Improvements to Time of Closest Approach Calculation

Elad Denenberg* and Pini Gurfil†
Technion — Israel Institute of Technology, 3200003 Haifa, Israel

DOI: 10.2514/1.G000435

Space debris constitutes a major and growing threat to space missions. To assess the probability of collision, the time of closest approach between the spacecraft and the debris must first be calculated. This paper presents three new methods for the calculation of the time of closest approach. The first is a surrogate-based optimization algorithm, using the Alfano–Negron Close Approach Software as the model, allowing a compromise between calculation speed and accuracy. The second is a generalization of this software, treating a cloud of debris as a continuous object and searching for the time of closest approach over initial conditions as well as time. The third uses Alfano–Negron Close Approach Software generalization as a model for surrogate-based optimization. Using the third method, a strategy for safe operation of a spacecraft cluster in a debris-rich environment is proposed. All methods are demonstrated and compared. It is shown that Alfano–Negron Close Approach Software and the generalization thereof are very fast in estimating the time of closest approach, and that the surrogate-based optimization algorithms are somewhat slower, but are highly accurate.

Nomenclature

$\boldsymbol{a}$	=	coefficients vector
$\boldsymbol{B}(d)$	=	standard monomials set
$\boldsymbol{B}^*(d)$	=	affine standard monomials set
C	=	cubic spline
$\boldsymbol{C}_p$	=	candidate covariance matrix
$\boldsymbol{c}_J^q$	=	candidate point
$D(\boldsymbol{c}_J)$	=	normalized distance score
$\boldsymbol{D}_\sigma$	=	diagonal shift function matrix
d	=	polynomial degree
$\boldsymbol{f}$	=	set of measurements
†	=	given function
f	=	distance function, km^2
g	=	polynomial
$\boldsymbol{K}$	=	Vandermonde basis of the null space
$\boldsymbol{k}$	=	monomial basis
$\boldsymbol{M}$	=	Macauly matrix
M	=	mean anomaly, rad
N	=	number of debris in a cloud
N_f	=	number of samples
N_v	=	number of variables
N'	=	number of satellites in a cluster
p	=	step size parameter
Q	=	quintic spline
$R(\boldsymbol{c}_J)$	=	normalized estimation score
$\boldsymbol{r}_d$	=	relative position vector, km
$\boldsymbol{S}_1$	=	row selection matrix
$\boldsymbol{S}_2$	=	shift matrix
s_c	=	candidate score
t_i	=	time point, sec
w_D	=	distance weight
w_R	=	estimation weight
$\boldsymbol{X}$	=	measurement points matrix
$\boldsymbol{x}_0$	=	initial state vector
$\boldsymbol{Z}$	=	singular value decomposition null space
$\boldsymbol{\epsilon}$	=	perturbing vector
σ_i	=	standard deviation
τ	=	normalized time variable

Received 26 January 2016; revision received 5 April 2016; accepted for publication 24 April 2016; published online 5 July 2016.

*Ph.D. Student, Technion Autonomous Systems and Robotics Program; eladden@campus.technion.ac.il.

†Associate Professor, Faculty of Aerospace Engineering; pgurfil@ae.technion.ac.il. Associate Fellow AIAA.

I. Introduction

THERE has been much work on calculating the probability of collision between two orbiting objects [1,2]. These works focused mostly on the calculation of the probability itself, including methods for calculating the integral of the probability distribution function [3–5], using different assumptions on the motion dynamics [6,7]. All of the mentioned methods require that the calculation of the probability be performed when both objects are at their closest approach. Thus, the first step in assessing the risk of collision is finding the time of closest approach (TCA) between two orbiting objects.

Two methods for finding the TCA of two orbiting objects have been commonly used. The first is propagating both objects with a small time step and finding the minima through exhaustive search. The accuracy of this method depends on the step size. In addition, this method tends to be computationally expensive. The second is using an approximation such as the Alfano–Negron Close Approach Software (ANCAS) [8,9]. ANCAS fits a cubic polynomial to the relative velocity to find the TCA and then a quintic polynomial to the distance for estimating the distance at the TCA. Because ANCAS approximates the distance, it introduces errors. Using this approximation of position in the calculation of probability introduces additional errors to the measurement and orbit propagation errors, and might render the calculation useless, especially when dealing with small satellites, the size of which may be a few orders of magnitude smaller than the overall position estimation error. In addition, the distance approximation error of ANCAS may lead to the false identification of an object as a threat, or far worse, to fail in identifying a threat. Moreover, the accuracy of both methods depends on the time step, but the user has little to no control of the error magnitude or the TCA-related distance.

When attempting to find the TCA of a spacecraft with respect to any of N cataloged objects, filters are used to save calculations. These filters sift the catalog to find only the objects posing a risk to the spacecraft. In 1984, Hoots suggested a set of filters that pick the relevant orbits [10]. These filters were a stepping stone for geometric methods, such as the one proposed by Klinkrad [11]. The first of the Hoots [10] filters starts by detecting objects with a perigee higher than the spacecraft apogee by a required minimum. The second filter is based on the geometry of the orbits, calculating the orbit ellipses closest points. If these points are farther than a required minimum, no conjunction will occur. The third filter checks the time at which the two orbiting objects arrive to the points found in the previous filter.

Healy [12] suggested a method for comparing two large catalogs using parallel computations. The method first propagates all objects of both catalogs using a large time step. Using the known state at each time point, a close approach is determined using a Taylor series expansion. The process is accelerated by sorting the relative pairwise distance between all objects.

Alarcón Rodríguez et al. [13] expanded the results of [10] by defining a critical volume and then developed filters that would sift out the objects not entering this volume. Khutorovsky et al. [14] have suggested some numerical methods for facilitating the search by identifying events within a certain distance, while relaxing the accuracy and then refining the results. Woodburn et al. [15] attempted to take perturbations into account and refine the Hoots filters [10]. These filters either proved to be inaccurate or did little to sift out orbits [16].

All the preceding filters indeed reduce the amount of calculations required; however, one is still left with a couple of thousands of potential objects and must resort back to propagating pairs or using ANCAS. These numbers multiply when dealing with a cluster of N' satellites, and the calculation time of $N' \times N$ can be quite large. In addition, these filters often rely on approximations (e.g., the Hoots time filter [10] and Healy's filters [12]), which can cause the algorithm to fail in finding critical conjunctions, as well as falsely classify nondangerous objects.

In this paper, we propose improvements of the existing TCA calculation methods. First, we suggest a method based on surrogate-based optimization (SBO) for calculating the distance between two orbiting objects. SBO is a general name for a family of analysis and optimization algorithms using simple, computationally efficient functions to describe and search for the minima of complex functions. Our suggested method is a compromise between speed and accuracy, achieving a user-defined error either with respect to time or distance, while maintaining computational efficiency. Second, ANCAS is generalized to allow searching over the initial state as well as over time. This new approach does not break the search of TCA into a series of discrete pairs, but rather treats the catalog as a continuous string of objects. Not segmenting the search allows for greater efficiency and reduces the calculation time significantly. Using the generalized ANCAS with SBO produces a method that is both fast and accurate for calculating the TCA of a spacecraft with respect to any object originating from a catalog of size N. Finally, a strategy is suggested for the quick and accurate calculation of TCA of any spacecraft belonging to a cluster of size N' and a catalog of N objects. This method can thus be applied to cluster flight and formation flight missions (e.g., SAMSON [17]).

II. Methodology

A. ANCAS as a Model for SBO

The search for the critical time is, in fact, a search for the minimum of the distance, that is, the objective function is

$$J = \min_t \|\boldsymbol{r}_d(t)\| \tag{1}$$

where $\boldsymbol{r}_d$ is the relative position vector between two orbiting objects.

Surrogate-based optimization [18] uses surrogate models that are simple and computationally efficient in lieu of highly accurate, yet complicated, functions to search for the minima. The optimization process can be described in four steps: First, a surrogate is fit to the function. Second, a minimum of the surrogate is found. Third, new data points are generated; these are usually the surrogate-found minimum and points in its vicinity. The original function is sampled at these points. Finally, the model is updated with the new data and the process repeats until convergence. Among the most common models in SBO are Kriging [19] and polynomial regression [20].

Polynomial regression works as follows: Let $\mathfrak{f}(\boldsymbol{x})$ be a given function, and let the monomial basis $\boldsymbol{k}$ be a vector containing all monomials up to degree d; $\boldsymbol{k}$ is ordered by rank. In this paper, a reverse lexicographic order is used, for example, for $N_v = 2$ variables, $\{x_1, x_2\}$, and $d = 3$:

$$\boldsymbol{k} = [1, x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3] \tag{2}$$

The polynomial can be written as

$$g(\boldsymbol{x}) = \boldsymbol{a}\boldsymbol{k} \tag{3}$$

in which $\boldsymbol{a} = [a_0, a_1, a_2, \ldots]^T$ The length of both $\boldsymbol{k}$ and $\boldsymbol{a}$ is $(N_v + d)!/(N_v!d!)$.

Let $\boldsymbol{f}$ be a set of samples of the function $\mathfrak{f}(\boldsymbol{x})$ at mesh points $\{\boldsymbol{x}^1, \boldsymbol{x}^2, \ldots, \boldsymbol{x}^{N_f}\}$, that is, $\boldsymbol{f} = [\mathfrak{f}(\boldsymbol{x}^1), \mathfrak{f}(\boldsymbol{x}^2), \ldots, \mathfrak{f}(\boldsymbol{x}^{N_f})]^T$. Then, for each $\boldsymbol{x}^i$ and $\mathfrak{f}(\boldsymbol{x}^i)$, the following linear equation is written:

$$\mathfrak{f}(\boldsymbol{x}^i) = \boldsymbol{a}\boldsymbol{k}_i \tag{4}$$

where $\boldsymbol{k}_i$ is a vector containing the values of monomials at the mesh points. For instance, for the previous example, $\boldsymbol{x}^i = \{x_1^i, x_2^i\}$ and $\boldsymbol{k}_i = [1, x_1^i, x_2^i, (x_1^i)^2, x_1^i x_2^i, \ldots, (x_2^i)^3]$.

All samples can be expressed as

$$\boldsymbol{X}\boldsymbol{a} = \boldsymbol{f} \tag{5}$$

in which

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{k}_1 \\ \boldsymbol{k}_2 \\ \vdots \\ \boldsymbol{k}_n \end{bmatrix} \tag{6}$$

is a matrix of appropriate dimensions.

The coefficients can be estimated using, for example, least squares:

$$\boldsymbol{a} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{f} \tag{7}$$

The greatest advantage of using the polynomial regression as a model is that the fitting becomes very simple, and finding the minimum is often easy as well, if a low-degree model is selected and the problem is univariate.

ANCAS also uses a polynomial fit to estimate the time point of the minimal distance. A distance function is defined as

$$f(t) = \boldsymbol{r}_d(t) \cdot \boldsymbol{r}_d(t) \tag{8}$$

It follows that

$$\dot{f} = 2\dot{\boldsymbol{r}}_d \cdot \boldsymbol{r}_d \tag{9}$$

$$\ddot{f} = 2(\ddot{\boldsymbol{r}}_d \cdot \boldsymbol{r}_d + \dot{\boldsymbol{r}}_d \cdot \dot{\boldsymbol{r}}_d) \tag{10}$$

A close approach will occur when f obtains a local minimum (i.e., when $\dot{f} = 0$ and $\ddot{f} > 0$). To approximate the time t^* for which $\dot{f} = 0$, a piecewise continuous spline is fitted to a sample set of $\dot{f}$. The spline is a set of cubic polynomials, that is, for every given pair of samples of $\dot{f}$ at time points t_i and t_{i+1}, a cubic is fitted to $\dot{f}$. The cubic is defined on the normalized time variable $\tau = [0, 1]$ The transformation between τ and t is given as

$$t(\tau) = t_i + \tau\Delta t \tag{11}$$

in which $\Delta t = t_{i+1} - t_i$ is the time interval.

The cubic polynomial is given by

$$C(\tau) = a_0 + a_1\tau + a_2\tau^2 + a_3\tau^3, \qquad 0 \le \tau \le 1 \tag{12}$$

in which

$$
\begin{aligned}
a_0 &= \dot{f}(t_i)\\
a_1 &= \ddot{f}(t_i)\Delta t\\
a_2 &= -3\dot{f}(t_i) - 2\ddot{f}(t_i)\Delta t + 3\dot{f}(t_{i+1}) - \ddot{f}(t_{i+1})\Delta t\\
a_3 &= 2\dot{f}(t_i) + \ddot{f}(t_i)\Delta t - 2\dot{f}(t_{i+1}) + \ddot{f}(t_{i+1})\Delta t
\end{aligned} \tag{13}
$$

If the acceleration $\ddot{\boldsymbol{r}}_d$ is not available in the specific propagator used (for instance in SGP4), one can estimate it to calculate $\ddot{f}$, or instead use four time points to fit the polynomial. In this case, $0 \le \tau_1, \tau_2 \le 1$. Then [8],

$$
\begin{aligned}
a_0 &= \dot{f}(t_i)\\
a_1 &= (\tau_2^3 - \tau_2^2)(\dot{f}(t_{i+1}) - \dot{f}(t_i)) + (\tau_1^2 - \tau_1^3)(\dot{f}(t_{i+2}) - \dot{f}(t_i))\\
&\quad + (\tau_1^3\tau_2^2 - \tau_1^2\tau_2^3)(\dot{f}(t_{i+3}) - \dot{f}(t_i))/\Lambda\\
a_2 &= (\tau_2 - \tau_2^3)(\dot{f}(t_{i+1}) - \dot{f}(t_i)) + (\tau_1^3 - \tau_1)(\dot{f}(t_{i+2}) - \dot{f}(t_i))\\
&\quad + (\tau_1\tau_2^3 - \tau_1^3\tau_2)(\dot{f}(t_{i+3}) - f(t_i))/\Lambda\\
a_3 &= (\tau_2^2 - \tau_2)(\dot{f}(t_{i+1}) - \dot{f}(t_i)) + (\tau_1 - \tau_1^2)(\dot{f}(t_{i+2}) - \dot{f}(t_i))\\
&\quad + (\tau_1^2\tau_2 - \tau_1\tau_2^2)(\dot{f}(t_{i+3}) - \dot{f}(t_i))/\Lambda\\
\Lambda &= \tau_1^3\tau_2^2 + \tau_1^2\tau_2 + \tau_1\tau_2^3 - \tau_1^3\tau_2 - \tau_1\tau_2^2
\end{aligned} \tag{14}
$$

The roots of Eq. (12) are found, and if a minimum of f is identified at time τ^*, the values of $\boldsymbol{r}_d(t^*)$ are calculated using three quintic polynomials fitted between $\boldsymbol{r}_d(t_i)$ and $\boldsymbol{r}_d(t_{i+1})$. For instance, if $\boldsymbol{r}_d = [r_x, r_y, r_z]^T$, then for r_x,

$$
Q_{r_x}(\tau) = b_0 + b_1\tau + b_2\tau^2 + b_3\tau^3 + b_4\tau^4 + b_5\tau^5, \qquad 0 \le \tau \le 1 \tag{15}
$$

in which

$$
\begin{aligned}
b_0 &= r_x(t_i)\\
b_1 &= \dot{r}_x(t_i)\Delta t\\
b_2 &= 0.5\ddot{r}_x(t_i)\Delta t^2\\
b_3 &= -10r_x(t_i) - 6\dot{r}_x(t_i)\Delta t - 1.5\ddot{r}_x(t_i)\Delta t^2 + 10r_x(t_{i+1})\\
&\quad - 4\dot{r}_x(t_{i+1})\Delta t + 0.5\ddot{r}_x(t_{i+1})\Delta t^2\\
b_4 &= 15r_x(t_i) + 8\dot{r}_x(t_i)\Delta t + 1.5\ddot{r}_x(t_i)\Delta t^2 - 15r_x(t_{i+1})\\
&\quad + 7\dot{r}_x(t_{i+1})\Delta t - \ddot{r}(t_{i+1})\Delta t^2\\
b_5 &= -6r_x(t_i) - 3\dot{r}_x(t_i)\Delta t - 0.5\ddot{r}_x(t_i)\Delta t^2 + 6r_x(t_{i+1})\\
&\quad - 3\dot{r}_x(t_{i+1})\Delta t + 0.5\ddot{r}_x(t_{i+1})\Delta t^2
\end{aligned} \tag{16}
$$

The approximated minimum distance would be

$$
r_{\min} = \sqrt{Q_{r_x}^2(\tau^*) + Q_{r_y}^2(\tau^*) + Q_{r_z}^2(\tau^*)} \tag{17}
$$

Equation (12) can be seen as a particular case of Eq. (3), in which the only variable is t (and therefore, in this case, $N_v = 1$, $\boldsymbol{k} = [1, t, t^2, t^3]$) and the degree of the polynomial is $d = 3$. However, there are two main differences between ANCAS and a polynomial regression [Eq. (3)]. The first is that ANCAS uses additional information $\dot{\boldsymbol{r}}_d$, which is known, and $\ddot{\boldsymbol{r}}_d$, which is sometimes known. The second is that ANCAS is iterative, in the sense that it fits the polynomial to small time intervals to create a piecewise continuous surrogate, rather than fit one polynomial to the whole function f [Eq. (8)].

Though simple to compute, ANCAS introduces errors that can be as large as hundreds of meters [9]. To allow control over the accuracy, we suggest using surrogate optimization with ANCAS as its model.

First, sample f over a given set of time points $\boldsymbol{t} = \{t_1, t_2, \ldots, t_{N_f}\}$. Then, fit the piecewise polynomial [Eq. (12)] to $\dot{f}$ between each set of four time points. Sample f at the time points identified as the ANCAS minima and repeat until convergence. Using this algorithm, both the tolerance in time and the tolerance in distance can be easily tuned. If one uses a propagator enabling the acceleration calculation as well, then in step 5 of Algorithm 1, the cubic will be fitted over each of the two points t_i and t_{i+1}. Repeatedly building the cubic is indeed much slower than fitting it once; however, it is a good compromise between ANCAS and an exhaustive search.

In Sec. III.A, illustrative examples of this algorithm's performance are presented.

B. Generalized ANCAS

Generalizing ANCAS allows spanning the surrogate polynomial over a large number of objects (from tens to tens of thousands), thus enabling the TCA calculation of a given satellite with respect to any member of a debris cloud. Assume that the state of N debris belonging to a certain cloud is known (e.g., all debris originated from a single explosion or collision). Each cloud object is distinguished by its initial state $\boldsymbol{x}_0$ at the satellite epoch. For instance, $\boldsymbol{x}_0$ could be a vector of the debris orbital elements $\boldsymbol{x}_0 = [a(t_0), e(t_0), i(t_0), \Omega(t_0), \omega(t_0), M(t_0)]$, in which a is the semimajor axis, e is the eccentricity, i is the inclination, Ω is the right ascension of the ascending node, ω is the argument of periapsis, and M is the mean anomaly; $\boldsymbol{x}_0$ could also be a vector of position and velocity. The objective function J is then

$$
J = \min_t \|\boldsymbol{r}_d(\boldsymbol{x}_0, t)\| \tag{18}
$$

Finding the TCA as a function of time and $\boldsymbol{x}_0$ is desired. To achieve this, a multivariate polynomial will be fit to

Algorithm 1 SBO — ANCAS

Input: $\boldsymbol{r}_d$ and $\dot{\boldsymbol{r}}$ over a list of time points $\boldsymbol{t}$. Tolerance in time TOL_t and tolerance in distance TOL_d
Output: List of TCA
1 **for** $i = 1$ to length of $\boldsymbol{t}$ **do**
2 define $\boldsymbol{t}_{\text{new}} = (t_i, t_{i+1}, t_{i+2}, t_{i+3})$;
3 **repeat**
4 calculate $\dot{f}$ and $\ddot{f}$ according to Eqs. (9) and (10) for each time point in $\boldsymbol{t}_{\text{new}}$;
5 fit the cubic polynomial Eq. (12) to $\dot{f}$, using Eq. (14) ;
6 **if** a minimum of C exists at time point t_m, **then**
7 estimate $r_{\min}$ using Eq. (17);
8 sample $\boldsymbol{r}_d$ and $\dot{\boldsymbol{r}}_d$ at time point t_m and calculate $\|\boldsymbol{r}_d(t_m)\|$
9 Define $\boldsymbol{t}_{\text{new}} = \boldsymbol{t}_{\text{new}} \setminus (t_j | \max(|t_j - t_m|)) \cup t_m$; /* define the new set of four time points to include t_m and its three closest neighbors */
10 **end**
11 **until** $\{|r_{\min} - \|\boldsymbol{r}_d(t_m)\|| \le \text{TOL}_d$ and $\max(|t_m - t_j|) \le \text{TOL}_t\}$;
12 add newly found minima to TCA list;
13 **end**

$$f(\boldsymbol{x}_0, t) = \boldsymbol{r}_d(\boldsymbol{x}_0, t) \cdot \boldsymbol{r}_d(\boldsymbol{x}_0, t) \tag{19}$$

as in the original ANCAS [see Eq. (8)].

Using such a multivariate polynomial to find the local minima is challenging in two aspects. Unlike a univariate polynomial, generating linear equations that would enable finding the coefficients $\boldsymbol{a}$ is not straightforward. Most often, upon sampling, the problem is either over- or underconstrained [i.e., there are too many or too few sampled points to generate equations, such as Eqs. (12) and (14)]. The second challenge is finding the polynomial's minima; unlike a univariate cubic, root finding is not trivial.

The solution to the first problem is using least squares. Overconstraining and then using Eq. (7) would enable finding the coefficients quite easily. Note that, in constructing the overconstrained linear system at each sampled point, both $f(\boldsymbol{x}_0, t)$ and $\dot{f}(\boldsymbol{x}_0, t)$ are known. If a different propagator is used, $\ddot{f}(\boldsymbol{x}_0, t)$ can also be used to reduce the number of points required for sampling, because each point would produce more equations.

To illustrate this, we consider a degenerate scenario in which only a single orbital element is different among the debris objects. The algorithms demonstrated in this paper are expected to perform well in a more general case, in which more than one element varies. Testing the algorithms in such a general case and comparing them to several filtering methods is beyond the scope of this work.

The case at hand assumes a cloud of N debris, differing only in mean anomaly M at the satellite epoch (this scenario is close to that of a cloud of debris in low Earth orbit approximately a day after an explosion [21] pp. 70–75). In this example, the objective function from Eq. (18) becomes

$$J = \min_t \|\boldsymbol{r}_d(M, t)\| \tag{20}$$

in which $\|\boldsymbol{r}_d(M, t)\|$ forms a surface, a part of which is shown in Fig. 1.

ANCAS fits a piecewise continuous polynomial to the distance function [Eq. (12)]. In the same manner, we suggest fitting a piecewise continuous polynomial of degree d to the distance function f, for example, if a cubic polynomial is used on the preceding example, Eq. (3) would be

$$\begin{aligned} g(M, t) = {} & a_0 + a_1 M + a_2 t + a_3 M^2 + a_4 Mt + a_5 t^2 + a_6 M^3 \\ & + a_7 M^2 t + a_8 M t^2 + a_9 t^3 \end{aligned} \tag{21}$$

Equation (21) can be rewritten as Eq. (3) where $\boldsymbol{a} = [a_0, a_1, \ldots, a_9]^T$ is the coefficients vector and $\boldsymbol{k} = [1, M, t, M^2, Mt, t^2, M^3, M^2t, Mt^2, t^3]$. In case the objects differ in more than one element, the monomial vector $\boldsymbol{k}$ would have to accommodate them as well. The example here was chosen with objects that differ in a single element for ease of demonstration.

In ANCAS, the distance function's first derivative $\dot{f}$ is sampled in given time intervals; here, the hypersurface spanned by Eq. (19) will be sampled in a time and mean anomaly mesh, that is, the distance function $f(M, t)$ and its first derivative $\partial f/\partial t$ will be calculated at each of four neighboring actual debris objects' mean anomalies $M_I, M_{I+1}, M_{I+2}, M_{I+3}$ (in which I is the debris counter, $I = 1, \ldots, N$) and four consecutive time steps $t_J, t_{J+1}, t_{J+2}, t_{J+3}$ (in which J is the time-point counter, $J = 1, \ldots, N_f$). Thirty-two linear equations can be written for each $4 \times 4 = 16$ time/mean anomaly points:

$$\begin{aligned} & a_0 = f(M_I, t_J) \\ & a_0 + a_1\mu_2 + a_2\tau_2 + a_3\mu_2^2 + a_4\mu_2\tau_2 + a_5\tau_2^2 + a_6\mu_2^3 \\ & \quad + a_7\mu_2^2\tau_2 + a_8\mu_2\tau_2^2 + a_9\tau_2^3 = f(M_{I+1}, t_{J+1}) \\ & \vdots \\ & a_2 + a_4\mu_2 + 2a_5\tau_2 + a_7\mu_2^2 + 2a_8\mu_2\tau_2 + 3a_9\tau_2^2 = \frac{\partial f}{\partial t}(M_{I+1}, t_{J+1}) \\ & \vdots \\ & a_2 + a_4\mu_4 + 2a_5\tau_4 + a_7\mu_4^2 + 2a_8\mu_4\tau_4 + 3a_9\tau_4^2 = \frac{\partial f}{\partial t}(M_{I+3}, t_{J+3}) \end{aligned} \tag{22}$$

in which $\mu_i = M_i - M_I$ and $\tau_i = t_i - t_I$, and therefore, $0 \le \tau_i \le t_{J+3} - t_J$ and $0 \le \mu_i \le M_{I+3} - M_I$. This can be formulated in matrix form as in Eq. (7), in which

$$\boldsymbol{X} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & \tau_2 & 0 & 0 & \tau_2^2 & 0 & 0 & 0 & \tau_2^3 \\ 1 & \mu_2 & 0 & \mu_2^2 & 0 & 0 & \mu_2^3 & 0 & 0 & 0 \\ 1 & \mu_2 & \tau_2 & \mu_2^2 & \mu_2\tau_2 & \tau_2^2 & \mu_2^3 & \mu_2^2\tau_2 & \mu_2\tau_2^2 & \tau_2^3 \\ & & & & & \vdots & & & & \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 2\tau_2 & 0 & 0 & 0 & 3\tau_2^2 \\ 0 & 0 & 1 & 0 & \mu_2 & 0 & 0 & \mu_2^2 & 0 & 0 \\ 0 & 0 & 1 & 0 & \mu_2 & 2\tau_2 & 0 & \mu_2^2 & 2\mu_2\tau_2 & 3\tau_2^2 \\ & & & & & \vdots & & & & \end{bmatrix} \tag{23}$$

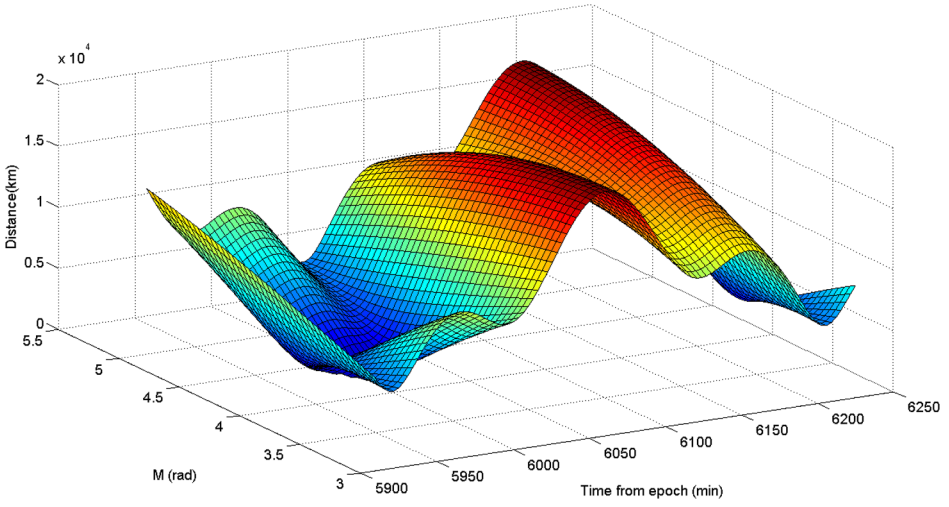


Fig. 1 Surface generated by the distance $\|\boldsymbol{r}_d(M,t)\|$.

The upper block of $\boldsymbol{X}$ includes the distance equations, and the lower block includes the first derivatives equations. In addition, we write

$$\boldsymbol{f} = \begin{bmatrix} f(M_I, t_J) \\ \vdots \\ \frac{\partial f}{\partial t}(M_I, t_J) \\ \vdots \end{bmatrix} \tag{24}$$

Note that, not only the f value is taken into account, but also $\dot{f}$. If $\ddot{f}$ were known, then fewer points would be sampled, because each point would generate three equations. In this example, 48 equations could be generated from $4 \times 4 = 16$ points. The coefficients $a_0, \ldots, a_9$ can be found using Eq. (7). The problem of finding the polynomial's minima can be formulated as solving the following equations:

$$\nabla g(\boldsymbol{x}_0, t) = \begin{pmatrix} \frac{\partial g}{\partial a} \\ \frac{\partial g}{\partial e} \\ \frac{\partial g}{\partial i} \\ \frac{\partial g}{\partial \Omega} \\ \frac{\partial g}{\partial \omega} \\ \frac{\partial g}{\partial M} \\ \frac{\partial g}{\partial t} \end{pmatrix} = 0 \tag{25}$$

Calculating the roots of Eq. (25) is possible using the method described in [22]. The method employs numerical linear algebra and system theory principles to convert the problem of root finding to an eigenvalue problem.

The first step in the algorithm described in [22] is constructing the Macaulay matrix. The Macaulay matrix is a multivariate generalization of the Sylvester matrix. In the example from Eqs. (20) and (21), finding the roots of the following system is desired:

$$\frac{\partial g}{\partial t} = a_2 + a_4 M + 2a_5 t + 2a_7 M^2 + a_6 M t + 3a_9 t^2 = 0 \tag{26}$$

$$\frac{\partial g}{\partial M} = a_1 + 2a_3 M + a_4 t + 3a_6 M^2 + 2a_7 M t + a_8 t^2 = 0 \tag{27}$$

The systems can be rewritten as

$$\boldsymbol{M}(2) = \begin{bmatrix} a_2 & a_4 & 2a_5 & 2a_7 & a_6 & 3a_9 \\ a_1 & 2a_3 & a_4 & 3a_6 & 2a_7 & a_8 \end{bmatrix} \tag{28}$$

$$\boldsymbol{k}(2) = [1, M, t, M^2, Mt, t^2]^T \tag{29}$$

$$\boldsymbol{M}(2)\boldsymbol{k}(2) = 0 \tag{30}$$

where $\boldsymbol{k}(2)$ is the second-degree two-variable monomial vector, and $\boldsymbol{M}(2)$ is the second-degree Macaulay matrix (i.e., it contains the coefficients of all equations up to degree $d = 2$). Each column in $\boldsymbol{M}(d)$ corresponds to a monomial in $\boldsymbol{k}$. The first row of $\boldsymbol{M}(2)$ represents Eq. (26), and the second row represents Eq. (27). To construct $\boldsymbol{M}(3)$, the third-degree Macaulay matrix, Eqs. (26) and (27) are multiplied by a monomial with $d \leq 3$ to generate the equations

$$\frac{\partial g}{\partial t} = a_2 + a_4 M + 2a_5 t + 2a_7 M^2 + a_6 M t + 3a_9 t^2 = 0 \tag{31}$$

$$M\frac{\partial g}{\partial t} = a_2 M + a_4 M^2 + 2a_5 M t + 2a_7 M^3 + a_6 M^2 t + 3a_9 M t^2 = 0 \tag{32}$$

$$t\frac{\partial g}{\partial t} = a_2 t + a_4 M t + 2a_5 t^2 + 2a_7 M^2 t + a_6 M t^2 + 3a_9 t^3 = 0 \tag{33}$$

$$\frac{\partial g}{\partial M} = a_1 + 2a_3 M + a_4 t + 3a_6 M^2 + 2a_7 M t + a_8 t^2 = 0 \tag{34}$$

$$M\frac{\partial g}{\partial M} = a_1 M + 2a_3 M t + a_4 M t + 3a_6 M^3 + 2a_7 M^2 t + a_8 M t^2 = 0 \tag{35}$$

$$t\frac{\partial g}{\partial M} = a_1 t + 2a_3 M t + a_4 t^2 + 3a_6 M^2 t + 2a_7 M t^2 + a_8 t^3 = 0 \tag{36}$$

The third-degree Macaulay matrix $\boldsymbol{M}(3)$ would then become

$$\boldsymbol{M}(3) = \begin{bmatrix} a_2 & a_4 & 2a_5 & 2a_7 & a_6 & 3a_9 & 0 & 0 & 0 & 0 \\ 0 & a_2 & 0 & 2a_5 & a_3 & 0 & 2a_7 & 3a_9 & a_6 & 0 \\ 0 & 0 & a_2 & a_3 & 0 & 2a_5 & a_6 & 2a_7 & 0 & 3a_9 \\ a_1 & 2a_4 & a_3 & 2a_6 & 3a_8 & a_7 & 0 & 0 & 0 & 0 \\ 0 & 0 & a_1 & 2a_4 & 0 & a_3 & 0 & 2a_6 & 0 & a_7 \\ 0 & a_1 & 0 & a_3 & 2a_4 & 0 & 2a_6 & a_7 & 3a_8 & 0 \end{bmatrix} \tag{37}$$

$$\boldsymbol{k}(3) = [1, M, t, M^2, Mt, t^2, M^3, M^2 t, Mt^2, t^3]^T \tag{38}$$

$$\boldsymbol{M}(3)\boldsymbol{k}(3) = 0 \tag{39}$$

The first row of $\boldsymbol{M}(3)$ is Eq. (26) multiplied by the first member of $\boldsymbol{k}$, which is one [Eq. (31)]. The second row of $\boldsymbol{M}(3)$ is Eq. (26) multiplied by the second member of $\boldsymbol{k}$, which is M [Eq. (32)]. Thus, the construction of the Macaulay matrix $\boldsymbol{M}(d)$ is accomplished by multiplying each equation by monomials up to degree d. To find the degree of the Macaulay matrix required for solving the problem at hand, the definitions of standard monomials and affine standard monomials are required.

The set of standard monomials $\boldsymbol{B}(d)$, is the set of monomials corresponding to the linearly dependent columns in $\boldsymbol{M}(d)$. The set of affine standard monomials $\boldsymbol{B}^*(d_G) \subseteq \boldsymbol{B}(d_G)$ is a subset of the standard monomials such that, for any $d \geq d_G$, $x^\alpha \in \boldsymbol{B}^*(d)$ and $x^\beta \in \boldsymbol{B}(d) \setminus \boldsymbol{B}^*(d)$, $|\beta - \alpha| \geq 1$. This also means that, for any $d_2 > d_1 \geq d_G$, $\boldsymbol{B}^*(d_1) = \boldsymbol{B}^*(d_2)$, whereas the degree of the nonaffine standard monomials will grow with a larger degree. The minimum required degree for the solution of the problem is d_G and is found by iteratively constructing the Macaulay matrix with rising degree and checking its columns until the set of affine standard monomials is found.

In the example presented, the affine standard monomials for $\boldsymbol{M}(3)$ are $\boldsymbol{B}(3) = [1, M, t, M^2]^T$, corresponding to the Macaulay matrix first, second, third, and fourth columns. It can be shown that these affine standard monomials are the same for any $d > 3$, and therefore, in this example, $d_G = 3$.

The nullity of $\boldsymbol{M}(3)$ is

$$\text{nullity}(\boldsymbol{M}(3)) = q(\boldsymbol{M}(3))) - \text{rank}(\boldsymbol{M}(3)) = 10 - 6 = 4 \quad (40)$$

in which q is the row length of $\boldsymbol{M}$. The Bézout number is defined as

$$m_b = \prod_i d_i = 4 \quad (41)$$

in which d_i is the degree of each equation [e.g., the degree of Eqs. (26) and (27)]. The Bézout number is equal to the nullity. They represent the number of roots as well as the number of linearly independent columns in the basis of $\boldsymbol{M}$'s null space and the number of standard monomials (as expected, they are equal, and we have four solutions to the system). Therefore, all solutions are affine and no infinite solutions exist.

To find the roots, the Macaulay matrix shifting property is used. The shifting property is formulated as

$$\boldsymbol{S}_1 \boldsymbol{k}\sigma(\boldsymbol{x}) = \boldsymbol{S}_2 \boldsymbol{k} \quad (42)$$

in which the shifting matrix $\boldsymbol{S}_2$ contains the multiplication of the $\boldsymbol{k}$ entries corresponding to the set of affine standard monomials by a shift function $\sigma(\boldsymbol{x})$. The matrix $\boldsymbol{S}_1$ is used to select the desired monomials from $\boldsymbol{k}$. The size of both matrices is identical; the number of rows would be the number of affine standard monomials, and the row length would be the same as the length of $\boldsymbol{k}$. To demonstrate the shifting property, let the shifting function be

$$\sigma(\boldsymbol{x}) = 1 + 2M + 3t \quad (43)$$

Then, the shifting property for the preceding example with $\boldsymbol{k}(3)$ yields

$$\overbrace{\begin{pmatrix} 1 \\ M \\ t \\ M^2 \end{pmatrix}}^{\boldsymbol{S}_1 \boldsymbol{k}} \overbrace{(1 + 2M + 3t)}^{\sigma(\boldsymbol{x})} = \overbrace{\begin{pmatrix} 1 + 2M + 3t \\ M + 2M^2 + 3Mt \\ t + 2Mt + 3t^2 \\ M^2 + 2M^3 + 3M^2 t \end{pmatrix}}^{\boldsymbol{S}_2 \boldsymbol{k}} \quad (44)$$

The selection and shifting matrices that correspond to $\sigma(x)$ are

$$\boldsymbol{S}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \quad (45)$$

$$\boldsymbol{S}_2 = \begin{bmatrix} 1 & 2 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 2 & 3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 2 & 3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 2 & 3 & 0 & 0 \end{bmatrix} \quad (46)$$

Let $\boldsymbol{\kappa}_i$ be a vector of the evaluation of the monomial basis $\boldsymbol{k}$ at the ith root of the system (25) (e.g., $\boldsymbol{\kappa}_i = [1, x_1^*, x_2^*, (x_1^*)^2, \ldots]^T$). Then, the matrix containing all $\boldsymbol{\kappa}_i$ as its columns is defined as the Vandermonde basis of the null space of $\boldsymbol{M}$. For a system with m_B roots,

$$\boldsymbol{K} = [\boldsymbol{\kappa}_1 \quad \cdots \quad \boldsymbol{\kappa}_{m_B}] \quad (47)$$

If the Vandermonde basis of the null space of $\boldsymbol{M}$ is known, the shifting property can be applied as

$$\boldsymbol{S}_1 \boldsymbol{K} \boldsymbol{D}_\sigma = \boldsymbol{S}_2 \boldsymbol{K} \quad (48)$$

in which $\boldsymbol{D}_\sigma$ is a diagonal matrix containing m_B evaluations of the shift function $\sigma(\boldsymbol{x})$,

$$\boldsymbol{D}_\sigma = \begin{bmatrix} \sigma(\boldsymbol{x}_1^*) & & \\ & \ddots & \\ & & \sigma(\boldsymbol{x}_{m_B}^*) \end{bmatrix} \quad (49)$$

Unfortunately, the Vandermonde basis of the null space is not known; however, if $\boldsymbol{V}$ is a nonsingular transformation matrix then $\boldsymbol{K} = \boldsymbol{Z}\boldsymbol{V}$, where $\boldsymbol{Z}$ is any basis of the null space of $\boldsymbol{M}$. Then,

$$\boldsymbol{S}_1 \boldsymbol{Z} \boldsymbol{V} \boldsymbol{D}_\sigma = \boldsymbol{S}_2 \boldsymbol{Z} \boldsymbol{V} \quad (50)$$

Dreesen et al. [22] suggest using singular value decomposition to numerically calculate $\boldsymbol{Z}$.

With some algebraic manipulations, we can write

$$\boldsymbol{V} \boldsymbol{D}_\sigma \boldsymbol{V}^{-1} = (\boldsymbol{S}_1 \boldsymbol{Z})^{-1} \boldsymbol{S}_2 \boldsymbol{Z} \quad (51)$$

Equation (51) is a generalized eigenvalue problem. The eigenvalues are $\boldsymbol{D}_\sigma$'s diagonals. To extract the roots, reconstruct $\boldsymbol{K}$ as

$$\boldsymbol{K} = \boldsymbol{Z}\boldsymbol{V} \quad (52)$$

Normalize each column of $\boldsymbol{K}$ so that the first element would be one. The values in these normalized columns correspond to the values of the vector $\boldsymbol{k}^T = [1, M, t, \ldots]^T$ at the root. Each column represents a different root.

These roots are the critical points of Eq. (21). To find the minima second, the derivative test is used, that is, the Hessian matrix

$$\boldsymbol{H} = \begin{bmatrix} \frac{\partial^2 g}{\partial M^2} & \frac{\partial^2 g}{\partial M \partial t} \\ \frac{\partial^2 g}{\partial M \partial t} & \frac{\partial^2 g}{\partial t^2} \end{bmatrix} \quad (53)$$

must be positive semidefinite. ANCAS uses Eq. (17) to estimate the distance at the minimum point; here, Eq. (21) can be used for that purpose.

It should be noted that the most computationally expensive stage in solving Eqs. (26) and (27) is the iterative phase for finding d_G and the affine standard monomials. In the problem posed in this paper, however, all meshes have the same shaped Macaulay matrix and the same d_G and affine standard monomials. Therefore, finding the affine standard monomials need not be repeated; it can be performed once before the algorithm is executed.

In the rest of this paper, the generalized ANCAS procedure described in this subsection will be referred to as GANCAS. This procedure is demonstrated in Sec. III.B.

C. Generalized ANCAS as a Model for SBO (SBO-GANCAS)

In Sec. II.A, the SBO methodology was used with ANCAS as a model to develop a method for finding the TCA between two orbiting objects. In this section, the general model described in Sec. II.B, together with the SBO methodology, is used for achieving an efficient way of calculating the TCA between a given satellite and a debris cloud. The main difference between the algorithm proposed here and the one described in Sec. II.A is in the candidate selection phase. Unlike Sec. II.A, wherein the only point that was evaluated was the polynomial model's minimum, here, evaluating only the model's minima points $\boldsymbol{x}^* = \{\boldsymbol{x}_0^*, t^*\}$ is not sufficient. The estimated solution's vicinity needs to be explored as well. This is accomplished

through a small perturbation of the found minima, by adding a vector $\boldsymbol{\epsilon}$ (such that $\|\boldsymbol{\epsilon}\| \ll \|\boldsymbol{x}^*\|$). A candidate solution in the minima's vicinity in the qth iteration of the SBO would be

$$\boldsymbol{c}_j^q = \boldsymbol{x}^* + \boldsymbol{\epsilon}_j \tag{54}$$

where $j = 1, \ldots, N_c$ is the candidate counter, and N_c is the total number of candidates generated.

A stochastic method for generating and selecting candidate points for evaluation was suggested in [23]. Candidate points are generated by adding normally distributed vectors of zero mean:

$$\boldsymbol{\epsilon}_j \in \mathcal{N}(0, C_p) \tag{55}$$

The covariance C_p is a diagonal matrix:

$$C_p = \boldsymbol{I}\begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_m^2 \end{bmatrix} \tag{56}$$

in which m is the length of the state $\boldsymbol{x}$. The standard deviations σ_i are

$$\sigma_i = pl_i \tag{57}$$

in which p is a user-defined step size and l_i is the length of the ith hypercube side. For the example given in Sec. II.B, $l_1 = M_{I+3} - M_I$ and $l_2 = t_{J+3} - t_J$.

The candidate's performance is evaluated by two criteria. The first is the relative distance estimation using the piecewise continuous polynomial model [Eq. (3)]. The second is the search space distance between each newly generated candidate and the candidates previously selected for evaluation. Each candidate is assigned a score $s_c \in [0, 1]$ based on these two criteria, with zero being the best score. Then,

$$s_c = w_R R(\boldsymbol{c}_j^q) + w_D D(\boldsymbol{c}_j^q) \tag{58}$$

in which w_R and w_D are weights ($w_d + w_R = 1$), $R(\boldsymbol{c}_j^q)$ is a normalized estimation score of candidate point $\boldsymbol{c}_j^q$ calculated based on Eq. (3),

$$R(\boldsymbol{c}_j) = \frac{g(\boldsymbol{c}_j^q) - \min_m(g(\boldsymbol{c}_m^q))}{\max_m(g(\boldsymbol{c}_m^q)) - \min_m(g(\boldsymbol{c}_m^q))} \tag{59}$$

and $D(\boldsymbol{c}_j)$ is a normalized minimum distance score between $\boldsymbol{c}_j^q$ and any member of a list of candidates previously chosen for evaluation, and is given by

$$D(\boldsymbol{c}_j) = 1 - \frac{\min_k(|\boldsymbol{c}_j^q - \boldsymbol{c}_k^{q-1}|) - \min_{m,k}(|\boldsymbol{c}_m^q - \boldsymbol{c}_k^{q-1}|)}{\max_m(\min_k(|\boldsymbol{c}_m^q - \boldsymbol{c}_k^{q-1}|)) - \min_{m,k}(|\boldsymbol{c}_m^q - \boldsymbol{c}_k^{q-1}|)} \tag{60}$$

In this work, a global search strategy was used. In the beginning of the process, the search is exploration oriented, giving higher priority to generating farther points. Then, slowly, the search becomes exploitation oriented, giving priority to tightly scattered points that score higher on $G(\boldsymbol{c}_j)$. This means that the search starts with a small w_R, and then increases it by Δw_R with each iteration, and with a large p, and then decreases it by Δp with each iteration. The parameters update procedure is described in Algorithm 2.

Algorithm 2 Search strategy

Input: Initial user defined tuning parameters Δw_R and Δp
Output: Updated step size p and estimation weight w_R
$w_R = w_R + \Delta w_R$;
$p = p - \Delta p$;
if $w_R \geq 1$, **then**
  $w_R = 1 - \Delta w_R * 2$;
  $\Delta w_R = \frac{\Delta w_R}{2}$;
end
if $p \leq 0$, **then**
  $p = 0.1$;
end

The candidates with the lowest s_c score are selected for evaluation, and for them, the values of f and $\dot{f}$ are computed. To update the model with the newly computed points, they can be added to previous evaluations of f and $\dot{f}$ in a sequential way, as described by Lawson and Hanson ([24] pp. 207–232).

For the first iteration $q = 0$, let $\boldsymbol{X}_0$ be the matrix representing the initial mesh and $\boldsymbol{f}_0$ be the vector of the samples in $\boldsymbol{X}_0$, as used in Sec. II.B. Let $\boldsymbol{X}_q$ be the mesh matrix and $\boldsymbol{f}_q$ be the sample vector corresponding to $\{\boldsymbol{c}_1^q, \ldots, \boldsymbol{c}_{N_c}^q\}$, the candidate points selected for sampling in iteration q. Then, for the example given in Sec. II.B, in which the polynomial degree was three, $\boldsymbol{X}_q$ and $\boldsymbol{f}_q$ would be

$$\boldsymbol{X}_q = \begin{bmatrix} 1 & \mu_1 & \tau_1 & \mu_1^2 & \mu_1\tau_1 & \tau_1^2 & \mu_1^3 & \mu_1^2\tau_1 & \mu_1\tau_1^2 & \tau_1^3 \\ & & & & & \vdots & & & & \\ 1 & \mu_j & \tau_j & \mu_j^2 & \mu_j\tau_j & \tau_j^2 & \mu_j^3 & \mu_j^2\tau_j & \mu_j\tau_j^2 & \tau_j^3 \\ & & & & & \vdots & & & & \\ 0 & 0 & 1 & 0 & \mu_j & 2\tau_j & 0 & \mu_j^2 & 2\mu_j\tau_j & 3\tau_j^2 \\ & & & & & \vdots & & & & \end{bmatrix} \tag{61}$$

$$\boldsymbol{f}_q = \begin{bmatrix} f(\boldsymbol{c}_1^1) \\ \vdots \\ f(\boldsymbol{c}_{N_f}^q) \\ \frac{\partial f}{\partial t}(\boldsymbol{c}_1^q) \\ \vdots \\ \frac{\partial f}{\partial t}(\boldsymbol{c}_{N_f}^q) \end{bmatrix} \tag{62}$$

To update the polynomial model, in each iteration, the following problem can be solved using least squares, as in Eq. (7):

$$\begin{bmatrix} \boldsymbol{X}_0 \\ \boldsymbol{X}_1 \\ \vdots \\ \boldsymbol{X}_q \end{bmatrix} \boldsymbol{a} = \begin{bmatrix} \boldsymbol{f}_0 \\ \boldsymbol{f}_1 \\ \vdots \\ \boldsymbol{f}_q \end{bmatrix} \tag{63}$$

The problem with iteratively computing the coefficients $\boldsymbol{a}$ of Eq. (63) with Eq. (7) is that the matrix dimensions keep growing, and so does the computational load. An effective alternative is using sequential least squares; by applying orthogonal QR decomposition, the matrix dimensions are kept constant and information is not lost. The QR decomposition of matrix $\boldsymbol{A}$ is formulated as

$$\boldsymbol{A} = \boldsymbol{QR} \tag{64}$$

where $\boldsymbol{Q}$ is an orthogonal matrix ($\boldsymbol{Q}^T\boldsymbol{Q} = \boldsymbol{I}$), and $\boldsymbol{R}$ is an upper triangular matrix. We can write

$$[\boldsymbol{X}, \boldsymbol{f}] = \boldsymbol{Q}[\boldsymbol{\Gamma}, \boldsymbol{d}]$$

where $\boldsymbol{X}$ and $\boldsymbol{f}$ are the same as defined in Eq. (5). Solving the following problem for $\boldsymbol{a}$ is the same as solving Eq. (5) [24]:

$$\boldsymbol{\Gamma a} = \boldsymbol{d} \tag{65}$$

The mesh matrix $\boldsymbol{X}$ and the sample vector $\boldsymbol{f}$ can be augmented in stages as follows: Let the accumulated matrix $[\tilde{\boldsymbol{X}}_q, \tilde{\boldsymbol{f}}_q]$ be defined as

$$[\tilde{\boldsymbol{X}}_q, \tilde{\boldsymbol{f}}_q] = \begin{bmatrix} \boldsymbol{\Gamma}_{q-1} & \boldsymbol{d}_{q-1} \\ \boldsymbol{X}_q & \boldsymbol{f}_q \end{bmatrix} \tag{66}$$

For computational convenience, $\boldsymbol{\Gamma}_0$ and $\boldsymbol{d}_0$ will be an empty matrix and vector. Rather than solving the growing equation (63), one can compute the coefficients $\boldsymbol{a}$ by decomposing

$$[\tilde{\boldsymbol{X}}_q, \tilde{\boldsymbol{b}}_q] = \boldsymbol{Q}[\boldsymbol{\Gamma}_q, \boldsymbol{d}_q] \tag{67}$$

then solving the following with least squares as per Eq. (7):

$$\boldsymbol{\Gamma}_q \boldsymbol{a} = \boldsymbol{d}_q \tag{68}$$

For a large enough q, $\boldsymbol{\Gamma}_q$ and $\boldsymbol{d}_q$ are of constant size. That is, if $\boldsymbol{a}$'s length is n, then the size of $\boldsymbol{\Gamma}_q$ would be $n \times n$ and the size of $\boldsymbol{d}_q$ would be $n \times 1$. In each iteration, the new augmented matrix is built, and the coefficients are updated using Eqs. (68) and (7). This process is repeated until the desired tolerance in distance TOL_d or of the mesh tolerances

$$\boldsymbol{T} = [\text{TOL}_M, \text{TOL}_t] \tag{69}$$

is reached.

The entire search process described in this subsection is presented in Algorithm 3.

From this point onward, Algorithm 3 will be referred to as SBO-GANCAS. An example of the algorithm's performance is presented in Sec. III.B.

D. Using SBO-GANCAS for an $N' \times N$ Problem

In recent years, a number of satellite formation flying missions have been conceived (LISA [25], MMS [26], Prisma [27], SAMSON [17]). Although the design of debris avoidance maneuvers for formations was discussed in the past [28], the problem of assessing the risk of collision between a member of a satellite cluster and cataloged debris has usually been broken down into a series of single satellite–debris mitigation problems. The new method described in Sec. II.C can be implemented on the problem of calculating the minimum distances between N' satellites belonging to a cluster and N debris. Because the cluster operates under maximum distance constraints, the satellites are quite close, and so their local distance minima will also be close in time, and they are likely to encounter the same debris. This is true under the following assumptions:

1) The time period for searching the TCA is short enough, and so the probability of debris interaction is negligible.

2) The spacecraft in the cluster are approximately at the same altitude.

Assumption 1 implies that the debris objects do not interact with one another, generating more debris, or changing the course. In the example given, because all debris objects have the same semimajor axis, this assumption holds. In the more general case, care needs be taken in the selection and definition of the cloud; it is reasonable to assume that the probability of collision between two debris objects originating from the same satellite is low enough. The second assumption suggests that the distance between the cluster members does not change significantly over the short course of the search time frame. Under these assumptions, one can find the local minima of one of the cluster satellites, and then seek the minima of the others in that minima's vicinity, thus diminishing the search space and speeding up the search significantly. The implementation of this strategy is illustrated in Sec. III.C.

III. Examples and Results

A. SBO-ANCAS

The ANCAS and SBO-ANCAS algorithms were tested both on a scenario taken from CelesTrak Satellite Orbital Conjunction Reports Assessing Threatening Encounters in Space (SOCRATES) and a scenario generated with FreeFlyer®. The problems were executed on an Intel i5 computer using MATLAB®. The propagator used was Vallado's SGP4 implementation [29]. Minor changes were introduced to the propagator to allow operation in parallel mode. The tolerances used were $\text{TOL}_t = 10^{-5}$ s and $\text{TOL}_d = 10^{-9}$ km (line 11 of Algorithm 1). The step size Δt in both scenarios was 200 s.

First, the SOCRATES scenario of the COSMOS 1607 and FENGYUN 1C DEB conjunction problem was examined and the results were compared with SOCRATES and FreeFlyer. The TCA was 2015-July-15 20:34:45.449 Coordinated Universal Time (UTC). The results are presented in Fig. 2a and Table 1. Second, two objects were generated using FreeFlyer. The results are presented in Fig. 2b and Table 1. Figure 2 shows the relative distance between the objects and the local minima identified by both algorithms. In Table 1, the real global minimum and the one identified by each algorithm are compared, as well as the number of times SGP4 was called. Because SGP4 is an analytical function, counting the number of times it was invoked is an adequate metric of the computational burden. Should an integrator be used as propagator, a different metric, such as the run time, would perhaps be more suitable.

As expected, SBO-ANCAS requires about twice the function calls ANCAS requires. However, SBO-ANCAS is very accurate and is able to find the exact minimum. Note that, in the SOCRATES example, ANCAS failed to properly identify the TCA within the given step size of 200 s.

In both examples, the average number of surrogate optimization cycles per time interval (as depicted in Algorithm 1, lines 3–11) was 5.13. The maximum amount of surrogate optimization loops per time

Algorithm 3 SBO with general ANCAS as a model

Input: List of points in M_I and t_J forming a mesh grid, mesh tolerance vector $\boldsymbol{T}$, distance tolerance TOL_d
Output: List of minima points
1 Divide the input into a hypercube mesh;
2 **for** all hypercubes, **do**
3 $q = 0$
repeat
4 calculate the distance function (19) for all points in the hypercube;
5 use Eqs. (51) and (52) to search for candidate minima;
6 **if** minima exist, **then**
7 generate candidates Eq. (54) as per Eq. (56);
8 update parameters for candidate selection as described in Algorithm 2;
9 score candidates Eq. (58) and select N_c best scoring for evaluation;
10 evaluate candidates Eq. (19);
11 update model using sequential least squares with Eq. (65);
12 **end**
13 $q = q + 1$
14 **until** convergence under tolerances given;
15 add found minima to list;
16 **end**

Table 1 Comparison of Algorithms for two objects

Algorithm	Minimum distance, km	TCA from epoch, min	Function calls	TCA error, s	Approximation error, km
		COSMOS 1607 and FENGYUN 1C DEB			
Real minimum	0.1047	3388.75	— —	— —	— —
ANCAS	140.74	3128.79	5856	259.96	140.64
SBO-ANCAS	0.1047	3388.75	12502	2.26e − 06	1.40e − 05
		FreeFlyer generated example			
Real minimum	1.4198	2880.01	— —	— —	— —
ANCAS	23.0925	2879.98	6390	0.02	21.67
SBO-ANCAS	1.4198	2880.01	9302	7.43e − 06	1.72e − 07

interval in the COSMOS 1607 and FENGYUN 1C DEB problem was 13, and in the FreeFlyer example it was 10.

In addition, the sensitivity of both algorithms to the size of the time step was examined. The results are presented in Figs. 3 and 4. It can be seen that, in both cases, ANCAS demonstrated sensitivity to the time step, whereas SBO-ANCAS was robust.

An attempt was made to find the time step with which the accuracy of ANCAS would resemble that of SBO-ANCAS. Running ANCAS with a time step lower than 0.001 s, which demanded over one billion function counts, did not achieve the desired goal.

B. Generalized ANCAS and SBO-GANCAS

In this subsection, the performance of ANCAS and SBO-ANCAS is compared with GANCAS and SBO-GANCAS. Based on the second example of the previous subsection, an example of a satellite debris cloud was generated as described in Sec. II.B. In this example, 2000 debris varying in mean anomaly from $-\pi \le M \le \pi$ were generated from the object defined as debris. ANCAS and SBO-ANCAS find the TCA of satellite–debris pairs, and so an exhaustive search over all 2000 debris was performed, annotated EX-ANCAS and EX-SBO-ANCAS. The tolerances used were the same as in the preceding subsection; in addition, for GANCAS and SBO-GANCAS, $\text{TOL}_M = 10^{-8}$.

Unlike the example shown in Sec. II.B, for better accuracy, the model used here is a quadratic polynomial, that is, in Eq. (3), $\boldsymbol{k} = [1, M, t, M^2, \ldots, t^4]$ and $\boldsymbol{a} = [a_0, a_1, \ldots, a_{14}]^T$.

The sampled intervals were the same as used in the previous example. For GANCAS and SBO-ANCAS, there was no need to sample all 2000 debris, however, and only 100 were sampled. The rest of the debris positions were estimated using the polynomial.

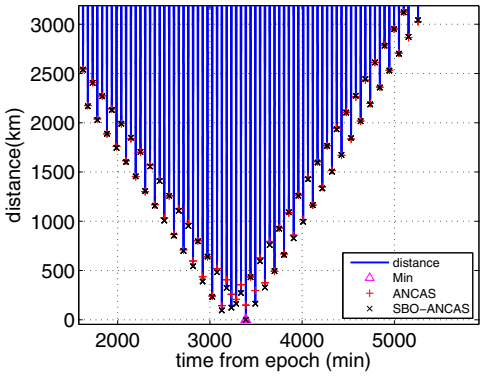
The Macaulay matrices were written, and it was found that the degree at which the standard affine monomials are found is $d_G = 5$. $\boldsymbol{M}(5)$ has nine affine standard monomials ($1, M, t, M^2, Mt, t^2, M^3, M^2t$, and M^4). Notice that here the first linearly dependent columns are not the first four, as was the case in Sec. II.B. Because a higher dimension polynomial fit was used, the system had nine affine roots. The appropriate selection matrix $\boldsymbol{S}_1$ was written, and a shifting matrix $\boldsymbol{S}_2$ for a random function

$$\sigma(M, t) = \boldsymbol{u}\boldsymbol{x} \tag{70}$$

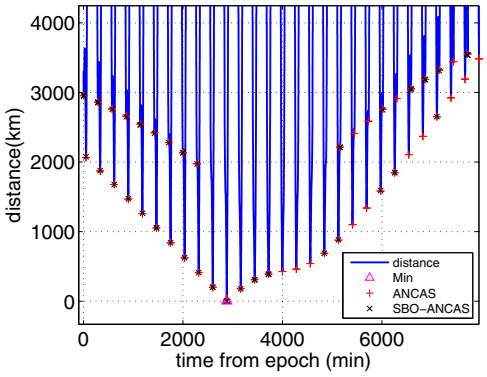
in which $\boldsymbol{u}$ is a vector of uniform distribution on the interval [0, 1] (e.g., $\boldsymbol{u} = [u_M, u_t]^T$), and $\boldsymbol{x} = [M, t]$ is the state. The Macaulay matrix $\boldsymbol{M}(5)$, the selection matrix $\boldsymbol{S}_1$, and the shifting matrix $\boldsymbol{S}_2$ are

$$\boldsymbol{S}_1 = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix} \tag{71}$$

$$\boldsymbol{S}_2 = \begin{bmatrix} 0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&u_M&u_t&0&0&0&0&0&0&0 \end{bmatrix} \tag{72}$$

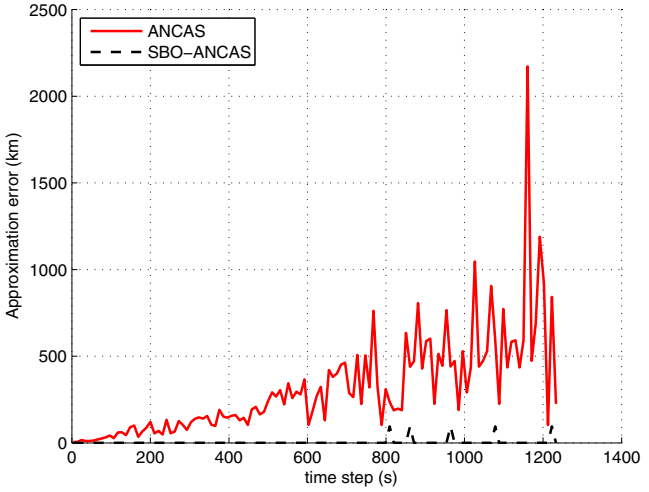


a) COSMOS 1607 and FENGYUN 1C DEB

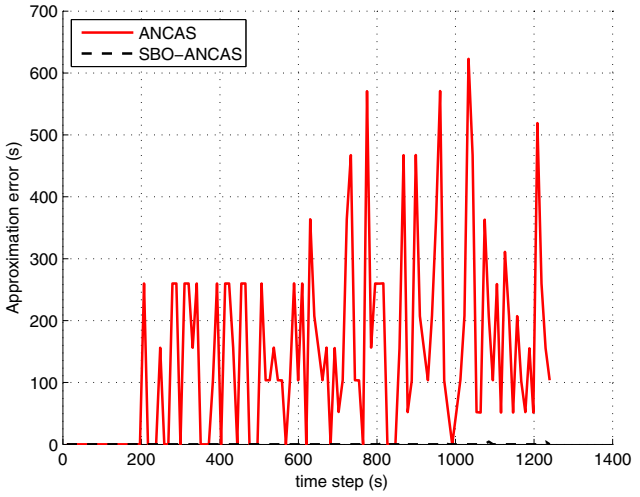


b) FreeFlyer® Generated Example

Fig. 2 ANCAS and SBO-ANCAS results.

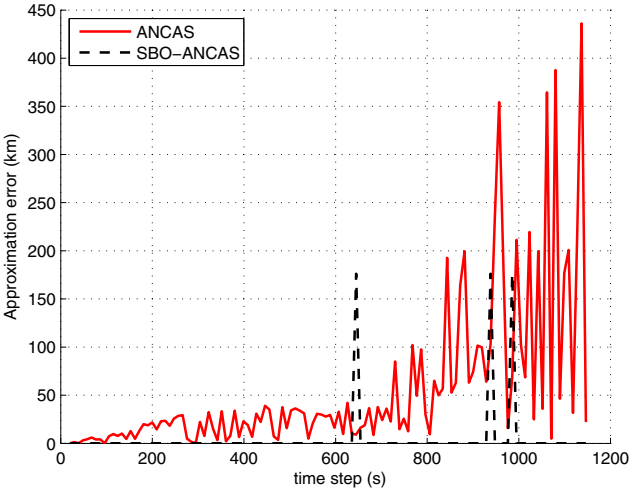


a) Error in Distance Approximation

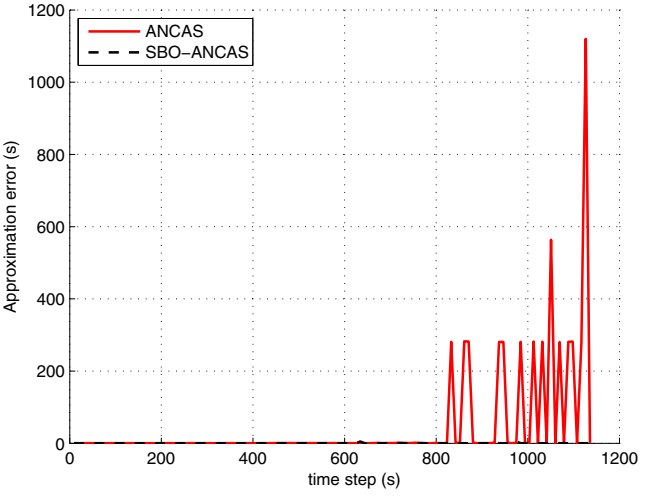


b) Error in Time

Fig. 3 Sensitivity to time step in the COSMOS 1607 and FENGYUN 1C DEB example.



a) Error in Distance Approximation



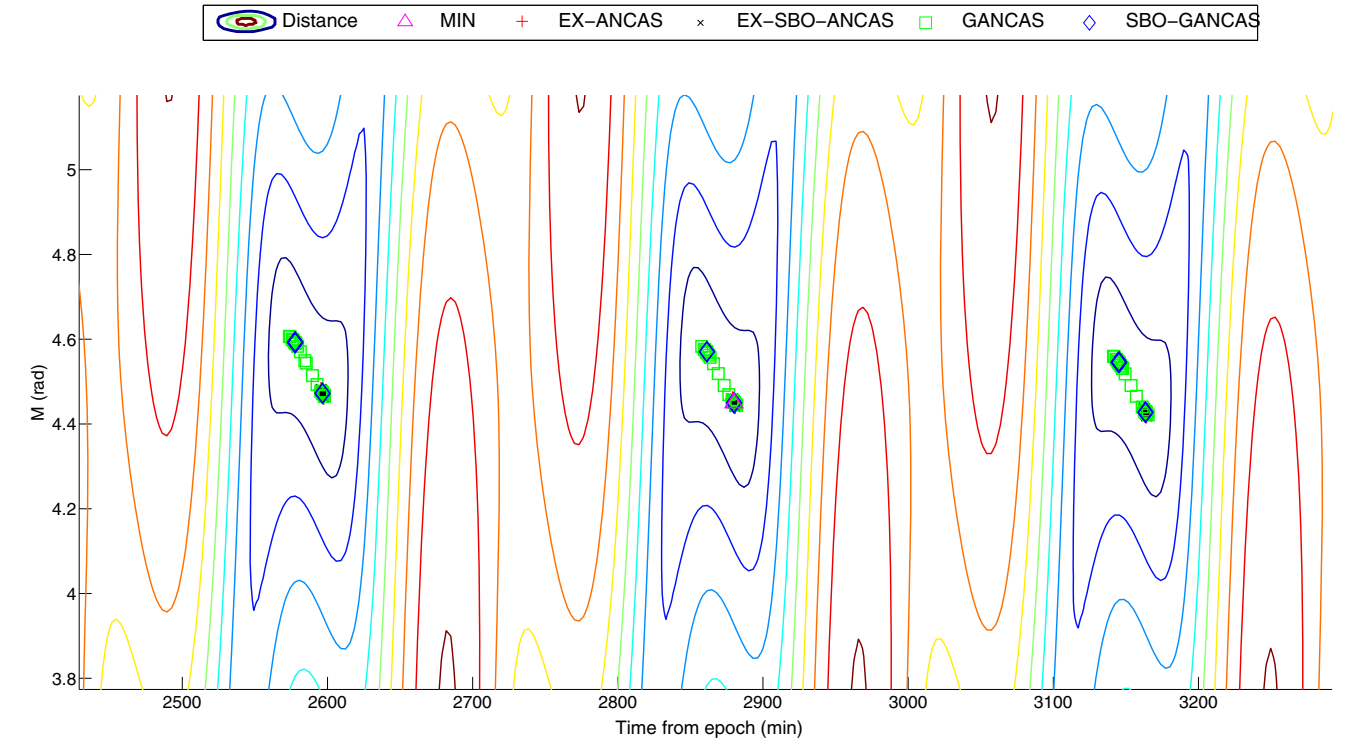
b) Error in Time

Fig. 4 Sensitivity to time step in the FreeFlyer generated example.

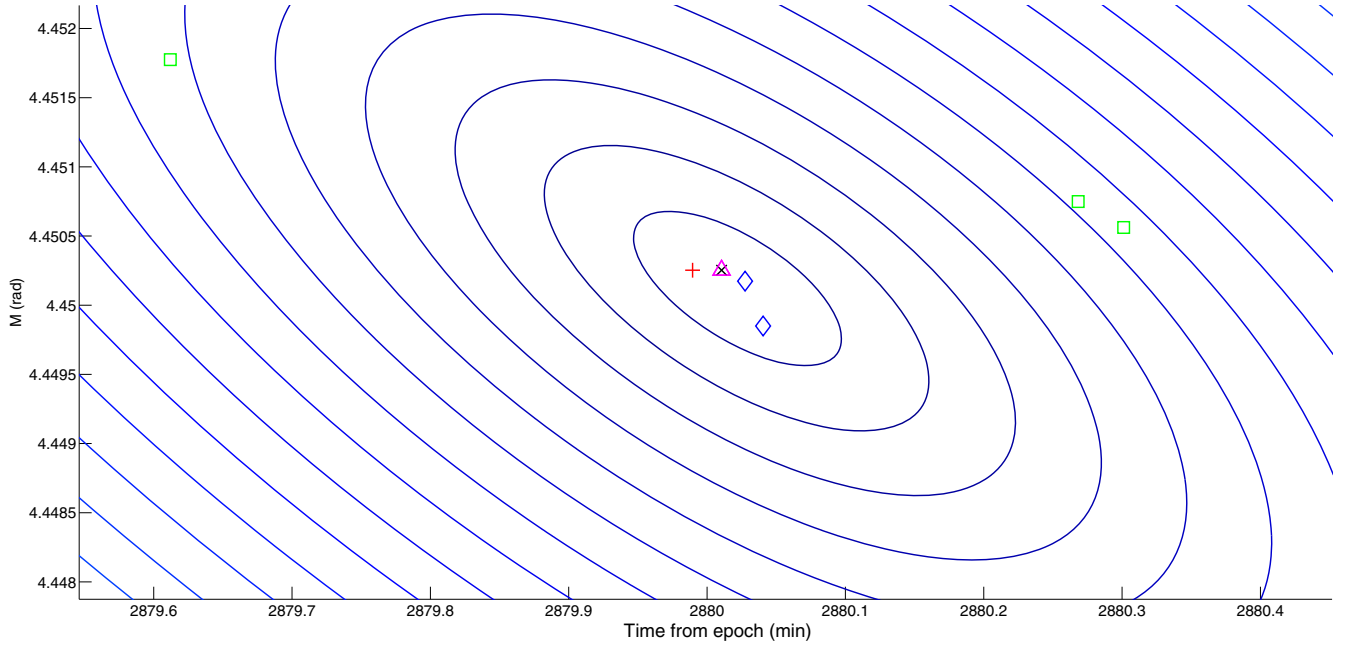
$$\boldsymbol{M}(5) = \begin{bmatrix}
a_1 & 2a_3 & a_4 & 3a_6 & 2a_7 & a_8 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & a_1 & 0 & 2a_3 & a_4 & 0 & 3a_6 & 2a_7 & a_8 & 0 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & a_1 & 0 & 2a_3 & a_4 & 0 & 3a_6 & 2a_7 & a_8 & 0 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & a_1 & 0 & 0 & 2a_3 & a_4 & 0 & 0 & 3a_6 & 2a_7 & a_8 & 0 & 0 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} & 0 & 0 \\
0 & 0 & 0 & 0 & a_1 & 0 & 0 & 2a_3 & a_4 & 0 & 0 & 3a_6 & 2a_7 & a_8 & 0 & 0 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} & 0 \\
0 & 0 & 0 & 0 & 0 & a_1 & 0 & 0 & 2a_3 & a_4 & 0 & 0 & 3a_6 & 2a_7 & a_8 & 0 & 0 & 4a_{10} & 3a_{11} & 2a_{12} & a_{13} \\
a_2 & a_4 & 2a_5 & 2a_7 & a_6 & 3a_9 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & a_2 & 0 & a_4 & 2a_5 & 0 & 2a_7 & a_6 & 3a_9 & 0 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14} & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & a_2 & 0 & a_4 & 2a_5 & 0 & 2a_7 & a_6 & 3a_9 & 0 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & a_2 & 0 & 0 & a_4 & 2a_5 & 0 & 0 & 2a_7 & a_6 & 3a_9 & 0 & 0 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14} & 0 & 0 \\
0 & 0 & 0 & 0 & a_2 & 0 & 0 & a_4 & 2a_5 & 0 & 0 & 2a_7 & a_6 & 3a_9 & 0 & 0 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14} & 0 \\
0 & 0 & 0 & 0 & 0 & a_2 & 0 & 0 & a_4 & 2a_5 & 0 & 0 & 2a_7 & a_6 & 3a_9 & 0 & 0 & a_{11} & 2a_{12} & 3a_{13} & 4a_{14}
\end{bmatrix} \tag{73}$$

Table 2 Comparison of algorithms for a $1 \times N$ scenario

Algorithm	Minimum distance, km	Debris M, rad	TCA from epoch, min	Function calls	TCA error, s	Approximation error, km
Real minimum	1.4198	4.4503	2880.01	— —	— —	— —
EX-ANCAS	23.0925	4.4722	2879.98	12,780,000	0.02	21.6727
EX-SBO-ANCAS	1.4198	4.4503	2880.01	18,954,530	2.26e − 06	1.40e − 05
GANCAS, found	7.7347	4.6003	895.58	639,000	1985.43	6.3149
GANCAS, nearest	19.6895	4.6010	895.28	63,900	1985.73	18.2697
SBO-GANCAS, found	1.1182	4.4502	2880.01	8,742,776	0.0001	0.3016
SBO-GANCAS, nearest	1.4198	4.4503	2880.01	93,020	2.26e − 06	1.40e − 05



a) General Results



b) Global Minimum

Fig. 5 General $1 \times N$ algorithm results.

Using Eqs. (51) and (52), local minima were found in every mesh hypercube. The SBO-GANCAS used quadratic polynomials and the same parameters as in GANCAS. The step size parameters described in Sec. II.C were used in a global search strategy. We started with $w_R = 0.5$ and $p = 0.5$, increased w_R and lowered p, as described in Algorithm 2, where $\Delta p = 0.01$ and initially $\Delta w_R = 0.2$.

The results are presented in Fig. 5 and Table 2. Figure 5 shows sections of the hypersurface generated by $\|\boldsymbol{r}_d(M, t)\|$, and the local minima identified by the different algorithms. Table 2 presents each algorithm run time and the global minimum the algorithm identified.

It can be seen in the table that GANCAS performed very fast; however, it also lacked in accuracy because it found a different global minimum. Both GANCAS and SBO-GANCAS find the minimum distance between a satellite and a continuous debris cloud (the algorithms treat M as a continuous variable). Therefore, once the minimum is found using GANCAS and SBO-GANCAS, SBO-ANCAS is run on the debris closest to the 10 best-found minima to

Table 3 Global minima of all satellites in cluster

Satellite	Minimum, km	TCA from epoch, min	Debris M
1	1.4198	2880.01	4.4503
2	4.5644	2690.89	0.2814
3	1.4851	2879.82	4.4597
4	2.7051	1084.17	2.5056
5	1.1143	989.56	0.4227

Table 4 Comparison of $N \times N'$ algorithms

Algorithm	Run time, s
ANCAS	2294.23
SBO-ANCAS	4352.36
GANCAS	315.65
SBO-GANCAS	2745.56
Suggested strategy	1659.83

find the minimum distance between the found minimum and the discrete number M representing a debris. It is to be noted that, as in the ANCAS/SBO-ANCAS case, SBO-GANCAS exhibited robustness to the mesh size compared with GANCAS.

Because SBO-GANCAS introduces random processes, the results presented here were examined statistically. SBO-GANCAS was able to identify the global minimum in 99.48% of 500 runs. The algorithm found the global minimum within the 10 smallest local minima in 96.89% of the runs. The results presented here are typical and do not deviate significantly from the average result over 500 runs.

C. Using SBO-GANCAS for an $N' \times N$ Problem

To demonstrate the strategy suggested in Sec. II.D, the example from Sec. III.B was extended here to a problem of a five-satellite cluster. The object defined as a satellite in the second example of Sec. III.A was duplicated with slightly different mean anomaly to create a five-satellite cluster. The distance between each cluster member was 50 km.

Table 3 presents the global minimum distance of each member of the cluster and a debris cloud. These minima were found using SBO-GANCAS for each of the satellites. The satellites local minima were close to each other.

A search area of ± 30 min in time and ± 0.2 rad in debris mean anomaly is sufficient to find all the other satellites local minima. Notice that care needs to be taken when the minimum is near $M_0 = 0$ or $M_0 = 2\pi$; the search neighborhood should be modulo 2π to accommodate the fact that the cloud of debris is circular, (i.e., if, for instance, a local minima of satellite 3 is found at about 5300 min with a debris for which initial mean anomaly was 0.1, then the search area must include 6.1 rad as well).

Table 4 presents a runtime comparison between the algorithms presented in this work and the suggested strategy. An exhaustive search using SBO-ANCAS for all the debris and cluster members used approximately 1.25 h; an exhaustive search employing the propagator with very low time step would take far longer. With this strategy, because the search space is narrowed, the global minimum of all cluster members is found within about 27 min of run time. The global minima of all the members of the cluster were identified using this strategy.

IV. Conclusions

Using ANCAS as a model for SBO is a compromise between accuracy and speed. For the computation of the TCA, this method is slower than ANCAS, but it allows one to control the accuracy and is much faster than propagating the two objects. The generalization of ANCAS can treat a cloud of debris as a continuous rather than a discrete list of objects. Using this method as a model for SBO yields an efficient and accurate way of calculating the TCA of a spacecraft relative to any member of a debris cloud. The advantages of this method are emphasized when dealing with an operation of a spacecraft cluster in a debris-rich environment. This is so because the new strategy narrows down the search space by relying on the cluster's known geometry. For a spacecraft cluster, the search of TCA using the methods and strategy proposed in this work can be even faster than using ANCAS, yet far more accurate.

References

[1] Alfano, S., "Review of Conjunction Probability Methods for Short-Term Encounters," *AAS Astrodynamics Specialist Conference*, American Astronautical Soc., Springfield, VA, 2007, pp. 07–148.

[2] Chan, F. K., *Spacecraft Collision Probability*, Aerospace Press, El Segundo, CA, 2008, pp. 1–12. doi:10.2514/4.989186

[3] Alfano, S., "A Numerical Implementation of Spherical Object Collision Probability," *Journal of Astronautical Sciences*, Vol. 53, No. 1, 2005, pp. 103–109.

[4] Patera, R. P., "Calculating Collision Probability for Arbitrary Space Vehicle Shapes via Numerical Quadrature," *Journal of Guidance, Control, and Dynamics*, Vol. 28, No. 6, 2005, pp. 1326–1328. doi:10.2514/1.14526

[5] Foster, J. L., Jr., "The Analytic Basis for Debris Avoidance Operations for the International Space Station," *Proceedings of the Third European Conference on Space Debris*, Vol. 1, edited by Huguette, S.-L., ESA Publications Division, ESA-SP-473, Noordwijk, The Netherlands, March 2001, pp. 441–445.

[6] Patera, R. P., "Satellite Collision Probability for Nonlinear Relative Motion," *Journal of Guidance, Control, and Dynamics*, Vol. 26, No. 5, 2003, pp. 728–733. doi:10.2514/2.5127

[7] Alfano, S., "Addressing Nonlinear Relative Motion for Spacecraft Collision Probability," *AAS/AIAA Astrodynamics Specialist Conference*, AIAA Paper 2006-6760, 2006. doi:10.2514/6.2006-6760

[8] Alfano, S., and Negron,, D., Jr., "Determining Satellite Close Approaches," *Journal of Astronautical Sciences*, Vol. 41, No. 2, 1993, pp. 217–225.

[9] Alfano, S., "Determining Satellite Close Approaches, Part II," *Journal of Astronautical Sciences*, Vol. 42, No. 2, 1994, pp. 143–152.

[10] Hoots, F. R., Crawford, L. L., and Roehrich, R. L., "An Analytic Method to Determine Future Close Approaches Between Satellites," *Celestial Mechanics*, Vol. 33, No. 2, 1984, pp. 143–158. doi:10.1007/BF01234152

[11] Klinkrad, H., "One Year of Conjunction Events of ERS-1 and ERS-2 with Objects of the USSPACECOM Catalog," *Second European Conference on Space Debris*, ESA-SP 393, ESOC, Darmstadt, Germany, March 1997, p. 601.

[12] Healy, L. M., "Close Conjunction Detection on Parallel Computer," *Journal of Guidance, Control, and Dynamics*, Vol. 18, No. 4, 1995, pp. 824–829. doi:10.2514/3.21465

[13] Alarcón Rodríguez, J., Martínez Fadrique, F., and Klinkrad, H., "Collision Risk Assessment with a Smart Sieve Method," *Joint ESA-NASA Space-Flight Safety Conference*, European Space Agency ESA SP-486, 2002, p. 159.

[14] Khutorovsky, Z. N., Boikov, V. F., and Kamensky, S. Y., "Direct Method for the Analysis of Collision Probability of Artificial Space Objects in LEO- Techniques, Results and Applications," *Proceedings of the First European Conference on Space Debris*, ESA-SD-01, 1993, pp. 491–508.

[15] Woodburn, J., Coppola, V., and Stoner, F., "A Description of Filters for Minimizing the Time Required for Orbital Conjunction Computations," *Advances in the Astronautical Sciences*, Vol. 135, No. 2, 2009, pp. 1157–1173.

[16] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, 4th ed., Springer, Microcosm Press, Hawthorne, CA, 2013, pp. 919–936.

[17] Gurfil, P., Herscovitz, J., and Pariente, M., "The SAMSON Project–Cluster Flight and Geolocation with Three Autonomous Nano-Satellites," *Proceedings of the 26th Annual AIAA/USU Conference Small Satellites*, Paper SSC12-VII-2, Utah State Univ., Logan, UT, Aug. 2012.

[18] Queipo, N. V., Haftka, R. T., Shyy, W., Goel, T., Vaidyanathan, R., and Tucker, P. K., "Surrogate-Based Analysis and Optimization," *Progress in Aerospace Sciences*, Vol. 41, No. 1, 2005, pp. 1–28. doi:10.1016/j.paerosci.2005.02.001

[19] Sacks, J., Welch, W. J., Mitchell, T. J., and Wynn, H. P., "Design and Analysis of Computer Experiments," *Statistical Science*, 1989, Vol. 4, No. 4, pp. 409–423. doi:10.1214/ss/1177012413

[20] Draper, N. R., Smith, H., and Pownell, E., *Applied Regression Analysis*, Vol. 3, Wiley, New York, 1966, pp. 135–148. doi:10.1002/9781118625590

[21] Klinkrad, H., *Space Debris: Models and Risk Analysis*, Springer, Praxis, Chichester, England, U.K., 2006, pp. 70–75. doi:10.1002/9780470686652.eae325

[22] Dreesen, P., Batselier, K., and De Moor, B., "Back to the Roots: Polynomial System Solving, Linear Algebra, Systems Theory," *Proceedings of the 16th IFAC Symposium on System Identification*

(SYSID 2012), Elsevier BV, July 2012, pp. 1203–1208.
doi:10.3182/20120711-3-BE-2027.00217

[23] Regis, R. G., and Shoemaker, C. A., "A Stochastic Radial Basis Function Method for the Global Optimization of Expensive Functions," *INFORMS Journal on Computing*, Vol. 19, No. 4, 2007, pp. 497–509.
doi:10.1287/ijoc.1060.0182

[24] Lawson, C. L., and Hanson, R. J., *Solving Least Squares Problems*, Vol. 161, Classics in Applied Mathematics, Soc. for Industrial and Applied Mathematics, Philadelphia, PA, 1974, pp. 207–232.
doi:10.1137/1.9781611971217

[25] Danzmann, K., "LISA Mission Overview," *Advances in Space Research*, Vol. 25, No. 6, 2000, pp. 1129–1136.
doi:10.1016/S0273-1177(99)00973-4

[26] Hesse, M., et al., "Theory and Modeling for the Magnetospheric Multiscale Mission," *Space Science Reviews*, Vol. 199, No. 1, 2016, pp. 1–54.
doi:10.1007/s11214-014-0078-y

[27] Gill, E., Montenbruck, O., and D'Amico, S., "Autonomous Formation Flying for the PRISMA Mission," *Journal of Spacecraft and Rockets*, Vol. 44, No. 3, 2007, pp. 671–681.
doi:10.2514/1.23015

[28] Frémeaux, C., "Collision Avoidance: From General Guidelines to Individual Realities," *66th International Astronautical Congress*, International Astronautical Federation (IAF), Paris, France, 2015.

[29] Vallado, D. A., and Crawford, P., "SGP4 Orbit Determination," *Proceedings of AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, AIAA, Reston, VA, 2008, pp. 18–21.
doi:10.2514/6.2008-6770/