Validation Methods for Industrial Machine Learning Problems

Trabajo Fin de Grado

Grado en Ingeniería Aeroespacial

Escuela Técnica Superior de Ingeniería
Aeronáutica y del Espacio

Universidad Politécnica de Madrid

Autor: Pablo Yeste Blesa

Tutor académico: Javier de Vicente Buendía

Marzo 2024

Resumen.

Las matrices de transición de estado constituyen una herramienta muy útil para el análisis de movimiento relativo entre vehículos espaciales. El hecho de ser formas lineales y además de soluciones semianalíticas se plasma en las siguientes ventajas:

a. Dan lugar a un coste computacional razonable, sin comprometer en exceso la precisión numérica.

b. Su carácter analítico permite extraer conclusiones interesantes sobre el funcionamiento del sistema.

Los modelos linealizados generalmente permiten obtener mejores resultados mediante una mejor modelización del sistema. En el caso de la dinámica espacial relativa, el análisis y la implementación de perturbaciones son vitales, al ser la parte normalmente despreciada y más difícil de modelar del problema. Uno de los efectos más comúnmente analizados en el caso de órbitas terrestres es el campo gravitatorio no esférico, debido esencialmente a ser el más relevante en magnitud en órbitas bajas. La expansión del potencial gravitatorio en armónicos esféricos permite obtener una aproximación ciertamente precisa del mismo, aunque, en función de cómo se aborde, su manejo analítico puede resultar altamente complejo.

Este trabajo trata de proporcionar una visión general sobre la dinámica relativa linealizada entre vehículos espaciales, analizando modelos progresivamente más complejos y precisos, así como introduciendo teorías de perturbaciones – cuyo uso último es la construcción de modelos para dinámica relativa. El concepto de seguridad orbital también es abordado, debido a su vital importancia en el movimiento relativo en cercanía (p.ej., vuelo en formación, *rendez-vous*...).

Abstract.

State Transition Matrices are a very useful and widely used tool for the analysis of linearised spacecraft relative motion. Their closed-form and linear characters bear two main advantages:

a. A quite good computational cost is achieved, without compromising accuracy more than desired.

b. Highly insightful knowledge about the dynamics of the system is obtained.

Linearised models return better outcomes the truer-to-life they are. With this in mind, the analysis and implementation of perturbations play an important role, as it is one of the main ways to achieve high accuracy. Earth's non-spherical gravity field is by far the most usually analysed perturbation in Earth orbits, as it has generally the biggest effect (in value). The way in which this perturbation is approached allows for a better or worse reliability of the results.

This thesis intends to provide an outlook on linearised relative motion, evaluating increasingly more complex motion models, and also analysing perturbation theories, whose ulterior use is relative motion. Orbit safety is also recurrently tackled, due to the vital role it plays in close-proximity relative motion.

Contents

Nomenclature

Acronyms

BF	Body-Fixed frame
CR3BP	Circular Restricted Three Body Problem
CW	Clohessy-Wiltshire
ECEF	Earth-Centered Earth-Fixed frame
ECI	Earth-Centered Inertial frame
GA	Gim-Alfriend
HCW	Hill/Clohessy-Wiltshire
Hi-Fi	High-Fidelity
ICs	Initial Conditions
IP	In-plane
IVP	Initial Value Problem
KOE	Keplerian OEs vector
LEO	Low Earth Orbit
LVLH	Local-Vertical, Local-Horizontal frame
MOD	Mean-of-Date frame
NSG	Non-spherical gravity
ODE	Ordinary Differential Equation
OEs	Orbital Elements
OOP	Out-of-plane
PBF	Pseudo-Body-Fixed frame
RTN	Radial-Transverse-Normal frame
SRP	Solar Radiation Pressure
STM	State Transition Matrix
TOD	True-of-Date frame
YA	Yamanaka-Ankersen

General

μ	Gravitational parameter of a celestial body	$\left[\frac{m^3}{s^2}\right]$
$\underline{a}_i$	Acceleration on body i	$\left[\frac{m}{s^2}\right]$
$\underline{F}_i$	Force on body i	$[N]$
a_e	Equatorial radius of a celestial body	$[m]$
m_i	Mass of body i	$[kg]$

Mathematics

$\Delta\bullet$	Variation or increment of a quantity
$\delta\bullet$	Relative quantity (deputy's minus chief's value)
Λ	Real or complex eigenvalue matrix
$\mathbb{I}$	Identity matrix
$\Phi(t, t_0)$	State Transition Matrix
J	Jordan form
V	Real or complex eigenvector matrix
$\boldsymbol{A}$	System's matrix of a system of ODEs
$\underline{f}$	Dynamics function of a system
$\underline{u}, \underline{y}$	State vector of a system of ODEs

Orbital elements

λ	Mean argument of latitude	$[rad]$
Ω	Right Ascension of Ascending Node	$[rad]$, [deg]
ω	Argument of periapsis	$[rad]$, [deg]
θ	True anomaly	$[rad]$, [deg]
a	Semimajor axis	$[m]$
E	Eccentric anomaly	$[rad]$
e	Eccentricity	$[---]$
F	Eccentric argument of latitude	$[rad]$
i	Inclination	$[rad]$, [deg]
M	Mean anomaly	$[rad]$

u	True argument of latitude	$[rad]$, [deg]

Perturbations

$\overline{OE}$	Mean orbital element vector	
$\underline{OE}$	Osculating orbital element vector	
ε	Small parameter	$[---]$
H	Hamiltonian of the system	
K	Modified Hamiltonian of the system	
p_i, P_i	Generalized momentum (original or transformed)	
q_i, Q_i	Generalized coordinate (original or transformed)	
R	Disturbing function	?
S, W	Generating function of the transformation	

Subindices

$\bullet_0$	Variable at $t = t_0$	
$\bullet_C$	Variable from the chief spacecraft	
$\bullet_D$	Variable from the deputy spacecraft	
$\vert_i$	Expressed in frame i	

Useful expressions

$\eta = \sqrt{1 - e^2}$		$[---]$
ϕ, λ	Geodetic latitude and longitude	$[rad]$, [deg]
$\rho = 1 + e\cos\theta$		$[---]$
h	Orbit altitude or orbit momentum	$[m]$ or $\left[\frac{m^2}{s}\right]$

List of Figures

List of Tables

Introduction.

Spacecraft relative motion is an essential part of distributed space systems analysis. It focuses on the dynamics of two or more vehicles, whose relative position (and generally attitude) should be controlled or at least monitored. Rendez-vous operations, formation flying and satellite constellations are three examples of relative motion, all of them being present in contemporary and past missions. Its relevance cannot be underestimated, and its complexity leads to the currently non-existent quorum on its modelling. However, it is quite widely accepted that linear models using State Transition Matrices (*i.e.* STMs onwards) provides a comforting balance between accuracy and computational cost. One could think that, given the huge computational resources that most people have access to, computational efficiency should not even be a concern. Be that as it may, satellites on-board computers have limited resources, both computational and energy-related. Any reduction in any activity is greatly welcomed, and that naturally includes the GNC computation. That is the reason why this thesis focuses on the review, analysis and implementation of STM models of relative motion.

A bottom-up approach to this topic is herewith followed, starting from the simplest relative motion model (HCW), building up complexity with elliptic models (YA) and finally reaching state-of-the-art perturbed models. With respect to perturbations, non-spherical gravity arises as the main concern, due to its prevailing effect over the other sources - third bodies and drag among others - for the Earth-centered orbits (excluding very low LEO orbits or very high orbits, like GEO).

Chapter 1 provides a brief description of the roots of this thesis. Relative motion and its main branches are firstly described, to then continue with STMs and its applications.

Chapter ?? starts with the development of the equations of relative motion, eventually reaching Hill equations, which model the linearised equations for circular unperturbed reference orbits. Several approaches to its solution are detailed, performing then a small trade-off analysis between them. Finally, orbit safety is presented, to be then applied to the circular, unperturbed case.

Chapter ?? looks for extending the analysis to elliptic (still unperturbed) reference orbits.

The state-of-the-art Yamanaka-Ankersen model is developed and tested against a High-Fidelity propagator. To end this chapter, the safe orbit theory is also extended to this elliptic case, through the usage of an analogue set of relative orbital elements and a slightly different relative reference frame.

Chapter ?? tackles the analysis of propagation theory. A brief analysis of averaging methods is discussed, including Von Zeipel's and Lie series' approach. Afterwards, the spotlight is turned on the non-spherical gravity field modelling. The classical Brouwer's theory is first briefed, to then move on to probably the main topic of the thesis: Kaula's theory, which tries to develop a general approach to the spherical harmonic description of the gravity field.

After this theoretical analysis, **chapter ??** moves back to the relative motion field, considering perturbations in this case. An implementation of Brouwer's theory is drafted through Gim-Alfriend STM, after which a survey of the state of the art of perturbed relative motion is presented. To conclude, perturbed orbit safety is simply sketched out.

The pure content of the thesis is then followed by a set of appendices to provide some background in certain relevant topics, if that is required or desired by the reader.

Appendix A provides a comprehensive review and description of the main orbital element sets.

In a somewhat parallel fashion, **appendix ??** focuses on the reference systems used or mentioned throughout the thesis, as well as the transformations between them and from/to orbital elements.

The variational formulation of the spacecraft equations of motion is introduced in **appendix ??**, providing context for some subjects of perturbation theory.

Appendix ?? is devoted to the software approach and structure of the developed content during the thesis.

Finally, **appendix ??** contains several relevant topics that are not big enough to be included in an individual appendix, and do not fully fit into the existing ones.

The goal of this thesis is then to provide an educated outlook on the linearised formulation of spacecraft relative motion, building up complexity as it unfolds, eventually reaching state-of-the-art models which are actually used in operational missions. Due to the structure of this field, this finds a niche in the manipulation and understanding of perturbations; as a result of which a considerable part of the thesis is devoted to it. The validation and trade-off evaluation of said models is naturally an essential part of this work as well. However, this thesis

is not a result-based work: it is rather a knowledge, analysis-based project, which should be kept in mind along its reading.

Chapter 1

The Validation Loop

1.1 Introduction

1.1.1 Case of study

For illustration of the proposed validation loop, a case of study has been used alongside the theoretical rationale of the loop to generate explanatory figures and tables, demonstrating the loop's effectiveness in real operations.

The case of study is an application of neural networks (NN) in aircraft stress engineering. For typical aircraft fuselage panel design, the dominant form of stiffened post buckling failure under shear loading is forced crippling[1]. This occurs when the shear buckles in the panel skin force the attached stiffener flanges to deform out-of-plane. There are other failure modes related to buckling, tension, and compression.

The aim of the NN model is to predict Reserve Factors (RF) that quantify failure likelihood for regions of the aircraft subject to different loads happening in flight maneuvers (vid. Figure 1.1). The possible failure modes are Forced Crippling, Column Buckling, In Plane, Net Tension, Pure Compression, and Shear Panel Failure.

Input data consists of 26 features (loads applied to a specific region of the aircraft and different maneuver specs). Output data consists of 6 reserve factors that quantify the stress failure likelihood (RF_1, RF_2, RF_3, RF_4, RF_5, and RF_6), where $RF_i \in [0, 5]$,where 0 means extreme risk and 5 means risk extremely low (in a logarithmic scale).

This study prioritizes assessing the efficacy of the developed validation tool, not optimizing the neural network model itself. Therefore, the specific architecture and additional features of the NN are intentionally treated as a black box. Our focus remains solely on its inputs –the 26 aforementioned features– and its outputs –the 6 reserve factors representing 6 distinct failure modes–. This simplification allows us to isolate and evaluate the performance of the validation tool without introducing confounding variables related to the specific neural network design.

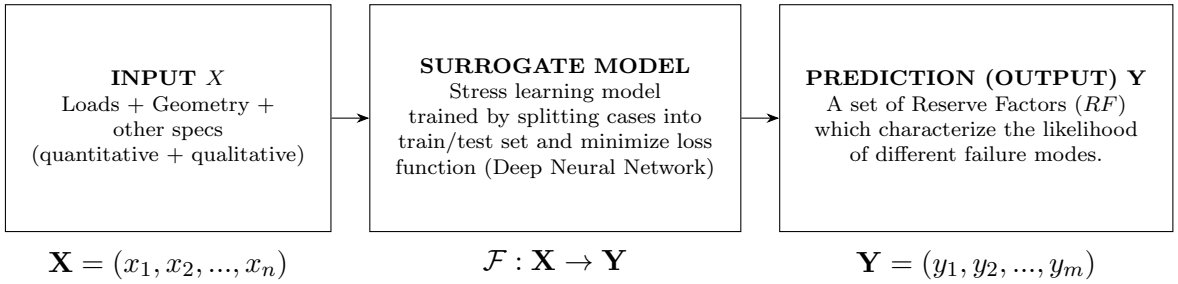


Figure 1.1: Surrogate model pipeline

1.1.2 Applicability

The applicability region can be intuitively defined as the set of the input and output values for which the model's prediction can be trusted. Applicability region is a refined idea of the *Operational Design Domain* of a model.

For a model F to be applicable to a certain new point x:

- x needs to be inside the *input* applicability region of F.
- The prediction $Y = F(x)$ needs to be inside the *output* applicability region of F.

Knowing whether a new point is either inside or outside applicability is a binary decision problem which can be addressed *a priori* (filtering) or be learned *a posteriori* (model boosting). In other words, the binary classification can be addressed from an unsupervised (geometric) learning approach, or a supervised learning approach:

A. Supervised applicability classifyers

Supervised applicability classifiers are based on exogenous criteria. Examples of them include:

- **Operational design domains.** Based on engineer/scientist expertise, only certain combinations and range of features make physical sense. These combinations are used to define a (geometrical) applicability region.
- **Error-filtered convex hull.** Initially the whole set under testing is in applicability range. After filtering those test cases whose prediction error exceeds a tolerance, the applicability region is defined as the convex hull of the remaining subset. The convex hull can be simply defined as the smallest convex set containing the data[2] (vid. Figure 1.3).

- **Error-labelled classifiers.** Initially a whole subset of the dataset (namely, the test set) is in applicability range. After labelling those test cases whose prediction error exceeds a tolerance as outside applicability, different binary classifiers are trained on the full dataset.

B. Unsupervised applicability classifyers

On the other hand, geometrical applicability classifiers are based on the premise of NN interpolation capabilities. It has been widely discussed (see *e.g.* [3–5]) that NN's performance relies on their interpolation capabilities, and that no extrapolation capabilities outside their applicability region should be assumed.

Subsequently, from a geometrical approach to the applicability classification problem, we shall define the input applicability region as the geometrical region in the input space where the model is known to work in interpolating regime.

There are plenty of definitions for the interpolating region of a given dataset. Some authors define it as the smallest hypercube enclosing the data[6] (vid. Figure 1.2) although this may seriously challenge interpolation due to isolated points falling into the interpolation region under this definition. Many authors (see v. gr. [7, 8]) define the interpolation region as the *convex hull* of the training data, *i.e.* :

Definition 1. *[9] Standard interpolation occurs for a sample* $\mathbf{x}$ *whenever this sample belongs to the convex hull of a set of samples* $\mathbf{X} \triangleq \{\mathbf{x}_1, ..., \mathbf{x}_N\}$.

In the above definition, $\mathbf{X}$ is the training set which has been used for training the model.

Recently, [9] have pointed out that, due to the so-called curse of dimensionality, extrapolation outside the training convex hull always ends up taking place if the input dimension is sufficiently large. The curse of dimensionality[10, pp. 17-18] can be illustrated by the fact that the unitary n-sphere's volume asymptotically diminishes to 0 as n increases. In the end, the effect is that, as the number of dimensions increase, more data is needed in order for the model to work in interpolating regime, as is showed by the following theorem:

Theorem 1.1.1. *[11] Given a d-dimensional dataset* $\mathbf{X} \triangleq \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ *with i.i.d. samples uniformly drawn from a hyperball, the probability that a new sample* $\mathbf{x}$ *is in interpolation regime (recall 1) has the following asymptotic behaviour:*

$$\lim_{d\to\infty} p(\mathbf{x} \in Hull(\mathbf{X})) = \begin{cases} 1 \iff N > d^{-1}2^{\frac{d}{2}} \\ 0 \iff N < d^{-1}2^{\frac{d}{2}} \end{cases} \tag{1.1}$$

The main answer to the argument of [9] is that interpolation does not occur in the ambient space, but in a latent, low-dimensional space of the input data[12]. This leads to a new definition of interpolation alternative to 1.

Interestingly, the authors in [12] go beyond demonstrating that interpolation occurs (at least in a latent representation of the ambient space). They unveil two more crucial conditions for optimal model performance:

- The training and testing data should share the same cumulative distribution. In situations where this is impossible, matching the tail distribution becomes essential.
- The testing points should not be isolated within the dataset. This means they should have similar characteristics to other data points and not represent extreme outliers that testing and training data follow the same cumulative distribution (or the same tail distribution whenever the former cannot be fulfilled) and that testing points be not isolated in the dataset.

Examples of unsupervised applicability classifiers include:

- **Input space range classifier.** The applicability region is defined as the boundary of the hypercube whose edges are the ranges $[min, max]$ of each input variable in the training set. In practice, this amounts to checking if each numerical feature X_i of the test point lies inside the range spanned by the training subset.
- **Input space convex hull classifier.** The applicability region is defined as the boundary of the convex hull of the training set.
- **Input space isolated region detector.** Even if a point is inside the convex hull of the training set, it can be very far away from other points, thus interpolation can be challenged. Isolated region detectors address this by checking the statistical vicinity of train points and comparing it to the distance of a given test point to the closest training point.

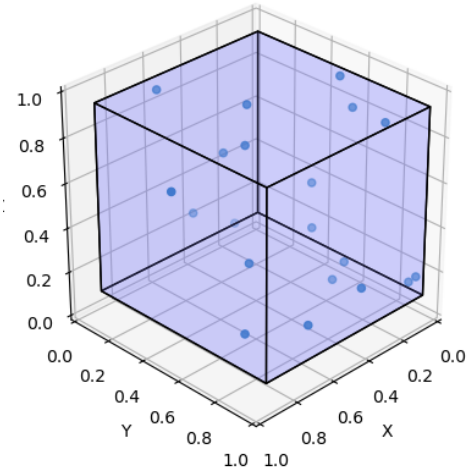


Figure 1.2: Smallest cube enclosing a set of 20 randomly generated points in a 3-dimensional euclidean space.

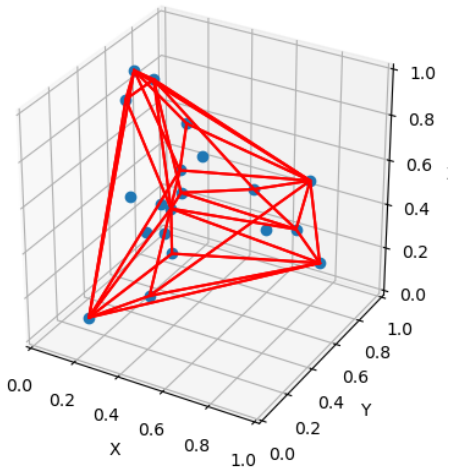


Figure 1.3: Convex hull enclosing the set of Figure 1.2. The convex hull's volume is always smaller than or equal to the volume of the hypercube.

1.1.3 Overview

(Sin acentos) Recorrido sumario por las distintas partes del analisis estadistico (plantillas) centrandose en su funcion en vez de su funcionamiento tecnico. Explicacion de la arquitectura general de la validacion.

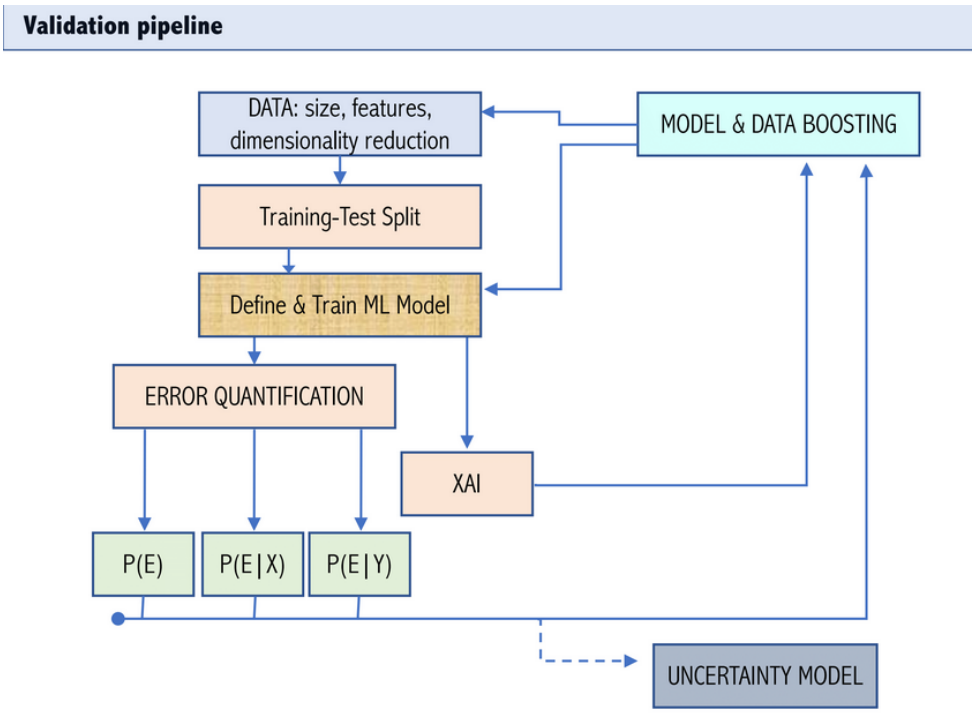


Figure 1.4: Validation pipeline

1.2 Train-test split

1.2.1 Preliminaries

In supervised learning applications, the dataset is typically divided into the training and the testing sets. Keeping different sets for each task is fundamental in order to prevent model bias. Typical figures for the train-test split ratio are 80%-20%. When the model being trained is very large, a third dataset (the validation set) may be needed for comparing different hyperparameter configurations, in which case the split is typically done at 60%-20%-20% for the train, test, and vaidation sets, resectively[10, pp. 20-21].

Training and evaluating the NN on the same dataset would result in the phenomenon called overfitting[10, pp. 19-20], which basically consists of the NN fitting the noise in the training data and thus losing generalization capabilities.

With the dataset split into the train, evaluation, and test sets, the standard training and validation loop is as follows: the model is trained to "fit" the data in the training set. After a certain amount of training, its performance is measured in the evaluation set. The test set is used to compare the performance of different hyperparameter configurations. Note that the NN is always evaluated with data not previously "seen" during training, and as such cannot develop

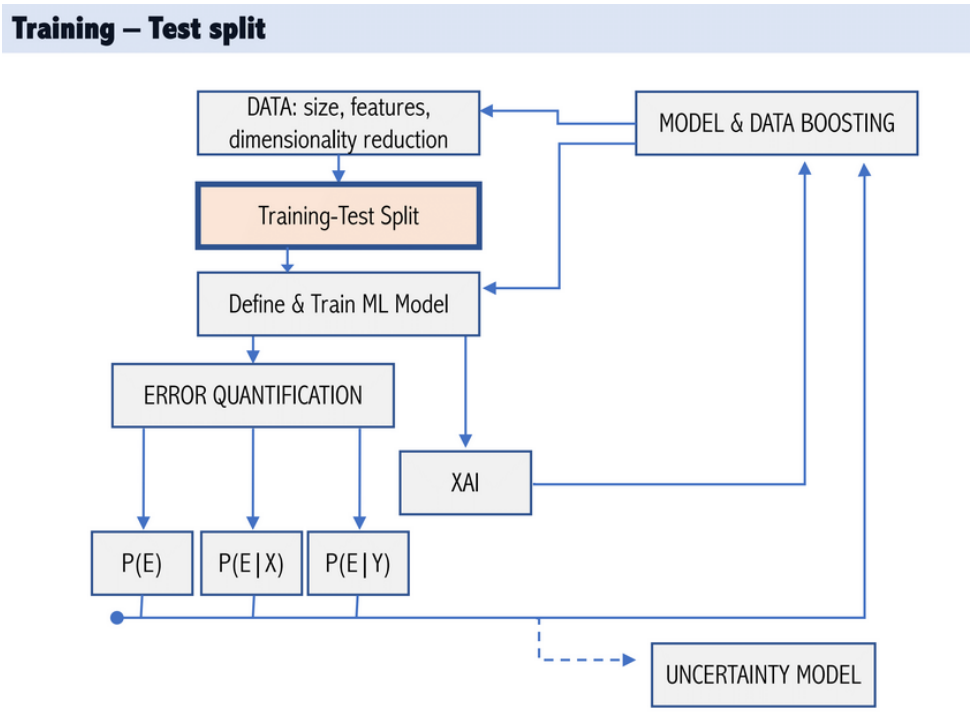


Figure 1.5: Location of the train-test split assessment in the overall validation pipeline

any bias for the training set (although bias for the validation or test sets may exist).

In the model validation loop, the first question that should be asked, even before training the model, is whether the dataset split is appropriate for training (vid Figure 1.5).

For our case of applicability (MSP18, explicar en otra seccion), we suppose we have a dataset which has been split into a training set

$$\mathcal{S}_{\text{train}} = \{(\mathbf{X^{tr}}, \mathbf{Y^{tr}})_i\}_{i=1}^{N_{training}}$$

and a test set

$$\mathcal{S}_{\text{test}} = \{(\mathbf{X^{te}}, \mathbf{Y^{te}})_i\}_{i=1}^{N_{test}},$$

where $\mathbf{X} = (X_1, X_2, \ldots, X_m)$ is the vector of input variables (also called feature vector), m is the dimensionality of the input parameter space, and $\mathbf{Y} = (Y_1, Y_2, \ldots, Y_q)$ is the vector of output variables, with dimensionality q. Individual variables X_k can be numerical or categorical. For instance, for stress problems such as MSP-S18, we have a combination of numerical (*e.g.* external loads) and categorical (eg discrete geometrical variables such as "Frame" or "Stringer", and purely categorical such as "dp"). Individual variables Y_k describe the prediction and these are typically numerical (for stress problems such as MSP-S18, $q = 6$ and these are so-called reserve factors, quantifying the failure likelihood of different failure modes).

The aim of this section is to propose a set of tests to evaluate to which extent splitting fulfils the necessary conditions for the subsequent surrogate model $\mathbf{Y} = F(\mathbf{X})$ to be appropriately trained on $\mathcal{S}_{\text{train}}$ and tested on $\mathcal{S}_{\text{test}}$. This basically means the latter be in the applicability region of a model trained on the former, in order for the training-validation loop of the model to be coherent.

1.2.2 Validation loop for the train-test split

1.2.2.1 Geometrical analysis

To address the applicability classification problem, points in the test set extremely far away from the training set (effectively outside applicability) need to be detected. If found, that would mean the train-test split is incorrect.

There is a wide range of criteria for classifying a point inside or outside applicability. The proposed validation loop makes such analysis following a hierarchical flow. For each point in the test set a decision is made based on where that test set point lies with respect to the training

set.

Because categorical variables are usually related to different physical conditions, the first step is to condition the analysis of each test point only with respect to the training subset of points whose categorical variables have exactly the same values than the test point under analysis. This conditioning on categorical variables defines, for each test point, a *voxel*, *i.e.* a region in the input space defined by the precise values of each categorical variable. At this point the first requirement comes into play: the number of training samples inside such voxel needs to be large enough (*e.g.* larger than zero).

Then, inside the voxel the ranges of each feature X_i of the training subset are computed, and then the maximum ranges are checked and defined. Their cartesian product defines the voxel's hypercuboid volume associated to the numerical features. It is checked whether the test point is inside or outside the hypercube. In practice, this amounts to checking if each numerical feature X_i of the test point lies inside the range spanned by the training subset. If all numerical features fall inside these ranges, the test set falls inside the training subset hypercuboid, otherwise, it falls outside. The points lying outside such hypercuboid are outside applicability and are flagged. The "hypercube requirement" is that such percentage of points outside applicability is zero, or very close to it.

If the test point under analysis is found to be outside the hypercuboid, the analysis finishes and the loop moves to the next point. Otherwise, the second step is to check whether this point not only lies inside hypercuboid, but further, whether it also lies inside the convex hull spanned by the training subset in a PCA99[1] projection. This hull has a smaller volume than the hypercuboid. If the test point under analysis is found to be outside the convex hull in the PCA99 projection, the analysis finishes and the loop moves to the next point.

Otherwise, then the third step is to check whether this point (not only lies inside hypercuboid and $\mathrm{CH}_{\mathrm{PCA99}}$, but also) lies inside the convex hull spanned by the training subset in ambient space. The volume of this hull is typically smaller than the one of the PCA99 projection by virtue of the curse of dimensionality, and if the test point lies inside this hull, interpolating properties of the surrogate model will suggest that the model is not require to generalize outside the region where it cannot generalize.

The step of the PCA projection is motivated by the curse of dimensionality, which makes every test point be in extrapolating regime with respect to the training data when the convex hull of

[1] A truncated PCA[13] projection in which the selected components explain at least 99% of the whole variance of the data.

the raw, unprocessed training data is used in a high-dimensional input space, as pointed out by [9]. The work of [12] effectively shows how in such cases, a low-dimensional space is more useful for applicability classification.

Each step provides different level of evidence of whether the point under analysis was adequately located, where the best is that it lies inside the convex hull of the corresponding voxel in ambient space, and the worst is that it lies outside the hypercuboid.

An additional check is to analyse whether p-hacking is happening, *i.e.* whether the test set point is "too close" to an actual training point (or indeed equal to a training point), what would falsely induce low test error of the model. For this, we measure the distance of the test point to the closest point of the training subset. This information is later used to assess different aspects of potential p-hacking and its impact on the decision of train-test split correctness.

The former procedure for classifying the applicability of test points is summarised in algorithm 1.

Output of algorithm 1 is depicted in Table 1.1. This information has to be processed to complete Table 1.4. The first column identifies the test point inside the test set. The next three columns show information about the corresponding voxel: existance, identification and size. Columns 5 to 7 show information related to pointwise distances inside the voxel. The last three columns show the relative position of the test point inside the voxel (inside the hypercube/convex hull in PCA99/convex hull in ambient space). In this algorithm, if the test point is found to be outside the hypercube (CH_{PCA99}), then its position with respect to CH_{PCA99} and $\text{CH}_{\text{ambient}}$ ($\text{CH}_{\text{ambient}}$) is not computed in the sake of computational efficiency.

To better understand how the voxels are computed, a reference of their categorical variables is given in Table 1.2. Points inside each voxel all have the same categorical values, which effectively act as a unique identifyer for the voxel. We see each voxel represents a different stringer section between two concrete frames, plus the categorical variable "np" which can take the values either 0 or 0.9.

Data in the column "inside vox. hypercube" of Table 1.1 has been rearranged for the sake of interpretability in Table 1.3. This table shows the points which lie outside the training voxel hypercube for every voxel where such points exist. Similar tables could be arranged with the number of points belonging to the hypercube but not to the convex hull in a PCA99 projection, or with those points belonging to the convex hull in a PCA99 projection but not in ambient space.

Algorithm 1: Train-test split geometrical analysis

Data: $\mathcal{S}_{\text{train}}, \mathcal{S}_{\text{test}}$, voxel_size_req
Result: inside_hypercube, inside_PCA99, inside_ambient, avg_train_dist, min_test_dist

Initialize empty lists: test_voxels, train_voxels ← [];
Initialize boolean arrays:
inside_hypercube, inside_PCA99, inside_ambient ← $[\text{False}, \ldots, \text{False}]_{1\times \text{len}(\mathcal{S}_{\text{test}})}$;
Initialize arrays for distances: avg_train_dist, min_test_dist ← $[\text{NaN}, \ldots, \text{NaN}]_{1\times \text{len}(\mathcal{S}_{\text{test}})}$;
foreach $\mathbf{X} = (\mathbf{X}_{numerical}, \mathbf{X}_{categorical})$ *in* $\mathcal{S}_{test}$ **do**
    Compute test_voxel by imposing $\mathbf{X}_{\text{categorical}}$;
    Append test_voxel to test_voxels;
end
foreach $\mathbf{X} = (\mathbf{X}_{numerical}, \mathbf{X}_{categorical})$ *in* $\mathcal{S}_{train}$ **do**
    Compute train_voxel by imposing $\mathbf{X}_{\text{categorical}}$;
    Append train_voxel to train_voxels;
end
$i \leftarrow 0$;
foreach *test_voxel in test_voxels* **do**
    Get train_voxel by imposing $\mathbf{X}_{\text{categorical}}$;
    Compute main nearest neighbour distance inside train_voxel:
    $\overline{d}_{\text{train}}(\text{train\_voxel}, \text{train\_voxel})$;
    avg_train_dist[i] ← $\overline{d}_{\text{train}}$;
    if *size(train_voxel)* $\geq$ *voxel_size_req* **then**
        foreach $\mathbf{X}$ *in test_voxel* **do**
            Compute minimum nearest neighbour distance: $d_{\text{test}}(\mathbf{X}, \text{train\_voxel})$;
            min_test_dist[i] ← d_{test};
            if $\mathbf{X}$ *not in hypercube* **then**
                break;
            else
                inside_hypercube[i] ← True;
                if $\mathbf{X}$ *not in CH_PCA99* **then**
                    break;
                else
                    inside_PCA99[i] ← True;
                    if $\mathbf{X}$ *in ambient_hull* **then**
                        inside_ambient[i] ← True;
                end
            end
            $i \leftarrow i + 1$;
        end
    end
end

Table 1.1: Output of algorithm 1.

	Vox. exists	Vox. ID	Vox. size	Min. test to train vox. dist.	Min. train to train vox. dist.	Avg. train to train vox. dist.	Inside vox. hypercube	Inside vox. CH (PCA99)	Inside vox. CH (ambient)
0	True	729	10	4.2222	2.3103	4.9535	False	NaN	NaN
1	True	703	10	6.8161	3.5068	6.2151	True	False	NaN
2	True	211	10	3.2955	1.6819	4.4763	False	NaN	NaN
3	True	799	8	NaN	NaN	NaN	NaN	NaN	NaN
4	True	531	6	NaN	NaN	NaN	NaN	NaN	NaN
...	...	...	...	...	...	...	...	...	...
1995	True	102	8	NaN	NaN	NaN	NaN	NaN	NaN
1996	True	536	11	3.3157	2.8551	4.4765	True	False	NaN
1997	True	770	6	NaN	NaN	NaN	NaN	NaN	NaN
1998	True	634	10	1.1911	1.8775	5.0745	True	False	NaN
1999	True	277	10	4.8021	3.1150	5.6874	True	False	NaN

2000 rows × 9 columns

Table 1.2: Voxel reference table

	Voxel
Voxel ID	
0	(0.0, Fr68-Fr69, Str40)
1	(0.9, Fr66-Fr67, Str36p)
2	(0.0, Fr69-Fr70, Str29)
3	(0.9, Fr69-Fr70, Str35)
4	(0.0, Fr65-Fr66, Str01)
...	...
827	(0.9, Fr64-Fr65, Str21)
828	(0.9, Fr66-Fr67, Str32)
829	(0.0, Fr64-Fr65, Str26p)
830	(0.0, Fr64-Fr65, Str29p)
831	(0.0, Fr67-Fr68, Str07)

751 rows × 1 columns

This information is used to check the following parameters and requirements, which are stored in Table 1.4:

- Whether there are enough training samples inside each training subset is compared with `voxel_size_req`.
- The percentage of test points inside the ranges of the training samples is compared with `hypercube_req`.
- The number of points inside the hypercube but outside `CH_PCA99` is compared with `CHMP_PCA99_negative_`
- The number of points inside the hypercube and `CH_PCA99` but outside `CH_ambient` is compared to the requirement `CHMP_ambient_negative_req`.
- The total number of points inside `CH_ambient` is compared to `CHMP_ambient_abs_req`.
- All these requirements are stored in Table 1.4.

Having analised the isolation level of test points, it is necessary to check wheter p-hacking is taking place. That is, whether test points are unrealistically close to train points, thus making the model evaluation flawed. This is measured using the Mann-Whitney U test[14]. In this test, the initial hypothesis H0 is that both distributions are the same. H1 is that the distribution underlying test-training distances is stochastically less than distribution underlying training-training distances, which would involve risk of p-hacking. H0 is rejected only with a 95% confidence. For instance, for the dataset of MSP-18, the resulting p-value is $p = 0.9793$, thus the null hypothesis is not rejected and the dataset can be assumed to be free of p-hacking.

1.2.2.2 Non-geometrical analysis

To help (the engineer in charge) make a decission about the dataset split, another approach is developed, complementary to the geometrical analysis carried out so far. This approach consists of focusing on the statistical distributions of the train and the test datasets. A proper train-test split is characterised by similar statistical distributions of both sets (test and training)[12]. When this cannot be achieved, a softer requirement is that each of the feature variables X_i need to have reasonably similar marginal distributions in the training and the test set.

The following analysis checks whether, for each individual input feature, the distribution of the training and test set is reasonably similar. This is also checked for the output variables

Table 1.3: Points outside voxel hypercube

Voxel ID	Voxel	Num. of test points belonging to voxel	Num. of test points outside voxel's hypercube
729	(0.0, 'Fr66-Fr67', 'Str28')	2	2
703	(0.0, 'Fr65-Fr66', 'Str29')	3	2
211	(0.0, 'Fr66-Fr67', 'Str40')	4	3
219	(0.0, 'Fr64-Fr65', 'Str35')	4	3
413	(0.9, 'Fr68-Fr69', 'Str05p')	5	4
...	...	...	...
561	(0.9, 'Fr69-Fr70', 'Str43')	2	1
436	(0.0, 'Fr64-Fr65', 'Str40p')	1	1
276	(0.0, 'Fr64-Fr65', 'Str31')	1	1
109	(0.0, 'Fr69-Fr70', 'Str04')	2	1
446	(0.9, 'Fr69-Fr70', 'Str21')	1	1

104 rows × 3 columns

Table 1.4: `reqs_results_table`. The first and the last requirements compare absolute figures of points at some isolation level with the overall number of points. Whereas the second and third requirements compare the difference in number of points between two consecutive levels with the total number of points.

	Desired rate (equal or less)	Obtained rate
Hypercube rate	0.1000	0.5780
CHMP PCA99 negative rate	0.1000	0.3880
CHMP ambient negative rate	0.1000	0.0340
CHMP ambient abs rate	0.1000	0.3880

(the ground true ones, thus this is independent from the surrogate model). Finally, the splitting can be done locally, *i.e.* region by region, by binning each numerical variable. The implementation of this analysis tackles each input variable X_i independently and for each input variable, it compares the training and the test set such that:

- If the variable X_i is 'categorical': both (train and test) categorical frequency distributions are plotted (Figure 1.7), a 2-sample χ^2 test[15, p. 431] is performed, and the p-value of such test is introduced in Table 1.5.

- If the variable X_i is numerical: both (train and test) numerical frequency distributions are plotted (Figure 1.6), a 2-sample Kolmogorov-Smirnov test[15, p. 454] is carried out, and the p-value of such test is introduced in the same table.

The null hypothesis for both tests is that both train and test distributions are the same. H0 is only rejected with a 95% confidence.

As stated earlier, applicability is not only a matter of the *input* space, but also of the *output* space. The 2-sample Kolmogorov-Smirnov test has been performed again on each of the six output variables. The p-values and their statistical distributions are depicted in Table 1.6 and Figure 1.8, respectively.

1.2.2.3 Train-test split. Conclusions

The proposed validation loop for the train-test split focuses, on the one hand, on the applicability classification problem, which makes use of a geometrical analysis which classifies test points at different levels of isolation. A series of requirements are defined based on the proportions of such isolated points. Plus, p-hacking is checked. The output of this analysis is showed in Figure 1.9.

On the other hand, the distributions of both input and output variables in the training set are checked to be reasonably similar to that of the test set. The main outputs are the p-values shown at Table 1.5 and Table 1.6.

Of course, any given dataset may meet some of the requirements, but not all (as is the case with the MSP-18 dataset). The engineer in charge should evaluate the ensemble output and decide whether it is necessary to redo the split completely or partially, depending on data

Table 1.5: P-values of the input variables. The test performed is a 2-sample Kolmogorov-Smirnov or a 2-sample χ^2 test, depending on the data being numerical (first 19 rows) or categorical ("dp", "Frame" and "Stringer").

p-values	
	p-value
factors	0.97409
FU.0410.14	0.08896
FU.0410.15	0.77249
FU.0410.16	0.53206
FU.0410.24	0.31840
FU.0410.25	0.54828
FU.0410.26	0.58953
FU.0420.14	0.19180
FU.0420.15	0.50018
FU.0420.16	0.58953
FU.0420.24	0.54015
FU.0420.25	0.34948
FU.0420.26	0.74838
FU.0430.14	0.16084
FU.0430.15	0.17195
FU.0430.16	0.51601
FU.0430.24	0.55646
FU.0430.25	0.42458
FU.0430.26	0.66518
dp	0.99969
Frame	1.00000
Stringer	1.00000

availability.

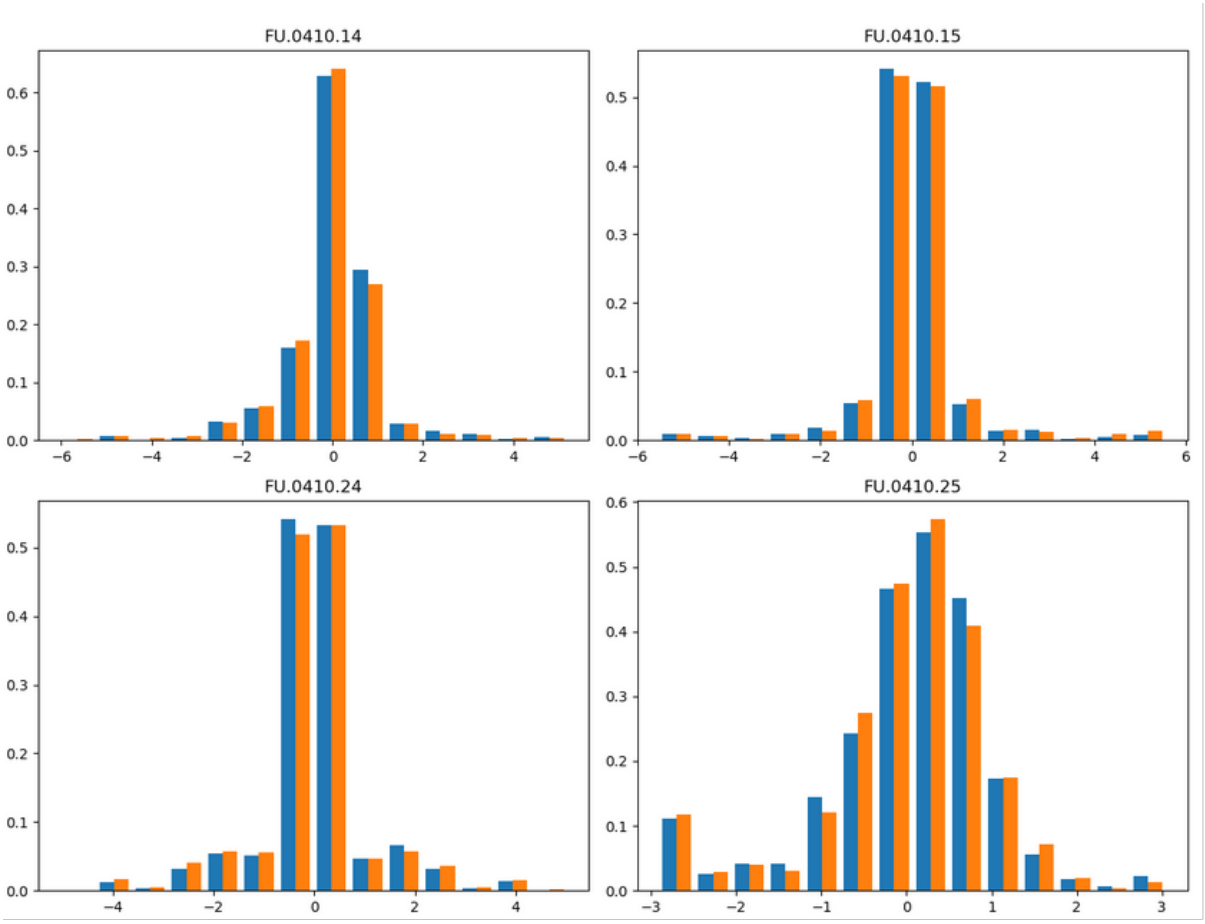


Figure 1.6: Input [numeric] variables distributions double histogram. Only the first four input variables have been plotted.

Table 1.6: P-values of the output variables distributions

p-values	KS
RF Forced Crippling	1.00000
RF Column Buckling	1.00000
RF In Plane	1.00000
RF Net Tension	0.24668
RF Pure Compression	1.00000
RF Shear Panel Failure	0.99995

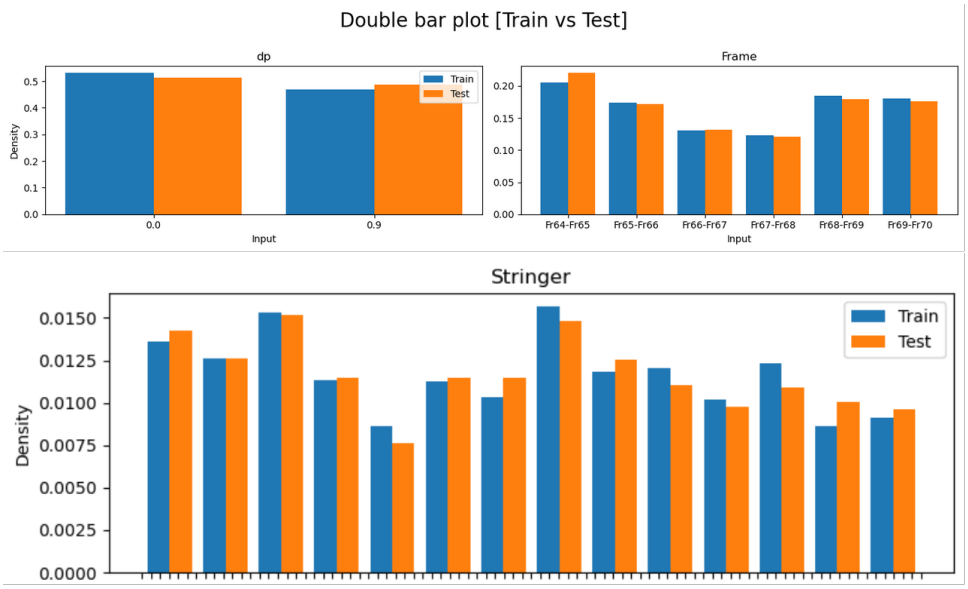


Figure 1.7: Input categorical variables distributions double histograms

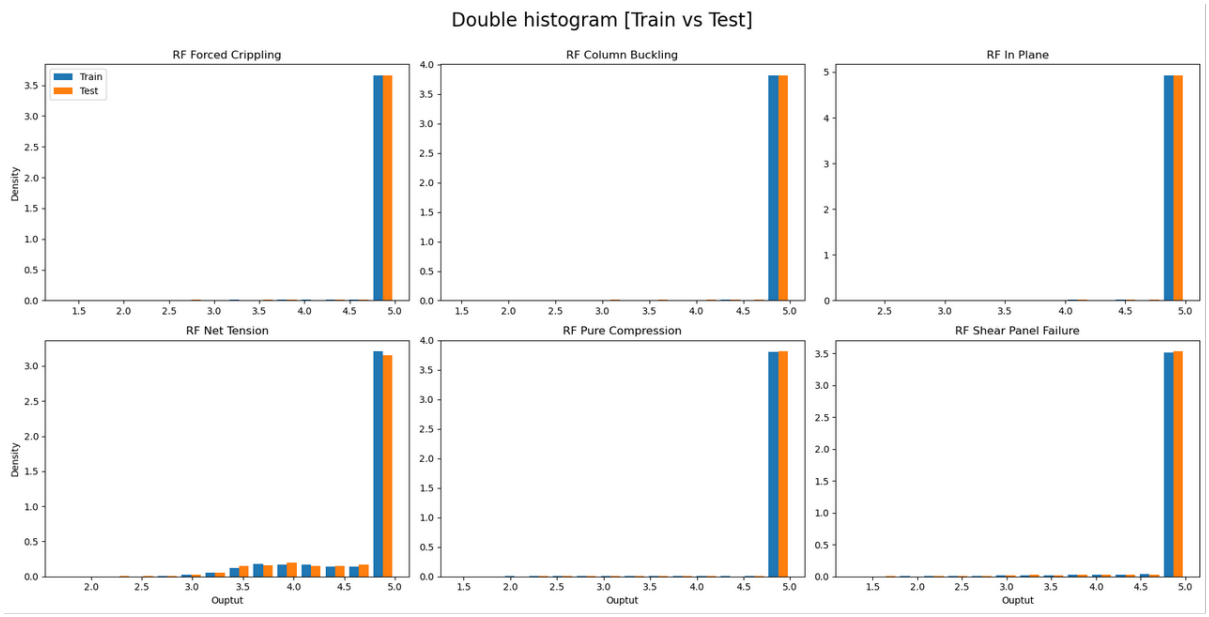


Figure 1.8: Output variables distributions double histograms

Requirement check

	Type	Check
Variable Compatibility Check	Optional	True
Voxel Size	Optional	None
Voxel Hypercube Rate	Optional	False
CHMP PCA99 (negative)	Optional	False
CHMP ambient (negative)	Optional	True
Mann Whitney U test pvalue	Optional	True

Figure 1.9: Requirements of the geometrical analysis

1.3 Global error quantification

This section corresponds to box labelled "Error quantification" in Figure 1.4.

After running the model, the simplest analysis of its performance consists of measuring the aggregated error of predictions against ground true values over the whole test set. Different metrics and criteria can be adopted for this task. Common error measures are the Mean Absolute Error (MAE) and the Root Mean Square Error (RMSE). These metrics account for the distance between points representing ground true values ($\mathbf{Y}$) and model's predictions ($\tilde{\mathbf{Y}}$), taken as the $L1$ norm (MAE) or the square of the $L2$ norm (RMSE). In many industrial applications of machine learning, difference between ground true and predicted values can be more or less important depending on whether that difference is due to *overestimating* or *underestimating*. Take, for instance, the case of MS-S18 model, whose predictions are a measure of the probability of failure of aeronautical structural components. Clearly, underestimating the risk is much more dangerous than overestimating it. For cases such as this one, a useful measure of the error is the **residue**. The residual error of a given point i is measured as

$$\mathbf{e}(i) = \mathbf{Y}(i) - \tilde{\mathbf{Y}}(i) \tag{1.2}$$

The drawback of the residual error is that, when using it as an aggregated indicator for the whole test set, residues can cancel out. A null MAE or RMSE account for a perfectly fitted model (ground true values and predictions are equal). That is not the case for the residual error.

An interesting scalar metric for the global performance of the model is the coefficient of determination R^2 [16] of the scatter distribution of $\mathbf{Y}$ vs $\tilde{\mathbf{Y}}$. R^2 is a measure of the goodness of fit of predicted to ground true values. Illustration of this is provided in Figure 1.10.

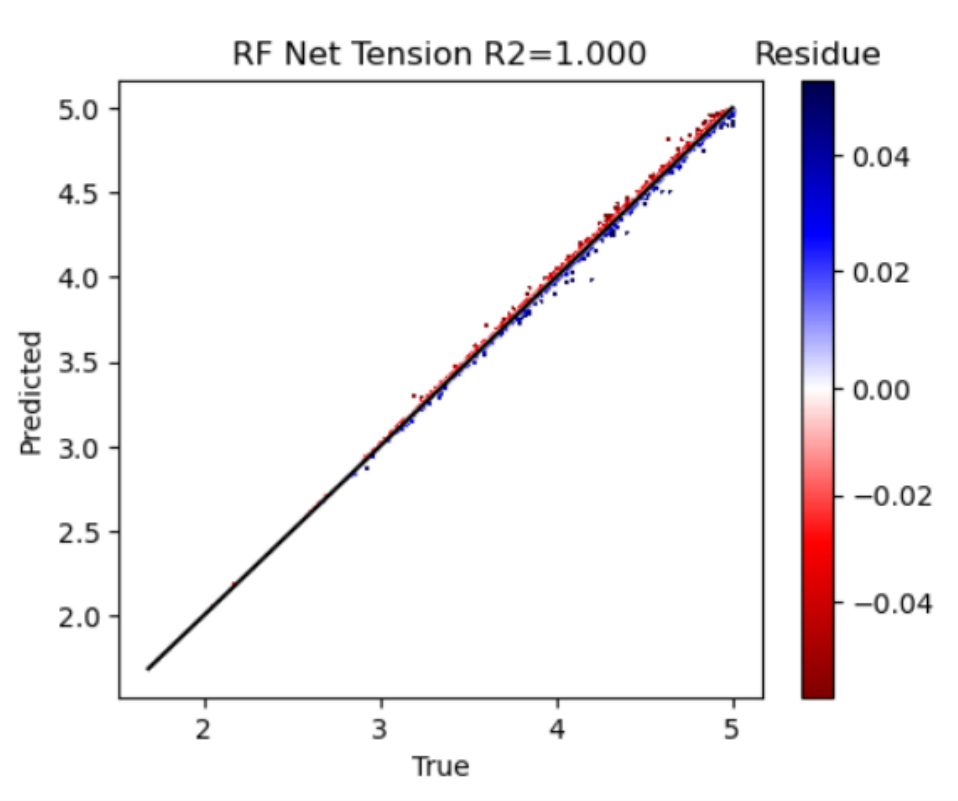


Figure 1.10: Scatter plot of ground true (x axis) against predicted values (y axis) and R^2 coefficient. In this case, $R^2 = 1,000$ indicates a perfect fit of predicted to ground true values. Mismatches due to underestimating failure risk are labelled in red, while those due to overestimating failure risk are labelled in blue. Similar graphs can be computed for every pair $\{y_j, \hat{y}_j\}$ of features in the output variables $\mathbf{Y} = \{y_1, y_2, \ldots, y_m\}$.

1.4 Distributed error quantification.

This section corresponds to box labelled "$P(E)$" in Figure 1.11. From now on, the terms "error" and "residue" (as defined in the previous section) will be used as synonyms.

Beyond global error statistics, the PDF of the residue, $P(e)$, is of great importance for

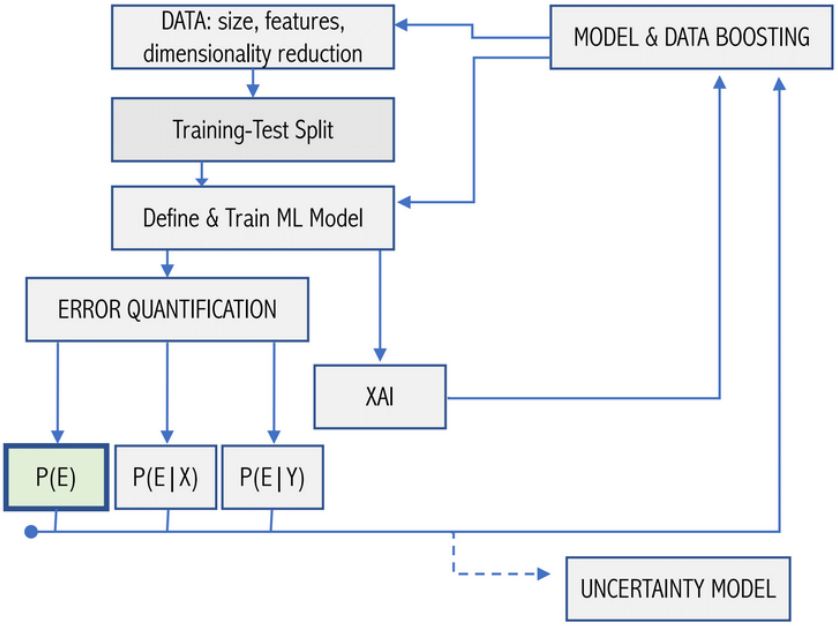


Figure 1.11: Location of section 1.4 in the validation pipeline.

validation purposes. The main goal of this section is finding the analytic expression of $P(e)$ and computing important statistics of it, and, when the first is not possible, computing the important statistics from the empirical distribution of the residue (sampled from the test set) with the method of bootstrapping[21]. There are three main reasons for studying $P(e)$:

A. Non gaussian error distribution

It is common in industry-applied problems to find situations in which the error between ground-true values and predictions coming from a regression model (let it be a surrogate ANN, some parametric regressor, etc.) which is supposed to fit any function describing a complex system follows absolutely non-Gaussian distributions (see *e.g.* [17–20]). Amongst the reasons for this, the most frequent are, on the one hand, non-homogeneous data sampling in the training set (leading to uncovered regions and isolated points) which can cause poor model performance due

to non-interpolation-regime operation, and on the other hand, the inherent difficulty encountered at predicting outputs for specific input configurations due to strongly non-linear physics or governing equations (mathematically this is manifests in high gradients). When the error is non-Gaussian, concentration statistics such as MAE or RMSE stop being informative. In such case, a comprehensive analysis on $P(e)$ is more adequate.

B. Outlier detection

Outliers are strange events in a population sampled from a known PDF, in the sense that it is not expected to find them, or that their position is far away from expected. Outlier detection helps identifying strange phenomena which the engineer in charge could decide to investigate. Imagine, for instance, that certain residue e_x was systematically sampled from the test set with an unusual frequency, compared with similar values. In this case, it would be necessary to assess whether this high frequency is statistically expectable from the residue's PDF or not. If e_x was found to be an outlier, determining the underlying reason triggering such high frequency would help with model boosting.

Outlier detection relies on knowing $P(e)$. The probability of finding a residue larger than a given magnitude x is measured as $p_{>x} = \int_x^\infty P(e)\, de$. If we find some x for which $p_{>x} << 1$, all samples of the residue $e > x$ would be classified as outliers.

C. Uncertainty measuring

The marginalised distribution of $P(e)$ is the first step in the journey towards building an **uncertainty model**. This is the ultimate milestone of the whole validation pipeline, since it provides precise information about *how much* and *when* the model's predictions are trustworthy. This is the whole point of section 1.7. The uncertainty model relies on the marginalised distribution of the residue for building confidence intervals which embed the model's error with a given statistical confidence level (coverage).

A simple plot can help us have a first intuition for the cause of subsequent results presented in this section. In Figure 1.12 ground true values and model's predictions are represented in a double histogram for MS-18.

If the model accurately predicts results along the whole output variables range (or equivalently it is not heteroscedastic [23, p. 374]), we would expect both the true and the predicted results to come from the same (unknown) distribution ($H0$). We can test this with a (2 sample)

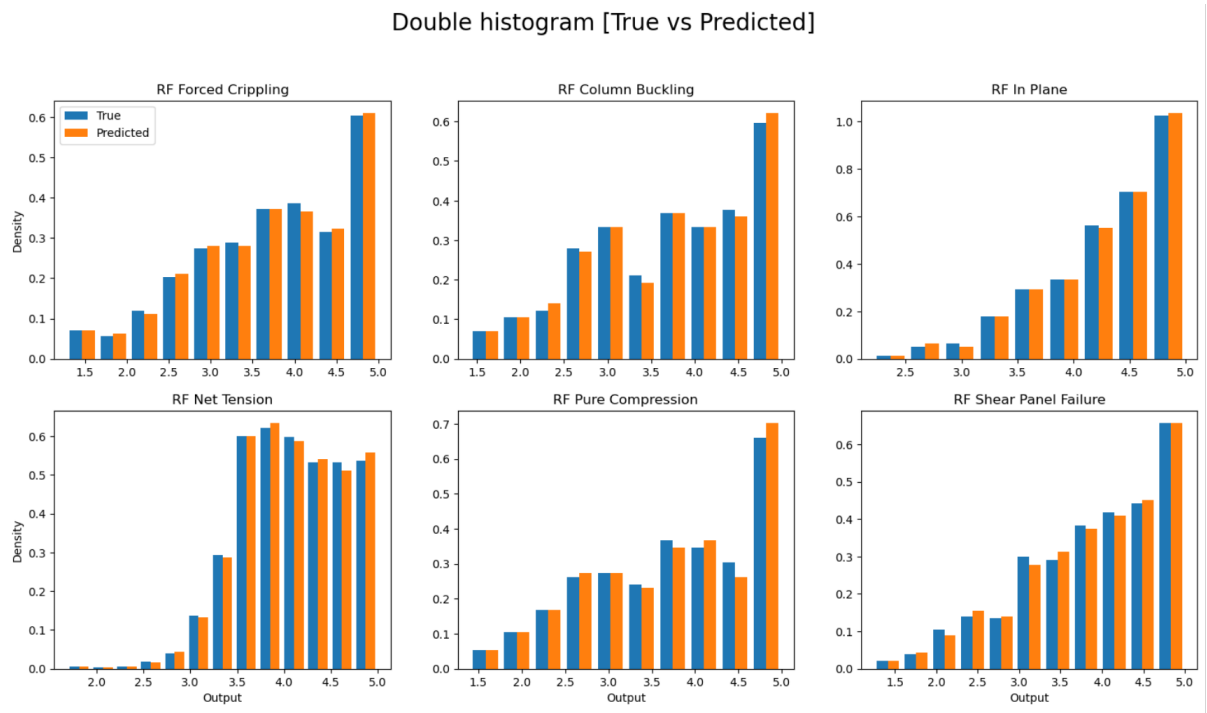


Figure 1.12: Double histogram depicting ground true values and model's predictions for the six output variables of MS-18.

goodness-of-fit test like the Kolmogorov-Smirnov. Once again, the null hypothesis is rejected only at a 95% confidence level (*i.e.* if the p-value derived from the K-S test is lower than 0.05). The corresponding p-values of the K-S test for the six output variables are depicted in Table 1.7.

In Table 1.7 we can see how the K-S test rejects the null hypothesis in some cases despite distributions in Figure 1.12 looking very similar. This is due to the K-S test sensitivity to the size of datasets.

As it has been previously mentioned, outlier detection and uncertainty models both rely on the PDF of the residue, $P(e)$. The main goal of this section is finding the analytical definition of $P(e)$ if possible, and measuring important statistics of it. This is addressed with a focus similar to that followed in Figure 1.12 and Table 1.7, but instead of assessing the fitness of the predictions ($\hat{y}$) distribution to the ground true values (y) distribution, we try to assess the goodness of fit of the empirical residue distribution to some well-known distributions. For reference, the (empirical) error distribution of the six output variables is depicted in Figure 1.13, as well as the corresponding cumulative distributions, given in Figure 1.14.

Under the assumption that $P(e)$ can be described by some well-known parametric distribution, we use the K-S test to compare distributions of Figure 1.13 to the following distributions:

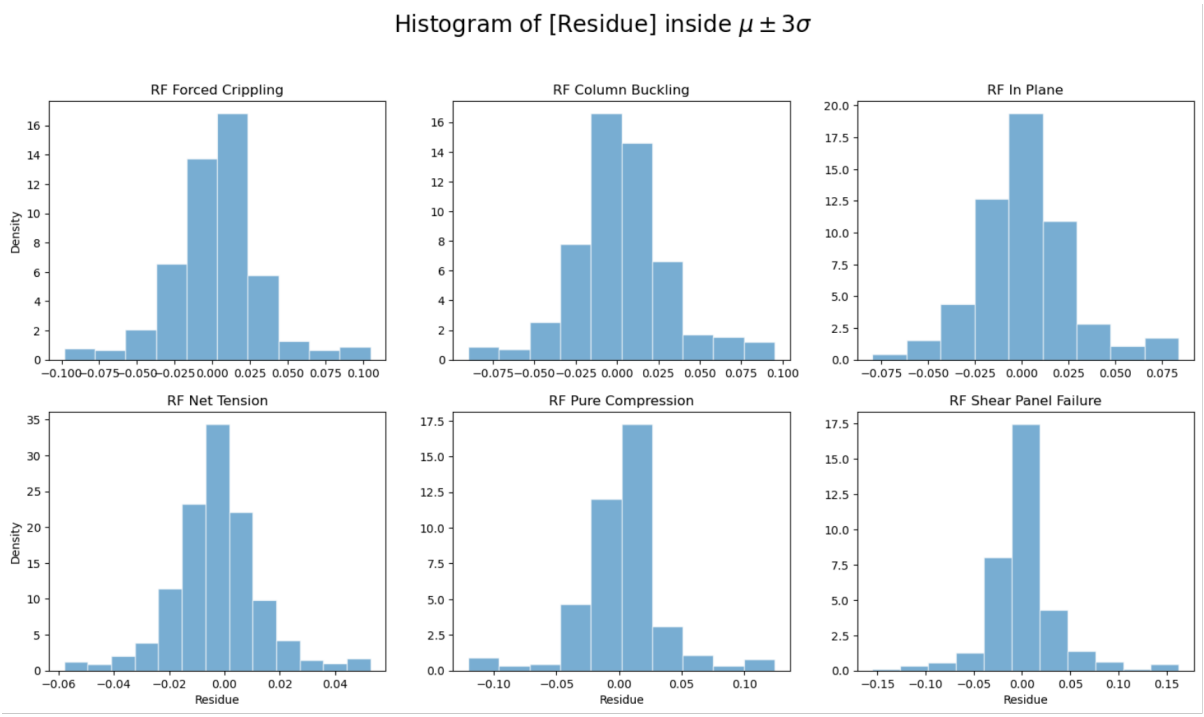


Figure 1.13: Empirical residue distribution sampled from the test set, for each of the six output variables of MS-S18. x-axis limits have been truncated to $\mu \pm 3\sigma$, where the most part of the error lies. Histograms have been appropriately binned for a correct visualization.

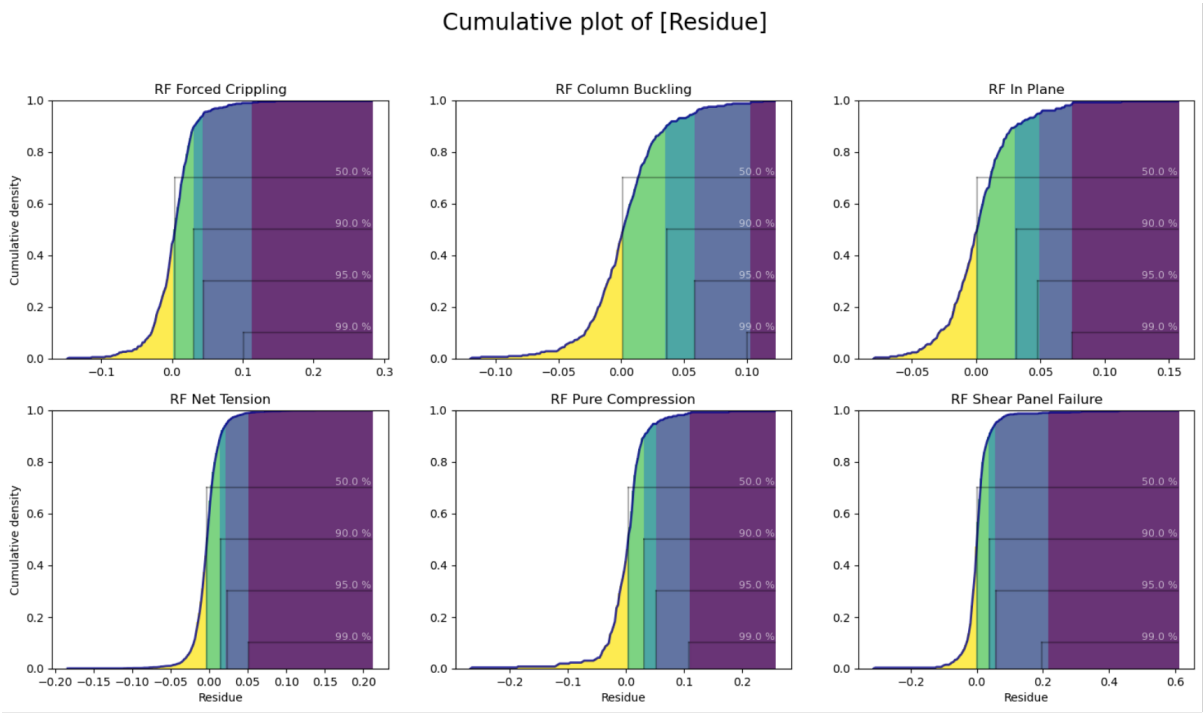


Figure 1.14: Cumulative error distributions corresponding to the PDFs showed in **??**.

Table 1.7: p-value results for the Kolmogorov-Smirnov test performed on distributions showed in Figure 1.12.

Hypothesis Tests Results (p-value)	
	KS
RF Forced Crippling	0.03763
RF Column Buckling	0.11906
RF In Plane	0.03428
RF Net Tension	0.60624
RF Pure Compression	0.08367
RF Shear Panel Failure	0.03510

Table 1.8: P-values of the K-S test comparing the empirical sample of $P(e)$ to each of the indicated distributions. The null hypothesis $H0$ (the empirical distribution has been sampled from the one figuring in a given column) is rejected when $p - \text{value} < 0.05$.

Fit to each output-metric				
	Norm	**Laplace**	**Cauchy**	**Johnsonsu**
RF Forced Crippling	0.004	0.138	0.138	0.613
RF Column Buckling	0.043	0.328	0.435	0.954
RF In Plane	0.005	0.691	0.691	0.616
RF Net Tension	0.000	0.060	0.001	0.843
RF Pure Compression	0.001	0.053	0.438	0.791
RF Shear Panel Failure	0.000	0.002	0.453	0.716

- Normal
- Laplace
- Cauchy

P-values from the corresponding K-S tests are given in

Some informative statistics of the error distribution are given in Table 1.9. The statistics are computed twice, once in the empirical distribution of $P(e)$ sampled from $\mathcal{S}^{\text{test}}$, and the other one using bootstrapping. Bootstrapping allows for calculation of any distribution statistic (take, for instance, the mean, any given quantile, the median, etc.) of an unknown distribution PDF $P(x)$ given a population of N elements sampled from $P(x)$. The bootstrapping method is

described in algorithm 2. This algorithm:

1. Samples N points with replacement from the original population S. N.B. replacement makes the new and the original populations (possibly) different.

2. Statistic x is calculated in the new population.

3. Steps 1 and 2 are repeated $M >> 1$ times, giving a collection of x's (X).

4. If M is sufficiently large, X converges to a Gaussian population. The bootstrapped CI for the statistic x with a 95% confidence is bounded by the percentiles 2.5% and 97.5% of X.

Algorithm 2: Non-parametric bootstrapping

Data: Population $S = \{S_1, S_2, \ldots, S_N\}$ with unknown PDF.
Result: Statistic x's CI

```
Initial ize list: CI = [0]_{1×2};
Initialize list: X = [0]_{1×M};
for i = 1, ..., M >> 1 do
    S_i ← C_S(N, N);
    X(i) ← x_{S_i};
end
CI(1) ← P_X^{2.5%};
CI(2) ← P_X^{97.5%};
```

A basic uncertainty model can be derived from Table 1.10, where informative percentiles of the error distribution are given as statistics calculated in $\mathcal{S}^{\text{test}}$ and as CI computed with the bootstrapping method. If this uncertainty model is trusted (validation of it is discussed in section 1.7), we can use this percentiles for bounding the error of new points which are not present in $\mathcal{S}^{\text{test}}$. Take, for instance, the 5^{th} and the 95^{th} percentiles of the error distribution. Looking into Table 1.10, we see that, for the output variable "RF Forced Crippling", they correspond to -0.0432 and 0.0448, resp. If our uncertainty model is correct, this means that, for new samples, 90% of the times the residue will be bounded by the interval $[-0.0432, 0.0448]$. If we wanted to calculate this bounds as confidence intervals, we would take the bootstrapped percentiles' instead. In such way, with a 95% confidence we would assume that 90% of the times the sampled error is in the interval $[-0.0353, 0.0586]$ (the lower and upper boundaries of the bootstrapped CIs for the 5^{th} and the 95^{th} percentiles, resp.).

Of course, the uncertainty model described in the last paragraph can be fine-tuned using

Table 1.9: Summary of error statistics.

Confidence level at 95%. BS=Confidence Interval (percentile bootstrap), WS=CI (wilson-score)

	count	min	mean	(BS) mean	median	(WS) median	std	(BS) std	IQR	(BS) IQR	kurtosis	(BS) kurtosis	skewness	(BS) skewness	max
RF Forced Crippling	383	-0.15	0.0032	0.0062 -0.00071	0.0041	0.0068 0.000079	0.034	0.039 0.028	0.030	0.035 0.025	14.	24. 2.7	1.4	3.1 -0.74	0.28
RF Column Buckling	319	-0.12	0.0027	0.0061 -0.00045	0.0014	0.0047 -0.0014	0.031	0.034 0.027	0.030	0.035 0.027	2.7	3.8 1.6	0.22	0.81 -0.33	0.12
RF In Plane	252	-0.08	0.0020	0.0059 -0.0013	0.00065	0.0028 -0.0022	0.027	0.031 0.023	0.027	0.030 0.023	5.0	8.3 0.84	1.1	1.9 0.13	0.16
RF Net Tension	2794	-0.18	-0.0026	-0.0019 -0.0034	-0.0028	-0.0022 -0.0033	0.018	0.02 0.017	0.016	0.016 0.015	17.	25. 6.2	0.72	2.0 -0.75	0.21
RF Pure Compression	265	-0.27	0.0019	0.0073 -0.0028	0.0040	0.0077 0.0012	0.041	0.052 0.032	0.029	0.034 0.025	15.	21. 5.1	-0.31	2.4 -2.9	0.26
RF Shear Panel Failure	695	-0.31	0.0038	0.0081 -0.000057	0.00073	0.0026 -0.0011	0.053	0.067 0.038	0.028	0.031 0.026	44.	63. 20.	4.0	5.9 0.037	0.61

Table 1.10: Bootstrapped percentiles (1, 5, 10, 90, 95 and 99) of the residue distribution, calculated with a 95% CI using Wilson score [22].

Confidence level at 95%. BS=Confidence Interval (percentile bootstrap), WS=CI (wilson-score)

	1%	(WS) 1%	5%	(WS) 5%	10%	(WS) 10%	median	(WS) median	90%	(WS) 90%	95%	(WS) 95%	99%	(WS) 99%
RF Forced Crippling	-0.0834	-0.0712 -0.115	-0.0432	-0.0353 -0.0548	-0.0286	-0.0253 -0.0361	0.00405	0.00676 0.0000785	0.0309	0.0389 0.027	0.0448	0.0586 0.0379	0.101	0.147 0.0768
RF Column Buckling	-0.0782	-0.0605 -0.111	-0.0441	-0.0337 -0.0509	-0.0283	-0.0246 -0.0373	0.00139	0.00472 -0.00138	0.0362	0.0453 0.0297	0.0582	0.0656 0.0442	0.100	0.113 0.0686
RF In Plane	-0.0586	-0.0449 -0.0798	-0.0386	-0.0306 -0.0449	-0.0284	-0.0203 -0.0337	0.000646	0.00281 -0.00220	0.0310	0.0418 0.0233	0.0481	0.0664 0.0355	0.0747	0.158 0.0664
RF Net Tension	-0.0506	-0.0444 -0.0633	-0.0277	-0.0261 -0.0298	-0.0198	-0.0187 -0.0213	-0.00280	-0.00224 -0.00329	0.0148	0.0158 0.0135	0.0231	0.0250 0.0211	0.0514	0.0596 0.0437
RF Pure Compression	-0.118	-0.0549 -0.267	-0.0428	-0.0352 -0.074	-0.0289	-0.0244 -0.0389	0.00400	0.00772 0.00124	0.0314	0.0407 0.0258	0.0521	0.0715 0.0369	0.109	0.259 0.0683
RF Shear Panel Failure	-0.0994	-0.0775 -0.123	-0.0498	-0.0392 -0.0578	-0.0307	-0.0258 -0.0361	0.000728	0.00259 -0.00108	0.0376	0.0445 0.0299	0.0572	0.0696 0.0489	0.197	0.316 0.0902

additional information about the error distribution. If we found $P(e)$ to be heteroscedastic, we could benefit from conditioning our uncertainty model to certain regions of the input (or the output) space. This is the main motivation for section 1.5, section 1.6 and section 1.7.

The non-parametric bootstrapping method described above is useful for when the parametrised PDF of the residue, $P(e)$, is unknown. But we can try to fit $P(E)$ to any (known) probability density function. If a match was found, bootstrapping would not be needed and the matching PDF's statistics could be directly used for uncertainty estimation. With this aim, we

1.5 Distributed error quantification: conditioning the error distribution on the input and the output space

1.6 Distributed error quantification: conditioning the error distribution on the output space

1.7 Uncertainty model estimation

Table 1.11: Output of algorithm 1: `reqs_results` table

	Voxel
Voxel ID	
0	(0.0, Fr68-Fr69, Str40)
1	(0.9, Fr66-Fr67, Str36p)
2	(0.0, Fr69-Fr70, Str29)
3	(0.9, Fr69-Fr70, Str35)
4	(0.0, Fr65-Fr66, Str01)
...	...
827	(0.9, Fr64-Fr65, Str21)
828	(0.9, Fr66-Fr67, Str32)
829	(0.0, Fr64-Fr65, Str26p)
830	(0.0, Fr64-Fr65, Str29p)
831	(0.0, Fr67-Fr68, Str07)

751 rows × 1 columns

The first step is check whether the train and test datasets follow the same empirical distributions. If test data is found to not follow the same distribution as the train data, the evaluation of the model would be biased or the test data would be outside input applicability. Take, for instance, the extreme case in which the statistical distribution for some input numerical variable follows a uniform distribution $x_{train} \sim U(a_1, a_2)$ whereas the same input variable in the test dataset follows another distribution $x_{test} \sim U(b_1, b_2)$ where $b_1 \leq a_2$. Clearly, x_{test} is outside the hypercube spanned by x_{train} and is outside the applicability region as a consequence.

Definition 2. *Let* $\mathbf{X}$ *represent some dataset, which has been split in* $\mathbf{X}^{TRAIN}$ *and* $\mathbf{X}^{TEST}$*.* $\mathbf{X}$ *is said to be order 1 [O(1)] well-split if and only if* $\mathbf{X}^{TEST} \subseteq Hull[\mathbf{X}^{TRAIN}]$*.*

Where we denote the convex hull of $\mathbf{X}^{TRAIN}$ *by* $Hull[\mathbf{X}^{TRAIN}]$*.*

Definition 3. *Let* $\mathbf{X}$ *represent the dataset defined in 2.* $\mathbf{X}$ *is said to be order 2 well-split if and only if*

Table 1.12: Voxel reference table. Lorem ipsum dolor sit amet, consectetuer adisciping elit.

	Voxel
Voxel ID	
0	(0.0, Fr68-Fr69, Str40)
1	(0.9, Fr66-Fr67, Str36p)
2	(0.0, Fr69-Fr70, Str29)
3	(0.9, Fr69-Fr70, Str35)
4	(0.0, Fr65-Fr66, Str01)
...	...
827	(0.9, Fr64-Fr65, Str21)
828	(0.9, Fr66-Fr67, Str32)
829	(0.0, Fr64-Fr65, Str26p)
830	(0.0, Fr64-Fr65, Str29p)
831	(0.0, Fr67-Fr68, Str07)

751 rows × 1 columns

1.8 Las movidas de Rodrigo

Orbit propagation is a very wide and varied field. It comprises many different approaches and methods to obtain more or less accurate results, with a higher or lower computational cost. Some examples are the numerical integration of the equations of motion in cartesian coordinates, the numerical integration of the variational equations (*i.e.* equations of motion expressed in OEs) and the development of closed-form solutions by simplifying the problem. Within this last group, State Transition Matrices arise.

The State Transition Matrix is a linearization procedure of a nonlinear dynamical system. It is used to approximate the dynamics of said system over short periods of time, allowing for a lower computational cost while maintaining an acceptable accuracy. This concept is not restricted to orbital mechanics, although it is one of the main fields in which it is used [24].

This section intends to provide some background in (a) its mathematical formulation and (b) its applications in orbit theory.

1.8.1 Concept: System dynamics.

Consider the uncontrolled, nonlinear dynamic system that is characterized through the state vector $\underline{y} = [y_1, y_2, \ldots, y_n]$. The Initial Value Problem (IVP) for this system may be expressed as:

$$[P] \equiv \begin{cases} \text{Eq.} & \dfrac{d\underline{y}}{dt} = \boldsymbol{F}(\underline{y}, t) \\ \text{ICs.} & \underline{y}(t_0) = \underline{y}_0 \end{cases} \tag{1.3}$$

where $\boldsymbol{F}(\underline{y}, t)$ represents the nonlinear dynamics of the system. This problem is unsolvable in general, mainly due to its nonlinearity. In the context of orbit propagation, the state vector $\underline{y}$ might be the position and velocity (be it relative or absolute) of the celestial body, and the dynamics function $\boldsymbol{F}$ contains the considered force model. In order to arrive at a closed-form, solvable problem, it is assumed that the solution $\underline{y}(t)$ can be expressed as:

$$\underline{y}(t) = \Phi(t, t_0)\underline{y}(t_0) \tag{1.4}$$

where $\Phi(t, t_0)$ is the State Transition Matrix (STM) of the system. This matrix allows the state vector at a certain epoch t to be calculated as the product of the matrix times the initial condition. This expression is obviously very favorable, but the question now is how does one

compute it. Its actual definition can be easily derived from (1.4) as:

$$\Phi(t, t_0) \equiv \frac{\partial \underline{y}}{\partial \underline{y}_0} \tag{1.5}$$

Yet again, how to compute it is not clear at all. There are three main options, depending on the situation:

A. If the nonlinear solution as a function of the initial condition is known, then the expression (1.5) is directly applied. This is an uncommon case, although simplified examples exist in the orbit propagation field. For example, the Keplerian motion equations expressed in Keplerian orbital elements (OEs) can be solved this way, due to the trivial remaining equations. Another example is the Clohessy-Wiltshire solution, from which the STM can be directly obtained. This process is detailed later on in section **??**.

B. The nonlinear solution is unknown in the original state space, but can be calculated in a different space through a transformation. Mathematically, this can be written as:

$$\Phi_y(t, t_0) = \frac{\partial \underline{y}}{\partial \underline{y}_0} = \frac{\partial \underline{y}}{\partial \underline{u}} \frac{\partial \underline{u}}{\partial \underline{u}_0} \frac{\partial \underline{u}_0}{\partial \underline{y}_0} \equiv W(t) \Phi_u(t, t_0) \left(W(t_0)\right)^{-1} \tag{1.6}$$

where $W(t)$ is the transformation matrix, where it is assumed that the transformation $\underline{y} = h(\underline{u})$ is known. An example of this kind of approach is the transformation of the Cartesian equations of motion into the Keplerian OEs, whose solution is known, as mentioned in A.. This is a very commonly used method in relative orbit propagation, as in [25, 26].

C. If none of the above can be performed, then the STM can be integrated itself, to be then used to calculate the state vector. This starts by differentiating (1.3) with respect to the initial condition $\underline{y}_0$, leading to:

- $$\frac{\partial}{\partial \underline{y}(t_0)} \frac{d\underline{y}(t)}{dt} = \frac{d}{dt} \frac{\partial \underline{y}(t)}{\partial \underline{y}(t_0)} = \frac{d}{dt} \boldsymbol{\Phi}(t, t_0)$$
- $$\frac{\partial \boldsymbol{F}(t, \underline{y})}{\partial \underline{y}(t_0)} = \frac{\partial \boldsymbol{F}(t, \underline{y})}{\partial \underline{y}(t)} \frac{\partial \underline{y}}{\partial \underline{y}(t_0)} = \boldsymbol{A} \boldsymbol{\Phi}(t, t_0)$$

$$\Rightarrow [P] \equiv \begin{cases} \text{Eq.} & \dfrac{d}{dt} \Phi(t, t_0) = \boldsymbol{A} \Phi(t, t_0) \\ \text{IC} & \Phi(t_0, t_0) = \mathbb{I}_{nxn} \end{cases} \tag{1.7}$$

This last method is a bit unrewarding, as it forces one to integrate an IVP. However, the problem in terms of $\Phi(t, t_0)$ (eq. (1.7)) might be simpler or more efficient than the original (eq. (1.3)), although it is rare. An example of this approach is shown later in section **??**.

1.8.2 Applications of STMs in celestial mechanics.

State Transition Matrices can be useful in a wide range of spacecraft dynamics applications. Some of the most important are [27]:

A. Precise Orbit Determination.

Precise Orbit Determination (POD) is a method through which the orbit of a flying spacecraft can be determined with a high accuracy [28]. This estimation is performed using general orbit determination algorithms, such as Kalman filtering or a batch least squares. It requires both high-precision geodetic receivers and high-precision dynamics models, where STMs comes to play. POD usually requires all typically important perturbations, such as non-spherical gravity, drag, tidal forces ...

B. Guidance, Navigation and Control (GNC).

GNC deals with the design the systems to control the spacecraft. It involves the determination of the desired trajectory (guidance), the instantaneous determination of the spacecraft's position (navigation) and the manipulation of the controllers to execute guidance commands (control). STMs become very useful specially in situations in which the linearization error is small, such as in rendez-vous, station-keeping or formation flying operations. They prove to be useful in all three branches:

- Guidance: STMs are a lightweight yet precise tool to generate reference trajectories.
- Navigation: Signal filtering makes extensive use of STMs for the propagation of the estimated state. This is certainly one of the most relevant applications.
- Control: Algorithms like robust online optimal control involve state propagation, using the STM.

C. Orbit design.

Alternatively, rather than propagating already defined orbits, it may be useful to solve the inverse problem: that is, find the orbit that satisfies a set of conditions (*e.g.* optimality). This

is specially relevant for the Circular Restricted Three Body Problem (CR3BP), or its particular case of Halo orbits (periodic 3D orbit near one of the Lagrange libration points). STMs are quite useful to determine an initial solution for said Halo orbits, and can also be used to evaluate the effect of a deviation in initial conditions or other parameters.

D. Covariance matrix propagation.

Some degree of uncertainty will always be assumed in the estimation of a spacecraft's position and velocity. This matter becomes really important in the context of collision avoidance, where ideally, said uncertainty would be exactly calculated. However, these values are usually not something inherently worrying. What is worrying indeed is that this uncertainty may propagate in a divergent fashion, leading to unbounded trajectories with its inherent collision risk.

Here is where the covariance matrix comes to light. Conceptually, it can be seen as an entity which indicates how certain are each of the components of the state vector. That is represented by their variances and covariances. Mathematically, it is defined as:

$$P(t) = E\left[\left(\hat{\underline{x}} - \underline{x}\right)\left(\hat{\underline{x}} - \underline{x}\right)^T\right]$$

If one knew the covariance at a certain epoch t_j, it is possible to map it to a different epoch t_k (latter or earlier) through the STM, that is:

$$P(t_k) = E\left[\left(\hat{\underline{x}_k} - \underline{x_k}\right)\left(\hat{\underline{x}_k} - \underline{x_k}\right)^T\right] = E\left[\Phi(t_k, t_j)\left(\hat{\underline{x}_j} - \underline{x_j}\right)\left(\hat{\underline{x}_j} - \underline{x_j}\right)^T \Phi^T(t_k, t_j)\right]$$

$$\Rightarrow P(t_k) = \Phi(t_k, t_j) P(t_j) \Phi^T(t_k, t_j)$$

Now, once the mathematical formalism has been stated, it is time to apply it to the situation at hand. Spacecraft state uncertainty has essentially two sources: estimation error (associated to navigation) and execution error (associated to control/manoeuvres).

In a fairly relaxed approach (*i.e.* assuming the state variables be decoupled), the covariance matrix associated to the estimation error has the following shape:

$$P_{est} = \begin{bmatrix} \delta x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \delta y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \delta z^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & \delta v_x^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & \delta v_y^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & \delta v_z^2 \end{bmatrix} \tag{1.8}$$

where $\delta\psi$ indicates the standard deviation of the variable ψ. An useful way to visualize the covariance is the probability ellipsoid, defined by:

$$\begin{bmatrix} \tilde{x} & \tilde{y} & \tilde{z} \end{bmatrix} P_{xyz}^{-1} \begin{bmatrix} \tilde{x} \\ \tilde{y} \\ \tilde{z} \end{bmatrix} = l^2$$

where $\tilde{x}, \tilde{y}, \tilde{z}$ are the coordinates of a point of the ellipsoid (referred to its center), P_{xyz} is the partition of the covariance matrix related to the position and l is the sigma coefficient. That is, for $l = 2$, the equation represents the ellipsoid in which the spacecraft will be found with a 2σ probability. An example of the evolution of this ellipsoid can be seen for a small radial hop in figure 1.15.

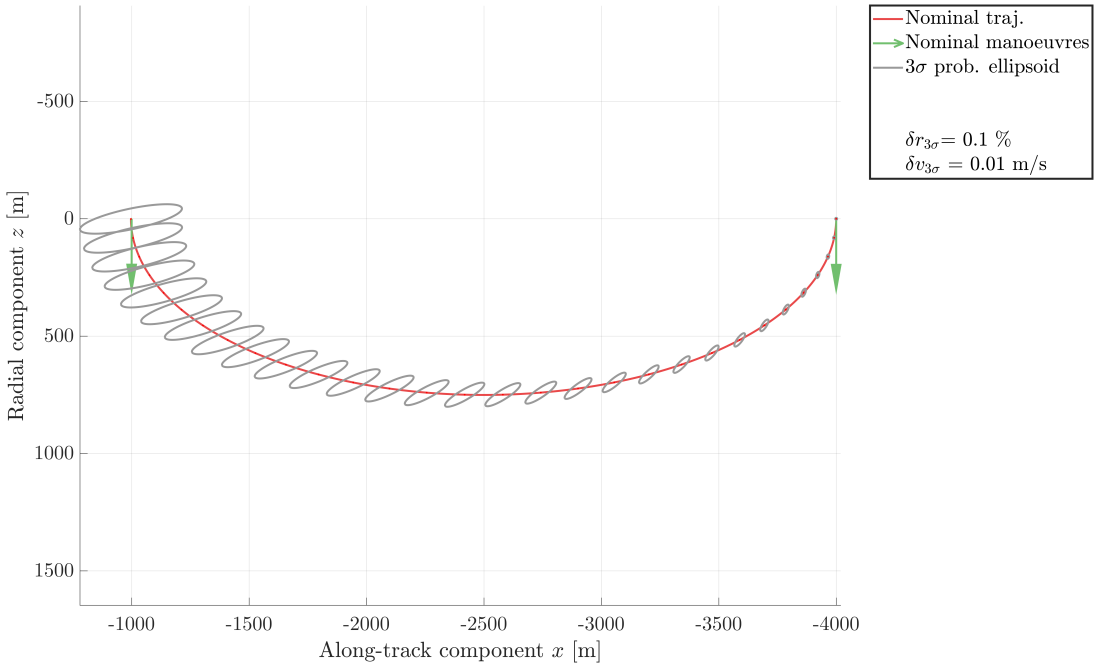


Figure 1.15: Covariance ellipsoid evolution along a radial hop.

Most of these concepts lie out of the thesis' scope. However, it is important to know that the STMs developed or quoted throughout this thesis can be used in many different fields of celestial mechanics.

Chapter 2

Conclusions and research outlook.

The increasingly higher relevance of relative motion is beyond question. Satellite constellations, rendez-vous operations and formation flying missions are becoming an everyday occurrence, and their growing success rates are no coincidence. Incrementally better models (*i.e.* truer-to-life, accounting for more effects) have an essential role in this.

In order to gain a comprehensible insight on this topic, this thesis has followed the same bottom-up structure, starting from the very simple Hill/Clohessy-Wiltshire equations up to the modelling of perturbations. The use of linear, closed-form models (through STMs) is an advantageous approach: firstly, computational cost drops, but even more importantly, deeper knowledge about the dynamics of the system is achieved. This allows for, among other things, the design of relative orbits with certain characteristics (*e.g.* safe orbits).

Any state-of-the-art model requires an analysis of the relevant perturbations to be carried out. For the case of Earth artificial satellites, this usually leaves the non-spherical gravity as the most important of them. Kaula's theory allows for the analysis of the effect of any spherical harmonic, which naturally leads for better accuracy models. Although it features certain singular cases (resonances and circular or equatorial orbits), these can be solved. Many applications can be branched from this theory, such as mean-to-osculating transformations, linearized relative motion models, resonance analysis ...

In conclusion, the modelling of perturbations is key for the construction of accurate linearised relative motion models as well as relative orbit design; both of which play a paramount role in the constantly growing discipline of distributed space systems.

Research outlook and future work.

The very wide horizon in which this topic is framed leads for countless possible research areas.

The evaluation of a robust implementation of Kaula's theory is the first on the list. This would be followed by an analysis of the resonant and secular terms, such as done by Chao [29]. The development of mean-to-osculating transformations through this theory is also considered,

which could derive into the development of linear relative motion models based on it.

In a somewhat more general sense, the survey, analysis and implementation of perturbed relative motion models (STMs) is to be tackled. A performance analysis of these models against a High-Fidelity and unperturbed models would be insightful, surely.

Orbit safety might also be an interesting topic to follow up on. A review of the existent models for non-circular, perturbed reference orbits would be insightful, and would continue on the here developed models for simpler cases (*i.e.* circular perturbed and elliptic unperturbed).

Further down the road, other types of perturbations might be analyzed, through the averaging methods here presented. Third bodies or drag are the most relevant ones, although second-order NSG effects are also to be considered.

Finally, real world scenarios may be analyzed, and eventually, a self-contained tool for this purpose could be regarded. Hopefully, this sort of roadmap looks as compelling as it does to the author.

Appendix A

Absolute and relative orbital element sets.

A.1 Introduction.

The description of a spacecraft's state is done via a state vector. While it can include several variables with other purposes (*e.g.* filtering), its only information throughout this thesis is the position and velocity. There are two main ways to describe them:

A. Through **cartesian coordinates**

B. Through **orbital elements**

While the first option yields a very explicit and graphic-ready description, the second one usually has two advantages over it. Firstly, orbital elements are generally more intuitive about both the orbit and the position on it. Secondly, as orbital elements are generally slow-varying, they allow for a bigger integration timestep without losing accuracy. This is quite clear when studying keplerian motion, as most of the elements remain constant. Variational formulation and Hamilton-Jacobi theory (with the notion of changing variables as the full solution of a problem) relate to this fact.

Throughout this thesis, several sets of orbital elements have been used. The goal of this appendix is to clarify on the definition and differences between them. Absolute orbital elements (OEs) will be described first, followed by relative OEs (ROEs).

A.2 Absolute element sets.

A.2.1 Workflow for transformations between absolute element sets.

Consider two different sets of OEs, denoted by $\underline{OE}$ and $\underline{\widetilde{OE}}$. The transformation function $\boldsymbol{G}_{OE\to\widetilde{OE}}$ between them is defined by:

$$\underline{\widetilde{OE}} = \boldsymbol{G}_{OE\to\widetilde{OE}}(\underline{OE}) \tag{A.1}$$

A numerous amount of element sets have been historically defined. Nevertheless, some of them are much more commonly used than others. Although this analysis will be restricted to a short number of sets (say n), the number of transformations becomes arduously large as n increases ($n(n-1)$).

In order to reduce the number of transformation functions $\boldsymbol{G}$, the later defined Keplerian OEs (KOEs) will be used as a pivot, that is, building only transformations to and from KOEs. This will in turn reduce the number of required functions to $2n$. The Keplerian set also has a further advantage: as it is the classical element set, almost every other set is defined explicitly in terms of it, so that transformations to and from them can be easily derived. A simple, graphical explanation of this is shown in figure A.1.

Figure A.1: Workflow for transforming between two arbitary absolute element sets.

A.2.2 Element sets.

A.2.2.1 Keplerian orbital elements (KOE).

The Keplerian set of OEs (KOE) is one of the most widely used and classic options. An usual definition is the following:

$$\begin{cases} a & \equiv & \text{Semimajor axis} & [L] \\ e & \equiv & \text{Eccentricity} & [--] \\ i & \equiv & \text{Inclination} & [rad] \\ \Omega \text{ or } RAAN & \equiv & \text{Right ascension of the ascending node} & [rad] \\ \omega & \equiv & \text{Argument of periapsis} & [rad] \\ M & \equiv & \text{Mean anomaly} & [rad] \end{cases} \tag{A.2}$$

The last element commonly varies across literature, being substituted by the true anomaly θ; or, when tackling the variation of orbital parameters, by the mean anomaly at $t = 0$ (M_0) or the perigee time T_0 [30]. Mean anomaly is used due to the simplicity of its unperturbed variational equation, as it has a constant rate (denoted by n). The geometrical meaning and definition of these elements is out from the scope of this thesis. Nonetheless, figure A.2 shows a simple geometrical drawing of the involved angles.

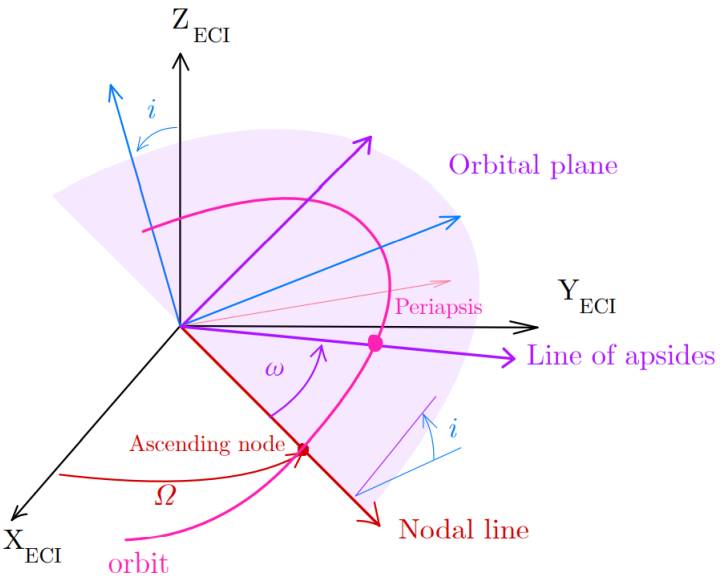


Figure A.2: Frame rotation from inertial to perifocal frame.

As it is seen in the figure before, the Keplerian elements become singular in two cases:

A. If the **inclination** is null, the orbital plane is coincident with the inertial reference (ECI x-y) plane. The ascending node is hence undefined in this case.

B. If the **eccentricity** is null, the periapsis is not defined, as it is the nearest point of the orbit around the central body. Thus, there is no angle defining its position, making the argument of periapsis singular.

These singularities are unfortunately quite common in orbit design. They correspond respectively with equatorial and circular orbits. In order to avoid this behaviour, many different elements sets have been defined. Wiesel [30] shows an intuitive approach in section 2.10, solving either problem with a graphic approach.

A.2.2.2 Eccentricity/inclination vectors orbital elements (EIOE).

This set, originally defined for geostationary orbits in absolute terms [31], is used mainly as a relative OE set. Though it is actually not used along this thesis, its definition is helpful for introducing the relative counterpart.

I.Eccentricity vector.

The notion of the eccentricity vector is quite basic, as it is, when in unperturbed motion, a constant of the dynamic system. It is defined as the eccentricity-sized vector pointing towards the perigee. Nonetheless, for this purpose, the eccentricity vector is defined as [32]:

$$\underline{e} = \left\{ \begin{array}{c} e_x \\ \\ e_y \end{array} \right\} = e \left\{ \begin{array}{c} \cos\varpi \\ \\ \sin\varpi \end{array} \right\} \tag{A.3}$$

where the argument of perigee ω might be substituted with the sum $\omega + \Omega$ [as in 31]. A graphical representation can be seen later in the relative definition A.5(a). As it arises from (A.3), it substitutes the eccentricity and argument of perigee from the Keplerian OE set.

II. Inclination vector.

The inclination vector is perpendicular to the orbital plane, similarly to the angular momentum, but inclination-sized. It is defined by its components as [31]:

$$\underline{i} = \left\{ \begin{array}{c} i_x \\ \\ i_y \end{array} \right\} = i \left\{ \begin{array}{c} \cos\Omega \\ \\ \sin\Omega \end{array} \right\}$$

The graphical interpretation is not as straightforward as for the eccentricity vector. It is clear that these components substitute the out-of-plane related elements i and Ω.

III.Element set.

The EI orbital element set is then composed of:

$$\begin{cases} a & \equiv & \text{Semimajor axis} & [L] \\ e_x = e\,\cos\omega & \equiv & \text{x-projection of } \underline{e} & [--] \\ e_y = e\,\sin\omega & \equiv & \text{y-projection of } \underline{e} & [--] \\ i_x = i\,\cos\Omega & \equiv & \text{x-component of } \underline{i} & [--] \\ i_y = i\,\sin\Omega & \equiv & \text{y-component of } \underline{i} & [--] \\ \lambda = \omega + M & \equiv & \text{Mean argument of latitude} & [rad] \end{cases} \tag{A.4}$$

A.2.2.3 Quasi-nonsingular orbital elements (QNSOE).

The quasi-nonsingular (QNS) orbital element set tackles the singularity existing in circular orbits [33], [34] [35]. It is quite similar to the formerly defined EI set, as it uses again the components of the eccentricity vector to substitute e and ω. The set is then defined as:

$$\begin{cases} a & \equiv & \text{Semimajor axis} & [L] \\ q_1 = e\,\cos\omega & \equiv & \text{x-projection of } \underline{e} & [--] \\ q_2 = e\,\sin\omega & \equiv & \text{y-projection of } \underline{e} & [--] \\ i & \equiv & \text{Inclination} & [rad] \\ \Omega & \equiv & \text{Right ascension of the ascending node} & [rad] \\ \lambda = \omega + M & \equiv & \text{Mean argument of latitude} & [rad] \end{cases} \tag{A.5}$$

Though some authors use a different order, this is the one used in this thesis, so as to keep the time-varying element on the last place.

A.2.2.4 Equinoctial orbital elements (EOE).

The QNS set of elements only solved half of the singularity problem. To solve both, thus enabling the description of equatorial and polar orbits, the equinoctial set of elements is defined

as:

$$\begin{cases} a & \equiv & \text{Semimajor axis} & [L] \\ P_1 = e\cos\varpi & \equiv & \textit{unclear physical meaning, similar to } e_x & [--] \\ P_2 = e\sin\varpi & \equiv & \textit{unclear physical meaning, similar to } e_y & [--] \\ Q_1 = \tan\frac{i}{2}\cos\Omega & \equiv & \textit{unclear physical meaning, similar to } i_x & [--] \\ Q_2 = \tan\frac{i}{2}\sin\Omega & \equiv & \textit{unclear physical meaning, similar to } i_y & [--] \\ L = \Omega + \omega + \theta & \equiv & \text{True longitude} & [rad] \end{cases} \tag{A.6}$$

where $\varpi = \Omega + \omega$. Not only does the order does change depending on the author, but also the symbols to refer to them. An example of its use is [33].

A.2.2.5 Delaunay orbital elements (DOE).

Delaunay elements arise when formulating the two-body problem through analytical mechanics **??**. Conversely to the previous element sets, who are clearly non-canonical (*i.e.* they do not satisfy Hamilton's equations), Delaunay elements are built based on the analytical perspective of the problem. Starting from the canonical set of elements (see appendix **??**), Delaunay elements are reached after performing a canonical transformation, leading to the following definition:

$$\begin{cases} L = \sqrt{\mu\, a} & \equiv & \textit{unclear physical meaning} & [L^{1/2}] \\ G = L\sqrt{1-e^2} & \equiv & \text{Angular momentum} & [L^{1/2}] \\ H = G\cos i & \equiv & \text{Polar component of angular momentum} & [L^{1/2}] \\ l = M & \equiv & \text{Mean anomaly} & [rad] \\ g = \omega & \equiv & \text{Argument of perigee} & [rad] \\ h = \Omega & \equiv & \text{Right ascension of ascending node} & [rad] \end{cases} \tag{A.7}$$

This set is mainly used in the context of perturbations, as it yields a very convenient expression for the perturbed Hamiltonian (see section **??**).

A.3 Relative sets.

Relative elements are at the deepest roots of spacecraft relative motion, offering several advantages over cartesian relative states. First and foremost, they are more intuitive, but they also lead to a reduction of linearisation errors when expanding the deputy's movement around the chief's orbit [36]. In general, relative elements are defined as:

$$\delta\underline{OE} = \boldsymbol{f}\left(\underline{OE_C}, \underline{OE_D}\right) \tag{A.8}$$

which is usually simplified by just taking the arithmetic difference between them, namely

$$\delta\underline{OE} = \underline{OE}_D - \underline{OE}_C \tag{A.9}$$

where the subscripts denote respectively the deputy and chief spacecraft. The question now is, how do transformations between ROEs work.

A.3.1 Workflow for transformations between ROEs.

As for the absolute elements, Keplerian elements will be used as a pivot point. That means that only the transformations from and to RKOEs must be implemented. There are then two types of transformations:

A) From any ROE set to RKOE

While authors provide with scenarios expressed in their own ROE set, the element choice for this thesis is the Keplerian set. That leads to the need of implementing a transformation from the former set to the latter. The considered inputs and outputs are:

- **Inputs**:
  - $\underline{\widetilde{ROE}} = \delta\underline{\widetilde{OE}}$: Any type of ROEs, whose absolute equivalents are known as a function of the KOEs ($\underline{\widetilde{OE}} = \boldsymbol{f}\left(\underline{KOE}\right)$)
  - $\underline{KOE}_C$: Chief spacecraft/reference orbit KOEs
- **Output**:
  - $\underline{RKOE} = \delta\underline{KOE}$: Keplerian ROEs

Taking equation (A.9) and particularizing it for KOEs:

$$\delta\underline{KOE} = \underline{KOE}_D - \underline{KOE}_C \tag{A.10}$$

while the second term is known (input), the second one must be calculated through a certain process:

1st Calculate chief's OEs in the source phase space (*i.e.* $\underline{\widetilde{OE}}_C$)

$$\underline{\widetilde{OE}}_C = \boldsymbol{G}_{KOE\rightarrow\widetilde{OE}}\left(\underline{KOE}_C\right)$$

2nd Compute deputy's OEs by direct addition

$$\underline{\widetilde{OE}}_D = \underline{\widetilde{OE}}_C + \delta\underline{\widetilde{OE}}$$

3rd Compute deputy's KOEs by back-transformation

$$\underline{KOE}_D = \boldsymbol{G}_{\widetilde{OE}\to KOE}\left(\underline{OE}_D\right)$$

4th Subtract chief's KOEs from deputy's

$$\delta\underline{KOE} = \underline{KOE}_D - \underline{KOE}_C$$

See graphic A.3 for a more visual explanation.

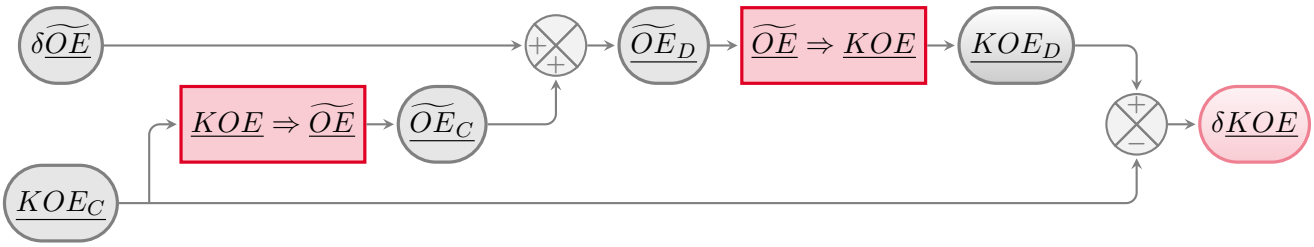


Figure A.3: Workflow for transforming any relative set into KOE.

B) From RKOE to any ROE set

In this case,the assumed inputs and outputs are:

- **Inputs:**
  - $\underline{RKOE} = \delta\underline{KOE}$: Keplerian ROEs
  - $\underline{KOE}_C$: Chief KOEs
- **Output**:
  - $\underline{\widetilde{ROE}} = \delta\underline{\widetilde{OE}}$: Any type of ROEs, whose absolute equivalents are known as a function of the KOEs ($\underline{\widetilde{OE}} = \underline{f}\left(\underline{KOE}\right)$)

For this transformation, the equation A.8 particularized for this case acquires the following shape:

$$\delta\widetilde{\underline{OE}} = \widetilde{\underline{OE}_D} - \widetilde{\underline{OE}_C} \tag{A.11}$$

Equation A.11 can be tackled in two main ways:

A. Using the pertinent transformations, compute the absolute elements for both spacecrafts $\widetilde{\underline{OE}_D}$, $\widetilde{\underline{OE}_C}$, and then calculate the arithmetic difference (in a A.3.1). See graphic A.4.

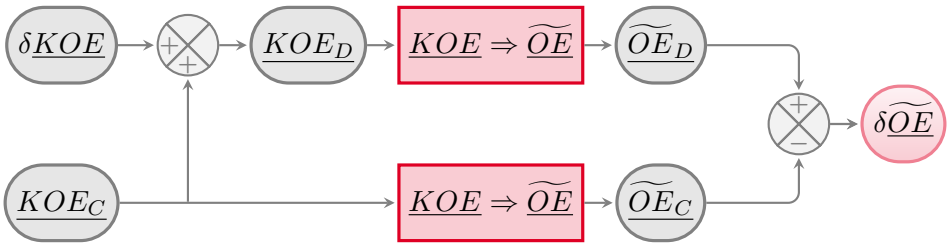


Figure A.4: Workflow for transforming RKOE into any other set.

B. Expand the deputy absolute OEs (*i.e.* $\widetilde{\underline{OE}_D}$) around the chief via a Taylor series expansion with respect to the Keplerian set of elements, retaining terms up to first order, achieving a linearised expression for the transformation. Mathematically:

$$\widetilde{\underline{OE}_D} = \widetilde{\underline{OE}}(\underline{KOE}_D) = \widetilde{\underline{OE}}(\underline{KOE}_C + \delta\underline{KOE})) = \widetilde{\underline{OE}_C} + \frac{\partial\widetilde{\underline{OE}}}{\partial\underline{KOE}}\delta\underline{KOE} + \mathcal{O}\left(\delta\underline{KOE}^2\right)$$

hence,

$$\delta\widetilde{\underline{OE}} \approx \widetilde{\underline{OE}_C} + \frac{\partial\widetilde{\underline{OE}}}{\partial\underline{KOE}}\delta\underline{KOE} - \widetilde{\underline{OE}_C} = \frac{\partial\widetilde{\underline{OE}}}{\partial\underline{KOE}}\delta\underline{KOE} \tag{A.12}$$

where the Jacobian matrix is generally simple, as it usually only implies polynomic or trigonometric functions. Equation (A.12) is then a first order approximation of (A.11). Its validity is then reduced to a close proximity between both spacecrafts, which should be assessed.

A.3.2 Element sets.

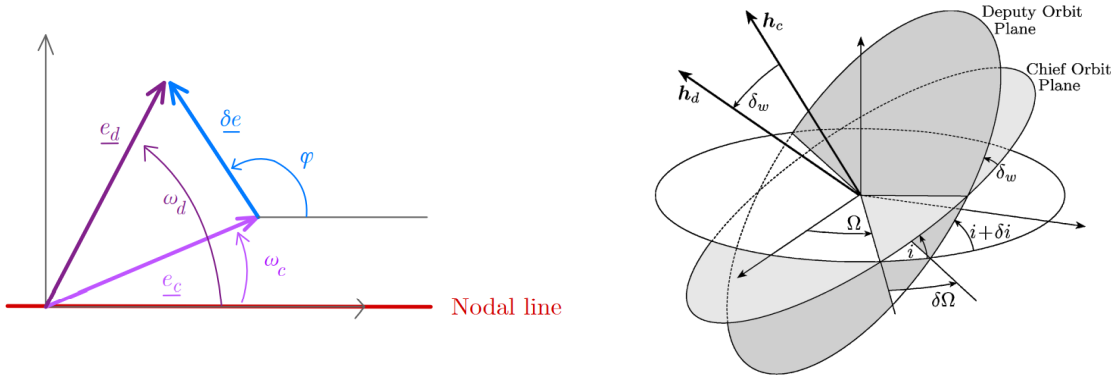
Besides the ones derived directly from its absolute counterparts, a couple of additional ROE sets will be herewith defined and explained. This is due to one of two reasons. The first one is that some ROE sets are only defined in relative terms, lacking any absolute equivalent. The second one is that it might be interesting to dive in the meaning of the relative sets, deriving

interesting relations that would otherwise be overlooked.

A.3.2.1 Relative eccentricity/inclination vectors orbital elements (REIOE).

This ROE set is the counterpart of the EI set (see A.2.2.2). It is nonetheless interesting to see the meaning and shape of it, as it is quite widely used in literature [32, 34, 35]. Firstly, the elements themselves are defined, to later analyze the meaning behind them:

$$\begin{cases} \delta a & \equiv & \text{Relative semimajor axis} & [L] \\ \delta e_x & \equiv & \text{x-component of } \delta\underline{e} & [--] \\ \delta e_y & \equiv & \text{y-component of } \delta\underline{e} & [--] \\ \delta i_x & \equiv & \text{x-component of } \delta\underline{i} & [--] \\ \delta i_y & \equiv & \text{y-component of } \delta\underline{i} & [--] \\ \delta\lambda & \equiv & \text{Relative mean argument of latitude} & [rad] \end{cases} \tag{A.13}$$



(a) Relative eccentricity vector.

(b) Relative inclination vector [35] .

Figure A.5: Relative ccentricity & inclination vectors.

Concept & meaning

The relative eccentricity vector components substitute the relative eccentricity and the relative argument of perigee. It is based on the eccentricity vector definition (A.3), and a graphical representation can be seen in figure A.5(a). Mathematically:

$$\delta\underline{e} = \begin{Bmatrix} \delta e_x \\ \\ \delta e_y \end{Bmatrix} = \delta e \begin{Bmatrix} \cos\varphi \\ \\ \sin\varphi \end{Bmatrix}$$

where φ is referred to as the AOP of the relative orbit. This vector rules the in-plane relative motion (hand in hand with δa and $\delta\lambda$). There are two ways of tackling the transformation from RKOE to this set (see A.3.1). Besides the exact nonlinear form, a linear version can be derived. Assuming that the difference in the eccentricity vector is due to that of the eccentricity and argument of perigee (δe, $\delta\omega$), it follows that:

$$\delta\underline{e} \approx \begin{bmatrix} \cos\omega & -e\,\sin\omega \\ \\ \sin\omega & e\,\cos\omega \end{bmatrix} \begin{Bmatrix} \delta e \\ \\ \delta\omega \end{Bmatrix} \tag{A.14}$$

where terms of second order and higher have been neglected. The relative inclination vector is defined in an alternative way [32] (comparing with the absolute counterpart). Mathematically:

$$\delta\underline{i} = \sin\delta i \begin{Bmatrix} \cos\theta \\ \\ \sin\theta \end{Bmatrix}$$

where θ is the analogue angle to φ in the eccentricity vector, being commonly referred to as the RAAN of the relative orbit. Once again, the linearised transformation from RKOE to this set may be analysed, considering the differences δi and $\delta\Omega$. Applying the law of sines and the law of cosines for spherical trigonometry and assuming small values of δi and $\delta\Omega$, it follows that:

$$\delta\underline{i} = \begin{Bmatrix} \delta i \\ \\ \sin i\delta\Omega \end{Bmatrix} \approx \begin{bmatrix} 1 & 0 \\ \\ 0 & \sin i \end{bmatrix} \begin{Bmatrix} \delta i \\ \\ \delta\Omega \end{Bmatrix} \tag{A.15}$$

where i is the inclination of the chief's orbit. Combining the results of (A.14) and (A.15) with the definitions of the remaining elements, an expression analogue to (A.12) is reached:

$$\begin{Bmatrix} \delta a \\ \delta e_x \\ \delta e_y \\ \delta i_x \\ \delta i_y \\ \delta \lambda \end{Bmatrix} \approx \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \cos\omega & 0 & 0 & -e\sin\omega & 0 \\ 0 & \sin\omega & 0 & 0 & e\cos\omega & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & \sin i & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix} \begin{Bmatrix} \delta a \\ \delta e \\ \delta i \\ \delta \Omega \\ \delta \omega \\ \delta M \end{Bmatrix} \tag{A.16}$$

A graphical representation of this concept can be seen in figure A.5(b).

However, this definition is changed in some references. In particular, the mean argument of latitude λ is sometimes substituted by the modified relative mean longitude, which is defined as:

$$\delta l_0 \equiv \delta\lambda + \delta\Omega\cos i$$

The advantage of this formulation is that the in-plane motion becomes decoupled. This will also be done in the following set of relative elements.

A.3.2.2 C set of relative orbital elements (CROE).

Defined by Peters & Noomen in [37], this set is also closely related with the orbit safety notion. It arises from the analysis of the Gauss Variational Equations (GVEs) applied to the relative dynamics between a deputy and a chief spacecraft, when the former performs a cotangential

transfer. The elements themselves are defined as:

$$\begin{cases} C_1 = \delta p = \eta^2 \delta a - 2\, a\, e \delta e & \equiv & \text{Relative parameter of the orbit} & [L] \\ C_2 = e\delta p - p\delta e & \equiv & \textit{unclear physical meaning} & [L] \\ C_3 = -e\, p\, (\delta\omega + \cos i \delta\Omega) & \equiv & \textit{unclear physical meaning} & [L] \\ C_4 = a\left(\delta\omega + \cos i\delta\Omega + \eta^{-1}\delta M\right) & \equiv & \text{Modified relative mean longitude} & [L] \\ C_5 = -p\,(\cos\omega\delta i + \sin i \sin\omega\delta\Omega) & \equiv & \textit{unclear physical meaning} & [L] \\ C_6 = p\,(\sin\omega\delta i - \sin i \cos\omega\delta\Omega) & \equiv & \textit{unclear physical meaning} & [L] \end{cases} \tag{A.17}$$

For a proper geometrical and conceptual description of the elements, please see [37]. As an introduction, the first four elements essentially determine the in-plane relative motion. C_1, C_2 & C_3 arise from a very intelligent interpretation of the GVEs, with C_4 completing the element set. On the other hand, elements C_5 and C_6 describe the out-of-plane motion.

Bibliography

[1] P. Bijlaard. "On the Buckling of Stringer Panels Including Forced Crippling". In: *Journal of the Aeronautical Sciences* 22.7 (1955), pp. 491–501 (cit. on p. 1).

[2] F. P. Preparata and M. I. Shamos. "Convex Hulls: Basic Algorithms". In: *Computational Geometry: An Introduction*. New York, NY: Springer New York, 1985, pp. 95–149. ISBN: 978-1-4612-1098-6. DOI: 10.1007/978-1-4612-1098-6_3. URL: https://doi.org/10.1007/978-1-4612-1098-6_3 (cit. on p. 2).

[3] D. Barrett et al. "Measuring abstract reasoning in neural networks". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by J. Dy and A. Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, Oct. 2018, pp. 511–520. URL: https://proceedings.mlr.press/v80/barrett18a.html (cit. on p. 3).

[4] B. M. Lake and M. Baroni. "Still not systematic after all these years: On the compositional skills of sequence-to-sequence recurrent networks". In: *CoRR* abs/1711.00350 (2017). arXiv: 1711.00350. URL: http://arxiv.org/abs/1711.00350 (cit. on p. 3).

[5] D. Saxton et al. "Analysing Mathematical Reasoning Abilities of Neural Models". In: *CoRR* abs/1904.01557 (2019). arXiv: 1904.01557. URL: http://arxiv.org/abs/1904.01557 (cit. on p. 3).

[6] T. Ebert, J. Belz, and O. Nelles. "Interpolation and extrapolation: Comparison of definitions and survey of algorithms for convex and concave hulls". In: *2014 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*. IEEE. 2014, pp. 310–314 (cit. on p. 3).

[7] W.-Y. Loh, C.-W. Chen, and W. Zheng. "Extrapolation errors in linear model trees". In: *ACM Transactions on Knowledge Discovery from Data (TKDD)* 1.2 (2007), 6–es (cit. on p. 3).

[8] P. Klesk. "Construction of a Neurofuzzy Network Capable of Extrapolating (and Interpolating) With Respect to the Convex Hull of a Set of Input Samples in R". In: *IEEE Transactions on Fuzzy Systems* 16.5 (2008), pp. 1161–1179. DOI: 10.1109/TFUZZ.2008.924337 (cit. on p. 3).

[9] R. Balestriero, J. Pesenti, and Y. LeCun. "Learning in high dimension always amounts to extrapolation". In: *arXiv preprint arXiv:2110.09485* (2021) (cit. on pp. 3, 4, 10).

[10] S. Marsland. *Machine Learning: An Algorithmic Perspective*. 2nd ed. Boca Raton, USA: Chapman & Hall/CRC, 2015 (cit. on pp. 3, 7).

[11] I. Bárány and Z. Füredi. "On the shape of the convex hull of random points". In: *Probability theory and related fields* 77 (1988), pp. 231–240 (cit. on p. 3).

[12] L. Bonnasse-Gahot. "Interpolation, extrapolation, and local generalization in common neural networks". In: *arXiv preprint arXiv:2207.08648* (2022) (cit. on pp. 4, 10, 13).

[13] H. Hotelling. "Analysis of a complex of statistical variables into principal components." In: *Journal of educational psychology* 24.6 (1933), p. 417 (cit. on p. 9).

[14] B. Rosner and D. Grove. "Use of the Mann–Whitney U-test for clustered data". In: *Statistics in medicine* 18.11 (1999), pp. 1387–1400 (cit. on p. 13).

[15] R. Velez Ibarrola and A. Garcia Perez. *Calculo de probabilidades y Estadistica Matematica*. 1st ed. Madrid, Spain: Universidad Nacional de Educacion a Distancia, 1994 (cit. on p. 15).

[16] D. Zhang. "A coefficient of determination for generalized linear models". In: *The American Statistician* 71.4 (2017), pp. 310–316 (cit. on p. 20).

[17] G. Chen, J. R. Gott, and B. Ratra. "Non-Gaussian Error Distribution of Hubble Constant Measurements". In: *Publications of the Astronomical Society of the Pacific* 115.813 (2003), p. 1269 (cit. on p. 22).

[18] P. Pernot, B. Huang, and A. Savin. "Impact of non-normal error distributions on the benchmarking and ranking of Quantum Machine Learning models". In: *Machine Learning: Science and Technology* 1.3 (2020), p. 035011 (cit. on p. 22).

[19] D. Smyl et al. "Learning and correcting non-Gaussian model errors". In: *Journal of Computational Physics* 432 (2021), p. 110152 (cit. on p. 22).

[20] L. Chai et al. "Using generalized Gaussian distributions to improve regression error modeling for deep learning-based speech enhancement". In: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 27.12 (2019), pp. 1919–1931 (cit. on p. 22).

[21] B. Efron. "Bootstrap methods: another look at the jackknife". In: *Breakthroughs in statistics: Methodology and distribution*. Springer, 1992, pp. 569–593 (cit. on p. 22).

[22] E. B. Wilson. "Probable inference, the law of succession, and statistical inference". In: *Journal of the American Statistical Association* 22.158 (1927), pp. 209–212 (cit. on p. 29).

[23] J. D. Jobson. *Applied multivariate data analysis: regression and experimental design*. Springer Science & Business Media, 2012 (cit. on p. 23).

[24] O. Montenbruck and E. Gill. *Satellite Orbits: Models, Methods and Applications*. 1st ed. Wessling, Germany: Springer, 2001 (cit. on p. 36).

[25] K. Yamanaka and F. Ankersen. "New State Transition Matrix for Relative Motion on an Arbitrary Elliptical Orbit". In: *Journal of Guidance, Control & Dynamics* 25.1 (2002), pp. 60–66. DOI: 10.2514/2.4875 (cit. on p. 37).

[26] D.-W. Gim and K. Alfriend. "State Transition Matrix of Relative Motion for the Perturbed Noncircular Reference Orbit". In: *Journal of Guidance Control and Dynamics* 26 (Nov. 2003), pp. 956–971. DOI: 10.2514/2.6924 (cit. on p. 37).

[27] S.-S. Exchange. *Practical Uses of an STM*. 2019. URL: https://space.stackexchange.com/questions/32916 (cit. on p. 38).

[28] ESA. *Precise Orbit Determination*. 2011. URL: https://gssc.esa.int/navipedia/index.php/Precise_Orbit_Determination (cit. on p. 38).

[29] C. Chao and F. Hoots. *Applied Orbit Perturbation and Maintenance*. Aerospace Press, 2018. ISBN: 9781523123346 (cit. on p. 43).

[30] W. E. Wiesel. *Modern Astrodynamics*. 2nd ed. Beavercreek, Ohio: Aphelion Press, 2010 (cit. on p. 47).

[31] M. Eckstein, C. Rajasingh, and P. Blumer. "Colocation Strategy and Collision Avoidance for the Geostationary Satellites at 19 Degrees West". In: *CNES International Symposium on Space Dynamics*. Vol. 25. Oberpfaffenhofen, Germany: DLR GSOC, Nov. 1989, pp. 60–66 (cit. on p. 48).

[32] S. D'Amico and O. Montenbruck. "Proximity Operations of Formation-Flying Spacecraft Using an Eccentricity/Inclination Vector Separation". In: *Journal of Guidance, Control & Dynamics* 29.3 (2006), pp. 554–563. DOI: 10.2514/1.15114 (cit. on pp. 48, 54, 55).

[33] D.-W. Gim and K. T. Alfriend. "Satellite Relative Motion Using Differential Equinoctial Elements". In: *Celestial Mechanics and Dynamical Astronomy* 92.4 (2005), pp. 295–336. DOI: 10.1007/s10569-004-1799-0 (cit. on pp. 49, 50).

[34] S. D'Amico. *Relative Orbital Elements as Integration Constants of Hill's Equations*. TN 05-08. Oberpfaffenhofen, Germany: Deutsches Zentrum für Luft- und Raumfahrt (DLR), 2005 (cit. on pp. 49, 54).

[35] H. Schaub. "Relative Orbit Geometry Through Classical Orbit Element Differences". In: *Journal of Guidance, Control & Dynamics* 27.5 (2004), pp. 839–848. DOI: 10.2514/1.12595 (cit. on pp. 49, 54).

[36] G. Gaias, C. Colombo, and M. Lara. "Accurate Osculating/Mean Orbital Elements Conversions for Spaceborne Formation Flying". In: (Feb. 2018). https://www.researchgate.net/publication/340378956_Accurate_OsculatingMean_Orbital_Elements_Conversions_for_Spaceborne_Formation_Flying (cit. on p. 50).

[37] T. Vincent Peters and R. Noomen. "Linear Cotangential Transfers and Safe Orbits for Elliptic Orbit Rendezvous". In: *Journal of Guidance, Control & Dynamics* 44.4 (2021), pp. 732–748. DOI: 10.2514/1.G005152 (cit. on pp. 56, 57).

[38] R. H. Lopes, I. Reid, and P. R. Hobson. "The two-dimensional Kolmogorov-Smirnov test". In: (2007).

[39] G. Schoups and J. A. Vrugt. "A formal likelihood function for parameter and predictive inference of hydrologic models with correlated, heteroscedastic, and non-Gaussian errors". In: *Water Resources Research* 46.10 (2010).

[40] B. D. Tapley, B. E. Schutz, and G. H. Born. *Statistical Orbit Determination*. 1st ed. Amsterdam, Netherlands: Elsevier, 2004.

[41] K. T. Alfriend and Srinivas. *Spacecraft Formation Flying*. 1st ed. Oxford, United Kingdom: Elsevier, 2010.

[42] R. H. Battin. *An Introduction to the Mathematics and Methods of Astrodynamics, Revised Edition*. 1st ed. Reston, Virginia: American Institute of Aeronautics and Astronautics, 1999.

[43] D. Brouwer and G. M-Clemence. *Methods of Celestial Mechanics*. Brackley Square House, London: Academic Press, 1961.

[44] H. Schaub and J. L. Junkins. *Analytical Mechanics of Space Systems*. Reston, VA: AIAA Education Series, Oct. 2003. DOI: 10.2514/4.861550.

[45] W. Fehse. *Automated Rendezvous and Docking of Spacecraft*. Cambridge Aerospace Series. Cambridge: Cambridge University Press, 2003. DOI: 10.1017/CBO9780511543388.

[46] P. K. S. Dennis D. McCarthy. *Time: From Earth Rotation to Atomic Physics*. Wiley-VCH, 2009. ISBN: 3527407804; 9783527407804.

[47] K. Wakker. *Fundamentals of Astrodynamics*. Jan. 2015. ISBN: 978-94-6186-419-2.

[48] A. H. Nayfeh. *Perturbation Methods*. Weinheim, Germany: Wiley-VCH, 2004.

[49] W. M. Kaula. *Theory of Satellite Geodesy*. Mineola, New York: Dover Publications, 2013.

[50] F. Tisserand. *Traité de Mécanique Céleste*. Vol. 1. Paris, 1889.

[51] J. Sullivan, S. Grimberg, and S. D'Amico. "Comprehensive Survey and Assessment of Spacecraft Relative Motion Dynamics Models". In: *Journal of Guidance, Control & Dynamics* 40.8 (2017), pp. 1837–1859. DOI: 10.2514/1.G002309.

[52] H. Schaub and K. T. Alfriedn. "Hybrid Cartesian and Orbit Element Feedback Law for Formation Flying Spacecraft". In: *Journal of Guidance, Control & Dynamics* 25.2 (2002), pp. 387–393. DOI: 10.2514/2.4893.

[53] N. Capitaine. "The Celestial Pole Coordinates". In: *Celestial Mechanics and Dynamical Astronomy* 48 (1990), pp. 127–143.

[54] D. D. McCarthy. *IERS Conventions (1992)*. 21. Paris, France: Central Bureau of IERS - Observatoire de Paris, 1996.

[55] J. Williams. *LVLH Transformations*. https://degenerateconic.com/uploads/2015/03/lvlh.pdf. 2014.

[56] H. Fiedler. *Analysis of TerraSAR-L Cartwheel Constellations*. https://elib.dlr.de/22345/. Oberpfaffenhofen, Germany: Deutsches Zentrum für Luft- und Raumfahrt (DLR), Nov. 2003.

[57] G. W. Hill. "Researches in the Lunar Theory". In: *American Journal of Mathematics* 1.2 (1878), pp. 129–147. DOI: 10.2307/2369304.

[58] W. H. Clohessy and R. S. Wiltshire. "Terminal Guidance System for Satellite Rendezvous". In: *Journal of the Aerospace Sciences* 27.9 (1960), pp. 653–658. DOI: 10.2514/8.8704.

[59] R. Broucke. "On the Matrizant of the Two-Body Problem". In: *Astronomy and Astrophysics* 6 (June 1970), p. 173.

[60] J. Tschauner and P. Hempel. “Optimale Beschleunigungsprogramme fÃ¼r das Rendezvous-Manöver”. In: *Astronautica Acta* 10 (1964), pp. 296–307.

[61] T. Carter. “State Transition Matrices for Terminal Rendezvous Studies: Brief Survey and New Example”. In: *Journal of Guidance Control and Dynamics* 21 (Jan. 1998), pp. 148–155. DOI: 10.2514/2.4211.

[62] O. Montenbruck, M. Kirschner, and S. D’Amico. “E/I-vector separation for safe switching of the GRACE formation”. In: *Aerospace Science and Technology* 10 (2006), pp. 628–635. DOI: 10.1016/j.ast.2006.04.001.

[63] G. Gaias, J.-S. Ardaens, and C. Colombo. “Precise line-of-sight modelling for angles-only relative navigation”. In: *Advances in Space Research* 67.11 (2021). Satellite Constellations and Formation Flying, pp. 3515–3526. DOI: 10.1016/j.asr.2020.05.048.

[64] G. Gaias, C. Colombo, and M. Lara. “Analytical Framework for Precise Relative Motion in Low Earth Orbits”. In: *Journal of Guidance, Control, and Dynamics* 43.5 (2020), pp. 915–927. DOI: 10.2514/1.G004716.

[65] D. Brouwer. “Solution of the problem of artificial satellite theory without drag”. In: *Astronomical Journal* 64.5 (Nov. 1959), p. 378. DOI: 10.1086/107958.

[66] R. H. Lyddane. “Small eccentricities or inclinations in the Brouwer theory of the artificial satellite”. In: *Astronomical Journal* 68 (Oct. 1963), p. 555. DOI: 10.1086/109179.

[67] A. Deprit. “Canonical transformations depending on a small parameter”. In: *Celestial Mechanics* 1.1 (Mar. 1969), pp. 12–30. DOI: 10.1007/BF01230629.

[68] A. Deprit. “Delaunay Normalisations”. In: *Celestial Mechanics* 26.1 (Jan. 1982), pp. 9–21. DOI: 10.1007/BF01233178.

[69] G. Hori. “Theory of General Perturbation with Unspecified Canonical Variable”. In: *Publications of the Astronomical Society of Japan* 18 (Jan. 1966), pp. 287–296.

[70] Y. Chihabi and S. Ulrich. “Spacecraft Formation Guidance Law using a State Transition Matrix With Gravitational, Drag and Third-Body Perturbations”. In: Jan. 2020. DOI: 10.2514/6.2020-1460.

[71] G. Gaias, J. Ardaens, and O. Montenbruck. “Model of J2 Perturbed Satellite Relative Motion with Time-Varying Differential Drag”. In: *Celestial Mechanics and Dynamical Astronomy* (2015).

[72] A. Koenig, T. Guffanti, and S. D'Amico. "New State Transition Matrices for Relative Motion of Spacecraft Formations in Perturbed Orbits". In: Sept. 2016. DOI: 10.2514/6.2016-5635.

[73] A. Biria and R. Russell. "A Satellite Relative Motion Model Including J2 and J3 via Vinti's Intermediary". In: Feb. 2016.

[74] K. Alfriend and H. Yan. "An Orbital Elements Based Approach to the Nonlinear Formation Flying Problem". In: Toulouse, France, Feb. 2016.

[75] Mathworks. *Convert complex diagonal form into real diagonal form*. 2022. URL: https://www.mathworks.com/help/matlab/ref/cdf2rdf.html.

[76] M. R. Delgado. *Lecture notes: Basics of Orbital Mechanics I*. Apr. 2008.

[77] F. G. Nievinski. *subtightplot*. https://www.mathworks.com/matlabcentral/fileexchange/39664-subtightplot. 2013.

[78] J. C. Lansey. *linspecer*. https://www.mathworks.com/matlabcentral/fileexchange/42673-beautiful-and-distinguishable-line-colors-colormap. 2015.

[79] Jan. *WindowAPI*. https://www.mathworks.com/matlabcentral/fileexchange/31437-windowapi. 2013.

[80] T. Davis. *Arrow3*. https://www.mathworks.com/matlabcentral/fileexchange/14056-arrow3. 2022.

[81] E. Duenisch. *latexTable*. https://www.mathworks.com/matlabcentral/fileexchange/44274-latextable. 2016.