Machine Learning for the prediction of aeronautical loads and stress

Edouard Fournier

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *14/10/2019* **par :**

Edouard FOURNIER

Méthodes d'apprentissage statistique pour la prédiction de charges et de contraintes aéronautiques.

JURY

Jean-Marc Bardet	Professeur, Univ. Paris I	Rapporteur
Christian Bes	Professeur, Univ. Toulouse III	Membre du Jury
Jean-François Dupuy	Professeur, INSA Rennes	Rapporteur
Fabrice Gamboa	Professeur, Univ. Toulouse III	Co-directeur de Thèse
Anne Gegout-Petit	Professeure, Univ. de Lorraine	Membre du Jury
Stéphane Grihon	Ingénieur R&D Airbus	Invité
Céline Helbert	Maîtresse de conférences, EC Lyon	Membre du Jury
Thierry Klein	Professeur, ENAC	Directeur de Thèse
Béatrice Laurent-Bonneau	Professeure, INSA Toulouse	Membre du Jury

---

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées

Unité de Recherche :
Institut de Mathématiques de Toulouse (UMR 5219)

Directeur(s) de Thèse :
Prof. Thierry KLEIN (directeur de thèse), *Prof. Fabrice GAMBOA (co-directeur de thèse)* et *Dr. Stéphane GRIHON (superviseur industriel)*

Rapporteurs :
Prof. Jean-Marc BARDET et *Prof. Jean-François DUPUY*

THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *14/10/2019* **par :**

Edouard FOURNIER

Méthodes d'apprentissage statistique pour la prédiction de charges et de contraintes aéronautiques.

JURY

Jean-Marc Bardet	Professeur, Univ. Paris I	Rapporteur
Christian Bes	Professeur, Univ. Toulouse III	Membre du Jury
Jean-François Dupuy	Professeur, INSA Rennes	Rapporteur
Fabrice Gamboa	Professeur, Univ. Toulouse III	Co-directeur de Thèse
Anne Gegout-Petit	Professeure, Univ. de Lorraine	Membre du Jury
Stéphane Grihon	Ingénieur R&D Airbus	Invité
Céline Helbert	Maîtresse de conférences, EC Lyon	Membre du Jury
Thierry Klein	Professeur, ENAC	Directeur de Thèse
Béatrice Laurent-Bonneau	Professeure, INSA Toulouse	Membre du Jury

---

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées

Unité de Recherche :
Institut de Mathématiques de Toulouse (UMR 5219)

Directeur(s) de Thèse :
Prof. Thierry KLEIN (directeur de thèse), *Prof. Fabrice GAMBOA (co-directeur de thèse)* et *Dr. Stéphane GRIHON (superviseur industriel)*

Rapporteurs :
Prof. Jean-Marc BARDET et *Prof. Jean-François DUPUY*

"All theories are legitimate, no matter.
What matters is what you do with them."

Jorge Luis Borges

Remerciements

Mes premiers remerciements vont à Thierry Klein, Fabrice Gamboa, Christian Bes ainsi qu'à Stéphane Grihon. Je ne pourrais pas vous remercier assez de vos conseils, de votre disponibilité, de vos encouragements, de vos relectures, de vos connaissances, de votre bienveillance et de votre bonne humeur sans lesquels cette thèse n'aurait jamais pu aboutir.

Je remercie très sincèrement Jean-Marc Bardet et Jean-François Dupuy d'avoir accepté d'être les rapporteurs de cette thèse. Je remercie également Béatrice Laurent-Bonneau, Céline Helbert et Anne Gégout-Petit d'avoir accepté de faire partie du jury de thèse.

J'ai eu beaucoup de chance de travailler au sein d'Airbus Operations. Ces trois dernières années ont été très riches aussi bien techniquement qu'humainement rendant quasi impossible une énumération complète de toutes les personnes que j'ai pu croiser. Ainsi, je tiens à remercier tous les individus aussi passionnants que passionnés avec qui j'ai pu échanger et qui ont su m'apprivoiser !

Je remercie tout particulièrement Célia, David, Thomas, Lucas, Dimitri et Julien qui me supportent (dans tous les sens du terme) depuis maintenant de nombreuses années.

Je suis infiniment reconnaissant envers l'ensemble de ma famille, en particulier mes parents qui ont toujours soutenu mes choix et qui m'ont donné la chance de pouvoir poursuivre les études que je voulais. Un grand merci à Charles qui reste toujours présent malgré nos querelles fraternelles, à Brigitte et Michel qui me guident depuis que je suis petit, à Sylvain de m'avoir écouté pendant tous ces déjeuners et à Anne-Charlotte d'être venue encourager son numéro 5 !

Pour finir, je remercie Lucie qui, par sa présence et son énergie, a rendu le tout beaucoup plus facile. Merci pour tous ces moments de réconfort, de joie et de découverte qui m'ont permis d'avancer et d'aller jusqu'au bout.

Table des matières

Table des figures

Liste des tableaux

Lexique aéronautique

- **Aircraft variant**: this is an adaptation (slight variation) of an initial aircraft. Due to the evolution of market needs, manufacturer must adapt an aircraft by increasing the range or the number of passengers for example.
- **Weight variant**: this is a weight adaptation of an initial aircraft. The weight aircraft variant has a different maximum take off weight than the initial aircraft but belongs to the same family.
- **Maximum take off weight (MTOW)**: is the maximum weight allowed for an aircraft to take off.
- **External loads**: are external forces applied on the structure and computed by a numerical code reflecting the physical behavior of the aircraft. Six quantities represent the external loads: Mx, My, Mz (the moments) and Tx, Ty, Tz (the shear forces).
- **Load case**: configuration case (speed, altitude, fuel mass, center of inertia, maximum take off weight, etc.) in which the external loads are computed.
- **Bending moment of a wing (Mx)**: is the reaction of the wing under a force which makes it bends.
- **Stress analysis**: this analysis concerns the computations relative to mechanical strength. For millions of load cases, a numerical code calculates if the structural elements withstand mechanical constraints (in compression, tension, etc.).
- **Gust**: or wind gust, is a short increase of the wind speed. Characterized by its wavelength, it has an impact on the aircraft similarly to a road bump on a car.

Introduction

This thesis is the result of a Ph.D. study carried out in a CIFRE framework between Airbus Operations SAS and the University of Toulouse 3 with the Toulouse Institute of Mathematics. Airbus has the objective of reducing simulator time for sizing an aircraft structure. First, using Machine Learning algorithms, we build metamodel to replace costly and time consuming calculation modules of aeronautical forces (loads) suffered by the structure. Secondly, from the study of the data, we provide a new statistical model to aeronautical engineers.

This thesis is composed of five chapters. The first chapter is an extended résumé of the thesis. The second chapter is a bibliographical study of the different algorithms and methods used. In the first part of Chapter 2, we review regression methods. In particular, we focus first on nonparametric methods using local polynomials [5]. Since this method has difficulties to scale up in large dimension, we focus on regression trees based algorithms. We detail Classification and Regression Trees [4] as well as classical associated methods (Random Forest [3], Bagging [1], Gradient Boosting [2], etc.). Finally, we focus on a sparse parametric regression method for high dimensional data: the Greedy algorithm [11]. In the second part, we give an overview of regression models with operators of deformation. In this framework, we want to register a collection of curves to a reference curve. We detail two classical models: the so called Shape Invariant Model (SIM) [10] and the so called Dynamic Time Warping [12]. The first one is a parametrical model where we consider operators of deformation that are affine functions acting independently on the input space and the output space. The second one is nonparametric and acts on the input space of the curve by conserving the order and by aligning peaks and valleys of curves to be registered.

The Chapter 3 is a preliminary test case study on real aeronautical data. In this study, the aim was to approximate the numerical code computing aeronautical loads. This study has been realized during a sprint project within Airbus. It has been shown in a previous project that regression trees work well for predicting loads when the maximum take off weight

of the aircraft has been fixed. Here, we are interested in predicting loads in an extrapolation case of maximum take off weight. We have several datasets corresponding to four different weight variants: the idea is to build a model from one dataset and predict the three other cases. The tested regression methods are regression-trees based models. In addition, we compare dimension reduction techniques and we try to quantify their influence on extrapolation performances of the tested regression methods. It appears that AdaBoost [9] associated to Random Forest with a Principal Component Analysis on the outputs offers good performances in score and computing time. This chapter corresponds to the published article [8].

The study in Chapter 3 led us to consider a semiparametric model that we describe in Chapter 4. Aeronautical loads are represented by a collection of curves having a similar shape. As a consequence, it is natural to think that they come from the deformation of a reference curve. More precisely, we consider that the collection derives from a rotation and a homothetic transformation applied to the reference curve. Here, we develop a new deformation model acting simultaneously on the input space and output space of the curves. We estimate the deformation parameters through a $M-$estimation procedure and we give the consistency and the asymptotic normality of the estimators. We complete this study with two numerical applications. We will show that our methodology is the most effective when the ratio signal on noise is good. Besides, results obtained indicate that our model is appropriate to the prediction of aeronautical loads. This chapter corresponds to the published article [7].

Chapter 5 is dedicated to the estimation of preliminary maximum aeronautical loads. These loads are critical for sizing properly an aircraft structure but numerous simulations are required to produce them. From the database of an initial aircraft, the idea is to build a model to predict the critical loads of two weight variants. The framework is a sparse polynomial regression based on a Taylor development of second order. The model is built thanks to the Greedy algorithm and the results obtained are very satisfying allowing a quick pre-sizing of the structure. This chapter corresponds to the published engineering note [6].

Bibliography

[1] Breiman, L.: Bagging predictors. Machine Learning **24**, 123–140 (1996)

[2] Breiman, L.: Arcing the edge. Technical Report (486) (1997). Statistics Department, University of California

[3] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)

[4] Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: Classification and Regression Trees. Wadsworth, Belmont, CA (1984)

[5] Cleveland, W.S.: Robust locally weighted regression and smoothing scatterplots. Journal of the American statistical association **74**(368), 829–836 (1979)

[6] Fournier, E., Grihon, S., Bes, C., Klein, T.: Prediction of preliminary maximum wing bending moments under discrete gust. Journal of Aircraft **56**(4), 1722–1725 (2019)

[7] Fournier, E., Grihon, S., Klein, T.: Semiparametric estimation of plane similarities: application to fast computation of aeronautic loads. Statistics **53**(5), 1168–1186 (2019)

[8] Fournier, E., Klein, T., Grihon, S.: A case study: Influence of dimension reduction on regression trees-based algorithms-predicting aeronautics loads of a derivative aircraft. Journal de la Société Française de Statistique **159**(3), 56–78 (2018)

[9] Freund, Y., Shapire, R.: A decision-theoretic generalization of on-line learning and application to boosting. Proceedings of the second European Conference on Computational Learning Theory pp. 23–37 (1995)

[10] Lawton, W., Sylvestre, E., Maggio, M.: Self modeling nonlinear regression. Technometrics **14**(3), 513–532 (1972)

[11] Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on signal processing **41**(12), 3397–3415 (1993)

[12] Ramsay, J.O., Li, X.: Curve registration. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **60**(2), 351–363 (1998)

Chapter 1

Résumé détaillé de la thèse

1.1 Introduction

La structure d'un avion est un système complexe dont la conception implique de nombreuses simulations générant alors une quantité astronomique de données. Le cas du calcul de charges et de contraintes aéronautiques pour un avion n'échappe pas à la règle. Il s'agit en fait du bilan des forces et des contraintes mécaniques que subit la structure. Ce processus peut être représenté comme suit:

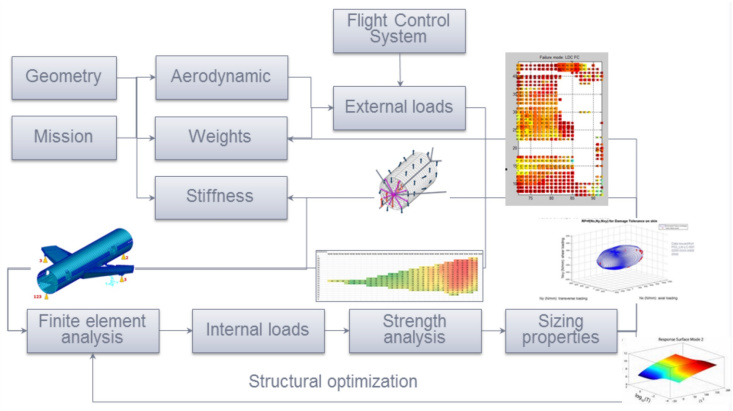


Figure 1.1: Organigramme du processus de calcul des charges et des contraintes aéronautiques

Le processus général, exposé en Figure 1.1, est exécuté massivement pour identifier les cas de charges (i.e une configuration aéronautique : manoeuvres, vitesse, chargement, épaisseur,...) qui sont critiques en termes de stress enduré par la structure et, bien sûr, les paramètres qui les rendent critiques. Le but final de ce processus est donc de dimensionner et de

concevoir la meilleure structure possible. Typiquement, pour une structure d'avion, des millions de cas de charges sont générés, et pour chacun d'entre eux sont également générés des millions de réponses structurelles (i.e comment les éléments de la structure réagissent sous de telles conditions). Le processus d'identification des charges nécessite donc la mise en oeuvre de simulations numériques massives et il est naturel de penser que des méthodes récentes de machines learning pourraient contribuer à réduire le nombre de simulations nécessaires

Pour un avion, donnons quelques ordres de grandeur en termes de quantité de données produites. Par exemple, les charges externes produites occupent environ 10^6 octets et les éléments structuraux et leur masse concernent 10^4 octets. Les facteurs de réserve (c'est-à-dire les quantités ressortant de l'analyse stress) occupent quant à eux 10^{12} octets. En somme, nous atteignons facilement 10^{18} à 10^{21} octets pour un seul avion.

Dans une optique d'amélioration des méthodes de travail et des outils, Airbus a beaucoup investi dans sa transformation digitale et le développement d'infrastructure permettant de traiter les données existantes ou nouvellement produites. Comme souligné par [11], bien que ces techniques soient utilisées dans de nombreux domaines tels que Internet et la Business Intelligence, l'industrie aéronautique peut également en tirer profit. En utilisant ces techniques dans notre contexte, Airbus a pour objectif principal de réduire le temps de calcul et la charge mémoire pour dimensionner une structure.

Nos travaux ont porté sur les points suivants :

- Exploiter et adapter des algorithmes d'apprentissage automatique du processus de calcul : nous cherchons à remplacer certains modules de la chaîne qui sont coûteux en temps et ressources en particulier les charges externes et l'analyse stress;
- Valoriser les données par l'extraction d'informations pour les experts métiers: typiquement, donner de nouveaux outils quantitatifs et visuels pour représenter les courbes de charge (les forces aéronautiques subies par la structure). Extraire de l'information et créer de la valeur des données pour les experts métiers;

Le manuscrit est composé de quatre chapitres. Le Chapitre 2 est une revue bibliographique des différents algorithmes et méthodes exploitées dans les chapitres suivants. Ce premier chapitre comporte deux grands volets. Une première partie concerne *les algorithmes de régression.* Nous nous intéressons en premier lieu aux modèles de régression non-paramétriques. En particulier, nous nous penchons sur les polynômes locaux [7] qui sont

une généralisation de l'estimateur de Nadaraya-Watson. Néanmoins, cette méthode de régression subit le fléau de la dimension et peine à passer en grande dimension. Par conséquent, nous regardons par la suite les méthodes de régression à base d'arbres. En particulier, nous détaillons le modèle CART [6] (Classification And Regression Tree) ainsi que les méthodes d'agrégation et d'échantillonnage en découlant (Random Forest [5], Bagging [3], etc.). Enfin, il sera question d'une méthode paramétrique parcimonieuse de régression pour la grande dimension : le modèle *Greedy* [10]. Le second volet de notre revue bibliographique fait quant à lui un tour d'horizon des *modèles de régression avec opérateurs de déformation*. Dans ce cadre, nous avons à notre disposition une collection de courbes que nous cherchons à recaler sur une courbe de référence. Cette dernière peut être connue sur tout son ensemble de définition, ou seulement sur un échantillon de points (une grille aléatoire), et peut être bruitée. Nous discutons ici de deux modèles classiques. Le premier est le Shape Invariant Model [9] (SIM) et est paramétrique. Les opérateurs de déformation sont des fonctions affines agissant indépendamment sur l'espace des entrées et des sorties des courbes. Le deuxième modèle est non paramétrique et nous rappelons la méthode générale appelée le Dynamic Time Warping [13]. Celle méthode agit sur les entrées des courbes en conservant l'ordre et en alignant ainsi les pics et les creux des courbes à recaler.

Le Chapitre 3 donne un panorama exploratoire des données et modèles qui seront explorés dans la suite de la thèse. Il s'agit d'une étude préliminaire où nous cherchons à approcher la chaîne de calcul des charges externes (cf Figure 1.1) à partir de données déjà existantes. Cette étude est issue d'un projet *sprint* interne à Airbus. Il a été montré dans un projet précédent que les techniques de régression à base d'arbres fonctionnent bien pour prédire les charges d'un avion dans le cas iso-masse (c'est-à-dire que la masse maximale au décollage est fixée). Ici, nous nous intéressons à la prédiction des charges dans le cas où la masse maximale au décollage varie. Plus précisément, nous cherchons à extrapoler les charges. Pour cela, supposons que nous ayons à notre disposition un jeu de données correspondant à un avion ayant une masse maximale au décollage $M0$ et un jeu de données correspondant à une masse maximale au décollage $M1$. L'idée est de construire un modèle à partir des données de l'avion $M0$, et de prédire les charges de l'avion $M1$. Ayant à notre disposition plusieurs jeux de données correspondant chacun à une masse maximale au décollage différente, nous pouvons ainsi évaluer la performance en extrapolation des modèles. Les modèles testés sont des modèles à base d'arbres. Par ailleurs, nous avons aussi comparé des techniques de réduction de dimension en essayant de quantifier l'influence de ces méthodes sur les performances en extrapolation en régression.

L'étude descriptive menée au Chapite 3 nous a conduit à un modèle semi-paramétrique que nous décrivons au Chapitre 4. Les charges sont représentées par des courbes ayant une forme régulière. Il est naturel de faire l'hypothèse que celles-ci proviennent donc de la déformation d'une courbe de référence. Plus précisément sous cette hypothèse, nous supposons que les courbes ont été obtenues par une rotation et une homothétie appliquées à la courbe mère (courbe de référence). Ici, nous développons un nouveau modèle de déformation de courbes de charges qui sera intégré dans le processus de modélisation. Plus précisément, les opérateurs de déformation agissent simultanément sur l'espace des entrées et l'espace des sorties des courbes. Nous mettons en place une procédure de M-estimation des paramètres du modèle. Nous étudions la consistence et la normalité asymptotique des estimateurs des paramètres du modèle. Nous considérons deux cadres : le premier correspond au cas où la courbe de référence est supposée non bruitée et connue sur son espace de définition. Ces premiers résultats serviront à montrer le second cas, plus réaliste, où la courbe de référence est définie sur la même grille aléatoire que la collection de courbes à recaler.

Le Chapitre 5 sera consacré à l'estimation préliminaire de charges maximales aéronautiques. Ces dernières sont déterminantes et critiques dans le dimensionnement d'une structure. Un nombre important de simulations est nécessaire pour produire ces charges maximales. A partir d'une base de données d'un avion initial, l'idée est de construire un métamodèle pour prédire les charges maximales de deux avions dérivés. En effet, toutes les charges produites par les simulations ne sont pas utiles. Il est donc naturel de se concentrer sur les charges les plus critiques que subit la structure. Nous mettons en place une méthode rapide de modélisation. Le cadre proposé est celui d'une régression polynomiale sparse obtenue par un développement de Taylor au second ordre. Le modèle est construit à l'aide d'une méthode de régression Greedy. Nous montrons en particulier que notre méthode donne une estimation préliminaire très performante des charges maximales subies par l'aile de deux avions dérivés. Cela permet ainsi un pré-dimensionnement rapide de la structure.

Donnons une rapide synthèse des principaux résultats obtenus dans les Chapitres 3, 4 et 5.

1.2 Etude de cas : Influence des réductions de dimension sur les modèles d'apprentissage d'arbres de régression - Prédiction de courbes de charges aéronautiques pour un avion dérivé

Dans l'industrie aéronautique, les besoins du marchés évoluent rapidement et la concurrence faire rage. Ceci implique la modification perpétuelle d'un avion en un minimum de temps - par exemple, l'augmentation du nombre de passagers avec la famille de l'A330 [1] - ce qui requiert alors un re-dimensionnement récurrent et coûteux de la structure. Dans notre étude, les modifications à apporter à l'avion concernent sa masse maximale au décollage (MTOW - maximum take off weight en anglais).

Dans un projet interne à Airbus ayant pour but de démontrer l'utilité des outils statistiques et de Machine Learning, il a été montré que la famille des arbres de régression [6] fonctionne bien pour prédire les charges aéronautiques dans le cas iso-masse. Notre étude, réalisée dans le cadre d'un projet *Sprint Proof of Value*, a pour but de quantifier la performance de ces algorithmes lorsque la masse maximale au décollage varie. Les charges concernées sont les moments de flexion que subit l'aile de l'avion. Rappelons que le moment de flexion est la réaction induite dans l'élément structurel quand une force extérieure appliquée à cet élément le fait fléchir. Ces charges sont obtenues à partir d'un code numérique traduisant la physique du problème. En d'autres termes, le code numérique, représenté par une fonction f prend en entrée des paramètres avions x tels que la vitesse, l'altitude, la quantité de carburant, etc. et donne en sortie y le moment de flexion associé subi par l'aile de l'avion, i.e $y = f(x)$.

Supposons que nous ayons à notre disposition les données (entrées + sorties) d'un avion ayant une masse maximale au décollage $M0$ et les données (entrées + sorties) correspondant à un avion ayant une masse maximale au décollage $M1$. En construisant un métamodèle, c'est-à-dire en estimant f, à partir des données de l'avion de MTOW $M0$, il nous est alors possible de prédire les sorties associées à l'avion de MTOW $M1$ et de quantifier l'erreur commise. Ainsi dans notre étude, nous comparons les méthodes de régression à base d'arbres dans un cadre d'extrapolation de masse maximale au décollage. Nous verrons également l'influence de différentes techniques de réduction de dimension sur ces algorithmes dans le cas de l'extrapolation.

Les données utilisées proviennent de quatre avions dérivés ayant une masse maximale au décollage respective de 238 tonnes, de 242 tonnes, de 247 tonnes et de 251t. A chaque avion est donc associé un jeu de données déjà calculé. Les données que nous avons à notre disposition sont

les paramètres avions utilisés dans la chaîne de calculs des charges. Les paramètres avions (vitesse, altitude, masse carburant, etc..) jouent le rôle des entrées d'un cas de charge, et nous voulons prédire les charges associées (les sorties) à ce cas de charge.

Un cas de charge est défini par ses 25 paramètres avions (ses entrées), et les moments de flexion (ses sorties) qui sont représentés par un vecteur de taille $k = 29$. De manière plus formel, les entrées sont définies par $\mathbf{X} = (X^1, ..., X^{25})$ où les X^j sont des variables quantitatives (i.e un paramètre avion), et $X^j = (x_1^j, ..., x_n^j)^T$, avec $n \approx 28\ 000$ observations. Les sorties sont définies par $\mathbf{Y} = (Y^1, ..., Y^{29})$ et $Y^j = (y_1^j, ..., y_n^j)^T$. Des exemples de moments de flexion sont présentés en Figure 1.2.

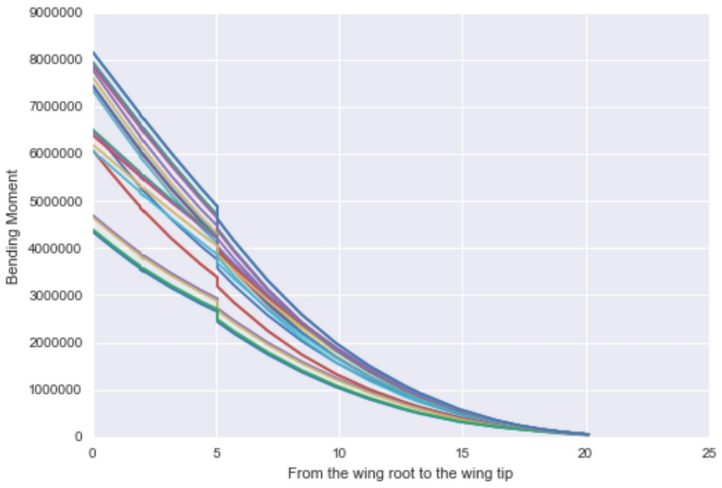


Figure 1.2: Exemples de moments de flexions le long d'une aile pour différents cas de charge

La première idée de cette étude est de construire un modèle à partir des données du 238t, puis de prédire les cas de charges du 242t, du 247t et 251t pour pouvoir comparer si la méthodologie est adaptée pour l'extrapolation. La seconde idée est de tester différentes méthodes de réduction de dimension. Nous souhaitons étudier leur influence sur les méthodes à base d'arbres de régression dans le cas d'une extrapolation bien définie.

Dans le cas des méthodes de réduction de dimension, nous nous sommes intéressé à: l'Analyse en Composantes Principales (ACP) euclidienne dans $\mathbb{R}^d$, à la transformation des courbes de moments de flexion par ajustement polynomial d'ordre p ou encore à un mélange de ces deux dernières méthodes. Pour quantifier l'influence des méthodes de réduction

de dimension, nous avons testé les 8 possibilités suivantes:

- (1) Aucune transformation des entrées et aucune transformation des sorties.

- (2) Aucune transformation des entrées et ACP sur les sorties : nous gardons l'espace originel des entrées et nous appliquons une ACP sur l'espace des sorties.

- (3) Aucune transformation des entrées et ajustement polynomial sur les sorties: nous gardons l'espace originel des entrées et nous remplaçons les sorties par les coefficients des polynômes ajustés.

- (4) Aucune transformation des entrées et ajustement polynomial sur les sorties et ACP sur les coefficients des polynômes: nous gardons l'espace originel des entrées et nous remplaçons les sorties par les coefficients des polynômes ajustés sur lesquels nous appliquons une ACP.

- (5) ACP inputs et aucune transformation des sorties: pas de transformation des sorties mais nous appliquons une ACP sur les entrées.

- (6) ACP inputs et ACP outputs: nous appliquons une ACP sur les entrées et les sorties.

- (7) ACP inputs et ajustement polynomial sur les sorties: nous appliquons une ACP sur les entrées, et nous remplaçons les sorties par les coefficients des polynômes ajustés.

- (8) ACP inputs et ajustement polynomial sur les sorties et ACP sur les coefficients des polynômes: nous appliquons une ACP sur les entrées et nous remplaçons les sorties par les coefficients des polynômes ajustés sur lesquels nous appliquons une ACP.

En complément de ces approches, nous avons testé les modèles de régression suivant : les Decision Trees [6], les Random Forest [5], la méthode AdaBoost [8] appliquée aux Decision Trees ou aux Random Forest, le Bagging [3] et enfin le Gradient Boosting [4]. En tout, une

cinquantaine de combinaisons ont été testées.

La consultation d'un simple R^2 ne nous permettant pas de quantifier l'erreur globale d'une courbe, nous utilisons le taux d'erreur pour une courbe défini de la façon suivante :

$$error(j) = \sqrt{\frac{\sum_{i=1}^{L}(\hat{y}(x_i) - y_j(x_i))^2}{\sum_{i=1}^{L} y_j^2(x_i)}}, \tag{1.1}$$

pour $j = 1, ..., n$, où n est la taille de l' échantillon sur lequel nous calculons l'erreur, $L = 29$ est le nombre de station le long de l'aile, où $y_j(x_i)$ la vraie valeur de la courbe j à la station i, et où $\hat{y}(x_i)$ est sa prédiction. Cela nous permet notamment d'avoir une idée de la qualité de nos prédictions. De plus, pour $j = 1, ..., n$, soit $\alpha \in [0, 1]$, nous utilisons la fonction de répartition empirique de l'erreur (CDF) s'écrivant:

$$\alpha \to G(\alpha) = \frac{1}{n}\sum_{j=1}^{n} \mathbb{1}_{(error(j)\leq\alpha)}. \tag{1.2}$$

La méthode Adaboost associée à Random Forest ainsi qu'une ACP sur les sorties est celle qui fonctionne le mieux sur nos données. Le tableau suivant donne les résultats moyens de cette méthode:

Table 1.1: $\mathbb{P}(error \leq 5\%)$, $\mathbb{P}(error \leq 10\%)$ et $\mathbb{E}(error)$ estimés pour les différents jeux de données 238t, 242t, 247t et 251t pour Adaboost associée à Random Forest

	238t	242t	247t	251t
$\mathbb{P}(error \leq 5\%)$	90%	89%	72%	71%
$\mathbb{P}(error \leq 10\%)$	96%	95%	89%	90%
$\mathbb{E}(error)$	11%	18%	18%	25%

Dans la Table 1.1, nous pouvons voir qu'AdaBoost associé à Random Forest donne de très bons résultats pour le jeu de données 242t. En effet, cette méthode atteint les 95% d'observations testées avec moins de 10% d'erreur. En revanche, dès que les données de test sont loin (247t et 251t), les résultats se dégradent. Une ACP sur les sorties améliore les résultats en moyennes, et cela peut être expliqué par la haute colinéarité des sorties, en revanche une ACP sur les entrées n'influe en rien les résultats et a plutôt tendance à les dégrader. Il est important d'insister sur le fait qu'utiliser une technique de réduction de dimension permet de réduire considérablement le

temps d'apprentissage.

Cette étude nous a conduit à identifier la problématique du Chapitre 3. En effet, bien que le faisceau de courbes fasse apparaitre une discontinuité, les courbes de charges sont très régulières et il est naturel de penser que toutes ces courbes ont été obtenues par la déformation d'une courbe mère. La famille de transformation considérée est relative à des similitudes du plan. Nous avons développé une méthode originale qui opère sur toute la courbe.

1.3 Estimation semi-paramétrique de similitudes du plan

Il est généralement utile, lorsque nous devons traiter un grand nombre de courbes différant légèrement de l'une à l'autre, de mettre en évidence un modèle de transformations paramétriques. En effet, l'application d'une méthode de réduction de dimension incluant une connaissance à priori peut mener à une meilleur compréhension de la variabilité du jeu de courbes en question. Une manière possible de procéder est de considérer que les courbes ont été obtenues à partir de la déformation d'une courbe mère. Ce point de vue fait référence au recalage de courbes dans la littérature. Autant que nous sachions, tous les modèles de la littérature considèrent des transformations agissant indépendamment sur l'argument et sur la valeur de la courbe de référence. Dans ce travail, nous considérons une famille de transformations paramétriques agissant simultanément sur l'argument et sur la valeur de la courbe. La transformation paramétrique en question implique un paramètre de rotation et un d'échelle. En d'autres termes, nous supposons que les courbes ont été obtenues par une rotation et une homothétie appliquées à la courbe mère. Plus exemple, prenons comme courbe de référence la function $f(x) := \frac{1}{2}(x-1)^2,\ x \in [0,1]$ en bleu Figure 1.3.

Lorsque nous appliquons un opérateur de rotation à la courbe de référence, l'espace des entrées est modifié. La Figure 1.3 (b) montre la courbe mère après rotation et la ligne verticale indique la coordonnée de l'origine de la courbe après transformation. L'espace des entrées est donc ensuite dilaté sur $[0,1]$ (Figure 1.3 (c)). Enfin, une homothétie est appliquée Figure 1.3 (d).

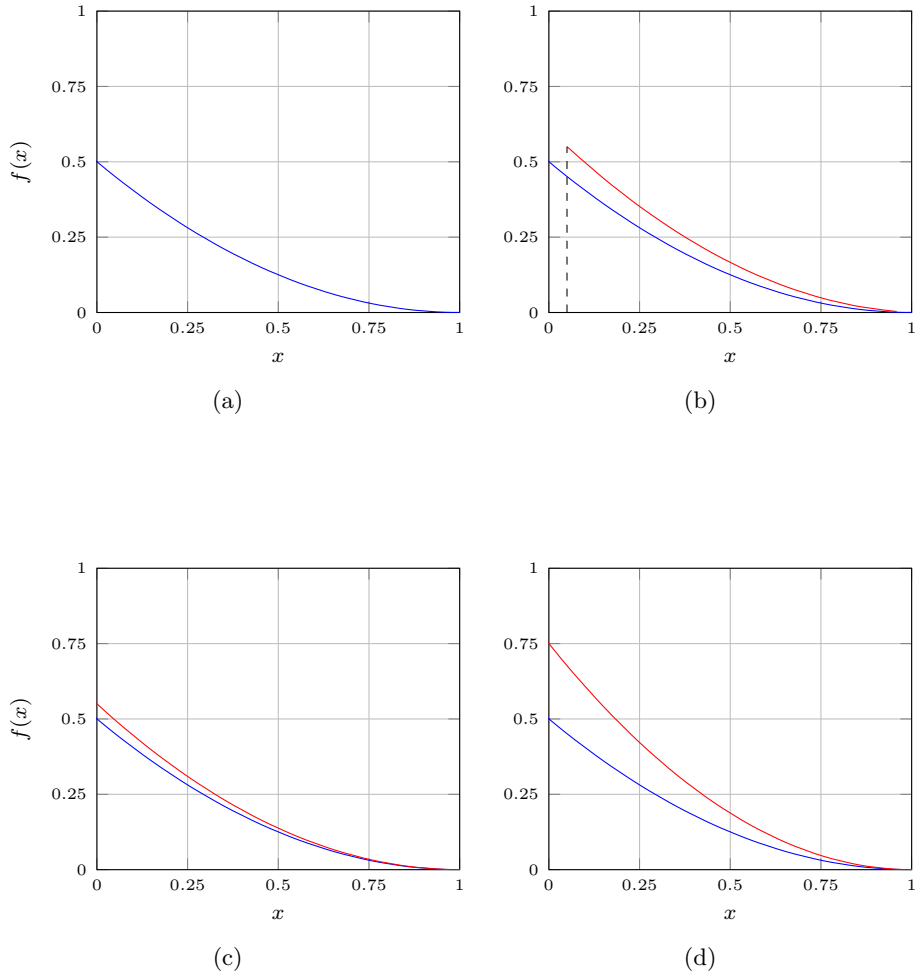


Figure 1.3: Etapes de transformations de la courbe de référence : (a) courbe mère initiale en bleu, (b) courbe mère après rotation (en rouge), (c) courbe mère après rotation et dilatation de l'espace des entrées, (d) courbe mère après toutes les transformations.

Nous travaillons avec des fonctions définies sur $[0, 1]$ et avec des similitudes agissant sur la courbe de référence $\tilde{C} := (x, \tilde{f}(x))_{x\in[0,1]}$. Le but est double: *estimer les paramètres de déformation* et *construire un métamodèle*. La première phase consiste à estimer les paramètres qui nous permettent de passer d'une courbe à la courbe de référence. La seconde est l'utilisation de ces paramètres pour représenter les courbes de charges aéronautiques comme présentées dans la section précédente et de prédire ces paramètres de déformation à partir des paramètres avions.

1.3.1 Cadre, modèle et résultats analytiques

1.3.1.1 Les observations

Dans le cadre de notre étude, nous avons à notre disposition $K+1$ courbes. Une est la courbe de référence $\tilde{C}$ aussi appelée courbe mère. Les K autres courbes $C_j,\ j=1,...,K$ sont supposées être les images de $\tilde{C}$ par le modèle de transformation décrit en dessous. Les K courbes sont observées sur une même grille aléatoire $\mathcal{D}_N := \{X_1,...,X_N\}$ où les $(X_i)_{i=1,...,N}$ sont des variables aléatoires iid suivant une distribution uniforme sur $[0,1]$. Ainsi, nous avons à notre disposition

$$C_j^N = (X_i, f_j(X_i))_{i=1,...,N,\ j=1,...,K}.$$

Nous considérerons les cas suivants

i) $\tilde{C}$ est connue partout: $\tilde{C} = (x, \tilde{f}(x)),\ \forall x \in [0,1]$,

ii) $\tilde{C}$ est connue sur $\mathcal{D}_N$: $\tilde{C} = (X_i, \tilde{f}(X_i))_{i=1,...,N}$.

1.3.1.2 Modèle de Transformation

Avant de définir le modèle de transformation, nous devons nous assurer que les fonctions $f_j,\ j=1,...,K$ and $\tilde{f}$ sont "admissibles".

Definition. Soit $\theta_0 \in]0, \frac{\pi}{2}[$, $\mathcal{F}_{\theta_0}$ est l'ensemble des applications défini par:

$$\begin{gathered}\mathcal{F}_{\theta_0} := \{f : [0,1] \to \mathbb{R}^+, f \text{ est differentiable sur } [0,1],\\ f(0) > 0, f(1) = 0,\\ \forall x \in [0,1], f'(x) < 0, f'(1) = 0, f'(x) > -\cot\theta_0\}.\end{gathered}$$

La classe des courbes $\mathcal{C}_{\theta_0}$ est définie tel que:

$$\mathcal{C}_{\theta_0} := \{(x, f(x)), x \in [0,1], f \in \mathcal{F}_{\theta_0}\}.$$

Définissons la famille de transformations paramétriques considérées.

Notation. *Pour toute fonction $T_\alpha : \mathbb{R}^2 \to \mathbb{R}^2$ dépendant d'un paramètre $\alpha \in \Theta$, nous dénotons par T_α^1 and T_α^2 ses deux coordonées.*

Nous définissons ainsi notre modèle de transformation.

Posons $0 < \theta_0 < \theta_1 < \frac{\pi}{2}$, et $0 < \lambda_{min} < \lambda_{max}$. Pour tout $\alpha := (\theta, \lambda) \in \Theta = [-\theta_0, \theta_1] \times [\lambda_{min}, \lambda_{max}]$, nous introduisons

$$T_\alpha := H_\lambda \circ \tilde{R}_\theta.$$

Notre modèle de transformation est donc la composition d'une rotation $\tilde{R}_\theta$ et d'un redimensionnement H_λ. Notons que les éléments de $\mathcal{F}_{\theta_0}$ sont des fonctions qui décroissent mais non brutalement. En effet, s'il y a un léger drop vertical sur le graphe de la fonction de $\mathcal{F}_{\theta_0}$ tel que $\forall x \in [0,1],\ |f'(x)| < \cot\theta_0$, si nous appliquons une rotation sur la courbe, alors la courbe transformée est encore une fonction. Les hypothèses suivantes sont utilisées par la suite pour nos résultats analytiques:

(H1): Les fonctions $(f_j)_{j=1,\dots,K}$, et $\tilde{f}$ appartiennent à $\mathcal{F}_{\theta_0}$.

(H2): Pour tout $\theta \in [-\theta_0, \theta_1]$, $\forall C \in \mathcal{C}_{\theta_0}$, toutes les secondes coordonnées de $\tilde{R}_\theta(C)$ sont positives.

(H3): $\Theta = [-\theta_0, \theta_1] \times [\lambda_{min}, \lambda_{max}]$.

(H4): $\tilde{C}$ est connue sur $[0,1]$.

(H5): $\tilde{C}$ est connue sur la grille $\mathcal{D}_N$.

Ainsi, le modèle considéré est

$$\begin{aligned} T_\alpha : \mathcal{C}_{\theta_0} &\to \mathcal{C}_{\theta_0} \\ C &\to T_\alpha(C) \end{aligned}$$

qui transforme chaque point $(x, f(x))$ de C en $\begin{pmatrix} \frac{(x-1)\cos\theta - f(x)\sin\theta}{\cos\theta + f(0)\sin\theta} + 1 \\ \lambda((x-1)\sin\theta + f(x)\cos\theta) \end{pmatrix}$: le modèle de transformation agit conjointement sur l'argument et la valeur de la fonction mère.

1.3.1.3 Modèle de régression

Rappelons que nous souhaitons recaler la courbe mère $\tilde{C}$ sur les autres courbes C_j $(j = 1, ..., K)$ grâce aux transformations de $\{T_\alpha, \alpha = (\theta, \lambda) \in \Theta\}$. Pour tout α, nous voulons comparer la valeur de la courbe transformée $(T_\alpha \tilde{C})(X_i)$ à $f_j(X_i)$. Puisque les abscisses sont affectés par les transformations, nous dénotons par $X_i(\alpha)$ le point tel que $T^1_\alpha(X_i(\alpha)) = X_i$. Considérons ainsi le modèle de régression suivant :

$$f_j(X_i) = T^2_{\alpha^*_j}(X_i(\alpha^*_j), \tilde{f}(X_i(\alpha^*_j))) + \epsilon_{j,i}, (j = 1, ..., K). \tag{1.3}$$

Où:

- (X_i) sont iid, de distribution uniforme sur $[0, 1]$. C'est le design sur lequel nous observons les courbes C_j;
- $\alpha^*_j = (\theta^*_j, \lambda^*_j)$ est le couple des vrais paramètres pour chaque courbe $(j = 1, ..., K)$;
- $\epsilon_{j,i},\ \forall i = 1, ..., N,\ \forall j = 1, ..., K$ sont iid de distribution $\mathcal{N}(0, \sigma^2)$. Ces variables sont supposées indépendantes de X_i.

1.3.1.4 Estimation

Nous nous plaçons dans le cas où les courbes à recaler sont bruitées. La courbe mère qui est supposée non-bruitée. Nous estimons les paramètres de déformations à travers une procédure de M-estimation et donnons leurs propriétés asymptotiques dans deux cas. Nous considérons tout d'abord le cas où la courbe mère est connue sur $[0, 1]$. Bien que non réalistes, ces résultats sont des pré-requis nécessaires pour montrer la consistence et la normalité asymptotique des estimateurs dans le second cas où la courbe mère est observée sur la même grille $\mathcal{D}_N$ que le jeu de courbes à recaler.

1.3.1.4.1 Estimation quand $\tilde{C}$ est connue sur $[0, 1]$

Fixons j. En suivant une procédure de M-estimation, nous définissons la fonction de contraste empirique pour recaler la courbe mère $\tilde{C}$ sur C_j $(j = 1, ..., K)$:

$$M^j_N(\alpha) = \frac{1}{N} \sum_{i=1}^{N} (f_j(X_i) - T^2_\alpha(X_i(\alpha), \tilde{f}(X_i(\alpha))))^2 \tag{1.4}$$

La fonction M^j_N est non-négative et son minimum devrait être atteint pour une valeur proche du vrai paramètre α^*_j. En effet, le théorème suivant donne la consistence et la normalité asymptotique de l'estimateur suivant :

$$\hat{\alpha}^j_N = \underset{\alpha \in \Theta}{\operatorname{argmin}}\ M^j_N(\alpha). \tag{1.5}$$

Theorem 1. *Supposons que les hypothèses H1, H2, H3 et H4 soient satisfaites. Alors*

$$i)\ \hat{\alpha}_N^j \xrightarrow[N\to+\infty]{\mathbb{P}} \alpha_j^*, \tag{1.6}$$

$$ii)\ \sqrt{N}(\hat{\alpha}_N^j - \alpha_j^*) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha_j^*}). \tag{1.7}$$

En particulier, la matrice de covariance a la forme suivante

$$\Gamma_{\alpha_j^*} = V_{\alpha_j^*}^{-1} 2\sigma^2, \tag{1.8}$$

avec $V_{\alpha_j^*} = 2\mathbb{E}[\dot{T}^2_{\alpha_j^*}\dot{T}^{2}_{\alpha_j^*}\boldsymbol{T}]$, *et* $\dot{T}^2_{\alpha_j^*}$ *est le vecteur des dérivées partielles de* $T^2_{\alpha_j}$ *par rapport à* α *et évaluées en* α_j^*.

1.3.1.4.2 Estimation quand $\tilde{C}$ est observée sur $\mathcal{D}$

Lorsque la courbe mère $\tilde{C}$ est observée sur la même grille que les autres courbes, $T_\alpha \tilde{C}$ n'est plus observable sur $\mathcal{D}$. Par conséquent, nous remplaçons $\tilde{f}$ par son interpolant linéaire $\tilde{f}_N$.

$$\tilde{f}_N(x) = \sum_{i=1}^{N} \Delta_i(x) \mathbb{1}_{x\in[X_{(i)}, X_{(i+1)})}, \tag{1.9}$$

où

$$\Delta_i(x) = \frac{\tilde{f}(X_{(i+1)}) - \tilde{f}(X_{(i)})}{X_{(i+1)} - X_{(i)}}\, x + \tilde{f}(X_{(i)}) - \frac{\tilde{f}(X_{(i+1)}) - \tilde{f}(X_{(i)})}{X_{(i+1)} - X_{(i)}} X_{(i)}. \tag{1.10}$$

$\tilde{f}_N$ appartenant à $\mathcal{F}_{\theta_0}$, et remplaçant $\tilde{C}$ par $\hat{C}$ dans (1.4), nous obtenons de manière analogue la fonction de contraste empirique

$$\hat{M}_N^j(\alpha) = \tfrac{1}{N} \textstyle\sum_{i=1}^{N} (f_j(X_i) - T_\alpha^2(X_i(\alpha, N), \tilde{f}_N(X_i(\alpha, N))))^2,$$

où $X_i(\alpha, N)$ est solution de l'équation $T_\alpha^1(u, \tilde{f}_N(u)) = X_i$.

En utilisant l'interpolant linéaire $\tilde{f}_N$, il est alors possible de montrer la consistence et la normalité asymptotique du M-estimateur défini par:

$$\hat{\hat{\alpha}}_N^j = \underset{\alpha\in\Theta}{\operatorname{argmin}}\ \hat{M}_N^j(\alpha). \tag{1.11}$$

Theorem 2. *Supposons que les hypothèses H1, H2, H3 et H5 soient satisfaites. Soit* $\tilde{f}_N$ *défini par (1.9) supposons* $\exists C > 0$ *tel que* $\forall x \in [0,1]$, $\tilde{f}'_N(x) \le C$, *et* $\tilde{f}'(x) \le C$, *alors*

$$i)\ \hat{\hat{\alpha}}_N^j \xrightarrow[N\to+\infty]{\mathbb{P}} \alpha_j^*, \tag{1.12}$$

$$ii)\ \sqrt{N}(\hat{\hat{\alpha}}_N^j - \alpha_j^*) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha_j^*}) \tag{1.13}$$

avec $\Gamma_{\alpha_j^*}$ *tel que défini dans (1.8).*

1.3.2 Application aux charges aéronautiques

Tout d'abord notons que les problèmes d'optimisation sont résolus par BFGS [12]. Les données à notre disposition sont les moments de flexion d'un avion, calculés pour 1152 configurations différentes. Chaque configuration est définie par 28 paramètres avions (vitesse de l'avion, masse, altitude, quantité de fuel, etc.) donnant un moment de flexion calculé en 45 stations le long de l'aile. D'une manière plus formelle, nous observons le couple $(X_j, Y_j)_{j=1,...,1152}$, où $X_j = (X_j^1, ..., X_j^{28})$ sont les entrées et $Y_j = (Y_j^1, ..., Y_j^{45})$ le moment de flexion. L'idée est de prédire ces moments de flexion pour différentes configurations dans un cas iso-masse. Le jeu de courbes est présenté en Figure 1.4.

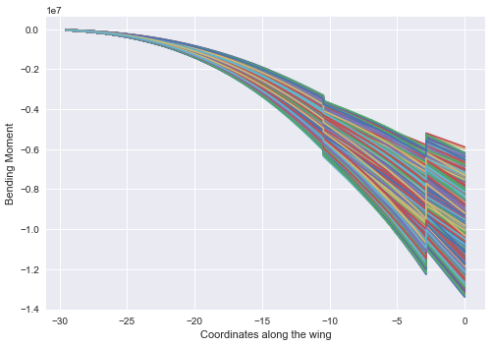


Figure 1.4: Représentation de tous les moments de flexion du jeu de données : l'emplanture de l'aile est située à l'origine.

Des discontinuités étant présentes à la troisième et à la vingtième station, nous appliquons notre méthodologie à chaque section indépendamment : nous supposons que la courbe moyenne de chaque section est la courbe mère. Après ajustement, chaque section de courbe est donc représentée par ses paramètres de déformation (de rotation et d'échelle).

Dans la même idée qu'exposée dans la première section, nous comparons notre méthode à 3 autres appliquées aux sorties : aucune réduction de dimension, une ACP, un ajustement polynomial. Pour construire nos métamodèles, nous utilisons l'Orthogonal Greedy Algorithm (OGA) aussi

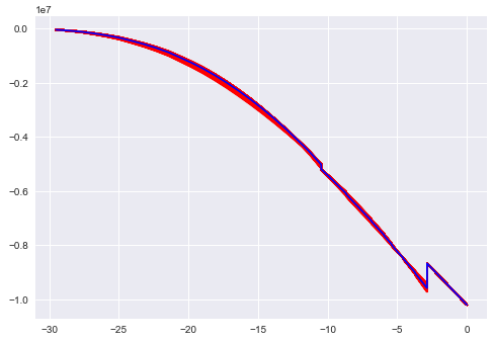


Figure 1.5: Courbes après recalage

connu sous le nom de Matching Pursuit Algorithm (des explications détaillées peuvent être trouvées dans [2], [14] et [10]). Nous considérons donc le problème d'approcher une fonction par une combinaison linéaire de variables.

Pour évaluer la qualité de nos modèles, nous utilisons la mesure d'erreur définie par (1.1) ainsi que la fonction de répartition empirique de l'erreur définie par (1.2). La Table 1.2, donne différents résultats de la fonction de répartition empirique en $\alpha = 1\%, 2\%, 5\%, 10\%$ ainsi que l'erreur moyenne. La Figure 1.6 donne les fonctions de répartitions empiriques des différentes méthodes.

Table 1.2: Estimation moyenne de $\mathbb{P}(error \leq 1\%)$, $\mathbb{P}(error \leq 2\%)$, $\mathbb{P}(error \leq 5\%)$ $\mathbb{P}(error \leq 10\%)$, $\mathbb{E}(error)$ calculés sur plusieurs échantillons tirés aléatoirement(25% du jeu de données total)

	Modèle de déformation	Ajustement polynomial	ACP	Aucune transformation
$\mathbb{P}(error \leq 1\%)$	17%	14%	16%	15%
$\mathbb{P}(error \leq 2\%)$	45%	45%	43%	51%
$\mathbb{P}(error \leq 5\%)$	88%	88%	86%	88%
$\mathbb{P}(error \leq 10\%)$	98%	97%	95%	98%
$\mathbb{E}(error)$	2.9%	2.9%	3%	2.8%

Les résultats en prédiction de l'association de notre modèle de déformation et de la méthode de régression Greedy sont équivalents au cas où nous n'appliquons aucune transformation ce qui confirme la bonne intuition de départ concernant le modèle sous-jacent. Les résultats concernant l'erreur moyenne surpassent les résultats obtenus dans la première section avec les arbres de régression et la $\mathbb{P}(error \leq 10\%)$ est légèrement supérieure, ce qui implique que nos erreurs maximales ont été largement diminuées.

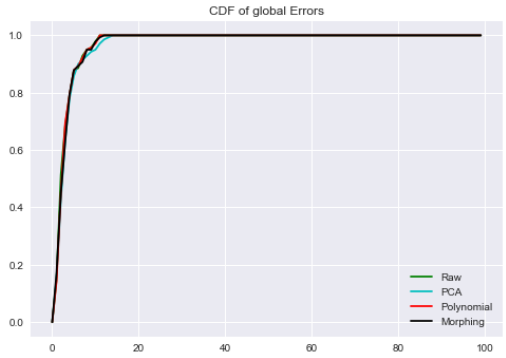


Figure 1.6: Fonction de répartition empirique de l'erreur ($\mathbb{P}(error \leq \alpha)$)

Enfin, en utilisant ce modèle de déformation, cela nous permet notamment d'obtenir une réponse physique et d'extraire de l'information concernant la variabilité des courbes de charges pour les ingénieurs et experts métiers.

1.4 Prédiction préliminaire du moment de flexion maximal d'une aile d'avion sous rafale discrète

A un stade précoce de développement, le modèle de simulation détaillé d'un avion dérivé n'est pas disponible et les charges sont estimées de manière empirique ou grâce à une expertise métier. Une mauvaise estimation de ces charges peut induire des incertitudes qui conduisent à un redimensionnement coûteux de la structure de l'avion dérivé. Comme dans la Section 1.2, nous considérons une variation de masse maximale au décollage pour les avions dérivés. Les charges concernées sont les moments de flexion de l'aile de l'avion sous une rafale discrète. La rafale discrète est caractérisée par sa longueur d'onde. Elle produit un effet similaire sur l'avion à celui d'un dos d'âne sur une automobile. Notre méthode repose sur la création d'un jeu de données critiques à partir de la base de données d'un avion initial. Ce jeu de données sera ensuite utilisé pour construire un modèle parcimonieux permettant de prédire les charges critiques de deux avions dérivés.

Dans un premier temps, intéressons nous à la construction de la base de données d'un avion initial. Cette base est issue d'un code numérique calculant les charges aéronautiques que subit une aile d'avion. Ce code numérique est basé sur un modèle aérodynamique prenant en entrée une configuration avion $\mathbf{x} \in \mathbb{R}^d$ avec $d = 20$ (vitesse, altitude, masse de fuel, longueur d'onde de la rafale, etc.) et donnant en sortie la réponse dynamique de l'avion. Il peut être représenté par la fonction $M_{K,T}$ définie par

$$M_{K,T} : \mathbb{R}^{20} \to \mathcal{M}_{K\times T}(\mathbb{R})$$
$$\mathbf{x} \to M_{K,T}(\mathbf{x}) = \begin{pmatrix} M(1,\mathbf{x},t_1) & ... & M(1,\mathbf{x},T) \\ ... & ... & ... \\ M(K,\mathbf{x},t_1) & ... & M(K,\mathbf{x},T) \end{pmatrix}$$

où :

- T correspond au nombre de pas de temps où est calculée la réponse dynamique de l'avion ($t \in \{t_1, ..., T\}$).
- $K = 45$ correspond au nombre de stations le long de l'aile de l'avion ($k \in \{1, ..., K\}$).

En d'autres termes, la réponse dynamique de l'avion est l'ensemble des valeurs des moments de flexion $M(k, \mathbf{x}, t)$ pour toutes les stations k, $1 \leq k \leq K$ et pour tous les pas de temps $t \in \{t_1, ..., T\}$.

Pour une configuration avion $\mathbf{x}$ donnée, et pour une station donnée, nous nous intéressons à la valeur maximale des moments de flexion de la réponse temporelle donnée par

$$M_B(k, \mathbf{x}) = \max_{t\in\{t_1,...,T\}} M(k, \mathbf{x}, t). \tag{1.14}$$

Dans un premier temps, l'objectif est d'estimer la fonction $M_B(k, \mathbf{x})$ pour toute station k et toute configuration $\mathbf{x}$. Dans un second temps, cela nous permettra de donner une estimation des valeurs critiques $M_B^*(k) = \max_{\mathbf{x}} M_B(k, \mathbf{x})$. Pour identifier la fonction $M_B(k, \mathbf{x})$, nous avons à notre disposition les observations $(\mathbf{x}_i, Y_i)_{i=1,...,n}$, où $n \approx 1500$ de l'avion initial ayant une masse maximale au décollage 235 tonnes. Dans la suite, nous utilisons les notations suivantes : $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^\mathbf{T} \in \mathcal{M}_{nxd}(\mathbb{R})$ est la matrice des entrées (une ligne de $\mathbf{X}$ est notée $\mathbf{x}_i$) et les sorties correspondantes sont $\mathbf{Y} = (M_B(k, \mathbf{x}_1), ..., M_B(k, \mathbf{x}_n))^\mathbf{T} = (Y_1, ..., Y_n)^\mathbf{T} \in \mathbb{R}^n$.

Nous modélisons les sorties $\mathbf{Y}$ par une fonction polynomiale $\hat{M}$. Plus précisemment, $\hat{M}$ est un polynôme de degré 2, c'est-à-dire que pour toute configuration $\mathbf{x}$ à la station k on a

$$\hat{M}(k, \mathbf{x}) = a_0(k) + \sum_{i=1}^{d} a_i(k)x_i + \sum_{i=1}^{d}\sum_{j=i}^{d} a_{i,j}(k)x_i x_j = \sum_{i=1}^{D} \omega_i(k)\Phi_i(\mathbf{x}). \tag{1.15}$$

Ici, $D = \frac{d^2+3d+2}{2} = 231$, les $\omega_i(k)$ $(i = 1, ..., D)$ sont les coefficients de régression à estimer et les Φ_i sont les fonctions de la base polynomiale d'ordre $p \leq 2$. Pour des raisons de parcimonie, nous allons utiliser un

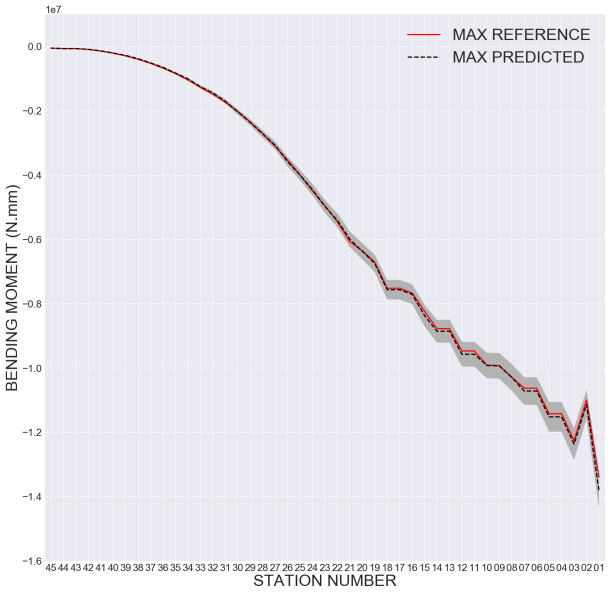
algorithme de choix de modèle pour lequel les coefficients sont estimés avec l'*Orthogonal Greedy Algorithm* (OGA) (aussi appelé *Orthogonal Matching Pursuit* [10]). Cet algorithme possède de bonnes propriétés computationnelles. L'utilisateur va fixer un nombre $l \leq D$ de fonctions de base à utiliser et l'algorithme va choisir itérativement la fonction de base à ajouter au modèle réduisant le plus le résidus. L'algorithme est alors décrit par

Algorithm 1 Algorithme OGA

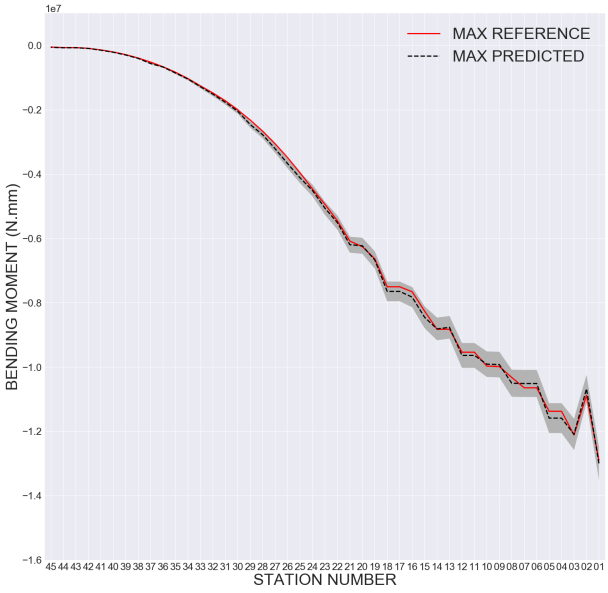
1: *Fixons* $l \in \mathbb{N}^*$
2: *Fixons* $h_0 = 0$
3: **for** $j = 1, ..., l$ **do**:
4: $\quad s(j) := \underset{1 \leq k \leq D}{\operatorname{argmin}} |\frac{1}{n} \sum_{i=1}^{n} (Y_i - h_j) \Phi_k(\mathbf{x}_i)|$
5: $\quad P^j := OLS\ sur\ vect\{\Phi_{s(1)}, ..., \Phi_{s(j)}\}$
6: $\quad h_j := P^j \mathbf{Y}$

h_j est la projection de $\mathbf{Y}$ sur le sous espace vectoriel $vect\{\Phi_{s(1)}, ..., \Phi_{s(j)}\}$. Le nombre optimal $l \in \mathbb{N}^*, 1 \leq l \leq D$ de coefficients à estimer est obtenu par Validation-Croisée. La procédure exprimée ci-dessus a été étendue aux autres stations le long de l'aile de l'avion, ainsi 45 modèles ont été construits.

Ces metamodèles sont alors utilisés pour prédire les valeurs des moments de flexion pour les configurations de deux avions dérivés ayant une masse maximale au décollage de 210 tonnes et 280 tonnes. Ces configurations avions sont dénotés par $\mathbf{x}_{210t,\ i=1,...,n}$ et $\mathbf{x}_{280t,\ i=1,...,n}$. A partir de ces valeurs prédites, la valeur extrême estimée pour chaque station est donnée par $\hat{M}^*_{210t}(k) = \max_{i=1,...,n} \hat{M}(k, \mathbf{x}_{210t,\ i})$ pour le 210t et de manière respective pour le 280t. Les valeurs maximales des moments prédits et réels sont comparés dans la Figure 1.7. Pour le 210t (respectivement le 280t) l'erreur maximale obtenue le long de l'aile est de 2% (respectivement 3.5%) avec une erreur relative à l'emplanture de l'aile de 0.8% (respectivement 0.9%).



(a)



(b)

Figure 1.7: Valeurs extrêmes des moments de flexions en chaque station : la courbe en rouge correspond aux vraies valeurs extrêmes, la noire en pointillée correspond au valeurs prédites - (a) Prédiction pour l'avion dérivé 210t, (b) Prédiction pour l'avion dérivé 280t

En conclusion, ce chapitre présente une méthode simple et efficace

pour estimer les cas de charges critiques d'avions dérivés dans le cadre d'un pré-dimensionnement. L'avantage notable de ce type de méthode est l'explicabilité du modèle pour les ingénieurs et la rapidité d'éxécution. Les résultats obtenus sont très encourageants et suggèrent également que le processus de design utilisé engendre de faibles variations prévisibles si l'avion et les configurations de vol ne changent pas trop. Cette méthode peut s'avérer utile pour un constructeur ayant toutes les données nécessaires et peut être appliquée à d'autres quantités d'intérêt ou parties structurelles.

Bibliography

[1] Airbus, C.A.: A330 family (2017). Available from : http://www.aircraft.airbus.com/aircraftfamilies/passengeraircraft/a330family/

[2] Barron, A.R., Cohen, A., Dahmen, W., DeVore, R.A.: Approximation and learning by greedy algorithms. Ann. Statist. **36**(1), 64–94 (2008). DOI 10.1214/009053607000000631. URL https://doi.org/10.1214/009053607000000631

[3] Breiman, L.: Bagging predictors. Machine Learning **24**, 123–140 (1996)

[4] Breiman, L.: Arcing the edge. Technical Report (486) (1997). Statistics Department, University of California

[5] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)

[6] Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: Classification and Regression Trees. Wadsworth, Belmont, CA (1984)

[7] Cleveland, W.S.: Robust locally weighted regression and smoothing scatterplots. Journal of the American statistical association **74**(368), 829–836 (1979)

[8] Friedman, J.H.: Greedy function approximation: A gradient boosting machine (1999)

[9] Lawton, W., Sylvestre, E., Maggio, M.: Self modeling nonlinear regression. Technometrics **14**(3), 513–532 (1972)

[10] Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on Signal Processing **41**(12), 3397–3415 (1993). DOI 10.1109/78.258082

[11] Manyika, J., al.: Big data: the next frontier for innovation, competition and productivity (2011). Mc Kinsley Global Institute

[12] Nocedal, J., Wright, S.J.: Numerical Optimization, second edn. Springer, New York, NY, USA (2006)

[13] Ramsay, J.O., Li, X.: Curve registration. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **60**(2), 351–363 (1998)

[14] Sancetta, A.: Greedy algorithms for prediction. Bernoulli **22**(2), 1227–1277 (2016). DOI 10.3150/14-BEJ691. URL https://doi.org/10.3150/14-BEJ691

Chapter 2

Bibliographical study

In this bibliographical study, we will detail the algorithms and methods set out and used in the following chapters. The first section is a short review dedicated to regression models and their associated algorithms used in the following chapters. In this section, we consider nonparametric regression models with the local polynomial regression model and tree-based algorithms. Then, we provide details about a sparse high dimensional parametrical model: the Greedy algorithm. In the second section, we focus on deformation models. Indeed, a common problem in statistics is to find a good representation of the data under study. When dealing with a set of curve, one way to proceed is to consider that the curves have been obtained by deforming a template. In this work, we focus on two deformation methods: a parametric one called the Shape Invariant Model and a nonparametric one called the Dynamic Time Warping.

2.1 A short review on regression methods and associated algorithms

Let us consider a random pair $\mathbf{Z} := (\mathbf{X}, \mathbf{Y})$ where here $\mathbf{X} \in \mathcal{X} := \mathbb{R}^p$ is called *inputs* or *features* and $\mathbf{Y} \in \mathcal{Y} := \mathbb{R}$ is named *output* or *target variable*. $\mathbf{X}$ and $\mathbf{Y}$ are linked with the following relation

$$\mathbf{Y} = f(\mathbf{X}) \ + \ \epsilon \tag{2.1}$$

$$\tag{2.2}$$

with $f(\mathbf{X}) = \mathbb{E}[\mathbf{Y}|\mathbf{X}]$ and $\epsilon = (\mathbf{Y} - \mathbb{E}[\mathbf{Y}|\mathbf{X}])$. In statistics, a classical problem is the estimation of the function f from observations $\mathbf{Z}_i = (\mathbf{X}_i, \mathbf{Y}_i)_{i=1,\dots,n}$. We assume that the observations are independent and identically distributed (i.i.d) with the same distribution as $(\mathbf{X}, \mathbf{Y})$, and ϵ is centered and uncorrelated.

2.1.1 Nonparametric regression models

2.1.1.1 Local polynomial regression

The approximation procedure can be tackled *locally* or *globally*. For the latter, a common practice consists in making the assumption that $\mathbf{X}$ and $\mathbf{Y}$ are bound by a linear combination of variables. Nevertheless, in most cases a nonlinear model is needed to estimate f. One way to achieve this goal is to consider that f belongs to a known class of functions having certain properties. We can, for example, consider the class of polynomials of degree $\leq d$. Nevertheless, even if widely used, this technique suffers from large biases problems (see [24] for more details). Besides, it does not take into account the fact that some points can have a major influence in the regression problem as pointed out by [24]. Finally, increasing the order of the polynomial estimators would be a partial solution since it reduces the bias but leads to less stable models. Hence, to counterbalance the drawbacks of the "global" polynomial regression approach, the idea is then to apply a regression model locally such as the local polynomial regression, initially exposed in [15] and studied in [18, 23, 24].

Let us briefly present the local polynomial regression in dimension $p = 1$ [24]. The aim is to estimate the regression function f in (2.2) at x_0, i.e $f(x_0) = \mathbb{E}[\mathbf{Y}|\mathbf{X} = x_0]$. If f is d times derivable at x_0, we can approximate f using the Taylor's expansion

$$f(x) \approx f(x_0) + ... + \frac{f^{(d)}(x_0)}{d!}(x - x_0)^d. \tag{2.3}$$

The idea is then to solve the following weighted least squares regression problem locally

$$(\mathcal{P}) : \min_{\beta \in \mathbb{R}^{d+1}} \sum_{i=1}^{n} \{\mathbf{Y}_i - \sum_{j=0}^{d} \beta_j (\mathbf{X}_i - x_0)^j\}^2 K_h(\mathbf{X}_i - x_0).$$

The kernel function $K_h(.) = \frac{K(\frac{\cdot}{h})}{h}$ acts as weight for the different observations in the neighborhood of x_0 with bandwidth h. The kernel function may have different forms such as Nadaraya-Watson [54, 69] and Gasser-Müller [32], but [24] recommends the Epanechnikov kernel defined as $K(.) = \frac{3}{4}(1 - z)_+^2$. Using the fact that K is a symmetric density function, the following estimator solves the problem $(\mathcal{P})$

$$\hat{\beta} = \begin{pmatrix} \hat{\beta}_0 \\ . \\ . \\ . \\ \hat{\beta}_d \end{pmatrix} = (\mathbf{X}^T \mathbf{W} \mathbf{X})^{-1} \mathbf{X}^T \mathbf{W} \mathbf{Y} \tag{2.4}$$

with

$$\mathbf{X} = \begin{pmatrix} 1 & (\mathbf{X}_1 - x_0) & ... & (\mathbf{X}_1 - x_0)^d \\ ... & ... & ... & ... \\ 1 & (\mathbf{X}_n - x_0) & ... & (\mathbf{X}_n - x_0)^d \end{pmatrix}$$

and

$$\mathbf{W} = \text{diag}\{K_h(X_i - x_0)\}.$$

Remark. If the weight matrix is the identity matrix, we reach the same results as the global polynomial regression approach. If we take the first two-columns, we reach the same results as the classical linear regression.

The choice of a small bandwidth may lead to computational problems since we fit numerous local model. Besides, a small bandwidth leads to the classical overfitting problem and a too big one to bias problem. Additionally, the order of the polynomial should not be too high to avoid computational issues as well.

As explained in [16, 59, 24], one can deal with the multivariate case $p \geq 1$ by considering nonnegative kernel function K of order p with compact support, i.e we have

$$K_B(\mathbf{z}) = \frac{1}{|B|} K(B^{-1}\mathbf{z})$$

where B is a non singular $p \times p$ matrix, and $|B|$ is its determinant. The analogue problem of problem $(\mathcal{P})$ still has an explicit solution. For first order polynomial, this solution is given by

$$\hat{\beta} = (\mathbf{X}_P^T \mathbf{W} \mathbf{X}_P)^{-1} \mathbf{X}_P^T \mathbf{W} \mathbf{Y}. \tag{2.5}$$

where

$$\begin{aligned} \mathbf{x}_0^T &= (x_{01}, ..., x_{0p}), \\ \mathbf{X}_i &= (X_{i1}, ..., X_{ip})^T, \\ \mathbf{X}_P &= \begin{pmatrix} 1 & (X_{11} - x_{01}) & ... & (X_{1p} - x_{0p}) \\ ... & ... & ... & ... \\ 1 & (X_{n1} - x_{01}) & ... & (X_{np} - x_{0p}) \end{pmatrix} \end{aligned}$$

and

$$\mathbf{W} = \text{diag}\{K_h(\mathbf{X}_i - \mathbf{x}_0)\}.$$

It is worth noting that we can have easily the same kind of formula for $d \geq 1$ but they becomes rapidly a complicated task. This method brings us to understand easily why such a regression method has difficulties to scale up, in terms of observations, variables, and order of polynomial. Besides, the notion of neighborhood in large dimension loses all meaning. Indeed, the larger is the dimension, the lower is the probability of finding an other observation in the same neighborhood. Hence, the idea would be to use a more global approach when the dimension grows.

In the next subsection, we present tree-based regression models that are one of the most popular method to overcome the issues presented above.

2.1.1.2 Regression trees based models

In this section, we describe one of the major class of algorithms: the Regression trees based models. While this subject has been the object of numerous publications from a theoretical, computational and numerical point of views since 1964, it is still an active field of research and even more with the massive amount of data that we are now facing with. The baseline regression trees will be discussed in the first subsection, before developing the Bagging in the second and Gradient Boosting in the third subsection. Random Forests will be detailed in the following and we will end with the so called AdaBoost algorithm.

2.1.1.2.1 Baseline regression trees

The very first regression tree in the literature appeared in 1964 with the Automatic Interaction Detection (AID) [65]. Since, the number of publications on this subject has not ceased to grow and many reviews of regression trees as been realized. Some of them provide a good and concise basis in the understanding of regression trees such as [34, 28], and others provides precise explanation of how this algorithm has evolved - we refer to [53] and more recently to [48] notably.

Before going into further details, one shall define the structure and the construction of a tree (see Figure 2.1). Simply speaking, the construction of a tree is the successive partitioning of the input space thanks to the outputs in the form of a sequence of nodes t. Let $(A_1, ..., A_M)$ be a partition of $\mathcal{X}$. To each element of the partition is associated a leaf $l_i,\ i = 1, ..., M$ (i.e a terminal node, the dashed circles in Figure 2.1). Hence, each element of the partition can be seen as a "splitting rule", dividing the space $\mathcal{X}$. In general, we will refer by $A(t)$ the element of the partition of $\mathcal{X}$ associated to the node t. In other words, we estimate f by a piecewise constant function $\hat{f}$ such that

$$\hat{f}(\mathbf{x}) := \sum_{i=1}^{M} c_i \mathbb{1}(\mathbf{x} \in A_i) \tag{2.6}$$

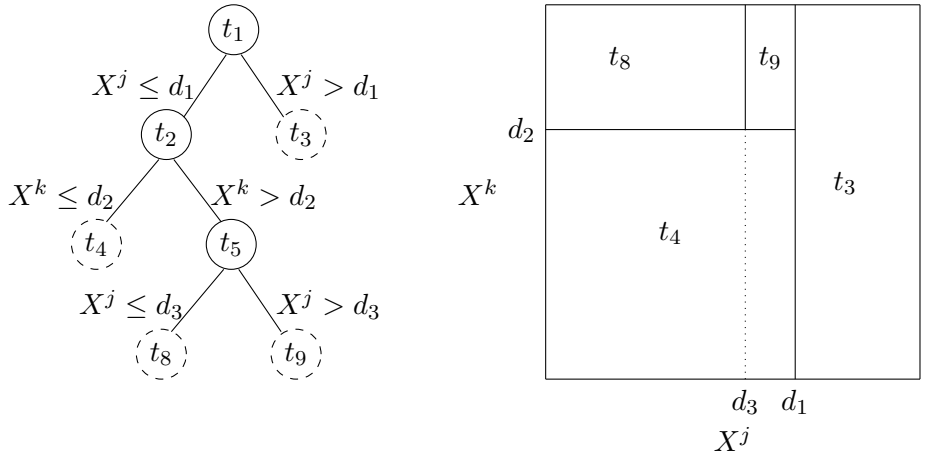


Figure 2.1: Example of construction of a tree: nodes are designed by $t_.$, dashed circles are terminal nodes (i.e leaves)

Back in the days, the AID algorithm has shown quickly its limitations for the main reason that it tends to overfit the data and leads to unstable predictions. Therefore, to overcome this likely drawback, Breiman proposed the so called Classification and Regression Tree Algorithm (CART) in [12]. We can develop this algorithm into two parts: the maximal tree construction and the tree pruning.

Concerning the maximal tree construction, this algorithm splits recursively the data in terms of sum of squared deviation (impurity function) in a binary manner. Consider that one want to split the node t. Recall that $A(t)$ is the element of the partition referring to the node t. Let $B_t := \{j,\ X_j \in A(t)\}$ and $n_t := \#B_t$. A split s is a binary rule which replaces $A(t)$ by two distincts subset $A(t_L)$ and $A(t_R)$ (t_L and t_R being respectively the left and right children nodes), i.e $A(t) := A(t_L) \cup A(t_R)$ with $A(t_L) \cap A(t_R) := \emptyset$. Let be S the set of possible splits s at a node t. s^* is the best split if:

$$\Delta R(s^*, t) := \max_{s \in S} R(t) - R(t_L) - R(t_R) \tag{2.7}$$

with $R(t) = \frac{1}{n_t} \sum_{j \in B_t} (Y_j - c_t)^2$. Due to the fact that the impurity function is the sum of squared deviation, c_t is necessarily set to $\hat{c}_t = \bar{Y}_t$. In other words, the division criteria corresponds to the split which minimizes

the within variance of the two resulting children nodes. A consequence of the choice of c_t is that it allows to compute and search quickly the optimal split. The splitting procedure is repeated but can be stopped at certain nodes for two reasons: the sub-sample of the considered node contains too little data according to a fixed threshold set by the user (generally 5, in AID, a node was considered as a leaf if $\Delta R(s^*, t) \leq 0.006 R(t_1)$), or the sample linked to the node is homogeneous and no other division is acceptable (that is to say that possible divisions lead to an empty child node). Hence, this procedure develops a maximal tree, noted T_{max}, and at each leaf is associated the average value of $\mathbf{Y}$s associated to the sub-sample of the leaf. As a consequence, and supposing that the tree has effectively $|M|$ leaves, the tree is a piecewise constant estimate of f such as defined in (2.6).

Nevertheless, such as the AID algorithm, this procedure leads to a large tree overfitting the data. From this standpoint, Breiman [12] added the procedure of *cost-complexity pruning* to counterbalance this problem. Let denote by $T \subset T_{max}$ a sub-tree with $|T|$ leaves. The cost-complexity pruning criterion aims to find the best equilibrium between the size of the tree and the goodness of fit and is defined as

$$C_\alpha(T) = \sum_{i=1}^{|T|} n_i R(t_i) + \alpha |T| \tag{2.8}$$

$$= \bar{R}(T) + \alpha |T| \tag{2.9}$$

with $\alpha \geq 0$. High values of α leads to small trees, i.e it penalizes complex trees. Nevertheless, testing all possible subtrees is prohibitive. Hence, Breiman shows in [12] that an iterative fashion way of cutting branches leads to an optimal solution. In the study [34], Genuer et al. detail the algorithm: consider that an internal node t of a tree T, they denote by T_t the subtree derived from this node, we have

Algorithm 2 Pruning algorithm for CART [12]

1: **procedure** (Inputs)
2: T_{max}
3: **procedure** (Initialization)
4: $\alpha_1 = 0$, $T_1 = T_{\alpha_1} = \underset{T \prec T_{max}}{\operatorname{argmin}} = \bar{R}(T)$
5: Initialize $T = T_1$, $k = 1$.

6: **procedure** (Iteration)
7: **while** $|T| > 1$ **do**:
8: Calculate $\alpha_{k+1} = \min_{t \in T} \frac{R(t) - \bar{R}(T_t)}{|T_t| - 1}$
9: Prune all T_t of T s.t $\bar{R}(T_t) + \alpha_{k+1}|T_t| = R(t) + \alpha_{k+1}$
10: Select the pruned subtree T_{k+1}
11: $T = T_{k+1}$, $k = k + 1$
12: **procedure** (Outputs)
13: Trees $T_1 \succ ... \succ T_K = \{t_1\}$
14: Parameters $(0 = \alpha_1; ...; \alpha_K)$

In fact, this algorithm comes from the following result ennounced by Breiman in [12] which states that for a maximal tree T_{max}, it exists an increasing sequence $(0 = \alpha_1; ...; \alpha_K)$ associated to a nested sequence of pruned trees $T_{max} \succeq T_1 \succ ... \succ T_K = \{t_1\}$ such that $1 \leq k \leq K$:

$$\forall \alpha \in [\alpha_k, \alpha_{k+1}[, \; T_k = \underset{T \preceq T_{max}}{\operatorname{argmin}} \; C_\alpha(T)$$
$$and \; \alpha \geq \alpha_K, \; T_K = \underset{T \preceq T_{max}}{\operatorname{argmin}} \; C_\alpha(T).$$

Finally, the best trees is obtained by optimizing the value of α, usually done by cross-validation [66].

Since the first version of CART, many works have been conducted around regression trees. In [48], the author details in his review that due to the nature of regression trees two research topics have been considered: those using a least-squares approach by considering the squared loss error, and those that act on the loss function directly. Friedman in [30] proposes a variant of CART called MARS, which leads to a more regular predictor in comparison with a piecewise constant estimate. It also worth noting that CART have been highly extended to survival data analysis - a synthesis can be found in [8] and references therein.

While regression trees dispose of natural ways for handling missing values and are computationally efficient, these methods tend to be unstable, i.e sensitive to the variability of the observations. Methods presented in the following are based on the so called ensemble methods [20]. Simply speaking, these methods use weak learners and aggregate them to predict the considered quantity of interest. Weak learners are built according to several methods, such as random sampling on the variables or observations for examples. These methods are in particular useful to increase the approximation capabilities of unstable learners such as trees. A simple way to aggregate weak learners is to use the average of their forecasts for ex-

ample but other techniques exist as we will show in the following subsections.

2.1.1.2.2 Bagging

The first ensemble method to be considered is the Bagging (standing for Boosting AGGregatING). Introduced by Breiman in [9], Bagging improves predictions capabilities of regression trees by considering different training sample and them aggregating the corresponding estimator. Breiman shows that these aggregated estimators are particularly efficient for regression trees.

Let us describe the procedure of Bagging:

- *Step 1:* From the full sample $\mathbf{Z} = \{(\mathbf{X}_1, \mathbf{Y}_1), ..., (\mathbf{X}_n, \mathbf{Y}_n)\}$, we build B bootstrap samples of size n by drawing with replacement from $\mathbf{Z}$. Hence, we have at our disposal B samples, and each sample is denoted by $\mathbf{Z}^b,\ b = 1, ..., B$;
- *Step 2:* For each sample $\mathbf{Z}^b,\ b = 1, ..., B$, one build $\hat{f}^b$. We now have a collection $\{\hat{f}^1, ..., \hat{f}^B\}$ of predictors;
- *Step 3:* Considering that we have a new observation $\mathbf{x}^*$ at our disposal we have not learned on or not seen before, the prediction for this new observation will be the aggregation of the collection of the predictors :

$$\hat{f}_{Bagging}(\mathbf{x}^*) = \frac{1}{B}\sum_{b=1}^{B} \hat{f}^b(\mathbf{x}^*) \tag{2.10}$$

As highlighted by Hastie and Tibshirani in [28], Equation (2.10) is a "Monte-Carlo estimate of the true bagging estimate, approaching it as $B \to \infty$" and is a good estimator when the true estimate $\hat{f}$ is nonlinear. This variability through bagging, although it helps improving accuracy and stability of unstable models such as trees, tends to hide the interpretability of bagged models and it makes it difficult to study their mathematical properties [28, 5]. An other drawback of this methods is the fact that predictors are not independent. As a consequence, much efforts have been made on building trees as much independent as possible to increase the capabilities of such strategy. This is the case in particular of Random Forest [11].

2.1.1.2.3 Random Forest

Random Forest, introduced by Breiman in [11], is based on bootstrap sampling and CART. Random Forest is part of the ensemble methods: it

builds several weak learners (trees, which have low biais and high variance) and gather them into a more stable model. Each tree is an estimator of the underlying function and built on a variation of the training set. As a consequence, each estimator leads to different results. For a new observation, the prediction is then the average value of all the predictions of all predictors as in Bagging.

Random Forests have been and are still widely studied. Many reviews have been realized, to name recent ones, the study [34] propose a concise development of Random Forest, giving theoretical properties, variants and examples. In the study [5] the authors give the most recent scientific advances of Random Forest, giving detailed practical explanations for people willing to go into this algorithm as well as the most recent extensions. In the book [28], the authors give also a solid basis of understanding Random Forest.

The main idea of Random Forest is to build trees as much independent as possible for increasing the capabilities of prediction. The procedure performed in Random Forest is the same as Bagging. The only difference lays in tree building procedure. Indeed, to increase the independence of trees, a sampling procedure is performed at each node on the variables, i.e we select randomly m variables among the p and the optimal splitting criteria is defined through these m drawn variables. Trees grow to the maximal size (i.e that the number of observations in the terminal leaves reaches a certain threshold) and are not necessarily pruned. As in Bagging, we get a collection $\{\hat{f}^1, ..., \hat{f}^B\}$ of predictors that are aggregated by equation (2.10) to give the prediction of a new observation $\mathbf{x}^*$.

Besides its good properties of generalization, Random Forest takes advantage of the *Out-Of-Bag Error*: already existing for the Bagging, the goal is to estimate the generalization error from the training data. As explained in [28]: *"For each observation, construct its random forest predictor by averaging only those trees corresponding to bootstrap samples in which the observation did not appear"*. Hence, we can stop the learning process when the OOB error settles down. The OOB error can be used also for quantifying a variable importance within the forest [11].

Numerous variants and extensions of Random Forest have been proposed in the last twenty years, to name a few: the Extremely Randomized Trees [35] which is very similar to the Random Forest algorithm but there is not bagging procedure. Trees are built on the full training sample, m variables are drawn from the p, and for each of them is picked randomly a threshold being used as a split candidate. The best candidate is kept and the procedure is repeated for the other nodes. One shall also mention

the Random Survival Forest [38]developed for survival analysis. An other important development is the expansion of Random Forest to Online Learning in [60] and upgraded in [43, 51]. Among all the recent papers dealing with Random Forests, the one published by [2] and called *The Generalized Random Forest* is of particular interest. Indeed, in this study the authors keep the most of Random Forest but considered that the forest is a type of adaptive nearest neighbor estimator using weights such as Boosting. This assumption allows to prove in a wider framework the consistency and asymptotic normality of their estimator notably.

2.1.1.2.4 AdaBoost

One thing that Bagging does not take into account is that each observation is not equally susceptible to be drawn randomly from the training set. Most of the time, we cannot assure this condition. As explained by [21]; "*in boosting, the probability of a particular example being in the training set of a particular machine depends on the performance of the prior machines on that example*". In other words, if a machine (a model) is able to predict and learn properly an observation, we do not need to learn more about it, but on observations which are difficult to learn on. Thus, these last ones will be more likely to be picked in a boosting sample. Adaboost was first introduced by [26, 27], and the following is a slightly modified version by [21] called AdaBoost.R2.

Algorithm 3 AdaBoost.R2 [21]

procedure (Inputs)
    $\mathbf{Z} = \{(\mathbf{X}_1, \mathbf{Y}_1), ..., (\mathbf{X}_n, \mathbf{Y}_n)\}$: training data
procedure (Initialization)
    $w_i = 1$, $i = 1, ..., n$: weight of each observation
    $p_i = \frac{w_i}{\sum w_i}$, $i = 1, ..., n$: probability to be drawn w.r.t w_i
    Set $\overline{L} = 1$: threshold for the loss function $L \in [0, 1]$
    Set $b = 1$
procedure (Iteration)
    while $\overline{L} > 0.5$ **do**:
        Draw a sample $\mathbf{Z}^b$ of size n from $\mathbf{Z}$ w.r.t p_i
        Build a model $\hat{f}^b$ on $\mathbf{Z}^b$

Calculate a loss L_i for each observation. The loss may be of any form as long as $L \in [0, 1]$
Calculate the average loss: $\overline{L} = \sum_{i=1}^{n} L_i p_i$
Assessment of the confidence in the predictor by calculating $\beta_b = \frac{\overline{L}}{1-\overline{L}}$
Update the weights $w_i \rightarrow w_i \frac{\beta_b^{1-L_i}}{c_b}$, with c_b being a normalizing constant
Update the probability p_i
$b = b + 1$
procedure (Outputs)
Outputs of each machine $\hat{f}^b$ are then weighted with $\ln \frac{1}{\beta_b}$, and the predictor is the (weighted) median

This ensemble technique can be used with Random Forest and Decision Trees Regressors. Besides, the main advantage of this algorithm is that it does not need to be calibrated. Nevertheless, it tends to overfit in attendance of noise and especially outliers. Indeed, AdaBoost will focus on outliers more than on the "average" behavior of the data leading to poor results. In order to deal with this likely drawback, Solomatin et al. propose *Adaboost.RT* in [64] for filtering observations which have a relative error exceeding a certain threshold leading to more stable results.

2.1.1.2.5 Gradient Boosting

The gradient boosting, intuited by [10] and developed by [29], is like every other boosting method: it combines weak learners $\hat{f}_m,\ m = 1, ..., M$. The goal stays the same, to explain $\mathbf{Y}$ by a function of $\mathbf{X}$.

As detailed in [28], "boosting is a way of fitting an additive expansion in a set of elementary "basis" functions". In our case, we use regression trees (CART). Hence, we consider that a tree is denoted by $T(x; \gamma)$, where γ corresponds to the parameters linked to the tree (splits and leaves values in particular). In other words, $\hat{f}$ has the following form:

$$\hat{f}(x) = \sum_{i=1}^{M} T(x; \gamma_i) \tag{2.11}$$

Originally, the idea of boosting lies in the definition of the Forward Stagewise Additive Modeling. Instead of tuning parameters of a tree, we iteratively add an other tree to the previous one to increase its capabilities by learning on the residual. Consider the loss function L being the squared error between a real observation y and its estimation $\hat{f}(x)$. The idea behind

the Gradient Boosting is to build a tree at each iteration as close as possible to the gradient of the loss function. The name of "gradient" comes from the fact that the gradient of the squared error is the negative residual (see [29] and [45]). Indeed, using the following loss function $L(y, f(x)) = \frac{1}{2}(y-f(x))^2$, its gradient is expressed as $\frac{\partial L(y,f(x))}{\partial f(x)} = f(x) - y$. The generic gradient tree-boosting algorithm is presented bellow

Algorithm 4 Gradient Tree Boosting Algorithm [28]

procedure (Initialization)
  $\hat{f}_0(x) = \underset{\gamma}{\text{argmin}} \sum_{i=1}^{N} L(y_i, \gamma)$
procedure (Iteration)
  for $m = 1 \ldots M$ **do**:
    for $i = 1 \ldots n$ **do**: $r_{im} = -\left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f=\hat{f}_{m-1}}$
    Fit a regression tree T_m to the targets r_{im} giving the terminal nodes (leaves) $l_{jm},\ j = 1, ..., J_m$
    for $j = 1 \ldots J_m$ **do**: $\gamma_{jm} = \underset{\gamma}{\text{argmin}} \sum_{i \in \{i \mid x_i \in l_{jm}\}} L(y_i, \gamma)$,
    Set $\gamma_m = (\gamma_{1m}, ..., \gamma_{J_m m})$
    Update $\hat{f}_m(x) = \hat{f}_{m-1}(x) + T_m(x; \gamma_m)$
procedure (Outputs)
  $\hat{f}(x) = \hat{f}_M = \sum_{i=1}^{M} T_m(x; \gamma_m)$,

Most of the time, improvements of this algorithm take place in the choice of the loss function or in the splitting node strategy. In the recent years, two algorithms stood out: XGBoost [14] and LightGBM [40]. These two methods are part of the extreme boosting family, that is to say that they belong to gradient boosting methods but are highly computationally efficient.

2.1.2 Sparse high dimensional parametric model: the Greedy algorithm

Boosting algorithms and Greedy algorithms are not so different. Indeed for Boosting, we increase the number of estimators by adding a slightly different tree from the previous one. For the Greedy algorithms, the idea is to work on an overcomplete dictionary of basis functions (i.e $p >> n$) by selecting automatically the best ones to solve the regression problem. At the end, the form of the final estimator is equivalent in the two cases, i.e a linear combination of functions. Although introduced for statistical estimation, Greedy algorithms have been used in many different fields such signal processing [17] or time series [3], and have been extensively studied in [67, 19, 4, 62]. For more details, we refer to [4, 62], where the authors make a review of the different existing algorithms and their properties.

Let us consider a finite dictionary of functions $\mathcal{G} := \{\mathbf{g}_1, ..., \mathbf{g}_N\}$ such that $\mathbf{g}_k : \mathbb{R}^p \to \mathbb{R},\ k = 1, ..., N$. We aim to estimate f in (2.2) by a finite linear combination of functions, i.e $\hat{f}(\mathbf{x}) = \sum_{k=0}^{K} b_k \mathbf{g}_k(\mathbf{x}),\ \sum_{k=0}^{K} |b_k| \leq B, \mathbf{x} \in \mathcal{X}$. $\mathcal{G}$ being known, the coefficients $b_k,\ k = 1, ..., K$ are estimated from the observations $\{(\mathbf{X}_i, \mathbf{Y}_i),\ i = 1, ..., n\}$. One must notice that f is not necessarily linear. Indeed, if we have a sufficient number of functions $\mathbf{g}_k$ at our disposal, we can estimate nonlinear functions with a linear combination of functions due to their complementarity [4].

Several algorithm have been developed over time to deal with this problem. One shall start with the procedure of the Pure Greedy Algorithm. The Pure Greedy Algorithm (also called L_2-Boosting, or Matching Pursuit [49, 17]) which is a Boosting algorithm. At each iteration, it fits the basis function which reduces the most the residual. The main difference stands in the shrinkage parameter (fixed to 0.1 according to [13]) that controls the parsimonious behavior of the algorithm. In this algorithm, this is the user who chooses the number K of basis functions that the model will be composed of. By setting $\mathbf{g}_k = \mathbf{g}_k(\mathbf{X}) = (\mathbf{g}_k(\mathbf{x}_1), ..., \mathbf{g}_k(\mathbf{x}_n)),\ k = 1, ..., N$, and using the following empirical inner product

$$\langle \mathbf{Y}, \mathbf{g}_k \rangle_n := \frac{1}{n} \sum_{i=1}^{n} \mathbf{Y}_i \mathbf{g}_k(\mathbf{x}_i) \text{ and } |\mathbf{g}_k|_n^2 := \langle \mathbf{g}_k, \mathbf{g}_k \rangle_n,$$

we have the following algorithm

Algorithm 5 Pure Greedy Algorithm [62]

procedure (Initialization)
    $\hat{f}_0 := 0$
    $K \in \mathbb{N}$
    $\nu \in (0, 1]$
procedure (Iteration)
    for $k = 1 \ ...K$ **do**:
        Solve $s(k) := \underset{j}{\text{argmax}}\ |\langle \mathbf{Y} - \hat{f}_{k-1}, \mathbf{g}_{(j)} \rangle_n|$
        Set $h_k(\mathbf{X}) := \langle \mathbf{Y} - \hat{f}_{k-1}, \mathbf{g}_{s(k)} \rangle_n \mathbf{g}_{s(k)}$
        Update $\hat{f}_k := \hat{f}_{k-1} + \nu h_k(\mathbf{X})$
procedure (Outputs)
    $\hat{f}_K$

As pointed in the review [62], this algorithm can be seen as variant of the Least-Angle Regression algorithm (LARS) [22]) when $\nu \underset{k \to +\infty}{\to} 0$. The

book [28] provides a good basis of understanding for the LAR method and its derivatives.

To avoid the bad approximation properties and the unfortunate calibration of the shrinkage parameter in the PGA, we consider in our studies the Orthogonal Greedy Algorithm (also known as the Matching Pursuit Algorithm [49, 17]). It has basically the same criterion as the PGA by choosing the number of basis functions to be used. Additionally at each iterations, it re-estimates the regression coefficients by using the OLS on the span of the subspace built with the selected basis functions. Hence, contrary to the PGA, we do not need to use a shrinkage parameter at each iteration:

Algorithm 6 Orthogonal Greedy Algorithm [62]

procedure (Initialization)
  $\hat{f}_0 := 0$
  $K \in \mathbb{N}$
procedure (Iteration)
  for $k = 1 \ldots K$ **do**:
    Solve $s(k) := \underset{j}{\operatorname{argmax}} \, |\langle \mathbf{Y} - \hat{f}_{k-1}, \mathbf{g}_{(j)} \rangle_n|$
    $P^k_{\mathbf{X}} :=$ OLS operator on $span\{\mathbf{g}_{s(1)}, \mathbf{g}_{s(2)}, ..., \mathbf{g}_{s(k)}\}$
    Update $\hat{f}_k := P^k_{\mathbf{X}} \mathbf{Y}$
procedure (Outputs)
  $\hat{f}_K$

This algorithm appears to perform very well but suffers if the dictionary contains too many variables, due to the OLS resolution particularly.

Many variant of these algorithms have been developped. To name a few, the Stepwise Projection Algorithm [4] has similar properties as the OGA. The Relaxed Greedy Algorithm [62] does not solve an OLS problem at each iteration and contrary to the PGA, this is not the h_k functions which are shrunk, but the previous estimation. By doing so, it lets space to (it relaxes) h_k functions to improve the model. There exist also constrained versions of the previously exposed Greedy algorithm: the Constrained Greedy Algorithm [62, 4] and the Frank-Wolfe Algorithm [62, 25]. Basically, these two algorithms are based on the same hypothesis as the ones above, the main difference is that the optimization problem solved at each iteration is done with respect to the following constraint : $\sum_{k=0}^{K} |b_k| < B < \infty$.

No matter the Greedy algorithm chosen, the user shall define the maximum number of variables to be picked from the dictionary. Even if this choice can be set arbitrarily, it can be optimized through Cross-Validation.

Nevertheless, these algorithms will tend to pick more variables than appropriate. This is especially the case of the OGA. Hence, a simple solution to avoid this drawback is to use an AIC or BIC criteria. Else, we refer to stopping rules studies and implemented for these iterative algorithms in [46].

2.2 Classical deformation models

When dealing with a large set of curves, it may be useful to use a parsimonious representation. Indeed, a sparse representation may lead to a better understanding of the variability within the population notably. One way to perform such reduction consists in considering that the set of curves has been obtained by deforming a template. This point of view is usually called curve registration and has been widely studied in statistics. This deformation may be achieved through the use of a parametric or a nonparametric model of transformations. The parametric transformation takes into account some prior knowledge on the curves (their form, features, etc.) contrary to a nonparametric transformation model. Two main statistical tasks are generally considered: the estimation of the template, and the estimation of the transformation parameters (for the parametric case) or the transformation functions (for the nonparametric case). In this section we will detailed two classical deformation models. The first is a parametric model of transformations: the Shape Invariant Model. The second subsection will be dedicated to a nonparametric approach: the Dynamic Time Warping.

2.2.1 Shape Invariant Model

Introduced in [44], the Shape Invariant Model (SIM) considers that the set of curves has been obtained by deforming a template (also called the reference curve) through a parametric deformation model. The authors estimate the deformation parameters and the unknown model function at the same time for the considered parametric family.

In the framework of the SIM, we observe n curves $y_i(x), i = 1, ..., n$ and we consider that these curves are represented by the following nonlinear mathematical model

$$y_i(x) = f(\theta_i; x) + error \tag{2.12}$$

where

$$f(\theta_i; x) = \theta_{0,i} + \theta_{1,i}\tilde{y}([x - \theta_{2,i}]/\theta_{3,i}), \tag{2.13}$$

with

- $\tilde{y}$ is the reference curve which will be deformed to fit y_i;
- $\theta_i = (\theta_{0,i}, \theta_{1,i}, \theta_{2,i}, \theta_{3,i})$, the vector of transformation parameters. $\theta_{0,i}$ is here to correct the offset of the template and $\theta_{1,i}$ is the scaling parameter. $\theta_{2,i}$ and $\theta_{3,i}$ are the parameters acting on the abscissa of the reference curve;

Hence, supposing that the reference curve is known and that the curves y_i have been observed at the points $x_{i,1}, ..., x_{i,p_i},\ i = 1, ..., n,$, one can solve easily the following nonlinear regression problem for each curve $y_i,\ i = 1, ..., n$

$$(\mathcal{P}) : \min_{\theta} \sum_{j=1}^{p_i} (y_i(x_{i,j}) - f(\theta; x_{i,j}))^2 \tag{2.14}$$

Unfortunately, the reference curve is most of the time unknown and one must estimate it to estimate the deformation parameters θ. In [44], the authors propose the following method to estimate the template. Choose reasonable m points $\tilde{x}_1, ..., \tilde{x}_m$ (not necessarily the same as $x_{i,1}, ..., x_{i,p_i}$). Assume that $\tilde{y}$ belongs to a collection of normalized shape functions and that the derivative of f outside some finite interval is negligible. Then, using

$$\tilde{y}(x) = \begin{cases} \sum_{k=1}^{m-1} (\frac{\tilde{y}_{k+1} - g_k}{\tilde{x}_{k+1} - \tilde{x}_k}[x - \tilde{x}_k] + \tilde{y}_k)\Delta_k(x),\ for\ \tilde{x}_1 \leq x \leq \tilde{x}_m \\ \tilde{y}_1,\ for\ x \leq \tilde{x}_1 \\ \tilde{y}_m,\ for\ x > \tilde{x}_m, \end{cases} \tag{2.15}$$

the function $\tilde{y}$ is represented by the vector $\tilde{\mathbf{y}} := (\tilde{y}_1, ..., \tilde{y}_m)$. As a consequence, giving a first initial value for $\hat{\theta}$ one can rewrite (2.14) for all curves to estimate $\tilde{\mathbf{y}}$ by solving

$$\hat{\tilde{\mathbf{y}}} = \underset{\tilde{\mathbf{y}} \in \mathbb{R}^m}{\operatorname{argmin}} \sum_{i=1}^{n} \sum_{j=1}^{p_i} (y_i(x_{i,j}) - f(\hat{\theta}(\tilde{\mathbf{y}}); x_{i,j}))^2. \tag{2.16}$$

Solving (2.16) to estimate $\tilde{\mathbf{y}}$ allows to update and give a new estimation of $\hat{\theta}$. We can then repeat this procedure to update the estimation of $\hat{\tilde{\mathbf{y}}}$. Accoriding to the authors, few iterations are needed to estimate both the parameters of deformation and the template function.

Many extension of the SIM have been developed. In [41], Kneip considers a general non linear parametric regression model with unknown template function. Many authors have worked on the SIM estimating either the template, the parameters or both. We refer to [41, 47, 37, 39, 31] for more details and examples on the SIM. More recently, these estimation topics have seen a

resurgence pushed by applications in image and signal processing. We refer for example to [36, 50, 1, 31, 6, 7, 33, 68] for models and techniques related to signal processing problems and the multidimensional extension of SIM defined on the plane.

2.2.2 Dynamic Time Warping

The Dynamic Time Warping (DTW) refers to the alignment of features of curves. In other words, the goal is to act on the range of the curves to align peaks and valleys between them. In the engineering literature, it appears first in [61] and was mathematically studied by Ramsay et al. in [56, 55, 57, 58]. The authors define a nonparametric technique to identify the smooth monotone transformation acting on the abscissa. In a more formal way, let us consider that we have at our disposal curves $y_j,\ j = 1, ..., J$ defined such that

$$\begin{aligned} y_j : [0, T_j] &\to \mathbb{R} \\ t &\to y_j(t) \end{aligned}$$

and a reference curve (also called pattern) denoted by $\tilde{y}$ and defined on $[0, \tilde{T}]$. Each curve will be transformed, registered, matched to this reference curve $\tilde{y}$ using the following model:

$$\tilde{y}(t) = y_j(h_j(t)) + \epsilon_j \tag{2.17}$$
$$= y_j \circ h_j(t) + \epsilon_j \tag{2.18}$$

with ϵ_j being centered and relatively small with respect to y_j. h_j is the transformation of t corresponding to y_j (also called the *time warping* function). Note that finding h_j is a constrained problem due to the fact that $h(0) = 0$ and $h_j(T_j) = \tilde{T}$.

The idea of the DTW is to estimate this function h. As explained in [56, 55], h is supposed to be a solution of a differential equation defining monotone functions having the properties of being strictly increasing ans that its first derivative is smooth and bounded almost everywhere. Using the notation D^d for the derivative of order $d,\ d > 0$ and the partial integration operator D^{-1} such that $D^{-1}f(t) = \int_0^t f(s)ds$, Ramsay in [ref] states the following theorem defining the class of functions of h:

Theorem. *Every function h solution of this class is representable as either*

$$h = C_0 + C_1 D^{-1}\{\exp(D^{-1}w)\}$$

or as a solution of the homogeneous linear differential equation

$$D^2h = wDf$$

where w is a Lebesgue square integrable function and C_0 and C_1 are arbitrary constants.

In fact, w refers to the relative curvature of h and is represented by a linear combination of B-splines functions

$$w(t) = \sum_{k=0}^{K} c_k B_k(t).$$

Hence, in the DTW, we solve the following problem for all $j = 1, ..., J$:

$$(\mathcal{P}_2) : \min_w \int ||\tilde{y}(t) - y_j(h_j(t))||^2 dt + \lambda \int w^2(t) dt$$

This is of course not realistic: most of the time, curves are observed discretely, and the reference curve is not known on the full domain. Hence, the integral is replaced by a sum of squared errors and the reference curve used in the first instance is the cross-sectional average. As explained in [55], the estimation of h can be done in a two-way stage procedure: by initially giving a value to the coefficients $c_k,\ k = 1, ..., K$, C_0 and C_1 can be estimated with a linear regression and then using a non-linear least squares procedure allows to estimate the coefficients $c_k,\ k = 1, ..., K$. Few iterations are needed to get a sufficient convergence.

As pointed out in [57, 58], many extensions of the DTW have been developed. In particular, the authors in [42] propose to estimate the warping function by local regression. For complementary material, other dynamic programming strategies can be found (see for example [63, 52]).

Concerning the regression method, we have seen in the first subsection in particular that regression trees-based methods are built on the subdivision of space to approximate the function f. Boosting methods clearly outperform classical ones thanks to the variety of estimators. For Greedy algorithms, they find their strengths in an overcomplete dictionary. In a way, it considers that in a multitude of variables or basis functions, a proper selection can be linearly arranged to estimate a complicated function f. Finally, these two approaches (Boosting algorithms such as XGBoost or LightGBM) and Greedy algorithms are based on the same "variability" philosophy by approximating with a huge number of estimators. For the tree-based methods, each estimator partition the problem space differently, hence each estimator can be seen as a representation of the initial problem, and at the end, the global representation of all the combined estimators

is not far from the reality. Concerning the Greedy algorithms, at each iteration, a new variable is selected, and this variable is supposed to strengthen weaknesses of the previously selected ones.

In the last section, we have seen that a deformation model is an efficient way of giving a parsimonious representation of the data under study when a parametric method is used. Nevertheless, even if it eases the understanding of the variability within the population, the reliability of the extracted information is conditioned by the choice of the proper template function. A nonparametric method such as the Dynamic Time Warping does not suffer from such limitation but looses in interpretability. Finally, note that both the SIM and the DTW are transformations acting independently on the abscissa axis and the ordinate axis.

Bibliography

[1] Allassonniere, S., Bigot, J., Glaunes, J., Maire, F., F., R.: Statistical models for deformable templates in image and shape analysis. Annales Mathematiques Blaise Pascal **20(1)**, 1–35 (2013)

[2] Athey, S., Tibshirani, J., Wager, S., et al.: Generalized random forests. The Annals of Statistics **47**(2), 1148–1178 (2019)

[3] Audrino, F., Bühlmann, P.L.: Volatility estimation with functional gradient descent for very high-dimensional financial time series. In: Research report/Seminar für Statistik, Eidgenössische Technische Hochschule (ETH), vol. 99. Seminar für Statistik, Eidgenössische Technische Hochschule (ETH) (2001)

[4] Barron, A.R., Cohen, A., Dahmen, W., DeVore, R.A., et al.: Approximation and learning by greedy algorithms. The annals of statistics **36**(1), 64–94 (2008)

[5] Biau, G., Scornet, E.: A random forest guided tour. Test **25**(2), 197–227 (2016)

[6] Bigot, J., Gadat, S.: A deconvolution approach to estimation of a common shape in a shifted curves model. Ann. Statist. **38**(4), 2422–2464 (2010). DOI 10.1214/10-AOS800. URL https://doi.org/10.1214/10-AOS800

[7] Bigot, J., Gamboa, F., Vimond, M.: Estimation of translation, rotation, and scaling between noisy images using the fourier–mellin transform. SIAM Journal on Imaging Sciences **2**, 614–645. ISSN 2936-4954

[8] Bou-Hamad, I., Larocque, D., Ben-Ameur, H., et al.: A review of survival trees. Statistics Surveys **5**, 44–71 (2011)

[9] Breiman, L.: Bagging predictors. Machine Learning **24**, 123–140 (1996)

[10] Breiman, L.: Arcing the edge. Technical Report (486) (1997). Statistics Department, University of California

[11] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)

[12] Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: Classification and Regression Trees. Wadsworth, Belmont, CA (1984)

[13] Buehlmann, P., et al.: Boosting for high-dimensional linear models. The Annals of Statistics **34**(2), 559–583 (2006)

[14] Chen, T., Guestrin, C.: Xgboost: A scalable tree boosting system. In: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, pp. 785–794. ACM (2016)

[15] Cleveland, W.S.: Robust locally weighted regression and smoothing scatterplots. Journal of the American statistical association **74**(368), 829–836 (1979)

[16] Cleveland, W.S., Devlin, S.J.: Locally weighted regression: an approach to regression analysis by local fitting. Journal of the American statistical association **83**(403), 596–610 (1988)

[17] Davis, G., Mallat, S., Avellaneda, M.: Adaptive greedy approximations. Constructive approximation **13**(1), 57–98 (1997)

[18] Devlin, S.: Locally-weighted multiple regression: Statistical properties and its use to test for linearity. Bell Communications Research technical memorandum (1986)

[19] DeVore, R.A., Temlyakov, V.N.: Some remarks on greedy algorithms. Advances in computational Mathematics **5**(1), 173–187 (1996)

[20] Dietterich, T.G.: Ensemble methods in machine learning. In: International workshop on multiple classifier systems, pp. 1–15. Springer (2000)

[21] Drucker, H.: Improving regressors using boosting techniques. Proceedings of the Fourteenth International Conference on Machine Learning pp. 107–115 (1997)

[22] Efron, B., Hastie, T., Johnstone, I., Tibshirani, R., et al.: Least angle regression. The Annals of statistics **32**(2), 407–499 (2004)

[23] Fan, J., Gasser, T., Gijbels, I., Brockmann, M., Engel, J.: Local polynomial regression: optimal kernels and asymptotic minimax efficiency. Annals of the Institute of Statistical Mathematics **49**(1), 79–99 (1997)

[24] Fan, J., Gijbels, I.: Local polynomial modelling and its applications. Chapman and HallCRC (1996)

[25] Frank, M., Wolfe, P.: An algorithm for quadratic programming. Naval research logistics quarterly **3**(1-2), 95–110 (1956)

[26] Freund, Y., Shapire, R.: A decision-theoretic generalization of on-line learning and application to boosting. Proceedings of the second European Conference on Computational Learning Theory pp. 23–37 (1995)

[27] Freund, Y., Shapire, R.: Experiments with a new boosting algorithm, machine learning. Proceedings of the Thirteenth Conference pp. 148–156 (1996)

[28] Friedman, J., Hastie, T., Tibshirani, R.: The elements of statistical learning, vol. 1. Springer series in statistics New York (2001)

[29] Friedman, J.H.: Greedy function approximation: A gradient boosting machine (1999)

[30] Friedman, J.H., et al.: Multivariate adaptive regression splines. The annals of statistics **19**(1), 1–67 (1991)

[31] Gamboa, F., Loubes, J.M., Maza, E.: Semi-parametric estimation of shifts. Electron. J. Statist. **1**, 616–640 (2007). DOI 10.1214/07-EJS026. URL https://doi.org/10.1214/07-EJS026

[32] Gasser, T., Müller, H.G.: Estimating regression functions and their derivatives by the kernel method. Scandinavian Journal of Statistics pp. 171–185 (1984)

[33] Gassiat, E., Lévy-Leduc, C.: Efficient semiparametric estimation of the periods in a superposition of periodic functions with unknown shape. Journal of Time Series Analysis **27**(6), 877–910 (2006)

[34] Genuer, R., Poggi, J.M.: Arbres cart et forêts aléatoires, importance et sélection de variables (2017). URL https://hal.archives-ouvertes.fr/hal-01387654

[35] Geurts, P., Ernst, D., Wehenkel, L.: Extremely randomized trees. Machine learning **63**(1), 3–42 (2006)

[36] Grenander, U.: General pattern theory. Oxford Science Publications (1993)

[37] Hardle, W., Marron, J.S.: Semiparametric comparison of regression curves. The Annals of Statistics pp. 63–89 (1990)

[38] Ishwaran, H., Kogalur, U.B., Blackstone, E.H., Lauer, M.S., et al.: Random survival forests. The annals of applied statistics **2**(3), 841–860 (2008)

[39] Ke, C., Wang, Y.: Semiparametric nonlinear mixed-effects models and their applications. Journal of the American Statistical Association **96**(456), 1272–1298 (2001)

[40] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.Y.: Lightgbm: A highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, pp. 3146–3154 (2017)

[41] Kneip, A., Engel, J.: Model estimation in nonlinear regression under shape invariance. The Annals of Statistics pp. 551–570 (1995)

[42] Kneip, A., Li, X., MacGibbon, K., Ramsay, J.: Curve registration by local regression. Canadian Journal of Statistics **28**(1), 19–29 (2000)

[43] Lakshminarayanan, B., Roy, D.M., Teh, Y.W.: Mondrian forests: Efficient online random forests. In: Advances in neural information processing systems, pp. 3140–3148 (2014)

[44] Lawton, W., Sylvestre, E., Maggio, M.: Self modeling nonlinear regression. Technometrics **14**(3), 513–532 (1972)

[45] Li, C.: A gentle introduction to gradient boosting (2016). College of Computer and Information Science, Northeastern University. Available from: http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

[46] Li, F., Triggs, C.M., Dumitrescu, B., Giurcăneanu, C.D.: The matching pursuit algorithm revisited: A variant for big data and new stopping rules. Signal Processing **155**, 170–181 (2019)

[47] Lindstrom, M.J.: Self-modelling with random shift and scale parameters and a free-knot spline shape function. Statistics in medicine **14**(18), 2009–2021 (1995)

[48] Loh, W.Y.: Fifty years of classification and regression trees. International Statistical Review **82**(3), 329–348 (2014)

[49] Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on signal processing **41**(12), 3397–3415 (1993)

[50] McGuire, M.: An image registration technique for recovering rotation, scale and translation parameters. Tech. rep., NEC Tech Report (1998). URL http://casual-effects.com/research/McGuire1998ParameterRecovery/index.html

[51] Mourtada, J., Gaïffas, S., Scornet, E.: Minimax optimal rates for mondrian trees and forests. arXiv preprint arXiv:1803.05784 (2018)

[52] Müller, M.: Dynamic time warping. Information retrieval for music and motion pp. 69–84 (2007)

[53] Murthy, S.K.: Automatic construction of decision trees from data: A multi-disciplinary survey. Data mining and knowledge discovery **2**(4), 345–389 (1998)

[54] Nadaraya, E.A.: On estimating regression. Theory of Probability & Its Applications **9**(1), 141–142 (1964)

[55] Ramsay, J.O.: Estimating smooth monotone functions. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **60**(2), 365–375 (1998)

[56] Ramsay, J.O., Li, X.: Curve registration. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **60**(2), 351–363 (1998)

[57] Ramsay, J.O., Silverman, B.W.: Functional data analysis. Springer (2005)

[58] Ramsay, J.O., Silverman, B.W.: Applied functional data analysis: methods and case studies. Springer (2007)

[59] Ruppert, D., Wand, M.P.: Multivariate locally weighted least squares regression. The annals of statistics pp. 1346–1370 (1994)

[60] Saffari, A., Leistner, C., Santner, J., Godec, M., Bischof, H.: On-line random forests. In: 2009 ieee 12th international conference on computer vision workshops, iccv workshops, pp. 1393–1400. IEEE (2009)

[61] Sakoe, H., Chiba, S., Waibel, A., Lee, K.: Dynamic programming algorithm optimization for spoken word recognition. Readings in speech recognition **159**, 224 (1990)

[62] Sancetta, A., et al.: Greedy algorithms for prediction. Bernoulli **22**(2), 1227–1277 (2016)

[63] Senin, P.: Dynamic time warping algorithm review. Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA **855**(1-23), 40 (2008)

[64] Solomatine, D.P., Shrestha, D.L.: Adaboost. rt: a boosting algorithm for regression problems. In: 2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541), vol. 2, pp. 1163–1168. IEEE (2004)

[65] Sonquist, J.A., Morgan, J.N.: The detection of interaction effects: A report on a computer program for the selection of optimal combinations of explanatory variables. 35. Survey Research Center, Institute for Social Research, University of Michigan (1964)

[66] Stone, M.: Cross-validatory choice and assessment of statistical predictions. Journal of the royal statistical society. Series B (Methodological) pp. 111–147 (1974)

[67] Temlyakov, V.N.: Nonlinear methods of approximation. Foundations of Computational Mathematics **3**(1) (2003)

[68] Vimond, M.: Efficient estimation for homothetic shifted regression models. Université Paul Sabatier, Laboratoire de statistique et probabilités (2006)

[69] Watson, G.S.: Smooth regression analysis. Sankhyā: The Indian Journal of Statistics, Series A pp. 359–372 (1964)

Chapter 3

A case study : Influence of dimension reduction on regression trees-based algorithms - Predicting aeronautics loads of a derivative aircraft

Edouard Fournier*[1,2,3], Thierry Klein†[1,2], Stéphane Grihon‡[3]

[1]Institut de Mathématiques, UMR5219; Université de Toulouse; CNRS, UPS IMT, F-31062 Toulouse Cedex 9, France
[2]ENAC - Ecole Nationale de l'Aviation Civile, Université de Toulouse, France
[3]Airbus France 316, Route de Bayonne, Toulouse France

Résumé

Dans ce chapitre, nous donnons un aperçu des données et modèles qui seront étudiés dans la suite de la thèse. Il s'agit d'une étude préliminaire où nous cherchons à modéliser la chaîne de calculs des charges externes à partir de données. Cette étude est issue d'un projet *sprint* interne

*edouard.fournier@airbus.com
†thierry.klein@math.univ-toulouse.fr or thierry01.klein@enac.fr
‡stephane.grihon@airbus.com

à Airbus. Il a été montré dans un projet interne précédent que les techniques de régression à base d'arbres sont efficaces pour prédire les charges d'un avion dans le cas iso-masse (c'est-à-dire que la masse maximale au décollage est fixée). Ici, nous nous intéressons à la prédiction des charges dans un cas d'extrapolation de masse maximale au décollage.

Dans cet optique, nous avons à notre disposition quatre jeux de données. Chacun correspond à des avions de masses maximales au décollage différentes : 238 tonnes, 242 tonnes, 247 tonnes et 251 tonnes. Chaque jeu de données a été produit à partir d'un code numérique représentant la physique du problème. L'idée est de construire un modèle à partir des données de l'avion 238 tonnes et de prédire les charges des avions 242 tonnes, 247 tonnes et 251 tonnes. Ayant à notre disposition les entrées et sorties pour les différents avions, nous pouvons ainsi évaluer la performance en extrapolation des modèles.

Les entrées appartiennent à $\mathbb{R}^{25}$ et les sorties à $\mathbb{R}^{29}$. Dans un premier temps, nous allons considérer les cinq méthodes de régression à base d'arbres suivantes : les Decision Trees [5], les Random Forest [4], la méthode AdaBoost [11] appliquée aux Decision Trees ou aux Random Forest, le Bagging [2] et enfin le Gradient Boosting [3]. Puis nous allons utiliser différentes techniques de réductions de dimension. Nous avons quantifié l'influence des méthodes de réduction de dimension en extrapolation. Les méthodes de réduction de dimension utilisées sont : l'Analyse en Composantes Principales (ACP) euclidienne dans $\mathbb{R}^d$, à la transformation des courbes de charges par ajustement polynomial d'ordre p ou encore à un mélange de ces deux dernières méthodes. En tout, 48 combinaisons ont été testées. Il s'avère que la méthode Adaboost associée à Random Forest et à une ACP appliquées sur les sorties offre de bons résultats en termes de précision et temps de calculs pour estimer les charges.

Ce chapitre correspond à l'article du même nom publié [8].

3.1 Introduction

In aircraft industry, market needs evolve quickly in a high competitiveness context. This requires adapting a given aircraft model in minimum time considering for example an increase of range or of the number of passengers such as the A330 family in [1]. In our case study, variants concern the maximum take-off weight of a given aircraft model. Depending on the configuration, the computation of loads and stress, as defined in [14, 13], to resize the airframe is on the critical path of this aircraft variant definition:

this is a time consuming (approximately a year for a new aircraft variant) and costly process, one of the reason being the high dimensionality and the large amount of data. Big Data approaches such as defined by [12] is mandatory to improve the speed, the data value extraction and the responsiveness of the overall process. This study has been realized during a proof of value sprint project within Airbus to demonstrate the usefulness of statistics and machine learning approaches in the Engineering field. In a previous internal project, it has been shown that the family of regression trees [5] works well to predict loads for different aircraft missions in an interpolation context. Thus, we can formulate our problem in this way: is it possible to use dimensional reduction and regression trees-based algorithms to predict loads in an extrapolation context (i.e outside the design space of a certain weight variant) to improve the actual process?

3.1.1 Industrial context

An airframe structure is a complex system and its design is a complex task involving today many simulation activities generating massive amounts of data. Such is the case of the process of loads and stress computations for an aircraft (that is to say the calculations of the forces and the mechanical strains suffered by the structure) and can be represented as follows:

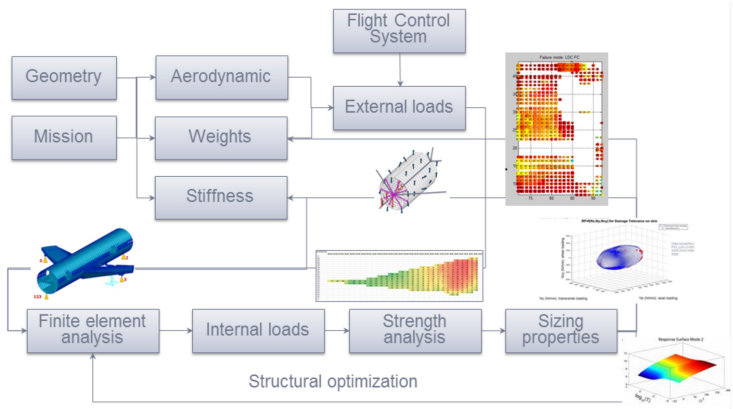


Figure 3.1: Flowchart for loads and stress analysis process

The overall process exposed in Figure 4.4 is run to identify load cases (i.e aircraft mission and configurations: maneuvers, speed, loading, stiffness...), that are critical in terms of stress endured by the structure and, of course, the parameters which make them critical. The final aim is to size and design the structure (and potentially to reduce loads in order to reduce the weight of the structure). Typically for an overall aircraft structure,

millions of load cases can be generated and for each of these load cases millions of structural responses (i.e how structural elements react under such conditions) have to be computed. As a consequence, computational times can be significant.

For a derivative aircraft, we can give some rough order of magnitudes in terms of quantities of produced data: External loads (10^6 of bytes); Weights: number of elements (10^4 of bytes); Internal loads: number of components by the number of external loads by the number of elements (10^{11} of bytes); Reserve Factors: number of internal loads by the number of failure modes (10^{12} of bytes). Hence, we easily reach 10^{18} to 10^{21} of bytes for a single derivative aircraft.

In an effort to continuously improve methods, tools and ways-of-working, Airbus has invested a lot in digital transformation and the development of infrastructures allowing to treat data (newly or already produced). The objective here is to exploit and adapt Machine Learning and optimization tools in the right places of the computational process. As pointed by [17], these techniques cover a large number of fields such as Internet and Business Intelligence but they can also benefit to the manufacturing industry (here aeronautics). The main industrial challenge for Airbus is to reduce lead time in the computation of loads and preliminary sizing of an airframe.

3.1.2 A simplistic load and stress model computation process example

In order to illustrate the process exposed in the previous subsection, let us consider a simplistic load model completed with equations calculating thickness used to correct the weight distribution of a wing structure similar to [6].

The structure contains a fuel tank at the wing tip with the dimensions Lf, Ctf, Cof as shown in Figure 3.2. The length of the wing is L, the chord length at wing root is Co and at the tip Ct. As a consequence, there are three different types of loads which affect the wing: the aerodynamic lift Q_{lift} (i.e the force which allows the aircraft to lift off and to maintain altitude) which depends on the length of the wing, the load factor and the total weight of the aircraft; the loads concerning the fuel and the fuel tank weight Q_{fuel} depending on the fuel weight and the dimension of the fuel tank; and the loads due to the wing structure $Q_{wingstructure}$ depending on the weight and the dimension of the wing. By adding these three types of loads, and providing the weight of the wing structure, the weights of the tank and the fuel contained, as well as the total weight of the aircraft and the load factor; Q_{total} provides the basis for calculating the shear force V

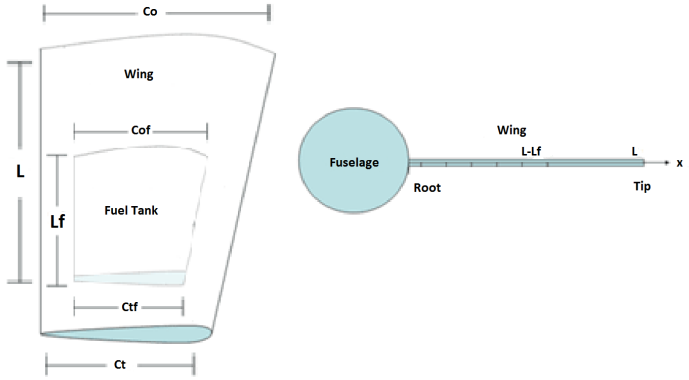


Figure 3.2: Scheme of the wing structure considered in the load model

(transverse forces near to vertical arising from aerodynamic pressure and inertia) and bending moment M (resulting from the shear forces) of the wing. The relations between these quantities are :

$$V(x) = -\int_0^L Q(x)dx,$$
$$M(x) = \int_0^L V(x)dx,$$

where x is the position along the wing. We consider that the wing is represented by a simplified rectangular box schematized by two parallel panels representing the covers (see Figure 3.3) : This is enough to distribute the fluxes induced by the bending moment.

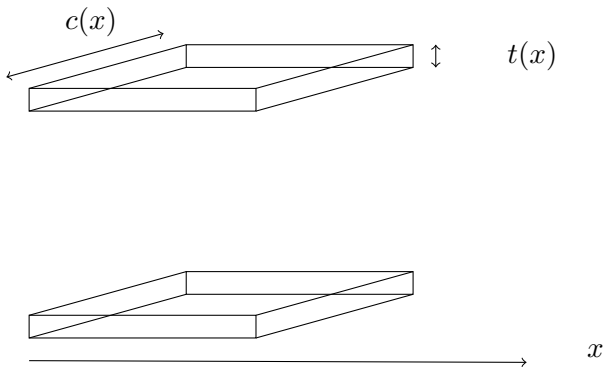


Figure 3.3: Form of the box (upper cover and under cover) of the wing

We can complete equations calculating thickness. Indeed, by considering the box has height $h(x)$ supposed linearly decreasing along the span, con-

sidering we must not exceed an allowable of σ_{max} tension and compression. Considering the fluxes in the wing covers are given by $N(x) = \frac{M(x)}{h(x)C(x)}$ thus we have the thickness distribution defined by:

$$t(x) = \frac{M(x)}{h(x)C(x)\sigma_{max}}$$

And by integrating we get the weight of the cover given by:

$$W_{cover} = 2\int_0^L \frac{M(x)}{h(x)C(x)\sigma_{max}}dx$$

Indeed, by considering that the wing takes the form of a box presented in Figure 3.3, by integrating $t(x)C(x)$ along x and by multiplying by 2ρ, where ρ is the density of the material used to fabricate the wing panels, we get the weight of the wing cover. More precisely, we obtain the minimum weight of the wing cover able to resist an allowable σ_{max} tension and compression. We assume that $W_{cover} = 30\%W_{wing}$, then we can extract the minimum weight of the wing structure able to resist an allowable σ_{max} tension and compression.

3.1.3 Data presentation

The data we have at our disposal are the aircraft parameters (features) which are used in the computing chain for calculating loads (outputs which correspond to moments and forces). We have data coming from the weight variant 238 tons (aircraft parameters and loads distribution along the wing); and we would like to predict those of the 242t and other weight variants (247t and 251t). All the different datasets have been previously computed and we use them to assess the capability of methods defined in the following sections to predict loads in such context. In fact, we hope to answer, by doing so, to the question: "What would the results have been if we had applied such a methodology to calculate the loads instead of the normal process for new weight variants?".

25 aircraft (A.C.) parameters play the role of features (lying in $\mathbb{R}$) of a load case and we would like to predict the associated loads (outputs) which are in $\mathbb{R}^k$. To simplify, we will focus on predicting bending moment along the wing which is, in our data, represented by a vector of size $k = 29$. In other words, each load case (i.e observation) is defined by its 25 features and its bending moment (output). The features are used to identify a typical aircraft event (maneuvers, gusts, continuous turbulences) with specific aerodynamic and weight conditions. Gusts are loads produced by environmental perturbations: sudden vertical or lateral wind blasts which

are required by certification organisms like EASA from statistical meteorological histories. Continuous turbulence cases are linked to the cumulative energy stored by the structure under a spectrum of random gusts. A typical maneuver is a 2.5g pull-up consisting in producing an increase aerodynamic lift by deflecting the elevator and increasing the angle of attach of the aircraft. This gives a bending moment close to the maximum value in competition with gust cases. The data base is constituted mainly by gusts (90% of all load cases) and we will focus on them. To begin, we shall focus on the 238t and 242t data before generalizing our results to other weight variants. A quick summary of the size of our different datasets is presented in Table 3.1:

Table 3.1: Description of the datasets

	238t(Train&Test)	242t(Validation)
Dimension data features	28391 rows x 25 col.	28391 rows x 25 col.
Dimension data outputs	28391 rows x 29 col.	28391 rows x 29 col.

In a more formal way, let be the 238t database of features defined by $\mathbf{X} = (X^1, ..., X^{25})$ where X^j are quantitative variables (i.e a A.C. parameter), and $X^j = (x_1^j, ..., x_{28391}^j)^T$. The 238t database of outputs is then defined by $\mathbf{Y} = (Y^1, ..., Y^{29})$ and $Y^j = (y_1^j, ..., y_{28391}^j)^T$. Aircraft parameters $\mathbf{X}$ (inputs) we have at our disposal in the training data base 238t are described in Table 3.2.

Contrary to the simplistic load calculation example, real simulations needs much more of information: the first ten variables are linked to the orientation of ailerons, spoilers and the rudder which are directional control surfaces (see Figure 3.4); the x-location of gravity center is an indicator concerning the location of the gravity center along the x-axis; the thrust is a calculated variable corresponding to the force which moves the aircraft forward (contrary to the drag force); and the load factors are global indicators which express the "amount of loads" the structure can withstand. All these features are processed by dynamic flight equations considering the flexible body behaviour of the aircraft through finite element models (Lagrange's equations): for further readings, we refer to [22].

Table 3.2: Description of the 238t dataset

Description	Distribution type	Mean	Std	Min	Max
Defl. Left inboard Elevator	Gaussian	0.015	0.034	-0.116	0.108
Stabilizer Setting	Mixture of Gaussian (2 modes)	-0.033	0.023	-0.093	0.0033
Defl. Spoiler 1 Left Wing	Bi Modal	-0.221	0.218	-0.436	0
Defl. Spoiler 2 Left Wing	Mixture of Gaussian (2 modes)	-0.266	0.262	-0.755	0.230
Defl. Spoiler 3 Left Wing	Mixture of Gaussian (2 modes)	-0.266	0.262	-0.755	0.230
Defl. Spoiler 4 Left Wing	Mixture of Gaussian (2 modes)	-0.266	0.262	-0.755	0.230
Defl. Spoiler 5 Left Wing	Mixture of Gaussian (2 modes)	-0.266	0.262	-0.755	0.230
Defl. Spoiler 6 Left Wing	Mixture of Gaussian (2 modes)	-0.266	0.262	-0.755	0.230
Defl. all speed inner Aileron	Gaussian	-0.029	0.086	-0.58	0.58
Defl. Low speed outer Aileron	Quadrimodal	-0.028	0.053	-0.157	0
Lower part Rudder Deflection	Gaussian	0	0.011	-0.072	0.072
Total A.C. Mass	Multimodal	195738	35428	135093	238000
Mach Number	Multimodal	0.716	0.19	0.372	0.93
True Airspeed	Multimodal	223	50	126	282
Altitude	Multimodal	6270	4519	0	12634
x-location of cg in % amc	Multimodal	0.297	0.114	0.140	0.42
Thrust(calculated)	Multimodal	131442	157160	0	415495
X-Load Factor	Gaussian	-0.020	0.107	-0.3	0.261
Y-Load Factor	Gaussian	0	0.08	-0.306	0.307
Z-Load Factor	Gaussian	1.024	0.43	-0.701	2.643
Fuel Tank mass TANK1L	Multimodal	392	1030	0	4341
Fuel Tank mass TANK2L	Multimodal	13008	12721	0	36295
Fuel Tank mass TANK3L	Multimodal	1883	1377	0	3087
Fuel Tank mass TANK1L	Multimodal	945	1029	0	2592
Left inner engine thrust	Multimodal	65721	78579	0	207747

The bending moment is calculated at 29 points along the wing - each point represents a station and stations are not equidistant (two more stations are located in the center wing box; we prefer to focus here on stations of the wing only). Thus Y^k represents the values of the bending moment taken at the k^{th} station. Through a change of coordinate system (aircraft system to wing system), we can easily plot bending moments (Figure 4.5):

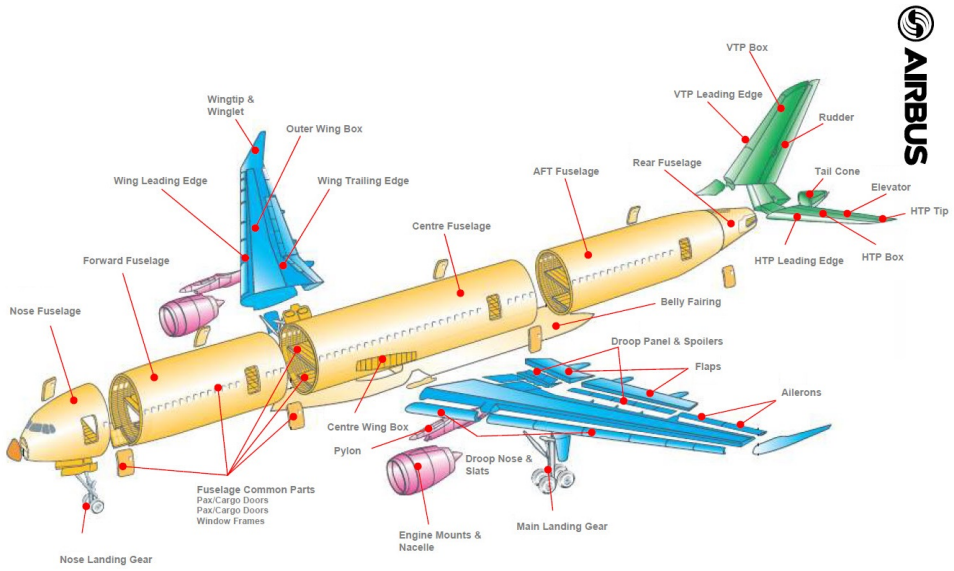


Figure 3.4: Airplane parts definition

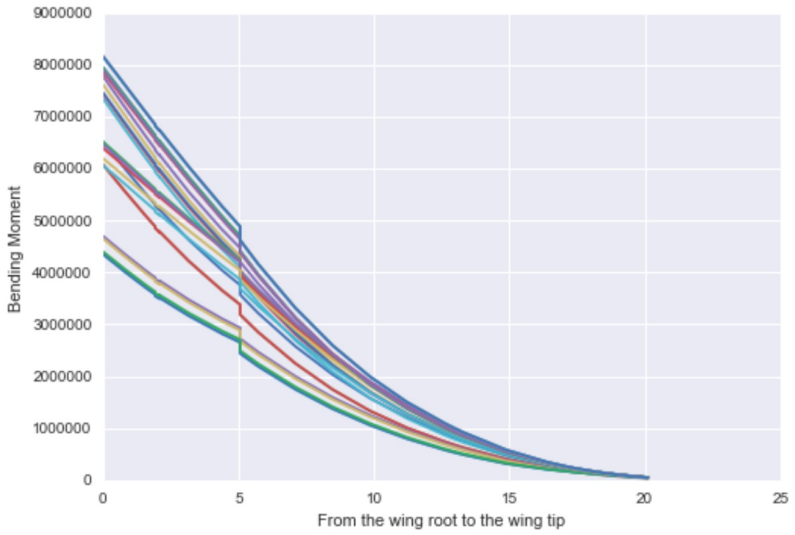


Figure 3.5: Examples of bending moments along the wing for different load cases

3.1.4 Industrial problem

Aircrafts (A.C.) have been developed for different maximum take-off weight (which is one of the many aircraft parameters used in the computing chain

to calculate the loads). Because the computation process exposed above for a new aircraft variant (a new weight variant in our case) can reach easily a year, the use of meta-models, optimization and statistic approaches such defined by [12] is mandatory to improve the speed and responsiveness of the overall process.

From this standpoint, we can expose the following problem: for each combination of A.C. parameters corresponding to a load case, and each load case being categorized into a load condition (family of load cases - gusts or maneuvers), can we give an estimation of the loads for different A.C. parameters for new weight variants (242t, 247t and 251t) knowing the loads of the weight variant 238t?

The mathematical problem of this project is an extrapolation problem. Is it possible to "extrapolate" loads of the 242 tons, 247t and 251t knowing loads of the 238t by using machine learning? To be more precise, can we find a function depending on aircraft parameters that allows us to estimate/extrapolate to 242t and other weight variants by learning from those of the 238t? In a previous project concerning loads, it has been shown that the family of regression trees works well on the data we have to deal with. As a consequence, different algorithms based on decision trees will be investigated. Besides, because of the dimension of our outputs, how do dimensional reduction techniques affect the capability of extrapolation of machine learning algorithms based on regression trees?

This paper is organized as follows: Section 2 is dedicated to the description of the three different techniques of dimension reduction we used in our study. Then in Section 3 we expose the different algorithms based on regression trees and finally we present in Section 4 our results.

3.2 Three dimensional reduction techniques

In order to improve the efficiency and speed of the modeling process, we compare several dimensional reduction techniques. We start by using a classical PCA on the inputs and also on the outputs. Then we consider a polynomial fitting and finally we mix the two methods. These dimensional reduction techniques will reduce the dimension of the output space. Each technique has been used on the 238t, and these allow us to reverse the technique to come back to the original output space easily.

3.2.1 Principal Components Analysis

In few words, the Principal Components Analysis (PCA), developed by [18] and formalized by [15] is a statistical method used to compress a matrix n

x p of quantitative variables into a smaller rank matrix. This method uses the variance-covariance matrix (or correlation matrix) to extract important factors (few in general) to represent observations in a smaller subspace. As a consequence, each observation is represented by coordinates into new components linked to these factors (this approach is similar to the SVD decomposition).

We apply the PCA in the space defined by the outputs (centered and reduced), and the Figure 3.6 shows the decline of the variance explained by each component as well as the cumulative percentage of the explained variance:

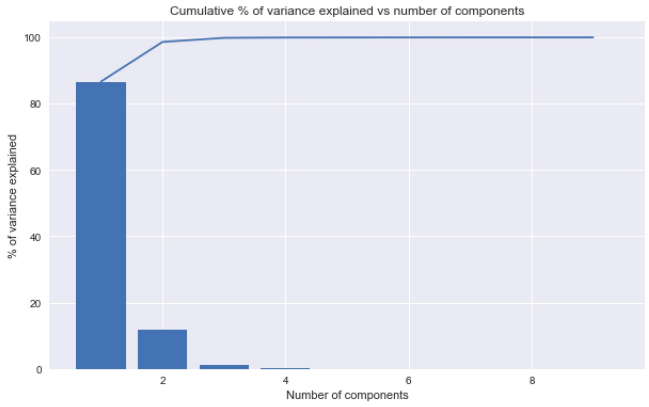


Figure 3.6: Cumulative percentage of the explained variance when applying a PCA on the raw outputs

The study of the eigenvalues shows that the six first components explain 99.99% of the total variance. When we look closer at the correlation of the original variables with the principal components, we see that all features have a similar correlation coefficient with the two first principal components.

3.2.2 Polynomial fitting

As we can see in Figure 4.5, a discontinuity always appears at the 12th station along the wing. Besides, the curves we observe are extremely regular. Consequently, it seems reasonable to fit a polynomial on the first part of the curve and another on the second. In order to choose properly the degree of each polynomial, we assess the quality of the fit by calculating a R-squared score for each curve.

Thus, we consider that it exists a polynomial function p of degree d for each part of the curve such as:

$$p(x) = a_0 x^d + ... + a_d$$

The coefficients $a_0, ..., a_d$ are obtained by minimizing the squared error by the least squares method.

To have an R-squared score greater than 99.9% for each curve and to avoid over-fitting by choosing too great degrees, the optimal couple of degrees is set to 2 for both polynomials. The dimension of the output space would be 6 instead of 29.

3.2.3 Polynomial fitting & Principal Components Analysis

By first applying polynomial fitting on the curves and then applying a PCA on the coefficients of the polynomials, we can decrease one more time the dimension of the output space from 6 to 4.

By keeping 4 principal components, the output space goes from 6 to the 4 dimensions and the precision is greater than 99.9% for at least 99% of the observations. Here follows the decline of the explained variance per component as well as the cumulative percentage of the explained variance (Figure 3.7):

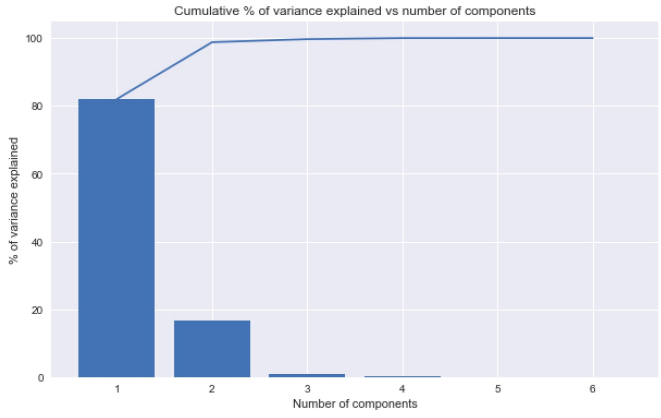


Figure 3.7: Cumulative percentage of the explained variance when applying a PCA on the coefficients of polynomials

In the following, we shall test the different dimensional reduction techniques above which will be compared to no dimensional reduction.

3.3 Regression based on Trees

In this section, different algorithms based on decision trees will be investigated. More precisely, the Classification and Regression Trees have been the source of numerous ensemble methods such as Bagging, Random Forest, the Gradient Boosting and AdaBoost and we explain how they work on the data we deal with. Recall we have at our disposal the 238t database of inputs which contains $\mathbf{X} = (X^1, ..., X^{25})$ where X^j are quantitative variables (i.e a A.C. parameter), and outputs are defined by $\mathbf{Y} = (Y^1, ..., Y^{29})$. For each individual, we observe a couple $Z_i = (X_i, Y_i)$ where $X_i = (X_i^1, ..., X_i^{25})$ and $Y_i = (Y_i^1, ..., Y_i^{29})$. We have thus a sample of observations of size $n = 28391$. The aim is to explain $\mathbf{Y}$ by a function of $\mathbf{X}$. For the sake of simplicity, we will consider the univariate regression $\mathbf{Y}^k$ (that is to say the value of the bending moment on the k^{th} station) by a function of $\mathbf{X}$.

3.3.1 Classification and Regression Trees (CART)

Classification and Regression Trees have been formalized by [5] and are decision trees. They consist of approximating a function F such as $F : \mathbf{X} \to \mathbf{Y}^k$. This algorithm considers all of 28391 observations and all of the 25 inputs. In no technical terms, the algorithms partitions the data into smaller and smaller sub-samples until all sub-samples are homogeneous in terms of output variables. Let us recall how the method works (see [5], [19]):

The construction of a tree is the successive partitioning of the output space thanks to the features in the form of a sequence of nodes. At the beginning, the full data set is linked to the initial node (also called the root) and is divided into two classes (two children nodes, left and right) accordingly to a division criteria. Thus, each child node represents a sub-sample of the data-set of the parent node, and recursively from each child node will arise two other children - if a node has no child, it is considered as a terminal node, also called a leaf. The observations belonging to each node must be the most homogeneous, and two children from a node must be the most heterogeneous. In fact at each node, a feature X^j is selected and the algorithm finds the threshold of X^j (thanks to an impurity measure, also called heterogeneity function or split function) which leads to the most homogeneous sample vs heterogeneous classes. The division criteria leads to know if a node must be a leaf or not, and finally associates each leaf to a value of Y^k.

A tree stop growing at a certain node for two reasons: the sub-sample contains too little data according to a fixed threshold set by the user, or the sample linked to the node is homogeneous and no other division is acceptable (that is to say that possible divisions lead to an empty child

node). The Figure 3.8 shows an example of construction of a tree.

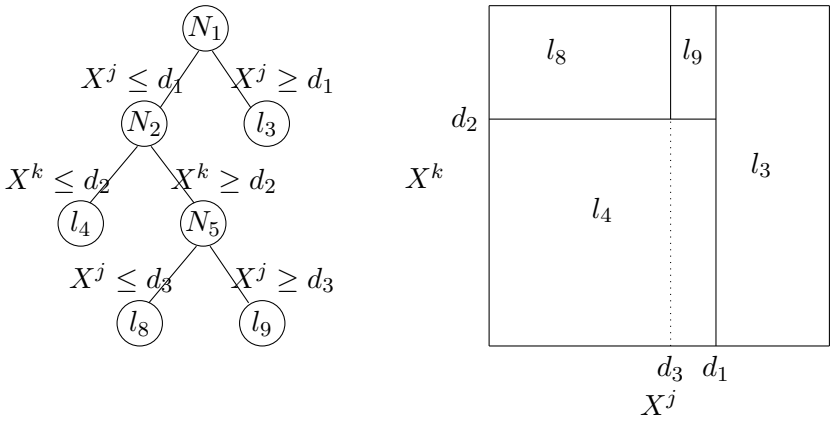


Figure 3.8: Example of construction of a tree [23] : Nodes are designed by N, and leaves by l

N_1 is the node containing all observations of $\mathbf{X}$, and other nodes or leaves contain a subsample of $\mathbf{X}$. Let be $\mathbf{I}_{l_j} := \{i,\ X_i \in l_j\}$. Then, the value of $\mathbf{Y}^k$ associated to l_j is defined by :

$$\mathbf{Y}^k_{l_j} = \frac{1}{\#\{\mathbf{I}_{l_j}\}} \sum_{i \in \mathbf{I}_{l_j}} Y^k_i \tag{3.1}$$

The value of Y^k associated to each leaf is then the average value of Y^ks associated to the sub-sample of the leaf.

At the end, this algorithm provides a huge tree with many leaves which can lead to over fitting. To avoid this effect, the tree must be pruned: we have to extract a sub-tree. Among a sequence of sub-trees, we keep the one which minimizes a criteria which depends most of the time of the generalization error and the complexity (the number of leaves): this method is called the cost complexity pruning. In our case, the generalization error (i.e the mean squared error) is calculated by cross-validation.

3.3.2 Bagging with regression trees

Bagging is an algorithm which aggregates trees and has been introduced by [2]. Let us consider the full sample $\mathbf{X}$ of size $n = 28391$. For $u = 1, ..., t$, we denote by $\mathbf{X}^{(n,u)}$ a sample of size n obtained by sampling with replacement $\mathbf{X}$. For each $\mathbf{X}^{(n,u)}$, we train a predictor p_u. $\{p_1, ..., p_t\}$ is therefore an

ensemble of predictors, predictors defined on different samples and are tree-based algorithms. Each individual X_i, $i = 1, ..., 28391$, belongs to t differents leaves (one for each tree) denoted by $l_{j_1}, ..., l_{j_t}$. So, by equation 5.3, we have t different values for the prediction of Y_i^k, i.e $(\mathbf{Y}^k_{l_{j_u}})_{u=1,...,t}$. The aggregated prediction value of Y_i^k is then defined by:

$$\hat{Y}_i^k = \frac{1}{t} \sum_{u=1}^{t} \mathbf{Y}^k_{l_{j_u}} \tag{3.2}$$

Sampling with replacement is most of the time associated to boosting sampling. The method explained above is named Bagging (stands for Boosting AGGregatING). Bagging improves predictions capabilities because it introduces differences between training samples which lead to variability of predictors. Breiman has shown that good candidates to boosting are classification and regression trees and neural networks.

3.3.3 Random Forest

Random Forests, introduced by [4], are based on bootstrap sampling and CART. As in Section 3.2, we first construct t sub samples with replacement of size n. When a tree is built, at each node of the tree, we draw randomly m inputs out of 25 (independently) and the optimal splitting criteria is defined through these m drawn variables. Trees grow to the maximal size and are not necessarily pruned.

Each tree is an estimator of the underlying function and built on a variation of the training set. As a consequence, each estimator leads to different results. Nevertheless, because of the numbers of estimators, the ensemble of trees (the forest), leads to a stable model. For a new observation, the prediction is then the average value of all the predictions of all predictors as in Bagging.

3.3.4 Gradient Boosting

The gradient boosting, intuited by cand developed by [11], is like every other boosting method: it combines weak learners. The goal stays the same, to explain $\mathbf{Y}^k$ by a function of $\mathbf{X}$ and instead of tuning parameters of this model, we iteratively add a model to the previous one to increase its capabilities. The name of "gradient" comes from the fact that the gradient of the squared error is the negative residual (see [11] and [16]). In our case, we use regression trees (CART). Here follows a simplified version of the Gradient Boosting Machine algorithm (for more details, see [11]):

Algorithm 7 Simplified Gradient Boosting Machine

procedure GBM
    Fit a decision tree F_1 on $\mathbf{X}$ (resp. $\mathbf{Y}^k$)
    Compute the error residuals $e_1 = \mathbf{Y}^k - F_1(\mathbf{X})$
    for $t = 2, ..., T$ **do**:
        Fit a decision tree F on $\mathbf{X}$ (resp. e_{t-1})
        $F_t(\mathbf{X}) = F_{t-1}(\mathbf{X}) + F(\mathbf{X})$
        Compute the error residuals $e_t = \mathbf{Y}^k - F_t(\mathbf{X})$
    The model is then the sum of all fitted trees

3.3.5 AdaBoost

One thing that Bagging does not take into account is that each observation is not equally susceptible to be drawn randomly from the training set. Most of the time, we cannot assure this condition. As explained by [7]; "in boosting, the probability of a particular example being in the training set of a particular machine depends on the performance of the prior machines on that example". In other words, if machine (a model) is able to predict and learn properly an observation, we do not need to learn more about it, but on observations which are difficult to learn on. Thus, these last ones will be more likely to be picked in a boosting sample. Adaboost was first introduced by [9, 10], and the following is a slightly modified version by [7] called AdaBoost.R2:

Initially, each observation is assigned by a weight $w_i = 1$, $i = 1, ..., n$. The algorithm is defined this way and continues till the average loss $\overline{L}$ goes under 0.5:

Algorithm 8 AdaBoost.R2

procedure ADB
  for $u = 1, ..., t$ **do**:
    The probability that the observation i is in the training set is directly obtain by $p_i = \frac{w_i}{\sum w_i}$. Draw with replacement a n-sized sample $\mathbf{X}^{(n,u)}$ (and its corresponding output $\mathbf{Y}_u^k$) from the training set $\mathbf{X}$ (and $\mathbf{Y}^k$).
    Build a model F_u on $\mathbf{X}^{(n,u)}$ (resp. $\mathbf{Y}_u^k$) by making a weak hypothesis $h_u : \mathbf{X}^{(n,u)} \rightarrow \mathbf{Y}_u^k$
    Pass $\mathbf{X}$ to the model to get each predictions $F_u(X_i), i = 1, ..., n$
    Calculate a loss for each observation. The loss may be of any form as long as $L \in [0, 1]$
    Calculate the average loss:$\overline{L} = \sum_{i=1}^{n} L_i p_i$
    Assessment of the confidence in the predictor by calculating $\beta = \frac{\overline{L}}{1-\overline{L}}$
    Update the weights $w_i \rightarrow w_i \beta^{1-L_i}$
  Outputs of each machine F_u are then weighted, and the predictor is the (weighted) median

Although this algorithm is noise and outliers sensitive, it does not need to be calibrated. This ensemble technique can be used with Random Forest and Decision Trees Regressors.

3.4 Prediction of loads for a new weight variant

In this section, we apply the techniques we described in Section 3 to our database and present the results we obtain.

3.4.1 Data preparation

Several options are possible to improve the capability of predictions of machine learning. For example, some of them are sensitive to the homogeneousness of the data they learn from, or the number of input variables, as well as outliers. Concerning the last case, we cannot consider outliers because every load cases have been validated thus we must consider all of them. In the first part, we will focus on clustering of our load cases of gusts to improve the ML performance. In the second part, we shall analyze the influence of different dimensional reduction techniques on the generalization capabilities of several algorithms based on regression trees.

To improve the capability of machine learning algorithms, clustering has been performed on the gust cases. From a weight variant to another,

loads experts are able to roughly estimate the form and intensity of the bending moments. To represent it a priori, we add the coefficients of the polynomials to the features to cluster our data and the K-means algorithm has been performed on these data (features and coefficients). The number of clusters was chosen with the experts and the elbow method using an Euclidean distance. A PCA has been performed and in the two first components, two clusters can be distinguished precisely (see Figure 3.9). In the following, these two clusters will be referred as Cluster 0 and Cluster 1:

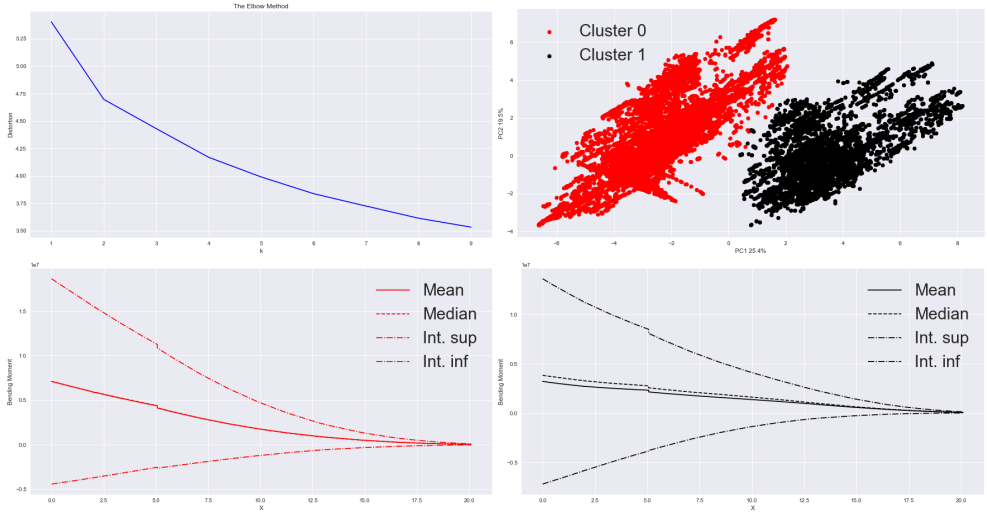


Figure 3.9: (a) Decrease of the Euclidean distorsion according to the number of clusters; (b): Scatter plot of individuals in the two PC; (c)&(d): Average, median, and Interval Inf. and Sup of bending moments of Clusters 0 and 1

As we can see in Figure 3.9, the average bending moment of the Cluster 0 is more linear than the one of Cluster 1. Besides, the cluster 1 is constituted by bending moment which are mainly positive and with higher value at the wing root. By looking closer at the A.C. parameters, we can see that most of variables have the same distribution with a slightly different mean value. Nevertheless, some of them are really different (see Table 3.3): this is the case for DQ_DEGL1 (Deflection left inboard Elevator), DSP_DEG1L (Deflection Spoiler 1 Left Wing), DP_DEGIL (Deflection all speed Inner Aileron), DP_DEGOL (Deflection low speed Outer Aileron) and even more for ENXF (X-Load Factor Body Axis), especially the distribution (see Figure 3.10 and 3.11):

Table 3.3: Comparison of variables means in the two clusters: DQ_DEGL1 (Deflection left inboard Elevator), DSP_DEG1L (Deflection Spoiler 1 Left Wing), DP_DEGIL (Deflection all speed Inner Aileron), DP_DEGOL (Deflection low speed Outer Aileron) ENXF (X-Load Factor Body Axis)

	DQ_DEGL1	DSP_DEG1L	DP_DEGIL	DP_DEGOL	ENXF
Cluster 1	0.0043	-0.00025	-0.0082	-0.0079	-0.0587
Cluster 0	0.0258	-0.4363	-0.0495	-0.0488	0.0173

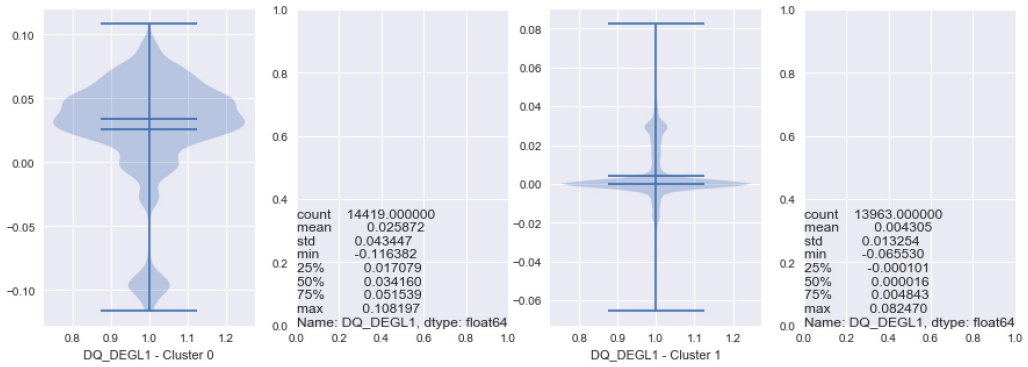


Figure 3.10: Comparison of DQ_DEGL1(Deflection left inboard Elevator) for the two clusters: the Cluster 0 is mainly constituted by load cases where the left inboard Elevator is active contrary to the Cluster 1

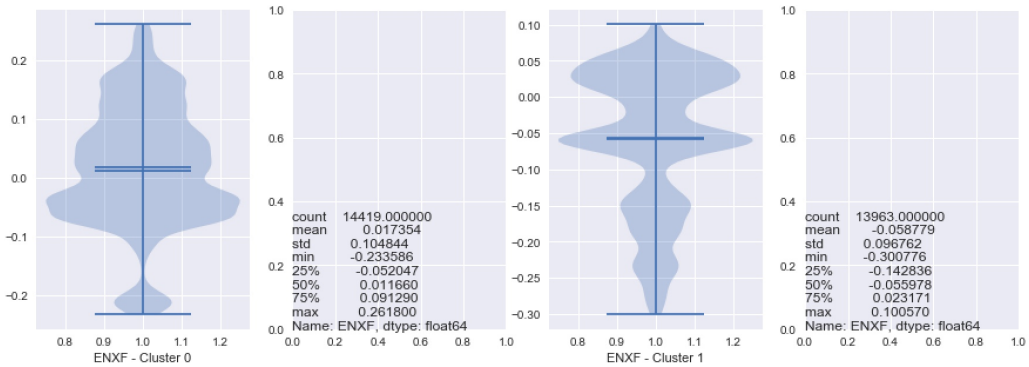


Figure 3.11: Comparison of ENXF (X-Load Factor Body Axis) for the two clusters: the Cluster 0 is mainly constituted by load cases where the X-load Factor Body Axis is positive contrary to the Cluster 1. Simply speaking, that means that the structure "warps" in a way for the Cluster 0, and the other way for the Cluster 1 (due to positive of negative gusts)

3.4.2 From 238t to 242t

Before presenting the results, it is important to explain more the R-squared score we have used in this project and why it is relevant in an engineering context. The R-squared, or also known as coefficient of determination, is a number that shows how well predictions are with respect to the explained variance. In other words, it is a measure of how well the model fits the data:

$$R^2 = 1 - \frac{\sum_i (y_i - \hat{y})^2}{\sum_i (y_i - \overline{y})^2}$$

In our case, we calculate a R^2 at each station of the wing. Indeed, by doing so, we maintain the engineering sense of accuracy of a curve. Because the variance for one curve can be extremely high - for example, we have at the root a value of 8 000 000 and at the wing tip it is closed to 0 - calculating a R^2 on all the values at the same times would lead to over-estimate the accuracy of our models because the total variance is higher and thus, the ratio between the squared error and the variance is really low.

The industrial goal was to have the higher R^2: in fact, this sprint project is part of a bigger project aiming to deliver models to accelerate pre-development of aircraft. Thus, the necessary condition is to have models precise enough and able to generalize simulations computed anteriorly to approximate, in our case, the computing chain of loads and stress. We agree that the R^2 can be misleading if the variance of the output is very high. As a consequence, by calculating a R-squared at each station (that is to say for each predictor) of the wing: we consider the variance only of the same kind of values in the outputs. The R-squared score given is then the average value of all R-squared calculated at each station.

To compare properly the results, from the 238t data set, we have drawn randomly a sample representing 80% of the observations, the last 20% represent the test set, and the 242t, 247t and 251t are our validation datasets, and we have repeated the process several time to see if a modification of the training set leads to unstable results in forecasting and generalizing.

To perform the comparison of algorithms presented above, we have used the scikit-learn library. Unfortunately, because we are trying to predict a field of vectors (we fit a model per station along the wing), just Random Forest is naturally implemented to do so and to take advantage of links which could exist between them. Simply speaking, when we fit a multioutput model with Random Forest, the impurity measure used at each

node has a "covariance" form such as defined in [20]. Then we used the MultiOutputRegressor for the other algorithms which fits an independent predictor per output vector (i.e per station): the MultiOutputRegressor is then an object containing as much predictors as outputs. As a recall, here are the algorithms we have tested the generalization capabilities: Adaboost based on decision trees regressors (ADB-DT); Adaboost based on Random Forest regressors (ADB-RF), Random Forest (RF), Bagging and Gradient Boosting (GBM). First, before checking the influence of dimensional reduction techniques we check which algorithms work the best on raw data:

Table 3.4: Mean/standard deviation of scores after random learning (80%) - testing (20%) - validation: (1) refers to Raw inputs + Raw outputs (no transformation on the data)

	Cluster 0			Cluster 1		
	Learning	Test	Validation 242t	Learning	Test	Validation 242t
ADB-DT (1)	0.9999/0	0.9756/0.04	0.956/0.001	0.999/0	0.983/0.003	0.967/0.001
ADB-RF (1)	0.9997/0	0.976/0.003	0.956/0.001	0.999/0	0.981/0.003	0.965/0.001
RF (1)	0.9917/0.003	0.96/0.004	0.92/0.003	0.994/0	0.966/0.005	0.925/0.003
Bagging (1)	0.9922/0.003	0.96/0.003	0.927/0.001	0.994/0	0.967/0.003	0.933/0.001
GBM (1)	0.8858/0	0.878/0.004	0.871/0.007	0.896/0	0.885/0.003	0.878/0

As we can see in Table 3.4, even if AdaBoost is not able to predict and take into account several outputs, the one based on decision tree regressors gets the better results. Random Forest combined with AdaBoost has 3% higher scores with a lower variability than RandomForest only. It is important to notice that GBM has the less degrowth from the test score to the validation score but the poorest score. Adaboost (based on decision trees or Random Forest) having the best results and the second less degrowth from the test score to the validation score (from 97.56% to 95.6%), we will focus on this algorithm to see the impact of dimensional reduction techniques.

To quantify the influence of dimensional reduction techniques on extrapolation capabilities, here follows the different configurations we need to compare:

- (1) Raw inputs + raw outputs: no data transformation.
- (2) Raw inputs + PCA outputs: we keep the original input space and we perform a PCA on the output space.

- (3) Raw inputs + polynomial fitting: we keep the original input space and replace the outputs by polynomial coefficients.

- (4) Raw inputs + polynomial fitting and PCA: we keep the original input space and replace the outputs by polynomial coefficients on which we perform a PCA.

- (5) PCA inputs + Raw outputs: we keep the original bending moment and we perform a PCA on the input space.

- (6) PCA inputs + PCA outputs: we perform a PCA on the design space, and another on the output space.

- (7) PCA inputs + polynomial fitting: we perform a PCA on the design space and replace the outputs by polynomial coefficients.

- (8) PCA inputs + polynomial fitting and PCA: we perform a PCA on the design space and replace the outputs by polynomial coefficients on which we perform a PCA.

Methods concerning the polynomial fitting are not shown due to lack of generalization and poor results. Other results are shown in Table 3.5.

Table 3.5: Mean/standard deviation of scores after several random learning (80%) - testing (20%) - validation 242t for the configurations: (1) Raw inputs + raw outputs; (2) Raw inputs + PCA outputs; (5) PCA inputs + Raw outputs; (6) PCA inputs + PCA outputs

	Cluster 0			Cluster 1		
	Learning	Test	Validation 242t	Learning	Test	Validation 242t
(1) ADB-RF	0.9997/0	0.976/0.003	0.956/0.001	0.999/0	0.981/0.003	0.965/0.001
(2) ADB-RF	0.9996/0	0.9751/0.002	0.956/0.0008	0.9996/0	0.9816/0.003	0.966/0.001
(5) ADB-RF	0.9996/0	0.9579/0.004	0.9120/0.001	0.9966/0	0.9680/0.004	0.9192/0.001
(6) ADB-RF	0.9995/0	0.9585/0.004	0.9136/0.003	0.9995/0	0.9684/0.004	0.9215/0.002
(1) ADB-DT	0.9999/0	0.9756/0.04	0.956/0.001	0.999/0	0.983/0.003	0.967/0.001
(2) ADB-DT	0.9998/0	0.9742/0.004	0.9565/0.001	0.9998/0	0.9823/0.005	0.9683/0.001
(5) ADB-DT	0.9999/0	0.9535/0.004	0.9145/0.001	0.9999/0	0.9670/0.005	0.9141/0.001
(6) ADB-DT	0.9998/0	0.954/0.004	0.9144/0.003	0.9998/0	0.9676/0.005	0.9247/0.003
(1) RF	0.9917/0.003	0.96/0.004	0.92/0.003	0.994/0	0.966/0.005	0.925/0.003
(2) RF	0.9923/0	0.9584/0.003	0.92/0.004	0.9937/0	0.9658/0.004	0.9255/0.001
(5) RF	0.9889/0	0.9407/0.004	0.8460/0.001	0.9899/0	0.9475/0.006	0.7675/0.001
(6) RF	0.9889/0	0.94/0.004	0.8681/0.004	0.9896/0	0.9665/0.004	0.7716/0.016

Remark. Parameters of algorithms can be consulted in the Appendix A.

PCA performed on the inputs does not improve results but reduces their variability for Random Forest. Nevertheless, we can see that a PCA applied only on the outputs improves slightly the average results when predicting the 242t for all algorithms. This is not surprising that applying a PCA does not highly improve the results since Random Forest and AdaBoost are natively able to deal with a large number of variables.

The results of ADB-RF are similar to ADB-DT. One major difference is the variability concerning the validation scores which is reduced against the other methods. From a cluster to another, results concerning the variability and the type of algorithms are the same; just the scores change.

AdaBoost with Random Forest or Decision Trees are similar, just the variability in scores is different. Indeed, due to the stable behavior of Random Forests, it is not surprising that AdaBoost performs better on Decision Trees than on Random Forests. Nevertheless, we can assume now that a PCA on the outputs improves the results and from now, we shall investigate how are the error distributed to understand better the lack of generalization capabilities of our model. In the following, just AdaBoost with Random Forest will be investigated concerning the extrapolation with a PCA applied on the outputs.

3.4.3 From 238t to 251t

The R-squared is not optimal to appreciate the quality of the fit: this score can hide poor results depending on the data people are dealing with. To assess the goodness of fit of our models, we defined for a curve of bending moment j the error rate as follows:

$$error(j) = \sqrt{\frac{\sum_{i=1}^{L}(\hat{y}(x_i) - y_j(x_i))^2}{\sum_{i=1}^{L} y_j^2(x_i)}}$$

For $j = 1, ..., n$, where n is the size of the sample we calculate the error rates, and where $L = 29$ is the number of stations along the wing. It allows us to have a physical idea of how far our predictions are. For this standpoint, we can easily compute the empirical cumulative distribution function (CDF): $\forall\ j = 1, ..., n$, let $\alpha \in [0, 1]$. The empirical CDF is defined as:

$$\alpha \to G(\alpha) = \frac{1}{n} \sum_{j=1}^{n} \mathbb{1}_{(error(j) \leq \alpha)}$$

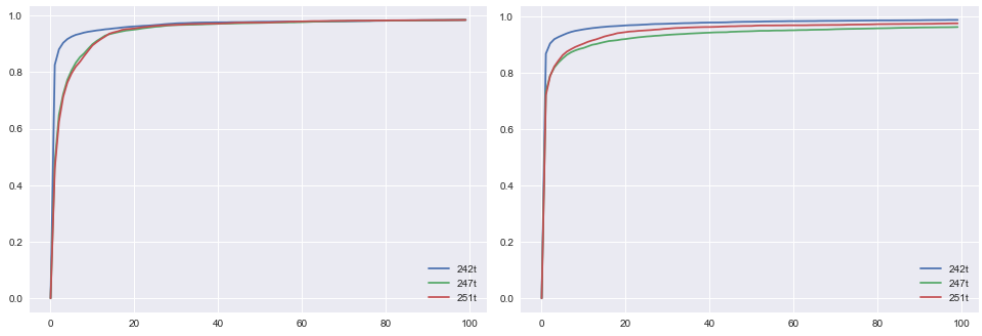


Figure 3.12: Empirical CDF of error rates ($\mathbb{P}(error \leq \alpha)$) concerning the extrapolation for Cluster 0 and Cluster 1: 242t (blue), 247t (green) and 251t (red)

The Table 4.4 gives more detailed information concerning the CDF of error rates in Figure 4.8:

Table 3.6: $\mathbb{P}(error \leq 5\%)$, $\mathbb{P}(error \leq 10\%)$ and $\mathbb{E}(error)$ for the different clusters and datasets 242t, 247t and 251t

	Cluster 0			Cluster 1		
	242t	247t	251t	242t	247t	251t
$\mathbb{P}(error \leq 5\%)$	88%	65%	63%	90%	79%	78%
$\mathbb{P}(error \leq 10\%)$	95%	89%	89%	95%	88%	90%
$\mathbb{E}(error)$	12%	17%	22%	23%	18%	27%

As soon as we try to generalize our results far from the training dataset, results drop. This is easily explain by the fact that some variables in the 247t and the 251t are far (in average) from the 238t: for example, the quantity of fuel in the first tank is 50% more important in the 242t, 117% in the 247t and 270% more important in the 251t. By looking at Deflection left inboard elevator, it is up to 50% different in the 247t and 251t than is the 238t and 242t. Unfortunately, theses features have a low importance according to Random Forest (see Appendix B). Besides, it is known that in some cases, slight changes of the features (especially the load factor along the Z-axis) can lead to very different behaviours.

3.5 Conclusion

Let us highlight now the contribution of this case study. As mentioned above, AdaBoost associated with Random Forest gives excellent results for observations which are not far from the training set. This is even more accurate when the outputs have similar forms for close design points and for load cases that are not impacted by the weight change roughly. As soon as we try to generalize the results for observations far from the learning data set or for load cases which leads to different behaviour, results drop. If we control the design space at the starting point, or add information concerning the form of the load to predict, or place us in an interpolation context, results would be even better.

A PCA on the outputs improves the results in average, and this can be explained because of the high co linearity of the outputs. Because of the presence of outliers and especially because all inputs matter, a PCA on the input space does not improve our results in average.

By trying to predict a vector (the shape of our training matrix is 28931x53) and not a point (it would have been 838 999x25), the speed of learning is exponentially decreased, and we keep the engineering information of the mathematical object.

Upcoming works concerning this project should investigate the following point: define a reliable method for extrapolation; test other dimensional reduction techniques as the shape invariant model approach such as defined by [21] which has been used in the petroleum industry; produce data in sub-spaces where there is a lack of information; investigate the fact that the optimal parameters obtained are maybe not optimal in term of generalization; consider other machine learning algorithms than those based on regression trees because they are known to be not optimal in a generalization problem, because they are considered as ?black-boxes? and because they do not give uncertainties; considering on-line learning: as soon as a new observation is available, the model should keep learning sequentially.

Airbus pursues the increasing knowledge capitalization and the development of new methods and tools for Research and Engineering through Big Data initiatives and the promising results of the sprint project, in which this case study has been achieved, are part of the root of upcoming bigger projects about Machine Learning in the load and stress process.

Appendix A: Models parameters

Models have been optimize through cross-validation (5-folds). The parameters which do not appear in the following table are set to default value of algorithm in scikit-learn. AdaBoost and Bagging use Decision trees as based estimators: due to time constraints, we have first optimized the parameters of Decision Trees alone on the data, and then optimize AdaBoost and Bagging parameters. Here follows the table containing the parameters of the models exposed in the previous sections.

Table 3.7: Models parameters through cross-validations (5 folds): models with an asterisk use the parameters of Decision Trees in the same column - (1) Raw inputs + raw outputs; (2) Raw inputs + PCA outputs; (5) PCA inputs + Raw outputs; (6) PCA inputs + PCA outputs.

		Cluster 0				Cluster 1			
	Parameters	(1)	(2)	(5)	(6)	(1)	(2)	(5)	(6)
RF	min_samples_leaf	5	2	5	2	3	10	5	3
	min_samples_split	10	11	3	12	14	6	8	10
	n_estimators	144	192	173	201	210	161	239	133
ADB-RF	learning_rate ADB	0.95	0.92	0.98	0.96	0.90	1.07	0.92	1.06
	n_estimators ADB	31	29	34	25	34	47	49	38
	n_estimators RF	19	23	11	13	22	24	24	12
	min_samples_leaf RF	17	15	4	3	6	3	2	9
	min_samples_split RF	13	17	4	17	7	7	4	17
DT	min_samples_leaf	4	3	3	2	17	19	15	3
	min_samples_split	9	7	12	7	18	15	12	6
ADB-DT(*)	learning_rate	1.09	0.96	1.07	0.93	1.03	0.93	0.93	1.02
	n_estimators	89	117	133	143	156	230	234	172
Bagging(*)	n_estimators	186	183	149	146	179	222	216	156
GBM	max_depth	8	10	10	13	15	14	14	14
	learning_rate	0.92	0.90	0.90	0.97	0.91	0.99	0.94	0.92
	n_estimators	42	46	52	67	161	149	69	65

One can notice that applying a PCA on the outputs leads to increase significantly the number of estimators in almost all cases when the min_samples_leaf and the min_samples_split are stable for RF and ADB-RF. Naturally, the number of estimators increases when the depth of the trees grows. A learning_rate above 1.0 seems to compensate a too large number of estimators and the more transformation we apply on our data, the more deeper are the trees underneath.

Appendix B: Features importance in Random Forests

The following table gives the features importance in Random Forest for the cases (1) Raw inputs + raw outputs and (2) Raw inputs +PCA outputs:

Table 3.8: Features Importance in Random Forests - (1) Raw inputs + raw outputs; (2) Raw inputs + PCA outputs

	Cluster 0		Cluster 1	
	(1)	(2)	(1)	(2)
Defl. Left inboard Elevator	0.0269	0.0771	0.0636	0.0719
Stabilizer Setting	0.0567	0.0171	0.0235	0.0155
Defl. Spoiler 1 Left Wing	0.0023	0.0056	0	0
Defl. Spoiler 2 Left Wing	0.0011	0.001	0.0001	0.0001
Defl. Spoiler 3 Left Wing	0.0013	0.0011	0.0001	0.0001
Defl. Spoiler 4 Left Wing	0.001	0.001	0.0001	0.0001
Defl. Spoiler 5 Left Wing	0.0012	0.001	0.0001	0.0001
Defl. Spoiler 6 Left Wing	0.0012	0.001	0.0001	0.0001
Defl. all speed inner Aileron	0.0283	0.0438	0.0090	0.0117
Defl. Low speed outer Aileron	0.0114	0.0158	0.001	0.0001
Lower part Rudder Deflection	0.0078	0.0095	0.0033	0.0036
Total A.C. Mass	0.139	0.1121	0.1314	0.1487
Mach Number	0.0052	0.0145	0.0074	0.0064
True Airspeed	0.0075	0.0243	0.0341	0.0240
Altitude	0.0121	0.0049	0.0099	0.0146
x-location of cg in % amc	0.0039	0.0086	0.0082	0.0067
Thrust(calculated)	0.0019	0.0017	0.0006	0.0004
X-Load Factor	0.0173	0.03	0.0281	0.0254
Y-Load Factor	0.0086	0.0161	0.0048	0.012
Z-Load Factor	0.6529	0.6045	0.6550	0.6462
Fuel Tank mass TANK1L	0.0016	0.0013	0.0028	0.0023
Fuel Tank mass TANK2L	0.0038	0.003	0.008	0.0046
Fuel Tank mass TANK3L	0.0015	0.0012	0.0042	0.0024
Fuel Tank mass TANK4L	0.0031	0.002	0.0033	0.0014
Left inner engine thrust	0.0022	0.0016	0.0006	0.0004

Features importance are stable from a method to another and the two most important features are identified: the mass and the Z-load factor. As said in the section 4.3, the importance of variables such as the Deflection left inboard elevator or the quantity of fuel in the first tank is small compared to those last two variables: thus, even if they change roughly for the other

weight variants, they have a low impact on the prediction of loads.

Bibliography

[1] Airbus, C.A.: A330 family (2017). Available from : http://www.aircraft.airbus.com/aircraftfamilies/passengeraircraft /a330family/

[2] Breiman, L.: Bagging predictors. Machine Learning **24**, 123–140 (1996)

[3] Breiman, L.: Arcing the edge. Technical Report (486) (1997). Statistics Department, University of California

[4] Breiman, L.: Random forests. Machine Learning **45**, 5–32 (2001)

[5] Breiman, L., Friedman, J., Olshen, R., Stone, C.J.: Classification and Regression Trees. Wadsworth, Belmont, CA (1984)

[6] Doherty, D.: Analytical modeling of aircraft wing loads using matlab and symbolic math toolbox (2009)

[7] Drucker, H.: Improving regressors using boosting techniques. Proceedings of the Fourteenth International Conference on Machine Learning pp. 107–115 (1997)

[8] Fournier, E., Klein, T., Grihon, S.: A case study: Influence of dimension reduction on regression trees-based algorithms-predicting aeronautics loads of a derivative aircraft. Journal de la Société Française de Statistique **159**(3), 56–78 (2018)

[9] Freund, Y., Shapire, R.: A decision-theoretic generalization of on-line learning and application to boosting. Proceedings of the second European Conference on Computational Learning Theory pp. 23–37 (1995)

[10] Freund, Y., Shapire, R.: Experiments with a new boosting algorithm, machine learning. Proceedings of the Thirteenth Conference pp. 148–156 (1996)

[11] Friedman, J.H.: Greedy function approximation: A gradient boosting machine (1999)

[12] Gandomi, A., Haider, M.: Beyond the hype: Big data concepts, methods, and analytics. Internation Journal of Information Management **35**, 137–144 (2015)

[13] Hjelmstad, K.D.: Fundamentals of Structural Mechanics. Springer US (2005)

[14] Hoblit, F.M.: Gust Loads on Aircraft: Concepts and Applications. AIAA Education Series, AIAA (1988)

[15] Hotelling, H.: Analysis of a complex of statistical variables into principal components. Journal of Educational Psychology **23**, 417–441 and 498–520 (1993)

[16] Li, C.: A gentle introduction to gradient boosting (2016). College of Computer and Information Science, Northeastern University. Available from: http://www.ccs.neu.edu/home/vip/teach/MLcourse/4_boosting/slides/gradient_boosting.pdf

[17] Manyika, J., al.: Big data: the next frontier for innovation, competition and productivity (2011). Mc Kinsley Global Institute

[18] Pearson, K.: On lines and planes of closest fit to systems of points in space. Philosophical Magazine **2**(11), 559–572 (1901)

[19] Quinlan, J.: Programs for machine learning. M. Kaufmann (1993)

[20] Segal, M., Xiao, Y.: Multivariate random forests. John Wiley and Sons, Inc. WIREs Data Mining Knowl Discov **1**, 80–87 (2011). DOI: 10.1002/widm.12

[21] Sergienko, E., Gamboa, F., Busby, F.: Shape invariant model approach for functional data analysis in uncertainty and sensitivity studies (2012)

[22] Torenbeek, E., Wittenberg, H.: Flight Physics: Essentials of Aeronautical Disciplines and Technology, with Historical Notes. Springer (2009)

[23] Wikistat: Arbres binaires de décision — wikistat (2016). Available from: http://wikistat.fr/pdf/st-m-app-cart.pdf

Chapter 4

Semiparametric estimation of plane similarities: Application to fast computation of aeronautic loads

Edouard Fournier*[1,2,3], Stéphane Grihon†[2], Thierry Klein‡[1,3]

[1]Institut de Mathématiques, UMR5219; Université de Toulouse; CNRS, UPS IMT, F-31062 Toulouse Cedex 9, France
[2]Airbus France 316, Route de Bayonne, Toulouse France
[3]ENAC - Ecole Nationale de l'Aviation Civile, Université de Toulouse, France

Résumé

L'étude descriptive menée au Chapite 3 nous a conduit à considérer un modèle semi-paramétrique que nous décrivons ici. Les charges sont représentées par des courbes ayant une forme similaire. Il est donc naturel de considérer qu'elles proviennent de la déformation d'une courbe de référence. Nous développons un nouveau modèle de déformation de courbes de charges qui sera intégré dans le processus de modélisation.

*edouard.fournier@airbus.com
†stephane.grihon@airbus.com
‡thierry.klein@math.univ-toulouse.fr or thierry01.klein@enac.fr

Plus précisément, nous nous donnons une courbe de référence et une famille paramétrique de transformations de la courbe de référence et nous supposons que chaque observation est l'image de la courbe mère par une des transformations de la famille paramétrique. L'originalité de notre approche consiste à considérer des opérateurs de déformation qui agissent simultanément sur l'espace des entrées et sur l'espace des sorties. Nous mettons en place une procédure de M-estimation des paramètres du modèle. Nous montrons la consistence et la normalité asymptotique des estimateurs des paramètres.

Nous complétons cette étude théorique par une mise en place numérique. Nous utilisons tout d'abord un modèle jouet pour comparer notre méthode à la méthode non paramétrique de Ramsay [22] appelée *Dynamic Time Warping*. Cette comparaison met en évidence que notre méthode est d'autant plus efficace que le rapport signal sur bruit est bon. Nous appliquons ensuite notre méthode à des données aéronautiques. Les résultats obtenus sur ces données réelles montrent que ce modèle de déformation paramétrique est bien adapté à la problématique de prédiction de charges avion.

Ce chapitre correspond au papier du même nom publié [6].

4.1 Introduction

It may be useful, when dealing with a large set of curves differing slightly from each other, to exhibit a parametric model of transformations. Indeed, usually a reduction taking into account some prior knowledge on the curves may lead to a better understanding of the variability within the population. One way to perform such reduction consists in considering that the set of curves has been obtained by deforming a template with parametric transformations. This point of view is usually called curve registration and has been widely studied in statistics. Two main statistical tasks are generally considered: the estimation of the template, and the estimation of transformation parameters.

For example, Härdel [13] considers the case of semiparametric models where the curves are nonparametric but are related in a parametric manner: the abscissa axis is shifted and the ordinate axis is rescaled. Regarding Golubev in [11], the idea is to estimate the period of an unknown signal with white noise conditions. In [9], Gasser and Kneip study the shape estimation of such templates, i.e structural features (extrema, inflection point) which can occur in a consistent manner among a sample of curves. The paper

[17] introduces the shape invariant model (SIM). In this paper, the authors estimate the deformation parameters and the unknown model function at the same time for a parametric family. In [15], Kneip considers a general non linear parametric regression model with unknown template function. Many authors have worked on the SIM estimating either the template, the parameters or both. We refer to [15, 18, 13, 14, 8] for more details and examples on the SIM. More recently, these estimation topics have seen a resurgence pushed by applications in image and signal processing. We refer for example to [12, 20, 1, 8, 4, 5, 10, 27] for models and techniques related to signal processing problems and the multidimensional extension of SIM defined on the plane. Concerning non parametric approaches, we refer to [22, 16]. In these papers, a nonlinear regression method is developed. It allows to aligned features of curves through the use of monotone functions acting on the abscissa axis (also called Dynamic Time Warping in the literature).

Hence, the set of transformations often consists in a parametric family of operator acting on curves [3]. In this paper, the parametric transformations involve rotation and scaling parameters. This model has been studied for image registration see for example [5] or [20]. According to our knowledge, all the models studied in the literature considered transformations that act independently on the argument and on the value of the template function. In this work, we considered a family of parametric transformations that act *jointly* on the argument and on the value of the template function. We work on functions defined on $[0, 1]$ and with plane similarities acting on the whole curve $\tilde{C} := (x, \tilde{f}(x))_{x\in[0,1]}$ representing the template.

Our aim is twofold:

- *Estimation of the deformation parameters*: we address the estimation of the deformation parameters and study the asymptotic properties of the estimator when the template is known on $[0, 1]$. Then, using these preliminary results we study the consistency and the asymptotic normality of our estimator in the more realistic case where the reference curve is defined on the same grid as the set of curves to be registered. In this paper, we consider that the set of curves to be registered are noisy but not the template. By supposing the legitimate existence of this template, it allows us to estimate the deformation parameters through a M-estimation procedure (see [26]).

- *Building a meta-model*: we will use the estimation of the deformation parameters in a real world application - the prediction of aeronautic loads. The idea will be to use and predict the transformation

parameters for different aircraft configurations. Using a proper representation of the problem, we get as good results as classical methods of dimension reduction techniques. Additionally, our methodology endows aerospace engineers with a better understanding of the variability within the set of load curves.

Our paper is organized as follow: in Section 4.2, we define our framework and the model of transformations. On the regression model, we use a M-estimation procedure to perform the estimation and study the asymptotic behavior of the estimators in two cases: when the reference curve $\tilde{C}$ is known, and when it is defined on the same grid as the set of curves to be registered. Section 4.3 is devoted to examples. We first compare our procedure to the Dynamic Time Warping method studied in [22] and using the **fda** package [23]. It appears, in particular, that for the class of functions we consider, our deformation strategy outperforms the warping method in the case where the noise does not exceed a certain threshold. Then, we apply our methodology to the prediction of aeronautic loads (see [7]): for different aircraft configurations we will predict the deformation parameters allowing us to compute the external constraints affecting the wing structure.

All the proofs are postponed to the last section.

4.2 Framework, model and analytic results

In this section, we describe the statistical model studied and give the asymptotic behavior of the M-estimators of the unknown parameters.

4.2.1 The observations

Notation. *Let be $x \in [0,1]$, and $f : [0,1] \to \mathbb{R}^+$. We denote by C the curve* $C := \begin{pmatrix} x \\ f(x) \end{pmatrix}_{x\in[0,1]}$.

In our framework, we have at hand $K+1$ curves. One is the reference curve denoted by $\tilde{C}$ that will also be called pattern. The K other curves $C_j,\ j = 1, ..., K$ are assumed to be the images of $\tilde{C}$ by the transformation model described bellow. The K curves are observed on the same random grid $\mathcal{D}_N := \{X_1, ..., X_N\}$ where $(X_i)_{i=1,...,N}$ are iid random variables with uniform distribution on $[0,1]$. Hence we have at our disposal

$$C_j^N = (X_i, f_j(X_i))_{i=1,...,N,\ j=1,...,K}.$$

We will consider the following two cases

i) $\tilde{C}$ is known everywhere: $\tilde{C} = (x, \tilde{f}(x))$, $\forall x \in [0,1]$,

ii) $\tilde{C}$ is only known on $\mathcal{D}_N$: $\tilde{C} = (X_i, \tilde{f}(X_i))_{i=1,...,N}$.

4.2.2 Transformation model

Before defining the transformation model linking the K observed curves to the pattern $\tilde{C}$, one must ensure that the functions f_j, $j = 1, ..., K$ and $\tilde{f}$ are "admissible". This is the aim of the next definition:

Definition. Let $\theta_0 \in]0, \frac{\pi}{2}[$, $\mathcal{F}_{\theta_0}$ is the set of applications defined by:

$$\begin{gathered}\mathcal{F}_{\theta_0} := \{f : [0,1] \to \mathbb{R}^+, f \text{ is differentiable on } [0,1], \\ f(0) > 0, f(1) = 0, \\ \forall x \in [0,1], f'(x) < 0, f'(1) = 0, f'(x) > -\cot\theta_0\}.\end{gathered}$$

The class of curves $\mathcal{C}_{\theta_0}$ is defined such that:

$$\mathcal{C}_{\theta_0} := \{(x, f(x)), x \in [0,1], f \in \mathcal{F}_{\theta_0}\}.$$

Let us now define the parametric family of transformation that we consider.

Notation. *For any function $T_\alpha : \mathbb{R}^2 \to \mathbb{R}^2$ depending on a parameter $\alpha \in \Theta$, we will denote by T_α^1 and T_α^2 its two coordinates.*

In the following, we define our transformation model.

Set $0 < \theta_0 < \theta_1 < \frac{\pi}{2}$, and $0 < \lambda_{min} < \lambda_{max}$. For any $\alpha := (\theta, \lambda) \in \Theta = [-\theta_0, \theta_1] \times [\lambda_{min}, \lambda_{max}]$, we introduce the transformations

$$T_\alpha := H_\lambda \circ \tilde{R}_\theta.$$

Our parametric set of transformations is the compositions of a rotation $\tilde{R}_\theta$ and a scaling on the second coordinate H_λ. More precisely

- $\tilde{R}_\theta$ is the rescaled rotation centered in $\begin{pmatrix}1\\0\end{pmatrix}$ and of angle θ. Note that if one performs just a rotation on a curve C of $\mathcal{F}_{\theta_0}$, the input space of the transformed curve is no longer $[0,1]$ but $[a_{\theta,C},1]$ so we must rescale the input space to $[0,1]$.That is

$$\begin{aligned}\tilde{R}_\theta : \mathcal{C}_{\theta_0} &\to \mathcal{C}_{\theta_0}\\ C &\to \tilde{R}_\theta(C),\end{aligned}$$

  which transforms each point $(x, f(x))$ of C to $\begin{pmatrix}\frac{(x-1)\cos\theta - f(x)\sin\theta + 1 - a_{\theta,C}}{1-a_{\theta,C}}\\ (x-1)\sin\theta + f(x)\cos\theta\end{pmatrix}$, where $a_{\theta,C} = -\cos\theta - f(0)\sin\theta + 1$.

  Notice that the elements of $\mathcal{F}_{\theta_0}$ are functions that are decreasing but not too strongly. Indeed, if there is a (smooth)-vertical drop on the graph of a function in $\mathcal{F}_{\theta_0}$ such that $\forall x \in [0,1],\ |f'(x)| < \cot\theta_0$, when operating a negative rotation on the curve, it prevents the transformed curve to be still a function. As a matter of fact, if the vertical drop does not respect the constraint, the transformed curve's graph does not look like a graph function but more to a "zig-zag" going locally forward and backward.

- The scaling transformation of parameter $\lambda > 0$ acting on the second coordinate of the curve:

$$\begin{aligned}H_\lambda : \mathcal{C}_{\theta_0} &\to \mathcal{C}_{\theta_0}\\ C &\to H_\lambda(C).\end{aligned}$$

  which transforms each point $(x, f(x))$ of C to $\begin{pmatrix}x\\ \lambda f(x)\end{pmatrix}$.

We now introduced the assumption that will be used:

(A1): The functions $(f_j)_{j=1,\dots,K}$, and $\tilde{f}$ belong to $\mathcal{F}_{\theta_0}$.

(A2): For any $\theta \in [-\theta_0, \theta_1]$, $\forall C \in \mathcal{C}_{\theta_0}$, all the second coordinate of $\tilde{R}_\theta(C)$ are non-negative.

(A3): $\Theta = [-\theta_0, \theta_1] \times [\lambda_{min}, \lambda_{max}]$.

(A4): $\tilde{C}$ is known on $[0,1]$.

(A5): $\tilde{C}$ is known on the grid $\mathcal{D}_N$.

Thus, the transformation model considered is as follows:

$$\begin{aligned} T_\alpha : \mathcal{C}_{\theta_0} &\to \mathcal{C}_{\theta_0} \\ C &\to T_\alpha(C) \end{aligned}$$

which transforms each point $(x, f(x))$ of C to $\begin{pmatrix} \frac{(x-1)\cos\theta - f(x)\sin\theta}{\cos\theta + f(0)\sin\theta} + 1 \\ \lambda((x-1)\sin\theta + f(x)\cos\theta) \end{pmatrix}$: the transformation model acts jointly on the argument and on the value of the template function.

4.2.3 Regression model

Reminding that we wish to adjust the reference curve $\tilde{C}$ on the other curves C_j $(j = 1, ..., K)$ by transformations belonging to $\{T_\alpha, \alpha = (\theta, \lambda) \in \Theta\}$. Notice that these transformations act on both axis. For any α, we want to compare the value of the transformed curve $(T_\alpha \tilde{C})(X_i)$ with $f_j(X_i)$. Since the abscissa points are affected by the transformation, we denote by $X_i(\alpha)$ the point such that $T^1_\alpha(X_i(\alpha)) = X_i$. For that reason, we introduce the following definition:

Definition. We denote by $x(\alpha, \tilde{f})$ the solution of the equation:

$$T^1_\alpha(u, \tilde{f}(u)) = x. \tag{4.1}$$

To ease the notation, we finally set $x(\alpha) := x(\alpha, \tilde{f})$, so $X_i(\alpha) = X_i(\alpha, \tilde{f})$. We consider the parametric regression model:

$$f_j(X_i) = T^2_{\alpha^*_j}(X_i(\alpha^*_j), \tilde{f}(X_i(\alpha^*_j))) + \epsilon_{j,i}, (j = 1, ..., K). \tag{4.2}$$

Where:

- (X_i) are iid, $[0,1]$-uniformly distributed. It is the design on which we observe the curves C_j;
- $\alpha^*_j = (\theta^*_j, \lambda^*_j)$ is the couple of true parameters for each curve $(j = 1, ..., K)$;
- $\epsilon_{j,i}$, $\forall i = 1, ..., N$, $\forall j = 1, ..., K$ are iid $\mathcal{N}(0, \sigma^2)$ random variables. These variables are assumed to be independent of X_i.

4.2.4 Estimation

We consider the case where the noise applies on the set of curves to be registered only, and not on the reference curve. We legitimately suppose the existence of a reference curve. Under this last assumption, we will estimate the deformation parameters through a M-estimation procedure. The first subsection refers to the estimation of the deformation parameters and its analytical results when the reference curve is known on $[0,1]$. Although this case is not realistic, these first results are necessary prerequisites to show the consistency and the asymptotic normality of the estimator in the more realistic case where the reference curve is defined on the same grid $\mathcal{D}_N$ as the set of curves to be registered - this is the topic of the last subsection.

4.2.4.1 Estimation when $\tilde{C}$ is known on $[0,1]$

For the sake of simplicity, let us fix j. Relying on a classical M-estimation procedure, we consider a semiparametric method to estimate the parameters and define consequently the following empirical contrast function to fit the reference curve $\tilde{C}$ to C_j $(j = 1, ..., K)$:

$$\begin{aligned} M_N^j(\alpha) &= \frac{1}{N}\sum_{i=1}^{N}(f_j(X_i) - T_\alpha^2(X_i(\alpha), \tilde{f}(X_i(\alpha))))^2 \\ &= \frac{1}{N}\sum_{i=1}^{N} m_\alpha^j(X_i). \end{aligned} \tag{4.3}$$

The random function M_N^j is non negative. Furthermore, intuitively, its minimum value should be reached close to the true parameter α_j^*. Indeed, the following theorem gives the consistency of the M-estimator, defined by :

$$\hat{\alpha}_N^j = \underset{\alpha\in\Theta}{\operatorname{argmin}}\ M_N^j(\alpha). \tag{4.4}$$

Recall that our empirical contrast function enters in the general theory of M-estimator. The Central Limit Theorem will be shown by using M-estimator arguments.

Theorem 3. *Assume that A1, A2, A3 and A4 are satisfied. Then*

$$i)\ \hat{\alpha}_N^j \xrightarrow[N\to+\infty]{\mathbb{P}} \alpha_j^*, \tag{4.5}$$

$$ii)\ \sqrt{N}(\hat{\alpha}_N^j - \alpha_j^*) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha_j^*}). \tag{4.6}$$

In particular, the covariance matrix has the following form

$$\Gamma_{\alpha_j^*} = V_{\alpha_j^*}^{-1} 2\sigma^2, \tag{4.7}$$

with $V_{\alpha_j^*} = 2\mathbb{E}[\dot{T}^2_{\alpha_j^*}\dot{T}^{2T}_{\alpha_j^*}]$*, and* $\dot{T}^2_{\alpha_j^*}$ *is the vector of partial derivatives of* $T^2_{\alpha_j}$ *w.r.t elements of* α *taken at* α_j^**.*

4.2.4.2 Estimation when $\tilde{C}$ is observed on $\mathcal{D}$

In this section, we consider the case where the reference curve $\tilde{C}$ is observed on the same grid $\mathcal{D} := (X_i)_{i=1,..,N}$ as the other curve C_j, i.e $\tilde{C} = \begin{pmatrix} x \\ \tilde{f}(x) \end{pmatrix}_{x\in D}$. By applying the transformation T_α to $\tilde{C}$, the transformed pattern $T_\alpha\tilde{C}$ is no longer observable on $\mathcal{D}$. As a consequence, one must make use of an approximation process over $\tilde{f}$. Let $\tilde{f}_N$ be the linear interpolate of $\tilde{f}$, defined by:

$$\tilde{f}_N(x) = \sum_{i=1}^{N} \Delta_i(x) \mathbb{1}_{x\in[X_{(i)},X_{(i+1)})}, \tag{4.8}$$

where

$$\Delta_i(x) = \frac{\tilde{f}(X_{(i+1)}) - \tilde{f}(X_{(i)})}{X_{(i+1)} - X_{(i)}}\ x\ +\ \tilde{f}(X_{(i)}) - \frac{\tilde{f}(X_{(i+1)}) - \tilde{f}(X_{(i)})}{X_{(i+1)} - X_{(i)}} X_{(i)}. \tag{4.9}$$

It is easy to see that $\tilde{f}_N$ belongs also to $\mathcal{F}_{\theta_0}$. Replacing $\tilde{C}$ by $\hat{C}$ in (4.3) we obtain

$$\hat{M}_N^j(\alpha) = \tfrac{1}{N}\textstyle\sum_{i=1}^{N}(f_j(X_i) - T^2_\alpha(X_i(\alpha, N), \tilde{f}_N(X_i(\alpha, N))))^2,$$

where $X_i(\alpha, N)$ is the solution to the equation $T^1_\alpha(u, \tilde{f}_N(u)) = X_i$.

Using the linear interpolate defined by (4.8), we show the consistency and asymptotic normality of our M-estimator defined as follow:

$$\hat{\hat{\alpha}}_N^j = \underset{\alpha\in\Theta}{\operatorname{argmin}}\ \hat{M}_N^j(\alpha). \tag{4.10}$$

Theorem 4. *Assume that A1, A2, A3 and A5 are satisfied. Let* $\tilde{f}_N$ *be defined by (4.8) and assume that* $\exists C > 0$ *s.t* $\forall x \in [0,1]$, $\tilde{f}'_N(x) \leq C$, *and* $\tilde{f}'(x) \leq C$*, then*

$$i)\ \hat{\hat{\alpha}}_N^j \xrightarrow[N\to+\infty]{\mathbb{P}} \alpha_j^*, \tag{4.11}$$

$$ii)\ \sqrt{N}(\hat{\hat{\alpha}}_N^j - \alpha_j^*) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha_j^*}) \tag{4.12}$$

with $\Gamma_{\alpha_j^*}$ *such defined in (4.7).*

Remark. Notice that the asymptotic variance of the estimator is the same as in Theorem 1.

4.3 Simulations and applications

In this section we illustrate the method on numerical applications. The first subsection is dedicated to some simulated example while the second to a real problem. The optimisation problems (5.6) and (4.10) will be numerically solved by using the BFGS algorithm [21]. In each case, in order to apply the results presented in the previous section, one shall define a reference curve. In the simulated example, the reference curve is known. In the real world application, we simply choose the average curve as the reference curve (note that choosing the curve with the lowest values would assure that (A2) is automatically satisfied).

4.3.1 Simulated example

We consider the following model:

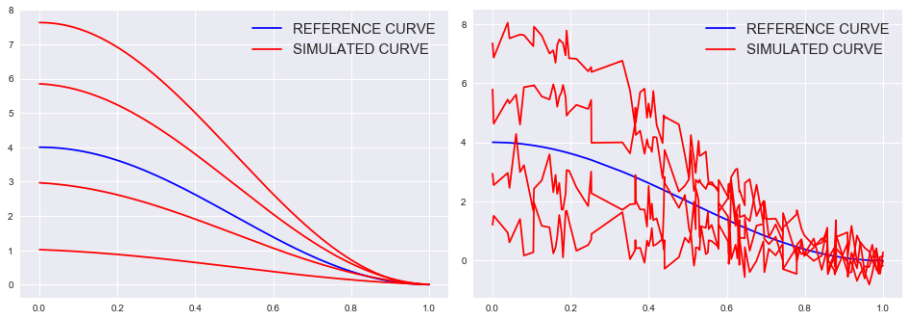
$$f_{\lambda,\theta}(x) = \lambda[(x-1)\sin\theta + g(x)\cos\theta],$$

with $g(x) = 2(\cos(\pi x) + 1)$. We observe $f_j(X_i) = f_{\lambda_j,\theta_j}(X_i) + \epsilon_{ij}$ for $i = 1, ..., N$, for $J = 25$ values of (λ, θ) with some iid errors ϵ_{ij}.

- The observations points $X_i,\ i = 1, ..., N$ are iid random variables with uniform distribution on $[0, 1]$.
- The parameters are chosen randomly with the following arbitrary distribution $(\lambda, \theta) \hookrightarrow \mathcal{U}([0, 15]) \times \mathcal{U}([-\frac{\pi}{3}, \frac{\pi}{30}])$.
- The errors are assumed to be $\mathcal{N}(0, \sigma^2)$.

The numerical experiment will be conducted for different values of $N = 50, 100, 1000$ and $\sigma^2 = 0, 0.01, 0.05, 0.1, 0.5, 1$. To avoid numerical issues, each curve is rescaled to $[0, 1]$. Some simulated data are shown in Figure 4.1. In this example, the reference curve is known. This is the function g displayed in blue. As shown in Figure 4.1 (b), one shall emphasize that by increasing the value of the noise variance we loss the properties that the functions $f_j,\ j = 1, ..., J$ belong to $\mathcal{F}_{\theta_0}$, which is not in our best interests.

In the following, we compare the method developed in the previous section to the famous non-parametric method of curve registration defined by J.O. Ramsay and X. Li in [22]. This method is also called Dynamic Time Warping (DTW): a non-parametric technique is used to estimate the smooth monotone transformations h_i applied on the features (i.e the x-abscissa) by



(a) (b)

Figure 4.1: Simulated data with : (a) $N = 100$ and $\sigma^2 = 0$, (b) $N = 100$ and $\sigma^2 = 0.5$

preserving their order. Hence, the features X_i are warped through h_i and the characteristic (peaks, valleys) of the reference curve and the target are aligned. In order to compare the two methodologies, we will use the following mean squared error (MSE) between the registered curve and f_{λ_j,θ_j}

$$MSE = \frac{1}{25}\sum_{j=1}^{25}\sum_{i=1}^{N}(f_{\lambda_j,\theta_j}(X_i) - \hat{f}_{\hat{\lambda}_j,\hat{\theta}_j}(X_i))^2$$

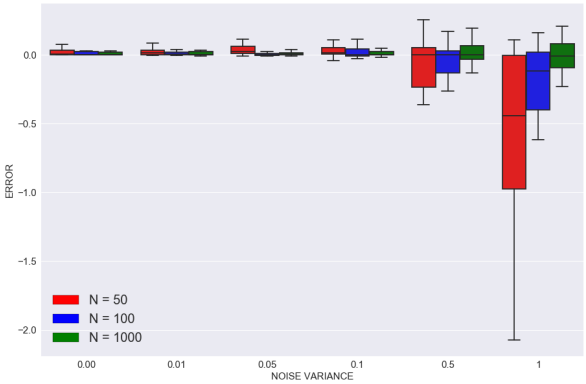
Results are displayed in Table 4.1. It comes at no surprise that increasing the noise's variance increases the MSE. Nevertheless, due to its transformation nature, Ramsay's method might take the augmentation of noise variance to its advantage contrary to our morphing strategy. However, when the noise is small, our methodology outperforms the other. This is due to the fact that the peaks and valleys of the curves are most of the time already aligned in this kind of function class inducing few deformations on the abscissa.

Table 4.1: Mean Squared Error (MSE) comparison between our morphing strategy and the DTW for different N and σ^2.

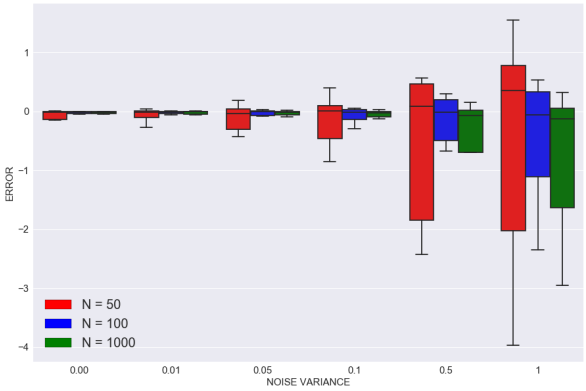
	Morphing			DTW		
	N = 50	N = 100	N = 1000	N = 50	N = 100	N = 1000
$\sigma^2 = 0.00$	**7.e-05**	**6.e-05**	**4.e-05**	0.0011	0.0014	0.0012
$\sigma^2 = 0.01$	**0.0006**	**0.0002**	**0.0001**	0.0061	0.0041	0.007
$\sigma^2 = 0.05$	**0.0032**	**0.0025**	**0.0024**	0.0076	0.0051	0.0092
$\sigma^2 = 0.10$	**0.0104**	0.0098	**0.0091**	0.012	**0.0086**	0.015
$\sigma^2 = 0.50$	0.2489	0.2449	0.2473	**0.1451**	**0.1413**	**0.1732**
$\sigma^2 = 1.00$	0.9942	0.9782	0.9892	**0.65**	**0.5974**	**0.79**

In Figure 4.2 are displayed the errors when estimating the deformation

parameters for the different values of the noise variance σ^2 and the number of observations N. The rotation parameter θ is the one which is the most sensitive to the number of observations N, and in average our morphing strategy seems to be slightly optimistic in the estimation of λ for a small noise variance.



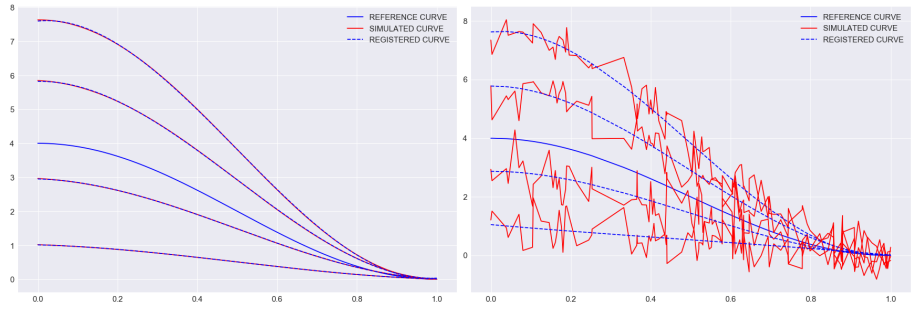
(a)



(b)

Figure 4.2: Boxplot of the errors for the estimation of (a) λ and (b) θ depending on the noise variance σ^2 and the number of observations N using the morphing method.

Examples of the registration process are displayed in Figure 4.3. Our methodology works better when the data we are dealing with are closed to the proposed model, else it tends to make more error undeniably. Nevertheless, compared to a nonparametric model, we can use the estimations of the deformation parameters to do meta-modeling as we will show in the next subsection.



(a) (b)

Figure 4.3: Results of the registration process with : (a) $N = 100$ and $\sigma^2 = 0$, (b) $N = 100$ and $\sigma^2 = 0.5$

4.3.2 Aeronautic loads

An airframe structure is a complex system and its design is a complex task that today involves many simulation activities generating massive amounts of data. This is, for example, the process of loads and stress computations of an aircraft. That is the computations of the forces and the mechanical strains suffered by the structure. The overall process exposed in Figure 4.4 is run to identify load cases (i.e aircraft mission and configurations: maneuvers, speed, loading, stiffness...), that are critical in terms of stress endured by the structure and, of course, the parameters which make them critical. The final aim is to size and design the structure (and potentially to reduce loads in order to reduce the weight of the structure). Typically for an overall aircraft structure, millions of load cases can be generated and for each of these load cases millions of structural responses (i.e how structural elements react under such conditions) have to be computed. As a consequence, computational times can be significant.

In an effort to continuously improve methods, tools and ways-of-working, Airbus has invested a lot in digital transformation and the development of infrastructures allowing to treat data (newly or already produced). The main industrial challenge for Airbus is to reduce lead time in the computation and preliminary sizing of an airframe as well as extracting value from already calculated loads. In this paper, we focus on the external loads of a wing: the shear forces (transverse forces near to vertical arising from aerodynamic pressure and inertia) and the associated bending moments (resulting from the shear forces, they represent the flexion of the wing) are calculated for each load case. Examples of bending moments are displayed in Figure 4.5.

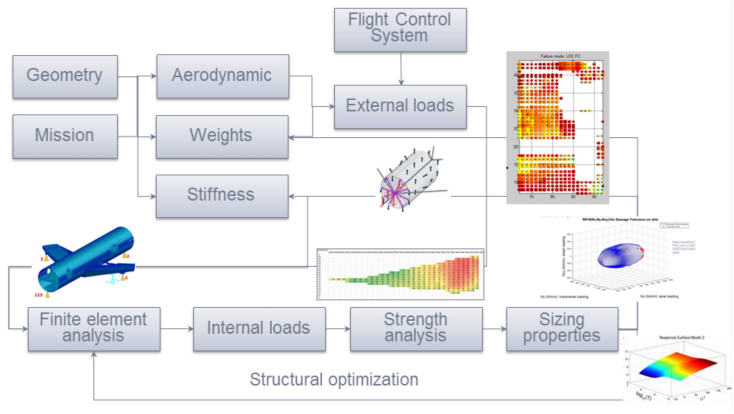
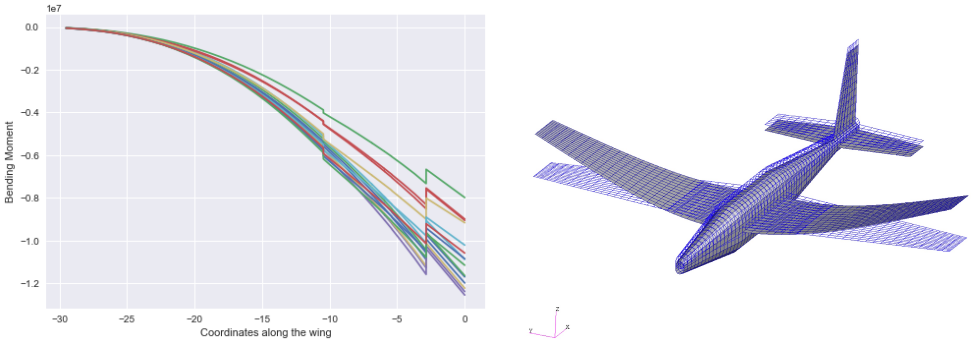


Figure 4.4: Flowchart for loads and stress analysis process



(a) (b)

Figure 4.5: (a) Examples of bending moments along the wing for different load cases - (b) Finite element model of a generic aircraft representing the wing deformation [24]

These external loads appeared to be extremely regular and one can legitimately suppose that there exists a link between all those curves. Indeed, it is natural to assume that there exists a reference bending moment (a reference curve) which can be morphed through a deformation model to give all the other curves.

In [7], the authors present an aeronautic model that computes the loads (forces and moments) on the wing of some aircraft model denoted by $ACM1$. They present several statistical methods in order to study these data. In this section, we will compare the method used in [7] with the model presented in Section 4.2 for a new aircraft model called $ACM2$. The data at our disposal represents bending moments of a wing (representing its flexion) of an aircraft calculated for 1152 different configurations (load

cases). Each configuration is defined by 28 features (speed of the aircraft, mass, altitude, quantity of fuel, etc.), leading to a bending moment calculated on 45 stations along the wing. In a more formal way, we observe the couple $(X_j, Y_j)_{j=1,...,1152}$, where $X_j = (X_j^1, ..., X_j^{28})$ are the features and $Y_j = (Y_j^1, ..., Y_j^{45})$ is the bending moment. The idea is to predict the bending moment for different configurations. The data are represented in Figure 4.6.

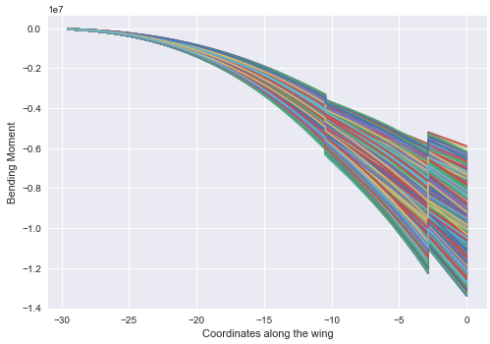


Figure 4.6: Representation of all the bending moments of a wing of our data base: the wing root is located at the zero origin, where the strains are maximum when the wing bends.

Due to the discontinuities at the 3^{rd} and 20^{th} stations, we apply our methodology to each section independently: we suppose, reasonably, that the average curve (of each section), can be used as the reference curve. Then, each section can be represented by its minimum and maximum values, and by its rotation and scaling coefficients λ_j and θ_j. Figure 4.7 assess the quality of the matching process (the reference curve used is the average bending moment). As a comparison, we display in Table 4.2 the MSE between the DTW and our deformation method (note that the MSE values are high due to the fact that the curves have high values).

Table 4.2: Mean Squared Error comparison between our morphing strategy and the DTW for the real world application.

	Morphing	DTW
MSE	**2.2e+09**	3.1e+09

Thus the dimensional space of the outputs is reduce to 12 instead of 45. We compare our method to three other methods of [7] applied on the outputs: no transformation (we call it raw - we build 45 models, one per station); a PCA (the three first principal components represent 99,9% of the explained variance - 3 models instead of 45); a polynomial fitting per section

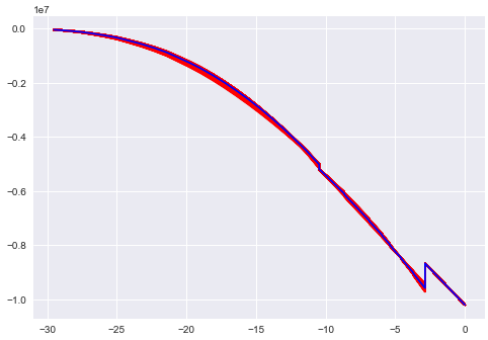


Figure 4.7: Results of the matching process

(of degree 4 for the first section, of degree 2 for the second and of degree 1 for the third section) which leads to 10 models instead of 45. The Table 4.3 sums up the number of outputs to predict depending on the method used.

Table 4.3: Number of outputs to be predicted depending on the method used on the raw outputs: Raw, Deformation Model, PCA, polynomial fitting

	Number of outputs	Names of outputs
Raw	45	Bending moment value at station 0 to 44
Deformation Model	12	$\theta_1, \theta_2, \theta_3, \lambda_1, \lambda_2, \lambda_3, min_1, min_2, min_3, max_1, max_2, max_3$
PCA	3	Principal components 1 to 3
Polynomial fitting	10	Coefficients of polynomials

The significant advantage of the reduction dimension techniques used is that the response of the model would have a physical form contrary to the simple linear models performed on the raw data. To build our models, we use the Orthogonal Greedy Algorithm (OGA) also known as the Matching Pursuit Algorithm. Detailed explanations can be found in [2], [25] and [19]. Roughly speaking, we consider the problem of approximating a function by a sparse linear combination of inputs.

To assess the goodness of fit of our models, we defined for a curve of bending moment j the error rate as follows:

$$error(j) = \sqrt{\frac{\sum_{i=1}^{45}(\hat{y}(x_i)-y_j(x_i))^2}{\sum_{i=1}^{45} y_j^2(x_i)}},\ j = 1, ..., n_{test},$$

where n_{test} is the size of the sample of test. We compute the error rates on (the sample of test is of 25% the size of the total database). It

gives an idea of how accurate our predictions are. For this standpoint, we can easily compute the empirical cumulative distribution function (CDF): $\forall\ j = 1, ..., n_{test}$, let $\alpha \in [0, 1]$. The empirical CDF is defined as:

$$\alpha \to G(\alpha) = \frac{1}{n} \sum_{j=1}^{n_{test}} \mathbb{1}_{(error(j) \leq \alpha)}$$

In Table 4.4, we give the values of $G(\alpha)$ for $\alpha = 1\%, 2\%, 5\%, 10\%$ and the mean error. In Figure 4.8 we give the plots of the function $G(\alpha)$ for the different methods.

Table 4.4: Average estimated $\mathbb{P}(error \leq 1\%)$, $\mathbb{P}(error \leq 2\%)$, $\mathbb{P}(error \leq 5\%)$ $\mathbb{P}(error \leq 10\%)$, $\mathbb{E}(error)$ calculated on several random test data set (25% of the size of the total dataset)

	Deformation Model	Polynomial Fitting	PCA	Raw
$\mathbb{P}(error \leq 1\%)$	17%	14%	16%	15%
$\mathbb{P}(error \leq 2\%)$	45%	45%	43%	51%
$\mathbb{P}(error \leq 5\%)$	88%	88%	86%	88%
$\mathbb{P}(error \leq 10\%)$	98%	97%	95%	98%
$\mathbb{E}(error)$	2.9%	2.9%	3%	2.8%

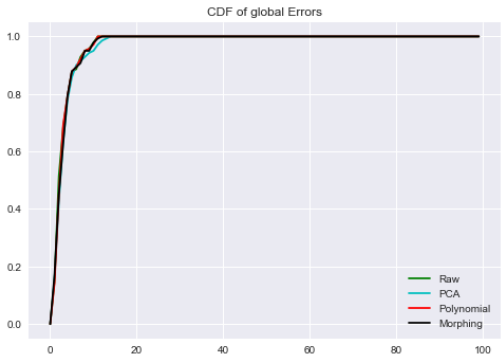


Figure 4.8: Empirical CDF of error rates ($\mathbb{P}(error \leq \alpha)$)

Concerning the approximation and prediction of loads, our model is equivalent in average to other tested methods, there are just slightly more observations with an error below 1%. Nevertheless, in our case, the linear models built through the deformation model are sparser than the other. Indeed, in average, 11 variables are chosen as optimal parameter of the greedy algorithm by cross-validation for the deformation model, 13 for the polynomial fitting, 15 for the PCA and 14 for the raw outputs one.

Even though the prediction of loads with the deformation model is so likely equivalent to none transformation, it obtains better results than the polynomial fitting and the PCA. Besides, using this deformation model gives a physical response contrary to a simple linear model per station whose response could be irregular.

We have shown that, when dealing with a large set of curves, if we provide and use a good representation, we can have: as good results as state of the art classical methods of dimension reduction techniques or none; and a better understanding of the variability within the population at the same time. It is the last point which is important particularly: indeed, it exists many ways to have good prediction results (non-parametric regression, black-box models). Using reduction dimension techniques without an underlying engineering judgement is not of significant interest: from our methodology, besides having a good prediction accuracy, we can extract knowledge for aerospace engineers to better understand the variation of behaviors of the wing depending on the reference behavior (in other words, it gives to engineers a physical interpretation and idea of how the wing will react to new constraints). Hence, in this sense, our proposed methodology outperforms the others.

4.4 Proofs and technical result

4.4.1 Technical result

This section is dedicated to the technical result used in the proof of Theorem 2.

Lemma 5. *Let $X_1, ..., X_N$ be N independent and identically distributed random variables with uniform distribution on $[0,1]$ and let $X_{(1)} \leq ... \leq X_{(N)}$ be the reordered sample . Let $a_N = O(\sqrt{N})$, then:*

$$\mathbb{P}(a_N \sup_j |X_{(j+1)} - X_{(j)}| \geq \epsilon) \xrightarrow[N\to+\infty]{} 0.$$

Proof of Lemma 3. Let $Z_1, ..., Z_{N+1}$ be N independent and identically distributed random variables with exponential distribution with parameter 1. It is a well known fact that $(\frac{Z_1}{\sum_{k=1}^{N+1} Z_k}, \frac{Z_1+Z_2}{\sum_{k=1}^{N+1} Z_k}, ..., \frac{Z_1+...+Z_N}{\sum_{k=1}^{N+1} Z_k}) \overset{(\mathcal{L})}{=} (X_{(1)}, ..., X_{(N)})$ and we have

$$X_{(j+1)} - X_{(j)} \overset{(\mathcal{L})}{=} \frac{Z_{j+1}}{\sum_{k=1}^{N+1} Z_k}. \tag{4.13}$$

Now, for $\epsilon > 0$

$$\begin{aligned}\mathbb{P}(\sup_j |X_{(j+1)} - X_{(j)}| \geq \epsilon) &\leq \sum_j \mathbb{P}(X_{(j+1)} - X_{(j)} \geq \epsilon) \\ &\leq N \max_j \mathbb{P}(X_{(j+1)} - X_{(j)} \geq \epsilon).\end{aligned}$$

By using (4.13), we have

$$\mathbb{P}(X_{(j+1)} - X_{(j)} \geq \epsilon) = (1-\epsilon)^{N-1}.$$

Then,

$$\mathbb{P}(\sup_j |X_{(j+1)} - X_{(j)}| \geq \epsilon) \leq N(1-\epsilon)^{N-1}.$$

The result follows replacing ϵ by $\frac{\epsilon}{a_N}$ and letting $N \to +\infty$. □

4.4.2 Proofs of Theorems 1 and 2

Proof of Theorem 1. To ease the notation, we do not display the dependency in j.

i) By (4.3) it is easy to see that $M_N(\alpha)$ is an empirical mean of iid bounded random variables. Thus, by the Strong Law of Large Number (SLLN)

$$M_N(\alpha) \xrightarrow[N\to+\infty]{p.s} M(\alpha),$$

with $M(\alpha) = \mathbb{E}[\epsilon^2] + \mathbb{E}[(T_\alpha^2(X(\alpha), \tilde{f}(X(\alpha))) - T_{\alpha^*}^2(X(\alpha^*), \tilde{f}(X(\alpha^*))))^2]$.

$M(\alpha)$ is continuous and has an obvious unique minimum α^*. Since Θ is compact, this implies that $\inf_{\alpha: d(\alpha,\alpha^*)\geq\epsilon} M(\alpha) > M(\alpha^*)$ is satisfied (see Problem 27 p. 84 in [26]).

It remains to prove that $\{m_\alpha : \alpha \in \Theta\}$ is a Glivenko-Cantelli class. Thanks to the remark following the proof of Theorem 5.9 in [26], this is an easy consequence of the continuity of $\alpha \to m_\alpha$ and the fact that the function is bounded by a continuous and integrable function on $[0, 1]$. Indeed, it exists at least a function f^* in $\mathcal{F}_{\theta_0}$ which bounds every other functions, and two constants $K_1 > 0, K_2 > 0$ such that

$$m_\alpha(x) \leq K_1(f^*(x) + K_2)^2,$$

and

$$\sup_{\alpha\in\Theta} |M_N(\alpha) - M(\alpha)| \xrightarrow[N\to+\infty]{\mathbb{P}} 0. \tag{4.14}$$

The result follows from the Theorem 5.7 in [26].

ii) The Central Limit Theorem will be a consequence of Theorem 5.23 in [26]. Recall that

$$m_\alpha(x) = [f(x) - \lambda((x(\alpha) - 1)\sin\theta + \cos\theta \tilde{f}(x(\alpha)))]^2.$$

By the Implicit Function Theorem, that is easy to see that $\alpha \to x(\alpha)$ is C^1 on a compact set. This implies that the norm of the gradient of m_α is uniformly bounded in α. Hence $\exists \dot{\phi}(x) \in L^1$ such that $||\nabla_\alpha m_\alpha(x)|| \leq \dot{\phi}(x)$ hence

$$|m_{\alpha_1}(x) - m_{\alpha_2}(x)| \leq \dot{\phi}(x) \times ||\alpha_1 - \alpha_2||.$$

In order to give an explicit formula for the limit variance, we apply the results of Example 5.27 in [26] where f_θ becomes in our case T^2_α and hence, we have

$$\sqrt{N}(\hat{\alpha}^j_N - \alpha^*_j) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha^*_j}),$$

with $\Gamma_{\alpha^*_j} = V^{-1}_{\alpha^*_j} 2\sigma^2$ and $V_{\alpha^*_j} = 2\mathbb{E}[\dot{T}^2_{\alpha^*_j} \dot{T}^{2\mathbf{T}}_{\alpha^*_j}]$.

□

Proof of Theorem 2.

i) To prove the consistency of $\hat{\hat{\alpha}}_N$ we have to show that

$$\sup_{\alpha\in\Theta} |\hat{M}_N(\alpha) - M(\alpha)| \xrightarrow[N\to+\infty]{\mathbb{P}} 0.$$

We have,

$$\sup_{\alpha\in\Theta} |\hat{M}_N(\alpha) - M(\alpha)| \leq \sup_{\alpha\in\Theta} |\hat{M}_N(\alpha) - M_N(\alpha)| + \sup_{\alpha\in\Theta} |M_N(\alpha) - M(\alpha)|.$$

It has been shown in the proof of Theorem 1 that

$$\sup_{\alpha\in\Theta} |M_N(\alpha) - M(\alpha)| \xrightarrow[N\to+\infty]{\mathbb{P}} 0.$$

It remains to prove that

$$\sup_{\alpha\in\Theta} |\hat{M}_N(\alpha) - M_N(\alpha)| \xrightarrow[N\to+\infty]{\mathbb{P}} 0.$$

To ease the notation, we write $T^2_\alpha(i,N) = T^2_\alpha(X_i(\alpha,N), \tilde{f}_N(X_i(\alpha,N)))$, and $T^2_\alpha(i) = T^2_\alpha(X_i(\alpha), \tilde{f}(X_i(\alpha)))$. Set

$$\begin{aligned} D_N(\alpha) &= M_N(\alpha) - \hat{M}_N(\alpha) \\ &= \frac{1}{N}\sum_{i=1}^{N} 2f(X_i)[T^2_\alpha(i,N) - T^2_\alpha(i)] - \frac{1}{N}\sum_{i=1}^{N}[T^2_\alpha(i,N) - T^2_\alpha(i)][T^2_\alpha(i,N) + T^2_\alpha(i)]. \end{aligned}$$

As f and T^2_α are continuous and bounded on $\Theta \times [0,1]$, this implies that:

$$\begin{aligned} |D_n(\alpha)| &\leq |\frac{1}{N}\sum_{i=1}^{N} 2f(X_i)[T^2_\alpha(i,N) - T^2_\alpha(i)]| + |\frac{1}{N}\sum_{i=1}^{N}[T^2_\alpha(i,N) - T^2_\alpha(i)][T^2_\alpha(i,N) + T^2_\alpha(i)]| \\ &\leq K(\frac{1}{N}\sum_{i=1}^{N}[T^2_\alpha(i,N) - T^2_\alpha(i)]^2)^{\frac{1}{2}} \\ &\leq K'(\frac{1}{N}\sum_{i=1}^{N}[(X_i(\alpha,N) - X_i(\alpha))(1+C) + (\tilde{f}_N(X_i(\alpha)) - \tilde{f}(X_i(\alpha))]^2)^{\frac{1}{2}}. \end{aligned}$$

By construction, there exists j such that $X_{(j)} \leq X_i(\alpha) \leq X_{(j+1)}$, and $X_{(j)} \leq X_i(\alpha,N) \leq X_{(j+1)}$ which leads to:

$$X_i(\alpha,N) - X_i(\alpha) = \gamma(X_{(j+1)} - X_{(j)}).$$

Besides, there exists $\gamma' > 0$ such that $\tilde{f}_N(X_i(\alpha)) = \gamma'\tilde{f}(X_{(j+1)}) + (1-\gamma')\tilde{f}(X_{(j)})$. C and γ' being uniform constants, we have

$$\begin{aligned} |\tilde{f}_N(X_i(\alpha)) - \tilde{f}(X_i(\alpha))| &\leq \gamma'|\tilde{f}(X_{(j+1)}) - \tilde{f}(X_{(j)})| \\ &\leq C\gamma'|X_{(j+1)} - X_{(j)}|, \end{aligned}$$

and

$$\begin{aligned} |D_n(\alpha)| &\leq K'(\frac{1}{N}\sum_{j=1}^{N}[X_{(j+1)} - X_{(j)}]^2)^{\frac{1}{2}} \\ \sup_{\alpha\in\Theta}|D_n(\alpha)| &\leq K'(\frac{1}{N}\sum_{j=1}^{N}\sup_{\alpha\in\Theta}[X_{(j+1)} - X_{(j)}]^2)^{\frac{1}{2}} \\ &\leq K'\sup_j|X_{(j+1)} - X_{(j)}|. \end{aligned}$$

By Lemma 1 $\mathbb{P}(K'\sup_j|X_{(j+1)} - X_{(j)}| \geq \epsilon) \xrightarrow[N\to+\infty]{} 0$. Hence D_N is bounded by an integrable and continuous function which goes to 0 in

probability on Θ

$$\sup_{\alpha\in\Theta}|\hat{M}_N(\alpha) - M(\alpha)| \xrightarrow[N\to+\infty]{\mathbb{P}} 0.$$

So we may conclude.

ii) First, we use that $\sqrt{N}(\hat{\hat{\alpha}}_N - \alpha^*) = \sqrt{N}(\hat{\hat{\alpha}}_N - \hat{\alpha}_N) + \sqrt{N}(\hat{\alpha}_N - \alpha^*)$. By Theorem 3, $\sqrt{N}(\hat{\alpha}_N - \alpha^*) \xrightarrow[N\to+\infty]{\mathcal{L}} \mathcal{N}(0, \Gamma_{\alpha^*})$ with Γ_{α^*} defined in (4.6). It remains to prove that $\sqrt{N}(\hat{\hat{\alpha}}_N - \hat{\alpha}) \xrightarrow[N\to+\infty]{\mathbb{P}} 0$.

Using the same arguments as in the proof i), we have

$$\mathbb{P}(\sqrt{N}\sup_{\alpha\in\Theta}|\hat{M}_N(\alpha) - M_N(\alpha)| \geq \epsilon) \leq \mathbb{P}(K\sqrt{N}\sup_{\alpha\in\Theta}|X_{(j+1)} - X_{(j)}| \geq \epsilon) \quad (4.15)$$

The right hand side of (4.15) converges to 0 by Lemma 3. This implies that $\sqrt{N}(\hat{\hat{\alpha}}_N - \hat{\alpha}) \xrightarrow[N\to+\infty]{\mathbb{P}} 0$.

□

4.5 Perspectives and conclusion

One of the main quality of our approach is that it is easy to implement and execute. The cost function being simple, we use a BFGS algorithm to find the optimal parameters, and because of the regularity of curves we deal with, the initial points for optimization can be easily defined. Furthermore, the search of the coordinate of the reference curve which is sent to the coordinate of the curve to fit can be easily implemented with a simple value search.

We have seen that our methodology performs very well when the data we are dealing with are close to the chosen model. Besides, the deformation parameters can be exploited through an explainable model such as the linear model used in the real world problem.

It seems that the deformation model is robust if the noise is controlled. An interesting extension of this work would be to study what is going on when the reference curve is noisy. A generalization of this work to less regular functions would be worthwhile. Finally, it would be interesting to include in the model a way to handle discontinuities in order to reduce the dimension and have a more global representation of the deformation.

Bibliography

[1] Allassonniere, S., Bigot, J., Glaunes, J., Maire, F., F., R.: Statistical models for deformable templates in image and shape analysis. Annales Mathematiques Blaise Pascal **20(1)**, 1–35 (2013)

[2] Barron, A.R., Cohen, A., Dahmen, W., DeVore, R.A.: Approximation and learning by greedy algorithms. Ann. Statist. **36**(1), 64–94 (2008). DOI 10.1214/009053607000000631. URL https://doi.org/10.1214/009053607000000631

[3] Bigot, J., Christophe, C., Gadat, S.: Random action of compact lie groups and minimax estimation of a mean pattern. IEEE Transactions on Information Theory **58**(6), 3509–3520 (2012)

[4] Bigot, J., Gadat, S.: A deconvolution approach to estimation of a common shape in a shifted curves model. Ann. Statist. **38**(4), 2422–2464 (2010). DOI 10.1214/10-AOS800. URL https://doi.org/10.1214/10-AOS800

[5] Bigot, J., Gamboa, F., Vimond, M.: Estimation of translation, rotation, and scaling between noisy images using the fourier–mellin transform. SIAM Journal on Imaging Sciences **2**, 614–645. ISSN 2936-4954

[6] Fournier, E., Grihon, S., Klein, T.: Semiparametric estimation of plane similarities: application to fast computation of aeronautic loads. Statistics **53**(5), 1168–1186 (2019)

[7] Fournier, E., Klein, T., Grihon, S.: A case study: Influence of dimension reduction on regression trees-based algorithms-predicting aeronautics loads of a derivative aircraft. Journal de la Société Française de Statistique **159**(3), 56–78 (2018)

[8] Gamboa, F., Loubes, J.M., Maza, E.: Semi-parametric estimation of shifts. Electron. J. Statist. **1**, 616–640 (2007). DOI 10.1214/07-EJS026. URL https://doi.org/10.1214/07-EJS026

[9] Gasser, T., Kneip, A.: Searching for structure in curve samples. Journal of the american statistical association **90**(432), 1179–1188 (1995)

[10] Gassiat, E., Lévy-Leduc, C.: Efficient semiparametric estimation of the periods in a superposition of periodic functions with unknown shape. Journal of Time Series Analysis **27**(6), 877–910 (2006)

[11] Golubev, G.: Estimation of the period of a signal with an unknown form against a white noise background. Problems Inform. Transmission **24**(4), 288–299 (1988)

[12] Grenander, U.: General pattern theory. Oxford Science Publications (1993)

[13] Hardle, W., Marron, J.S.: Semiparametric comparison of regression curves. The Annals of Statistics pp. 63–89 (1990)

[14] Ke, C., Wang, Y.: Semiparametric nonlinear mixed-effects models and their applications. Journal of the American Statistical Association **96**(456), 1272–1298 (2001)

[15] Kneip, A., Engel, J.: Model estimation in nonlinear regression under shape invariance. The Annals of Statistics pp. 551–570 (1995)

[16] Kneip, A., Li, X., MacGibbon, K., Ramsay, J.: Curve registration by local regression. Canadian Journal of Statistics **28**(1), 19–29 (2000)

[17] Lawton, W., Sylvestre, E., Maggio, M.: Self modeling nonlinear regression. Technometrics **14**(3), 513–532 (1972)

[18] Lindstrom, M.J.: Self-modelling with random shift and scale parameters and a free-knot spline shape function. Statistics in medicine **14**(18), 2009–2021 (1995)

[19] Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on Signal Processing **41**(12), 3397–3415 (1993). DOI 10.1109/78.258082

[20] McGuire, M.: An image registration technique for recovering rotation, scale and translation parameters. Tech. rep., NEC Tech Report (1998). URL http://casual-effects.com/research/McGuire1998ParameterRecovery/index.html

[21] Nocedal, J., Wright, S.J.: Numerical Optimization, second edn. Springer, New York, NY, USA (2006)

[22] Ramsay, J.O., Li, X.: Curve registration. Journal of the Royal Statistical Society: Series B (Statistical Methodology) **60**(2), 351–363. DOI 10.1111/1467-9868.00129

[23] Ramsay, J.O., Wickham, H., Graves, S., Hooker, G.: fda: Functional Data Analysis (2018). URL https://CRAN.R-project.org/package=fda. R package version 2.4.8

[24] Ritter, M., Dillinger, J.: Nonlinear numerical flight dynamics for the prediction of maneuver loads. IFASD 2011 pp. 1–10 (2011)

[25] Sancetta, A.: Greedy algorithms for prediction. Bernoulli **22**(2), 1227–1277 (2016). DOI 10.3150/14-BEJ691. URL https://doi.org/10.3150/14-BEJ691

[26] Van der Vaart, A.W.: Asymptotic statistics. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press (1998)

[27] Vimond, M.: Efficient estimation for homothetic shifted regression models. Université Paul Sabatier, Laboratoire de statistique et probabilités (2006)

Chapter 5

Prediction of preliminary maximum wing bending moments under discrete gust

Edouard Fournier*[1,2,4], Stéphane Grihon†[2], Christian Bes‡[3], Thierry Klein§[1,3]

[1]Institut de Mathématiques, UMR5219; Université de Toulouse; CNRS, UPS IMT, F-31062 Toulouse Cedex 9, France
[2]Airbus France 316, Route de Bayonne, Toulouse France
[3]Mechanical Engineering Department, Université de Toulouse, 118 Route de Narbonne, Cedex 4, Toulouse
[4]ENAC - Ecole Nationale de l'Aviation Civile, Université de Toulouse, France

Résumé

Ce chapitre est consacré à l'estimation préliminaire de charges maximales aéronautiques. Ces charges sont déterminantes et critiques pour le dimensionnement d'une structure. Un nombre important de simulations est nécessaire pour produire ces charges maximales. A partir d'une base de données d'un avion initial, l'idée est de construire un métamodèle pour prédire les charges maximales de deux avions dérivés. En effet, toutes les charges produites par les simulations ne

*edouard.fournier@airbus.com
†stephane.grihon@airbus.com
‡christian.bes@univ-tlse3.fr
§thierry.klein@math.univ-toulouse.fr or thierry01.klein@enac.fr

sont pas utiles. Il est donc naturel de se concentrer sur les charges les plus critiques que subit la structure.

Nous mettons en place une méthode rapide de modélisation. Le cadre proposé est celui d'une régression polynomiale parcimonieuse obtenue par un développement de Taylor au second ordre. Le modèle parcimonieux est construit à l'aide d'une méthode de régression *Greedy*. Nous utilisons le métamodèle obtenu pour estimer les charges maximales subies par l'aile de deux avions dérivés. Les résultats obtenus sont très satisfaisants. La méthode permet donc un pré-dimensionnement rapide de la structure.

Ce chapitre correspond à l'engineering note publiée [2].

5.1 Introduction

Many methodologies (see [4, 10, 6, 11]), have been proposed to quickly identify among a very large number of flight conditions and maneuvers (i.e., steady, quasi-steady and unsteady loads cases) the ones which give the worst values for structural sizing (e.g., bending moments, shear forces, torques,...). All of these methods use both the simulation model of the aircraft under development and efficient algorithms to find out the critical points of the flight envelope. At the preliminary structural design phases detailed models are not available and airframe's loads are estimated by empirical relationships or engineering judgments. These approximations can induce load uncertainties and may lead to expensive redesign activities through the upcoming detailed sizing process (see [7, 5]). In the context of preliminary design phase for a new aircraft variant, to overcome this likely drawback, we propose a method based on the huge and reliable database of an initial aircraft from which the new variant belongs. This new aircraft variant is, what we call in the following, a new weight variant aircraft: the aircraft has a different maximum take off weight but has no geometric change with respect to the initial aircraft. More precisely, from the load cases of this initial database, response surfaces are identified as functions of preliminary parameters (flight conditions and structural parameters). Then, these response surfaces are used to predict quickly the weight aircraft variant quantities of interest for preliminary structural design studies. Although the proposed method can be readily extended to any structural quantity of interest and to any flight conditions and maneuvers, it is presented here for the prediction of the bending moments due to discrete gust at different locations along a wing span.

This note is organized as follows. Section 2 presents the initial aircraft

database where its values are derived from a detailed aeroelastic model. This database is composed of the maximum temporal value of the bending moment due to the discrete gust at any wing span location and for any point inside the flight envelope. These maximum values are identified by few preliminary parameters (altitude, mass, speed, gust length, etc.). Section 3, describes the Orthogonal Greedy Algorithm (OGA) which permits to obtain the coefficients of the response surfaces from the initial database as a function of the preliminary parameters. This algorithm is based on parsimony principle and aims to protect against under and over fitting of the response surface when it is used to predict the maximum bending moment for a weight variant aircraft. Section 4 presents results of the predictions and the confidence bounds of the expected maximum temporal bending moment along the wing span for weight aircraft variants. These surface response predictions are further compared and validated with existing weight aircraft variants database results. Finally, in Section 5, conclusion and future works are presented.

5.2 Initial aircraft database

The database of the initial aircraft wing at hand is built from a detailed aeroelastic model which have been updated from ground and flight tests. This aeroelastic model under gust [18, 6] can be expressed in the Laplace domain as follows

$$(s^2\mathbf{M} + s\mathbf{C} + K - q_\infty \mathbf{Q}_{GG}(s))z(s) = \frac{q_\infty}{\mathbf{V}}\mathbf{Q}_{Gg}(s)u(s) \tag{5.1}$$

with $u(s)$ is the gust sequence, $z(s)$ is the structural response, $\mathbf{K}$, $\mathbf{C}$, $\mathbf{M}$ are respectively the stiffness, the damping and the mass matrices, $\mathbf{Q}_{GG}$ is the motion-induced unsteady aerodynamic matrix, $\mathbf{Q}_{Gg}$ is the gust-induced unsteady aerodynamic matrix, q_∞ is the freestream dynamic pressure, V is the the air velocity. This aeroelastic equation (5.1) is often transformed into modal coordinates to reduce the size of the computational problem and the unsteady generalized forces are approximated by rational functions in the Laplace domain [12]. Using space-state formulation, or inverse Fourier transform, the loads and structural forces (shear forces, bending moments, torques) are computed from internal loads given by

$$\mathbf{F}(t) = -\mathbf{M}\ddot{z}(t) - \mathbf{F}_a(t) + \mathbf{F}_g(t), \tag{5.2}$$

where $\mathbf{F}_a(t)$ are the temporal unsteady aerodynamic forces and $\mathbf{F}_g(t)$ are the temporal gust forces.

From the simulation model (5.2), the maximum temporal bending moment due to the discrete gust is computed for each flight point within the flight envelope. Airbus describes preliminary parameters by a vector of parameters $\mathbf{x} \in \mathbb{R}^d$, with $d = 20$. These preliminary parameters are: the aircraft mass, the zero fuel aircraft mass, the quantity of fuel, the true air speed, the Mach number, the altitude, the load factors (NX, NY, NZ), the coordinates of the center of gravity (CGx, CGy, CGz) and the moments of inertia (Ixx, Ixy, Ixz, Iyy, Iyz, Izz), the gust length (H) of the discrete gust and the flight profile alleviation factor F_g. Note that the gust velocity profile has the following form [18]

$$u(t) = \frac{U_{max}}{2}(1 - \cos\frac{2\pi t}{H}), \tag{5.3}$$

with $0 \leq t \leq \frac{H}{V}$ or $u(t) = 0$ if $t > \frac{H}{V}$, H (in feet) being the gust length also called the gust gradient distance. Each gust velocity profile has its own maximum gust velocity U_{max} given for a reference gust velocity U_{ref} (depending on the altitude of the aircraft). U_{max} satisfies

$$U_{max} = U_{ref}F_g(\frac{H}{350})^{\frac{1}{6}}. \tag{5.4}$$

F_g depends on the altitude, the maximum take off weight, the maximum landing weight and the maximum zero fuel weight. The bending moments are time dependent and due to the natural frequencies of the aircraft, a sufficient number of gust gradient distance in the range of 30 feet to 350 feet must be considered to get the maximal temporal bending moment with $0 \leq t \leq \frac{2H}{V}$. Note that in our study, 12.5 times the mean geometric chord of the aircraft's wing does not exceed 350 feet. For each preliminary parameter $\mathbf{x}$ and any location k, $1 \leq k \leq K$ ($K = 45$) of the wing span, the simulation model (5.2) gives the temporal bending moments $M(k, \mathbf{x}, t)$. Hence, for a given preliminary parameter $\mathbf{x}$ and a given section k, the maximum temporal bending moments is given by

$$M_B(k, \mathbf{x}) = \max_{0 \leq t \leq \frac{2H}{V}} M(k, \mathbf{x}, t). \tag{5.5}$$

Then, $M_B(k, \mathbf{x})$ constitute the initial database where the response surfaces will be fitted. For structural sizing, the quantities of interest are $M_B^*(k) = \max_{\mathbf{x}} M_B(k, \mathbf{x})$ for all k, which is the maximum of the maximum temporal bending moment over the preliminary parameters envelope. In other words, for any wing location k, $M_B^*(k)$ represents the most critical bending moment due to the discrete gust over the flight envelope. It should be noticed that for a given aircraft, the critical preliminary parameter which gives the maximum of the maximum temporal bending moment (i.e, $M_B^*(k)$) can differ from a wing span location to an other. Note that the critical preliminary parameter is not necessarily the same from a section

to another. Moreover, for a given location on the wing span, the critical preliminary parameter is not necessarily the same than the one of the initial aircraft.

5.3 Maximal bending moments vs Preliminary parameter response surface

Although there exist a lot of surrogate models such as GP-Kriging (see [4, 10]), Regression Trees [3] or Neural Networks [10] to approximate bending moments, we have chosen to develop a second order polynomial expansion. This response surface has the advantage to give interpretable results of the influence of each component of the preliminary parameters. For a given location k along the wing span and a preliminary parameter $\mathbf{x}$, the maximal temporal bending moment is approximated by

$$\hat{M}(k,\mathbf{x}) = a_0(k) + \sum_{i=1}^{d} a_i(k)x_i + \sum_{i=1}^{d}\sum_{j=i}^{d} a_{i,j}(k)x_i x_j = \sum_{i=1}^{D} \omega_i(k)\Phi_i(\mathbf{x}). \quad (5.6)$$

For each location k, the $\omega_i(k)$ ($i = 1, ..., D,\ D = \frac{d^2+3d+2}{2} = 231$) are the regression coefficients to be estimated from the values of the initial database $M_B(k,\mathbf{x})$ over the preliminary parameters envelope. The number of points of the preliminary parameters envelope is denoted by n. This number of regression coefficients is quite low compared to the number n of bending moments contained in the initial database. Instead of using Ordinary Least Square [13] to estimate all the D coefficients, we prefer to use an algorithm which is based on parsimony principles that aims to avoid overfitting. Among the existing algorithms such as Ridge [9, 17], Lasso [17], we choose the Orthogonal Greedy Algorithm (OGA) [14, 8]. Ordinary Least-Squared (OLS) problem may give results with a better accuracy. Nevertheless, the main advantage of the OGA is its sparse representation: this sparseness allows to select the most relevant coefficients which prevents against over fitting when predicting weight variants. This is not generally the case when using Ordinary Least-Squared where all the coefficients are taken into account. Moreover, due to its sequential structure, the OGA has a really good computer time compared to an OLS problem.

The Orthogonal Greedy Algorithm works as follows. From the initial database and for a given location k, $1 \leq k \leq K$, along the wing span, we have n maximum temporal bending moments $M_B(k,\mathbf{x}_i), i = 1, ..., n$. We build the matrix of preliminary parameters $\mathbf{X} = [\mathbf{x}_1, ..., \mathbf{x}_n]^{\mathbf{T}} \in \mathcal{M}_{nxd}(\mathbb{R})$ (a row of $\mathbf{X}$ is a $\mathbf{x}_i$) and the corresponding vector of maximum temporal bending moment at the station k denoted

by $\mathbf{M} = (M_B(k, \mathbf{x}_1), ..., M_B(k, \mathbf{x}_n))^{\mathbf{T}} = (M_1, ..., M_n)^{\mathbf{T}} \in \mathbb{R}^n$. Knowing the different polynomial basis functions to be used, we can then build the matrix $\Phi = (\Phi_j(\mathbf{x}_i))_{\substack{i=1,...,n \\ j=1,...,D}} \in \mathcal{M}_{nxD}(\mathbb{R})$.

The number $l \in \mathbb{N}^*, 1 \leq l \leq D$, is the number of regression coefficients to be estimated and is fixed by the user. The OGA algorithm can be described by the following steps

Algorithm 9 Algorithm OGA

1: *Set:* $l \in \mathbb{N}^*$
2: *Set:* $h_0 = 0$
3: **for** $j = 1, ..., l$ **do**:
4: $\quad s(j) := \underset{1 \leq k \leq D}{\operatorname{argmin}} |\frac{1}{n} \sum_{i=1}^{n} (M_i - h_j) \Phi_k(\mathbf{x}_i)|$
5: $\quad P^j := OLS\ operator\ on\ span\{\Phi_{s(1)}, ..., \Phi_{s(j)}\}$
6: $\quad h_j := P^j \mathbf{M}$

h_j is the projection of M onto the orthogonal $span\{\Phi_{s(1)}, ..., \Phi_{s(j)}\}$. The aim of the algorithm is to choose the best l functions among the D initial basis functions which minimize the residuals. A Cross-Validation [16] technique is applied to chose the optimal number of unsignificant basis functions among the l. Notice that criteria such as AIC [1] or BIC [15] can be also implemented. These criteria are measures penalizing models with too many basis functions, thereby allowing to have parsimonious models. Roughly speaking, AIC and BIC make balance between the number of coefficients to be identified (i.e, the number of regressors) and the least squared error.

Once the regression coefficients are obtained by the OGA, the response surfaces are used to predict quickly the expected maximum temporal bending moment $\hat{M}(k, \mathbf{x}_{wv})$ for each preliminary parameter $\mathbf{x}_{wv,\ i},\ i = 1, ..., n_{wv}$ of the new aircraft variant and at any location k of the wing span. Notice that the preliminary parameters of the weight variant are different from those of the initial database aircraft. From these values, the maximum of the expected maximum temporal values over the preliminary parameters envelop of the weight variant is directly extracted at each location $\hat{M}^*_{wv}(k) = \max_{i=1,...,n_{wv}} \hat{M}(k, \mathbf{x}_{wv,\ i})$. Moreover, if the error has a normal distribution, we can easily compute the prediction intervals as in [13]. All of these quantities of interest are then used for structural design studies.

5.4 Numerical experiments

The initial database comes from an Airbus aircraft having 235t as maximum take off weight with $K = 45$ locations distributed along the wing span. This database is composed of $n = 1560$ preliminary parameters representing the flight and structural envelope. Recall that each preliminary parameter is represented by 20 values. From each of these preliminary parameters, the temporal maximal bending values are computed along the 45 wing span locations. Using the OGA algorithm with $l = 80$ and a Cross-Validation technique (6-folds), for each wing span location we have identified around 50 regression coefficients (instead of 231 coefficients when applying Ordinary Least Squares). The 210t and 280t are weight aircraft variants already developed and have the same wing geometry as the 235t. For each new weight variant, each database is composed of $n_{wv_{210}} = n_{wv_{280}} = 1560$ preliminary parameters, but the preliminary parameter envelops are different. Therefore, it is possible to compare and validate the response surface predictions with reliable bending moments contained in the existing database of the aircraft variants (see Figures 5.1 (b) and (d)).

At any location k of the wing span, using the associated surface responses we have computed the expected maximum of the maximum temporal bending moment of the new aircraft variants, denoted by $\hat{M}^*_{wv}(k) = \max_{i=1,\dots,n_{wv}} \hat{M}(k, \mathbf{x}_{wv,\,i})$, over the preliminary parameter envelope (in black in Figure 5.1). This maximum $\hat{M}^*_{wv}(k)$ has been compared to the maximum bending moment $M^*_{B,\,wv}(k)$ given by the database (in red in Figure 5.1). The 210t aircraft gives a maximal relative error along the wing span of 2% and a relative error at the wing root of 0.8% when comparing the response surfaces predictions with the 210t existing database. The 280t aircraft gives a maximal relative error along the wing span of 3.5% and a relative error at the wing root of 0.9%. After having checked the normality of the residuals using Kolmogorov-Smirnoff and Shapiro-Wilk tests, we have computed the 99% prediction interval bounds (see [13]) provided by the response surfaces (in grey in Figures 5.1 (a) and (c)). It should be noted that the maximum bending moments provided by the two databases are quite close to the center of the prediction interval for any location of the wing span. Indeed, the maximal width of these prediction intervals is 8% over the wing span. Moreover, all the bending moments of the 210t and 280t never exceed the lower bound of the prediction intervals. The above results obtained on real world aircraft cases show in particular that the second order response surfaces are quite competitive for preliminary design studies.

Besides the good prediction results in terms of preliminary design study, the proposed methodology requires a short computational time on a standard desk computer. It has taken around 10 minutes for computing the

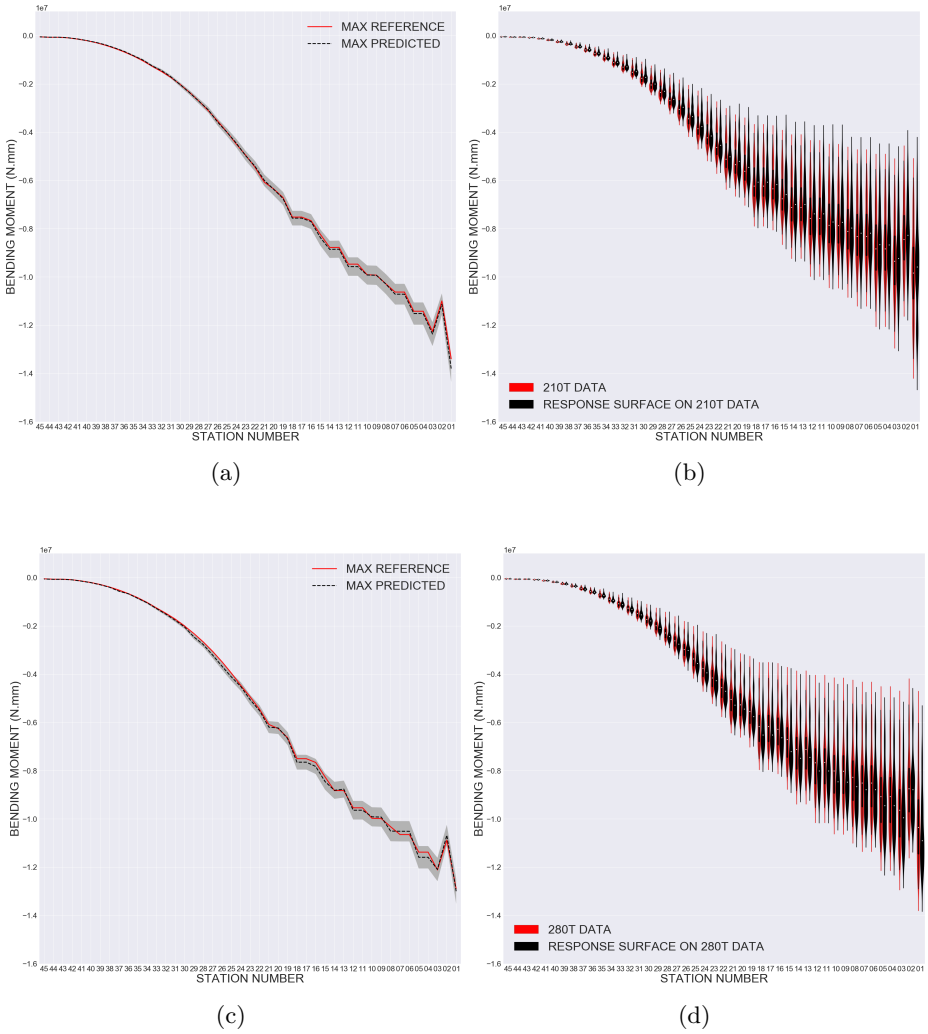


Figure 5.1: Maximum bending moment (the red line corresponds to the true maximum bending moment, the black dashed line corresponds to the predicted maximum, the grey zone corresponds to the prediction interval) - Data distribution and Response surfaces for all the stations: (a) Extrapolation of 210t maximum, (b) Data distribution and Response surfaces of 210t, (c) Extrapolation of 280t maximum, (d) Data distribution and Response surfaces of 280t

expected and the prediction intervals of the maximum bending moments along the wing span for the two weight variants. More precisely, we have built 45 surface responses (i.e the coefficients for each location) from the $1560 \times 45 = 70200$ initial data of the 235t. We have predicted the $1560 \times 45 \times 2 = 140400$ temporal maximum bending moments of the 210t and 280t. Finally, we have extracted from the response surface both the

expected maximum of the maximum temporal bending moment and the associated prediction intervals at each location of the wing span for the two aircraft variants.

5.5 Conclusion

This note has presented a reliable and fast methodology for estimating critical load cases for weight variants aircraft at the preliminary design phase. After having identified the set of preliminary parameters, the proposed methodology used the Orthogonal Greedy Algorithm to identify the coefficient of the second order polynomial response surfaces from an initial database aircraft from which the weight variants aircraft belongs. This Orthogonal Greedy Algorithm is highly efficient in terms of computational time and parsimony representation for estimating the regression coefficients. We reported very encouraging results concerning a real world case of wing bending moments. Indeed, having identified response surfaces for the 235t Airbus aircraft, the maximal errors predictions are 2% for the 210t variant and 3.5% for the 280t variant. It suggests also that the design process will produce small predictable variations in structural form and behavior in response to moderate variations of the aircraft and flight conditions (i.e, a second order approximation). Hence, this approach would be very valuable for any manufacturer with access to all the necessary data. Future related work will attempt to extend our approach to other structural parts and flight conditions (i.e continuous turbulence, landings, fuselage, ...).

Bibliography

[1] Akaike, H.: Information theory and an extension of the maximum likelihood principle. Second International Symposium on Information Theory pp. 267–281 (1973). DOI 10.1007/978-1-4612-1694-0_15

[2] Fournier, E., Grihon, S., Bes, C., Klein, T.: Prediction of preliminary maximum wing bending moments under discrete gust. Journal of Aircraft **56**(4), 1722–1725 (2019)

[3] Fournier, E., Klein, T., Grihon, S.: A case study: Influence of dimension reduction on regression trees-based algorithms-predicting aeronautics loads of a derivative aircraft. Journal de la Société Française de Statistique **159**(3), 56–78 (2018)

[4] Haddad Khodaparast, H., Georgiou, G., Cooper, J., Riccobene, L., Ricci, S., Vio, G., Denner, P.: Efficient worst case "1-cosine" gust loads prediction. Journal of Aeroelasticity and Structural Dynamics **2**(3) (2012). DOI 10.3293/asdj.2012.17

[5] Howe, D.: Aircraft loading and structural layout. AIAA Education Series. (2004). DOI 10.2514/4.477041

[6] Khodaparast, H.H., Cooper, J.: Rapid prediction of worst-case gust loads following structural modification. AIAA journal **52**(2), 242–254 (2014). DOI 10.2514/1.J052031

[7] Lomax, T.L.: Structural loads analysis for commercial transport aircraft: theory and practice. American Institute of Aeronautics and Astronautics (1996). DOI 10.2514/4.862465

[8] Mallat, S.G., Zhang, Z.: Matching pursuits with time-frequency dictionaries. IEEE Transactions on Signal Processing **41**(12), 3397–3415 (1993). DOI 10.1109/78.258082

[9] Marquardt, D.W., Snee, R.D.: Ridge regression in practice. The American Statistician **29**(1), 3–20 (1975). URL http://www.jstor.org/stable/2683673

[10] Nazzeri, R., Haupt, M., Lange, F., Sebastien, C.: Selection of Critical Load Cases Using an Artificial Neural Network Approach for Reserve Factor Estimation. Deutsche Gesellschaft fur Luft-und Raumfahrt-Lilienthal-Oberth eV (2015)

[11] Papila, M., Haftka, R.T.: Response surface approximations: noise, error repair, and modeling errors. AIAA journal **38**(12), 2336–2343 (2000). DOI 10.2514/2.903

[12] Poirion, F.: Multi-mach rational approximation of generalized aerodynamic forces. Journal of aircraft **33**(6), 1199–1201 (1996). DOI 10.2514/3.47076

[13] Rencher, A.C., Schaalje, G.B.: Linear models in statistics. John Wiley & Sons (2008)

[14] Sancetta, A.: Greedy algorithms for prediction. Bernoulli **22**(2), 1227–1277 (2016). DOI 10.3150/14-BEJ691

[15] Schwarz, G.: Estimating the dimension of a model. The annals of statistics **6**(2), 461–464 (1978). DOI 10.1214/aos/1176344136

[16] Stone, M.: Cross-validatory choice and assessment of statistical predictions. Journal of the royal statistical society. Series B (Methodological) pp. 111–147 (1974)

[17] Tibshirani, R.: Regression shrinkage and selection via the lasso. Journal of the Royal Statistical Society, Series B **58**, 267–288 (1994)

[18] Zhao, Y., Yue, C., Hu, H.: Gust load alleviation on a large transport airplane. Journal of Aircraft **53**(6), 1932–1946 (2016). DOI 10.2514/1.C033713

Conclusions

This thesis deals with statistical learning methods for predicting aeronautical loads and stress.

The first problem we have encountered was to predict loads for different maximum take off weight. In a previous project concerning loads, it has been shown that the family of regression trees works well on the data we have to deal with. Nevertheless, it is a well-known fact that regression-trees based models are not efficient in extrapolation. As a consequence, we have associated the regression-trees based methods to dimension reduction techniques and evaluated their extrapolation influence. It appears that, due to the high co linearity of the outputs, a PCA on the outputs increases slightly the prediction results. Even if we obtained proper results for extrapolation, the scores obtained for the weight variants 247 tonnes and 251 tonnes should have been better. Indeed, the maximum mass variation does not exceed 6% of the 238 tonnes aircraft, hence the loads should not be that different. Aeronautical engineers found the source of the problem being that the 247t and 251t datasets have been produced from a slightly different aerodynamic model than the 238t and the 242t, leading to uncatchable differences by the regression algorithms.

The exploratory study in Chapter 3 led us to consider a semiparametric model in Chapter 4. Indeed, loads are represented by curves having a similar shape. It is thus natural to consider that they have been obtained through the deformation of a reference curve. As a consequence, we have developed a new deformation model to be integrated in the modeling process. This deformation model considers deformation operators acting both on the input space and the output space. We have shown that our model is appropriate for the aeronautical loads. Besides, associated to the Orthogonal Greedy Algorithm, it produces a performant method to predict loads in an explainable and a physical way.

The last Chapter has been dedicated to the preliminary estimation of maximal aeronautical loads. In the framework of a sparse polynomial regression obtained by a second order Taylor development, we can estimate

maximal loads suffered by the wing of two weight variants. Results are very satisfying and allow a quick pre-sizing of the structure.

For going forward, we could consider a fine extrapolation mesh to asses the capabilities of any Machine Learning algorithm in this context but would require massive simulations. The explainability of black box models and the estimation of the prediction error in an extrapolation context would be also a good option for future works.

TITLE: Machine Learning for the prediction of aeronautical loads and stress.

Abstract

This thesis focuses on Machine Learning and information extraction for aeronautical loads and stress data. In the first time, we carry out a study for the prediction of aeronautical loads curves. We compare regression trees based models and quantify the influence of dimension reduction techniques on regression performances in an extrapolation context. In the second time, we develop a deformation model acting simultaneously on the input and the output space of the curves. We study the asymptotic properties of the estimators of the deformation parameters. This deformation model is associated to the modeling and predicting process of aeronautical loads. Finally, we give a simple and efficient method for predicting critical loads.

KEYWORDS: Machine Learning, Regression, Deformation model of curves, Aeronautical Loads and Stress.

AUTEUR : Edouard FOURNIER.

TITRE : Méthodes d'apprentissage statistique pour la prédiction de charges et de contraintes aéronautiques.

DIRECTEURS DE THESE : Thierry KLEIN & Fabrice GAMBOA.

LIEU ET DATE DE SOUTENANCE : Salle de conférence MIP, Bâtiment 1R3, Université Paul Sabatier, 14 Octobre 2019.

Résumé

Cette thèse s'intéresse à l'apprentissage et à la représentation de données de charges et contraintes aéronautiques. Nous réalisons dans un premier temps une étude préliminaire pour la prédiction des courbes de charges aéronautiques. Nous comparons des méthodes de régression à base d'arbres et quantifions l'influence de techniques de réduction de dimension sur les performances en régression dans un cadre d'extrapolation. Dans un second temps, nous développons un modèle de déformation agissant simultanément sur les entrées et sorties des courbes. Nous étudions les propriétés asymptotiques des estimateurs des paramètres de déformation. Ce modèle de déformation est associé au processus de modélisation et prédiction des charges aéronautiques. Dans un troisième temps, nous donnons une méthode simple et efficace de prédiction de charges critiques.

MOTS-CLES : Apprentissage, Régression, Modèle de déformation de courbes, Charges et Contraintes Aéronautiques.

DISCIPLINE ADMINISTRATIVE : MITT : Domaine Mathématiques : Mathématiques appliquées.

INTITULE ET ADRESSE DU LABORATOIRE : Université Toulouse 3 Paul Sabatier. Institut de Mathématiques de Toulouse (UMR 5219) 118 Route de Narbonne, 31400.