}$ in Fig. 4(a) and velocity components in Fig. 4(b). It is clearly that the along-track error e_y is much larger than that of e_x and e_y. The most important task is to reduce the orbit prediction error of the along-track position error e_y. Therefore, in the following study, we will only use e_y as the study object and demonstrate the performance of reducing e_y through the ML approach.

The performance metrics P_{ML} used to quantify the trained SVM model is defined as the ratio between the summation of absolute deviations between e and e_{ML} at each data point, and the summation of absolute true errors e of all data point within the testing data. The definition of P_{ML} can be expressed as [3],

$$P_{\mathrm{ML}} = 100\% \cdot \frac{\sum_{i=1}^{n} |e - \hat{e}_{\mathrm{ML}}|}{\sum_{i=1}^{n} |e|} = 100\% \cdot \frac{\sum_{i=1}^{n} |e_{\mathrm{res}}|}{\sum_{i=1}^{n} |e|}, \tag{11}$$

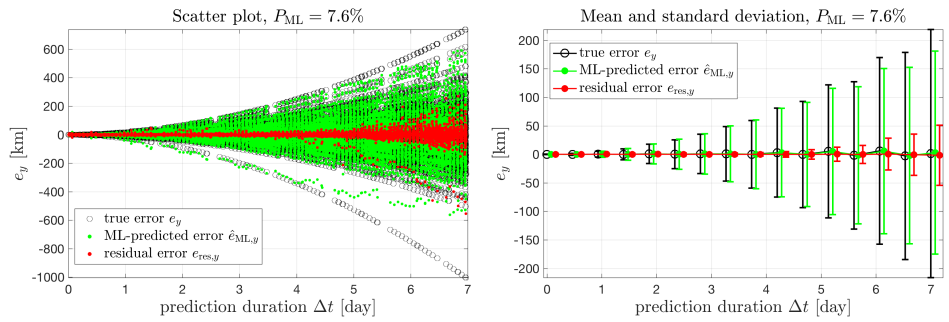
where n is the size of the testing data, and $\boldsymbol{e}_{\mathrm{res}} = \boldsymbol{e} - \hat{\boldsymbol{e}}_{\mathrm{ML}}$ is the residual error after ML-modification, obtained by subtracting the orbit prediction error $\boldsymbol{e}_{\mathrm{ML}}$ from the true error $\boldsymbol{e}$. The metrics reaches its lower boundary zero when the ML-predicted error is identical to the true error, but has no upper boundary. The metrics P_{ML} actually measures the percentage of the residual error $\boldsymbol{e}_{\mathrm{res}}$ with respect to the true error of the testing data, so a smaller P_{ML} means more errors are compensated by the trained SVM model, and thus the performance is better.

Fig. 5 demonstrates the performance of the trained SVM model on the training data. In the left scattering plots, the black circles represent true prediction errors e_y, the green dots represent ML-generated prediction errors $e_{\mathrm{ML},y}$, and the red dots represent the residual prediction errors $e_{\mathrm{res},y}$ after subtracting $e_{\mathrm{ML},y}$ from e_y. In the right error bar plots, the data points are clustered into several groups, then for each group, the center dot represents the mean value of different errors distinguished by the color same to the left plots, and the length of the bar from the top (or bottom) to the center dot represents the standard deviation. The error bars have are slightly displaced for clarity.

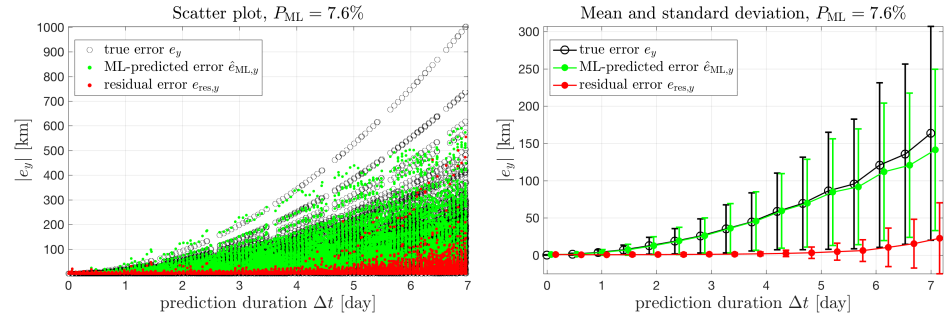
The real value of the results are shown in Fig. 5(a), and the absolute value in Fig. 5(b). The metrics P_{ML} is 7.6%, meaning that the average performance of the trained SVM is very good on the training data. In both figures, the standard deviation has been greatly reduced. The mean absolute value has also been greatly reduced. This is helpful for observations with a circular vision, because, after the ML-modification, the true state is closer to the center of the view. However, since we are more concerned about the actual performance and limits of the SVM model in this paper, only the case with real value will be demonstrated in the following paper. These results reveal that the SVM model has captured the underlying patter in the training data.

More importantly, the SVM can not only reduced the orbit prediction of the training dataset averagely, it can also, at majority of cases, reduce the error of the orbit prediction at a specific epoch. To strictly evaluate this capability, the individual performance metrics is utilized, which is first introduced in one of our earlier studies [3]. The individual metrics p_{ML} at a specific set $\boldsymbol{\Lambda}_i$ of input learning variables is defined as,

$$p_{\mathrm{ML}}|_{\boldsymbol{\Lambda}_i} = 100\% \cdot \left.\frac{|e - \hat{e}_{\mathrm{ML}}|}{|e|}\right|_{\boldsymbol{\Lambda}_i} = 100\% \cdot \left.\frac{|e_{\mathrm{res}}|}{|e|}\right|_{\boldsymbol{\Lambda}_i}. \tag{12}$$

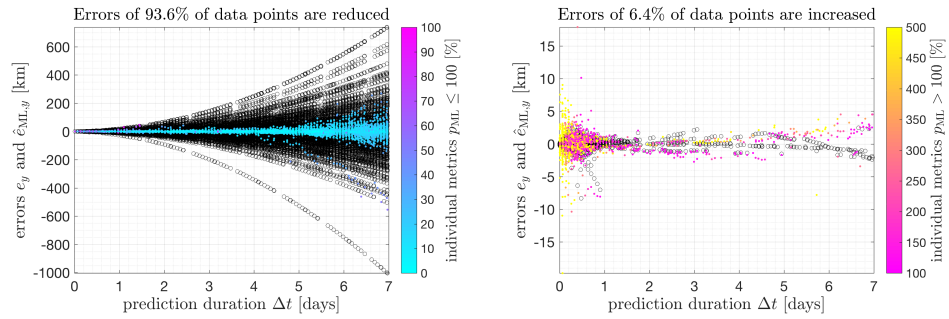


(a) real value



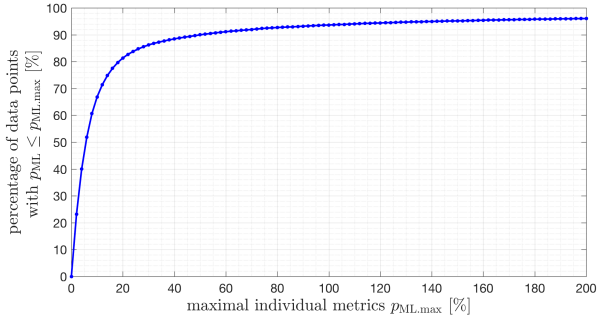
(b) absolute value

Fig. 5. Learning performance of the trained SVM model on the training data in week 1–3.



(a) Increased data points

(b) Decrased data points



(c) Distribution of p_{ML}

Fig. 6. Individual performance metrics of the training results in Fig. 5.

And in the following paper, the metrics P_{ML} will be referred to as average metrics.

The individual performance of the trained SVM model on the training data, using the same data as in Fig. 5, is demonstrated in Fig. 6, where the black circles represent true errors and the dots colored by p_{ML} represent residual errors. Fig. 6(a) shows the data points whose true errors are reduced by the trained SVM model. Fig. 6(b) shows all the data points whose the residual errors are somehow larger than the original true errors, where $p_{\mathrm{ML}} > 500\%$ are all colored by yellow. Comparing the two figures reveals that 93.6% of data points can been reduced, meaning that the trained SVM model has learned the underlying pattern in the training data. Fig. 6(c) demonstrates the distribution of p_{ML}. The vertical axis is the percentage of the data points in the whole training data, whose p_{ML} are less than the corresponding value $p_{\mathrm{ML.max}}$ represented by the horizontal axis. For example, the point at $(100, 93.6)$ corresponds to the results in Figs. 6(a) and 6(b), meaning p_{ML} of 93.6% of data points are less than 100%, which means the prediction error has been reduced.

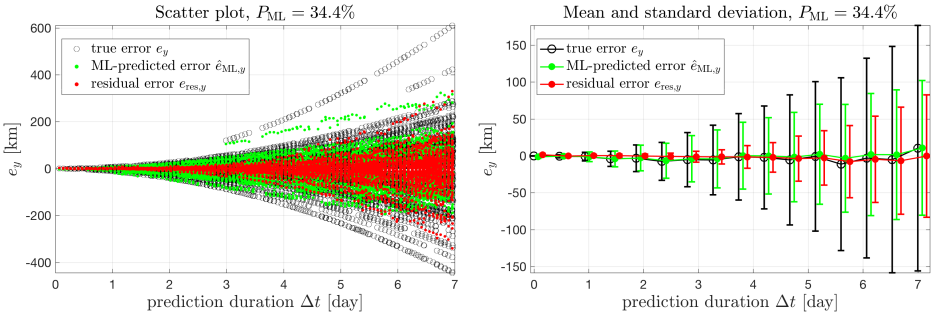
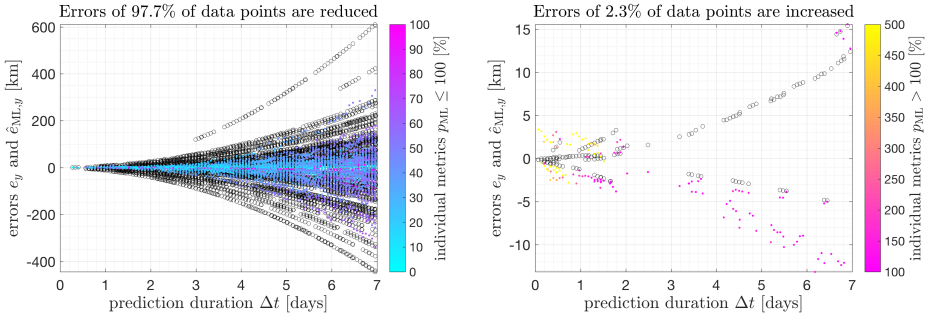
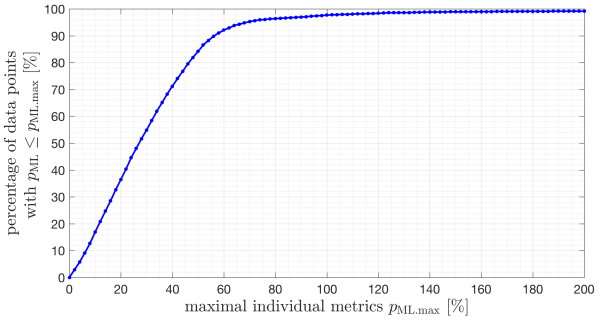


Fig. 7. Average performance of the trained SVM model on the testing data. (week 1–3 ⇒ week 4)



(a) individual metrics p_{ML}



(b) distribution of p_{ML}

Fig. 8. Individual performance of the trained SVM model on the testing data. (week 1–3 ⇒ week 4)

Figs. 7 and 8 demonstrates the performance of the trained SVM model on the testing data not used during the training, which is the data in week 4. The average metrics P_{ML} is 34.4%, in Fig. 7, and the standard deviation of prediction

errors have been greatly reduced. Additionally, in Fig. 8(a), the prediction error of 97.7% percent of testing data has been reduced. In Fig. 8(b), the curve shows that on the testing data, the orbit prediction errors of about 70% of the data have been reduced to less than 40% of the original errors, and 40% of the data to less than 20% of the original errors.

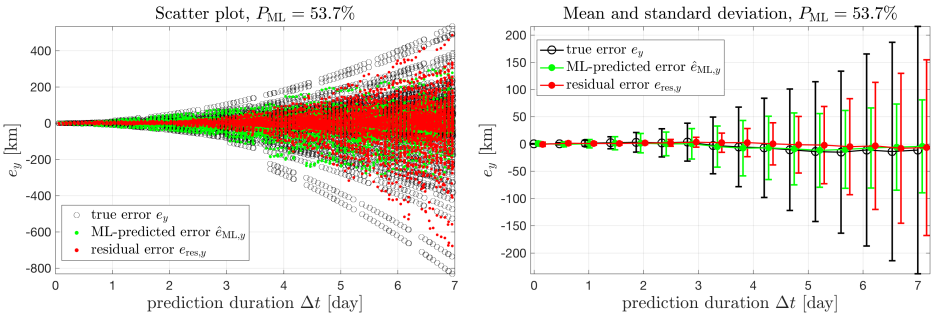
It is interesting to notice that, in both Figs. 6(b) and 8(a), the true and residual errors of those data points with $p_{\mathrm{ML}} > 100\%$ are relatively small, compared with those with $p_{\mathrm{ML}} \leq 100\%$. This indicates that there are indeed some specific cases that the trained SVM model cannot improve the prediction accuracy. Apart from the possibility that it is a limit of the trained SVM model, it is also possible that they are outliers that do not obey the same pattern.

As a conclusion, the validation results using the testing data reveal that the trained SVM model can capture the underlying pattern, and the pattern can be generalized to unknown data with both good average and individual performances.

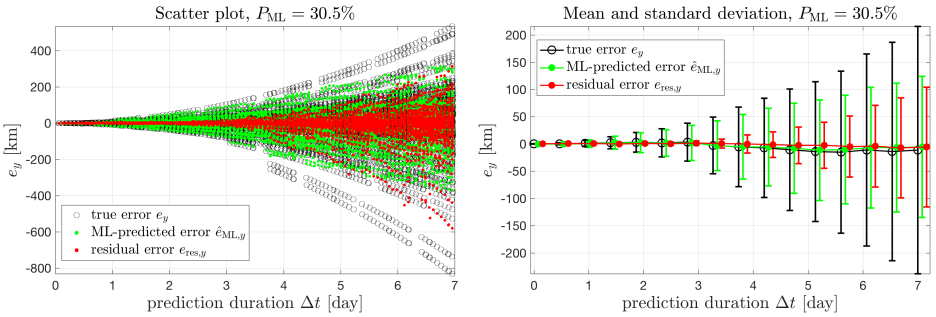
4.2. Effect of Training Data Size

In order to study the effect of the training data size, the simulation is extended to 7 weeks. The testing data is chosen as the data in week 7 and fixed in this subsection. Then, the training data size is chosen as the previous 1 to 6 weeks to investigate the effect of the training data size.

Two examples are demonstrated in Fig. 9. In Fig. 9(a), when using only the historical data in the previous one week, the average performance P_{ML} is 53.7%, meaning around half of the prediction errors are retained. But in Fig. 9(b), when using the historical data in the previous three week, P_{ML} is reduced to 30.5%, meaning about 70% of errors are compensated. It is reasonable because, physically, when more data are available, the embedded information is more likely to be recovered, if it is possible. Additionally, this result also implies that we can actually learn useful knowledge from the historical data, otherwise more data will only introduce more irrelevant noise.



(a) training data: week 6 ⇒ testing data: week 7

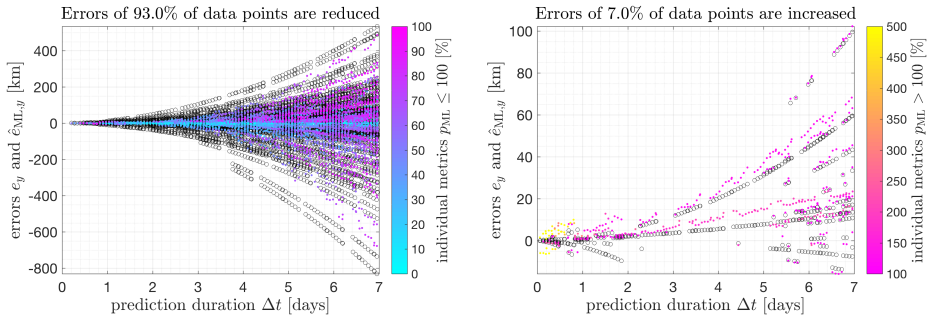


(b) training data: week 4–6 ⇒ testing data: week 7

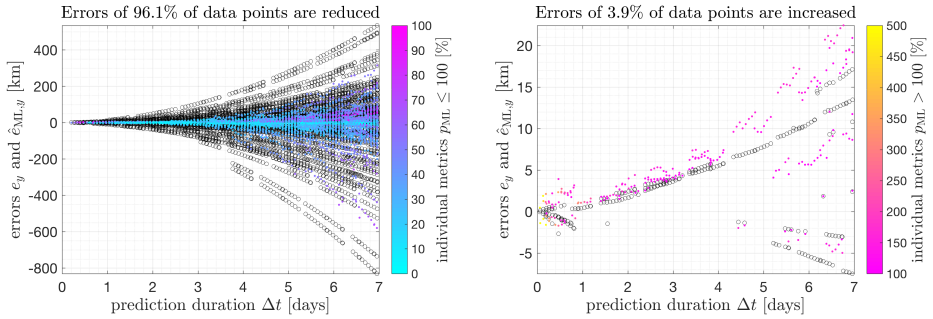
Fig. 9. Average performance of the SVM model trained by different size of training data. The testing data is fixed as the data in week 7.

The individual metrics p_{ML} and the distribution curves are shown in Fig. 10. When only one week is used, the errors of 93.0% of the data points are reduced but many of p_{ML} are large. But when three weeks are used, the errors of 96.1%

of the data points are reduced, and more importantly, most of p_{ML} are small. And Fig. 11 more clearly demonstrates that the errors are more significantly reduced when more training data is used.

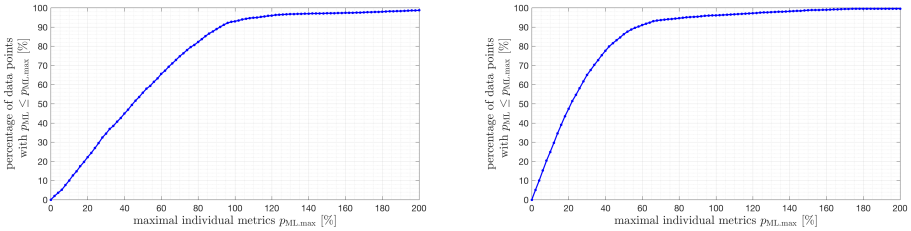


(a) training data: week 6 ⇒ testing data: week 7



(b) training data: week 4–6 ⇒ testing data: week 7

Fig. 10. Individual performance of the SVM model trained by different size of training data. The testing data is fixed as the data in week 7.



(a) training data: week 6 ⇒ testing data: week 7 (b) training data: week 4–6 ⇒ testing data: week 7

Fig. 11. Distribution of p_{ML} in Fig. 10.

Additionally, the time length of the training data is varied from 7 to 42 days, and the average performance is summarized in Fig. 12. At the beginning, as the data size increases, P_{ML} decreases quickly, meaning more and more prediction errors are compensated. However, when the data size is adequate, after reaching 28 days, there are only few improvements even if more data is included. Meanwhile, the curve is not smooth fluctuates a little, which is possibly caused by the fact that the prediction errors in different training data are not identical, thus the resulted SVM may have slightly higher or lower performance. In the following paper, we will use data in four weeks to train the SVM model, and we remark that this choice is meant to be optimal in any sense.

As a conclusion, including more training data will improve the performance of a trained SVM model, but the performance will not be further improved after adequate data has been provided.

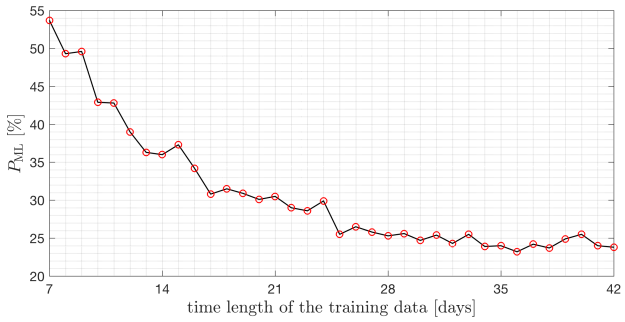


Fig. 12. Curve of the average performance metrics P_{ML} with respect to training data size.

4.3. Effect of Prediction Duration in the Future

In this section, we investigate the time span of the ML-modification, which means the farthest future interval that the trained SVM model could successfully reduce the orbit prediction error. The results will show that there are limits on the correction capability.

As illustrated in Fig. 13, two different schemes are investigated in this subsection. The first scheme is the overall prediction, meaning that the future time interval starts at the current time and ends at a future epoch. In the practice, this scheme resembles the situation of satellite conjunction analysis. We remind that the overall prediction scheme cannot exceed the maximum prediction duration $\Delta t_{\max}$ of the training data, as we have discussed in Sec. 4.1. However, if a longer prediction is required, we can collect more training data with a large enough $\Delta t_{\max}$, and then train a new SVM model. The second scheme is the piecewise prediction, meaning that each time interval in the future is fixed and the trained SVM model only works on orbit predictions that falls onto this interval. In the practice, this scheme resembles the situation to schedule the observation, measurement, or communication, where the moving windows are usually short and piecewise.

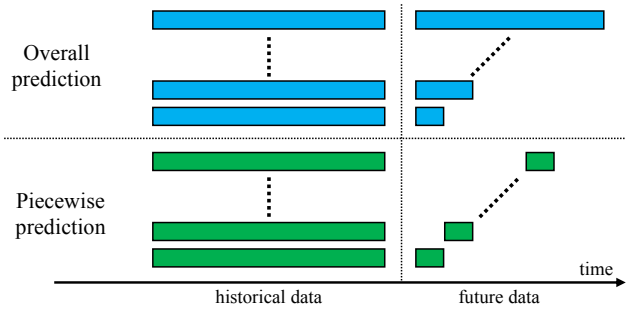


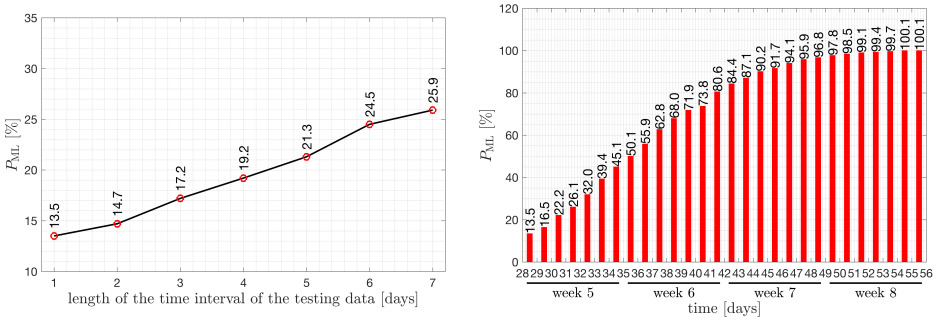
Fig. 13. Illustration of short and integral interval

The training data is fixed to be the data in first four weeks, i.e., weeks 1 to 4. And the testing data is varying in the following 4 weeks, i.e., weeks 5 to 8.

The result of overall prediction scheme is shown in Fig. 14(a). The horizontal axis is the length of the time interval of the testing data, starting from the begining of week 5. P_{ML} keeps increasing as the time interval of the overall prediction scheme increases. The number above each circles indicates the specific average metrics P_{ML} on the corresponding testing data. It is interesting to observe an almost linear relationship in the figure. The trained SVM model has a better performance when the prediction duration is shorter. It is good to observe that the result of the ML approach agrees with intuitive expectations.

In Fig. 14(b), the result of the piecewise prediction scheme is demonstrated, where the prediction windows is fixed as one day. The horizontal axis is the day number, each bar standards for the result on each testing data, and the number at the top of each bar gives the specific average metrics P_{ML} on the corresponding testing data. We note in the piecewise prediction scheme, an orbit prediction can fall beyond Δt of end epoch of the training data, but it still cannot

be based on an estimate within the piecewise prediction interval. In the practice, this corresponds to the situation that new estimates and data are available, but the trained SVM model has not been updated yet. In Fig. 14(b), when the prediction interval is beyond two weeks, more than 80% of orbit prediction errors are left over. So it implies that the SVM model should be updated and orbit prediction should be made not too far in the future.



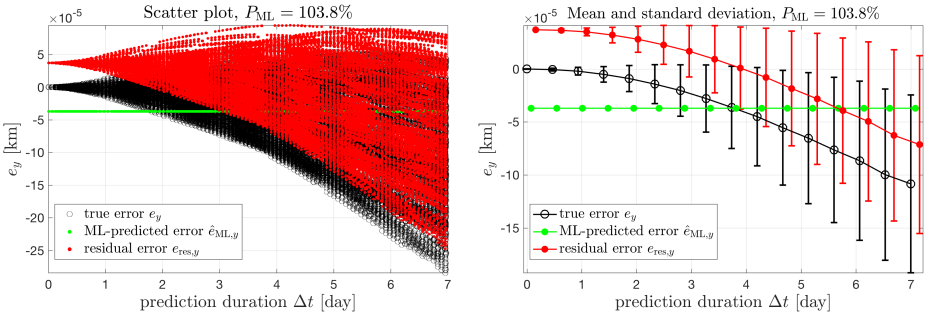
(a) overall prediction scheme

(b) piecewise prediction scheme

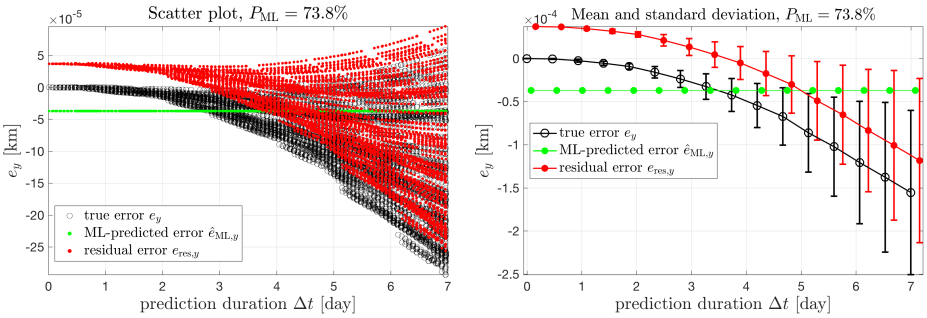
Fig. 14. Performance of the trained SVM model on different testing data.

4.4. Randomness in The System

In this subsection, the effect of the randomness during the measurement process is investigated. We would like the SVM model to capture the relationship between the learning and target variables, rather than to fit the randomness effect.



(a) training data



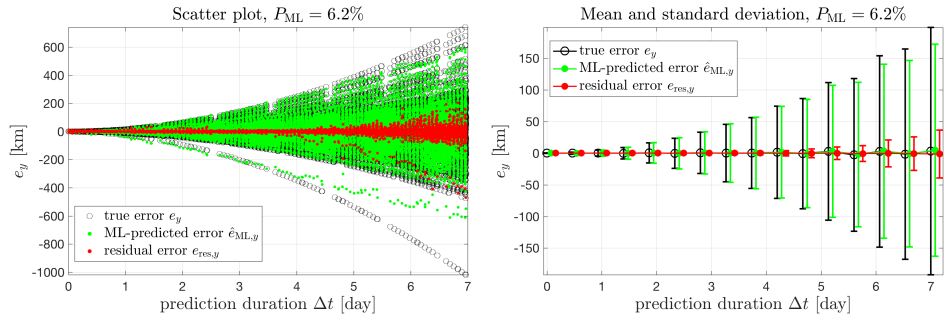
(b) testing data

Fig. 15. Performance of the SVM model when there is no model errors or measurement errors. (week 1–4 $\Rightarrow$ week 5)

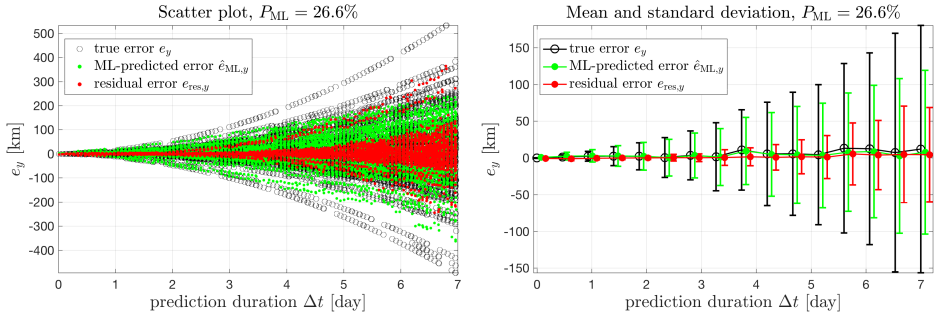
When the assumed model is identical to the true model, and the measurement error is zero, the performances of the trained SVM model on the training and testing data are shown in Fig. 15. In this case, the LS estimator will converge

to the true state, and the only error is due to the numerical roundoff and propagation. As shown in the figure, the true error e_y is very small, and the ML-predicted error $\hat{e}_{\mathrm{ML}.y}$ is a constant. As the error is too small, the training of the SVM is quickly terminated after the initialization. In other words, the propagation error has no relation to the designed learning variables, and thus cannot be learned by the SVM model.

In another case, we still assume the true and assumed models are identical, but add the measurement noise. In this way, the LS estimator will not converge to the true state. The results are shown in Fig. 16. The metrics P_{ML} on the training data is just 6.2%, and the residual error is possibly due to the randomness introduced during the measurement. The metrics P_{ML} on the testing data is 26.6%, which means the pattern learned from the training data cannot be completely generated onto the testing data. This is reasonable, because the SVM model is expected to learn the intrinsic error of the estimator, but it cannot completely remove the randomness introduced during the measurement. These two observations implies that the ML approach is possible to reduce but cannot remove the randomness in the system.



(a) training data



(b) testing data

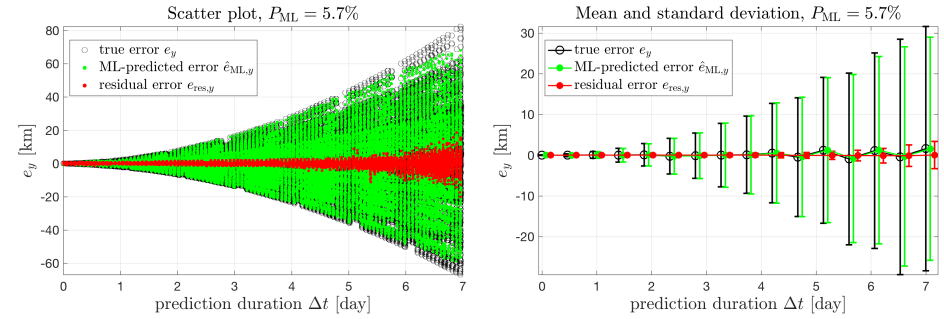
Fig. 16. Performance of the SVM model when there is no model error but only measurement error. (week 1–4 $\Rightarrow$ week 5)

Based on these observations and speculations, the case where there is no measurement error is examined first, which means all σ_α, σ_η, and σ_ρ of simulated ground stations have been set to zero. As a comparison, another case where the σ values are 2 times of the original values are also examined. The training data is fixed to be the data in weeks 1 to 4, and the testing data is fixed to be the data in week 5.

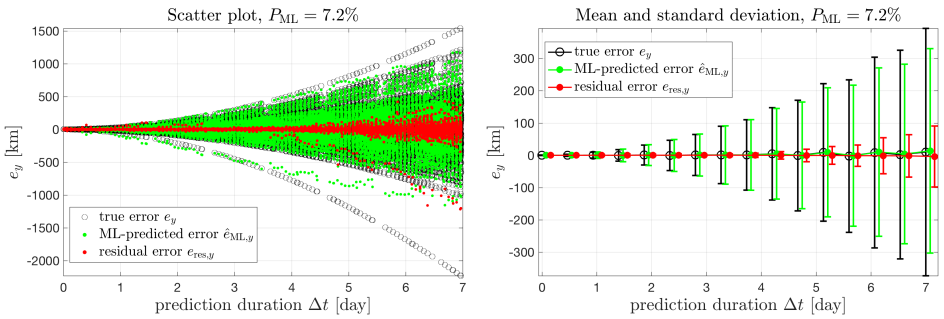
The results of the two cases are demonstrated in Fig. 17. The training data is chosen as the data in weeks 1 to 4, and the testing data is the data in week 5. As revealed by the result on the training data in Fig. 17(a), P_{ML} is 5.7%, meaning that the error cannot be completely compensated even though there is not measurement error. The residual errors are possibly due to many factors, which include but is not limited to:

- the model error between the true and assumed models;
- insufficient learning variables or training data.

The comparison results in Fig. 17(b) shows that the metrics P_{ML} has been slightly increased, which means the per-



(a) 0 times of $(\sigma_\alpha, \sigma_\eta, \sigma_\rho)$ of ground stations. Maximum standard deviation is 3.2 km.



(b) 2 times of $(\sigma_\alpha, \sigma_\eta, \sigma_\rho)$ of ground stations. Maximum standard deviation is 92.2 km.

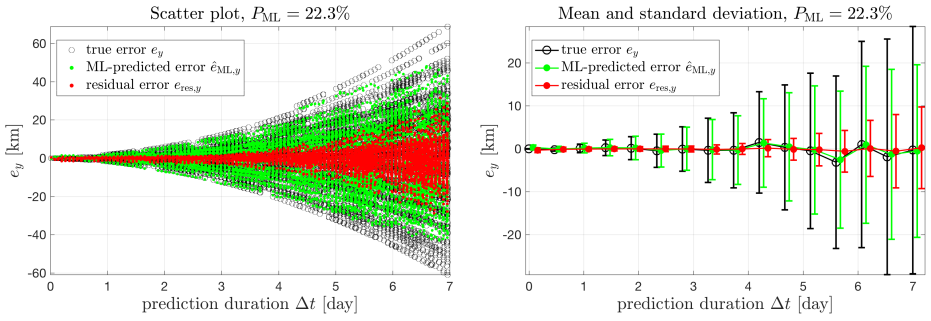
Fig. 17. Performance of the SVM model on training data with different measurement errors. (week 1–4)

formance are worse and more errors are retained. This is reasonable because more estimation errors are introduced in this case.

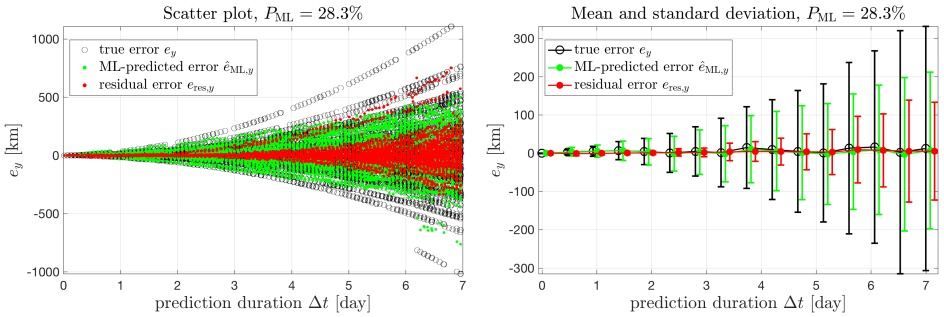
Another significant difference between Figs. 17(a) and 17(b) is that the magnitudes of the error are different. The maximum $|e_y|$ in Fig. 17(a) is only around 80 km, while that in Fig. 17(b) can reach as much as 1500 km. And the magnitudes for standard deviations are also very different, and in Fig. 17(b) it is almost 10 times larger than in Fig. 17(a). We remark that P_{ML} is defined as a relative metrics to evaluate the performance of the trained SVM model within a system with specific measurement errors, so it is not proper to directly compared P_{ML} among different systems. In order to make a more meaningful comparison, we use the maximum standard deviation of the residual error, which is represented by the longest red error bar in figures. The maximum standard deviation is 3.2 km in Fig. 17(a) and 92.2 km in Fig. 17(b), which reveals significant difference when the measurement error is changed.

The performance on the testing data for the two cases are demonstrated in Fig. 18. Similarly, the average metrics P_{ML} show smaller difference, 22.3% to 28.3%, but the maximum standard deviations are very different, 9.3 km to 133.8 km. And in Fig. 19, the curves show the maximum standard deviation of $e_{\mathrm{res},y}$ when the multiple of measurement standard deviations is varied from 0 to 10, on training and testing data separately. As the measurement error grows, the residual errors on both training and testing data are increasing as expected. But when the multiple is larger than 7, the maximum standard deviation starts to decrease on training data. At the first sight, it is possibly due to the overfitting effect of the machine learning algorithm, which means the model tries to fit the noise rather than the real information. However, this phenomenon can also imply that the SVM model starts to reduce measurement errors.

As a conclusion, as long as the training data is adequate, the SVM model can capture the underlying relationship between learning and training variables. We also remark that in the situations of very large randomness in the system, the trained SVM model shows better performance than expected, where further studies are required.

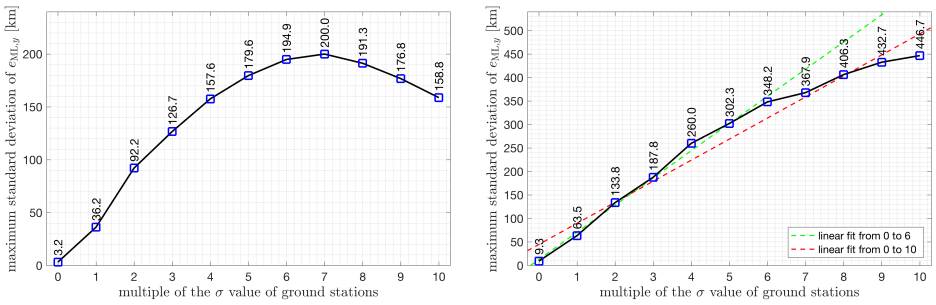


(a) 0 times of $(\sigma_\alpha, \sigma_\eta, \sigma_\rho)$ of ground stations. Maximum standard deviation is 9.3 km.



(b) 2 times of $(\sigma_\alpha, \sigma_\eta, \sigma_\rho)$ of ground stations. Maximum standard deviation is 133.8 km.

Fig. 18. Performance of the SVM model on testing data with different measurement errors. (week 1–4 $\Rightarrow$ week 5)



(a) maximum standard deviation on training data

(b) maximum standard deviation on testing data

Fig. 19. Performance of the trained SVM model with different multiples of the measurement standard deviations of the simulated ground stations. (week 1–4 $\Rightarrow$ week 5)

5. CONCLUSIONS

In this paper, the limits of the machine learning (ML) approach in improving the orbit prediction accuracy is investigated. Specifically, the support vector machine (SVM) model is designed and trained with available variables at a current epoch, and then it is used to reduce the orbit prediction error in a future epoch. The trained SVM model has been shown to be able to capture the underlying relationship between the learning variables and the desired orbit prediction error. Through a series of numerical experiments, some critical limits of the trained SVM model have been demonstrated. The trained SVM model can capture the underlying pattern, and this pattern can be generalized to unknown data with both good average and individual performances. The next question is whether the performance of the SVM model can be further improved and whether there is a limit. The results in this paper show that, including more training data will improve the performance of a trained SVM model, but the performance will not be further improved after adequate data has been used. The results also show that the trained SVM model cannot be generalized to too far

future, so the SVM model should be updated in the practice and orbit prediction should be made not too far.

Future studies include applying the proposed ML approach on RSO catalogs and examining the performance of the current ML approach on multiple RSOs.

ACKNOWLEDGES

The authors would acknowledge the research support from the Air Force Office of Scientific Research (AFOSR) FA9550-16-1-0184 and the Office of Naval Research (ONR) N00014-16-1-2729. Large amount of simulations are supported by the SOE HPC cluster at Rutgers University.

REFERENCES

1. Creon Levit and William Marshall. Improved orbit predictions using two-line elements. *Advances in Space Research*, 47(7):1107–1115, 2011.
2. Jullian Rivera and Xiaoli Bai. Improving the Orbit Propagation Accuracy of Two-Line-Element Satellite. In *67th International Astronautical Congress*, pages 26–30, Guadalajara, Mexico, September 2016.
3. Hao Peng and Xiaoli Bai. (status: Under review) Improving Orbit Prediction Accuracy through Supervised Machine Learning. *Advances in Space Research*, 2017.
4. Hao Peng and Xiaoli Bai. (status: Under review) Recovering Area-to-Mass Ratio From Consistent Errors through Data Mining. *Acta Astronautica*, 2017.
5. Yaser S. Abu-Mostafa, Malik Magdon-Ismail, and Hsuan-Tien Lin. *Learning from Data*. AMLBook, 2012.
6. Tom M. Mitchell. *Machine Learning*. McGraw-Hill Education, 1 edition edition, 1997.
7. Sebastian Pokutta. Machine Learning in Engineering Applications and Trends, 2016.
8. Christos Ampatzis and Dario Izzo. Machine learning techniques for approximation of objective functions in trajectory optimisation. *Proceedings of the International Joint Conference on Artificial Intelligence 2009 Workshop on Artificial Intelligence in Space*, 2009(September), 2009.
9. Jouni Hartikainen, Mari Seppänen, and Simo Särkkä. State-Space Inference for Non-Linear Latent Force Models with Application to Satellite Orbit Prediction. *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 903–910, 2012.
10. Srinagesh Sharma and James W. Cutler. Robust Orbit Determination and Classification: A Learning Theoretic Approach. IPN Progress Report 42-203, Jet Propulsion Laboratory, November 2015.
11. C. Förste, S. L. Bruinsma, R. Shako, O. Abrikosov, F. Flechtner, J.-C. Marty, J.-M. Lemoine, C. Dahle, H. Neumeyer, F. Barthelmes, R. Biancale, G. Balmino, and R. König. A new release of EIGEN-6: The latest combined global gravity field model including LAGEOS, GRACE and GOCE data from the collaboration of GFZ Potsdam and GRGS Toulouse. volume 14, page 2821, April 2012.
12. William M Folkner, James G Williams, Dale H Boggs, Ryan S Park, and Petr Kuchynka. The Planetary and Lunar Ephemerides DE430 and DE431. JPL Interplanetary Network Progress Report 42-196, JPL, 2014.
13. Luc Maisonobe, Véronique Pommier, and Pascal Parraud. Orekit: An Open Source Library for Operational Flight Dynamics Applications. April 2010.
14. Keric Hill, Chris Sabol, and Kyle T Alfriend. Comparison of Covariance Based Track Association Approaches Using Simulated Radar Data. *The Journal of the Astronautical Sciences*, 59(1-2):281–300, June 2012.
15. John L. Crassidis and John L. Junkins. *Optimal Estimation of Dynamic Systems*. Chapman and Hall/CRC Applied Mathematics & Nonlinear Science. CRC Press, second edi edition, 2011.
16. David Vallado, Paul Crawford, Ricahrd Hujsak, and T.S. Kelso. Revisiting Spacetrack Report #3. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*. American Institute of Aeronautics and Astronautics, August 2006.
17. David A. Vallado. *Fundamentals of Astrodynamics and Applications*. The McGraw-Hill Companies, Inc., 1997.

18. Barbara Hammer and Kai Gersmann. A Note on the Universal Approximation Capability of Support Vector Machines. *Neural Processing Letters*, 17(1):43–53, February 2003.

19. Alexander J Smola and Bernhard Schölkopf. A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222, August 2004.

20. Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. Springer New York, New York, NY, 2000.

21. Ke-Lin Du and M. N. S. Swamy. *Neural Networks and Statistical Learning*. Springer London, London, 2014.

22. Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. Springer New York, New York, NY, 2009.