Détection de ruptures et identification des causes ou des symptômes dans le fonctionnement des turboréacteurs durant les vols et les essais

Cynthia Faure



THÈSE DE DOCTORAT DE
l'UNIVERSITÉ PARIS 1 PANTHÉON SORBONNE

Présentée par

Cynthia FAURE

Pour obtenir le grade de

DOCTEUR EN MATHÉMATIQUES APPLIQUÉES de l'Université Paris 1

Détection ruptures et identification des causes ou des symptômes dans le fonctionnement des turboréacteurs durant les vols et les essais

---

sous la direction de Jean-Marc Bardet et Madalina Olteanu

Composition du jury:

Marie	CHAVENT	Université de Bordeaux	Rapporteur
Erwan	LE PENNEC	École polytechnique	Rapporteur
Jérôme	LACAILLE	Safran Aircraft Engines	Examinateur
Prénom	NOM	Université	Examinateur
Prénom	NOM	Université	Examinateur
Jean-Marc	BARDET	Université Paris 1 Panthéon Sorbonne	Directeur
Madalina	OLTEANU	Université Paris 1 Panthéon Sorbonne	Directrice

Sommaire

1 Introduction

1.1 Motivations

En statistiques, l'étude des séries temporelles occupe une place très importante, puisque dans de très nombreux contextes les données observées le sont au cours du temps. Les études les plus classiques consistent par exemple à modéliser les séries temporelles, estimer les paramètres de ces modèles statistiques, réaliser des tests pour faire de la sélection de modèle et utiliser les modèles pour faire de la prévision.

Nous nous plaçons dans un autre contexte. Au vu d'une série temporelle uni- ou multi-variée, nous ne cherchons pas tant à modéliser la série qu'à détecter les changements de modèles, c'est-à-dire les ruptures, qui peuvent être annonciatrices d'évènements particuliers. Nous cherchons à repérer ces ruptures, à détecter les anomalies, aussi bien par des méthodes *offline*, c'est-à-dire une fois recueilli l'ensemble des données, que par des méthodes *online*, c'est-à-dire au fur et à mesure de l'enregistrement des données.

Ce travail s'est effectué dans le contexte d'un contrat de recherche entre le laboratoire SAMM de l'Université Paris 1 Panthéon-Sorbonne et le pôle *Health Monitoring* de la société *Safran Aircraft Engines*. Le rôle essentiel de ce pôle est de contribuer à la surveillance préventive des moteurs d'avion, tant au moment de leur conception (construction, essai) qu'au cours de leur exploitation en vol.

La société *Safran Aircraft Engines* appartient au groupe *Safran*, leader international en aéronautique, défense et sécurité. *Safran Aircraft Engines* produit et vend des moteurs d'avions civils et militaires ainsi que des services de maintenance de ces moteurs. Notre travail est exclusivement consacré aux moteurs d'avions civils.

Même si le principe de base d'un moteur "à réaction" est plutôt simple, le fonctionnement en est extrêmement sophistiqué, en raison de l'extrême robustesse et l'impressionnante fiabilité qui sont exigées pour des raisons évidentes de sécurité. Un moteur est un système complexe dont le fonctionnement dépend d'un nombre important de pièces et de sous-systèmes. La Figure 1 représente la coupe d'un moteur CFM56-7B qui équipe les Boeing 737.

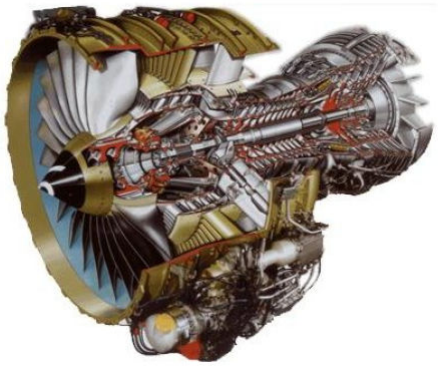


Figure 1: Coupe d'un CFM56-7B, moteur du Boeing 737.

Assurer le bon fonctionnement d'un moteur est une tâche complexe, qui exige le travail conjoint de nombreux professionnels. Des inspections visuelles, des révisions régulières, des changements de pièces et de composants sont évidemment nécessaires. Mais les moteurs font aussi l'objet d'une surveillance en direct, grâce à des capteurs embarqués qui enregistrent des données de vol, de contexte et de fonctionnement. On dispose donc de séries temporelles multivariées qui fournissent des informations importantes quant au bon déroulement du vol et qui sont utilisées pour détecter une éventuelle anomalie. Le cadre applicatif de notre travail de thèse est de proposer de nouvelles techniques utilisables pour détecter des anomalies dans le fonctionnement des moteurs d'avion et analyser leur exploitation en vue de créer de futurs moteurs plus efficaces et mieux adaptés à leur utilisation.

De très nombreux outils ont été développés pour analyser les données de vols afin d'assurer la sécurité, d'améliorer la disponibilité des moteurs et d'optimiser les coûts de maintenance des moteurs.

Beaucoup de travaux portent sur les phases "stationnaires" des séries temporelles enregistrées par les capteurs, voir par exemple les travaux de (Rabenoro, Lacaille, Cottrell, & Rossi, 2014), (Bellas, Bouveyron, Cottrell, & Lacaille, 2013) ou (Lacaille & Gerez, 2011). Cependant les phases "transitoires" n'ont pas été exploitées en profondeur jusqu'à présent, bien qu'elles soient intéressantes à analyser pour détecter des prémices d'anomalies. Par ailleurs, les méthodes de détection de phases transitoires dans les séries temporelles ne sont guère abondantes dans la littérature. L'un des objectifs de notre travail de thèse est de proposer des méthodes permettant de détecter automatiquement les phases transitoires de séries temporelles, de les analyser, de les grouper selon leurs ressemblances et de repérer celles qui semblent récurrentes ou atypiques.

La plupart des méthodes en usage ne traitent que le cas univarié et il s'agit donc d'étendre nos propositions au contexte bivarié, ce qui est très important dans le cadre applicatif, puisque les données disponibles sont multivariées. Nous utilisons pour cela des cartes de Kohonen emboîtées.

Un deuxième objectif de notre travail consiste à définir un outil permettant de retrouver des tronçons de vols similaires à un tronçon donné. Il s'agit d'aider les experts qui aujourd'hui ne peuvent pas mettre en relation automatiquement un tronçon jugé atypique avec des tronçons similaires déjà observés.

Enfin, et plus généralement l'idée principale de cette thèse est de proposer une méthode automatique pouvant transformer une série temporelle en une suite de labels, représentant une suite de comportements distincts. Du point de vue applicatif, l'idée est de résumer les vols entiers par une suite de labels, ce qui permet de les comparer et de les analyser plus facilement, et aussi de détecter et d'identifier des vols atypiques.

1.2 Plan de la thèse

Le manuscrit est composé de 6 chapitres.

Ce premier chapitre d'introduction présente l'objectif de la thèse, les motivations et les contributions.

Le chapitre 2, intitulé "*Safran Aircraft Engines* et le *Prognostic Health Monitoring*", introduit la notion de maintenance des moteurs d'avion, les méthodes utilisées aujourd'hui chez *Safran Aircraft Engines*. De plus, les données de vols disponibles et utilisées pour les applications y sont également présentées et détaillées.

Le chapitre 3, intitulé "Recherche de phases transitoires", présente des méthodes de détection automatique des phases transitoires dans les séries temporelles univariées. Plusieurs méthodes de détection de ruptures existantes dans la littérature sont utilisées afin de repérer des phases. Ainsi, nous développons deux algorithmes de détection de ruptures dans la pente (*offline* et *online*) qui permettent de segmenter une série temporelle. Cette méthode flexible et rapide est appliquée sur des données simulées et sur des données réelles.

Le chapitre 4, intitulé "*Clustering* de phases transitoires", présente deux méthodes de *clustering* appliquées aux phases transitoires en contexte univarié. La première est l'algorithme du SOM (*Self-Organizing Map*) relationnel qui est basé sur les dissimilarités. Nous introduisons une mesure de dissimilarité entre les phases transitoires. La deuxième méthode est l'algorithme du SOM numérique en représentant les phases par des vecteurs de caractéristiques.

Le chapitre 5, intitulé "*Clustering* et analyse de phases bivariées", étend la méthodologie de *clustering* des phases transitoires au contexte bivarié. Quelques méthodes de *clustering* de séries temporelles multivariées sont présentes dans la litérature. L'idée est d'utiliser les algorithmes SOM et une mesure de dissimilarité entre phases afin de construire des classes bivariées contenant des couples de phases ayant la même allure. Dans ce chapitre, nous développons deux outils : l'un permet de

rechercher des phases similaires à une phase donnée et l'autre de détecter des anomalies.

Le chapitre 6, intitulé "Analyse statistique de séquences labellisées", montre comment transformer les vols en séquences labellisées à partir des résultats du *clustering* bivarié. En utilisant l'algorithme SOM relationnel avec une dissimilarité construite à partir de la méthode *Optimal Matching*, nous classons les séquences labellisées sur une carte de Kohonen. Ensuite un critère est proposé pour identifier les vols "volages" ou atypiques.

La thèse finit par les conclusions et perspectives.

Cette thèse a donné lieu à plusieurs publications dans des conférences internationales et des conférences métiers. En voici la liste :

- *Comparison of three algorithms for parametric change-point detection*, ESANN, 2016 (Faure, Bardet, Olteanu, & Lacaille, 2016)
- *Indexation of Bench Test and Flight Data*, PHM Society, 2016 (Faure, Lacaille, Bardet, & Olteanu, 2016)
- *Using Self-Organizing Maps for Clustering and Labelling Aircraft Engine Data Phases*, WSOM, 2017 (Faure, Bardet, Olteanu, & Lacaille, 2017)
- *Design Aircraft Engine Bivariate Data Phases using Change-Point Detection Method and Self-Organizing Maps*, ITISE, 2017

2 *Safran Aircraft Engines* et le *Prognostic Health Monitoring*

Ce chapitre présente le processus de maintenance des moteurs d'avion, définit ce qu'est le *Prognostic Health Monitoring* en aéronautique ainsi que les différentes méthodes mises en œuvre chez *Safran Aircraft Engines*, et présente les données utilisées pour cette thèse.

2.1 Maintenance des moteurs d'avion

La maintenance des moteurs d'avion est primordiale. Elle correspond le plus souvent au remplacement préventif de pièces de moteurs d'avion. Chaque pièce est inspectée régulièrement. Il est possible d'effectuer un changement de pièce lors d'une inspection visuelle préventive dont le but est de repérer ou d'identifier des pièces endommagées ou partiellement usées.

Pour aider les équipes de maintenance et renforcer la surveillance des moteurs, de nombreux capteurs sont embarqués dans l'avion et permettent de calculer des indicateurs dont le comportement en cas d'anomalie alerte les ingénieurs experts. Par exemple, une augmentation de la consommation de carburant en croisière est un indicateur d'usure. De même la température de l'air à la sortie de la turbine haute pression (appelée *Exhausted Gas Temperature*, EGT) augmente avec l'usure du moteur. Plus précisément une diminution de la marge EGT, c'est-à-dire de la différence entre la limite supérieure de l'EGT qu'on ne doit pas dépasser et l'EGT mesurée, signale un phénomène d'usure.

Les indicateurs peuvent également notifier la présence de saletés dans les moteurs. Dans ce cas, on pratique un lavage (*water wash*) en propulsant une grande quantité d'eau à l'entrée du moteur qui tourne sans allumage. Ce processus de prévention est également une opération de maintenance.

Dès qu'une anomalie est constatée, une opération de maintenance est effectuée. Ces opérations sont très coûteuses, parfois longues et peuvent aller jusqu'à la dépose du moteur. Mais en tout état de cause, cette maintenance est indispensable autant du point de vue de la sécurité que pour éviter les coûts entraînés par des retards ou des immobilisations non programmées de l'appareil. D'où l'importance du développement de méthodes précises de détection d'anomalies, qui déclencheront les interventions.

2.2 *Prognostic Health Monitoring*

On comprend donc aisément que la maîtrise des opérations de maintenance est cruciale pour les compagnies aériennes. Afin de les prévenir, *Safran Aircraft Engines* développe des méthodes de surveillance, appelées *Prognostic Health Monitoring* (PHM). Le but est de détecter très tôt des anomalies qui pourraient entraîner plus tard des opérations de maintenance de grande ampleur ou même des pannes. Il s'agit aussi d'éviter de passer du temps à inspecter une partie non concernée du moteur, voire à le déposer entièrement pour finalement constater que cela était inutile et néanmoins très coûteux. Le *Prognostic Health Monitoring* fournit une aide au diagnostic de l'état d'un moteur d'avion, de façon préventive et la plus exacte possible, pour minimiser, optimiser et planifier les opérations de maintenance.

Le rôle des experts métier est donc essentiel et repose sur leur connaissance fine du fonctionnement des moteurs et sur leur expérience irremplaçable. Si un évènement opérationnel s'est produit, le constructeur du moteur peut décider d'envoyer sur place des experts en diagnostic des moteurs ou de faire rapatrier les pièces défectueuses pour les analyser et identifier l'origine de l'évènement qui a engendré une opération de maintenance. Les experts motoristes doivent aider les compagnies aériennes à identifier l'origine d'une anomalie en s'appuyant sur leur savoir-faire métier et leur retour d'expérience (RetEx). Le rôle du PHM est de les assister dans leur tâche en développant des méthodes d'Apprentissage Statistique (*Machine Learning*) comme outils d'aide à la décision.

2.3 Les techniques actuellement mises en œuvre à *Safran Aircraft Engines*

Les capteurs embarqués mesurent des informations tout au long du vol (du *taxi-out* au *taxi-in*) : des données sur le contexte extérieur (température extérieure, pression, altitude,...) et des données liées au moteur (vitesse de rotation haute pression, vitesse de rotation basse pression, EGT, température d'huile, position de la manette,...). La Figure 2 représente une coupe d'un moteur et les emplacements de quelques capteurs qui relèvent la température à la sortie de la turbine haute pression (EGT), la vitesse de rotation haute pression (N2), la vitesse de rotation basse pression (N1) et le débit carburant (FMV, *Fuel Metering Valve*).

Au cours d'un vol, des messages sont transmis au sol par radio via un système de communication codé, par exemple le système ACARS (*Aircraft Communications Addressing and Reporting System*). Il s'agit d'un résumé des paramètres de vol relevés à des moments importants (tels que le décollage, la croisière, l'atterrissage, etc...). Mais l'ensemble des mesures est enregistré à bord tout au long du vol et ce sont ces séries temporelles que nous étudions dans la suite de notre travail.

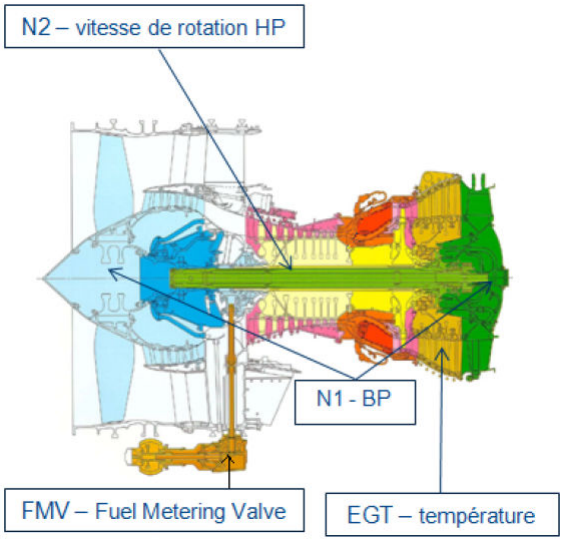


Figure 2: Coupe d'un moteur avec quelques exemples de capteurs.

2.3.1 Savoir-faire métier

Grâce au savoir des experts, on peut définir des seuils permettant de décider si les mesures sont normales ou anormales. On peut ainsi mettre en place un processus de détection automatique des anomalies potentielles. Ces seuils sont choisis de façon à minimiser les risques de laisser passer une anomalie sans la détecter. Cette technique est essentiellement univariée puisqu'on étudie chaque mesure séparément. De manière générale, ces indicateurs sont calculés à bord puis retransmis au sol. Il peut s'agir de mesures brutes issues de capteur ou alors des mesures normalisées telles que la marge EGT.

Lorsqu'une anomalie est détectée (par comparaison à des critères de performance définis), un opérateur métier donne son avis en tant qu'expert afin d'expliquer l'origine de cette anomalie.

Des scénarios-type d'évolution des paramètres sont développés à *Safran Aircraft Engines* pour aider les opérateurs métier à analyser les données observées.

A chaque scénario, les experts associent une classe d'anomalie et identifient ainsi l'origine de l'anomalie potentielle. Mais il peut arriver qu'une anomalie corresponde à deux scénarios différents ou même qu'aucun scénario répertorié ne corresponde. L'examen des scénarios n'est donc pas suffisant. C'est pourquoi il est nécessaire d'utiliser d'autres méthodes d'analyse fondées sur l'ensemble des données mesurées.

2.3.2 Les algorithmes de détection d'anomalies développés chez *Safran Aircraft Engines*

De nombreuses méthodes de détection d'anomalies durant les vols et les essais ont été développées chez *Safran Aircraft Engines* dans le cadre du *Prognostic Health-Monitoring*.

Certaines méthodes se fondent entièrement ou partiellement sur le savoir-faire métier des experts. Par exemple dans (Lacaille & Gerez, 2011), Lacaille *et al.* utilisent un algorithme de détection d'anomalies très simple. Cet algorithme travaille en temps presque réel avec un diagnostic toutes les dix secondes sur des bancs d'essais. Les situations aberrantes possibles sont répertoriées et permettent de définir des seuils. L'algorithme identifie rapidement des phénomènes complexes et inusuels en utilisant une distance entre les observations et ces seuils. Mais il nécessite une étape d'analyse pour comprendre la raison de la détection. Une autre version bien plus légère, se focalisant sur les couples de capteurs est opérationnelle sur les bancs d'essais des moteurs en développement. Dans (Rabenoro et al., 2014), Rabenoro *et al.* cherchent à démultiplier le savoir-faire métier en produisant un très grand nombre d'indicateurs qu'on étudie simultanément pour repérer les comportements atypiques. Cet algorithme a été utilisé pour identifier la pertinence de rapports de vols au format ACARS envoyés par les avions. Une première version utilise un ensemble d'arbres de décision et réalise une identification correcte mais difficilement interprétable. Une autre version dérivée de la première utilise un réseau bayésien un peu moins efficace, mais qui propose une solution de classification clairement interprétable et donc bien plus facile à exploiter (voir Figure 3).

	pas d'alarme	alarme	total
sain	996	277	1273
anomalie	81	262	343

	pas d'alarme	alarme	total
sain	1156	117	1273
anomalie	195	148	343

Figure 3: Matrices de confusion à partir des 20 meilleurs indicateurs en utilisant les Random Forests (à droite) puis un réseau bayésien (à gauche).

Dans d'autres cas, la détection d'anomalies passe par la sélection de variables. Dans (Seichepine, Ricordeau, & Lacaille, 2011), Seichepine *et al.* définissent un algorithme qui traite des données de vols issues des enregistreurs et déchargées au sol après chaque vol. Cet algorithme établit des tubes de confiance autour de courbes transitoires sélectionnées aléatoirement. Ces tubes sont issus de la modélisation par des fonctions oracles représentant la physique du vol combinée à une régression. Ils permettent d'identifier des mesures mal conditionnées et ajoutent une information de qualité à chaque courbe. Dans (Rabenoro et al., 2014) après avoir défini un très grand nombre d'indicateurs, Rabenoro a choisi la méthode de mRMR (*Minimum Redundancy Maximum Relevance*) pour sélectionner les plus représentatifs à partir de la variable N1.

Une autre façon de détecter des anomalies est d'utiliser les cartes Auto-Organisées de Kohonen (*Self-Organizing Maps*, SOM). Par exemple dans (Cottrell et al., 2009), Cottrell *et al.* projettent sur une carte bi-dimensionnelle les données de vols mesurées afin d'analyser la trajectoire de ces données vol après vol. Dans (Come, 2011), Côme *et al.* utilisent également cette méthode pour visualiser l'évolution des données "saines" corrigées des données exogènes. A partir des trajectoires observées, ils peuvent ainsi prédire l'évolution d'un vol selon le début de sa trajectoire. Chaque vol étant classé sur la carte de Kohonen, il devient possible d'étudier la trajectoire de l'état d'un moteur et d'identifier une dérive vers une zone de la carte correspondant à des dégradations éventuelles. Dans (Bellas, Bouveyron, Cottrell, & Lacaille, 2014), les données sont classées selon les variables d'environnement et Bellas *et al.* calculent les données de comportement du moteur corrigées des données exogènes en utilisant un modèle de régression différent pour chaque classe d'environnement. Les données corrigées sont alors projetées sur une carte de Kohonen en utilisant l'algorithme SOM, et pour chaque classe de Kohonen on définit une région de confiance. Toute observation qui n'appartient pas à cette région de confiance est suspectée d'être une anomalie.

D'autres méthodes ont également été développées. Dans (Lacaille & Gerez, 2011), Lacaille *et al.* analysent les séries temporelles en utilisant deux algorithmes: le premier corrige les variables moteur de l'influence du contexte et le deuxième utilise une méthode de détection de ruptures *online* pour détecter les changements abrupts. Ensuite dans (Bellas, Bouveyron, Cottrell, & Lacaille, 2012), Bellas *et al.* utilisent la méthode HDDC (*High Dimensional Data Classification*) avec une étape d'ajustement afin de détecter des *outliers*. Dans (Bellas et al., 2013), ils implémentent un algorithme d'inférence *online* établi sur l'algorithme EM et une variante de l'Analyse en Composantes Principales pour réduire la dimension des données.

Toutes ces méthodes reposent sur des algorithmes d'apprentissage à partir des données réelles et sont toujours combinées et validées par le savoir-faire métier. Notre approche pour la détection d'anomalies diffère des méthodes précédentes. Nous développons une méthode automatique pour détecter les phases transitoires qui n'ont jamais été explorées dans les données de vol puis nous les analysons.

2.4 Données de vol

Au début de notre travail de thèse, nous avons étudié plusieurs sources de données (différents types de moteurs, différentes compagnies, données de vol ou d'essai, ...). Nous avons choisi d'utiliser dans la suite du travail des données de vols, mesurées par des capteurs embarqués, sur un même type de moteur d'avion (CFM) et pour une même compagnie.

Les variables mesurées sont de trois types :

- Les variables moteur (endogènes) qui décrivent le contexte mécanique du moteur

Nom de variable	Description	Unité
Endogène		
N1	Vitesse de rotation basse pression	%
N2	Vitesse de rotation haute pression	%
VIB1	Vibrations selon N1	inch per sec
VIB2	Vibrations selon N2	inch per sec
T12	Température avant compression	°C
T3	Température avant combustion	°C
XM	Vitesse de l'avion	%
EGT	Température de l'air à la sortie	%
OIL TEMP	Température d'huile	%
Exogène		
ALT	Altitude	pied
TLA	Manette pilote	radian
T1	Température à l'entrée du fan	°C
P1	Pression entrée moteur	bar
Catégorielle		
Flight_Mod	Mode du vol	10 états

Table 1: Quelques noms de variables et leurs descriptions

comme les vitesses de rotation basse et haute pressions, les fréquences des vibrations, les températures du moteur avant et après combustion, la vitesse, l'EGT, la température de l'huile, etc...

- Les variables d'environnement (exogènes) qui décrivent le contexte extérieur du vol comme l'altitude, l'angle de la manette, la température extérieure, la pression extérieure, etc...

- Une variable catégorielle (Flight_Mod) à 10 modalités qui indique le mode (taxi, décollage, croisière...) du vol.

On présente quelques variables ainsi que leurs unités dans la Table 1.

Dans la suite de notre travail, on utilise la base de données de cette compagnie d'une taille de 164 Go. La base contient 549 vols sur différents itinéraires et les observations ont été recueillies à partir de 4 avions et 8 moteurs de même type. Chaque vol est défini comme une série temporelle multivariée de dimension 140 comprenant 139 variables moteur et environnement, dont beaucoup sont redondantes, et la variable catégorielle Flight_Mod. La durée des vols étudiés a un minimum de 1h, un maximum de 7h30 et une moyenne de 2h50.

La première étape a consisté à préparer la base de données pour pouvoir l'utiliser : une même variable était représentée par différents identifiants, la fréquence d'acquisition des données n'était pas la même d'un vol à un autre, ni d'une variable

à une autre, il y avait des données manquantes et quelques valeurs aberrantes. Nous avons retenu pour chaque variable un seul identifiant, échantillonné toutes les séries à la même fréquence (8 Hz), estimé les valeurs manquantes par lissage et remplacé les valeurs aberrantes selon les indications des experts. La Figure 4 représente 11 de ces variables et illustre leur diversité. Leur signification est présentée dans la Table 1.

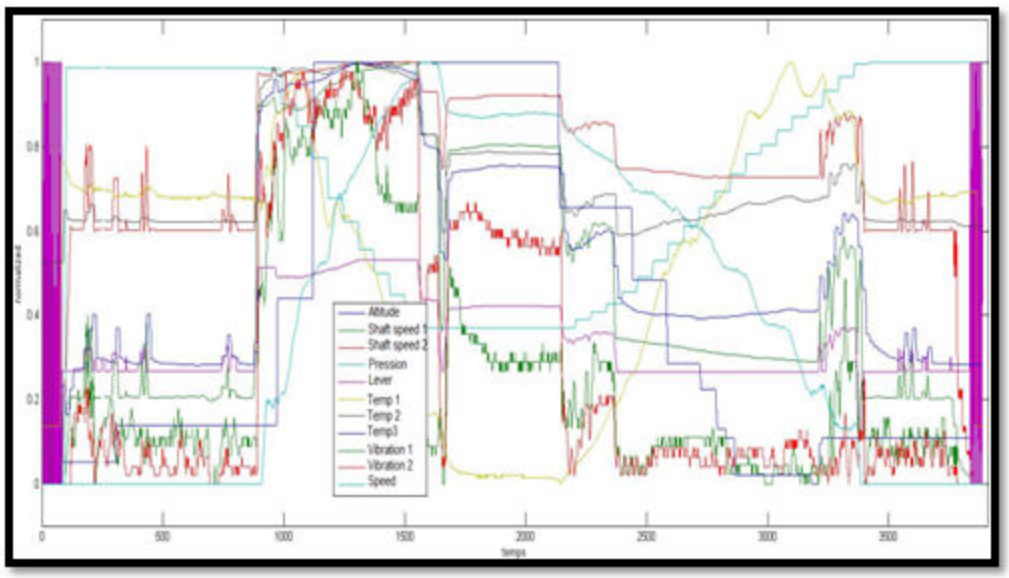


Figure 4: Représentation de 11 variables mesurées pendant un vol de 2h. L'unité de temps est la seconde. Les données ont été normalisées de façon à être comprises entre 0 et 1.

Quant à la variable Flight_Mod, elle a 10 modalités:

Nom	Description	Code
Pré-vol	Instant avant le vol	0
Démarrage moteur	Période de démarrage moteur	1
Taxi-out	Roulage de l'avion jusqu'à la piste de décollage	2
Décollage	Montée de l'avion	3
Climb	Phase de transition entre le décollage et la croisière	4
Croisière	Avion stable en altitude	5
Descente	Période de descente	6
Approche	Atterrissage	7
Landing roll	Roulage de l'avion sur la piste	8
Taxi-in	Entrée de l'avion dans l'aérogare	9

Table 2: Liste des 10 niveaux de la variable Flight_Mod accompagnés de leur description et de leur code.

Nous avons d'abord étudié les séries temporelles correspondant à la variable N1 (vitesse de rotation basse pression exprimée en % du seuil maximum) et à la variable T3 (température avant la combustion exprimée en degrés). La variable N1 est commandée par le pilote qui augmente ou diminue l'entrée de carburant au moyen d'un levier.

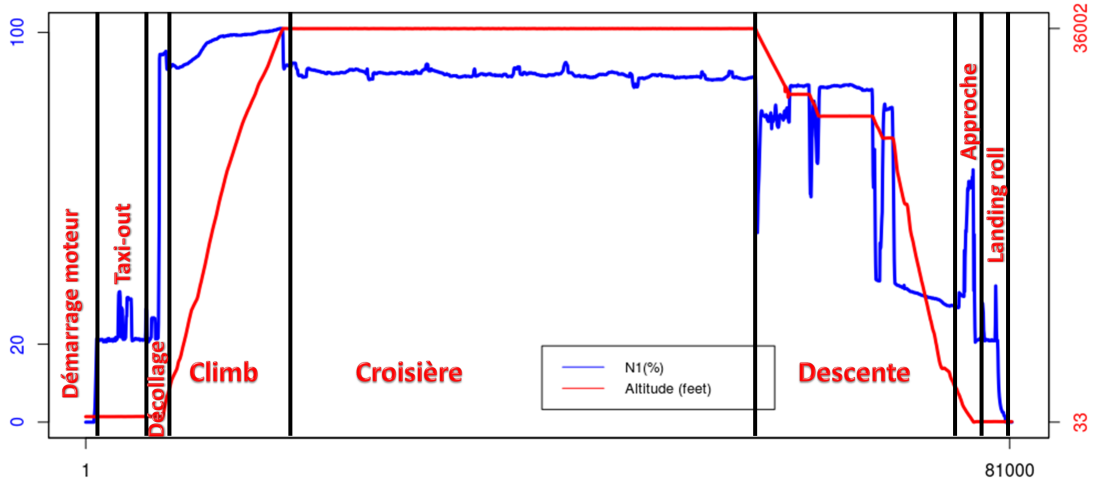


Figure 5: Représentation de N1 (vitesse de rotation et proportionnelle à la poussée du moteur) en bleu et de ALT (altitude barométrique issue des mesures de pression extérieures) en rouge durant un vol complet selon les 8 modes de vol associées (les 3 autres modes sont très courts c'est pourquoi ils ne sont pas représentés).

On représente sur la Figure 5 les variables N1 et ALT selon les modes de vol. Leur signification est présentée dans la Table 2. Chaque mode de vol est associé à un comportement caractéristique de la variable N1: pendant les modes de *taxi-out* et *taxi-in*, N1 reste en moyenne égale à 20%. Le mode de décollage est décrit par une augmentation de la vitesse N1 qui passe de 20% à 90 %. Le mode du *climb* est défini par une brève augmentation suivie par une petite diminution. Le mode de croisière est assez stable (environ 75 %). Les modes les plus difficiles à analyser sont les modes descente, approche et *landing roll* car ils sont très proches.

La Table 3 présente quelques statistiques élémentaires des variables N1 et T3.

	N1	T3
Moyenne des minima	0%	0°C
Ecart-type des minima	±0%	0°C
Moyenne des maxima	99%	533°C
Ecart-type des maxima	±1%	12°C
Moyenne des médianes	76%	377°C
Ecart-type des médianes	±21%	66°C

Table 3: Les moyennes et les écart-types sont calculés sur les 549 vols.

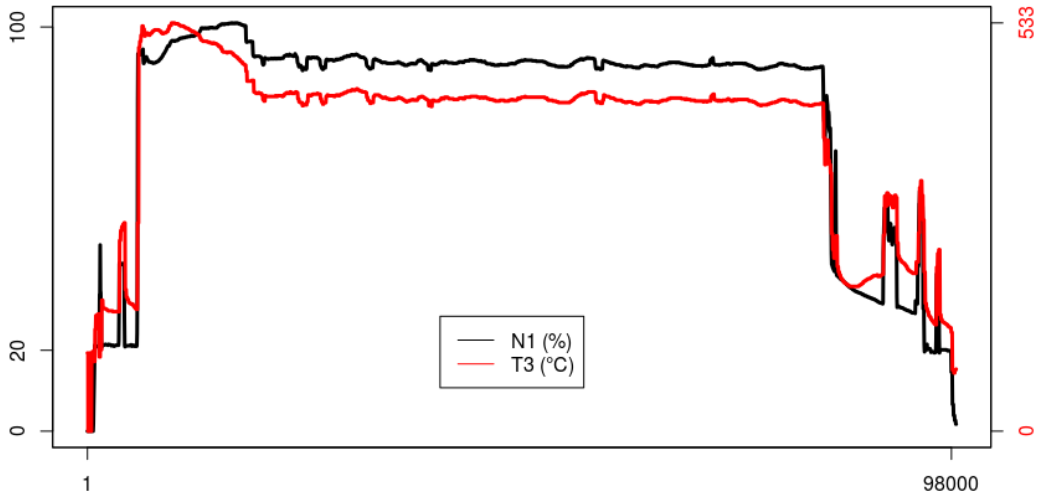


Figure 6: Représentation de N1 (en noir) et de T3 (en rouge) durant un vol complet

Ces deux variables ont la même allure et sont modélisées, dans ce qui suit, par des fonctions linéaires par morceaux. La méthodologie proposée dans ce manuscrit est assez générale pour prendre en compte d'autres approximations, mais pour ces deux variables (N1 et T3) le modèle linéaire par morceaux est suffisant et illustre bien nos propos.

Bien que nous ayons choisi de ne retenir dans un premier temps que les variables N1 et T3, les méthodes que nous présentons dans la suite peuvent être facilement adaptées aux autres variables.

La première partie de notre travail consiste à détecter des ruptures dans les séries temporelles observées, pour séparer les phases stables des phases transitoires. Le but est de repérer d'éventuels comportements "anormaux" dans ces phases transitoires, qu'il s'agit de classer et d'identifier. Dans un premier temps, on ne considère que des séries temporelles univariées. On utilisera comme exemple d'application

la variable "vitesse de rotation basse pression", c'est-à-dire la variable N1, qui a l'avantage d'être assez facile à interpréter pendant les différents modes de vol. Ces derniers sont toujours fournis et ils permettront de valider les algorithmes lors de la classification.

3 Recherche des phases transitoires

Dans ce chapitre, nous définissons ce qu'est une phase transitoire dans le contexte du *Prognostic Health Monitoring* et nous proposons une méthodologie de détection de ruptures *offline* et *online* afin de les identifier. Ces méthodes sont ensuite appliquées aux données réelles introduites précédemment.

Définir une phase transitoire est difficile pour différentes raisons, la définition même variant selon les chercheurs. Premièrement, cette définition est fortement liée au choix de l'algorithme de sa détection. Deuxièmement, la présence de bruit peut s'avérer être une contrainte dans la recherche de petites phases transitoires. Ensuite les longues phases transitoires peuvent être facilement détectées alors que les plus courtes, qui sont tout aussi intéressantes, passent inaperçues. Pour toutes ces raisons, il n'existe pas de méthode de détection standard des phases transitoires.

3.1 Etat de l'art

Peu d'études sur la détection de phases transitoires dans les séries temporelles univariées ont été réalisées. Dans (Walwer, Calais, & Ghil, 2016), Walwer *et al.* parlent de déformation transitoire dans un signal d'un point de vue géophysique. Pour détecter ces transitoires, une méthode d'analyse de système dynamique (Multichannel Singular Spectrum Analysis) fondée sur les corrélations temporelle et spatiale des caractéristiques géophysiques est proposée. Dans un contexte d'étude géographique (Ji & Herring, 2011), Ji *et al.* définissent quant à eux des signaux transitoires comme ceux qu'on peut différencier d'un mouvement dit "stable" ou d'un modèle de bruit stochastique. De plus, selon lui, un transitoire doit être cohérent dans l'espace, c'est-à-dire distinguer les signaux liés à un effet de tectonique de ceux qui ne le sont pas. Il propose de les détecter avec une approche de filtrage spatiotemporel en combinant l'estimation de filtre de Kalman et l'analyse en composantes principales (ACP). Dans un contexte totalement général (Belanger, 2013), Bélanger étudie les phénomènes transitoires dans les séries temporelles. Mais jusqu'à présent il n'existait pas de définition des phases transitoires dans le contexte du *Health Monitoring*, bien que ce soit un sujet important pour les experts.

A la suite de Bélanger (Belanger, 2013), nous proposons d'utiliser des méthodes d'apprentissage statistique (*Machine Learning*) pour définir mathématiquement des phénomènes tels que des changements de comportement, des différences de régime, des écarts ou des variations d'intensité (dans notre cas, il s'agit de changement en niveau). Bien que ce soit évident, il est important d'indiquer que la détection d'un événement localisé dans le temps, implique l'identification d'une "nouveauté". Que cet évènement soit soudain et visible à l'œil nu, ou qu'il apparaisse lentement et soit peu visible, dans tous les cas il peut être vu comme une rupture dans la distribution des données.

Une phase transitoire est par définition une phase **non stable**. Reste à définir la stabilité. Pour cela, dans le cas d'une série linéaire par morceaux, on peut calculer la série différenciée (différence entre la valeur à l'instant $t+1$ et la valeur à l'instant t) et considérer que la phase est stable tant que cette série différenciée est approximativement centrée.

Le découpage du signal en succession de phases transitoires et stabilisées est réalisé par une méthode de détection de ruptures. Deux choix s'offrent à nous: cette détection peut être réalisée *offline* ou *online*.

3.2 Détection de points de rupture *offline*

L'un des avantages de la détection *offline* est de pouvoir tenir compte de la série temporelle toute entière lors de la segmentation. Dans un premier temps, nous décrivons une méthode générale de détection de ruptures en univarié, que nous déclinons en plusieurs variantes.

3.2.1 Etat de l'art

Les algorithmes de détection de points de rupture *offline* considèrent d'abord la série temporelle toute entière (Basseville & Nikiforov, 1993), (Darkhovski, 1994), puis font un retour en arrière dans le temps pour identifier les points de rupture. Ces méthodes sont très abondantes dans la littérature et souvent utilisées en sciences de l'ingénieur, biologie, histoire, médecine, etc... Plusieurs approches existent.

En supposant que les données suivent une certaine distribution statistique appartenant à une famille paramétrique donnée, il peut exister un point ou plusieurs dans la série à partir desquels les paramètres changent. Le but est de détecter ces points et éventuellement d'estimer les paramètres avant et après le changement. Par exemple, dans (Picard, 1985), Picard propose une méthode de détection de changement dans la moyenne en utilisant un critère basé sur la vraisemblance et souligne les "effets de bord" (lorsqu'il y a un changement au début ou à la fin de la série, mais trop peu de données pour avoir des estimations robustes).

Une approche courante est de partitionner la série en plusieurs segments homogènes. Cette approche est adaptée aux méthodes *offline*, étant donné qu'il est généralement nécessaire d'analyser la série toute entière pour définir les segments. Il existe plusieurs manières d'aborder le problème car on peut modéliser la série de différentes façons :

- Modèle de régression : Le principe est de supposer que la série temporelle suit un modèle de régression avec différents paramètres pour chaque segment. Dans (Guthery, 1974), Guthery utilise la méthode de *k-segmentation* avec un critère

de maximum de vraisemblance. Un cas particulier du modèle de régression est le modèle linéaire. Takeuchi *et al.* (Takeuchi & Yamanishi, 2006) utilisent un modèle autorégressif et la log-vraisemblance pour détecter les points de rupture. L'ordre du modèle est donné par le critère SIC (Schwarz Information Criterion).

- Modèle d'ondelettes : La série étant décomposée en ondelettes, dans (Ogden & Parzen, 1996), Odgen *et al.* identifient les points de rupture à partir des coefficients. Dans (Killick, Eckley, & Jonathan, 2013), Killick *et al.* construit un test basé les coefficients d'ondelette pour détecter des changements de "structures de covariance" dans les séries temporelles.

Une fois choisie la méthode de détection d'une rupture, on adopte différentes stratégies de recherche :

- Le *Top-down* : Cette technique procède par dichotomie. La série est divisée en deux portions au premier point de rupture trouvé. On réitère dans les portions gauche et droite de la série. Cette méthode est très coûteuse en temps de calcul (Lowe, 1987). Cette approche est utilisée pour le monitoring d'entreprise par Krishnamurthy *et al.* dans (Krishnamurthy, 2012)
- Le *Bottom-up* : On commence par diviser la série en plusieurs sections (les plus petites et les plus homogènes possibles). On combine ensuite les sections successives similaires et on arrête de les combiner lorsque les sections ne sont plus similaires. Il s'agit également d'une méthode très coûteuse en temps de calculs (Berman, 1978).
- La programmation dynamique (qu'on explicitera plus tard) (Jackson, 2005), (Killick R. & Eckley, 2012).

3.2.2 Méthodologie

Nous présentons dans la suite la méthodologie générale pour la détection de ruptures dans un cadre *offline*.

Soit $\mathbb{Y} = (Y_1, \ldots, Y_N)$ une suite de variables aléatoires à valeurs dans $\mathbb{R}$. Soit $\mathbb{X} = (X_1, \ldots, X_N)$ une suite de vecteurs aléatoire à valeurs dans $\mathbb{R}^p$. Pour tout $t \in \{1, \ldots, N\}$, on suppose que Y_t peut être écrit comme une fonction de X_t:

$$Y_t = g_\theta(X_t, \varepsilon_t), \tag{1}$$

où g_θ est une fonction déterministe connue de $\mathbb{R}^{p+1}$ dans $\mathbb{R}$ de paramètre $\theta \in \Theta$, $\Theta \subset \mathbb{R}^q$ et (ε_t) est un bruit blanc non observé, centré de variance σ^2 et indépendant de X_t.

Dans le cas particulier où le bruit est additif, l'équation (1) s'écrit:

$$Y_t = g_\theta(X_t) + \varepsilon_t. \tag{2}$$

On suppose qu'il existe un nombre inconnu K^* de ruptures paramétriques dans la relation entre (Y_t) et (X_t), *i.e.* qu'il existe une suite d'entiers inconnus $(\tau_1^*, \dots, \tau_{K^*}^*)$ tels que $1 < \tau_1^* < \tau_2^* < \cdots < \tau_{K^*}^* < N$ et $\theta_j^* \neq \theta_{j+1}^*, \forall j = 0, \cdots, \tau_{K^*}^* - 1$.

Le nombre de ruptures K^* est *a priori* et en général inconnu. Par convention, on pose $\tau_0^* = 0$ et $\tau_{K^*+1}^* = N$. Il existe donc $K^* + 1$ paramètres inconnus $(\theta_0^*, \theta_1^*, \dots, \theta_{K^*}^*)$ dans l'ensemble Θ satisfaisant:

$$Y_t = g_{\theta_i^*}(X_t, \varepsilon_t) \quad \text{pour} \quad t \in \left\{\tau_i^*+1, \tau_i^*+2, \dots, \tau_{i+1}^*\right\}, \quad \text{pour} \quad i = 0, \dots, K^*. \tag{3}$$

Le but est d'estimer K^*, $(\tau_1^*, \dots, \tau_{K^*}^*)$, $(\theta_0^*, \theta_1^*, \dots, \theta_{K^*}^*)$ et σ^2, qui sont les paramètres du modèle. L'estimation de ces paramètres peut être très coûteuse et en particulier une recherche exhaustive des ruptures a une complexité exponentielle. De nombreux auteurs ont proposé des méthodes pratiques de calcul de ces estimateurs. Un critère est, par exemple, de minimiser la somme des carrés des résidus, c'est-à-dire la fonction d'erreur quadratique comme décrit dans (Bai & Perron, 1998). Etant donné que le nombre de points de rupture est inconnu, (Lavielle & Moulines, 2000) proposent d'ajouter un terme de pénalité à la fonction d'erreur. En effet il est aisé de comprendre que plus le nombre de ruptures est grand, plus l'erreur est petite, sans que ce soit pertinent. C'est pourquoi le terme de pénalité est nécessaire.

La méthode de détection de ruptures *offline* consiste donc à minimiser en $(K, (\tau_i), (\theta_i), \sigma^2)$ la fonction coût:

$$\mathcal{F}(Y, X, (\theta_i), \sigma^2, K, (\tau_i)) = \sum_{i=0}^{K} \sum_{t=\tau_i+1}^{\tau_{i+1}} C\big(Y_t, X_t, \theta_i, \sigma^2\big) + \beta\, f(K)\,, \tag{4}$$

où la fonction d'erreur C peut être définie de différentes manières (par exemple l'erreur quadratique, la -log-vraisemblance, etc...). Le terme $\beta f(K)$ est le terme de pénalité qui évite d'obtenir un nombre excessif de ruptures. Ce terme est souvent linéaire (par exemple dans les critères AIC (Akaike, 1973) ou BIC (Schwarz, 1978)) par rapport au nombre de paramètres et puisque dans le modèle (1)-(3) le nombre de paramètres est proportionnel à K, nous posons $\beta f(K) = \beta K$.

Pour $1 \leq u < v \leq N$, soit $\hat{\theta}_{u,v} = \arg\min_{\theta\in\Theta} \sum_{t=u+1}^{v} C\big(Y_t, X_t, \theta, \sigma^2\big)$ l'estimateur de θ calculé sur l'intervalle $\{u+1, ..., v\}$. Alors le problème peut être réécrit comme suit:

$$\big(\hat{K}, \hat{\tau}_1, \dots, \hat{\tau}_{\hat{K}}, \hat{\sigma}^2\big) = \underset{K, 1<\tau_1<\tau_2<\cdots<\tau_K<N, \sigma^2}{\arg\min} \left\{ \sum_{i=0}^{K} \sum_{t=\tau_i+1}^{\tau_{i+1}} C\big(Y_t, X_t, \hat{\theta}_{\tau_i,\tau_{i+1}}, \sigma^2\big) + \beta\, K \right\} \tag{5}$$

Le problème est simplifié lorsqu'on sait que la série temporelle ne présente qu'une rupture T. On calcule T en minimisant les fonctions d'erreur restreintes aux intervalles $[1, \tau]$ et $[\tau + 1, N]$:

$$T = \underset{\tau=1,\cdots,N}{\arg\min} \sum_{t=1}^{\tau} C(Y_t, X_t, \hat{\theta}_{1,\tau}, \sigma^2) + \sum_{t=\tau+1}^{N} C(Y_t, X_t, \hat{\theta}_{\tau+1,N}, \sigma^2) \tag{6}$$

Mais dans le cas où le nombre de ruptures est inconnu, il faut utiliser des approches algorithmiques basées sur la programmation dynamique.

Plusieurs approches basées sur la programmation dynamique permettent de minimiser la fonction coût. Trois de ces algorithmes sont implémentés ici: la partition optimale (Jackson, 2005), le *pruned exact linear time method* [PELT] (Killick R. & Eckley, 2012) et la méthode de l'heuristique de pente [SEP] (Arlot & Massart, 2009).

Dans la suite sont présentés rapidement les trois algorithmes implémentés.

3.2.3 Partition optimale

L'algorithme de la partition optimale (PO) (ou *Optimal Partition* [OP]) a été introduit par (Jackson, 2005). Pour éviter la recherche exhaustive du minimum (5), il utilise la programmation dynamique pour obtenir une complexité quadratique.
Avec les notations précédentes, un point de rupture apparaît entre deux instants u et v s'il existe l avec $u < l < v$ tel que

$$\sum_{t=u+1}^{l} C(Y_t, X_t, \hat{\theta}_{u,l}, \sigma^2) + \sum_{t=l+1}^{v} C(Y_t, X_t, \hat{\theta}_{l,v}, \sigma^2) + \beta < \sum_{t=u+1}^{v} C(Y_t, X_t, \hat{\theta}_{u,v}, \sigma^2) \tag{7}$$

En utilisant le critère (7), cet algorithme scanne de gauche à droite la série et calcule itérativement la valeur minimale $Z(u)$ de la fonction coût $\mathcal{F}(Y_t, X_t, (\hat{\theta}_i), \sigma^2, K, (\tau_i))$ sur l'intervalle $[1, u]$, $\forall u \leq N$.

Le minimum $Z(u)$ se calcule par récurrence comme ci-dessous:

$$Z(u) = \min_{K_u, 1<\tau_1<\tau_2<\cdots<\tau_{K_u}<u, \sigma^2} \sum_{i=0}^{K_u} \left[\sum_{t=\tau_i+1}^{\tau_{i+1}} C(Y_t, X_t, \hat{\theta}_{\tau_i,\tau_{i+1}}, \sigma^2) + \beta \right] \tag{8}$$

$$Z(u) = \min_{u'=1,\dots,(u-1)} \min_{K_{u'},1<\tau_1<\tau_2<\cdots<\tau_{K_{u'}-1}+1=u',\sigma^2} \sum_{i=0}^{K_{u'}-1} \left[\sum_{t=\tau_i+1}^{\tau_{i+1}} C(Y_t,X_t,\hat{\theta}_{\tau_i,\tau_{i+1}},\sigma^2)+\beta \right] \quad (9)$$
$$+ \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta$$

$$Z(u) = \min_{u'=1,\dots,(u-1)} \left\{ Z(u') + \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta \right\} \quad (10)$$

On remarque que l'argument u' pour lequel le minimum $Z(u)$ est atteint correspond à la dernière rupture avant u, on le note u^*. On définit un vecteur R de dimension N tel que $\forall u = 1, ..., N$, $R(u) = u^*$ l'instant de la dernière rupture avant u.

L'algorithme PO peut être résumé comme ci-dessous dans la *Procédure 1*.

Procédure 1 : Partition Optimale [PO]

- Initialiser $Z(1) = -\beta$ et définir R de taille N comme un vecteur dont toutes les composantes sont égales à 1.
- Pour $u = 1, \cdots, N$:
  Calculer itérativement (de gauche à droite)
  $Z(u) = \min_{u'=1,\dots,(u-1)} \left\{ Z(u') + \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta \right\}$.
  Noter u^* l'instant de la dernière rupture avant u égal à $\arg\min_{u'=1,\dots,(u-1)} \left\{ Z(u') + \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta \right\}$ et $R(u) = u^*$.
- Calculer itérativement (de droite à gauche) les points de rupture:

$$\hat{\tau}_K = R(N) \ , \ \hat{\tau}_{K-1} = R(\hat{\tau}_K) \ , \ \cdots$$

3.2.4 *Pruned Exact Linear Time* (PELT)

Introduit par (Killick R. & Eckley, 2012), l'algorithme *Pruned Exact Linear Time* [PELT] est un algorithme de complexité linéaire grâce au principe d'élagage pour la recherche des instants de rupture, en prenant comme point de départ le même principe que la méthode PO. L'idée est d'enlever à chaque itération les instants ne pouvant pas être candidats comme prochains instants de rupture. Il s'agit d'un élagage donnant lieu à moins de calculs à chaque itération et réduisant par conséquent le temps de calcul global. L'élagage est justifié par la propriété suivante, prouvée dans (Killick R.

& Eckley, 2012) : soient trois instants $u < \ell < v$, si l'inégalité (11) :

$$\sum_{t=1}^{u} C(Y_t, X_t, \hat{\theta}_{1,u}, \sigma^2) + \sum_{t=u+1}^{\ell} C(Y_t, X_t, \hat{\theta}_{u,\ell}, \sigma^2) + \beta \geq \sum_{t=1}^{\ell} C(Y_t, X_t, \hat{\theta}_{1,\ell}, \sigma^2) \quad (11)$$

est vérifiée, alors en raison de (7), u ne peut pas être le dernier point de rupture avant ℓ (ni avant v). L'étape correspondant à u dans les itérations peut être supprimée. On note dans la suite P l'ensemble des instants de 1 à N qui n'ont pas été élagués. L'algorithme est décrit ci-dessous dans la *Procédure 2*.

Procédure 2 : Pruned Exact Linear Time [PELT]

- Initialiser $Z(1) = -\beta$, $P = \{1\}$ et définir R comme un vecteur dont toutes les composantes sont égales à 1.
- Pour $u = 1, ..., N$:
  Calculer itérativement (de gauche à droite)
  $Z(u) = \min_{u' \in P} \left\{ Z(u') + \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta \right\}$.
  Noter u^* l'instant de la dernière rupture avant u égal à $\arg\min_{u' \in P} \left\{ Z(u') + \sum_{t=u'+1}^{u} C(Y_t, X_t, \hat{\theta}_{u',u}, \sigma^2) + \beta \right\}$ et $R(u) = u^*$.
  Mettre à jour l'ensemble des possibles points de rupture P.
- Calculer itérativement (de droite à gauche) les points de rupture:

$$\hat{\tau}_K = R(N) \ , \ \hat{\tau}_{K-1} = R(\hat{\tau}_K) \ , \ \cdots$$

3.2.5 Heuristique de pente/*Slope Estimation Procedure* (SEP)

Dans les deux précédents algorithmes, le terme de pénalité est à définir au préalable. Le choix le plus classique est celui du BIC (Schwarz, 1978), car il permet en général d'obtenir des modèles parcimonieux et des estimateurs consistants. Cependant, cet *a priori* sur le terme de pénalité n'est pas toujours adapté aux données. Récemment, des modèles de calibration du terme de pénalité basés sur les données ont été introduits par plusieurs auteurs (Arlot & Massart, 2009), (Baudry J-P. & B., 2010) et utilisés (Bardet, Kengne, & Wintenberger, 2012). Nous avons choisi de suivre ici la méthodologie proposée dans (Bardet et al., 2012), l'heuristique de pente, où le terme de pénalité est calculé par une procédure d'estimation de pente (*Slope Estimation Procedure* [SEP]). On suppose que le nombre de points de rupture est majoré par une borne supérieure K_{max}. Cet entier K_{max} est défini *a priori* et détermine la complexité de l'algorithme. Pour toutes les valeurs de K comprise entre 1 et K_{max}, on détermine les minima de la fonction coût et on les projette sur un graphique en fonction de K.

On observe en général, comme l'a remarqué (Baudry J-P. & B., 2010), que

le coût est décroissant et varie linéairement en fonction de K lorsque K est "suffisamment grand" comme on le voit dans la Figure 7.

Bardet *et al.* proposent une méthode dans (Bardet et al., 2012) qui consiste à choisir comme terme de pénalité le double de la pente observée sur le graphique. L'algorithme est décrit dans la *Procédure 3*. On note $\mathbb{C}(K)$ le minimum de la fonction coût lorsque K est donné.

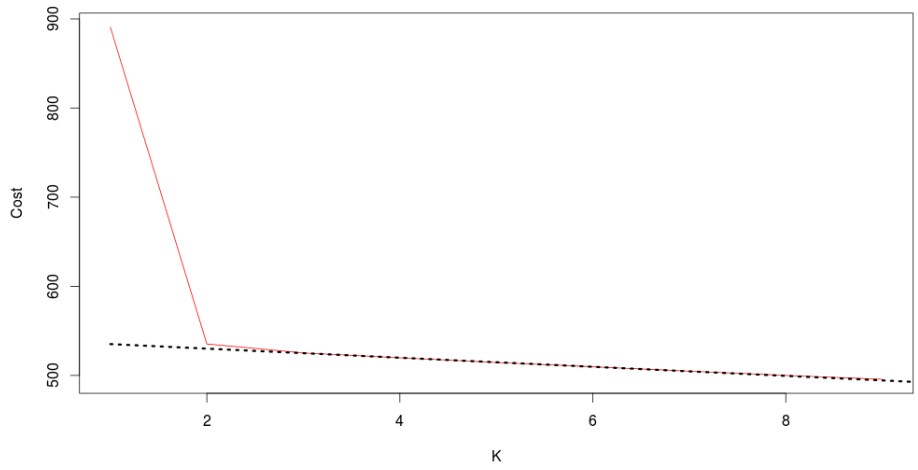


Figure 7: Comportement typique d'une fonction coût pour $1 \leq K \leq K_{max} = 9$. Plus le nombre de ruptures fixé est grand, plus l'erreur est petite. La deuxième courbe en pointillés représente la régression linéaire dont la pente sert à calculer le terme de pénalité.

Procédure 3: Slope Estimation Procedure

1. Calculer avec la programmation dynamique pour $K = 1, ..., K_{max}$:

$$\mathbb{C}(K) = \min_{1<\tau_1<...<\tau_K<N,\sigma^2} \sum_{i=0}^{K} \sum_{t=\tau_i+1}^{\tau_{i+1}} C(Y_t, X_t, \hat{\theta}_{\tau_i,\tau_{i+1}}, \sigma^2)$$

2. Dessiner le graphe $(K, \mathbb{C}(K))_{1\leq K\leq K_{max}}$
3. Estimer la pente γ par un modèle de régression linéaire pour K supérieur à un certain seuil
4. Définir le terme de pénalité $\beta = -2\gamma$

Déterminer ce seuil revient à détecter une rupture sur le graphe $(K, \mathbb{C}(K))_{1\leq K\leq K_{max}}$. Pour cela, il suffit de minimiser l'expression de l'équation (6). Une fois déterminée la pénalité par cette méthode, la minimisation du coût se fait en utilisant la programmation dynamique car les calculs ont déjà été réalisés dans l'étape 1 de la Procédure 3.

La méthode d'heuristique de pente a une complexité $O(K_{max}N^2)$ à com-

parer avec les complexités des méthodes PO ($O(N^2)$) et PELT ($O(N)$).

3.2.6 Applications sur données simulées

Les variables que nous étudions et qui sont liées au fonctionnement des turboréacteurs peuvent être approximées par des fonctions linéaires par morceaux. Cette approximation bien que grossière permet de tester les différentes méthodes. C'est pourquoi nous nous intéressons à des données simulées simplifiées et linéaires par morceaux pour lesquelles entre deux ruptures r et s, $Y_t = \theta_{r,s}^{(1)} + \theta_{r,s}^{(2)} t + \varepsilon_t$. Cela correspond à choisir dans le modèle défini par l'équation (1), pour $r < t \leq s$:

$$\begin{aligned} X_t &= (1, t) \\ \theta_{r,s} &= (\theta_{r,s}^{(1)}, \theta_{r,s}^{(2)}) \\ g_\theta(X_t, \varepsilon_t) &= \ < X_t, \theta_{r,s} > + \varepsilon_t \end{aligned} \tag{12}$$

où $< \cdot, \cdot >$ désigne le produit scalaire classique.

La fonction d'erreur $\hat{C}$ est choisie empiriquement selon la méthode du *Minimum Description Length* (MDL) (Rissanen, 1978), (Davis, Lee, & Rodriguez-Yam, 2006). Pour tout intervalle $[u, v]$, on a donc:

$$\hat{C}(Y_t, X_t, \hat{\theta}_{u,v}, \hat{\sigma}^2) = 3\ln(v-u) + (v-u)\log(2\pi\hat{\sigma}^2) \tag{13}$$

où $\hat{\sigma}^2$ est :

$$\hat{\sigma}^2 = \frac{1}{v-u} \sum_{t=u+1}^{v} \left(Y_t - \hat{\theta}_{u,v}^{(1)} - \hat{\theta}_{u,v}^{(2)} t\right)^2 \tag{14}$$

Initialement, nous avions considéré comme fonction d'erreur l'erreur quadratique de la régression linéaire. Cependant sur les tronçons centrés de la série temporelle, cette fonction d'erreur était nulle. Ainsi défini, l'algorithme renvoyait beaucoup trop de ruptures. Pour contourner ce problème, nous utilisons le critère du MDL pour estimer le nombre de ruptures, la position de ces ruptures et les paramètres du modèle comme dans (Davis & Yau, 2013).

Les trois algorithmes ont été implémentés avec le logiciel **R** sur Ubuntu 14.04.3 server, 23 processors Intel(R) Xeon(R) CPU X5675 3.07GHz, avec 6 coeurs chacun.

Dans l'implémentation, la minimisation de la fonction d'erreur dans l'équation (13) conduit à un phénomène de sur-apprentissage, à un découpage trop fin de l'intervalle, à la détection d'un grand nombre de ruptures parasites et à des estimations de σ^2 proches de 0. Dans ce cas, le terme $\log(2\pi\hat{\sigma}^2)$ tend vers $-\infty$ et pour résoudre ce problème numérique, on approxime $\log(2\pi\hat{\sigma}^2)$ par $\log(1 + 2\pi\hat{\sigma}^2)$. On

remplace donc $\hat{C}$ dans l'équation (13) par:

$$\hat{C}'(Y_t, X_t, \hat{\theta}_{u,v}, \hat{\sigma}^2) = 3\ln(v-u) + (v-u)\log(1+2\pi\hat{\sigma}^2) \tag{15}$$

Deux scénarios ont été considérés pour les simulations : (**A**) une série temporelle de taille $N = 300$ et trois ruptures aléatoires, et (**B**) une série temporelle de taille $N = 1000$ et six ruptures aléatoires. Pour tous les scénarios, les séries temporelles ont été générées 1000 fois et une condition a été rajoutée pour que les pentes de deux segments consécutifs ne soient pas identiques. Les coefficients $\theta^{(1)}$ et $\theta^{(2)}$ de chaque segment sont générées aléatoirement par une distribution uniforme. Les bruits sont quant à eux obtenues par une loi normale centrée. La variance dépend du niveau de bruit choisi pour la simulation.

Voici un exemple de simulation (Figure 8):

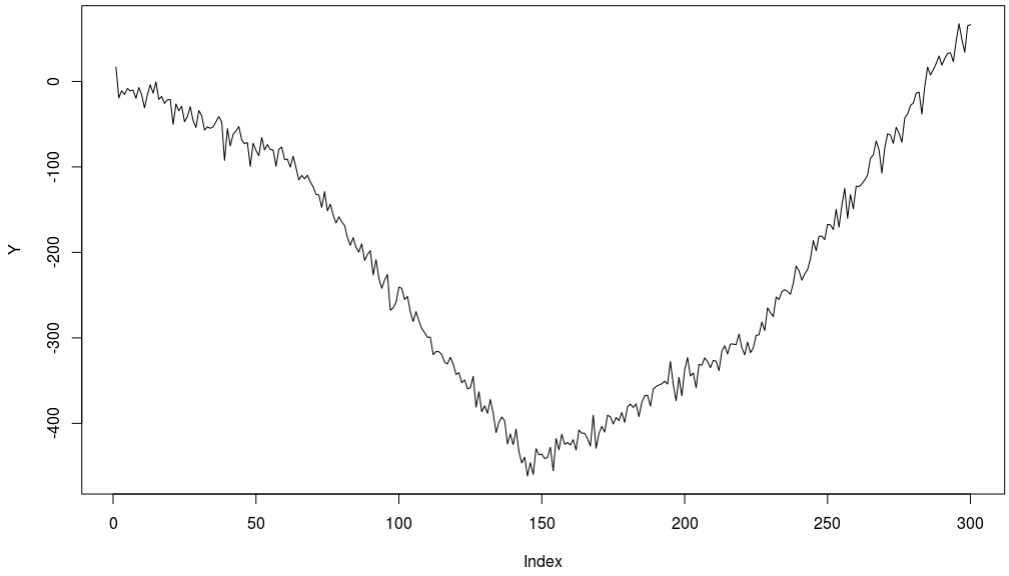


Figure 8: Un exemple de trajectoire simulée pour $N = 300$ où $\sigma = 10\%$ de la moyenne.

Afin de comparer les trois méthodes, nous avons défini trois critères de performance:

- Le nombre de ruptures bien détectées sur les 1000 séries.
- La précision, mesurée par la différence absolue entre les positions réelles et estimées des points de rupture, une fois détecté le bon nombre de ruptures.
- Le temps de calcul pour une simulation.

La pénalité du BIC a été utilisée pour les méthodes PO et PELT. L'arbre de la Figure 9 illustre l'ensemble des critères choisis.

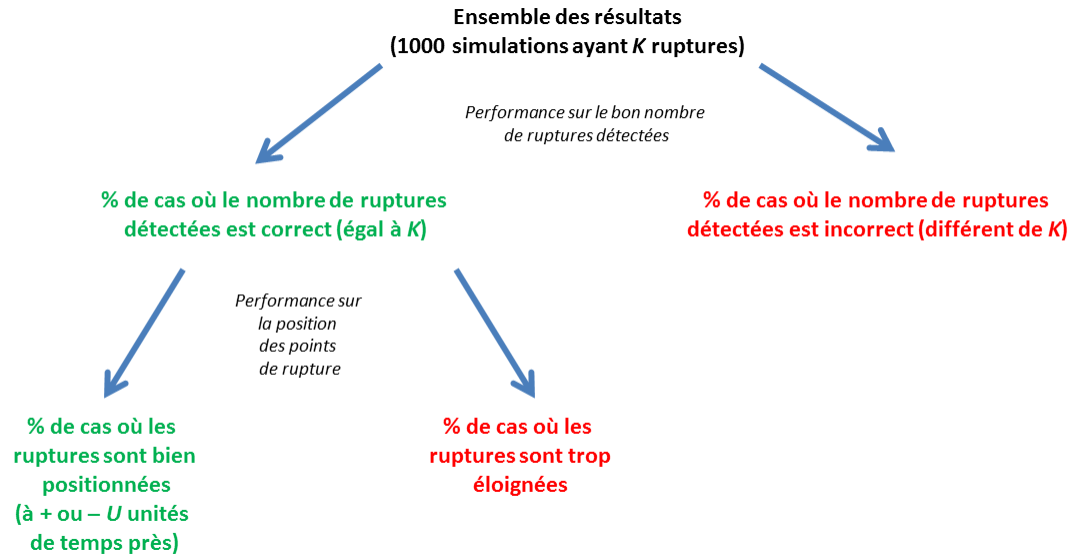


Figure 9: Arbre illustrant les différents critères choisis

Puisque le bruit ajouté aux données simulées peut avoir pour effet de décaler les instants de rupture dans le temps, on définit un seuil de décalage acceptable "petit" qu'on a fixé dans la suite à plus ou moins U unités de temps près. Nous testons plusieurs bruits : 3%, 10% et 20% de la moyenne $\bar{Y}_t$.

Les résultats des simulations sont représentés dans la Table 4.

$\mathbf{A_1}$: $N = 300, K^\star = 3, \sigma = 3\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 10$)
% des résultats détectant le nombre exact de ruptures	95%	94,0%	**100%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	95,9%[3,4]	96,7% [3,3]	**99%** [2,7]
Temps de calcul	2,4%[0,2]	**0,1%** [0,0]	2,7% [1,4]
$\mathbf{A_2}$: $N = 300, K^\star = 3, \sigma = 10\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 10$)
% des résultats détectant le nombre exact de ruptures	**90%**	89,0%	**90%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	85,2%[7,6]	90,0% [5,4]	**93%** [4]
Temps de calcul	2,1%[0,1]	**0,1%** [0,0]	2,3% [0,8]
$\mathbf{A_3}$: $N = 300, K^\star = 3, \sigma = 20\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 10$)
% des résultats détectant le nombre exact de ruptures	87%	86,5%	**89%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	**97,3%**[9,1]	95,6% [8,7]	93,9% [10,5]
Temps de calcul	1,9%[0,1]	**0,1%** [0,0]	2,9% [1,0]
$\mathbf{B_1}$: $N = 1000, K^\star = 6, \sigma = 3\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 15$)
% des résultats détectant le nombre exact de ruptures	98%	97,7%	**99,7%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	**99,8% [1,9]**	97,2% [2,8]	**99,8% [2,9]**
Temps de calcul	34,7%[6,1]	**0,6%** [0,2]	34,5% [7,1]
$\mathbf{B_2}$: $N = 1000, K^\star = 6, \sigma = 10\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 15$)
% des résultats détectant le nombre exact de ruptures	84,6%	84,2%	**95%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	97,0% [7,2]	99,0% [1,0]	**99,8% [2,0]**
Temps de calcul	32,6%[5,1]	**0,4%** [0,1]	33,1% [6,3]
$\mathbf{B_3}$: $N = 1000, K^\star = 6, \sigma = 20\%$	PO-BIC	PELT-BIC	SEP($K_{max} = 15$)
% des résultats détectant le nombre exact de ruptures	83,4%	83%	**90%**
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	98,1% [8,7]	98,1% [9,6]	**98,7% [10,1]**
Temps de calcul	31,9%[7,3]	**0,3%** [0,1]	32,3% [6,3]

Table 4: Performances sur des données simulées (1000 simulations pour chaque scénario). L'écart-type est indiqué entre crochets.

En théorie les résultats de PO-BIC et PELT-BIC devraient être identiques puisque ce sont toutes deux des méthodes exactes. Toutefois on remarque que le PELT-BIC est un petit peu moins performant que la partition optimale puisqu'il sous-estime légèrement le nombre exact de ruptures. Cette sous-estimation se produit lorsqu'une rupture n'est pas très visible comme sur la courbe de la Figure 10. Dans cet exemple, PO-BIC trouve le bon nombre de ruptures aux instants 64, 146 et 225 alors que le PELT-BIC ne détecte pas la rupture en 225. La raison pour laquelle PO-BIC ne fait pas d'erreur est qu'il procède de manière exacte, alors que pour le PELT-BIC on utilise l'approximation définie dans l'équation (15).

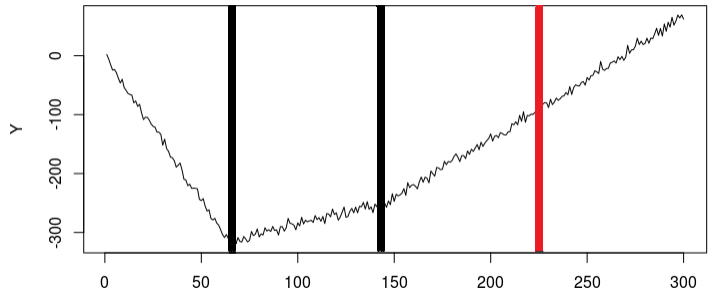


Figure 10: Exemple de simulation dans laquelle une rupture (en rouge) n'est quasiment pas visible (ruptures à 64,145 et 225).

Il se peut que même si

$$\sum_{t=u}^{\ell} C(Y_t, X_t, \theta_{u,\ell}, \sigma^2) + \sum_{t=\ell+1}^{v} C(Y_t, X_t, \theta_{\ell+1,v}, \sigma^2) + \beta < \sum_{t=u}^{v} C(Y_t, X_t, \theta_{u,v}, \sigma^2)$$

pour $u < \ell < v$ et qu'il y ait en fait une rupture en ℓ, on ait

$$\sum_{t=u}^{\ell} C'(Y_t, X_t, \theta_{u,\ell}, \sigma^2) + \sum_{t=\ell+1}^{v} C'(Y_t, X_t, \theta_{\ell+1,v}, \sigma^2) + \beta \geq \sum_{t=u}^{v} C'(Y_t, X_t, \theta_{u,v}, \sigma^2)$$

où C' est l'approximation de C définie par l'équation (13) et qu'on ne détecte donc aucune rupture. Ceci se produit lorsque la différence

$$\sum_{t=u}^{v} C(Y_t, X_t, \theta_{u,v}, \sigma^2) - \sum_{t=u}^{\ell} C(Y_t, X_t, \theta_{u,\ell}, \sigma^2) - \sum_{t=\ell+1}^{v} C(Y_t, X_t, \theta_{\ell+1,v}, \sigma^2) - \beta$$

est positive mais proche de 0.

On ne détecte alors pas de rupture alors qu'il devrait y en avoir une. C'est ce qui explique la sous-estimation du nombre de ruptures par le PELT-BIC.

Lorsque $N = 300$, PO-BIC et PELT-BIC ont des résultats très similaires mais le PELT-BIC est beaucoup plus rapide. Plus le bruit augmente, moins les performances sur les données sont bonnes. Toutefois les résultats sont corrects. L'heuristique de pente (SEP) a une meilleure performance que les deux autres méthodes, mais elle est beaucoup plus coûteuse en terme de calcul (complexité $O(N^2)$).

3.3 Détection de points de ruptures *online*

Les performances des méthodes décrites dans la partie 3.2 dépendent étroitement de la taille des données (excepté l'heuristique de pente qui elle, est en revanche très coû-

teuse en temps de calcul). Si on utilise l'algorithme PELT *offline*, les critères de pénalité tels que le BIC ($K \ln(N)$) entraînent que plus la longueur des données est courte, plus on détectera de petites ruptures souvent non retenues par les experts. Inversement sur les séries temporelles les plus longues, les petites ruptures (de la même amplitude que celles détectées sur les séries plus courtes) ne sont pas toujours repérées. Cela pose un problème en pratique car des séries temporelles similaires peuvent être segmentées différemment selon le critère de pénalité choisi. C'est pourquoi nous définissons une méthode de détection de ruptures *online* qui dépend moins de la taille des données. Cette méthode peut également être très utile lorsque les données sont acquises et enregistrées à la volée.

3.3.1 Etat de l'art

Les méthodes de détection de ruptures *online* sont abondantes dans la littérature (Basseville, Nikiforov, et al., 1993). Plusieurs approches existent.

Nous introduisons dans un premier temps les algorithmes élémentaires, appelés également ***cartes de contrôle*** Shewhart (Crathorne & Shewhart, 1933), très utilisés dans le domaine du contrôle de qualité (Harris, 1989). L'idée est de définir un seuil indiquant si un processus univarié est toujours sous contrôle ou non. Ces méthodes permettent de détecter si un évènement "inhabituel" apparaît durant le processus. Il en existe plusieurs variantes. La méthode de Shewhart a été étudiée par Page (E. Page, 1954). L'algorithme de carte de contrôle basé sur les moyennes mobiles géométriques se révélant plus efficace que celui de Shewhart a été introduit par Roberts dans (Roberts, 1959). L'idée supplémentaire est d'attribuer des poids aux observations (si elles sont récentes, la valeur du poids est importante sinon elle est faible). Une autre variante est l'algorithme *finite moving average chart* dont le principe est similaire à la méthode de Page mais utilise un ensemble de poids fini (E. S. Page, 1954). Plusieurs travaux sont basés sur la même approche (Jamali & Jinlin, 2006), (Kovavrik, 2013), etc...

Une autre approche des méthodes de détection *online* est celle des algorithmes CUSUM. Elle a été introduite par (E. S. Page, 1954). L'idée est de calculer itérativement la somme cumulée définie sur la valeur du rapport de vraisemblance. Cette somme est ensuite comparée à un seuil pour déterminer si un changement est apparu dans le signal univarié. Plusieurs variantes ont ensuite été introduites (Phillips, 1969), (Hinkley, 1969), (Shao & Zhang, 2010), etc...

Les algorithmes bayésiens sont également utilisés et permettent de distinguer un changement dans les paramètres des séries temporelles. Dans (Girshick & Rubin, 1952), Girshick *et al.* ont été les premiers à utiliser cette méthode en supposant une probabilité *a priori* sur l'apparition d'un point de rupture dans les processus markoviens. La version optimale de cet algorithme a été réalisé par Shiryaev dans (Shiryaev, 1961). De nombreuses variantes de cette méthode sont également disponibles (Adams & MacKay, 2007), (Anderson & Grimes, 2008).

3.3.2 Méthode de détection de ruptures *online*

Nous présentons maintenant l'algorithme pour la détection de ruptures *online*.

Nous nous plaçons dans le même cadre que dans la section 3.2.2 en considérant un modèle additif comme dans l'équation (2). On définit une fenêtre de longueur variable $[D(t), t]$ avec $1 \leq D(t) < t$. L'idée est de faire glisser cette fenêtre de gauche à droite le long de la série temporelle.

A chaque glissement de la fenêtre, on cherche à savoir s'il y a une rupture entre le début de la fenêtre $D(t)$ et la fin de la fenêtre t. La première fenêtre est notée $[1, L]$ où on choisit L suffisamment grand pour qu'il soit "raisonnable" d'y trouver une rupture.

Cela signifie qu'on cherche s'il y a une différence significative entre les distributions des séries restreintes aux intervalles $[D(t), s]$ et $[s+1, t]$ de tailles respectives $N_1 = s - D(t) + 1$ et $N_2 = t - s$, où s est une rupture potentielle. La démarche est illustrée dans la Figure 11. Pour cela on effectue donc un test de Chow.

Soient les hypothèses: $\begin{cases} H_0 : \theta_1 = \theta_2 \\ H_1 : \theta_1 \neq \theta_2 \end{cases}$

Sous H_1, le modèle s'écrit: $\begin{cases} Y_u = g_{\theta_1}(X_u) + \varepsilon_u, D(t) \leq u \leq s \\ Y_u = g_{\theta_2}(X_u) + \varepsilon_u, s+1 \leq u \leq t \end{cases}$

Sous H_0, le modèle s'écrit: $Y_u = g_\theta(X_u) + \varepsilon_u, D(t) \leq u \leq t$

La statistique de test est donnée par : $F(s) = \frac{SC-(SC_1+SC_2)}{q} \times \frac{t-D(t)+1-2q}{SC_1+SC_2}$

où

- q est la dimension de θ
- $SC = \sum_{u=D(t)}^{t} \left(Y_u - g_{\hat{\theta}}(X_u)\right)^2$
- $SC_1 = \sum_{u=D(t)}^{s} \left(Y_u - g_{\hat{\theta}_1}(X_u)\right)^2$
- $SC_2 = \sum_{u=(s+1)}^{t} \left(Y_u - g_{\hat{\theta}_2}(X_u)\right)^2$

Cette statistique suit sous H_0 une loi de Fisher de paramètres q et $t - D(t) + 1 - 2q$. On maximise $F(s)$ pour $s = D(t), \cdots, t$ et on pose $\hat{s} = \underset{s=D(t),\cdots,t}{\arg\max} (F(s))$.

On calcule alors la *p-value* et on rejette l'hypothèse H_0 si elle est inférieure à un certain seuil α. Dans ce cas, il y a nécessairement une rupture $\hat{\tau}$ sur l'intervalle $[D(t), t]$. Pour la trouver, on utilise l'algorithme défini dans l'équation (6) sur cet

intervalle. Ensuite la fenêtre glissante est réinitialisée : $D(t)$ devient $\hat{\tau} + 1$ et on remplace t par $D(t) + L$.

Si on accepte l'hypothèse H_0, c'est-à-dire si la *p-value* est supérieure à α, alors on incrémente de 1 la taille de la fenêtre qui devient $[D(t), t + 1]$. On continue ainsi jusqu'à ce que la borne supérieure de la fenêtre soit supérieure ou égale à N.

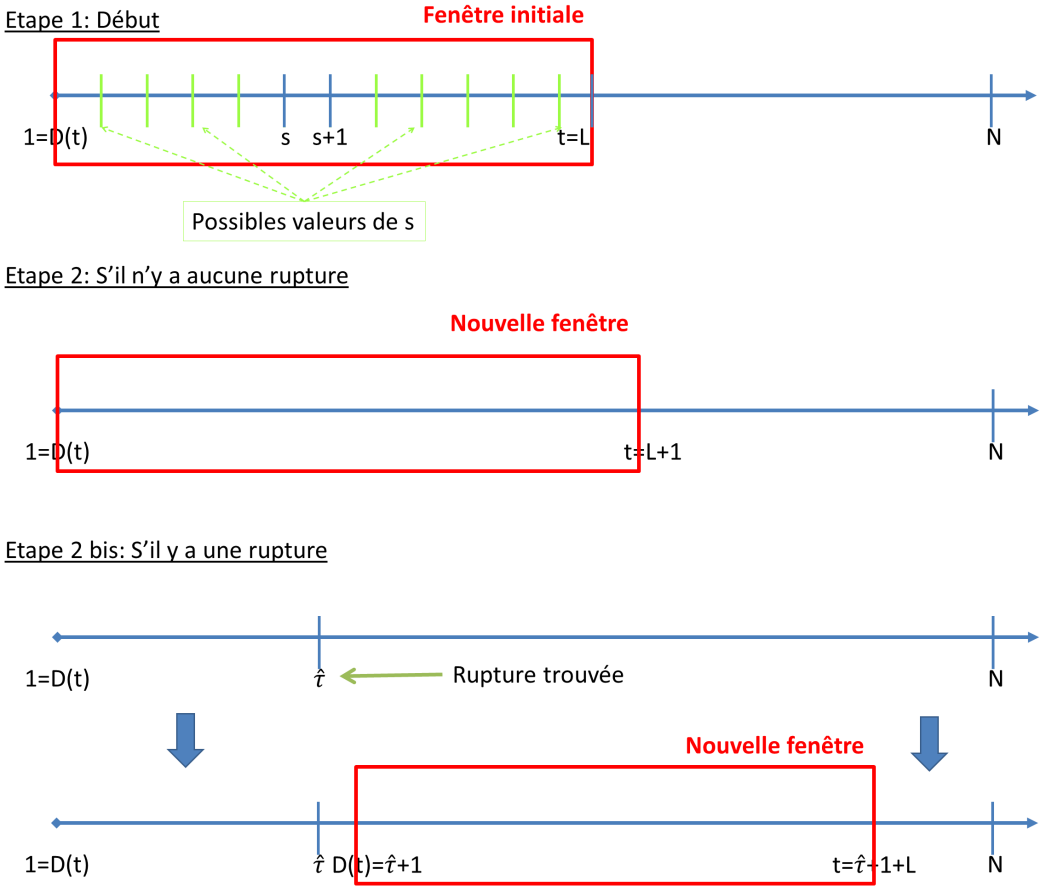


Figure 11: Schéma récapitulatif de la méthodologie *online*.

La méthode est décrite dans la *Procédure 4*:

Procédure 4: Procédure online

- Initialisation : $D(t) = 1, t = L$
- Tant que $t \leq N$
  - Calculer $F(s)$ pour $s = D(t), \cdots, t$ et définir $\hat{s} = \underset{s=D(t),\cdots,t}{\arg\max}(F(s))$
  - Réaliser un test de Chow sur la partition $[D(t), \hat{s}]$ et $[\hat{s}+1, t]$: Calculer la *p-value*
  - Si *p-value* est significatif:
    - Calculer la rupture $\hat{\tau}$ entre $D(t)$ et t avec un algorithme de détection d'une rupture
    - Réinitialiser $D(t) := \hat{\tau} + 1$ et $t := \hat{\tau} + 1 + L$
  - Sinon: incrémenter la taille de la fenêtre: $D(t) := D(t)$ & $t := t + 1$

Si de nouvelles données $(Y_N + 1, Y_N + 2, \cdots)$ sont acquises, on conserve les points de rupture déjà détectés et on fait une mise à jour de la série pour continuer l'exploration.

3.3.3 Applications sur données simulées

Dans cette partie nous utilisons les données simulées selon les scénarios A et B introduits dans la partie 3.2.5, et le modèle linéaire par morceaux de l'équation (12).

Pour le test de Chow, on cherche s'il y a une différence significative entre les paramètres θ_1 et θ_2 des séries restreintes aux intervalles $[D(t), s]$ et $[s+1, t]$ de tailles respectives $N_1 = s - D(t) + 1$ et $N_2 = t - s$.

Soit le modèle $Y_u = \theta^{(1)} + \theta^{(2)}u + \varepsilon_u$ où $\theta = (\theta^{(1)}, \theta^{(2)})$ est le vecteur de paramètres du modèle linéaire.

Sous H_0, le modèle s'écrit:

$$Y_u = \theta^{(1)} + \theta^{(2)}u + \varepsilon_u \, , D(t) \leq u \leq t$$

Sous H_1, le modèle s'écrit: $\begin{cases} Y_u = \theta_1^{(1)} + \theta_1^{(2)}u + \varepsilon_u, \ D(t) \leq u \leq s \\ Y_u = \theta_2^{(1)} + \theta_2^{(2)}u + \varepsilon_u, \ s+1 \leq u \leq t \end{cases}$, avec $(\theta_1^{(1)}, \theta_1^{(2)}) \neq (\theta_2^{(1)}, \theta_2^{(2)})$

Pour le calcul de SC, SC_1 et SC_2 on utilise la fonction d'erreur C définie

dans l'équation (15) :

- $SC_1 = \sum_{u=D(t)}^{s} C(Y_u, X_u, \theta_{D(t),s}, \sigma^2)$ la valeur de la fonction d'erreur sur les données restreintes à l'intervalle $[D(t), s]$.
- $SC_2 = \sum_{u=s+1}^{t} C(Y_u, X_u, \theta_{s+1,t}, \sigma^2)$ la valeur de la fonction d'erreur sur les données restreintes à l'intervalle $[s+1, t]$.
- $SC = \sum_{u=D(t)}^{t} C(Y_u, X_u, \theta_{D(t),t}, \sigma^2)$ la valeur de la fonction d'erreur sur les données restreintes à l'intervalle $[D(t), t]$.

On applique la méthode décrite dans la section 3.3.2. Nous avons testé plusieurs scénarios:

- N : 300 et 1000
- Bruits : 3%, 10% et 20% de la moyenne μ
- Niveau α : 0,01, 0,05 et 10^{-3}

Dans les Tables 5 et 6, on présente les résultats sur les données simulées.

$\mathbf{A_1} : N = 300, Bruit = 3\%\mu, K^\star = 3$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	97,7%	**98,3%**	96,7%
% sur-estimation	0,5%	0%	0%
% sous-estimation	1,8%	1,7%	3,3%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	97,5% [7,8]	**99,4%** [2,4]	97,5% [1,9]
Temps de calcul	**0,68%**[0,1]	0,71% [0,3]	0,75% [0,4]
$\mathbf{A_2} : N = 300, Bruit = 10\%\mu, K^\star = 3$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	**96,5%**	95,9%	95,7%
% sur-estimation	0%	0%	0%
% sous-estimation	3,4%	4,1%	4,3%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	99,3% [4,8]	**99,8%** [3,9]	99,1% [5,6]
Temps de calcul	**0,70%**[0,2]	0,74% [0,53]	0,81% [0,6]
$\mathbf{A_3} : N = 300, Bruit = 20\%\mu, K^\star = 3$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	**93,3**%	92,6%	90%
% sur-estimation	0%	0%	0%
% sous-estimation	6,7%	7,4%	10,0%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 5$)	97,9% [8,2]	**98,2%** [6,5]	95,9% [10,4]
Temps de calcul	**0,86%**[0,2]	0,97% [0,3]	1,12% [0,8]

Table 5: Performances sur des données simulées (1000 simulations pour chaque scénario A_1, A_2 et A_3). L'écart-type est indiqué entre crochets.

$\mathbf{B_1} : N = 1000, Bruit = 3\%\mu, K^\star = 6$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	**100%**	99%	99%
% sur-estimation	0%	0%	0%
% sous-estimation	0%	1%	1%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	98,9% [4,0]	**99,1%** [4,1]	98,9% [4,0]
Temps de calcul	**6,94%**[0,3]	7,01% [0,5]	7,05% [0,4]
$\mathbf{B_2} : N = 1000, Bruit = 10\%\mu, K^\star = 6$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	**97,6%**	97,3%	96%
% sur-estimation	0%	0%	0%
% sous-estimation	2,4%	2,7%	4%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	96,3% [6,1]	**97,1%** [5,8]	96,5% [4,3]
Temps de calcul	**7,49%**[0,4]	7,58% [0,6]	7,92% [0,6]
$\mathbf{B_3} : N = 1000, Bruit = 20\%\mu, K^\star = 6$	$\alpha = 0,05$	$\alpha = 0,01$	$\alpha = 10^{-3}$
% des simulations détectant le nombre exact de ruptures	95,5%	**96%**	94,9%
% sur-estimation	0%	0%	0%
% sous-estimation	4,5%	4%	5,1%
Performance ($\lvert\tau_i - \hat{\tau}_i\rvert \leq 10$)	97,6% [7,1]	**97,9%** [5,9]	97,6% [7,2]
Temps de calcul	**7,63%**[0,3]	8,07% [0,6]	8,25% [0,6]

Table 6: Performances sur des données simulées (1000 simulations pour chaque scénario B_1, B_2 et B_3). L'écart-type est indiqué entre crochets.

Les calculs sont réalisés sur Ubuntu 14.04.3 server, 23 processors Intel(R) Xeon(R) CPU X5675 3.07GHz, avec 6 cœurs chacun. On observe que les résultats sont très bons et les temps de calcul sont raisonnables. Pour $N = 300$ et pour tous les niveaux α de test, les erreurs correspondent essentiellement à une sous-estimation du nombre de ruptures, qui augmente lorsque le niveau du test diminue. Lorsque $N = 1000$, les résultats sont excellents, légèrement dégradés lorsqu'on choisit 10^{-3} comme niveau de test.

La méthode *online* a des résultats plus uniformes par rapport à la méthode *offline* concernant la performance. Elle est également plus sensible aux ruptures quasiment invisibles qui ne sont pas repérées par PELT-BIC par exemple. Les performances de l'algorithme *online* sont donc meilleures que celles de l'algorithme *offline* PELT-BIC, mais au prix d'une complexité en $O(N^2)$ au lieu de $O(N)$.

On remarque enfin que le pourcentage de détections du nombre exact de ruptures diminue lorsque le bruit augmente (jusqu'à 20% de la moyenne), ce qui n'est pas surprenant, puisque le bruit peut "cacher" des ruptures de petite intensité.

En conclusion, on a ainsi défini une méthode de détection de ruptures *on-*

line moins dépendante de la taille des données que la méthode *offline* et qui permet d'obtenir de bons résultats.

3.4 Application aux données réelles

3.4.1 Méthodologie *offline*

Nous avons vu dans la section 3.2.5 que le PELT-BIC réalise le meilleur compromis entre la précision des résultats et le temps de calcul. C'est donc cette méthode qui est utilisée sur les séries temporelles univariées des variables N1 et T3 qui font partie de la base de données réelles décrite dans la section 2.4.2. Dans les Figures 12 (Exemple 1) et 13 (Exemple 2), on représente la variable N1 (image de gauche) pour deux vols complets différents. Le PELT-BIC est appliqué sur ces deux signaux unidimensionnels et les ruptures détectées sont représentées dans l'image de droite par des lignes verticales rouges.

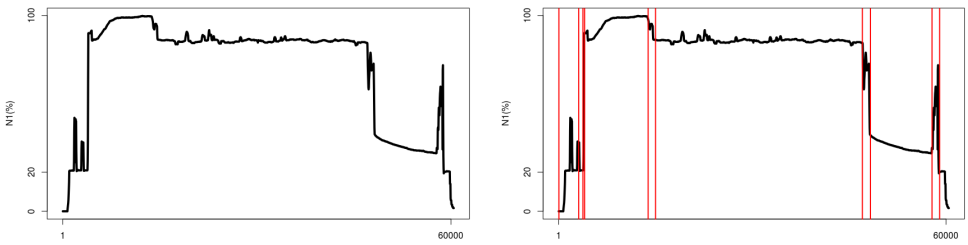


Figure 12: Résultats du PELT-BIC sur la vitesse de rotation N1 (Exemple 1): A gauche la variable N1 observée sur un vol complet et à droite les points de rupture détectés avec le PELT-BIC.

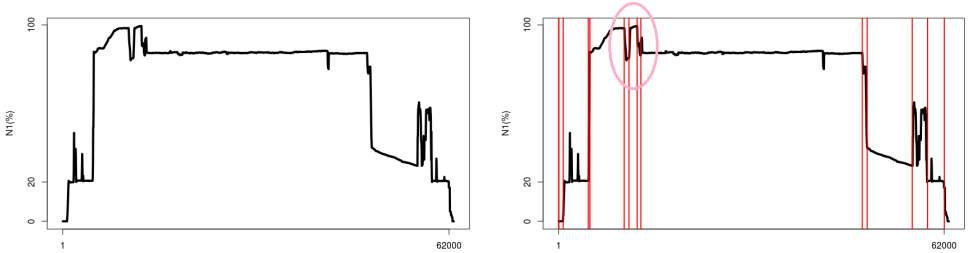


Figure 13: Résultats du PELT-BIC sur la vitesse de rotation N1 (Exemple 2): A gauche la variable N1 observée sur un vol complet et à droite les points de rupture détectés avec le PELT-BIC.

Globalement, les points de rupture sont bien détectés, surtout les plus in-

téressants. Par exemple, le décollage (forte augmentation du N1 passant de 20% à 90 %) est bien délimité par deux ruptures. Dans la Figure 12, le *climb* est segmenté en deux parties, d'abord la lente accélération suivant le décollage puis la petite décélération.

Dans la Figure 13, la fin du *climb* est divisée en trois segments (cercle rose). On note que la descente à la fin du *climb* est suivie d'une montée en régime puis d'une descente. La dernière rupture dans cette phase a lieu après le *climb* (quasiment au début de la phase de croisière). Comme ce changement est bref, cela est fort intéressant que la rupture soit détectée après car on peut ainsi étudier les différentes fins de *climb* qui ont ce comportement et les isoler afin de comprendre leur origine, même si elles sont détectées séparément.

De plus, dans la Figure 12, les bruits apparaissant durant la phase de croisière ne sont pas détectés par le PELT-BIC, ce qui est tout à fait pertinent puisqu'ils sont de très faibles amplitudes par rapport au signal lui-même (en annexe, on applique PELT-BIC sur les mêmes séries temporelles restreintes à la phase de croisière uniquement).

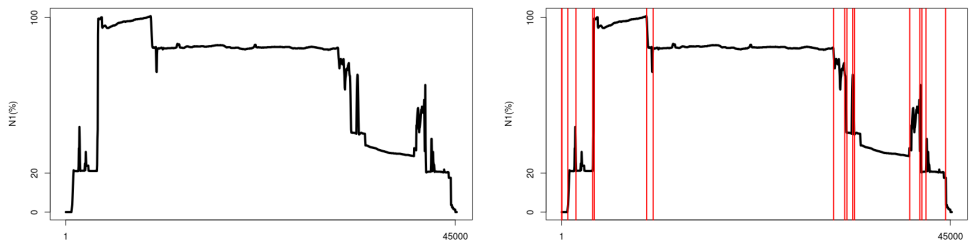


Figure 14: Résultats du PELT-BIC sur la vitesse de rotation N1 (Exemple 3): A gauche la variable N1 observée sur un vol complet et à droite les points de rupture détectés avec le PELT-BIC.

Dans la Figure 14 sont représentées la vitesse de rotation N1 pour un autre vol (Exemple 3) et les ruptures détectées par le PELT-BIC. On remarque que le *climb* est également segmenté par 3 ruptures (voir Figure 15) comme dans l'Exemple 1. Toutefois après la descente du *climb*, une petite descente suivie d'une montée est observée (encerclées en bleu). On note que le PELT-BIC a détecté le point de rupture après la montée. Ce comportement ne serait pas intéressant durant la phase de *taxi*, mais s'il est détecté à la suite du *climb*, il est intéressant de l'analyser et de chercher ce qui l'a provoqué. En outre, dans la phase de descente/approche, les changements abrupts sont bien délimités. Les grandes variations apparaissant durant la descente sont intéressantes, spécialement les fortes accélérations.

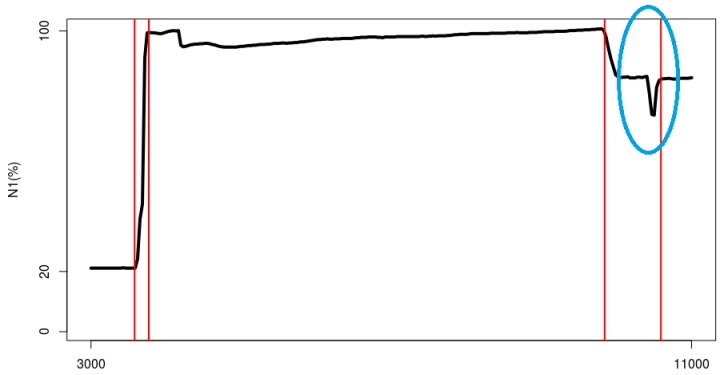


Figure 15: Zoom sur le *climb* (Exemple 3).

La signification de chaque point de rupture détecté doit cependant passer par la validation d'un expert avant toute conclusion.

L'ensemble des méthodes de détection de ruptures feront l'objet d'un package R qui est toujours en cours de développement.

3.4.2 Méthodologie *online*

Dans ce chapitre, nous appliquons la méthode de détection de ruptures *online* développée dans la partie 3.3.2. Nous utilisons comme exemples d'application les mêmes vols mentionnés dans la partie 3.4.1. Dans les Figures 16 (Exemple 1), 17 (Exemple 2) et 18 (Exemple 3), on représente la vitesse de rotation basse pression N1 (image de gauche) pour trois vols complets différents ainsi que les ruptures détectées par les méthodes *online* et *offline*. Sur l'image de gauche on indique les ruptures calculées par la méthode *online*. Sur l'image de droite on ajoute les ruptures détectées par la méthode *offline* PELT-BIC.

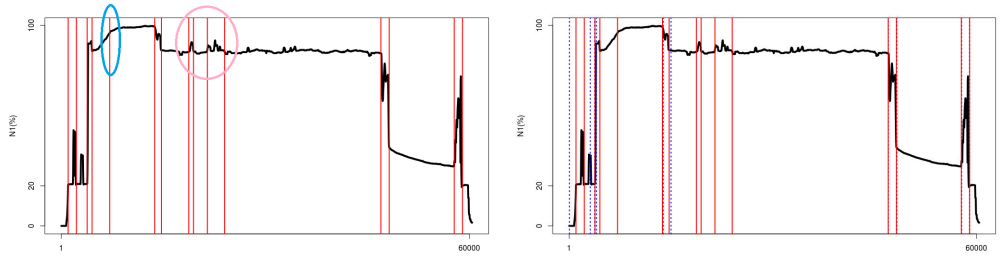


Figure 16: Exemple 1: A gauche la variable N1 observée et les points de rupture détectés par la méthode *online* (en rouge) et à droite la superposition des ruptures détectées par la méthode *offline* PELT-BIC (en bleu pointillé)

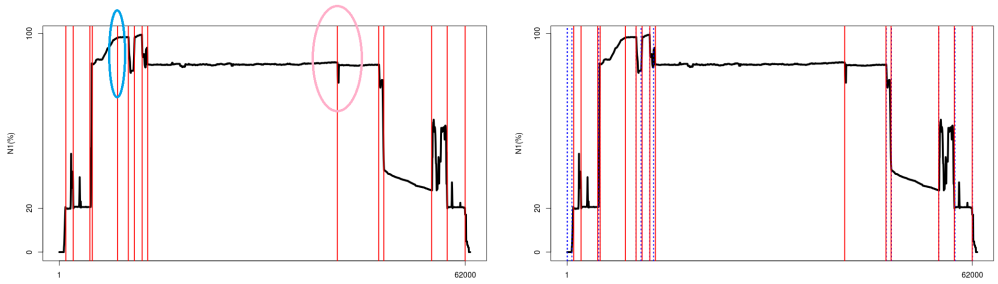


Figure 17: Exemple 2: A gauche la variable N1 observée et les points de rupture détectés par la méthode *online* (en rouge) et à droite la superposition des ruptures détectées par la méthode *offline* PELT-BIC (en bleu pointillé)

On remarque qu'il y a beaucoup de similarités entre les résultats obtenus par la méthode *offline* PELT-BIC et la méthode *online*. Cependant dans les deux exemples et pour la méthode *online*, on détecte plus de ruptures que par la méthode *offline* PELT-BIC (surtout pour les variations de faible amplitude). On remarque dans les Figures 16 et 17 que les ruptures présentes dans les cercles roses ne sont pas détectées par la méthode PELT-BIC. Mais la phase de décollage est bien délimitée par deux ruptures dans les deux exemples.

On observe également une différence dans le *climb*. Celui-ci est découpé en deux parties par la rupture indiquée dans le cercle bleu pour les Exemples 1 et 2. Ceci permet de mettre en évidence des comportements différents des *climbs*: montée rapide et longue phase de stabilisation pour l'Exemple 1, phase de montée plus longue suivie d'une courte phase de stabilisation pour l'Exemple 2 en ce qui concerne la montée en régime, ce qui n'est pas possible avec la méthode PELT-BIC.

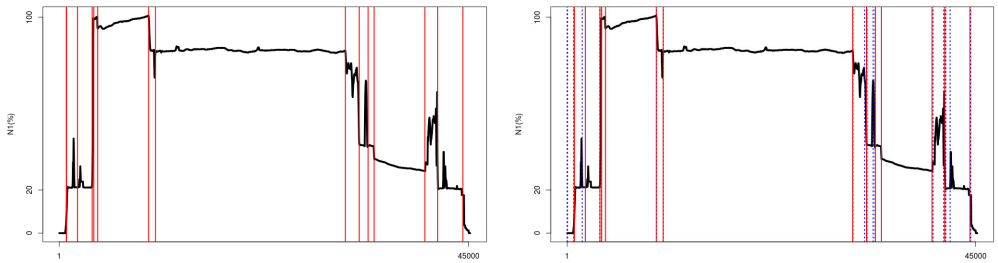


Figure 18: Exemple 3: A gauche la variable N1 observée et les points de rupture détectés par la méthode *online* (en rouge) et à droite la superposition des ruptures détectées par la méthode *offline* PELT-BIC (en bleu pointillé).

Dans la Figure 18 (Exemple 3), les ruptures obtenues par les deux méthodes sont quasiment les mêmes. La descente qui suit le *climb* est bien visible puisqu'elle correspond à une seule phase. De plus, contrairement aux deux exemples précédents,

la phase de *climb* n'est pas divisée en deux parties. Elle correspond à une montée en régime lente et n'a pas de partie stabilisée. Cependant les changements de grande amplitude pendant la descente sont mieux détectés par la méthode *offline* que par la méthode *online*.

Globalement la différence entre les deux méthodes est plutôt subtile. La méthode *online* est un peu plus sensible aux petits changements de faible amplitude survenant lorsque les mesures sont quasiment constantes, alors que la méthode *offline* est plus sensible aux changement de forte amplitude durant une phase de forte variation comme celles qui se produisent durant les phases de descente et d'approche. Sur le conseil des experts, nous utilisons dans la suite la méthode *offline* PELT-BIC qui a l'avantage d'être plus rapide.

3.4.3 Des points de rupture aux phases transitoires

Les phases sont délimitées par les points de rupture détectés par l'algorithme PELT-BIC. Rappelons qu'une phase est transitoire dans le contexte univariée lorsqu'elle n'est pas stable et donc lorsque le signal différencié n'est pas approximativement centré. Cependant certaines phases transitoires (non stables) peuvent correspondre à des petits incidents comme un trou sur la piste ou un petit coup de manette du pilote. Pour ces "incidents", il y a en général retour rapide à la valeur initiale (cela ne les rend pas moins importants et il peuvent être intéressants à analyser, surtout s'ils sont porteurs de signaux faibles). Donc on ne cherche à détecter que les phases transitoires dont la différence entre la valeur finale et la valeur initiale est supérieure à un certain seuil. Ce seuil est défini par les experts.

Trois types de phases sont identifiés selon la valeur de cette différence: les phases croissantes, les phases décroissantes et les phases stabilisées (non transitoires).

Si on considère un seuil égal à 10%, après identification, sur l'ensemble des données de vol étudiées, plus de 4000 phases sont recensées pour chaque type de transitoires (croissantes et décroissantes). Sur la Figure 19, 7 phases transitoires croissantes, calculées à partir de la variable N1, sont mises en évidence sur un vol tout entier et dans la Figure 20, on a représenté ces 7 phases en les calant à gauche.

Dans les Figures 19 (vol tout entier) et 20 (phases croissantes détectées calées à 1), le tronçon noir représente le démarrage moteur et commence à 0%. Le rouge est la transition entre le démarrage moteur et le *taxi-out*. Le vert représente le décollage (saut de 20% à 80%). Le bleu qui est plus long représente le *climb*. Les tronçons jaune, bleu clair et fushia sont localisés durant la descente.

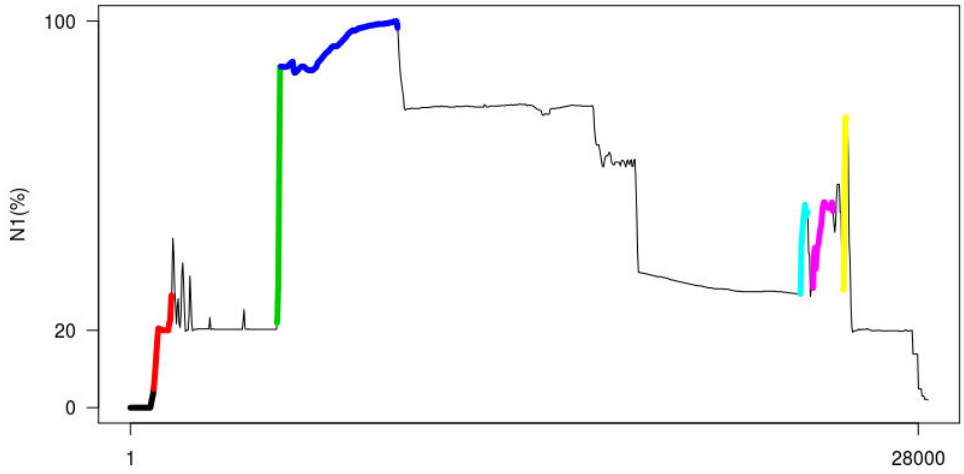


Figure 19: Représentation des 7 phases croissantes mises en évidence sur le vol entier.

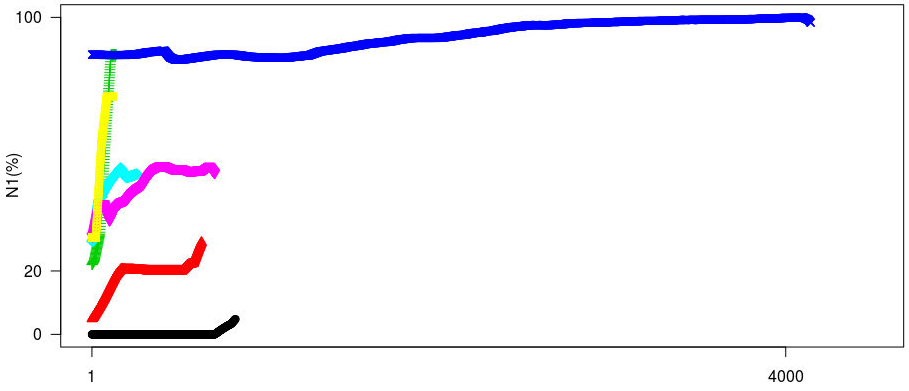


Figure 20: Représentation des 7 phases croissantes détectées dans un vol. Elles ont été calées à gauche. Elles sont de niveaux moyens et de longueurs différents.

Dans la Figure 21 sont représentées les phases transitoires croissantes de N1 pour un même moteur durant 85 vols.

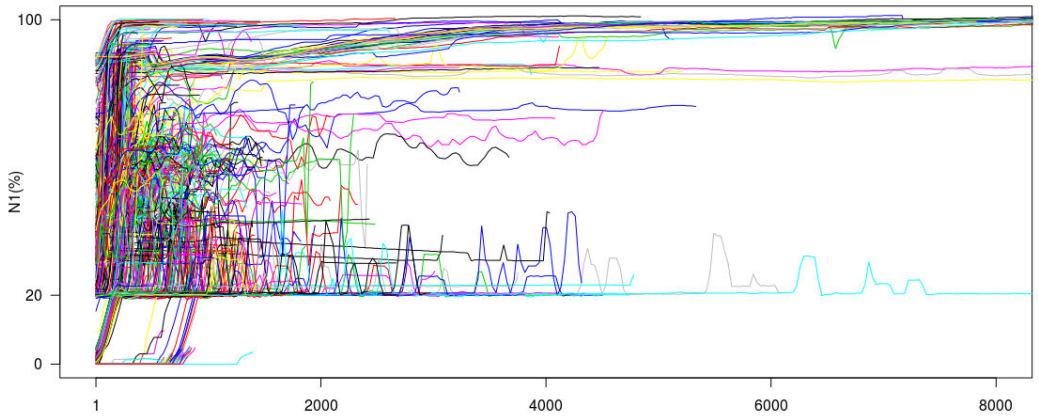


Figure 21: Représentation des phases croissantes de 85 vols d'un moteur, calées à gauche. La majorité des valeurs est supérieure à 20% puisqu'en croisière et dans la plupart des modes de vol, N1 est supérieure à 20%.

Dans la Figure 19, il est facile d'identifier le mode dans lequel les phases transitoires se déroulent car on peut les situer chronologiquement. Dans la Figure 20, on peut facilement identifier quelques phases à l'oeil nu (courbes noire, rouge, verte, bleue) mais on peut avoir un doute sur les courbes jaune, fushia et bleue claire. Dans la Figure 21, l'identification est impossible. Ce qui montre que sans le contexte du vol, on ne peut aisément identifier toutes les phases. De plus le mode de vol peut changer au cours d'une même phase.

C'est pourquoi nous avons mis en place un outil permettant d'identifier le ou les mode(s) de vol d'une phase. Pour cela on utilise la variable Flight_Mod mentionnée dans la section 2.4. Cette variable indique le mode de vol à chaque instant du vol. Toutefois, cette variable n'existe pas dans le cas des données d'essai. Ainsi elle ne peut être utilisée que pour l'identification des phases. Dans la Figure 22, on représente le vol de la Figure 19 en indiquant les modes successifs. Le codage utilisé (0,...,9) est celui qui a été introduit dans la Table 2 de la section 2.4. Sur l'axe de temps, on a représenté la variable FM tous les 700 pas de temps.

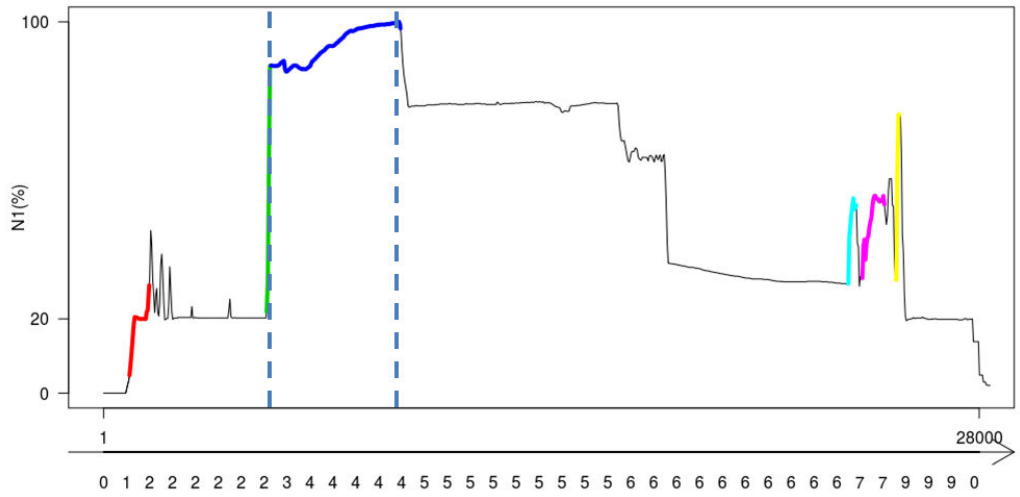


Figure 22: Représentation des 7 phases transitoires croissantes mises en évidence sur le vol entier ainsi que les modes de vol représentés tous les 700 pas sur l'axe fléché.

Dans la Figure 22, on voit bien que plusieurs modes de vols se succèdent. La durée du mode varie selon son type. Par exemple, les durées de croisière et de descente sont plus longues que les durées de *taxi-out* et *taxi-in*. On observe aussi qu'au cours d'une phase transitoire, on peut changer de mode de vol. Par exemple la phase colorée en bleue foncée passe du mode 3 (décollage) au mode 4 (*climb*).

Chaque phase transitoire peut être étiquetée par la suite à l'aide des modes par lesquels elle passe. On note cette étiquette ID_FM. On considère par exemple la phase transitoire croissante bleue claire qui se déroule pendant la descente (Figure 23, pas de temps différent de la Figure 22), à chaque instant de cette phase, le mode de vol est "descente" représenté par le codage '6' et donc l'étiquette ID_FM de cette phase est 6.

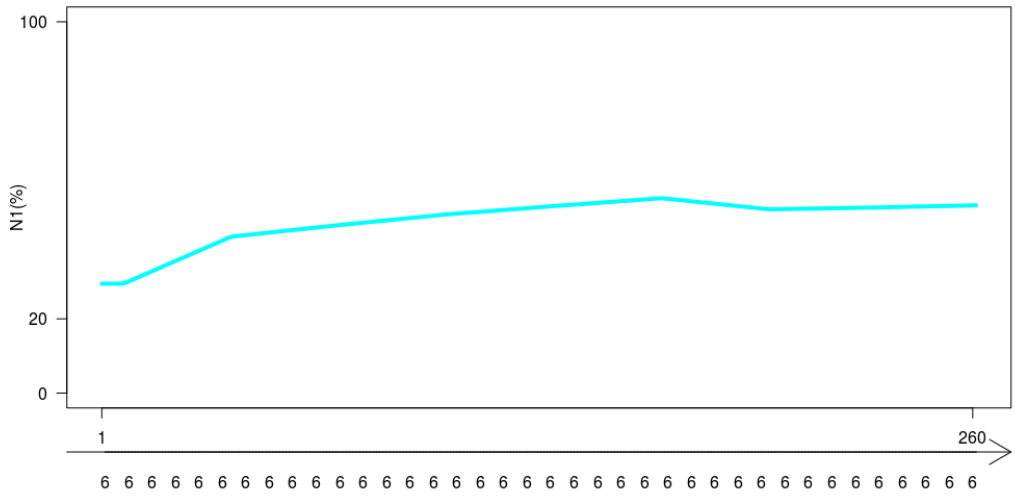


Figure 23: Représentation de la phase croissante bleue claire du vol de la Figure 20.

Lorsqu'une phase passe d'un mode à un autre, son étiquette ID_FM est formée de deux chiffres; lorsqu'une phase correspond à 3 modes différents, son étiquette est formée de trois chiffres. Après examen, on a remarqué qu'au maximum 3 modes de vol se retrouvent dans une même phase et donc l'étiquette ID_FM est constitué d'un ou deux ou trois chiffres. Dans les Figures 24 et 25, on représente des phases ayant 2 modes et 3 modes, avec leur étiquette.

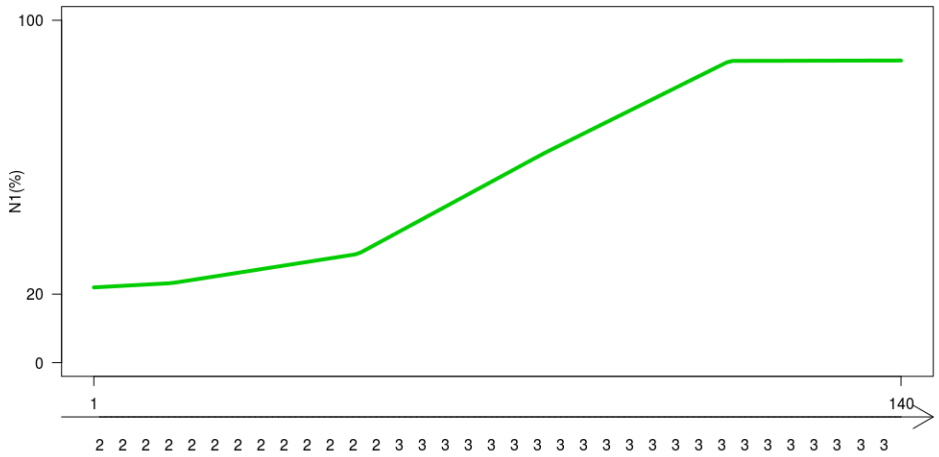


Figure 24: Représentation d'une phase croissante d'un vol ayant 2 modes différents; ID_FM='23'.

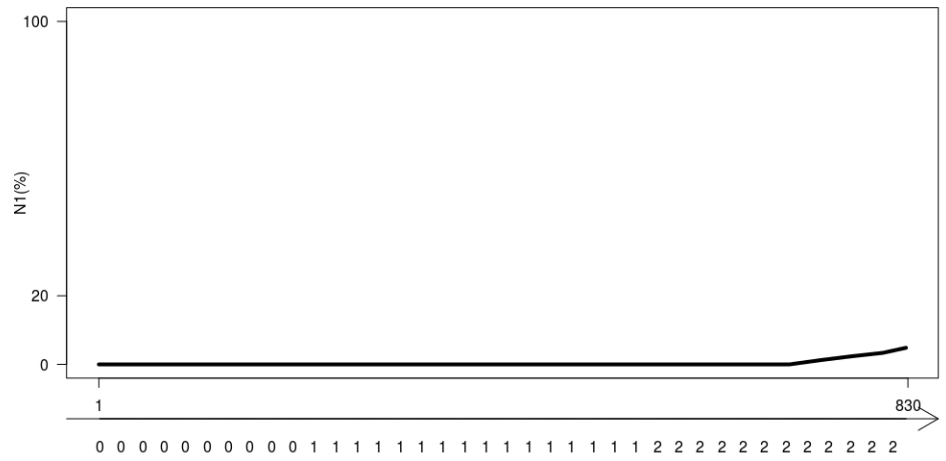


Figure 25: Représentation d'une phase croissante d'un vol ayant 3 modes; ID_FM='012'.

Dans la suite, les étiquettes ID_FM des phases transitoires seront utilisées pour les caractériser.

3.5 Conclusions

Nous avons implémenté une méthode de segmentation d'un signal univarié afin d'en extraire les phases transitoires et les phases stabilisées.

Parmi les méthodes implémentées, le PELT-BIC représente le meilleur compromis du point de vue des critères de performance (tant en *online* qu'en *offline*). L'ensemble des résultats (sur données simulées et données réelles) a été validé par ces critères de performance et également par les experts.

La méthode *online* est très proche de la méthode *offline*. Elle est un peu plus sensible aux changements abrupts de faible amplitude tels que ceux qui se produisent dans les croisières et elle est moins dépendante de la longueur de la série temporelle (contrairement au *offline* PELT-BIC). Cependant cette méthode est plus coûteuse que le PELT-BIC en temps de calcul.

Les points de ruptures détectés et validés définissent une segmentation de chaque série temporelle observée, qui peut alors être caractérisée par la suite de ses phases transitoires et stabilisées. Dans le chapitre suivant, on s'intéresse à toutes les phases transitoires croissantes et décroissantes tous vols confondus. Il est primordial dans la suite de ce travail que la fréquence d'acquisition soit la même d'un vol à un autre dans le but de les comparer et de les analyser.

4 *Clustering* des phases transitoires

Nous cherchons à étudier et classer les phases transitoires extraites de l'ensemble des vols de la base de données. Après avoir identifié les phases croissantes, décroissantes et stabilisées, on ne tient plus compte de la position de la phase dans la série temporelle et chaque segment est maintenant calé au temps 1.

Dans la suite, nous nous bornons à travailler sur les phases croissantes et décroissantes puisque les phases stabilisées apportent peu d'information sur les anomalies. Comme il s'agit de classer les phases transitoires croissantes (ou décroissantes) sans connaître la variable Flight_Mod correspondante, les données à classer ne sont pas labellisées et nous devons utiliser des méthodes de classification non supervisées (ou *clustering*). Pour simplifier la présentation, on s'intéresse d'abord aux seules phases transitoires croissantes, mais les résultats empiriques sont obtenus dans les deux cas. Rappelons que ces phases sont déterminées par la méthode *offline* puisque les deux méthodes *offline* et *online* sont quasiment équivalentes (voir la partie 3.4).

Dans ce chapitre, après avoir rappelé quelques notions sur les algorithmes de classification non supervisée, nous introduisons l'algorithme de Kohonen (*Self-Organizing Map* - SOM), le SOM Kernel et le SOM relationnel. A l'aide de deux de ces méthodes, nous déterminons des classes de phases transitoires (croissantes et décroissantes) de la variable N1. Nous comparons les deux méthodes du point de vue de la qualité du *clustering* en tenant compte des avis des experts. Ensuite nous analysons l'effet de la variable température (T3) sur les clusters de phases transitoires de N1. Pour cela, chaque cluster est à son tour partitionné en sous-clusters selon la variable T3. Nous utilisons une carte de Kohonen à l'intérieur de chaque cluster, de façon hiérarchisée.

Rappelons que le but des méthodes de *clustering* est de regrouper des observations/objets/individus dans des classes homogènes et bien séparées. Une fois définie une dissimilarité entre observations, il s'agit de construire des classes telles que les dissimilarités prises deux à deux entre observations d'une même classe soient "petites", et les dissimilarités entre observations de classes différentes soient "grandes".

En pratique on rencontre deux situations :

1. Si les observations sont des vecteurs dans $\mathbb{R}^p$, X_i et $X_{i'}$, on définit en général la dissimilarité entre X_i et $X_{i'}$ en prenant le carré d'une distance euclidienne de X_i et $X_{i'}$, soit
$$D^2(X_i, X_{i'}) = (X_i - X_{i'})\Lambda(X_i - X_{i'})^T,$$
où Λ est une matrice symétrique définie positive (des pondérations, des variances, ...).

2. Si les observations proviennent d'espaces plus abstraits (séries temporelles,

graphes, ...), on peut alors définir soit des mesures de similarité (des noyaux, ...) ou des mesures de dissimilarité. Dans ce cas, les observations sont connues à travers leur matrice de distances ou de dissimilarités $\mathbb{D} = (D_{i,i'})$. Ces dissimilarités peuvent être données *a priori*, ou calculées selon le contexte, c'est ce qui est fait dans la section 4.2.3.

4.1 Algorithmes de *clustering* dans le cas euclidien

On suppose qu'on a N données d'échantillonnage $(x_1, x_2, \cdots, x_N)$ dans l'espace euclidien $\mathbb{R}^p$, réparties en U classes $(C_1, C_2, \cdots, C_U)$, obtenues par une méthode de *clustering* quelconque. L'ensemble des données est noté Ω. Ces U classes forment une partition de Ω,

$$\Omega = C_1 \cup C_2 \cup \cdots \cup C_U$$

et

$$\forall i \neq j, C_i \cap C_j = \emptyset.$$

Si N_u est la cardinalité de la classe C_u alors $N = \sum_{u=1}^{U} N_u$. On note $\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i$ le centre de gravité de l'ensemble des données et $\forall u = 1, \cdots, U$, $\bar{x}_u = \frac{1}{N_u}\sum_{x_i \in C_u} x_i$ le centre de gravité de la classe C_u.

On définit la somme des carrés totale par

$$S_T(C_1, C_2, \cdots, C_U) = \sum_{i=1}^{N} \|x_i - \bar{x}\|^2 \tag{16}$$

On définit également la somme des carrés intra et la somme des carrés inter:

$$S_{\textit{Intra}}(C_1, C_2, \cdots, C_U) = \sum_{u=1}^{U} \sum_{x_i \in C_u} \|x_i - \bar{x}_u\|^2 \tag{17}$$

$$S_{\textit{Inter}}(C_1, C_2, \cdots, C_U) = \sum_{u=1}^{U} N_u \|\bar{x}_u - \bar{x}\|^2 . \tag{18}$$

On sait que :

$$S_T = S_{Intra} + S_{Inter}.$$

Pour réaliser une "bonne" classification pour un nombre de classes fixé, on cherche à minimiser la somme des carrés intra, ce qui revient à maximiser la somme des carrés inter. Cela équivaut à obtenir des classes homogènes et bien séparées. Dans

les implémentations algorithmiques, ce problème n'admet pas de solution unique et on obtient en général un minimum local de la somme des carrés intra, dépendant de l'initialisation. Si le nombre de classes n'était pas fixé, la minimisation de la somme des carrés intra conduirait à une solution triviale nulle à N classes, où chaque observation constitue une classe.

L'une des méthodes de *clustering* les plus utilisées est la **classification hiérarchique ascendante** (CHA) (Gower & Ross, 1969). Le nombre de classes n'est pas fixé *a priori*. La situation initiale correspond à N classes à un élément, et les regroupements successifs conduisent à une seule classe contenant toutes les observations. Cet algorithme fournit en fait une suite de partitions emboîtées de l'ensemble des données.

Une autre méthode très utilisée est la méthode des **centres mobiles** (appelée également algorithme de Forgy) (Forgy, 1965). On se donne le nombre de classes U *a priori*. On initialise U prototypes $p_1, p_2, \cdots, p_U$ aléatoirement dans l'espace des observations et on en déduit les classes $C_1, \cdots, C_U$ par la méthode des plus proches voisins, c'est-à-dire qu'on assigne à la classe C_u les observations plus proches de p_u que des autres prototypes p_v avec $v \neq u$. Les prototypes sont ensuite recalculés en prenant les centres de gravité des classes et ainsi de suite. On redéfinit les classes, on recalcule les prototypes, etc.... Cette méthode est déterministe (à part l'initialisation des prototypes), on montre que la somme des carrés intra S_{Intra} est décroissante et converge vers un minimum local.

La méthode d'**apprentissage compétitif** (ou *K-means*) (Lloyd, 1982) est la version *online* de la méthode des centres mobiles. Le but est partitionner les N observations en U classes en minimisant la somme des carrés intra. A chaque unité u, est attaché un prototype p_u élément de $\mathbb{R}^p$. Au temps $t = 0$, les prototypes sont initialisés aléatoirement et notés $p(0) = (p_1(0), ..., p_U(0))$. L'algorithme *K-means* est défini itérativement comme suit :

- on tire aléatoirement au temps t une observation $x(t+1)$
- on détermine le numéro w du prototype gagnant :

$$w(x(t+1), p_1(t), p_2(t), \cdots, p_U(t)) = \arg\min_u \{\|x(t+1) - p_u(t)\|\};$$

- on met à jour les prototypes en posant :

$$\begin{cases} p_w(t+1) = p_w(t) + \mu(t)(x(t+1) - p_w(t)) \\ p_u(t+1) = p_u(t) \ \ pour \ \ u \neq w \end{cases}$$

où $\mu(t)$ est un paramètre d'apprentissage (positif, inférieur à 1 et décroissant ou constant).

Pour ce qui est de l'optimisation numérique, si la suite des paramètres $\mu(t)$ vérifie les conditions classiques de Robbins-Monro, à savoir si

$$\sum_{t>0} \mu(t) = +\infty \quad \sum_{t>0} \mu^2(t) < +\infty,$$

alors on peut montrer que cet algorithme converge vers un minimum (local) de S_{Intra}. C'est une méthode *online* puisqu'à chaque étape, il suffit d'avoir en mémoire l'état courant des prototypes et la nouvelle observation. On verra par la suite que l'algorithme de Kohonen est une généralisation de cette méthode.

L'**algorithme EM** (*Expectation-Maximization*) (Dempster, Laird, & Rubin, 1977) est un algorithme très connu qui généralise la méthode *K-means*. On suppose que les données observées proviennent d'un mélange de U distributions d'une famille paramétrique connue. La méthode EM permet d'estimer les paramètres en cherchant le maximum de vraisemblance, quand la maximisation directe de celle-ci n'est pas possible. La méthode est itérative, la première étape ("*E-step*") consiste à calculer l'espérance de la vraisemblance complète, puis la seconde étape ("*M-step*") maximise cette quantité, telle qu'elle a été estimée à l'étape précédente. Le *clustering* est réalisé *a posteriori* via MAP.

4.2 Cartes auto-organisées (*Self-Organizing Maps*) [SOM]

L'algorithme SOM en version *online*, défini par T. Kohonen dès 1982 (Kohonen, 2001), est un algorithme d'apprentissage non supervisé, très utilisé dans différents domaines applicatifs. C'est un algorithme de classification (*clustering*) qui généralise l'algorithme d'apprentissage compétitif en rajoutant une structure de voisinage entre les classes. Ainsi des données proches dans l'espace des données appartiennent à la même classe (comme pour tout algorithme de classification) ou à des classes voisines. Cette structure de voisinage est en général bi-dimensionnelle (à l'image d'une grille) et en disposant les classes sur la grille, on obtient une représentation visuelle des données multidimensionnelles sur une carte (dite carte de Kohonen). L'algorithme SOM est de plus facile à programmer, sa complexité est linéaire par rapport au nombre de données, et il est particulièrement adapté aux bases de données de grande taille. Défini initialement pour des données numériques de $\mathbb{R}^p$, l'algorithme a été étendu à divers types de données (qualitatives ou relationnelles) (Cottrell, Olteanu, Rossi, & Villa-Vialaneix, 2016). Dans la suite, nous ne présentons que des versions *online*.

4.2.1 SOM numérique

Dans sa version initiale, SOM est défini pour des vecteurs dans $\mathbb{R}^p$. On considère U unités (quelques fois appelées neurones) disposées sur un réseau régulier, en général de dimension 2 (réseau en grille), mais il est également possible de considérer un réseau de dimension 1 (réseau en ficelle), voire des réseaux sur des cylindres (données

périodiques) ou des sphères. En réalité, l'algorithme SOM peut être défini pour des réseaux de n'importe quelle dimension, mais alors il perd l'essentiel de ses qualités de visualisation.

On définit une distance entre les couples d'unités du réseau. La distance $d(u, v)$ entre les unités u et v peut être définie de plusieurs façons, nous la prendrons dans la suite égale à la distance du plus court chemin de l'unité u à l'unité v sur la grille.

Si $\mathbb{U} = \{1, ..., U\}$, et t est le temps (au sens de l'algorithme d'apprentissage), on définit une fonction de voisinage $h_t : \mathbb{R}^+ \to \mathbb{R}^+$. Si cette fonction ne dépend pas du temps, elle est notée h. La fonction h_t satisfait les conditions suivantes :

- si u et v sont deux unités, $h_t(d(u, v))$ est une fonction décroissante de la distance $d(u, v)$ sur le réseau, nulle si cette distance est supérieure à un certain seuil.
- $h_t(d(u, u)) = 1$.

Un des choix les plus classiques est une fonction binaire *step*, qui vaut 1 si la distance $d(u, v)$ est inférieure à un certain rayon (qui peut décroitre avec le temps) et 0 sinon. On peut aussi définir une fonction de voisinage plus régulière (fonction de forme gaussienne) en posant:

$$h_t(d(u, v)) = \exp(-\frac{d(u, v)^2}{2\sigma^2(t)})$$

où $\sigma^2(t)$ peut diminuer au cours du temps pour réduire l'intensité et la portée du voisinage.

A chaque unité u, est attaché un prototype p_u élément de $\mathbb{R}^p$. Au temps $t = 0$, les prototypes sont initialisés aléatoirement et notés $p(0) = (p_1(0), ..., p_U(0))$.

L'algorithme SOM est défini itérativement comme suit :

- à chaque étape t, une donnée $x(t + 1)$ est choisie au hasard
- étape d'affectation : l'unité gagnante est définie par :

$$w(x(t + 1)) = \underset{u\in\{1,...,U\}}{\arg\min} ||x(t + 1) - p_u(t)||^2$$

- étape de représentation : tous les prototypes sont mis à jour en posant :

$$p_u(t + 1) = p_u(t)+\mu(t)h_t\left(d(u, w(x(t + 1)))\right)\left(x(t + 1) - p_u(t)\right), u = 1, \cdots, U$$

où $\mu(t)$ est un paramètre d'apprentissage (positif, inférieur à 1 et décroissant ou constant).

Après l'apprentissage, pour tout $u = 1, ..., U$, la classe C_u est définie ici par la méthode des plus proches voisins comme l'ensemble des données initiales les plus proches de p_u que de tout autre prototype. La carte de Kohonen est alors la représentation du diagramme de Voronoï où le contenu de chaque classe sont disposés en fonction de la structure du voisinage. Cette carte a alors deux propriétés:

- La propriété de quantification : les prototypes représentent l'espace de données initiales aussi précisément que possible.
- La propriété d'auto-organisation : les prototypes conservent la topologie des données.

Deux indicateurs de qualité permettent ainsi d'évaluer l'organisation et la quantification de la classification obtenue:

L'**erreur topographique** quantifie l'organisation de la carte. On compte le nombre de fois où la seconde unité gagnante (second BMU-*best matching unit*) est une voisine directe de l'unité gagnante (BMU) pour chaque observation. L'erreur topographique est égale à 0 si toutes les secondes unités gagnantes sont dans le voisinage direct des unités gagnantes et correspond à une très bonne préservation de la topologie des observations.

L'**erreur de quantification** mesure la qualité du *clustering*. Elle est égale à la distance moyenne entre les observations et les prototypes auxquels elles sont assignées. Elle est égale à:

$$\frac{1}{N}\sum_{u=1}^{U}\sum_{x_i \in C_u} \|x_i - p_u\|^2$$

où p_u est le prototype de la classe u.

4.2.2 SOM Kernel

L'algorithme initial présenté précédemment peut être étendu à des cas plus généraux pour des données plus complexes que des vecteurs numériques. Nous nous intéressons au cas où les données $(x_1, \cdots, x_N) \in \mathcal{X}$, où $\mathcal{X}$ est un espace abstrait, autre que $\mathbb{R}^p$ (graphes, texte, images, séries temporelles,...). Il se pose alors deux questions. Qu'est-ce qu'un prototype dans ce cas? Comment calcule-t-on la distance entre un prototype et une observation?

Supposons que les données sont connues via un noyau (Kernel) $\mathbf{K} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ tel que :

- K est symétrique, c'est-à-dire que $K(x_i, x_j) = K(x_j, x_i), \forall i, j = 1, \cdots, N$,
- K est défini positif, $\forall m > 0, \forall (x_i)_{i=1,\cdots,m} \in \mathcal{X}, \forall (\alpha_i)_{i=1,\cdots,m} \sum_{i,j=1}^{m} \alpha_i \alpha_j K(x_i, x_j) \geq 0$.

On peut alors adapter l'algorithme SOM comme l'ont proposé (MacDonald & Fyfe, 2000). Pour cela on utilisé la propriété fondamentale des noyaux, démontrée dans (Aronszajn, 1950), et qui permet de réécrire le noyau comme un produit scalaire :

Theorem 1. *Il existe un espace de Hilbert $\mathcal{H}$, également appelé espace de caractéristiques et une application $\psi : \mathcal{X} \to \mathcal{H}$ tel que $K(x_i, x_j) = \langle \psi(x_i), \psi(x_j) \rangle_{\mathcal{H}}$.*

Ainsi, les prototypes peuvent être définis dans l'espace $\mathcal{H}$ comme égaux à des combinaisons convexes des projections $(\psi(x_i))$:

$$p_u(t) = \sum_{i=1}^{N} \omega_{u,i}(t) \psi(x_i)$$

où $\omega_{u,i}(t) \geq 0$ et $\sum_{i=1}^{N} \omega_{u,i}(t) = 1$. On note $\omega_u = (\omega_{u,1}, \cdots, \omega_{u,N})^T$, où T est l'opérateur de transposition.

Par ailleurs, à partir de la matrice du noyau calculée sur un échantillon $\mathbf{K} = (K(x_i, x_j))_{i,j=1,\cdots,N}$, on peut calculer la distance entre une observation et un prototype à partir de la distance engendrée par K dans l'espace $\mathcal{H}$:

$$\begin{aligned} \|\psi(x_i) - p_u(t)\|_{\mathcal{H}}^2 &= \left\| \psi(x_i) - \sum_{j=1}^{N} \omega_{u,j} \ \psi(x_j) \right\|_{\mathcal{H}}^2 \\ &= \left\langle \psi(x_i) - \sum_{j=1}^{N} \omega_{u,j} \ \psi(x_j), \psi(x_i) - \sum_{j=1}^{N} \omega_{u,j} \ \psi(x_j) \right\rangle \\ &= \mathbf{K}_{ii} - 2\mathbf{K}_i \, \omega_u(t) + (\omega_u(t))^T \mathbf{K} \omega_u(t) \end{aligned} \tag{19}$$

où $\mathbf{K}_i$ est la i-ème ligne du noyau $\mathbf{K}$. On remarque alors que la distance ne s'écrit qu'en fonction de $\mathbf{K}$ et des ω_u.

On peut alors définir l'unité gagnante comme pour le SOM numérique. Dans la version *online* de l'algorithme (MacDonald & Fyfe, 2000), la mise à jour des prototypes se fait au moyen de la mise à jour des coordonnées de (ω_u):

$$\omega_u(t+1) = \omega_u(t) + \mu(t) h_t \left(d(u, w(x_i(t+1))) \right) \left(\mathbb{1}_i - \omega_u(t) \right), \forall u = 1, \cdots, U$$

où $x_i(t+1)$ est l'observation au temps $t+1$ et où $\mathbb{1}_i$ est un vecteur de dimension N dont toutes les composantes sont nulles sauf la i-ème qui est égale à 1.

4.2.3 SOM relationnel

Un cas encore plus général consiste à considérer que les observations sont connues à travers des dissimilarités uniquement. C'est le cas par exemple des dissimilarités entre séquences calculées par la méthode *Optimal Matching* (Needleman & Wunsch, 1970) ou des distances du plus court chemin sur un graphe.

Nous considérons qu'on dispose d'une matrice de dissimilarités $\mathbb{D} = (D(x_i, x_j))_{i,j=1,..,N}$, telle que :

- $D(x_i, x_j) = D(x_j, x_i)$,
- $D(x_i, x_j) \geq 0$,
- $D(x_i, x_i) = 0$.

Dans ce cas, grâce au théorème de Goldfarb (Goldfarb, 1984), on peut montrer que les données peuvent être plongées dans un espace pseudo-euclidien.

Theorem 2. *Il existe deux espaces euclidiens $\mathcal{E}_1$ et $\mathcal{E}_2$ munis de produits scalaires définis positifs et deux applications $\psi_1 : \{\mathcal{X}\} \to \mathcal{E}_1$, $\psi_2 : \{\mathcal{X}\} \to \mathcal{E}_2$ tels que:*

$$D(x_i, x_j) = \|\psi_1(x_i) - \psi_1(x_j)\|^2_{\mathcal{E}_1} - \|\psi_2(x_i) - \psi_2(x_j)\|^2_{\mathcal{E}_2}$$

En notant $\psi = (\psi_1, \psi_2) : \mathcal{X} \times \mathcal{X} \to \mathcal{E} = (\mathcal{E}_1, \mathcal{E}_2)$, on peut toujours définir les prototypes comme des combinaisons convexes des observations projetées dans ce nouvel espace (Olteanu & Villa-Vialaneix, 2015) :

$$p_u(t) = \sum_{i=1}^{N} \omega_{u,i}(t)\psi(x_i)$$

où $\omega_{u,i}(t) \geq 0$ et $\sum_{i=1}^{N} \omega_{u,i}(t) = 1$. On pose $\omega_u = (\omega_{u,1}, \cdots, \omega_{u,N})^T$.

Similairement au cas du noyau, on pose que la distance $\|\psi(x_i) - p_u(t)\|^2_{\mathcal{E}}$ s'écrit au moyen de $\mathbb{D}$ et des ω_u par

$$\|\psi(x_i) - p_u(t)\|^2_{\mathcal{E}} = \mathbb{D}_i\omega_u(t) - \frac{1}{2}\omega_u(t)^T\mathbb{D}\omega_u(t).$$

L'étape d'affectation se fait comme pour le SOM numérique et s'en suit la mise à jour des prototypes de la même manière que pour le SOM Kernel.

Précisons ici que si les dissimilarités sont des distances euclidiennes au carré, alors le SOM relationnel est équivalent au SOM numérique.

4.2.4 Méthodes de classification/*clustering* de séries temporelles

Dans la littérature, de nombreuses méthodes de *clustering* de séries temporelles ont été développées dans divers domaines d'applications tels que la médecine, l'ingénierie, l'énergétique, l'économie, la finance, les sciences sociales, etc... Les séries temporelles sont regroupées par catégories essentiellement pour repérer des "patterns" (des évolutions du comportement) intéressants. On a deux types d'analyse : la première cherche les patterns les plus fréquents dans les séries temporelles et la seconde s'oriente plutôt vers la recherche de patterns non récurrents.

Les données composées de séries temporelles sont complexes par nature. Il est possible d'avoir des données manquantes, des fréquences différentes et surtout des longueurs différentes (ce qui est notre cas). Les méthodes de clustering de séries temporelles sont divisées en trois grandes catégories (Warren Liao, 2005), (Rani & Sikka, 2012) : les approches basées sur l'extraction de caractéristiques, les approches basées sur la proximité et les approches basées sur des modèles.

Les méthodes basées sur les caractéristiques : ces méthodes consistent à extraire un certain nombre (fixe) de caractéristiques des séries temporelles. L'avantage des techniques d'extraction de caractéristiques est que l'objet "complexe" série temporelle est représenté par un vecteur de dimension constante même si les séries sont de longueurs différentes. Quelques exemples sont énumérés:

- dans (Faloutsos & Faloutsos, 1994), Faloutsos utilise la transformation de Fourier discrète pour extraire des caractéristiques sur chaque série temporelle (en prenant les Q premiers coefficients du développement en série) et applique une méthode de *R-trees* (Beckmann, Kriegel, Schneider, & Seeger, 1990) pour le *clustering*.
- dans (Guo, Jia, & Zhang, 2008), Guo *et al.* utilisent une approche fondée sur la transformation des séries temporelles en vecteurs de caractéristiques par l'algorithme de l'analyse en composantes indépendantes (ICA) (Comon, 1994) et construisent le *clustering* de ces caractéristiques avec la méthode des centres mobiles.

Les méthodes basées sur la proximité : le but est de définir une notion de distance (ou de similarité) entre les séries temporelles et des les regrouper par classes avec une méthode de *clustering*. De nombreuses distances sont présentées dans la littérature (Keogh & Kasetty, 2003). : les séries de même longueur et les séries peuvent avoir des longueurs différentes. La plus connue est naturellement la distance euclidienne (Keogh & Kasetty, 2003) mais elle n'est pas toujours adaptée aux données. Nous présentons quelques autres mesures. On peut considérer ici deux cas :

- les séries ont la même longueur :

  - dans (Berndt & Clifford, 1994), Berndt opère la technique du *Dynamic Time Warping* (DTW) en utilisant la programmation dynamique pour détecter des patterns récurrents dans les séries temporelles;
  - dans (Iglesias & Kastner, 2013), Iglesias compare plusieurs distances: euclidienne, Mahalanobis (Mahalanobis, 1936) et celle calculée à partir du coefficient de corrélation de Pearson (Pearson, 1895). Pour construire les *clusters*, il utilise la méthode de *Fuzzy c-Means* (FCM) (Dunn, 1973);
  - dans (Kalpakis, Gada, & Puttagunta, 2001), Kalkapis définit une méthode de *clustering* de séries temporelles de type ARIMA. La distance euclidienne est utilisée et la méthode de *clustering* est l'algorithme de *Partition Around Medoïds* (Kaufman & Rousseeuw, 1987).

- les séries peuvent avoir des longueurs différentes :
  - dans (Yang & Leskovec, 2011), Yang *et al.* utilisent une dissimilarité qui est invariante en temps et en changement d'échelle pour calculer la proximité entre les séries temporelles. Ils appliquent ensuite la méthode de *clustering K-Spectral Centroid* sur ces données de dissimilarités pour étudier l'évolution de popularité de contenu publié en ligne (tels que des Tweets, etc...).
  - dans (Paparrizos & Gravano, 2017), Paparrizos *et al.* introduisent deux méthodes de clustering basées également sur des dissimilarités : *k-Shape* et *k-MultiShapes*. Il utilisent la même dissimilarité que dans (Yang & Leskovec, 2011).
  - dans (Camacho, Perez-Quiros, & Saiz, 2006), Camacho *et al.* décident tout simplement de tronquer toutes les séries à la longueur de la série la plus courte.
  - dans (Caiado, Crato, & Peña, 2009), Caiado *et al.* définissent une distance basée sur les périodogrammes. Les ordonnées du périodogramme sont calculées à partir des fréquences de Fourier et lorsque deux séries n'ont pas la même longueur, ces fréquences ne sont pas les mêmes, et donc ne sont pas comparables. Pour dépasser cette difficulté, il calcule les périodogrammes par paires en utilisant une fréquence commune entre les deux périodogrammes.

Les méthodes basées sur des modèles : ces méthodes supposent que chaque série temporelle est issu d'un modèle paramétrique. Les séries temporelles sont considérées similaires lorsque les paramètres du modèle caractérisant chaque série sont similaires :

- dans (Xiong & Yeung, 2004), Xiong *et al.* proposent une méthode pour classer des séries temporelles univariées. Ils supposent que les séries temporelles sont générées par k modèles ARMA différents et chaque modèle est associé à une classe. Ils utilisent l'algorithme EM pour estimer les paramètres du modèle.

- Gao *et al.* adoptent une approche similaire et l'appliquent à des données de vent dans (Gao, He, & Chen, 2009). D'abord, ils construisent un modèle ARMA basé sur les vitesses du vent puis testent ce qu'ils appellent "l'effet ARCH" (*Autoregressive Conditional Heteroscedasticity*) des résidus du modèle. Ils créent alors un modèle ARMA-ARCH qu'ils comparent à un modèle ARMA et présentent de meilleurs résultats.

- dans (Baragona, 2001), Baragona utilise des méthodes "meta-heuristiques" pour partitionner un ensemble de séries temporelles en différentes classes de manière à ce que le maximum des corrélations croisées entre les séries appartenant à une même classe soit supérieur à un certain seuil défini *a priori*.

4.3 Applications aux données de vol

Nous allons nous intéresser maintenant aux phases transitoires extraites dans le chapitre précédent. Introduisons quelques notations. Soit L le nombre de vols. On note chaque vol $Y_l = \{Y_{l,1}, \cdots, Y_{l,N_l}\}$ pour $l = 1, \cdots, L$, où l est le numéro du vol et N_l sa longueur. On note $Y_{l,u,v}$ le segment de Y_l délimité entre les indices u et v. Les K_l ruptures de chaque vol l sont notées $(\tau_{l,1}, \cdots, \tau_{l,k}, \cdots, \tau_{l,K_l})$ et on pose pour simplifier $\tau_{l,0} = 0$ et $\tau_{l,K_l+1} = N_l$. Les $K_l + 1$ tronçons des séries temporelles ainsi délimités sont notés $Y_{l,\tau_{l,k}+1,\tau_{l,k+1}}$ pour $l = 1, \cdots, L$ et $k = 0, \cdots, K^l$.

Après avoir identifié les phases croissantes, décroissantes et stabilisées, on ne tient plus compte de la position de la phase dans la série temporelle et chaque segment est maintenant calé au temps 1. Si A_l est le nombre de phases croissantes pour le vol l, on note $\mathbb{A} = \sum_{l=1}^{L} A_l$ le nombre total de phases croissantes dans les données. De même si B_l est le nombre total de phases décroissantes pour le vol l, on note $\mathbb{B} = \sum_{l=1}^{L} B_l$ le nombre de phases décroissantes dans les données. Afin de simplifier, l'ensemble des phases croissantes calées à 1 est noté $\{Z_{l,a}\}$, où $l = 1, \cdots, L$ et $a = 1, \cdots, A_l$ et l'ensemble des phases décroissantes calées à 1 $\{Z_{l,b}\}$, où $l = 1, \cdots, L$ et $b = 1, \cdots, B_l$.

Nous avons besoin d'une méthode très flexible et rapide pouvant regrouper au sein d'une même classe les phases transitoires ayant la même allure. Cette méthode doit être robuste et la qualité du *clustering* doit être validée autant par des calculs d'erreur que par l'avis des experts.

Dans un premier temps, nous classons les phases transitoires au moyen de la version relationnelle de SOM. Pour cela, nous définissons plusieurs dissimilarités.

4.3.1 SOM relationnel

L'approche relationnelle nous a semblé judicieuse étant donnée la nature de la base de données. Comme les phases transitoires sont de longueurs inégales, il semble intuitif

de passer par la définition d'une dissimilarité entre phases transitoires qui pourra ne pas tenir compte des longueurs de ces phases.

Mesure de dissimilarité

Plusieurs mesures de dissimilarité/similarité ont été calculées pour des séries temporelles de taille inégale. Dans (Berndt & Clifford, 1994), Rabiner *et al.* développent l'algorithme de *Dynamic Time Warping* qui permet de calculer la proximité entre deux séries temporelles. Ils cherchent l'appariement optimal entre deux séries, en utilisant des transformations non linéaires des séries pour déterminer une mesure de leur similarité. Nous avons testé cette méthode en nous bornant aux phases de décollage. Les résultats n'ont pas été prometteurs, car la déformation des phases transitoires amenait à les confondre avec des phases transitoires associées à des modes de vol différents.

Par la suite, nous avons testé deux dissimilarités pour créer les cartes de Kohonen. Nous introduisons d'abord la distance euclidienne glissante normalisée.

Soient Z_{l_1,a_1} et Z_{l_2,a_2} deux phases transitoires croissantes de tailles respectives L_{l_1,a_1} et L_{l_2,a_2}. Comme nous cherchons à regrouper les phases ayant le même comportement et que les phases sont de longueurs inégales, on cherche à faire correspondre les deux phases par glissement de manière à ce que leur "distance" soit la plus petite possible. On prend comme exemple deux phases de décollage (voir Figure 26-a courbe rouge et Figure 26-b la courbe bleue).

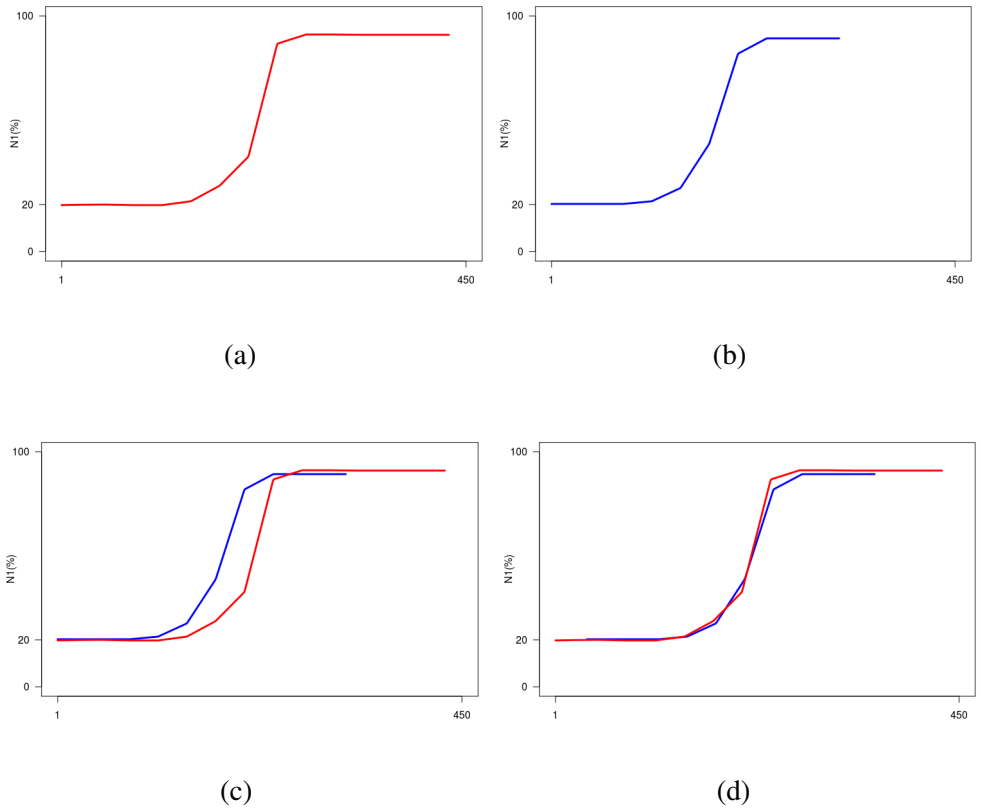


Figure 26: Diverses représentations de deux décollages

Ces deux phases se ressemblent mais la courbe rouge qui est plus longue que la bleue, commence sa "montée" après la bleue, se stabilise aussi après la bleue. C'est ce qu'on observe sur la Figure 26-c. Comme l'important est de déceler des similitudes de comportement, il est plus judicieux de calculer la distance en réalignant ces deux phases par rapport à leur pente (voir la Figure 26-c). Pour obtenir ce réalignement, on a glissé la plus petite des deux phases sur la plus grande. On a ensuite calculé à chaque étape de glissement de la courbe bleue vers la droite la distance euclidienne entre cette courbe et la plus grande rognée à gauche et/ou à droite de façon à égaliser les deux longueurs. La dissimilarité est alors calculée comme la plus petite distance entre les deux courbes au cours de ce glissement. C'est ce qu'on appelle une distance euclidienne glissante normalisée. Elle est similaire à la dissimilarité introduite par Yang *et al.* dans (Yang & Leskovec, 2011) mais nous n'utilisons pas de *scaling shifting* (changement en ordonné) et nous normalisons par la longueur de la courbe.

Pour formaliser tout cela, en supposant que $L_{l_2,a_2} > L_{l_1,a_1}$, la dissimilarité entre Z_{l_1,a_1} et Z_{l_2,a_2} s'écrit:

$$dis(Z_{l_1,a_1}, Z_{l_2,a_2}) = \min_{a=\{1,\cdots,L_{l_1,a_1}-L_{l_2,a_2}\}} \frac{\left\| Z_{l_1,a_1} - Z_{l_2,a_2,(a,a+L_{l_2,a_2})-1} \right\|_2}{2L_{l_1,a_1}} \qquad (20)$$

où $Z_{l_2,a_2,(i,j)}$ est le tronçon de Z_{l_2,a_2} entre les instants i et j.

On calcule les dissimilarités de toutes les phases transitoires croissantes prises deux à deux et on obtient une matrice de dissimilarités. Nous appliquons ensuite une méthode de *clustering* avec comme données d'entrée cette matrice.

Résultats pour les phases croissantes

A partir de la matrice de dissimilarités des phases transitoires croissantes calculée comme ci-dessus (20), on construit la carte de Kohonen à l'aide de l'algorithme SOM relationnel et on obtient la carte de la Figure 27.

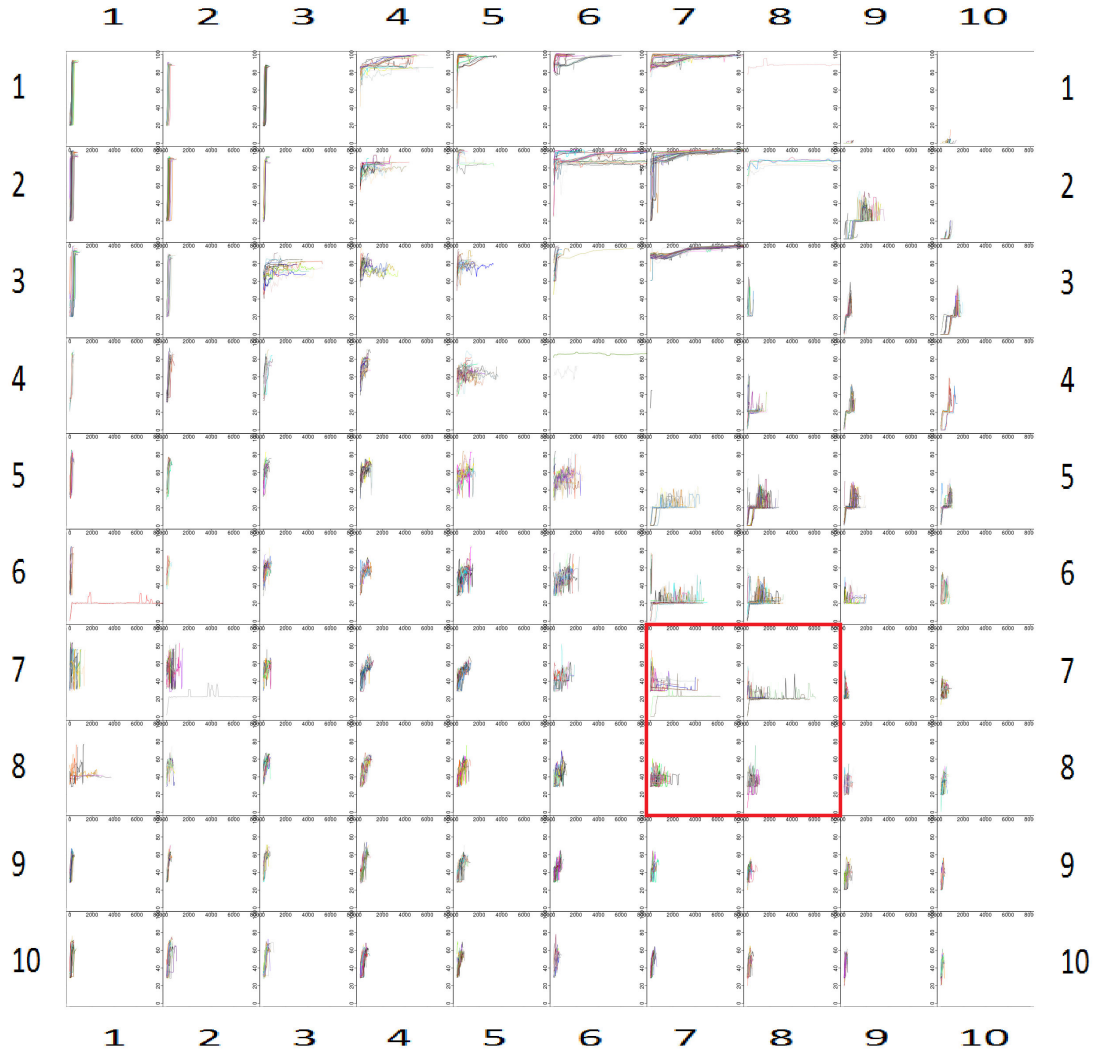


Figure 27: Réprésentation de la carte de Kohonen 10 x 10 des phases transitoires croissantes (SOM relationnel).

Nous avons choisi une grille de dimension 10 x 10 et utilisé la version du SOM relationnel fournie par le package R SOMbrero (Villa-Vialaneix et al., 2017). On observe que les classes sont bien réparties sur la carte. En haut à gauche, on retrouve des phases de décollage. En haut à droite, les phases de taxi sont nombreuses. Au milieu (en haut), on observe des phases de *climb* et quelques phases de croisière. Le reste de la carte représente essentiellement les phases de descente. Cependant quelques classes ne peuvent pas être validées par les experts comme on peut le remarquer sur la Figure 28.

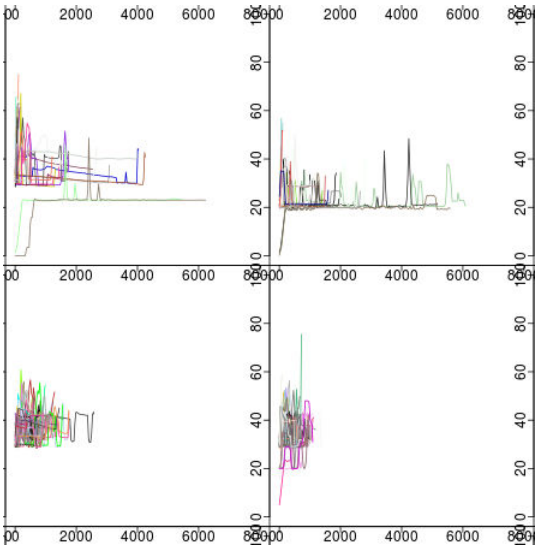


Figure 28: Zoom sur les 4 classes entourées en rouge de la carte de Kohonen obtenue Figure 27.

On note que dans la classe (7-7) une phase se déroulant pendant le mode taxi se trouve classée avec des phases de descente. Dans la classe (8-8), on observe une phase de décollage.

Une telle carte ne peut être exploitée par les experts et pour la suite de notre travail car des phases n'ayant pas le même comportement sont mélangées au sein d'une même classe.

Pour améliorer la classification, nous avons tenté de comprendre les défauts de la notion de dissimilarité introduite précédemment et d'y remédier. En fait, si la différence de longueurs entre deux phases est très grande, il peut arriver que la dissimilarité entre ces deux phases soit petite bien que les deux phases ne se ressemblent pas du tout. En effet une distance minimum très faible peut être atteinte pour un alignement bien précis de la plus courte sur la plus longue phase. La Figure 29 illustre ceci.

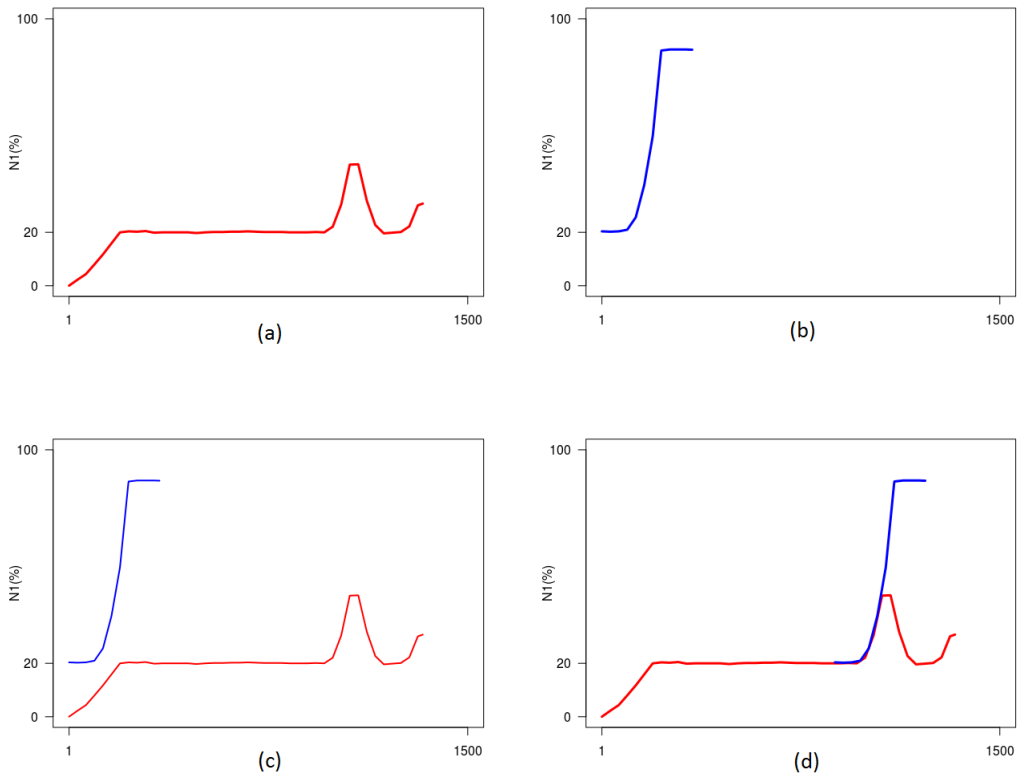


Figure 29: Réalignement de deux phases transitoires croissantes

Ces deux phases de taxi (courbe rouge) et de décollage (courbe bleue) ne se ressemblent pas du tout (Figure 29-c). Pourtant si on fait glisser la plus petite sur la plus grande, on obtient une dissimilarité proche de 0.

Pour contourner ce problème, nous introduisons une deuxième mesure de dissimilarités. On considère deux cas selon que les deux longueurs des deux séries sont proches ou très différentes. Pour cela, on définit un seuil α, et selon sa valeur, on considère que deux phases sont soit suffisamment proches pour appliquer la première dissimilarité, soit assez éloignés pour utiliser une autre dissimilarité.

Soient deux phases Z_{l_1,a_1} et Z_{l_2,a_2} de longueurs respectives L_{l_1,a_1} et L_{l_2,a_2}. On suppose que $L_{l_1,a_1} \leq L_{l_2,a_2}$ et on définit un seuil α.

Si $|L_{l_1,a_1} - L_{l_2,a_2}| < \alpha$ alors

$$\tilde{dis}(Z_{l_1,a_1}, Z_{l_2,a_2}) = \min_{a=\{1,\cdots,L_{l_1,a_1}-L_{l_2,a_2}\}} \frac{\left\| Z_{l_1,a_1} - Z_{l_2,a_2,(a,a+L_{l_2,a_2}-1)} \right\|_2}{2L_{l_1,a_1}} \tag{21}$$

où $Z_{l,a,(i,j)}$ est le tronçon de $Z_{l,a}$ pris entre les instants i et j.

Si $|L_{l_1,a_1} - L_{l_2,a_2}| \geq \alpha$ alors

$$\tilde{dis}(Z_{l_1,a_1}, Z_{l_2,a_2}) = \min_{a=\{1,\cdots,L_{l_1,a_1}-L_{l_2,a_2}\}} \frac{\left\| \tilde{Z}_{l_1,a_1,(a,L_{l_2,a_2}-L_{l_1,a_1}-a)} - Z_{l_2,a_2} \right\|_2}{2L_{l_2,a_2}}$$

où $\tilde{Z}_{l,a,(i,j)}$ est l'agrandissement de $Z_{l,a}$ afin que $\tilde{Z}_{l_1,a_1}$ ait la même taille que Z_{l_2,a_2}. Lors du glissement, s'il manque i points à gauche et j points à droite de la phase, on complète par les valeurs $Z_{l_1,a_1,1}$ à gauche et $Z_{l_1,a_1,L_{l_1,a_1}}$ à droite.

On calcule la nouvelle matrice de dissimilarités de toutes les phases transitoires croissantes prises deux à deux et on construit la carte de Kohonen par la méthode du SOM relationnel représentée Figure 30.

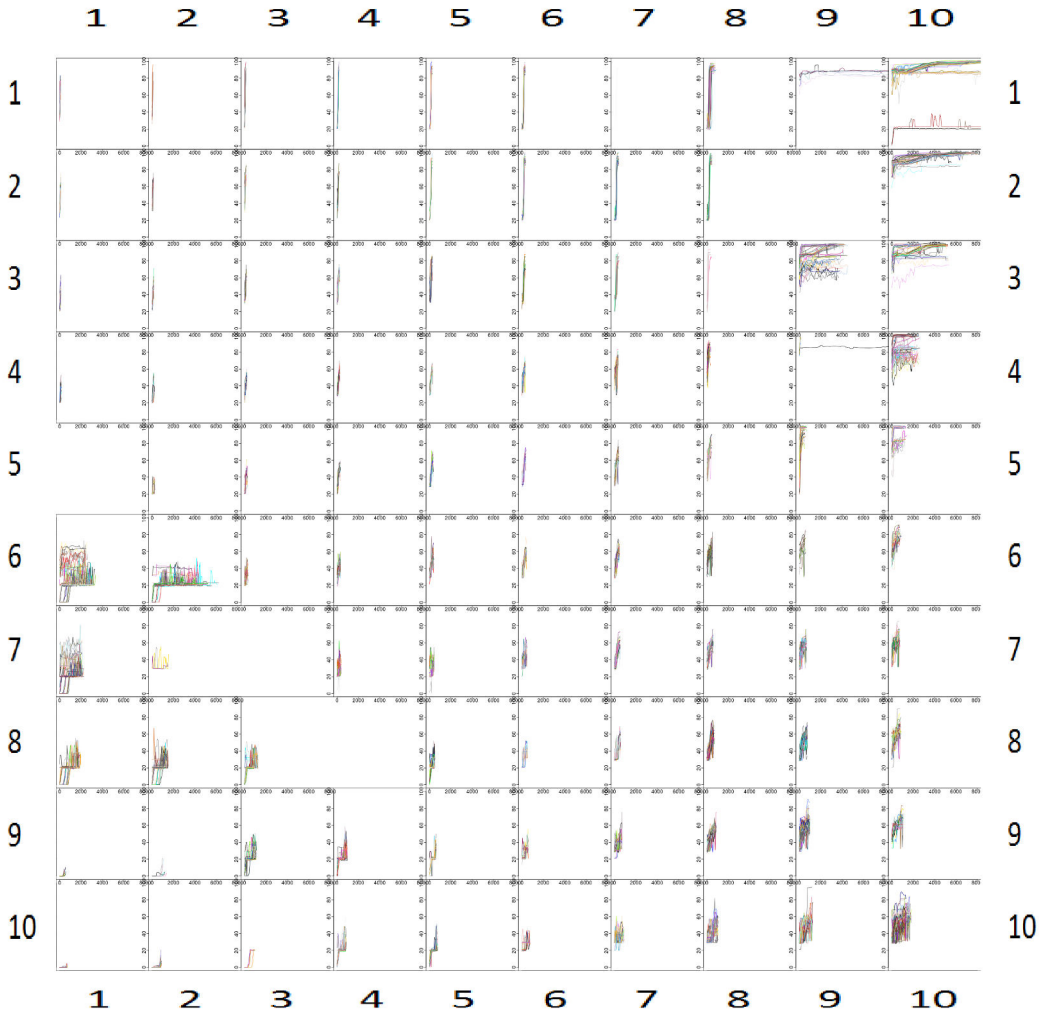


Figure 30: Réprésentation de la carte de Kohonen 10 x 10 des phases transitoires croissantes (SOM relationnel).

On observe que le *clustering* est meilleur que dans la première carte car l'allure des phases dans chaque classe est plus homogène (les phases de décollage sont bien classées avec les phases de décollage). Cependant il reste un gros problème dans la classe en haut à droite (1ère ligne et 10ème colonne). On y trouve des phases de *climb* mélangées avec des phases de *taxi*. Cela est dû à la normalisation appliquée dans le calcul de la dissimilarité (division par $2L$). Une telle classe n'est guère intéressante pour les experts, ni pour la suite de notre travail.

Conclusion

L'utilisation d'une méthode de *clustering* basée sur une dissimilarité ne fournit pas de résultats satisfaisants aux yeux des experts, puisqu'elle classe des phases de taxi dans les mêmes classes que des phases de *climb*. Aucune des dissimilarités testées ne semblent donc adaptées à nos données. Nous nous tournons donc vers une autre méthode.

4.3.2 SOM numérique

Les résultats précédents n'étant pas complètement satisfaisants, nous avons fait le choix alternatif d'utiliser une méthode fondée sur l'extraction des caractéristiques suivie d'un SOM classique.

Méthodologie

Cette approche nécessite d'associer à chaque phase transitoire un vecteur de caractéristiques.

L'idée est de "caractériser" chaque phase transitoire par un même nombre fini de valeurs numériques. On pose M_1 le nombre de caractéristiques numériques extraites des phases transitoires et $\mathbb{A}$ le nombre total de phases transitoires croissantes sur l'ensemble des vols. On note $X_{l,a} = (X_{l,a,m}, m = 1, \cdots, M_1)$, le vecteur des caractéristiques de la a-ème phase croissante du l-ème vol, où $l = 1, \cdots, L$, $a = 1, \cdots, A_l$.

L'ensemble des vecteurs $X_{l,a}$ sont ensuite regroupés via un SOM numérique. Une fois les classes de Kohonen construites, on calcule deux indicateurs de qualité, pour évaluer l'organisation (erreur topographique) et la quantification de la classification obtenue (erreur de quantification).

On cherche ensuite à croiser la variable Flight_Mod avec les classes pour identifier les étiquettes majoritaires de chaque classe et valider les résultats. Pour cela on utilisera l'identification ID_FM mentionnée dans la partie 3.4.

En général, on sait qu'il est préférable de définir des cartes de Kohonen de grande taille pour obtenir une bonne qualité de projection et des classes bien homogènes. Cependant, du point de vue de l'identification et de l'interprétation des classes, il est préférable de regrouper les classes de Kohonen en un petit nombre de super-classes. Grâce à une classification hiérarchique ascendante (CHA) des prototypes on définit le nombre nécessaire pour que le pourcentage de variance expliquée soit supérieur à 80%.

On peut également croiser les super-classes avec la variable Flight_Mod et utiliser à nouveau ID_FM pour valider les résultats.

En résumé, la méthodologie proposée ici consiste en par les étapes suivantes:

1. Extraire les caractéristiques pour chaque phase transitoire.
2. Classer les vecteurs de caractéristiques en utilisant l'algorithme SOM.
3. Calculer les erreurs topographique et de quantification pour qualifier le *clustering* obtenu.
4. Regrouper les classes de Kohonen en super-classes au moyen d'une classification ascendante hiérarchique ou d'un algorithme des centres mobiles appliqués aux prototypes.
5. Identifier les superclasses à l'aide de la variable Flight_Mod.

Résultats pour les phases transitoires croissantes

Chaque phase transitoire croissante est "caractérisée" par sa forme. Comme on l'a vu dans la partie 2.4.2, le décollage fait un saut rapide de 20% à presque 80%, le *climb* est défini par une lente accélération entre 80% et 90% etc... C'est pourquoi nous choisissons avec les experts des caractéristiques liées à la forme de la courbe. La première valeur en ordonnée, la dernière valeur en ordonnée et la longueur permettent d'identifier les sauts (lents ou rapides) dans le temps (pour les décollages, les *climbs* et les *taxis*). La variance, la valeur atteinte à mi-parcours, qu'on appellera valeur du milieu, et la médiane sont utiles pour identifier essentiellement les descentes et quelque fois les croisières. On calcule également les variances et les moyennes des demi-courbes situées de part et d'autre du milieu.

Ces dernières caractéristiques apportent des précisions sur l'évolution du comportement des phases transitoires croissantes. Le but est que le choix de ces caractéristiques permette d'éviter les problèmes rencontrés avec la méthode directe des dissimilarités, c'est-à-dire d'éviter que des phases de croisière et de taxi ne se trouvent dans une même classe. Pour cela la valeur de début et la valeur de fin sont deux caractéristiques qui permettront de distinguer ces deux types de phases.

En résumé les 10 caractéristiques retenues sont :

- Valeur de début
- Valeur de fin
- Valeur du milieu
- Longueur de la phases
- Médiane
- Variance de la phase entière
- Variance à gauche
- Variance à droite
- Moyenne à gauche
- Moyenne à droite

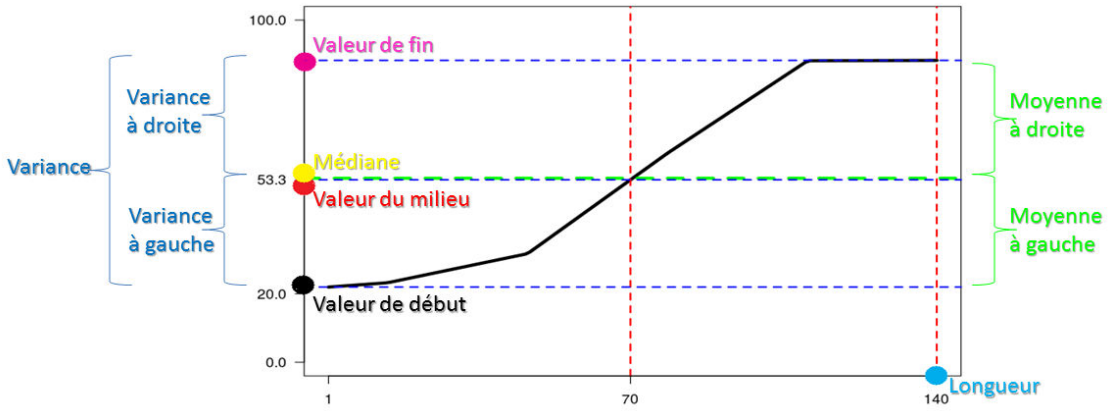


Figure 31: Liste des caractéristiques choisies.

On illustre dans la Figure 31 chaque caractéristique énumérée dans la liste précédente.

Ces caractéristiques décrivent de manière simplifiée les formes des différentes phases transitoires croissantes, qui peuvent être vues comme des courbes linéaires par morceaux. Voir les Figures en fin de section 3.4.

Plusieurs tailles de carte de Kohonen sont testées. Le nombre de phases transitoires croissantes est très important (4351 phases croissantes), c'est pourquoi nous avons testé les tailles suivantes: 11x11, 12x12 et 50x50. Nous présentons la carte 11x11.

Etant donné que l'algorithme SOM est stochastique, nous l'avons exécuté 200 fois et deux types d'erreurs sont calculées pour chaque carte. On obtient en moyenne: erreur de quantification (0.052 [0.033]) et erreur topographique (0.064 [0.012]). Les erreurs calculées pour chaque carte sont petites (l'écart-type est également petit). Cela montre une carte très stable et une classification homogène.

On sélectionne ensuite la carte ayant la plus petite erreur de quantification (erreur de quantification : 0.051) et son erreur topographique est de 0.062 ce qui est satisfaisant.

Les résultats sont calculés par la version du SOM numérique fournie par le package R SOMbrero (Villa-Vialaneix et al., 2017). Toutefois la fonction initiale de visualisation de la carte ne permet pas de projeter des courbes de longueurs inégales dans les classes de la carte. Un travail de visualisation très poussé a été réalisé pour pouvoir observer les phases sur une carte de Kohonen. Les codes initiaux ont été modifiés de manière à pouvoir projeter des courbes de longueurs inégales sur la carte tout en gardant la même échelle pour pouvoir les comparer (les difficultés informatiques sont détaillées en annexe).

La Figure 32 présente la carte 11x11 sélectionnée. Les contributions des 10 composantes des vecteurs de caractéristiques sont représentées sur la Figure 34.

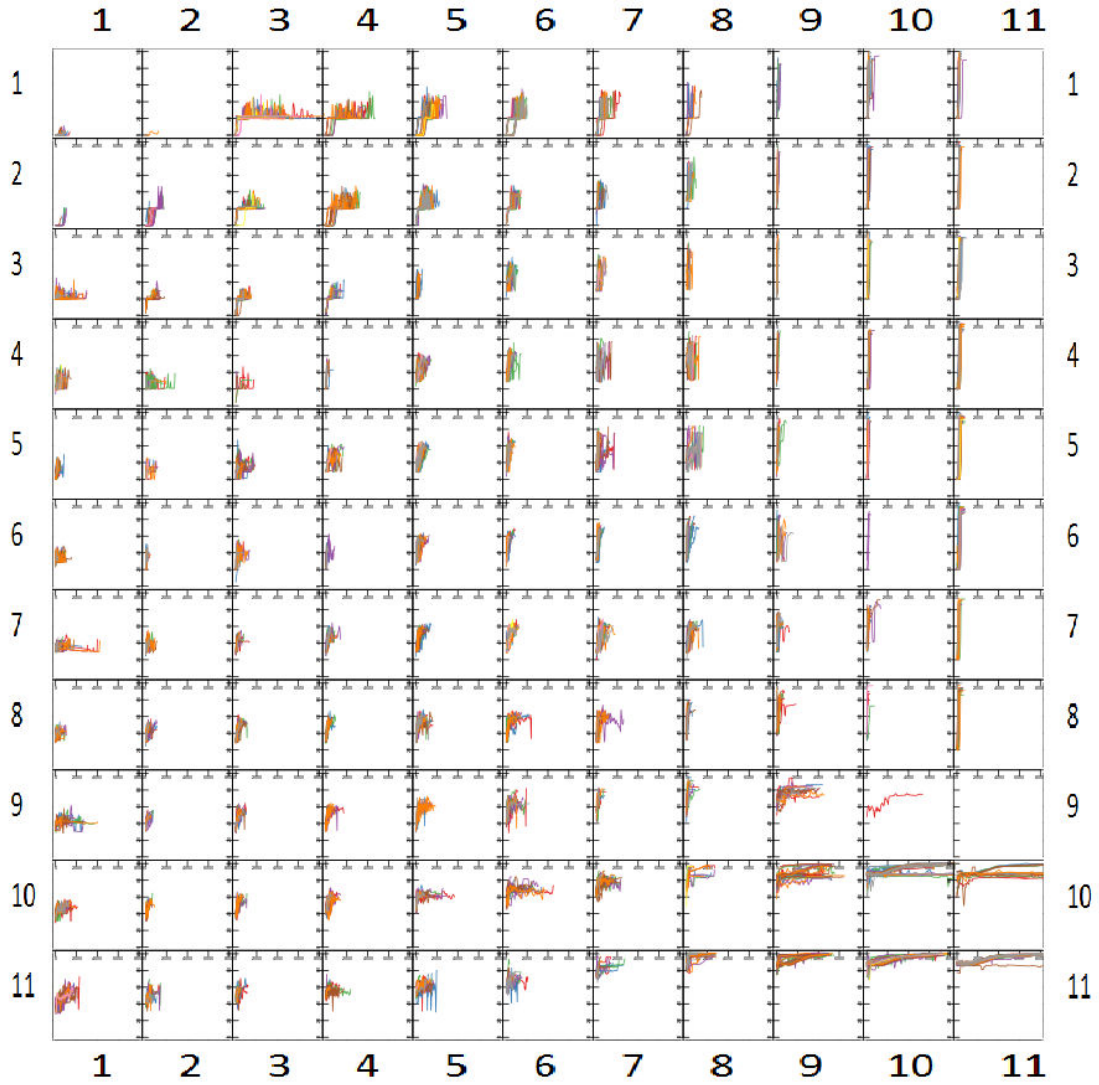


Figure 32: SOM numérique 11x11 des phases croissantes.

Dans la Figure 32, on constate que la plupart des classes sont bien homogènes, mais que les classes situées tout en bas à droite de la carte contiennent des phases d'allures assez différentes. Certaines phases semblent être des *climbs* et d'autres phases ont l'air d'être des croisières. A l'oeil nu, les classes en haut à gauche de la carte semblent être des *taxis* et en haut à droite les classes ressemblent fortement à des décollages. Le reste de la carte est difficilement identifiable à partir de ces visualisations uniquement.

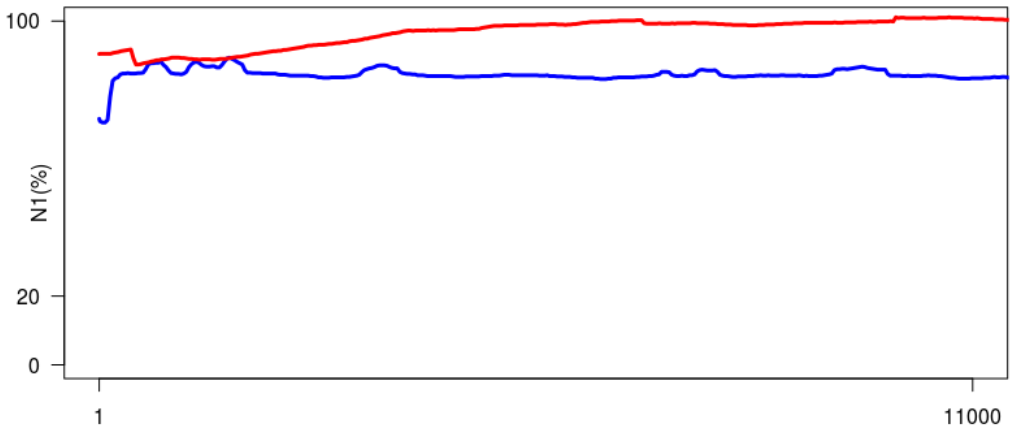


Figure 33: Représentation graphique de deux phases transitoires croissantes appartenant à la classe (10,11) de la carte de la Figure 32

Dans la Figure 33, on retrouve un exemple de deux phases n'ayant pas la même forme mais appartenant à la même classe (10,11). On peut deviner que la courbe rouge est un *climb* et la courbe bleue est une croisière.

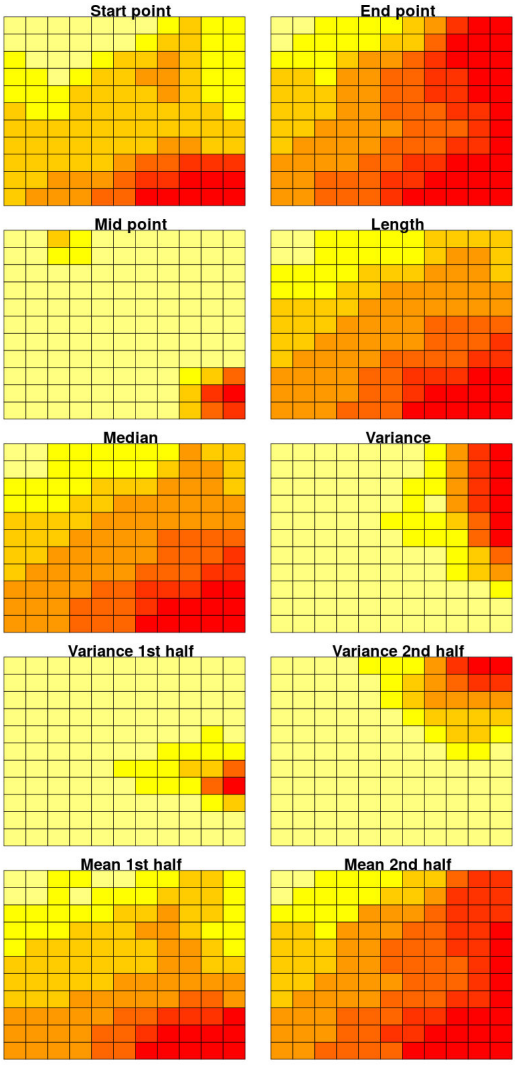


Figure 34: Représentation de la contribution de chaque caractéristique des phases croissantes. L'échelle de couleur va du rouge foncé (valeur maximum) au jaune clair (valeur minimum).

On remarque (Figure 34) que six variables sont particulièrement discriminantes : la valeur de début, la valeur de fin, la longueur, la médiane et les moyennes des première et deuxième moitiés. Les variances sont surtout significatives dans les classes en haut à droite en comparant avec la carte de la Figure 32. Le point du milieu est à son maximum dans les classes en bas à droite.

On cherche à identifier les phases transitoires dans chaque classe. Pour cela on utilise les étiquettes ID_FM définies dans la section 3.4. Rappelons que chaque phase transitoire croissante correspond à un certain nombre de modes de vol (1 à 3 modes maximum par phase). Elle peut donc être codée par une suite d'entiers représentant les modes de vol, c'est ce qu'on appelle l'étiquette ID_FM. Chaque classe est identifiée par l'étiquette ID_FM majoritaire parmi l'ensemble des phases qu'elle

contient.

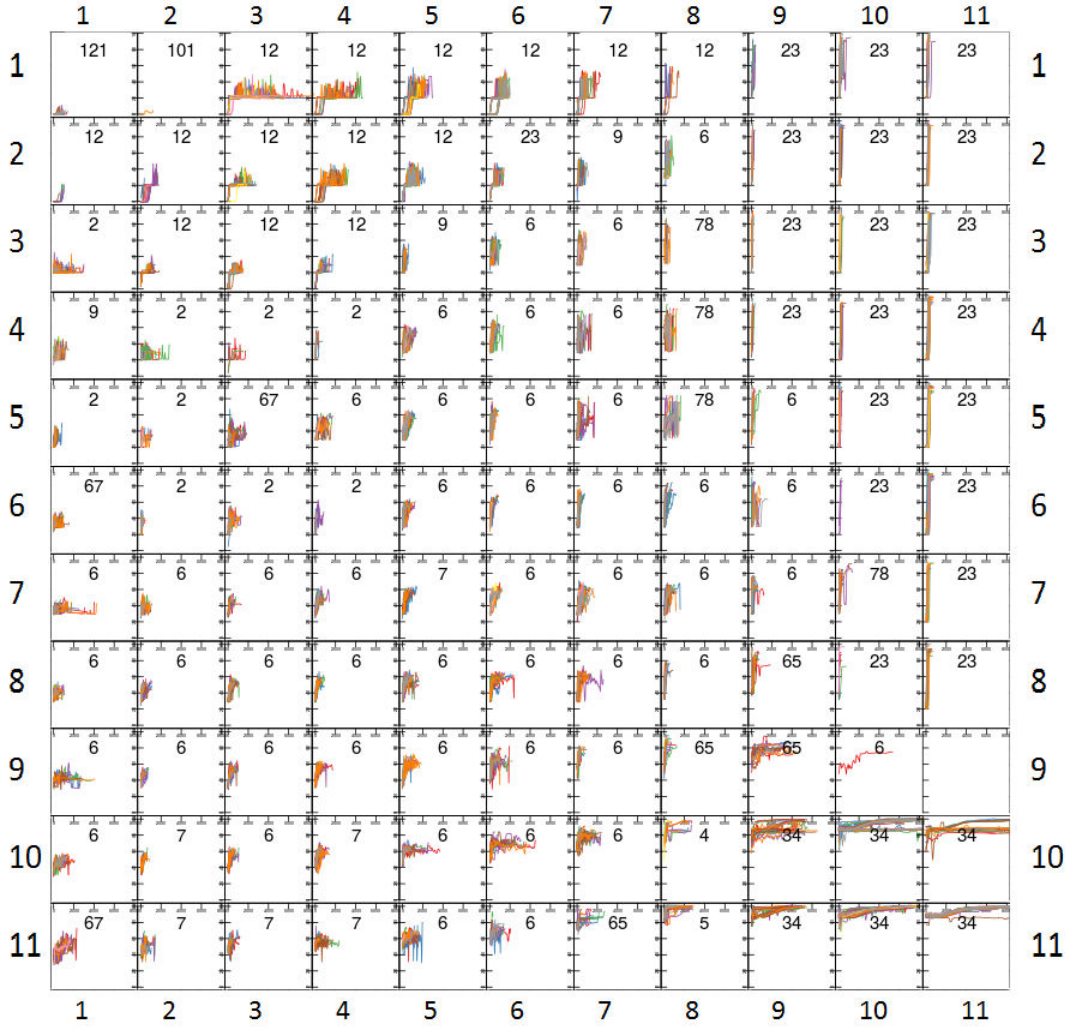


Figure 35: SOM numérique 11x11 des phases transitoires croissantes avec l'ID_FM majoritaire affichée par classe.

Dans la Figure 35, on observe que les classes en haut à gauche sont composées de phases transitoires croissantes se déroulant principalement en mode démarrage moteur (101 et 121) et de *taxi in* et *out* (12, 23 et 9). Les classes en haut à droite regroupent des phases se déroulant durant le mode de décollage (23). En bas à droite de la carte, on retrouve des phases de *climb* (34). Le reste de la carte est formé de classes comprenant une majorité d'étiquettes de descente (6, 67 et 7). On constate aussi que les classes obtenues sont bien homogènes.

On cherche maintenant à réduire le nombre de clusters pour une meilleure description. Pour cela on réalise une classification hiérarchique ascendante des prototypes de la carte de Kohonen et on étudie le pourcentage d'inertie expliquée en fonction du nombre de classes. Ce pourcentage est représenté dans la Figure 36.

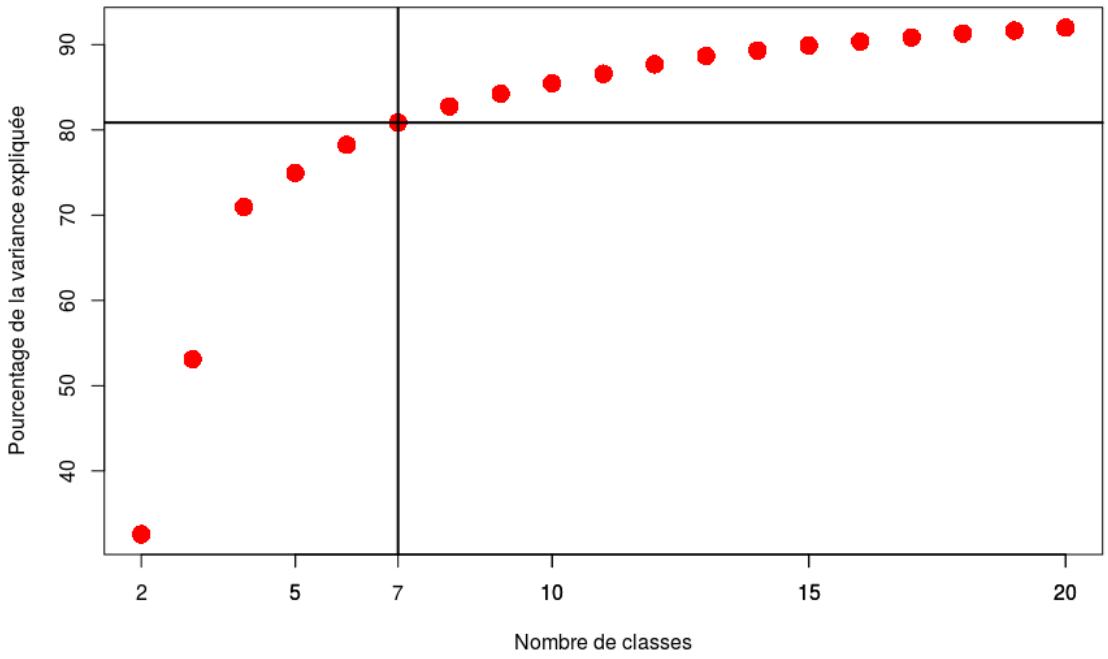


Figure 36: Pourcentages de l'inertie expliquée des phases transitoires croissantes obtenue par la méthode CHA en fonction du nombre de classes.

En tenant compte de ces résultats, 80% de l'inertie expliquée correspond à 7 superclasses. Chaque superclasse est distinguée par une couleur de fond sur la carte de Kohonen Figure 37.

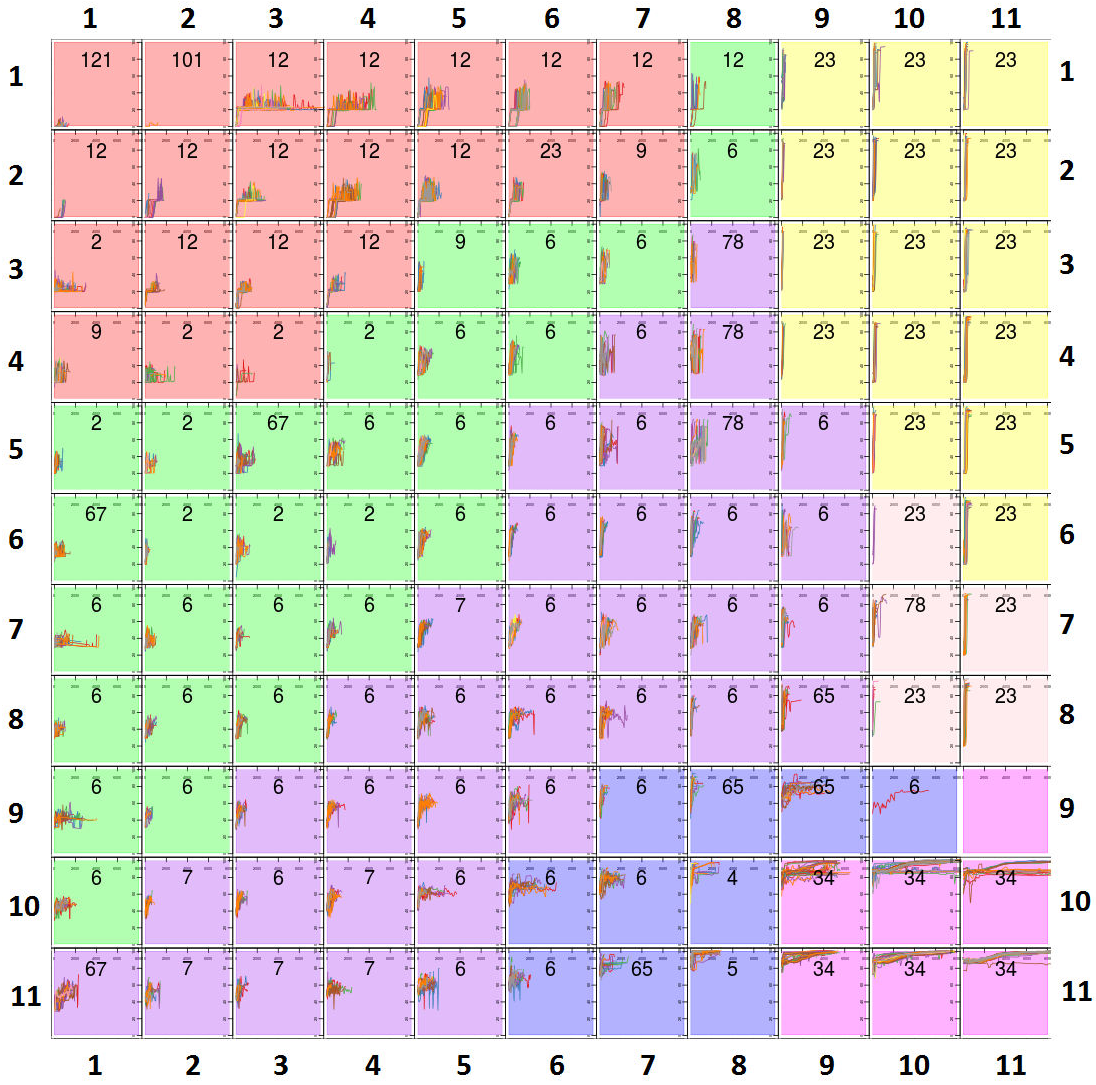


Figure 37: Carte de Kohonen 11x11 des phases croissantes, l'ID_FM majoritaire est affichée dans chaque classe. La couleur de fond représente l'appartenance à l'une des 7 superclasses.

On remarque que deux superclasses colorées en jaune et en rose clair regroupent des phases de décollage et on remarque que la superclasse rose claire contient également des phases numérotées 78 qu'on décrira plus tard. Trois superclasses sont composées essentiellement de phases de descente (violet, bleu et vert). La superclasse de taxis est de couleur rouge. En rose, la superclasse regroupe majoritairement les phases de *climb*.

Description des superclasses de phases transitoires croissantes

Chaque superclasse peut être décrite également par des statistiques descriptives : la moyenne et l'écart-type de 6 autres variables (altitude - ALT, température d'huile - OIL TEMP, vitesse de rotation haute pression - N2, pression extérieure - P0, température avant compression - T12 et température avant combustion - T3) mesurées sur les mêmes intervalles que les phases transitoires croissantes. Nous n'avons pas pu utiliser toutes les variables de la Table 1 car certaines étaient indisponibles pour quelques vols. Les résultats sont présentés dans la Table 7.

On note $CC_1, \cdots, CC_7$ les 7 superclasses des phases transitoires croissantes. Dans la Figure 38 sont représentées les correspondances entre les $CC_1, \cdots, CC_7$ et les couleurs de la Figure 37.

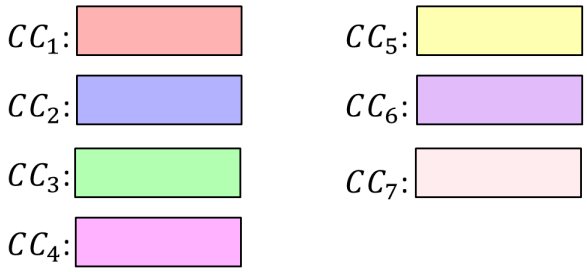


Figure 38: Correspondances entre les $CC_1, \cdots, CC_7$ et les couleurs de la Figure 37.

- Pour la superclasse CC_1, les valeurs moyennes de l'altitude (ALT), de la vitesse de rotation (N2) et de la température avant combustion (T3) sont petites. La température d'huile (OIL TEMP) est inférieure à la moyenne générale alors que la pression (P0) et la température avant compression (T12) sont plus élevées que la moyenne générale.
- Pour la superclasse CC_2, l'altitude et la température T3 sont très élevées alors que la température T12 est très en-dessous de la moyenne.
- Pour la superclasse CC_3, les valeurs moyennes de P0 et de T12 sont élevées et l'altitude ALT est faible.
- Pour la superclasse CC_4, la température T3, l'altitude et les vitesses N1 et N2 sont très élevées. La pression atmosphérique est proche de la moyenne générale.
- Pour la superclasse CC_5, la température T12 et la pression P0 sont élevées. On remarque que l'altitude est assez faible.
- Pour la superclasse CC_6, la température d'huile est très haute.

	N1	ALT	OIL TEMP	N2	P0	T12	T3
CC_1	13.04	2857.94	88.01	36.69	10.11	14.38	71.85
EC	10.42	4192.93	44.56	27.58	6.12	8.88	103.86
CC_2	82.78	**27638.38**	93.58	71.91	5.18	-6.20	398.35
EC	12.20	10416.13	9.83	38.48	2.94	20.24	63.05
CC_3	39.51	4386.15	98.72	53.56	12.79	19.15	241.75
EC	5.74	5299.62	14.74	37.03	2.31	9.02	38.04
CC_4	**93.85**	22980.80	85.61	**91.32**	7.60	4.86	**494.04**
EC	3.44	7782.39	7.21	22.23	1.96	12.03	36.08
CC_5	54.29	5157.96	92.79	83.07	**14.31**	**19.53**	249.48
EC	3.14	8734.35	11.72	9.93	0.91	7.11	40.58
CC_6	52.86	7409.59	**100.73**	47.96	11.41	15.26	293.95
EC	5.47	8339.59	10.11	42.35	3.31	12.80	33.59
CC_7	67.18	6484.79	95.29	76.39	12.93	17.68	333.21
EC	8.53	9669.93	12.08	32.92	3.34	9.65	50.32
Moy. générale	44.90	24610.30	94.27	66.15	6.59	-0.95	355.45
ANOVA							
F value	19319	1495	232	8126	306586	905	20053
P value	4.4e-3	1.2e-2	4.6e-2	7.1e-3	1.5e-3	2.3e-2	4.1e-3

Table 7: Moyennes, écarts-types [EC] et ANOVA des 6 variables supplémentaires selon les 7 superclasses de phases transitoires croissantes. Les valeurs les plus faibles sont soulignées et les valeurs les plus fortes sont écrites en gras.

- Pour la superclasse CC_7, l'altitude est faible et la température T12 est proche du maximum.

Une ANOVA est réalisée sur les superclasses et les résultats sont présentés dans la table 7. Ils montrent que les superclasses sont significatives d'un point de vue statistique.

L'identification des 7 superclasses se fait ensuite au moyen des étiquettes ID_FM des phases transitoires croissantes qu'elles contiennent. Nous ne présenterons que deux superclasses de phases transitoires croissantes mais les autres superclasses sont disponibles en annexe.

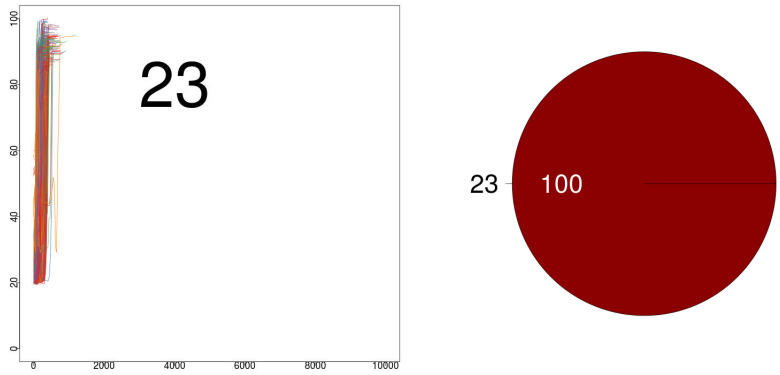


Figure 39: Représentation de la superclasse 5 des phases croissantes et le diagramme en camembert des modes de vol la composant.

La superclasse 5 n'est composée que de phases de décollage (23).

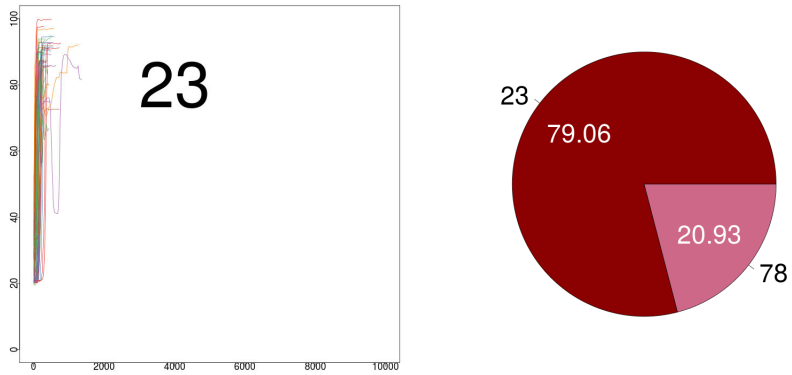


Figure 40: Représentation de la superclasse 7 des phases croissantes et le diagramme en camembert des modes de vol la composant.

La dernière superclasse des phases transitoires croissantes contient 80% de phases de décollage. Les phases d'approche de cette classe correspondent à des instants où le pilote remet les gaz.

4.3.3 Résultats pour les phases transitoires décroissantes

Nous appliquons maintenant la même approche sur les phases transitoires décroissantes (200 exécutions, même taille de grille, etc...). On extrait les mêmes caractéristiques que précédemment des phases transitoires décroissantes et on classe les vecteurs de caractéristiques en utilisant l'algorithme SOM sur une carte de Kohonen 11x11.

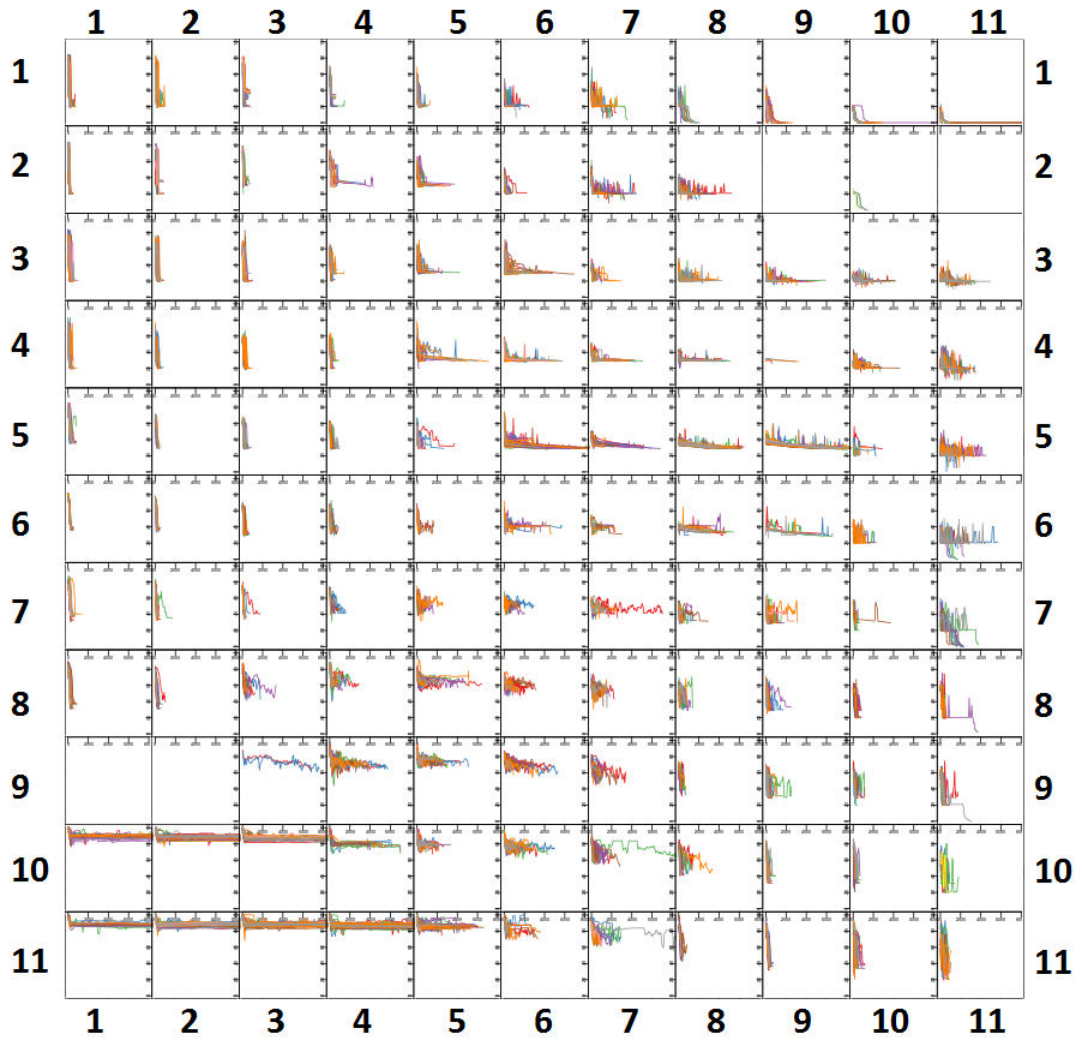


Figure 41: Carte de Kohonen 11x11 des phases décroissantes.

Dans la Figure 41, on représente la carte de Kohonen pour laquelle les erreurs de quantification et topographique sont les plus faibles. Les classes sont bien homogènes. On observe un peu plus de classes vides que dans la carte de la Figure 32.

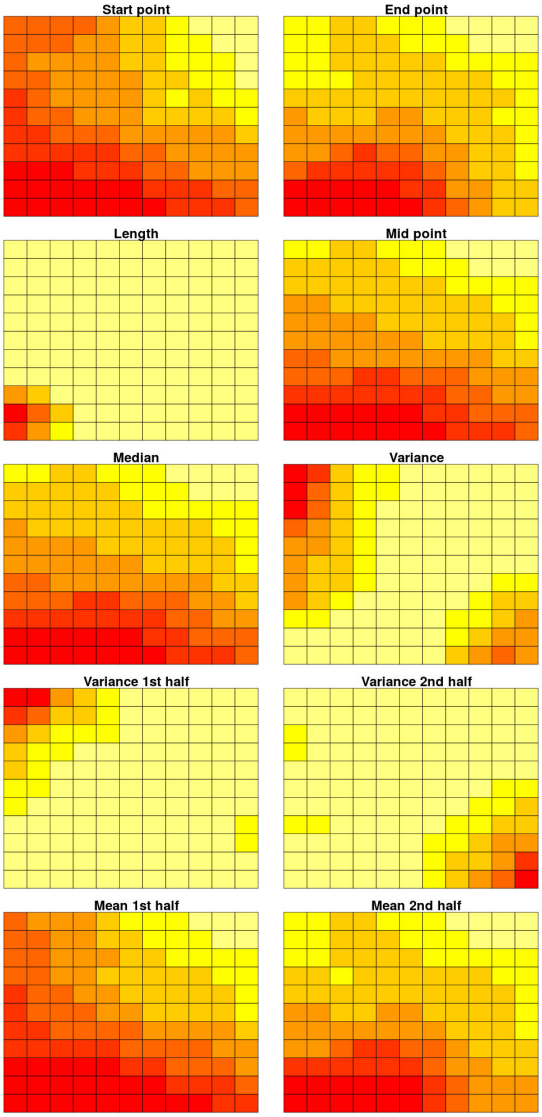


Figure 42: Représentation de la contribution de chaque caractéristique des phases décroissantes. L'échelle de couleur va du rouge foncé (valeur maximum) au jaune clair (valeur minimum).

Dans la Figure 42, on remarque que six variables sont particulièrement discriminantes: la valeur de début, la valeur de fin, la longueur, la médiane, la variance à gauche et la variance à droite. Les variances sont significatives pour les classes situées en haut à gauche et en bas à droite. La longueur est importante dans les classes situées en bas à gauche.

On identifie chaque classe par l'étiquette ID_FM majoritaire parmi les phases qu'elle contient.

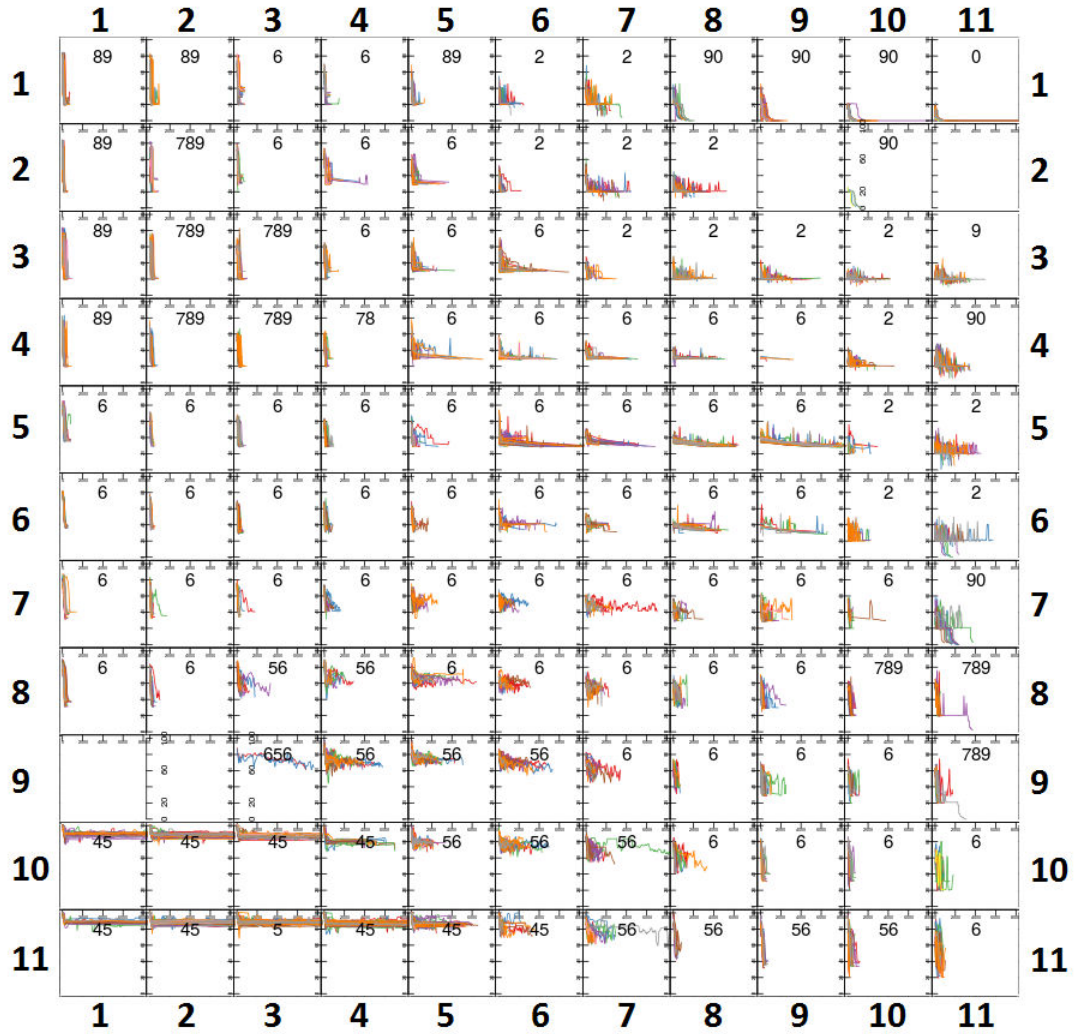


Figure 43: Carte de Kohonen 11x11 des phases transitoires décroissantes avec l'ID_FM majoritaire affichée par classe.

Sur la carte de la Figure 43, on observe en bas à gauche des phases transitoires croissantes se déroulant principalement en fin de climb (45) /croisière (56). En haut à droite, on retrouve des phases de taxis *in* (90) et *out* (0 et 2). En haut à gauche, les classes sont composées de phases d'approches (89 et 789) et le reste de la carte représente des classes contenant des phases de descente (6 et 56).

Pour déterminer le nombre de superclasses à construire, on calcule les pourcentages d'inertie expliquée en fonction du nombre de classes, en appliquant la méthode de classification hiérarchique ascendante aux prototypes de la carte de Kohonen.

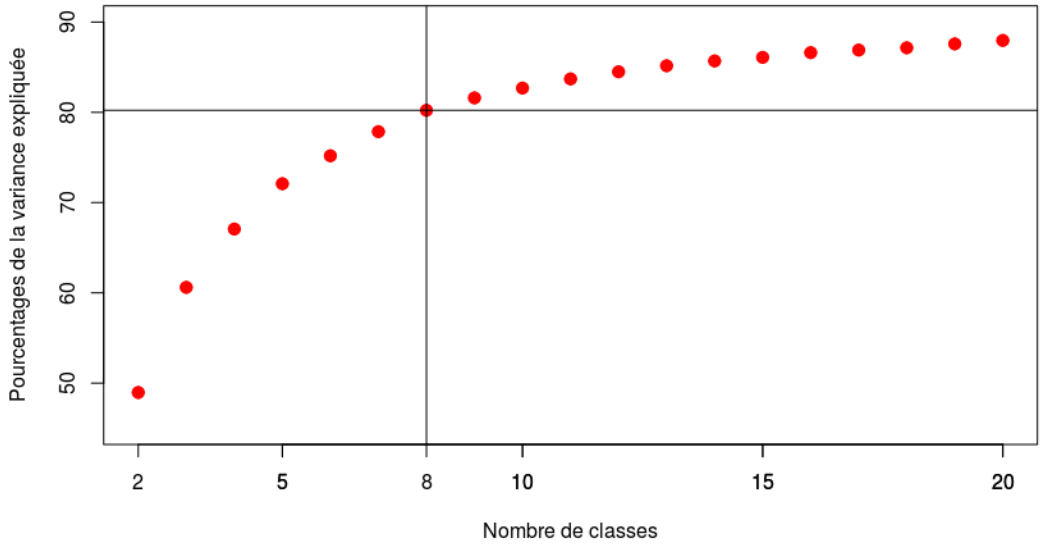


Figure 44: Pourcentages de l'inertie expliquée obtenue par la méthode CHA en fonction du nombre de classes

Cette fois, on fixe à 8 le nombre de superclasses, ce qui correspond à environ 80% d'inertie expliquée.

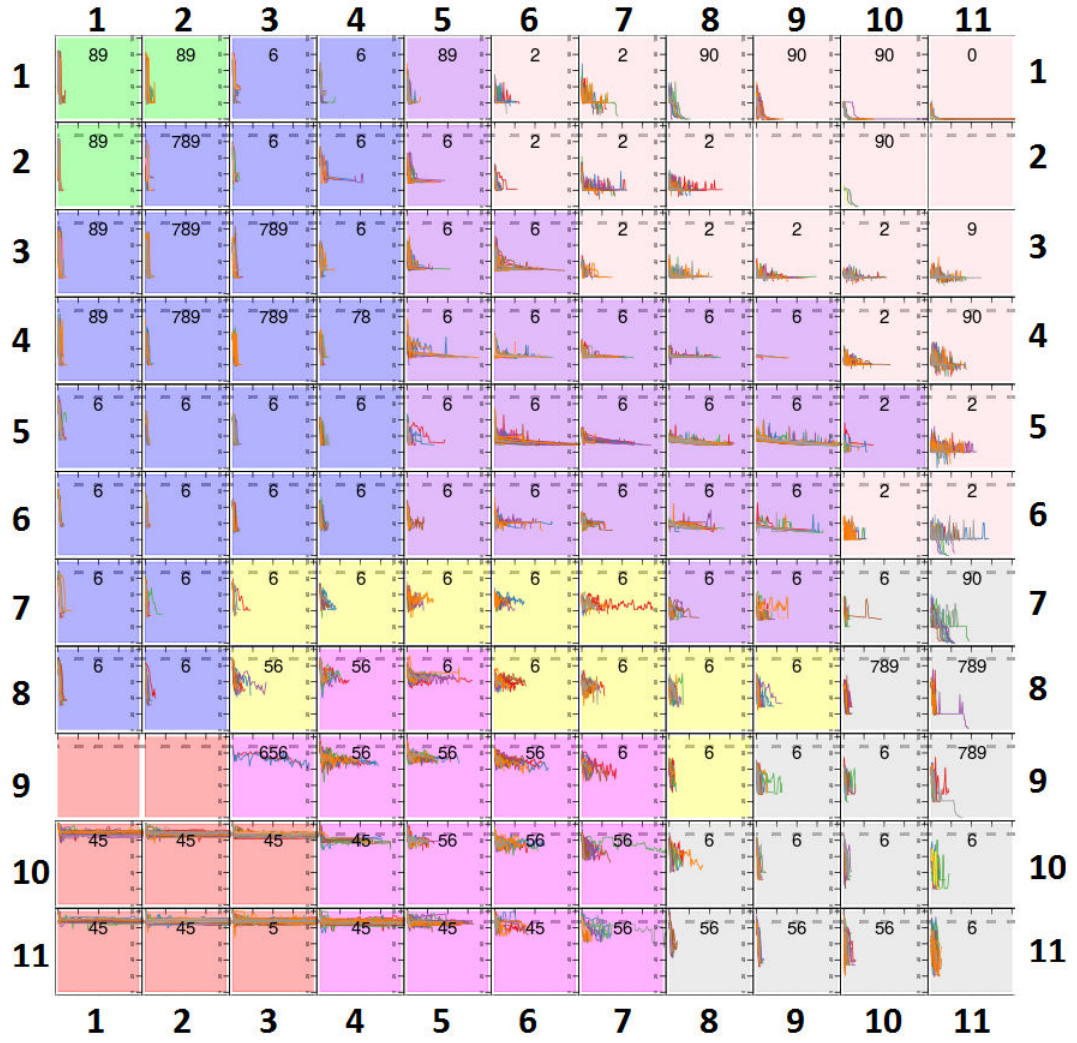


Figure 45: Carte de Kohonen 11x11 des phases transitoires décroissantes, l'ID_FM majoritaire est affichée dans chaque classe. La couleur de fond représente l'appartenance à l'une des 8 superclasses.

On observe deux superclasses qui ne sont composées que de phases de descente (jaune clair et violet), mais les phases des classes de couleur jaune clair sont plus courtes. La superclasse rose claire est composée quant à elle de phases de taxi *in* (l'avion sort de l'aérogare) et *out* (l'avion rentre dans l'aérogare). La superclasse rouge (en bas à gauche) contient des phases composées de fin de *climb* et début de croisière (les phases sont longues et stabilisées). En fushia, les classes sont composées de phases de croisière et début de descente. La superclasse bleue est homogène et est composée de descentes et de phases d'approche. La superclasse grise est plus hétérogène que les autres superclasses.

Description des superclasses de phases transitoires décroissantes

On note $CD_1, \cdots, CD_8$ les 8 superclasses des phases transitoires décroissantes. Dans la Figure 46 sont représentées les correspondances entre les $CD_1, \cdots, CD_8$ et les couleurs de la Figure 45.

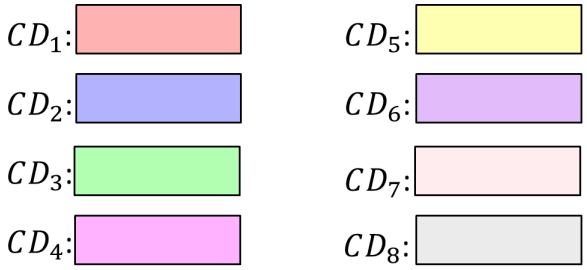


Figure 46: correspondances entre les $CD_1, \cdots, CD_8$ et les couleurs de la Figure 45.

On calcule pour chaque superclasse les moyennes et écarts-types de 6 variables: l'altitude ALT, la température d'huile OIL TEMP, la vitesse de rotation N2, la pression P0, la température avant combustion T12 et la température avant combustion T3. Les résultats sont présentés dans la Table 8 :

- Pour la superclasse CD_1, plusieurs mesures sont très élevées: les vitesses de rotation N1 et N2, l'altitude ALT et la température avant combustion T3. La pression P0 et la température avant combustion T12 sont quant à elles très faibles.
- Pour la superclasse CD_2, la vitesse de rotation N2 est peu élevée.
- Pour la superclasse CD_3, l'altitude et la vitesse de rotation N2 sont très faibles ainsi que la température d'huile.
- Pour la superclasse CD_4, l'altitude est haute et la température T12 est assez basse.
- Pour la superclasse CD_5, l'altitude est assez basse.
- Pour la superclasse CD_6, la température T12 est assez grande.
- Pour la superclasse CD_7, la vitesse de rotation N1 est très faible alors que la pression est très élevée.
- Pour la superclasse CD_8, toutes les valeurs des variables sont très proches de celles de la superclasse CD_5.

	N1	ALT	OIL TEMP	N2	P0	T12	T3
CD_1	**87.32**	**37023.96**	95.74	**75.59**	3.15	-13.76	**411.19**
EC	1.77	1558.23	5.83	22.50	0.25	22.04	17.15
CD_2	45.68	12934.96	95.59	37.24	9.22	7.70	308.28
EC	7.87	12244.19	9.35	38.34	4.61	18.14	36.10
CD_3	37.69	1721.34	90.65	16.80	14.35	**19.29**	309.80
EC	4.26	1548.31	7.07	30.32	0.32	8.01	32.64
CD_4	78.22	28073.95	92.00	67.09	4.60	-7.03	390.89
EC	7.24	9375.83	9.24	39.02	2.33	19.62	41.74
CD_5	53.76	12011.54	98.13	58.09	9.28	9.20	323.43
EC	5.57	10676.81	10.51	38.01	3.97	16.34	31.29
CD_6	34.63	8810.70	**104.17**	43.68	10.19	14.47	238.64
EC	4.35	6410.74	10.68	36.42	2.77	9.29	34.04
CD_7	15.65	2477.59	92.24	33.47	**14.43**	18.56	160.20
EC	8.94	4136.32	20.67	29.12	1.15	6.72	40.64
CD_8	50.27	12227.22	94.20	43.91	9.90	9.09	322.83
EC	12.61	12860.24	10.30	39.53	4.70	18.14	50.53
Moy. générale	43.19	24610.30	94.27	66.15	6.59	-0.95	355.45
ANOVA							
F value	327	57	91	174	129	194	402
P value	3.6e-2	8.4e-2	7.1e-2	5.0e-2	6.1e-2	4.3e-2	3.8e-2

Table 8: Moyennes, écarts-types [EC] et ANOVA des 6 variables supplémentaires selon les 8 superclasses de phases transitoires croissantes. Les valeurs les plus faibles sont soulignées et les valeurs les plus fortes sont écrites en gras.

Une ANOVA est réalisée sur les superclasses et les résultats sont présentés dans la Table 8. Si on fixe un niveau de 5%, on remarque que trois des tests ne sont pas significatifs. Par ailleurs, de manière générale, les statistiques de test sont beaucoup plus faibles que dans le cas des phases croissantes et les *p-values* sont plus élevées. Ceci signifie que les variables d'environnement étudiées sont beaucoup moins discriminantes pour les superclasses des phases transitoires décroissantes.

L'identification des 8 superclasses se fait au moyen des étiquettes ID_FM des phases transitoires décroissantes qu'elles contiennent. Nous ne présentons que deux superclasses mais les autres superclasses sont fournies en annexe.

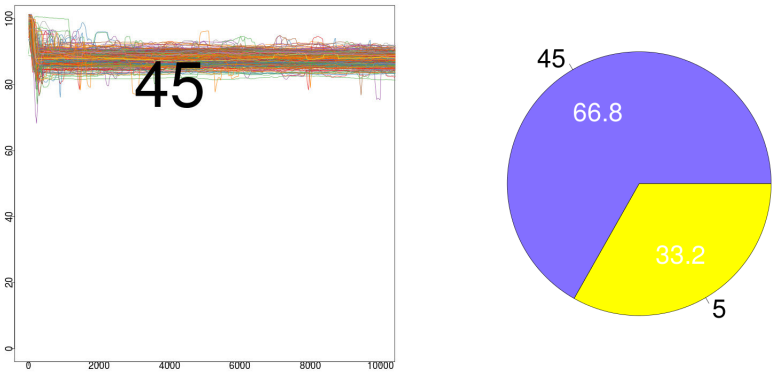


Figure 47: Représentation de la superclasse 1 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La première superclasse est composée de phases de croisière (45 et 5). C'est pourquoi dans la Table 8, l'altitude est élevée.

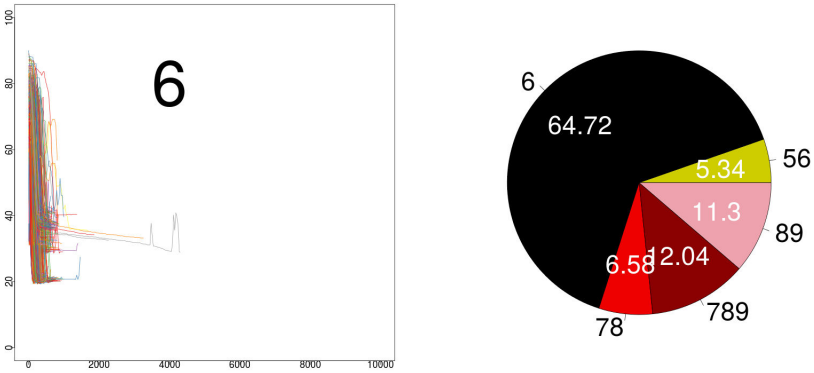


Figure 48: Représentation de la superclasse 2 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La composition des modes de vol de la superclasse 2 est similaire à celle de la superclasse CC_6. Elle est composée de phases ayant des changements abrupts (de 80% à 20%). Ces phénomènes ont généralement lieu durant la descente (56 et 66) ainsi que durant l'approche (78 et 789) et les phases d'atterrissage (89). D'après la Table 8, l'altitude n'est pas très élevée ce qui signifie que ces phases se déroulent non loin du sol.

4.4 Conclusion

Pour réaliser le *clustering* de phases transitoires de tailles inégales, nous avons testé deux méthodes: SOM relationnel appliquée à une dissimilarité basée sur une distance euclidienne glissante et SOM numérique. Les résultats obtenus par l'approche relationnelle n'ont pas été validés par les experts car des phases de taxi et de croisière étaient mélangées dans une même classe. En revanche, la méthode basée sur l'algorithme SOM numérique a fourni de très bons résultats. Nous avons construit une carte de Kohonen (11x11) et analysé les *clusters* qui sont bien homogènes. Cette méthode a été validée par des calculs d'erreurs (de quantification et topographique) et la prise en compte de l'étiquette ID_FM.

Ensuite nous avons réduit le nombre de classes en construisant des superclasses qui peuvent être décrites au moyen des statistiques descriptives élémentaires et des étiquettes des phases les composant.

Intéressons-nous maintenant au clustering de phases transitoires dans un contexte bivarié.

5 *Clustering* et analyse des phases bivariées

Jusqu'à ce point les phases transitoires croissantes ou décroissantes et leurs regroupements en classes ont été déterminés sur la base d'une seule variable (la variable N1). Il s'agit maintenant, à la demande des experts métier, d'introduire d'autres variables, et d'étudier ces phases dans un contexte multivarié.

Pour cela, nous étudions le comportement d'une variable supplémentaire mesurée sur les intervalles de temps qui délimitent les phases transitoires d'une première variable. En prenant en compte deux ou plusieurs variables, on cherche à analyser l'évolution d'une seconde variable durant une phase transitoire d'une première variable, le délai de réponse, l'existence de phases "rares", etc...

5.1 Méthodologie

Introduisons des notations générales. Soient V_1 et V_2 deux caractéristiques d'intérêt. On considère l'ensemble des phases transitoires croissantes définies sur la variable V_1, obtenues après segmentation des vols. On les note $Z_{l,a}$, $l = 1, \cdots, L$ où L est le numéro du vol et $a = 1, \cdots, A_L$, où A_l est le nombre de phases transitoires croissantes déterminées pour le vol L.

La première difficulté vient du fait que les phases transitoires $Z_{l,a}$ sont de longueurs différentes, et que les ruptures de la première variable V_1 ne correspondent pas à celles qu'on détecterait sur V_2.

On rappelle que les phases transitoires $Z_{l,a}$ ont été regroupées en superclasses CC_i, ce sont les superclasses de niveau 1 (bâties à partir de la variable V_1). Pour égaliser la taille de toutes les phases transitoires croissantes qui appartiennent à une même superclasse CC, on commence par sélectionner une courbe "réprésentative" de CC. Si $\tilde{dis}$ est la fonction de dissimilarité définie dans (21), la courbe représentative est donnée par:

$$Z_{\check{l},\check{a}} = \underset{(Z_{l_1,a_1}) \in CC}{\arg\min} \left(\sum_{(Z_{l_2,a_2}) \in CC, l_1 \neq l_2 \, ou \, a_1 \neq a_2} \tilde{dis}(Z_{l_1,a_1}, Z_{l_2,a_2}) \right) \tag{22}$$

C'est donc la courbe la plus proche en moyenne des phases de la même superclasse. Ensuite toutes les phases transitoires de la superclasse CC sont réalignées par rapport à $Z_{\check{l},\check{a}}$ comme expliqué dans la section 4.2.3 mais avec une autre dissimilarité:

$$\breve{dis}(Z_{\breve{l},\breve{a}}, Z_{l,a}) = \frac{\left\| Z_{l,a} - Z_{\breve{l},\breve{a}(q)} \right\|}{\left\| Z_{l,a(q)} \right\|} \tag{23}$$

où $Z_{\breve{l},\breve{a}(q)}$ est la translation de $Z_{\breve{l},\breve{a}}$ sur la phase $Z_{l,a}$.

Si une phase transitoire est plus courte à gauche et/ou à droite que $Z_{\breve{l},\breve{a}}$ après réalignement, alors on l'agrandit et si elle est plus longue à gauche et/ou à droite on la raccourcit, sinon on la garde telle quelle. Toutes les phases transitoires croissantes $Z_{l,a}$ ainsi réalignées ont donc la même longueur et sont notées $\breve{Z}_{l,a}$

Puis à chaque phase transformée $\breve{Z}_{l,a}$, on associe le vecteur $\breve{W}_{l,a}$, qui est le tronçon de courbe de la variable V_2 pris entre les instants de début et de fin de $\breve{Z}_{l,a}$. On note $(\breve{Z}, \breve{W})_{l,a}$ pour $l = 1, \cdots, L$ et $a = 1, \cdots, A_l$ l'ensemble des couples ainsi définis. La longueur de $\breve{Z}_{l,a}$ et $\breve{W}_{l,a}$ est la même que celle de la courbe représentative de la superclasse CC.

Afin de pouvoir réaliser le *clustering*, on remplace chaque courbe $\breve{W}_{l,a}$ par le vecteur de ses caractéristiques comme dans la section 4.2.1. La nature et le nombre M_2 des caractéristiques extraites sont les mêmes quelle que soit la superclasse considérée. Ainsi pour chaque superclasse CC, on construit une carte de Kohonen à partir de ces vecteurs de caractéristiques. Puis on réduit le nombre de classes comme précédemment en calculant des superclasses de niveau 2 qui forment une partition de chaque superclasse de niveau 1. On fait de même pour les superclasses de phases transitoires décroissantes. La Figure 49 fournit un schéma récapitulatif de la méthodologie proposée.

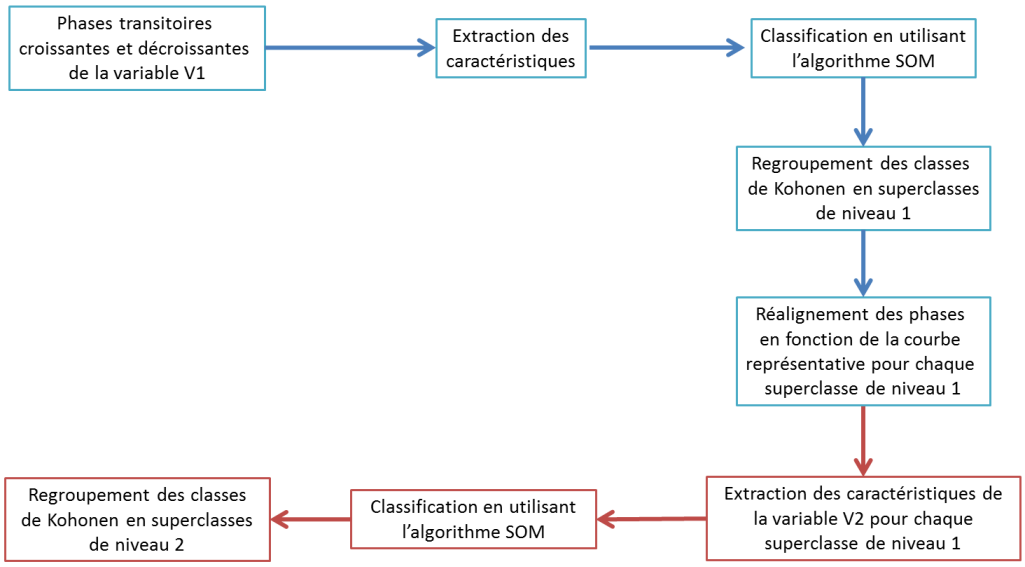


Figure 49: Schéma récapitulatif de la méthodologie de classification

5.2 Applications aux données réelles

Dans le contexte des données de vol et dans un cadre bivarié, on considère la vitesse de rotation N1 pour la variable V_1 et on choisit comme seconde variable V_2 la température avant combustion T3. On pourrait faire d'autres choix, mais la vitesse de rotation N1 et la température T3 ont une relation causale qu'il est intéressant de mettre en évidence. En effet la vitesse de rotation N1 résulte d'une action produite par le levier sur la vanne de fuel qui entraîne la vitesse de rotation N2 puis la température avant combustion T3.

Tout d'abord, on calcule les 7 courbes représentatives des 7 superclasses $CC_i, i = 1, \cdots, 7$ de niveau 1 et les 8 courbes représentatives des 8 superclasses $CD_j, j = 1, \cdots, 8$ selon la définition de l'équation (22). Dans la Figure 50, nous représentons en noir les courbes représentatives de quelques superclasses de niveau 1.

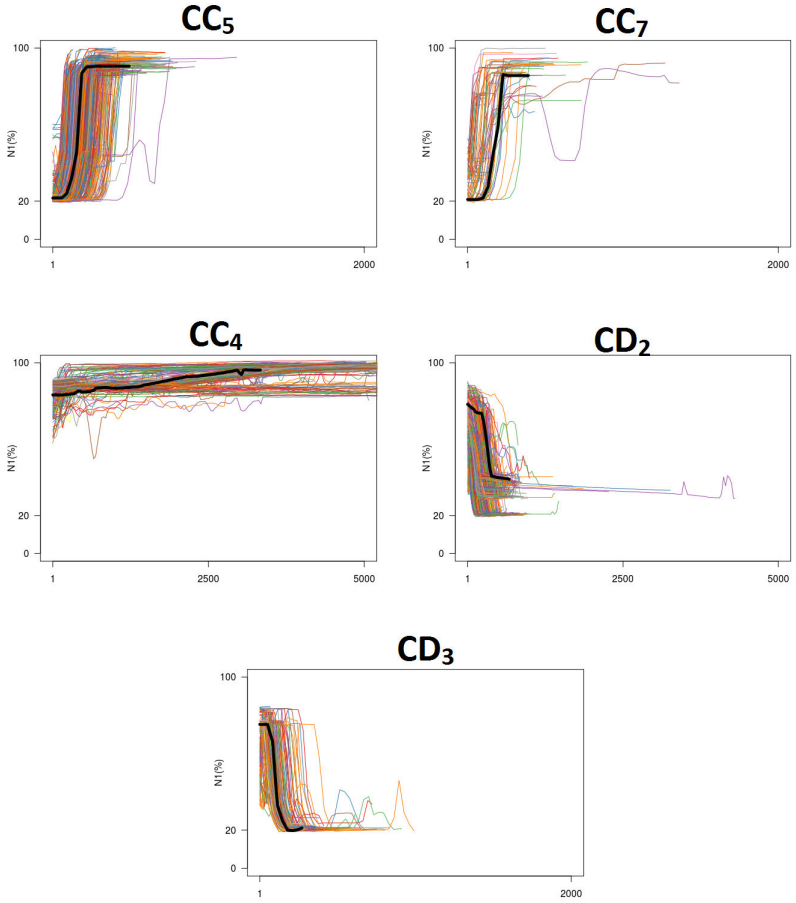


Figure 50: Représentation de la courbe représentative (en noir épais) de quelques superclasses.

Après avoir identifié les courbes représentatives, les autres phases sont

réalignées comme représenté dans la Figure 51. Il s'agit de réaliser une transformation de toutes les phases de la superclasse de sorte qu'elles coïncident le plus possible à la courbe représentative. Nous appliquons à chaque phase la dissimilarité (23) afin de trouver le meilleur réalignement.

Chaque phase $Z_{l,a}$ est ensuite transformée afin d'égaliser sa taille à la courbe représentative : pour raccourcir, on coupe tout simplement; pour rallonger, on agrandit.

Soit $[\tau_{l,k_1}, \tau_{l,k_2}]$ l'intervalle sur lequel la phase $Z_{l,a}$ est définie, noté $Z_{l,a,\tau_{l,k_1},\tau_{l,k_2}}$. La transformation de $Z_{l,a}$ s'écrit alors:

$$\breve{Z}_{l,a} = Z_{l,a,(\tau_{l,k_1}+1+\alpha_1),(\tau_{l,k_2}+\alpha_2)} \tag{24}$$

où α_1 et α_2 sont différents pour chaque phase.

Si $\tau_{l,k_1} + 1 + \alpha_1 < 0$ (on n'a plus de données, car c'est le début du vol), alors on a des valeurs manquantes à gauche de la phase (on a $\tau_{l,k_1} + 1 + \alpha_1$ valeurs manquantes) que l'on complète par la première valeur du vol $Y_{l,1}$.

Si $\tau_{l,k_2} + \alpha_2 > N_l$ (on n'a plus de données, car c'est la fin du vol) alors on a des valeurs manquantes à droite de la phase (on a $\tau_{l,k_2} + \alpha_2 - N_l$ valeurs manquantes) que l'on complète par la dernière valeur du vol Y_{l,N_l}.

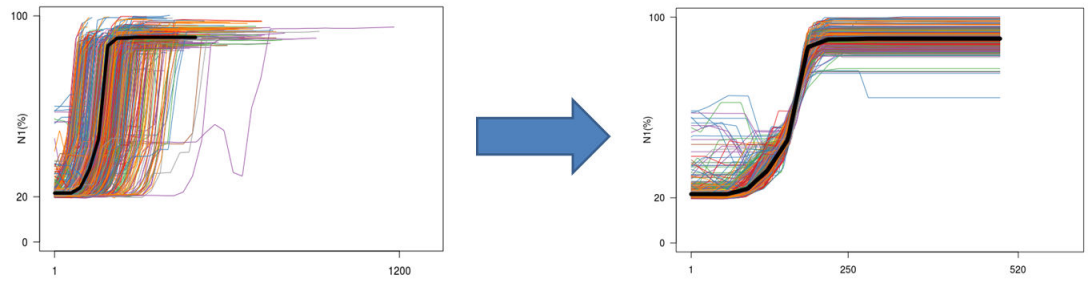


Figure 51: Représentation de la superclasse CC_5 après transformation des phases transitoires croissantes de la superclasse CC_5. Il s'agit de phases de décollage comme on l'a vu das la Figure 85.

On considère ensuite la variable V_2, c'est-à-dire les valeurs de la variable T3 découpées aux mêmes intervalles de temps que les phases $\breve{Z}_{l,a}$. Comme indiqué précédemment, ces courbes en T3 sont transformées en vecteurs de caractéristiques. On choisit les mêmes que celles utilisées pour les phases en N1, sauf la longueur puisqu'elle est commune à toutes les courbes : valeur de début, valeur de fin, valeur du milieu, médiane, variance, variance de la première moitié, variance de la deuxième moitié, moyenne de la première moitié, moyenne de la deuxième moitié. Pour chacune des 15 superclasses (phases croissantes et décroissantes) de niveau 1, on construit une

carte de Kohonen et des superclasses de niveau 2.

Le nombre de superclasses de niveau 2 est choisi de telle sorte qu'on obtienne 80% de la variance expliquée et il peut dépendre de la superclasse de niveau 1.

Les résultats de ces classifications de second niveau sont présentés dans les Figures 52, 53 pour les phases transitoires croissantes de N1 et dans la Figure 54 pour les phases transitoires décroissantes (le reste est disponible en annexe). On recense au total 102 superclasses de niveau 2 (42 pour les phases transitoires croissantes et 60 pour les phases transitoires décroissantes).

Les superclasses de niveau 2 sont présentées sur la même figure avec la même échelle afin de pouvoir les comparer entre elles. Elles sont séparées par des traits noirs, dans chaque case à gauche on trouve les phases en N1 $\breve{Z}_{l,a}$ et à droite les courbes en T3 $\breve{W}_{l,a}$.

On note :

- CC_i^j la j-ème superclasse de niveau 2 de phases transitoires croissantes de la i-ème superclasse de niveau 1 pour $i = 1, \cdots, 7$ et $j = 1, \cdots, n_i$.
- CD_k^l la l-ème superclasse de niveau 2 de phases transitoires décroissantes de la k-ème superclasse de niveau 1 pour $k = 1, \cdots, 8$ et $j = 1, \cdots, m_k$.

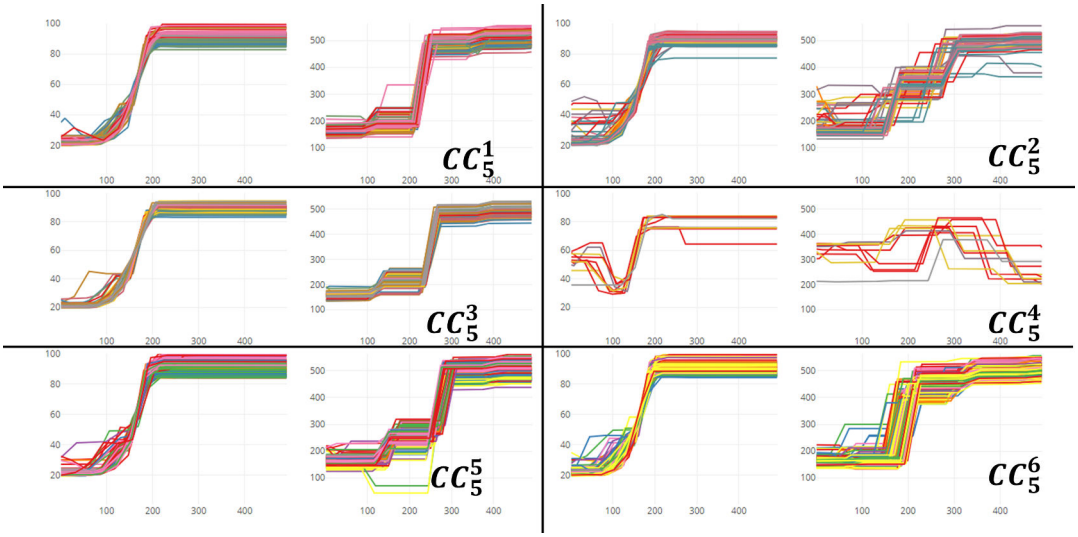


Figure 52: Partition de la superclasse CC_5 en 6 superclasses de niveau 2 notées CC_5^j pour $j = 1, \cdots, 6$. Pour chaque superclasse l'image de gauche représente les phases N1 et celles de droite les courbes T3.

La Figure 52 présente les superclasses de niveau 2 construites à partir de la superclasse CC_5 qui ne comprend que des phases de décollage. On remarque que les

montées en régime N1 ont été synchronisées grâce à la procédure de réalignement. Il est alors plus facile de comparer les courbes T3. Tout d'abord, on observe que les allures des courbes T3 sont globalement croissantes. C'est le cas dans les superclasses CC_5^1, CC_5^2, CC_5^3, CC_5^5 et CC_5^6. Toutefois la forme de ces sauts de température est différente d'une superclasse à l'autre. Par exemple dans les superclasses CC_5^1 et CC_5^6, les courbes de T3 n'ont quasiment qu'un changement abrupt. Les superclasses CC_5^3 et CC_5^5 regroupent des courbes T3 dont la croissance présente un plateau (moment de stabilisation entre deux augmentations) et à 200 degrés pour CC_5^3 à 300 degrés pour CC_5^5. De plus on observe que quelques sauts des courbes de la superclasse CC_5^2 se produisent plus tard que dans les autres superclasses. Cette particularité a intéressé les experts métier qui souhaitent l'analyser et comprendre son origine. La superclasse CC_5^4 se distingue des autres superclasses : les sauts de N1 et de T3 sont souvent précédés d'une descente. Ces comportements particuliers ont bien été "isolés" dans une superclasse.

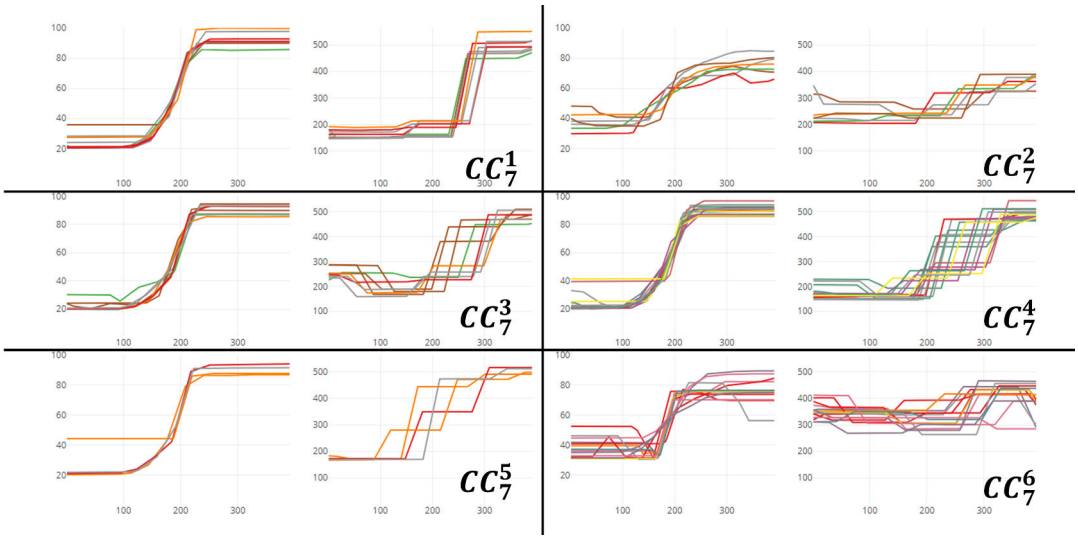


Figure 53: Partition de la superclasse CC_7 en 6 superclasses de niveau 2 notées CC_7^j pour $j = 1, \cdots, 6$. Pour chaque superclasse l'image de gauche représente les phases N1 et celles de droite les courbes T3.

Dans la Figure 53, on analyse la partition de la superclasse CC_7 en 6 superclasses de niveau 2. Cette superclasse CC_7 est composée majoritairement de phases de décollage et ces superclasses ont une certaine ressemblance avec celle de CC_5. Il y a une exception, les superclasses CC_7^2 et CC_7^6 sont différentes, les phases en N1 qui ressemblent à des décollages sont en réalité des phases d'approche, les courbes T3 commencent à plus de 300 degrés. Le *clustering* bivarié permet donc de séparer les phases d'approche des phases de décollage dans une même superclasse de niveau 1.

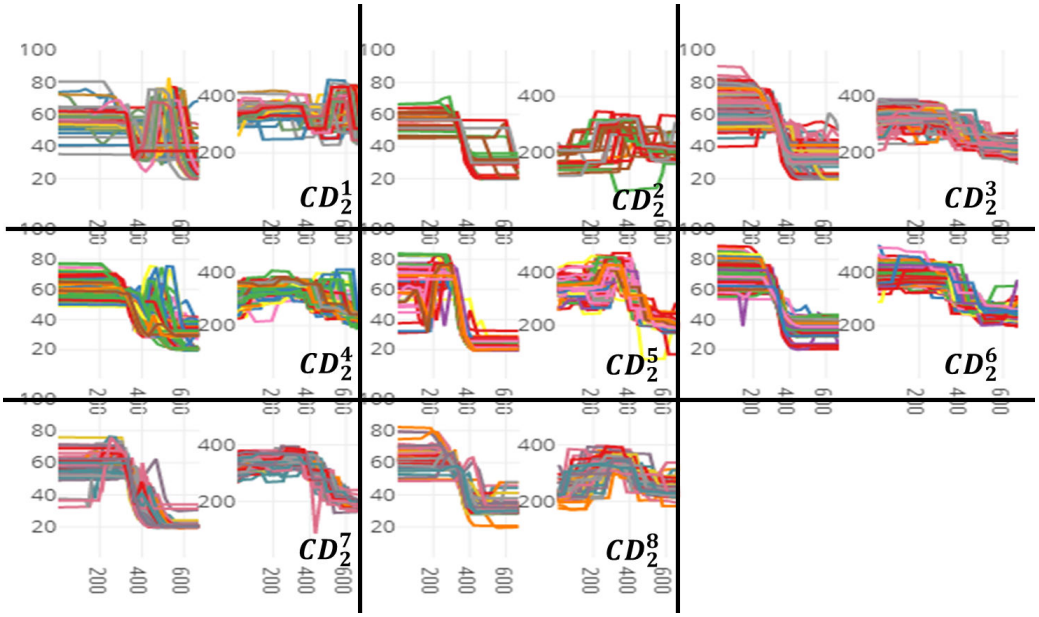


Figure 54: Partition de la superclasse CD_2 en 6 superclasses de niveau 2 notées CD_2^j pour $j = 1, \cdots, 8$. Pour chaque superclasse l'image de gauche représente les phases N1 et celles de droite les courbes T3.

On donne maintenant un exemple de *clustering* de phases décroissantes. Dans la Figure 54 (composée de phases de descente), les phases N1 des 8 superclasses de niveau 2 se ressemblent, comme prévu. Les superclasses de niveau 2 sont très homogènes en T3. Les superclasses CD_2^3, CD_2^5, CD_2^6 et CD_2^7 regroupent des courbes qui décroissent lentement en T3. Mais la superclasse CD_2^7 est composée de courbes T3 initialement stables qui diminuent progressivement. La valeur initiale de la température est plus élevée dans la superclasse CD_2^6 que dans les autres. Les courbes T3 des superclasses CD_2^2, CD_2^5 et CD_2^8 ont des formes très semblables (un intervalle de stabilité suivie d'une montée suivie d'une descente). Mais dans la superclasse CD_2^5, les valeurs initiales de T3 sont plus élevées que dans les superclasses CD_2^2 et CD_2^8. Enfin les valeurs de T3 dans la superclasse CD_2^2 restent plus ou moins constantes.

Le *clustering* bivarié permet donc d'analyser plus précisément les phases transitoires :

- Grâce au réalignement, on peut mettre en évidence des délais de réponse entre variables.
- Le *clustering* de niveau 2 permet de distinguer à l'intérieur d'une même superclasse de niveau 1 des ensembles bien homogènes que l'on peut donc identifier.

5.3 Analyse des résultats

Après avoir segmenté chaque vol et réalisé une classification des phases transitoires croissantes et décroissantes de N1, nous tentons maintenant d'analyser ces résultats. Dans un premier temps, nous identifions les phases atypiques des superclasses de niveau 1. Etant donné une phase/courbe, nous développons une méthode qui permet de lui associer l'ensemble des phases/courbes les plus proches au sens de l'indice de dissimilarité défini dans la section 4.2.3.

Enfin, nous proposons un algorithme pour comparer les vols entiers et pouvoir repérer les vols similaires ou les vols atypiques.

5.3.1 Détection d'anomalies

Les couples de vecteurs $(\breve{Z}, \breve{W})_{l,a}$ définis dans la section 5.1 (phase N1 notée $\breve{Z}_{l,a}$ et courbe T3 notée $\breve{W}_{l,a}$) et regroupés dans une même superclasse de niveau 1 après réalignement, forment en général un ensemble homogène. Mais certaines phases et/ou courbes présentent des "formes" atypiques.

L'idée ici est de construire dans chaque superclasse de niveau 1 (CC_i ou CD_k) deux tubes de confiance, l'un pour les phases N1 et l'autre pour les courbes T3. Condérons par exemple une superclasse CC_i dont tous les éléments ont la longueur λ_i. On commence par construire des intervalles de confiance en chaque valeur de t, $1 \leq t \leq \lambda_i$ (point par point). La propriété de normalité étant difficilement vérifiable à chaque instant dans le temps, nous avons choisi de construire des intervalles de confiance empiriques par la méthode des quantiles.

On définit pour tout instant t tel que $1 \leq t \leq \lambda_i$, l'ensemble de phases $\left\{\breve{Z}_{l,a}(t), \breve{Z}_{l,a} \in CC_i\right\}$ et on note $q_{t,5\%}$ et $q_{t,95\%}$ les quantiles de cet ensemble. L'intervalle $[q_{t,5\%}, q_{t,95\%}]$ est un intervalle de confiance empirique à 90% de $\breve{Z}_{l,a}(t)$.

On peut alors poser le "tube de confiance" des phases $\breve{Z}_{l,a}$ qui sont dans la superclasse CC_i comme l'ensemble des courbes entièrement comprises entre :

- le bord supérieur du tube défini par : $\{q_{t,95\%}, 1 \leq t \leq \lambda\}$
- le bord inférieur du tube défini par : $\{q_{t,5\%}, 1 \leq t \leq \lambda\}$.

De la même façon on construit les tubes de confiance empiriques pour toutes les courbes T3 d'une même superclasse.

Une fois définis les tubes de confiance empiriques à 90%, on dit qu'une phase N1 ou une courbe T3 est **atypique** si elle a au moins 10 points successifs à

l'extérieur du tube correspondant. Parmi celles-ci, on calcule leur distance par rapport aux courbes du tube de confiance et on sélectionne les plus éloignées.

En examinant systématiquement les 15 superclasses de niveau 1 (7 pour les phases transitoires croissantes et 8 pour les transitoires décroissantes), on repère 234 phases N1 et 298 courbes T3 ayant un comportement atypique. Ces phases et ces courbes sont signalées aux experts métier pour analyse et identification. Elles ne correspondent pas toutes à des anomalies au sens du fonctionnement du moteur. Mais inversement, on peut penser que les véritables anomalies se trouvent parmi ces exemples. On présente quelques exemples de phases/courbes atypiques dans les Figures 55, 56 et 57.

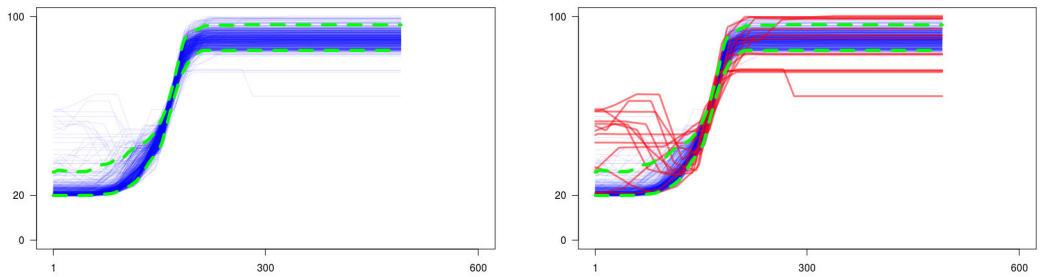


Figure 55: Représentation des phases N1 (en bleu) de la superclasse CC_5 après réalignement ainsi que le tube de confiance empirique dessiné en pointillés vert et les phases "atypiques" repérées en rouge (image de droite).

Dans la Figure 55, on remarque que le tube de confiance empirique représente bien la forme générale des courbes de la superclasse. Sur l'image de droite, quelques phases d'allure normale se trouvent en dehors du tube de confiance car leurs valeurs initiales et/ou finales sont légèrement plus basses et/ou plus fortes que les autres. On observe également des phases atypiques. Par exemple, il y a quelques phases ayant une diminution de N1 avant la montée en régime.

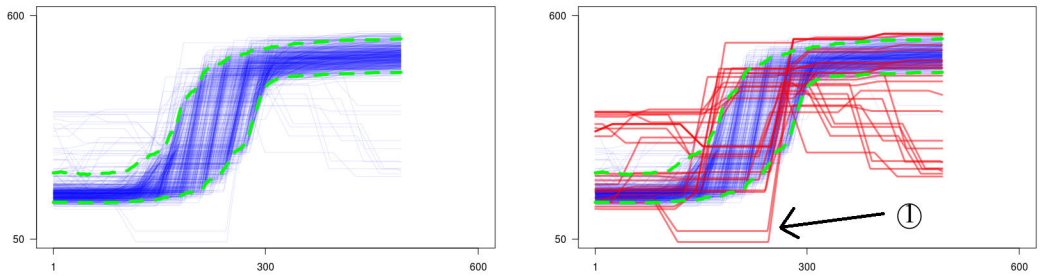


Figure 56: Représentation des courbes T3 (en bleu) de la superclasse CC_5 après réalignement ainsi que le tube de confiance empirique dessiné en pointillés vert et les courbes "atypiques" dessinées en rouge (image de droite).

Même remarque pour la Figure 56, quelques phases d'allure normale se trouvent en dehors du tube de confiance car leurs valeurs initiales et/ou finales sont légèrement plus basses et/ou plus hautes que les autres. Les courbes atypiques signalées par la flèche noire sont les courbes dont les températures sont très basses et celle avec une valeur initiale plus grande que celles des autres courbes. Dans la Figure 57, on représente uniquement les phases atypiques à la fois du point de N1 et T3. Remarquons aussi que les deux courbes T3 signalées dans la Figure 56 ne se retrouvent pas parmi celles de la Figure 57. Elles sont atypiques en T3, mais pas en N1. Dans les Figures 58 et 59, on extrait deux portions de vol de la superclasse CC_5 comprenant une phase N1 ou une courbe T3 atypique.

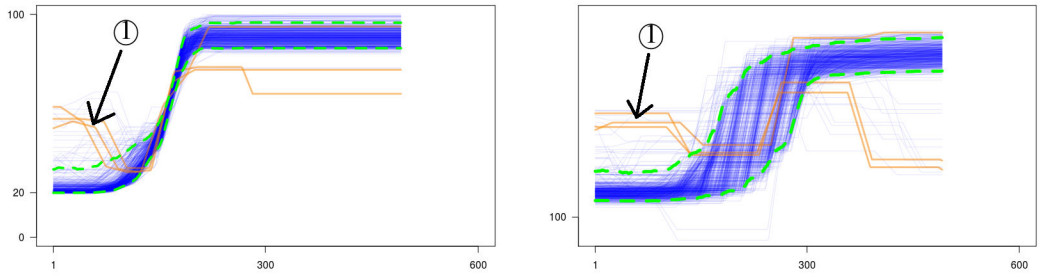


Figure 57: Représentation des couples (phase N1, courbe T3) de la superclasse CC_5 ainsi que des deux tubes de confiance correspondants dessinés en vert et les couples de courbes atypiques (du point de vue de N1 **et** de T3) dessinées en orange. Le couple signalé par ① est pris comme exemple dans la Figure 58.

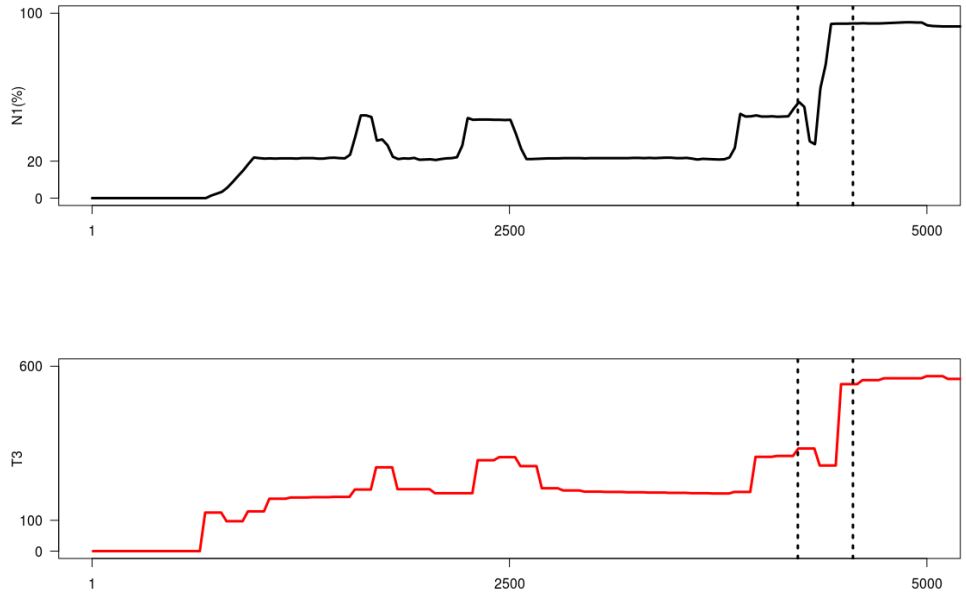


Figure 58: Exemple 1: Représentation d'une partie d'un vol (le premier tiers). On a représenté un couple ① (phase N1, courbe T3) dont les deux composantes sont atypiques et qui est repéré dans la Figure 57.

Dans la Figure 58, on a délimité entre pointillés une phase N1 et une courbe T3 toutes deux atypiques (voir Figure 57, ①). Cependant le recours à un expert métier est nécessaire pour déterminer l'origine de ce comportement.

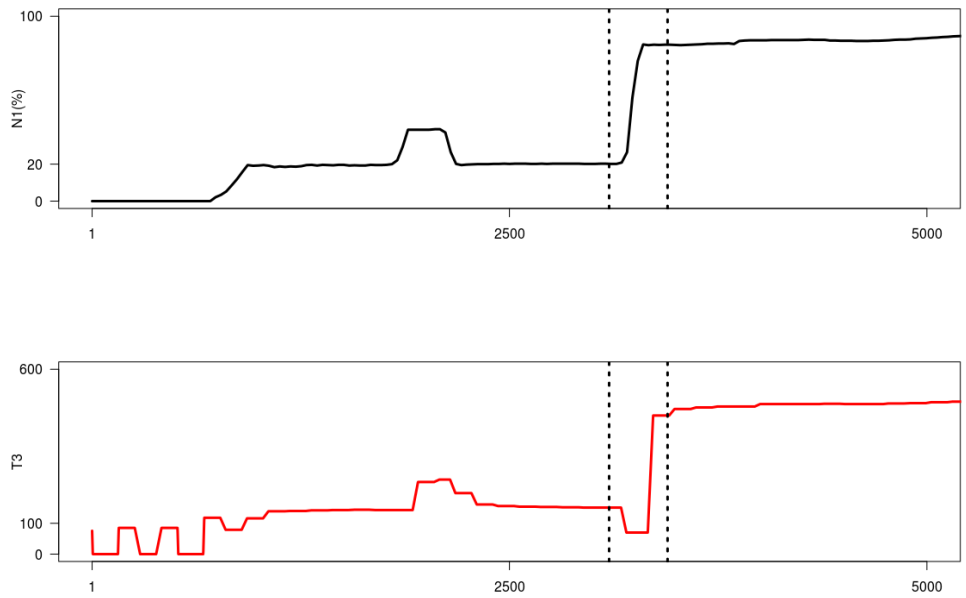


Figure 59: Exemple 2: Représentation d'une partie d'un vol (le premier tiers). On a déterminé une courbe T3 atypique telle que la phase N1 correspondante est normale.

Dans la Figure 59, la courbe T3 est atypique, on y observe des valeurs de températures particulièrement basses (voir Figure 56, ①). Cependant le tronçon N1 correspondant est parfaitement normal.

Grâce à cet outil, on peut repérer les phases/courbes atypiques. Pour une meilleure exploitation des résultats, l'outil fournit les identifiants, les valeurs, l'intervalle de temps des courbes atypiques.

5.3.2 Recherche de similarité entre les courbes

Une autre application de notre méthodologie a été de réaliser des recherches de similarité entre une courbe donnée (possiblement atypique aux yeux des experts) et les courbes classées par notre algorithme.

Soit un couple $(\tilde{Z}, \tilde{W})$ (phase en N1 et courbe en T3) n'appartenant pas à la base de données initiales. On se pose la question de rechercher de manière automatique un ensemble de couples (phase N1, courbe T3) déjà étudiés et proches de $(\tilde{Z}, \tilde{W})$ au sens de l'indice de dissimilarité $\tilde{dis}$ défini à la section 4.2.3.

La phase $\tilde{Z}$ est tout d'abord représentée par le vecteur de ses caractéristiques comme on les a définies dans la section 4.3.2, et on cherche la superclasse de niveau 1 à laquelle elle appartient.

Pour cela, pour chaque superclasse de niveau 1, on calcule la moyenne de toutes les phases qu'elle contient. Cette moyenne est différente de la courbe représentative de la superclasse définie à la section 5.1). La moyenne de la superclasse appartient à l'espace des caractéristiques et la courbe représentative est un tronçon de série temporelle.

Soit U^c le nombre de clusters de la carte de Kohonen appliquée aux phases croissantes. On note $C_u^c, u = 1, \cdots, U^c$ l'ensemble des classes et $p_u^c, u = 1, \cdots, U^c$ l'ensemble des prototypes correspondants. Soit $U_i^c = \{u / C_u^c \subset CC_i\}$ l'ensemble des indices des clusters appartenant à la superclasse CC_i, pour tout $i = 1, \cdots, 7$. Similairement, on note U^d le nombre de clusters de la carte de Kohonen appliquée aux phases décroissantes, $C_u^d, u = 1, \cdots, U^d$ l'ensemble des classes, $p_u^d, u = 1, \cdots, U^d$ l'ensemble des prototypes correspondants et $U_j^d = \left\{u / C_u^d \subset CD_j\right\}$ l'ensemble des indices des clusters appartenant à la superclasse CD_j, pour tout $j = 1, \cdots, 8$.

La moyenne des phases transitoires croissantes de la superclasse CC_i est définie par :

$$\overline{CC_i} = \frac{\sum_{u \in U_i^c} |C_u^c| \, p_u^c}{\sum_{u \in U_i^c} |C_u^c|}$$

et celle des phases transitoires décroissantes de la superclasse CD_j par :

$$\overline{CD_j} = \frac{\sum_{u \in U_j^d} \left|C_u^d\right| p_u^d}{\sum_{u \in U_j^d} |C_u^d|}.$$

On note $v_{\tilde{Z}}$ le vecteur de caractéristiques de $\tilde{Z}$ et $\|.\|$ la distance euclidienne dans l'espace des caractéristiques. Selon l'allure de $\tilde{Z}$ (phase croissante ou décroissante), elle est affectée à une superclasse CC_i ou CD_j.

- Si $\tilde{Z}$ est une phase croissante, elle est affectée à la classe CC_i telle que la distance $\left\|\overline{CC_i} - v_{\tilde{Z}}\right\|$ est minimum.
- Si $\tilde{Z}$ est une phase décroissante, elle est affectée à la classe CD_j telle que la distance $\left\|\overline{CD_j} - v_{\tilde{Z}}\right\|$ est minimum.

Il est raisonnable de restreindre la recherche des phases N1 proches de $\tilde{Z}$ au sens de la dissimilarité $\tilde{dis}$, à la superclasse à laquelle $\tilde{Z}$ a été affectée. En effet même s'il n'y a pas équivalence entre la proximité au sens de l'indice $\tilde{dis}$ et au sens de la distance euclidienne dans l'espace des caractéristiques, la superclasse de $\tilde{Z}$ contient nécessairement un certain nombre de phases similaires.

Pour la courbe $\tilde{W}$, on réalise différentes étapes:

- Réaligner $\tilde{Z}$ sur la courbe représentative de sa superclasse selon la formule (24);

- Appliquer à $\tilde{W}$ la même transformation (translation, raccourcissement et/ou allongement), on note $\tilde{W}'$ le résultat de cette transformation ;
- Représenter $\tilde{W}'$ par son vecteur de caractéristiques;
- Affecter $\tilde{W}'$ à une superclasse de niveau 2 comme précédemment.

On présente ensuite quelques exemples dans les Figures 60 à 65. Un premier exemple (Figures 60 à 63) est composé d'une phase N1 et d'une courbe T3. La phase N1 est affectée à la superclasse CC_5 et la courbe T3 à la superclasse CC_5^1. Les couples similaires trouvés ont la même allure que le couple $(\tilde{Z}, \tilde{W})$.

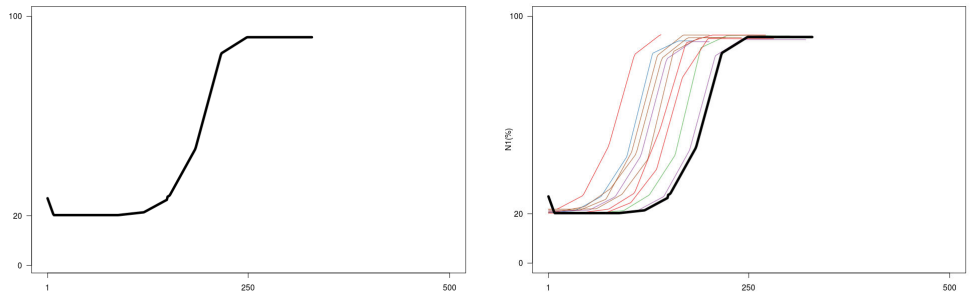


Figure 60: Exemple 1 : Représentation d'une phase N1 $\tilde{Z}$ (image de gauche) et d'un ensemble de phases similaires à cette phase (image de droite).

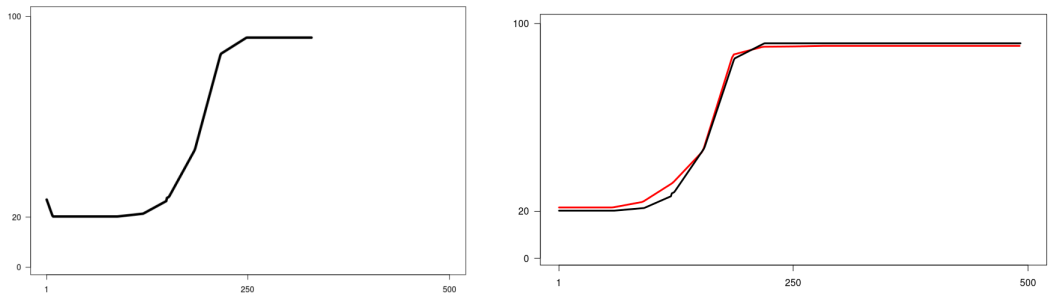


Figure 61: Exemple 1 : Représentation de la phase N1 $\tilde{Z}$ avant et après réalignement sur la courbe réprésentative en rouge.

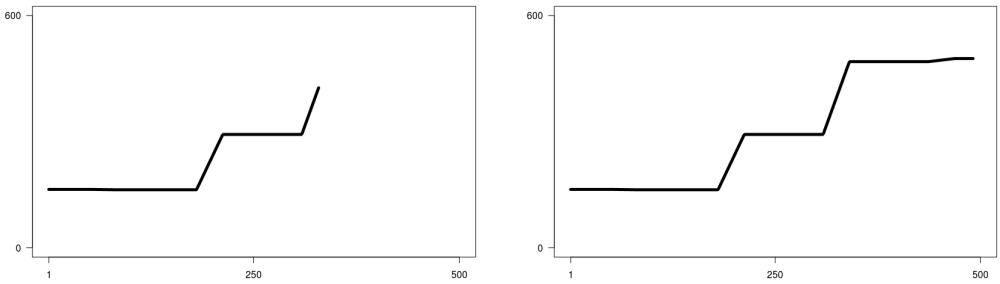


Figure 62: Exemple 1 : Représentation de la courbe T3 $\tilde{W}$ avant et après transformation.

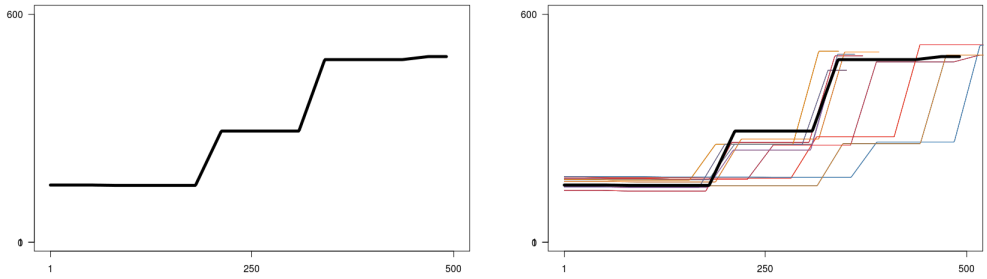


Figure 63: Exemple 1 : Représentation de la courbe T3 $\tilde{W}'$ (image de gauche) et d'un ensemble de courbes similaires (image de droite).

Le deuxième exemple présenté ici contient une phase N1 ayant un plateau (bien que ce soit une phase de décollage). Les résultats sont présentés dans la Figure 64. Cette phase N1 est affectée à la superclasse CC_5 et les phases similaires trouvées ont la même allure.

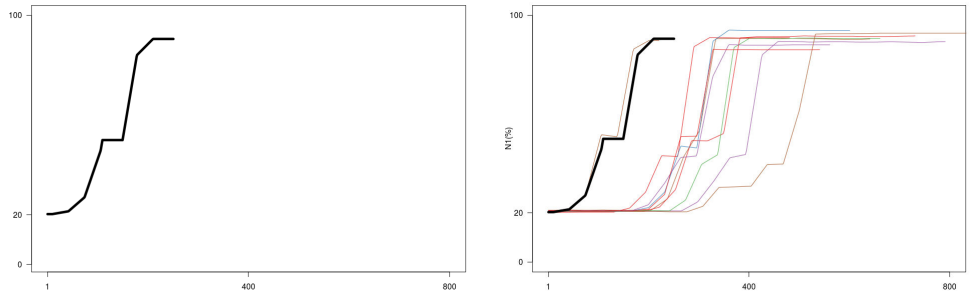


Figure 64: Exemple 2 : Représentation d'une phase N1 à plateau (image de gauche) et d'un ensemble de phases similaires (image de droite).

Enfin, on considère une phase de *climb* très lent, le résultat étant présenté dans la Figure 65.

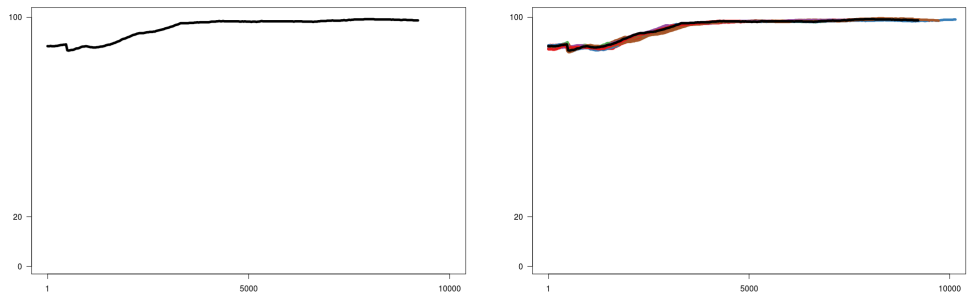


Figure 65: Exemple 3 : Représentation d'une phase de *climb* très lent (image de gauche) et d'un ensemble de phases similaires (image de droite).

La phase N1 est affectée à la superclasse CC_4 et les phases similaires sont pratiquement confondues entre elles.

5.4 Conclusion du chapitre

Dans ce chapitre nous avons proposé un *clustering* bivarié emboîté. Après avoir calculé des superclasses de phases croissantes et décroissantes à partir d'une variable d'intérêt dans le chapitre précédent, nous avons rajouté ici une seconde dimension. Pour ce faire, pour chaque superclasse trouvée précédemment, les phases la composant ont été réalignées sur une courbe représentative afin qu'elles aient toutes la même longueur. Une fois les phases réalignées, on a pu rajouter les courbes d'une seconde variable délimitées aux mêmes instants que les phases de niveau 1. La même méthodologie de clustering que celle du chapitre 4 a été ensuite appliquée sur les courbes de la seconde variable afin de créer des superclasse de niveau 2.

Cette nouvelle représentation permet de comparer des séries temporelles bivariées et de repérer les comportements récurrents et non récurrents par superclasse. Le *clustering* de niveau 2 permet d'affiner les superclasses de niveau 1.

Pour repérer les phases atypiques, nous avons mis en place un outil permettant de repérer dans chaque superclasse de niveau 2 des couples de phases/courbes atypiques au moyen de tubes de confiance. Ces résultats ont été ensuite transmis aux experts afin qu'ils les analysent.

Enfin nous pouvons maintenant faire une recherche non exhaustive de phases similaires à une phase donnée. Cette recherche de similarité est très rapide et les résultats sont excellents.

Par la suite, nous étudions les vols en entier en nous aidant des différentes méthodes de *clustering* mises en place.

6 Analyse statistique de séquences labellisées

Jusqu'à maintenant on s'est intéressé uniquement à l'identification et à la typologie des phases de vol. Dans ce nouveau chapitre, le but est d'étudier les vols dans leur ensemble en utilisant les résultats des chapitres précédents: la segmentation des séries temporelles et le *clustering* bivarié.

Il s'agit de répondre à trois questions :

- Quels sont les vols similaires à un vol donné?
- Quels sont les vols ayant un comportement "anormal"?
- Comment déterminer un vol "représentatif" parmi un ensemble de vols?

La stratégie que nous adoptons ici est d'exprimer chaque vol "segmenté" à partir de la suite des labels de ses segments. Un label est une succession de lettres et/ou de numéros désignant/résumant une phase transitoire ou stabilisée identifiée par la méthode de détection de ruptures et affectée à une classe suite au *clustering*. Cette suite de labels est alors considérée comme un code qui résume le vol toute entier et est désignée dans le reste du chapitre par l'expression **séquence labellisée**.

6.1 Etat de l'art

La labellisation des séries temporelles peut prendre plusieurs formes. On peut, par exemple, définir des typologies de séries temporelles et identifier chaque classe par un label. Ici, notre démarche est différente, nous cherchons à représenter chaque série temporelle par une suite de labels (séquence), cette suite de labels étant considérée comme un résumé de la série toute entière.

Les travaux sur la labellisation de séries temporelles ne sont pas très nombreux. On recense beaucoup de méthodes de *clustering* qui permettent d'identifier les comportements récurrents et non récurrents comme on l'a expliqué dans la section 4.2.4, mais il y a peu de références sur la labellisation.

Dans (Peter, Höppner, & Berthold, 2013), Peter *et al.* introduisent un algorithme labellisant les séries temporelles en les découpant selon des intervalles réguliers et en les discrétisant en utilisant des seuils prédéfinis. Les séries temporelles sont alors approchées par des séries constantes par morceaux. Les bornes des intervalles ne sont donc pas calculées comme des points de rupture des séries contrairement à notre démarche. Elles sont choisies de manière arbitraire.

Par ailleurs, on peut rapprocher la notion de série temporelle labellisée sous la forme d'une séquence de labels de celle de séries temporelles catégorielles

ou de celle de données longitudinales. Avec cette analogie, on peut se référer à la littérature sur les données longitudinales, notamment en Sciences Humaines et Sociales. Plusieurs méthodes sont proposées dans la littérature pour comparer des séquences longitudinales. L'approche que nous adoptons ici consiste à définir une distance/dissimilarité et à construire des typologies.

Pour calculer des dissimilarités entre deux séquences, il existe plusieurs classes d'approches. On a par exemple les métriques basées sur un comptage des attributs communs : distances basées sur la plus longue sous séquence commune, sur le plus long préfixe commun, sur le plus long suffixe commun, sur le nombre de *matching* sous séquences, sur le nombre de *matching subsequences weighted by the minimum shared time* et la distance "*subsequence vectorial representation*".

Une deuxième famille de distances sont les distances d'édition : *optimal matching* (OM), *localized OM*, *spell-length-sensitive OM*, *OM of spell sequences*, *OM of transition sequences*, *Hamming*, *dynamic Hamming*, et le *time warp edit distance*.

Une description exhaustive des distances disponibles pour l'analyse des données longitudinales (dans le contexte des Sciences Humaines et Sociales) est disponible dans (Studer & Ritschard, 2016).

Dans ce travail, nous nous sommes intéressés particulièrement à la méthode *Optimal Matching* que nous décrivons brièvement dans la suite. Egalement appelée distance de Levenstein, elle a été initalement introduite en biologie pour l'analyse des protéines et des séquences d'ADN (Needleman & Wunsch, 1970). Elle a été ensuite utilisée par Abbott pour la première fois en sciences sociales dans (Abbott & Forrest, 1986), ainsi que dans diverses applications (Wu, 2000), (Abbott & Tsay, 2000) et (Halpin & Chan, 1998). La méthode consiste à "transformer" une séquence A en une séquence B en effectuant une succession de changements de labels (**substitution, suppression et ajout**). Chaque changement a un coût et la distance entre deux séquences correspond au coût minimal associé aux opérations nécessaires pour transformer une séquence A en une séquence B. Alors que pour les données génétiques, la définition des coûts est immédiate et a un sens biologique, dans le cas des données en sciences humaines et sociales ou en ingénierie, la définition des coûts n'est pas toujours évidente. On peut alors recourir à des heuristiques issues des statistiques exploratoires et définir les coûts à partir des matrices de transition empiriques entre les labels, etc... Un état de l'art détaillé sur le calcul des coûts a été proposé dans (Studer & Ritschard, 2016).

6.2 Labellisation des vols

Contexte univarié

Tout d'abord, la labellisation de chaque série temporelle univariée est relativement simple. Comme nous l'avons vu précédemment, chaque vol est segmenté par un algorithme de détection de ruptures. Nous avons ensuite défini chaque segment comme étant soit une phase stabilisée, transitoire croissante ou transitoire décroissante. Grâce au *clustering* univarié sur les séries décrites par la variable V_1 chaque phase transitoire en V_1 est associée à une superclasse de niveau 1 (CC_i ou CD_j). Par conséquent chaque vol sera codé comme une suite de labels, définis comme suit. La lettre correspond à la nature de la phase (S pour phase stabilisée, A pour phase transitoire croissante et D pour phase transitoire décroissante)

Pour les deux types de phases transitoires, la lettre est suivie d'un chiffre qui représente la superclasse de niveau 1 à laquelle elle appartient. Nous présentons quelques exemples de séquences labellisées:

Exemple 1 : "A1 -> D7 -> S -> A5 -> A4 -> D4 -> D6 -> A6 -> D6 -> A6 -> S -> D6 -> A6 -> D2 -> D7"

Exemple 2 : "A1 -> S -> D7 -> A5 -> S -> D4 -> S -> D8 -> S -> A6 -> D5 -> A2 -> D8 -> S -> A6 -> D3 -> S -> D7"

Contexte bivarié

Après la labellisation par les résultats du clustering univarié, nous introduisons le contexte bivarié. Les séries sont alors décrites par les variables V_1 et V_2. Chaque courbe en $V2$ correspondant à la phase en V_1 est associée à une superclasse de niveau 2 (sachant le premier label par la superclasse de niveau 1).

Ainsi chaque vol sera codé comme une suite de labels, définis comme dans le cas univarié, mais la lettre est cette fois suivie de deux chiffres : le premier représente le numéro de la superclasse de niveau 1 et le second le numéro de la superclasse de niveau 2.

Nous présentons les mêmes exemples de séquences labellisées[1] :

Exemple 1 : "A11 -> A12 -> D74 -> S -> A53 -> A44 -> D42 -> D44 -> D61 -> A65 -> D65 -> A65 -> S -> D63 -> A67 -> D27 -> D75"

[1]Une phase a été considérée comme transitoire dans un premier temps mais en raison de la règle définie dans 3.4.3, il peut arriver que plusieurs phases stabilisées se suivent. Par exemple, on peut observer l'enchaînement suivant : phase ayant une augmentation de 9,5%, phase ayant une diminution de 3%. Ces deux phases sont considérées comme des phases stabilisées.

Exemple 2 : "A11 -> A12 -> S -> D72 -> A53 -> S -> D42 -> S -> D81 -> S -> A69 -> D51 -> A26 -> D81 -> S -> A68 -> A65 -> D32 -> S -> D75"

Quelques remarques et statistiques empiriques

Chaque vol repéré par son numéro (1 à 549) correspond à une séquence labellisée qui le résume. Bien qu'en principe des vols différents pourraient avoir la même séquence labellisée, on n'a pas observé ce cas dans notre base de données et donc chaque séquence labellisée code un vol et un seul.

La longueur d'une séquence labellisée est définie par le nombre de labels la composant. Les 549 séquences labellisées ont en moyenne $22,5$ labels avec un minimum de 10 et un maximum de 35. La Figure 66 représente la distribution des 20 labels les plus fréquents.

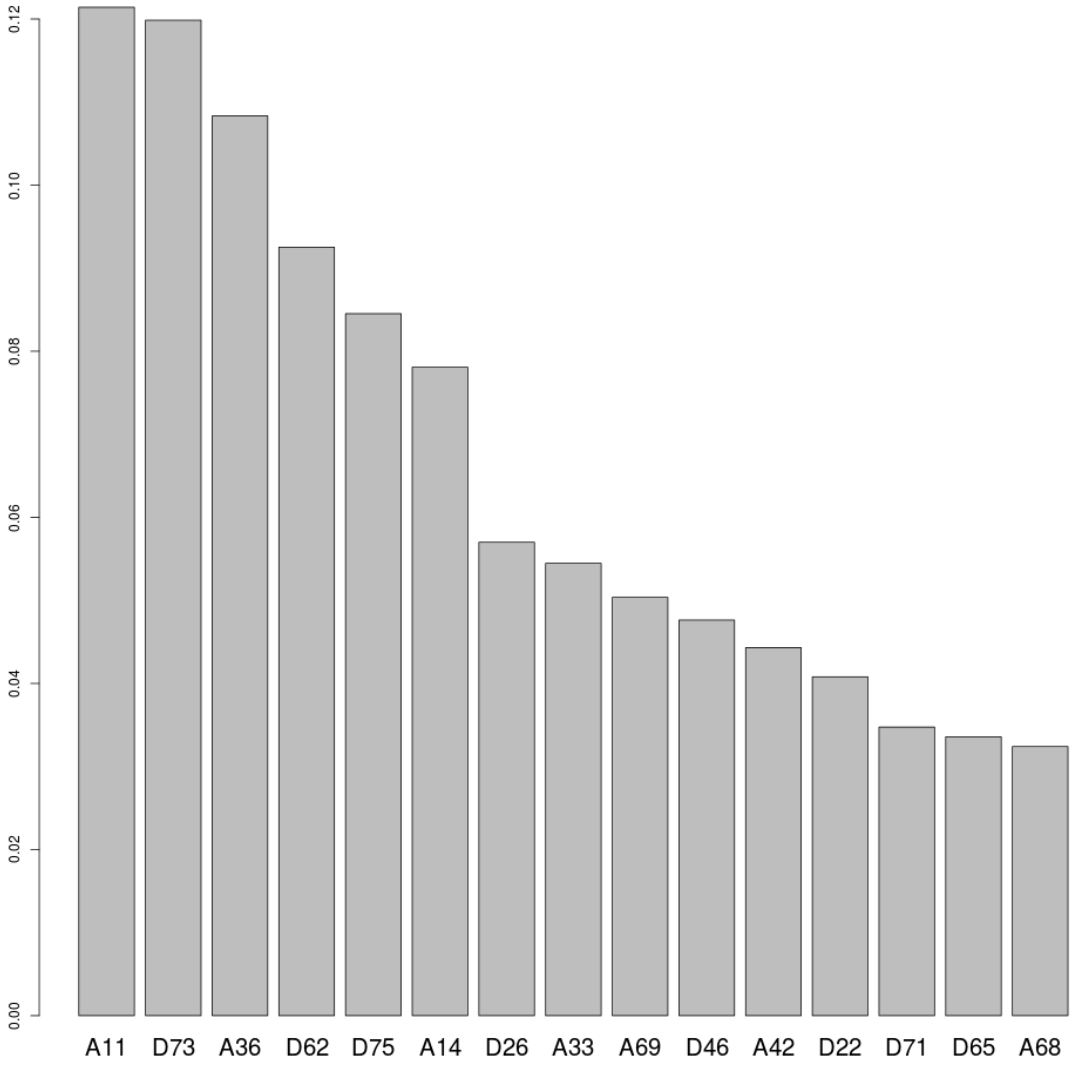


Figure 66: Distribution des labels les plus fréquents en dehors du label stabilisé S

Le label le plus fréquent (en dehors du label stabilisé S) est le label A11 (469 occurences) qui représente un label de taxi. Globalement les labels les plus fréquents sont les phases de taxi (A11, A14, D73, D75, A36 et A33) et de descente (D62 et D26).

Nous réalisons ensuite une analyse des transitions entre labels. Les transitions les plus fréquentes sur des séquences de deux, trois ou quatre labels sont les suivantes

- Deux labels :
  - D62->A36 (286 occurrences);

  - S->D73 (171 occurrences);
  - S->D26 (147 occurrences);
  - S->A46 (122 occurrences).

- Trois labels :
  - A14->A11->S (86 occurrences);
  - S->D26->D62 (66 occurrences);
  - S->D26->S (61 occurrences);
  - S->D46->S (58 occurrences).

- Quatre labels :
  - A14->A11->D73->S (37 occurrences);
  - S->D26->D62->A36 (35 occurrences);
  - A61->S->D26->S (34 occurrences);
  - D45->D26->D62->A36 (31 occurrences).

Nous remarquons que la plupart des transitions se déroulent durant les phases de taxi (S->D73, A14->A11->S et A14->A11->D73->S), les phases de descente (S->D26, S->D26->D62 et S->D26->D62->A36) et également les fin de *climb* (S->D46->S).

Par ailleurs, à partir de sous-séquences de longueur 5, les transitions deviennent uniques.

6.3 *Optimal Matching*

Pour pouvoir comparer les séquences labellisées, nous utilisons la méthode *Optimal Matching*. Comme nous l'avons expliqué, il faut tout d'abord définir les coûts de chaque transition. Dans (Studer & Ritschard, 2016), Studer *et al.* présentent un catalogue de différentes méthodes de calcul des coûts. Nous en avons essayé plusieurs (basées sur le nombre labels successifs en commun entre les séquences puis sur les labels en commun entre les séquences), toutefois les résultats des coûts de substitution ne correspondait pas à "réalité" de nos données (séquences labellisées). Par exemple, le coût de substitution entre une phase croissante et une autre phase croissante était plus élevée que le coût de substitution entre une phase croissante et une autre phase décroissante.

Pour contourner ce problème, nous définissons ces coûts à partir des cartes de Kohonen construites durant le *clustering* univarié et bivarié. Ainsi les coûts de substitution reflèteront plus la nature des labels utilisés.

Plusieurs types de transitions sont définis. Il faut considérer plusieurs cas :

1. coût d'ajout d'un label;
2. coût de suppression d'un label;
3. coût de substitution entre deux labels :
   (a) entre un label de phase croissante ou décroissante et un label de phase stabilisée (et vice versa);
   (b) entre un label de phase croissante et un label de phase décroissante (et vice versa);
   (c) entre un label de phase croissante Aij et un autre label de phase croissante Akl. Deux cas sont alors distingués:
      i. si $i = k$ (phases appartenant à la même superclasse de niveau 1);
      ii. si $i \neq k$ (phases n'appartenant pas à la même superclasse de niveau 1);
   (d) entre deux labels de phases décroissantes comme dans 3.(c).

On commence par calculer les coûts 3.(c).i et 3.(c).ii à partir des cartes de Kohonen construites dans les sections 4.3.2 et 4.4.2.

Soient Aij et Akl deux labels de phases transitoires croissantes.

Si $i = k$ et $j \neq l$, les phases appartiennent à la même superclasse de niveau 1, soit CC_i et nécessairement à des superclasses de niveau 2 différentes [2]. Le coût de la substitution est alors :

$$c(Aij, Ail) = \frac{\left\| \overline{CC_i^j} - \overline{CC_i^l} \right\|_2}{\max_{j,l} \left\| \overline{CC_i^j} - \overline{CC_i^l} \right\|_2}. \tag{25}$$

où $\overline{CC_i^j}$ et $\overline{CC_i^l}$ sont les moyennes des superclasses de niveau 2 CC_i^j et CC_i^l.

Si $i \neq k$, la substitution est beaucoup plus coûteuse. Dans ce cas, on définit le coût comme la somme d'une transition de niveau 1 et une transition de niveau 2. Ainsi, le coût dépend de la distance entre les moyennes des superclasses de niveau 1 $\overline{CC_i}$ et $\overline{CC_k}$, puis pour prendre en compte le changement de superclasse de niveau 2, on doit chercher la distance entre les superclasses de niveau 2, soient CC_i^j et CC_k^l. Comme ces deux superclasses n'appartiennent pas à la même carte, on calcule la distance euclidienne entre les moyennes $\overline{CC_i^j}$ et $\overline{CC_k^l}$ dans l'espace des caractéristiques définies pour représenter les courbes de la variable V_2.

[2] Si $i = k$ et $j = l$ il s'agit de la même phase et il n'y a pas de substitution

On définit donc le coût de substitution entre le label Aij et le label Akl par :

$$c(Aij, Akl) = \frac{\left\|\overline{CC_i} - \overline{CC_k}\right\|_2}{\max\limits_{i,k}\left\|\overline{CC_i} - \overline{CC_k}\right\|_2} + \frac{\left\|\overline{CC_i^j} - \overline{CC_k^l}\right\|_2}{\max\limits_{j,l}\left\|\overline{CC_i^j} - \overline{CC_k^l}\right\|_2}. \tag{26}$$

Soient $Di'j'$ et $Dk'l'$ deux labels de phases transitoires décroissantes. De la même manière, le coût de substitution entre un label $Di'j'$ et un label $Dk'l'$ avec $i' = k'$ est défini par :

$$c(Di'j', Di'l') = \frac{\left\|\overline{CD_{i'}^{j'}} - \overline{CD_{i'}^{l'}}\right\|_2}{\max\limits_{j',l'}\left\|\overline{CD_{i'}^{j'}} - \overline{CD_{i'}^{l'}}\right\|_2}. \tag{27}$$

Si $i' \neq k'$ alors :

$$c(Di'j', Dk'l') = \frac{\left\|\overline{CD_{i'}} - \overline{CD_{k'}}\right\|_2}{\max\limits_{i',k'}\left\|\overline{CD_{i'}} - \overline{CD_{k'}}\right\|_2} + \frac{\left\|\overline{CD_{i'}^{j'}} - \overline{CD_{k'}^{l'}}\right\|_2}{\max\limits_{j',l'}\left\|\overline{CD_{i'}^{j'}} - \overline{CD_{k'}^{l'}}\right\|_2}. \tag{28}$$

Etant donnée la nature des labels, le coût de substitution entre un label de phase croissante et un label de phase décroissante doit être le plus élevé possible car on ne peut échanger facilement un label de phase croissante et un label de phase décroissante. Sinon il serait moins coûteux de changer Aij en Akl en passant par $Di'j'$ si $c(Aij, Akl) > c(Aij, Di'j') + c(Di'j', Akl)$, ce qu'on ne souhaite pas. Ce coût de substitution est alors égal à :

$$2 \times \max(\max_{i,j,k,l}(c(Aij, Akl)), \max_{i',j',k',l'}(c(Di'j', Dk'l'))). \tag{29}$$

Le coût de substitution entre un label de phase croissante et un label de phase stabilisée doit être supérieur au maximum des coûts de substitution entre deux labels de phase croissante. Sinon il serait moins coûteux de transformer label de phase croissante Aij en un autre label de phase croissante Akl en passant par une label de S, c'est-à-dire $c(Aij, Akl) > c(Aij, S) + c(S, Akl)$, ce qu'on ne souhaite pas. Donc :

$$\max_{i,j,k,l}(c(Aij, Akl)). \tag{30}$$

De même le coût de substitution entre un label de phase décroissante et un label de phase stabilisée est égal au maximum des coûts de substitution entre deux labels de phase décroissante. Soit:

$$\max_{i,j,k,l}(c(Dij, Dkl)). \tag{31}$$

Notons que le coût d'ajout ou de suppression d'un label est le même pour tous les labels et se doit être aussi coûteux. En effet, supprimer puis ajouter un label doit être aussi coûteux que le coût de substitution entre un label de phase croissante et un label de phase décroissante. On ne souhaite pas remplacer facilement label de phase croissante par un label de phase décroissante. Il est alors égal à l'expression (29).

Matrice de dissimilarité

On applique maintenant la méthode *optimal matching* sur notre base de données de séquences labellisées. Le package "TraMineR" (Gabadinho, Ritschard, Muller, & Studer, 2011) permet d'analyser des séquences longitudinales (séries temporelles catégorielles). La méthode *optimal matching* y est déjà implémentée et les coûts peuvent être définis manuellement. Nous utilisons les coûts mentionnés précédemment. Dans la Table 9 est représentée partiellement la matrice des coûts de substitution.

	$A11$	$A12$	$A13$	$A14$	$A21$	$A22$	$A23$	$A24$	$A25$
$A11$	**0,00**	**0,14**	**0,62**	**0,92**	$1,29$	$1,95$	$1,97$	$1,78$	$1,20$
$A12$	**0,14**	**0,00**	**0,65**	**1,00**	$1,28$	$1,65$	$1,59$	$1,78$	$1,42$
$A13$	**0,62**	**0,65**	**0,00**	**1,00**	$1,77$	$1,78$	$1,76$	$1,01$	$1,59$
$A14$	**0,92**	**1,00**	**1,00**	**0,00**	$1,79$	$1,00$	$1,70$	$1,14$	$1,59$
$A21$	$1,29$	$1,28$	$1,77$	$1,79$	**0,00**	**0,02**	**0,18**	**0,03**	**0,29**
$A22$	$1,95$	$1,65$	$1,78$	$1,00$	**0,02**	**0,00**	**0,16**	**0,05**	**0,27**
$A23$	$1,97$	$1,59$	$1,76$	$1,70$	**0,18**	**0,16**	**0,00**	**0,21**	**0,24**
$A24$	$1,78$	$1,78$	$1,01$	$1,14$	**0,03**	**0,05**	**0,21**	**0,00**	**0,30**
$A25$	$1,20$	$1,42$	$1,59$	$1,59$	**0,29**	**0,27**	**0,24**	**0,30**	**0,00**

Table 9: Représentation partielle de la matrice des coûts de substitution.

Les coûts de substitution entre un label commençant par "A1" et un autre label débutant par "A1" sont plus petits que ses coûts entre un label commençant par "A1" et un autre label débutant par "A2" (ce qu'on souhaite).

La dissimilarité entre deux séquences labellisées correspond au coût minimal associé aux opérations nécessaires pour transformer une séquence en une autre, donc au minimum de la somme des coûts des changements nécessaires.

On définit ainsi la matrice de dissimilarité Δ. Le terme $\Delta(i, j)$ est la dissimilarité entre les séquences labellisées i et j. Nous illustrons la distribution de cette matrice dans la Figure 67 :

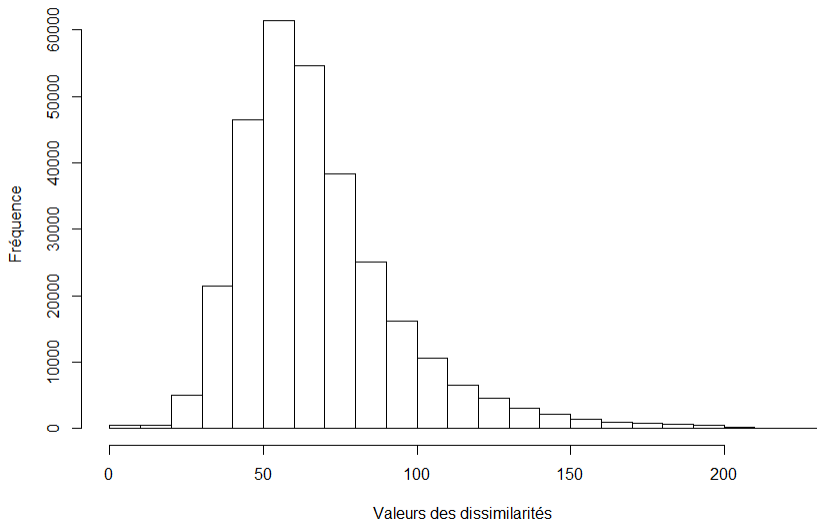


Figure 67: Distribution de la matrice de dissimilarités

Exemple de deux séquences labellisées similaires (la dissimilarité OM entre ces deux séquences est égale à 2,47, ce qui est d'après la Figure 67 très faible) :

"A12 -> S -> A12 -> **A71** -> S -> D21 -> D61 -> A34 -> D61 -> A63 -> **D24** -> S -> D75"

"A12 -> S -> A12 -> **A24** -> S -> D21 -> D61 -> A34 -> D61 -> A63 -> **D27** -> S -> D75"

A partir de la matrice Δ, nous pouvons identifier la séquence labellisée et par conséquent le vol associé le plus "représentatif" comme nous avons déterminé la courbe représentative d'une superclasse dans la section 5.1. Le vol représentatif est illustré dans la Figure 68. On utilise un code couleur pour caractériser le type de phase sur le graphe:

- Noir : Phase stabilisée
- Rouge : Phase transitoire croissante
- Bleu : Phase transitoire décroissante

L'allure du vol semble normale. On observe quelques variations de très faible amplitude dans la phase de croisière.

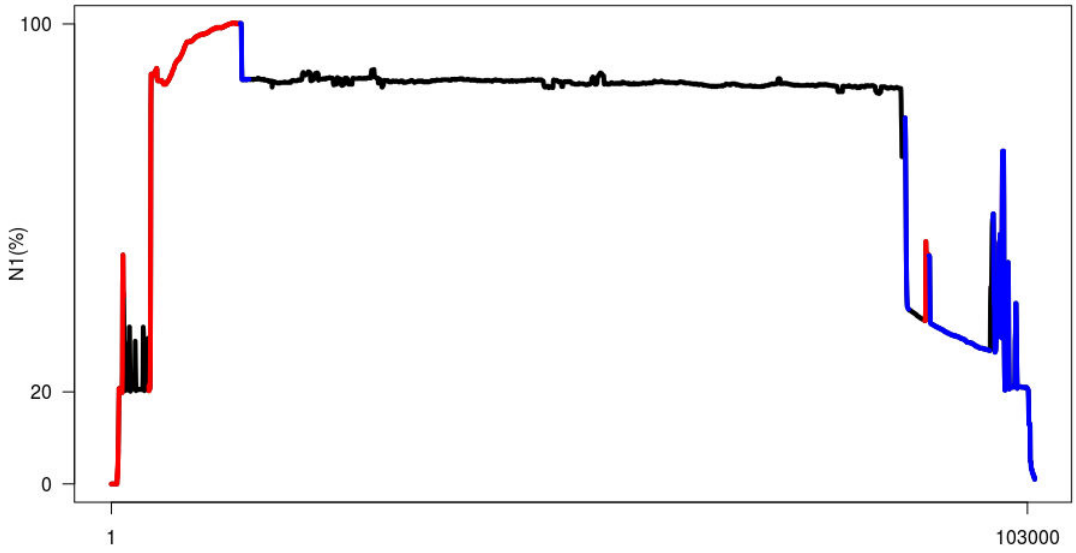


Figure 68: Vol représentatif

6.3.1 SOM relationnel appliqué sur les séquences labellisées

On souhaite maintenant trouver une typologie des vols de notre base de données. Pour cela, et à partir de la matrice de dissimilarités Δ, nous construisons une nouvelle carte de Kohonen en utilisant cette fois la version relationnelle de l'algorithme.

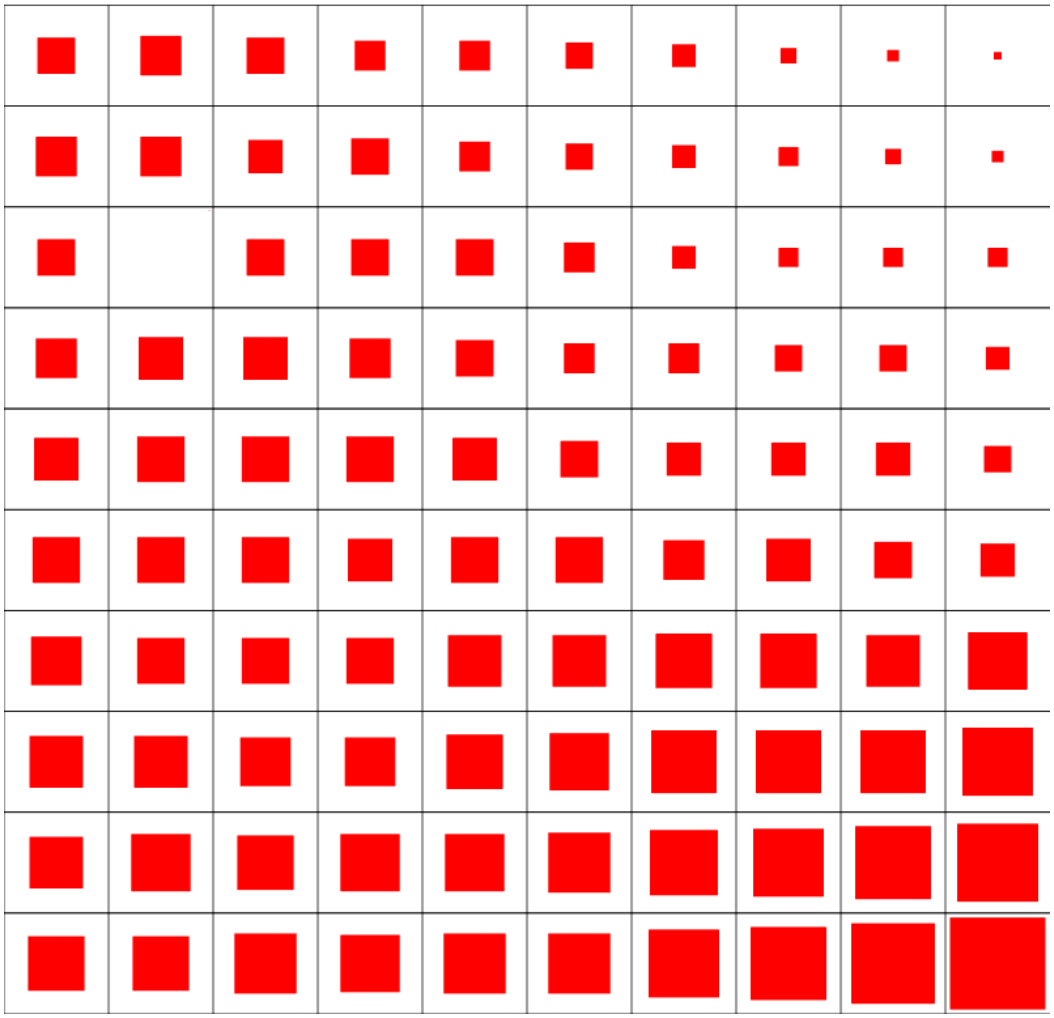


Figure 69: Distribution de la moyenne des longueurs des séquences labellisées par cluster pour une carte 10x10.

La Figure 69 représente une carte de Kohonen 10x10 où est représentée dans chaque classe la moyenne des longueurs des séquences labellisées. Plus le carré rouge dans la classe est grand, plus les séquences labellisées composant la classe ont un nombre de labels est élevée. On remarque qu'en haut à droite de la carte se situent des classes ayant une longueur de séquences très petite et en bas à droite de la carte des classes ayant une longueur moyenne très élevée.

De la même manière, nous souhaitons observer la densité des labels A, D et S dans chaque classe. Nous représentons dans la Figure 70, la densité du label D (la densité des labels A et S sont présentés en annexe).

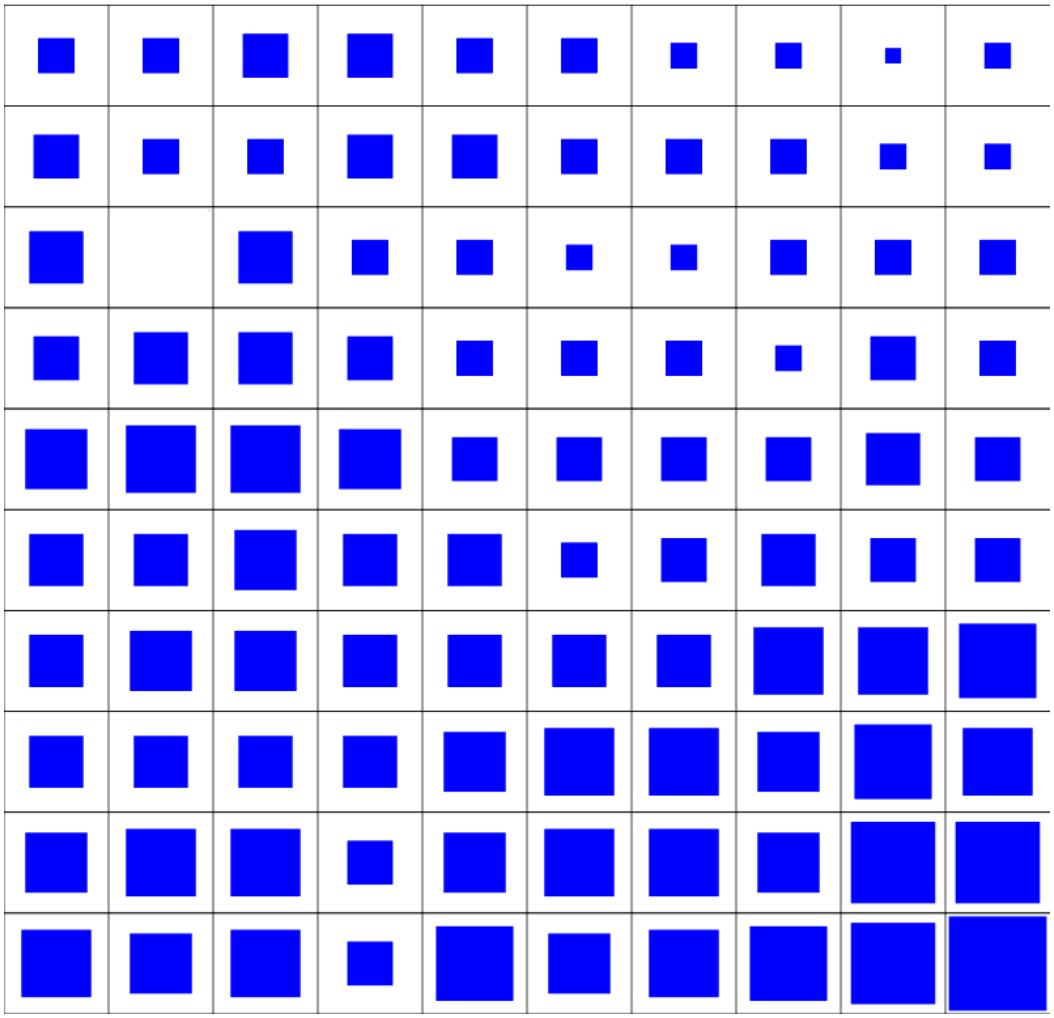


Figure 70: Distribution de la proportion du label D par classe pour une carte 10x10

Cette carte est assez similaire à celle de la Figure 69. En effet, on comprend aisément que les séquences labellisées de petite longueur contiennent moins de labels D que celles qui sont plus longues.

On souhaite maintenant regrouper les classes de Kohonen en un petit nombre de superclasses, grâce à une classification hiérarchique ascendante (CHA).

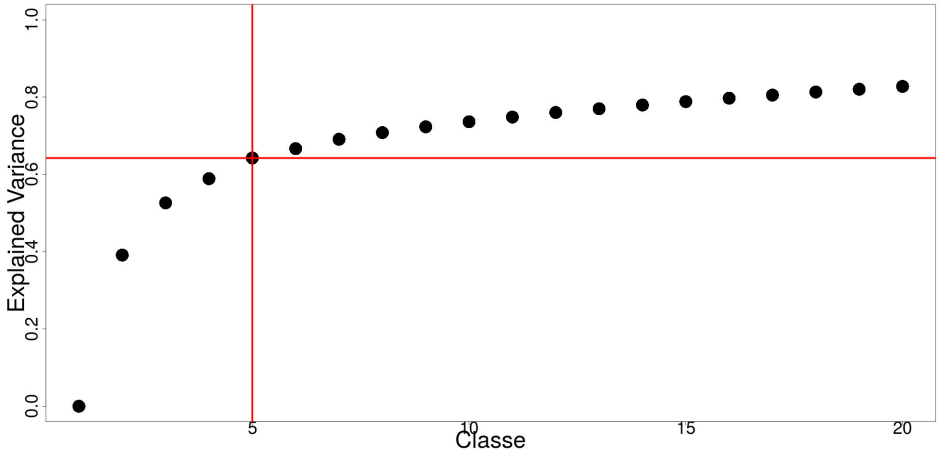


Figure 71: Pourcentages de l'inertie expliquée des séquences labellisées obtenue par la méthode CHA en fonction du nombre de classes.

Au vu du pourcentage d'inertie expliquée de la Figure 71, on fixe le nombre de superclasses égal à 5, pour un pourcentage de variance expliquée supérieure à 60%.

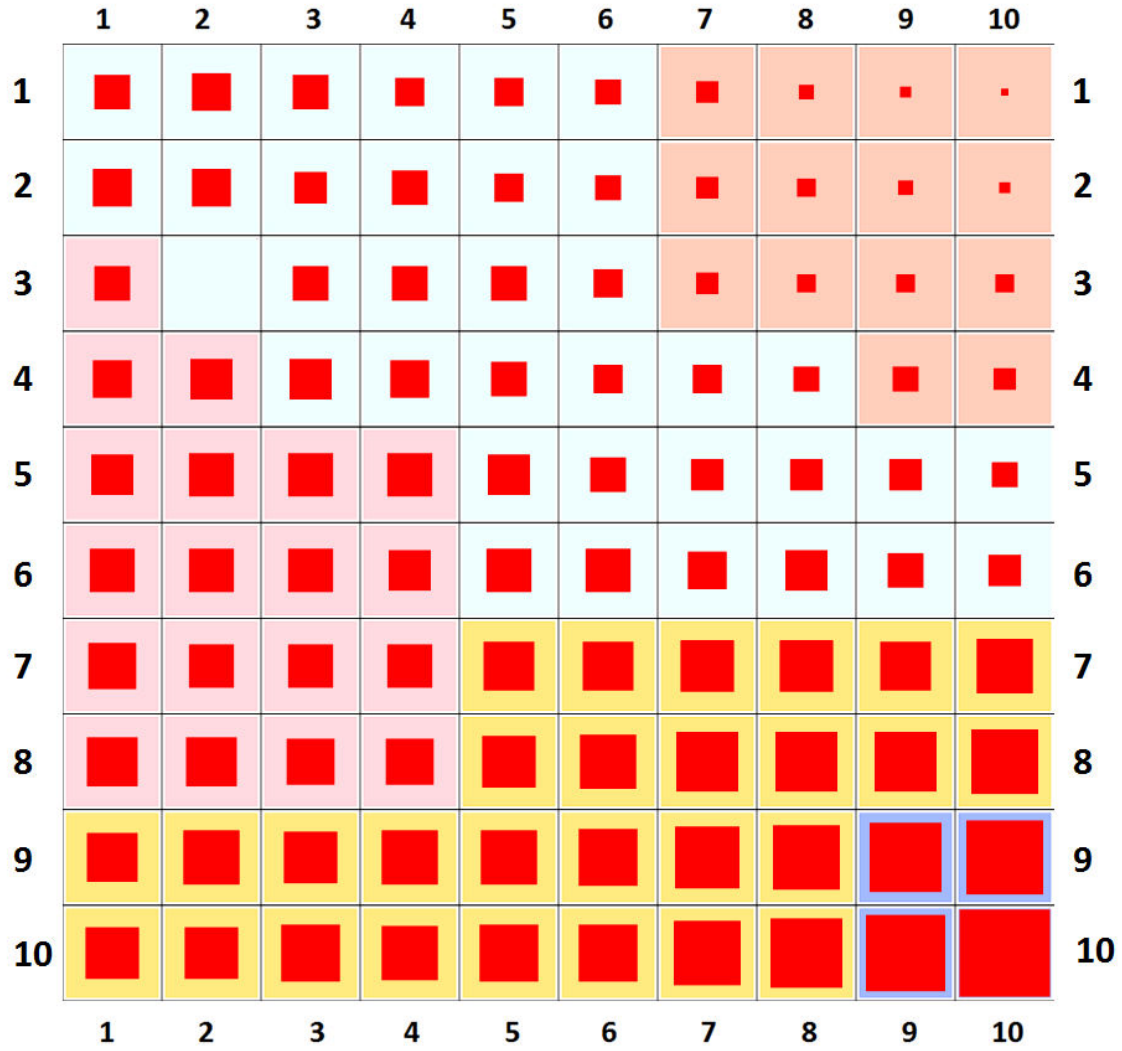


Figure 72: Distribution de la moyenne des longueurs des séquences labellisées par classe pour une carte 10x10 avec les 5 superclasses représentées en couleur de fond

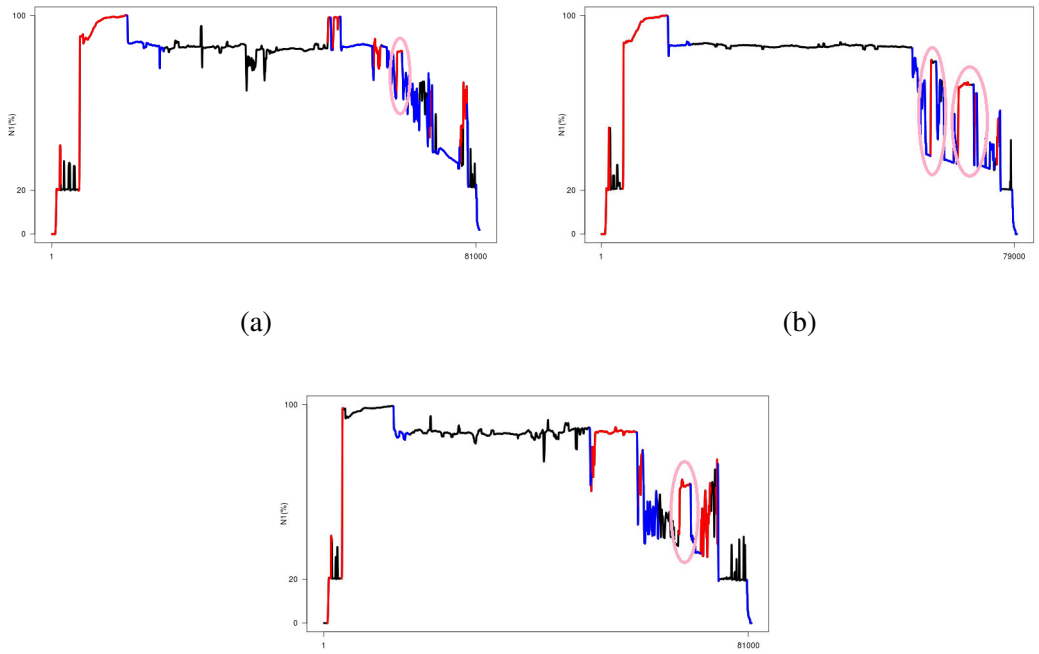
Dans la Figure 72, deux superclasses se distinguent. La superclasse bleue (en bas à droite) contient des séquences de longueurs moyennes très grandes et celle en rouge est composée de séquences de longueurs moyennes très petite.

Pour mieux identifier les superclasses, on calcule les moyennes et écarts-type des densités des labels A, D, S.

Superclasse	A	Ecart-type A	D	Ecart-type D	S	Ecart-type S
1	10,10	1,71	10,96	1,68	7,29	2,74
2	8,53	1,25	9,74	1,29	5,16	2,02
3	6,79	1,33	8,08	1,17	5,05	1,91
4	5,07	1,24	6,89	1,15	3,04	1,55
5	14,04	2,66	14,46	2,94	9,50	2,45

On observe à nouveau que deux superclasses se détachent. Dans la superclasse 4, les densités des trois labels sont très faibles (en rouge dans la Figure 72) et dans la superclasse 5, les densités sont au contraire très élevées (en bleu dans la Figure 72).

Nous passons maintenant à la représentation des vols associés aux séquences labellisées dans certaines classes. Dans les Figures 73 et 74, on décrit les vols associés aux séquences labellisées appartenant aux classes (1,2) et (6,2). On reprend le même code couleur que celui utilisé pour la représentation du vol représentatif.

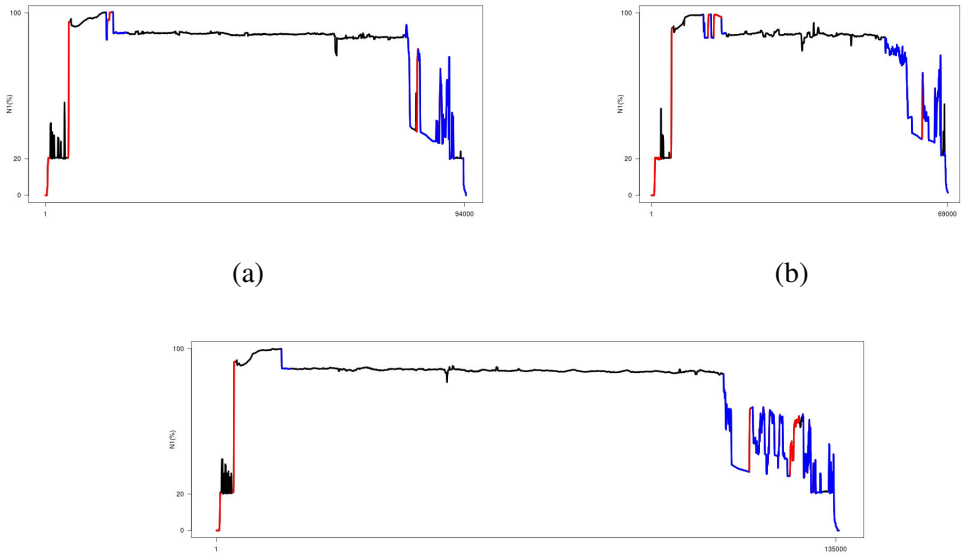


(a) (b)

(c)

Figure 73: Représentation de trois vols associés aux séquences labellisées de la classe de Kohonen (1,2).

Dans la Figure 73 représentant trois vols de la classe (1,2), on observe que les phases de *climb* de ces trois vols présentent des caractéristiques similaires et que les phases de croisière sont très bruitées dans l'image (a) et (c) et stabilisées dans la figure (b). On note également que les descentes comportent des variations de fortes amplitudes. Entourées en rose, on observe les formes récurrentes avec le code couleur dans les trois vols.



(a) (b)

(c)

Figure 74: Représentation de trois vols associés aux séquences labellisées appartenant à la classe de Kohonen (6,2).

Dans la Figure 74 représentant la classe (6,2), on observe que dans deux des trois graphiques, les phases de croisière sont quasi-stabilisées. Cependant sur un des trois, la phase de croisière est composée de variations de faible amplitude. On remarque toutefois que dans chacune des figures, il y a au moins une petite accroche vers le bas. Les phases de *climb* sont assez similaires.

6.4 Identification des vols "volages"

6.4.1 Méthodologie

Comme on sait que l'algorithme de Kohonen est stochastique et que son résultat n'est pas unique, on suivra la démarche de De Bodt dans (de Bodt, Cottrell, & Verleysen, 2002), pour étudier la fiabilité des cartes de Kohonen. Cette méthode a également été reprise par Bourgeois *et al.* dans (Bourgeois, Cottrell, Déruelle, Lamassé, & Letrémy, 2015) afin de déterminer les associations volages (instables). De Bodt considèrent plusieurs exécutions de l'algorithme SOM pour un même jeu de données et une même taille de grille. Ils analysent ensuite les paires d'observations qui sont (presque) toujours voisines, celles qui ne le sont (presque) jamais et celles dont l'association paraît aléatoire (c'est ce qu'on appelle les paires volages).

Rappelons rapidement la démarche de De Bodt *et al.*. On considère qu'on dispose de N observations, $(x_1, \cdots, x_N)$ appartenant à $\mathcal{X}$. Pour une taille de grille

donnée, on construit R cartes de Kohonen avec des initialisations aléatoires. Pour chaque carte et pour chaque paire d'observations x_i et x_j, on définit une mesure de voisinage: $VOISIN^r_{i,j} = 0$ si x_i et x_j ne sont pas voisins directs durant la r-ème exécution de l'algorithme et $VOISIN^r_{i,j} = 1$ s'ils le sont. On note $V_{i,j} = \sum_{r=1}^{R} VOISIN^r_{i,j}$, le nombre de fois où x_i et x_j sont voisins directs durant les R exécutions de l'algorithme. On définit ensuite un indice de stabilité $\mathcal{M}_{i,j} = \frac{V_{i,j}}{R}$ afin de quantifier la moyenne de $VOISIN_{i,j}$ sur l'ensemble des exécutions de l'algorithme.

Un test statistique est ensuite utilisé pour vérifier la stabilité de $\mathcal{M}_{i,j}$. Si U le nombre d'unités de la grille et si on ne tient compte des effets de bord sur une carte bi-dimensionnelle, on constate que la zone de voisinage directe est composée de 9 unités (8 unités voisines plus le centre). Par conséquent, si l'association entre x_i et x_j était complètement aléatoire alors la probabilité que x_i et x_j soient voisins directs serait de $\frac{9}{U}$.

Comme $V_{i,j} = \sum_{r=1}^{R} VOISIN^r_{i,j}$ est le nombre de fois où x_i et x_j sont voisins durant les R exécutions indépendantes, il est aisé de voir que sous l'hypothèse d'association aléatoire, $V_{i,j}$ serait distribuée selon une loi binomiale de paramètres R et $\frac{9}{U}$. En approximant la loi binomiale par une loi gaussienne (car R est grand et $\frac{9}{U}$ pas trop petit), on peut établir la région de rejet du test pour les hypothèses H_0: "L'association de x_i et x_j paraît aléatoire" contre H_1: "x_i et x_j sont voisins ou éloignés".

La région de rejet du test sur $V_{i,j}$ avec un risque de 5% est donnée par :

$$]-\infty, R\frac{9}{U} - 1.96\sqrt{R\frac{9}{U}(1-\frac{9}{U})}[\ \bigcup \ ]R\frac{9}{U} + 1.96\sqrt{R\frac{9}{U}(1-\frac{9}{U})}, +\infty[.$$

Pour $\mathcal{M}_{i,j}$, la région de rejet est

$$]-\infty, \frac{9}{U} - 1.96\sqrt{\frac{9}{UR}(1-\frac{9}{U})}[\ \bigcup \ ]\frac{9}{U} + 1.96\sqrt{\frac{9}{UR}(1-\frac{9}{U})}, +\infty[.$$

Pour simplifier les notations, posons

$$A = \frac{9}{U} \quad \text{et} \quad B = 1.96\sqrt{\frac{9}{UR}(1-\frac{9}{U})}.$$

En pratique, pour chaque paire (x_i, x_j), on calcule l'indice $\mathcal{M}_{i,j}$ et on applique la procédure suivante:

- Si l'indice est supérieur à $A + B$ alors x_i et x_j sont (presque) toujours voisins de manière significative, la paire est attractive.
- Si l'indice est inférieur à $A - B$ alors x_i et x_j ne sont (presque) jamais voisins de manière significative, la paire est répulsive.

- Si l'indice est compris entre $A - B$ et $A + B$ alors l'association entre x_i et x_j est aléatoire, la paire est volage.

A partir de ces résultats, on calcule des pourcentages de paires attractives, de paires répulsives et de paires volages. On peut ensuite identifier les observations les plus volages en utilisant un seuil (une observation est dite volage si elle appartient à un nombre important de paires volages).

6.4.2 Robustesse des résultats

Intéressons nous maintenant à la robustesse des résultats du *clustering* des séquences labellisées et à l'existence éventuelle de vols plutôt volages, difficilement attribuable à un *cluster* dans la base de données. Comme précédemment expliqué, on exécute 100 fois l'algorithme SOM relationnel et on impose une taille 10x10. On calcule ensuite les pourcentages de paires attractives, répulsives et volages (voir Table 10).

Paires attractives (%)	Paires répulsives (%)	Paires volages (%)
30,02	58,09	11,8

Table 10: Résultats des pourcentages des paires de classes attractives, répulsives et volages de la carte de Kohonen.

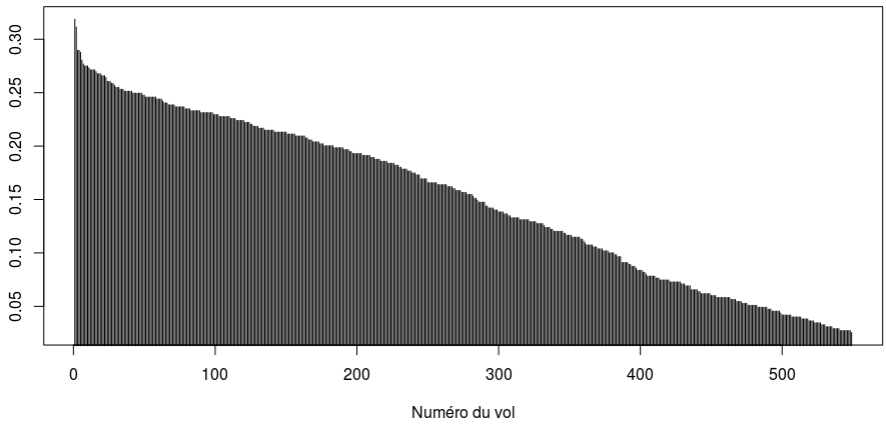


Figure 75: Séquences labellisées réordonnées selon le nombre de paires volages auxquelles elles appartiennent

On identifie ensuite les séquences labellisées/vols les plus volages. Ils se trouvent dans les classes situées au bord de la carte de la Figure 72. Nous les représentons dans les Figures 76 à 79 (le reste des vols volages se trouvent en annexe). Toutes

les figures ont été mises à l'échelle de manière à ce que leur taille soit proportionnelle à la durée du vol. Pour nous aider dans l'interprétation, l'altitude est également représentée.[3]

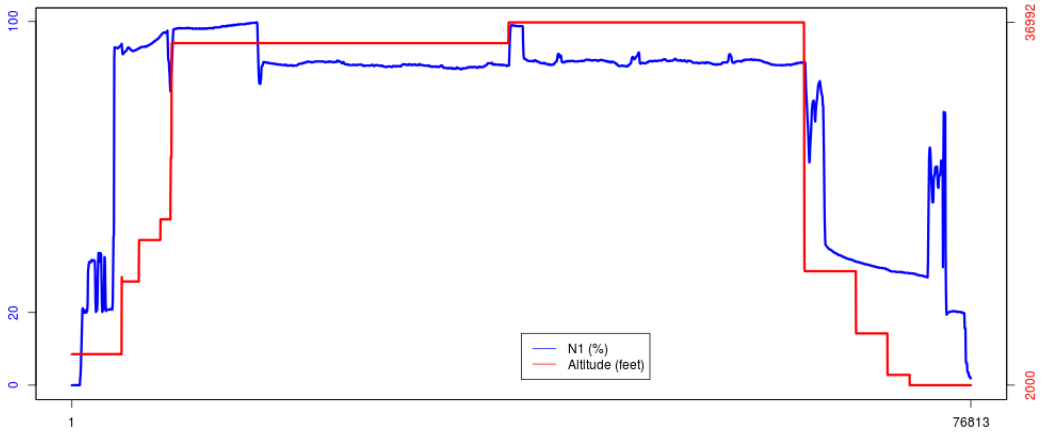


Figure 76: Vol volage (Exemple 1)

La Figure 76 a une allure assez similaire au vol représentatif (voir la Figure 68). Toutefois la phase de *climb* est très longue et elle contient une petite variation inhabituelle durant le *climb*.

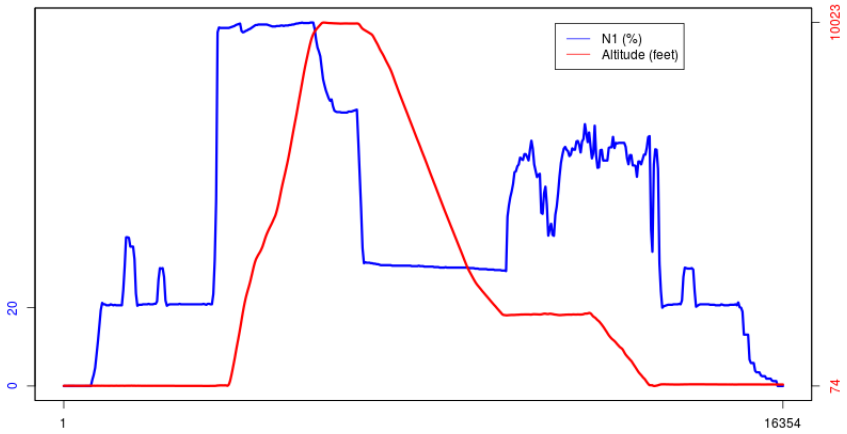


Figure 77: Vol volage (Exemple 2)

Dans la Figure 77, on remarque qu'il s'agit d'un vol très court mais qui s'est

[3]Il est possible que la fréquence des données d'altitude ne soit pas la même d'un vol à l'autre. C'est pourquoi certaines altitudes ont une allure en escalier.

déroulé correctement bien que l'allure de la courbe N1 semble atypique.

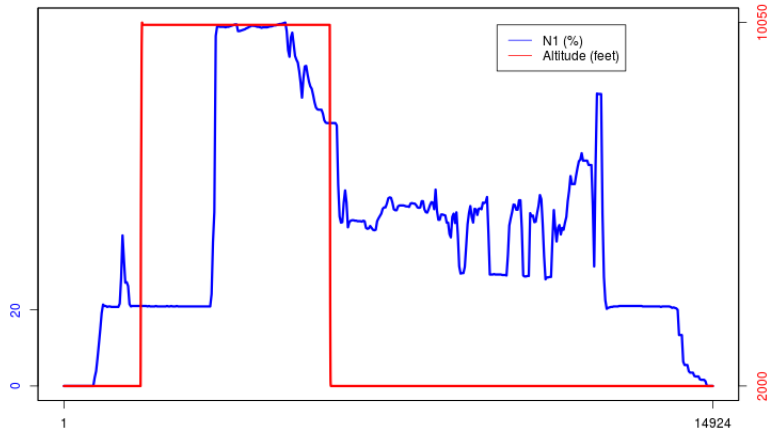


Figure 78: Vol volage (Exemple 3)

Pour l'exemple 3 (Figure 78), les informations concernant l'altitude ne coïncident pas avec l'allure de la courbe en N1. On ne peut conclure.

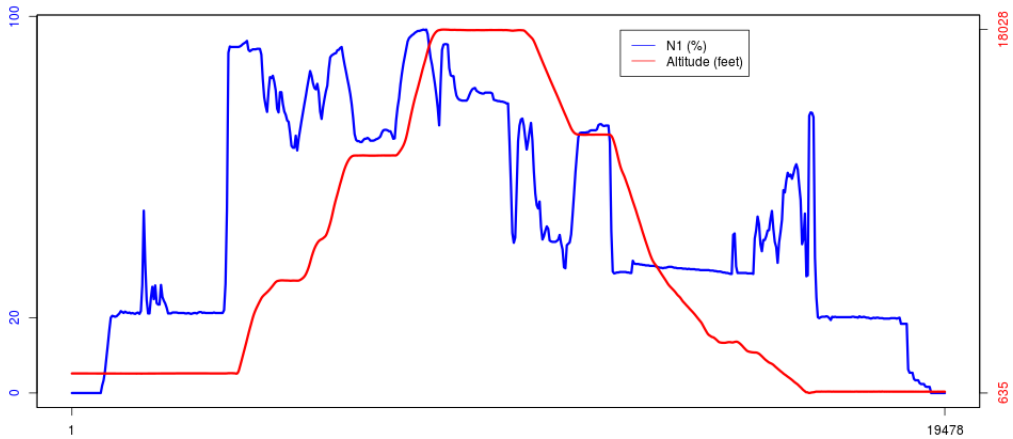


Figure 79: Vol volage (Exemple 4)

Dans la Figure 79, le vol semble s'être déroulé normalement d'après l'altitude, bien que la montée s'est effectuée par niveaux.

Les Figures 76 à 79 illustrent des vols tout à fait atypiques. L'ensemble des vols volages ont été soumis aux experts métier pour analyse.

6.5 Conclusion

Après avoir transformé chaque vol en une suite de labels, nous avons mis en place un outil permettant l'analyse de ces séquences labellisées. Grâce à la méthode *optimal matching* et à la définition de coûts adaptés aux données, nous avons créé une matrice de dissimilarités entre les séquences labellisées.

Ensuite nous avons construit une carte de Kohonen associée à cette matrice. L'analyse des classes de cette carte a montré que chaque classe regroupe des séquences labellisées similaires ainsi que les vols qui leur sont associés.

Le calcul du pourcentage de paires volages et l'identification de ces vols volages a permis de répertorier plusieurs vols ayant une allure atypique.

Ainsi la méthode que nous avons appliquée, nous permet d'analyser des séquences de vols, d'identifier des vols similaires et de repérer les vols atypiques.

7 Conclusions et perspectives

7.1 Conclusions

Dans l'introduction de ce manuscrit, nous avons rappelé l'importance des détections précoces d'anomalies dans le fonctionnement des moteurs d'avions grâce à des capteurs, en mettant en évidence leurs éventuelles conséquences logistiques, financières et même en matière de réputation des compagnies aériennes. En effet une anomalie détectée tardivement ou non détectée peut être coûteuse ou amener à des retards. Nous avons rapidement résumé les méthodes employées par *Safran Aircraft Engines*, notamment en *Prognostic Health Monitoring*, et basées essentiellement sur le savoir-faire métier et sur l'apprentissage automatique. Le travail effectué dans cette thèse afin d'aider les experts dans l'exploitation des données consiste à les résumer automatiquement par une suite de labels, en détectant les éventuelles anomalies et en créant un outil permettant de rechercher rapidement des courbes de vols similaires à une courbe donnée.

Dans un premier temps, nous avons décidé de repérer les phases transitoires et stabilisées dans les séries temporelles monovariées et de nous focaliser sur les phases transitoires. Pour cela, plusieurs méthodes ont été utilisées (OP, PELT, SEP et *online*) et les algorithmes PELT-BIC et *online* ont tous deux de bons résultats pour découper chaque série temporelle en une succession de phases transitoires et stabilisées. Toutefois l'algorithme PELT-BIC représentait le meilleur compromis du point de vue des critères de performance. Ainsi nous avons mis en place un outil permettant aux experts de repérer les phases transitoires et stabilisées avec une complexité linéaire. Ce travail a donné lieu à un article à ESANN 2016 (Faure, Bardet, et al., 2016).

Ensuite, le travail de thèse nous a conduit à classer l'ensemble des phases transitoires (croissantes et décroissantes) dans un contexte bivarié en fonction des caractéristiques de leur évolution en construisant des cartes de Kohonen (SOM numérique) et des superclasses de niveau 1 (vitesse de rotation N1) et de niveau 2 (température avant combustion T3). Par cette méthodologie, nous avons proposé un outil permettant d'analyser les comportements récurrents et non récurrents des phases bivariées dans une même superclasse de niveau 1. L'ensemble de la méthodologie fait l'objet de deux articles (Faure, Olteanu, Bardet, & Lacaille, 2017) et (Bardet, Faure, Lacaille, & Olteanu, 2017).

Pour permettre aux experts métier de repérer des phases/courbes atypiques dans les données, un autre outil a été développé à partir de tubes de confiance calculés pour chacune des superclasses. En utilisant les propriétés des cartes de Kohonen, les experts pourront également rechercher rapidement des phases/courbes similaires à une phase/courbe donnée.

Finalement, tout le travail mené durant la thèse a permis de décrire chaque vol comme une suite de labels. Grâce à la méthode *optimal matching* et à la définition des coûts de substitution à partir des cartes de Kohonen comme évoquées précédemment, nous avons ainsi établi une dissimilarité entre les séquences labellisées. L'algorithme SOM relationnel a débouché sur l'élaboration d'une nouvelle carte de Kohonen dont chaque classe est composée de séquences labellisées. Nous avons ainsi créé une méthodologie nouvelle permettant aux experts de repérer des vols similaires grâce au *clustering* et des vols atypiques grâce à l'analyse des vols volages.

7.2 Perspectives

7.2.1 Perspectives industrielles

En premier lieu, il conviendrait bien évidemment d'appliquer l'ensemble de ces méthodes aux données de bancs d'essai et d'en analyser les résultats. Les données d'essai sont différents des données de vol (par exemple il n'y a aucun contexte météorologique). De plus durant les essais, le moteur est souvent exploité au maximum de sa capacité, ce qui engendre des formes de courbes atypiques, intéressantes à analyser. De plus, les informations concernant les manipulations appliquées au moteur sont également disponibles.

Suite à ce travail de thèse, il est prévu de délivrer l'ensemble des algorithmes sous forme de *package* afin qu'ils soient utilisables par les experts. Deux projets sont déjà proposés. Tout d'abord, l'entreprise souhaite analyser des vols dans le but de mieux comprendre l'exploitation des moteurs et d'améliorer les durée de vie et optimiser la maintenance. Une nouvelle base de données beaucoup plus fournie que celle que la nôtre sera exploitée et les outils que nous avons développés seront appliqués à cette base de données.

Puis une autre demande d'application sur le bruit des signaux est formulée car quelques experts se sont aperçus que le passage d'un bruit blanc à un bruit coloré dans les séries temporelles est annonciateur d'une anomalie. Ces anomalies peuvent être détectées par le PELT-BIC.

7.2.2 Perspectives méthodologiques

Dans un premier temps, il serait intéressant de réitérer toute notre démarche avec la méthode de détection de ruptures *online*. Nous avons vu que la méthode *online* est plus sensible aux variations de faibles amplitudes et ne dépend pas de la taille des données, contrairement à la méthode PELT. Ces résultats devront être comparés aux résultats actuels. Cette méthode s'applique également sur des données arrivant à la volée.

Dans la section 4.3.2, intitulée "SOM numérique", nous avons choisi de séparer les phases transitoires croissantes des phases transitoires décroissantes pour construire les cartes de Kohonen. Il serait tentant de réduire le nombre de cartes en appliquant l'algorithme SOM numérique sur toutes les phases transitoires confondues (stabilisées, croissantes et décroissantes).

Le choix des caractéristiques définies dans la section 4.3.2 résumant les phases transitoires constituent une étape importante dans la méthodologie. D'autres caractéristiques peuvent être également choisies, notamment celles basées sur les ondelettes comme dans (Popivanov & Miller, 2002), où Popivanov *et al.* utilisent les coefficients d'ondelettes pour rechercher des similarités dans les séries temporelles.

Dans la section 5.1, intitulée "Méthodologie" du chapitre 5 "*Clustering* et analyse des phases bivariées", la sélection de la courbe représentative pourrait se faire différemment. Nous pouvons également utiliser les vecteurs de caractéristiques pour déterminer cette courbe qui a un rôle crucial dans le *clustering* de niveau 2.

8 Bibliographie

References

Abbott, A., & Forrest, J. (1986). Optimal matching methods for historical sequences. *The Journal of Interdisciplinary History*, *16*(3), 471–494.

Abbott, A., & Tsay, A. (2000). *Sequence Analysis and Optimal Matching Methods in Sociology: Review and Prospect* (Vol. 29) (No. 1).

Adams, R. P., & MacKay, D. J. (2007). Bayesian online changepoint detection. *arXiv preprint arXiv:0710.3742*.

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. *2nd International Symposium on Information Theory*, 267–281.

Anderson, J. a., & Grimes, T. (2008). A Novel Approach to Bayesian Online Changepoint Detection. *American Behavioral Scientist*, *51*(8), 1059–1060.

Arlot, S., & Massart, P. (2009). Data-driven calibration of penalties for least-squares regression. *The Journal of Machine Learning Research*, *10*, 245–279.

Aronszajn, N. (1950). Theory of reproducing kernels. *Transactions of the American Mathematical Society*, *68*(3), 337–337.

Bai, J., & Perron, P. (1998). Estimating and Testing Linear Models with Multiple Structural Changes. *Econometrica*, *66*(1), 47–78.

Baragona, R. (2001). A simulation study on clustering time series with metaheuristic methods. *Quaderni di Statistica*, *3*, 1–26.

Bardet, J.-M., Faure, C., Lacaille, J., & Olteanu, M. (2017). Design Aircraft Engine Bivariate Data Phases using Change-Point Detection Method and Self-Organizing Maps. In *Itise 2017*.

Bardet, J.-M., Kengne, W., & Wintenberger, O. (2012). Multiple breaks detection in general causal time series using penalized quasi-likelihood. *Electronic Journal of Statistics*, *6*, 435–477.

Basseville, M., & Nikiforov, I. V. (1993). Detection of Abrupt Changes. *Change*, *2*(4), 729–730.

Basseville, M., Nikiforov, I. V., et al. (1993). *Detection of abrupt changes: theory and application* (Vol. 104). Prentice Hall Englewood Cliffs.

Baudry J-P., M. C., & B., M. (2010). Slope heuristics: overview and implementation. *RR-INRIA*(7223).

Beckmann, N., Kriegel, H.-P., Schneider, R., & Seeger, B. (1990). The R*-tree: an efficient and robust access method for points and rectangles. *ACM SIGMOD Record*, *19*(2), 322–331.

Belanger, G. (2013). On detecting transient phenomena. *The Astrophysical Journal*, *773*(1), 66.

Bellas, A., Bouveyron, C., Cottrell, M., & Lacaille, J. (2012). Robust Clustering of High-Dimensional Data. In *Esann* (pp. 25–27).

Bellas, A., Bouveyron, C., Cottrell, M., & Lacaille, J. (2013). Model-based clustering of high-dimensional data streams with online mixture of probabilistic PCA.

Advances in Data Analysis and Classification, *7*(3), 281–300.

Bellas, A., Bouveyron, C., Cottrell, M., & Lacaille, J. (2014). *Anomaly Detection Based on Confidence Intervals Using SOM with an Application to Health Monitoring* (Vol. 295).

Berman, P. (1978). *The study of macro and micro implementation of social policy* (Vol. 26) (No. 2).

Berndt, D., & Clifford, J. (1994). Using dynamic time warping to find patterns in time series. *Workshop on Knowledge Knowledge Discovery in Databases*, *398*, 359–370.

Bourgeois, N., Cottrell, M., Déruelle, B., Lamassé, S., & Letrémy, P. (2015). How to improve robustness in Kohonen maps and display additional information in Factorial Analysis: Application to text mining. *Neurocomputing*, *147*(1), 120–135.

Caiado, J., Crato, N., & Peña, D. (2009). Comparison of Times Series with Unequal Length in the Frequency Domain. *Communications in Statistics - Simulation and Computation*, *38*(3), 527–540.

Camacho, M., Perez-Quiros, G., & Saiz, L. (2006). Are European business cycles close enough to be just one? *Journal of Economic Dynamics and Control*, *30*(9-10), 1687–1706.

Come, E. (2011). Aircraft engine fleet monitoring using Self-Organizing Maps and Edit Distance. *Machine Learning*, 298–307.

Comon, P. (1994). Independent component analysis, A new concept? *Signal Processing*, *36*(3), 287–314.

Cottrell, M., Gaubert, P., Eloy, C., François, D., Hallaux, G., Lacaille, J., & Verleysen, M. (2009). Fault Prediction in Aircraft Engines Using Self-Organizing Maps. In *Advances in self-organising maps* (pp. 37–44).

Cottrell, M., Olteanu, M., Rossi, F., & Villa-Vialaneix, N. (2016). Theoretical and applied aspects of the self-organizing maps. In *Advances in self-organizing maps and learning vector quantization* (pp. 3–26). Springer.

Crathorne, A. R., & Shewhart, W. A. (1933). Economic Control of Quality of Manufactured Product. *The American Mathematical Monthly*, *40*(6), 353.

Darkhovski, B. S. (1994). Nonparametric Methods in Change-Point Problems: A General Approach and Some Concrete Algorithms. *Lecture Notes-Monograph Series*, *23*(1993), 99–107.

Davis, R. A., Lee, T. C., & Rodriguez-Yam, G. A. (2006). Structural break estimation for nonstationary time series models. *Journal of the American Statistical Association*, *101*, 223-239.

Davis, R. A., & Yau, C. Y. (2013). Consistency of minimum description length model selection for piecewise stationary time series models. *Electronic Journal of Statistics*, *7*(1), 381–411.

de Bodt, E., Cottrell, M., & Verleysen, M. (2002). Statistical tools to assess the reliability of self-organizing maps. *Neural Netw*, *15*(8-9), 967–978.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B*, *39*(1), 1–38.

Dunn, J. C. (1973). A fuzzy relative of the ISODATA process and its use in detecting compact well-separated clusters. *Journal of Cybernetics*, *3*(3), 32–57.

Faloutsos, C., & Faloutsos, C. (1994). Fast subsequence matching in time-series databases. *ACM SIGMOD Record*, *23*(2), 419–429.

Faure, C., Bardet, J.-M., Olteanu, M., & Lacaille. (2016). Comparison of three algorithms for parametric change-point detection. In *Esann 2016 proceedings* (pp. 2–7). Bruges (Belgium).

Faure, C., Bardet, J.-M., Olteanu, M., & Lacaille. (2017). Using self-organizing maps for clustering and labelling aircraft engine data phases. In *Wsom 2016.*

Faure, C., Lacaille, J., Bardet, J.-M., & Olteanu, M. (2016). Indexation of Bench Test and Flight Data. In *Phm society.*

Faure, C., Olteanu, M., Bardet, J. M., & Lacaille, J. (2017, June). Using self-organizing maps for clustering anc labelling aircraft engine data phases. In *2017 12th international workshop on self-organizing maps and learning vector quantization, clustering and data visualization (wsom)* (p. 1-8).

Forgy, E. W. (1965). Cluster analysis of multivariate data: efficiency versus interpretability of classifications. *Biometrics*, *21*(3), 768–769.

Gabadinho, A., Ritschard, G., Muller, N. S., & Studer, M. (2011). Analyzing and visualizing state sequences in R with TraMineR. *Journal of Statistical Software*, *40*(4), 1–37.

Gao, S., He, Y., & Chen, H. (2009). Wind speed forecast for wind farms based on ARMA-ARCH model. In *1st international conference on sustainable power generation and supply, supergen '09.*

Girshick, M. A., & Rubin, H. (1952). A Bayes approach to a quality control model. *The Annals of mathematical statistics*, *23*(1), 114–125.

Goldfarb, L. (1984). A unified approach to pattern recognition. *Pattern Recognition*, *17*(5), 575–582.

Gower, J. C., & Ross, G. J. S. (1969). Minimum Spanning Trees and Single Linkage Cluster Analysis. *JOurnal of the Royal Statistical Society*, *18*(1), 54–64.

Guo, C., Jia, H., & Zhang, N. (2008). Time series clustering based on ICA for stock data analysis. In *2008 international conference on wireless communications, networking and mobile computing, wicom 2008.*

Guthery, S. B. (1974). Partition regression. *Journal of the American Statistical Association*, *69*(348), 945–947.

Halpin, B., & Chan, T. W. (1998). Class careers as sequences: An optimal matching analysis of work-life histories. *European Sociological Review*, *14*(2), 111–130.

Harris, L. (1989). *Quality control and industrial statistics . acheson j. duncan, irwin, 1986. number of pages: 1123.* Wiley Online Library.

Hinkley, D. V. (1969). Inference about the intersection in two-phase regression. *Biometrika*, *56*(3), 495–495.

Iglesias, F., & Kastner, W. (2013). Analysis of Similarity Measures in Times Series Clustering for the Discovery of Building Energy Patterns. *Energies*, *6*(2), 579–597.

Jackson, B. e. a. (2005). An algorithm for optimal partitioning of data on an interval. *IEEE Signal Processing Letters*, *12*(2), 105–108.

Jamali, A. S., & Jinlin, L. I. (2006). False Alarm Rates for the Shewhart Control Chart with Interpretation Rules. *Control*, *2006*(1987), 238–241.

Ji, K. H., & Herring, T. A. (2011). Transient signal detection using GPS measurements: Transient inflation at Akutan volcano, Alaska, during early 2008. *Geophysical Research Letters*, *38*(6).

Kalpakis, K., Gada, D., & Puttagunta, V. (2001). Distance measures for effective clustering of ARIMA time-series. *Proceedings 2001 IEEE International Conference on Data Mining*, 273–280.

Kaufman, L., & Rousseeuw, P. J. (1987). *Clustering by means of medoids.*

Keogh, E., & Kasetty, S. (2003). On the Need for Time Series Data Mining Benchmarks: A Survey and Empirical Demonstration. In *Data mining and knowledge discovery* (Vol. 7, pp. 349–371).

Killick, R., Eckley, I. A., & Jonathan, P. (2013). A wavelet-based approach for detecting changes in second order structure within nonstationary time series. *Electronic Journal of Statistics*, *7*(1), 1167–1183.

Killick R., F. P., & Eckley, I. (2012). Optimal detection of changepoints with a linear computational cost. *JASA*, *107*(500), 1590–1598.

Kohonen, T. (2001). *Self-Organizing Maps* (Vol. 30).

Kovavrik, M. (2013). Volatility change point detection using stochastic differential equations and time series control charts.

Krishnamurthy, B. (2012). A Top Down Approach to Enterprise Monitoring Using Change Point Detection. In *2012 annual srii global conference* (pp. 859–862).

Lacaille, J., & Gerez, V. (2011). Online Abnormality Diagnosis for real-time Implementation on Turbofan Engines and Test Cells. *Phm*, 1–9.

Lavielle, M., & Moulines, E. (2000). Least-squares estimation of an unknown number of shifts in a time series. *Journal of Time Series Analysis*, *21*, 33–59.

Lloyd, S. P. (1982). Least Squares Quantization in PCM. *IEEE Transactions on Information Theory*, *28*(2), 129–137.

Lowe, D. G. (1987). Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, *31*(3), 355–395.

MacDonald, D., & Fyfe, C. (2000). The kernel self-organising map. In *{Knowledge-Based} intelligent engineering systems and allied technologies, 2000. proceedings. fourth international conference on* (Vol. 1, pp. 317–320).

Mahalanobis, P. C. (1936). On the generalized distance in statistics. *Proceedings of the National Institute of Sciences (Calcutta)*, *2*, 49–55.

Needleman, S. B., & Wunsch, C. D. (1970). A general method applicable to the search for similiarities in the amino acid sequence of two proteins. *Journal of molecular biology*, *48*(3), 443–453.

Ogden, T., & Parzen, E. (1996). Change-point approach to data analytic wavelet thresholding. *Statistics and Computing*, *6*(2), 93–99.

Olteanu, M., & Villa-Vialaneix, N. (2015). On-line relational and multiple relational SOM. *Neurocomputing*, *147*(1), 15–30.

Page, E. (1954). Control charts for the mean of a normal population. *Journal of the Royal Statistical Society. Series B (Methodological)*, 131–135.

Page, E. S. (1954). Continuous Inspection Schemes. *Biometrika*, *41*(1/2), 100.

Paparrizos, J., & Gravano, L. (2017). Fast and Accurate Time-Series Clustering. *ACM Transactions on Database Systems*, *42*(2), 1–49.

Pearson, K. (1895). Note on Regression and Inheritance in the Case of Two Parents. *Proceedings of the Royal Society of London (1854-1905)*, *58*(-1), 240–242.

Peter, S., Höppner, F., & Berthold, M. R. (2013). Pattern graphs: combining multivariate time series and labelled interval sequences for classification. , 5–18.

Phillips, M. (1969). A survey of sampling procedures for continuous production. *Journal of the Royal Statistical Society. Series A (General)*, 205–228.

Picard, D. (1985). Testing and estimating change-points in time series. *Advances in applied probability*, *17*(4), 841–867.

Popivanov, I., & Miller, R. J. (2002). Similarity Search Over Time-Series Data Using Wavelets. *Proceedings 18th International Conference on Data Engineering (ICDE)*, 212–221.

Rabenoro, T., Lacaille, J., Cottrell, M., & Rossi, F. (2014). Anomaly detection based on indicators aggregation. In *Proceedings of the international joint conference on neural networks* (pp. 2548–2555).

Rani, S., & Sikka, G. (2012). Recent Techniques of Clustering of Time Series Data: A Survey. *International Journal of Computer Applications*, *52*(15), 1–9.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

Roberts, S. W. (1959). Control Chart Tests Based on Geometric Moving Averages. *Technometrics*, *1*(3), 239–250.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464.

Seichepine, N., Ricordeau, J., & Lacaille, J. (2011). Data mining of flight measurements. *AIAA InfoTech*(March), 1–9.

Shao, X., & Zhang, X. (2010). Testing for change points in time series. *Journal of the American Statistical Association*, *105*(491), 1228–1240.

Shiryaev, A. (1961). The problem of the most rapid detection of a disturbance in a stationary process. In *Soviet math. dokl* (Vol. 2).

Sievert, C., Parmer, C., Hocking, T., Chamberlain, S., Ram, K., Corvellec, M., & Despouy, P. (2017). plotly: Create interactive web graphics via 'plotly.js' [Computer software manual]. (R package version 4.7.1)

Studer, M., & Ritschard, G. (2016). What matters in differences between life trajectories: A comparative review of sequence dissimilarity measures. *Journal of the Royal Statistical Society. Series A: Statistics in Society*, *179*(2), 481–511.

Takeuchi, J. I., & Yamanishi, K. (2006). A unifying framework for detecting outliers and change points from time series. *IEEE Transactions on Knowledge and Data Engineering*, *18*(4), 482–492.

Villa-Vialaneix, N., Mariette, J., Olteanu, M., Rossi, F., Bendhaiba, L., & Boelaert, J. (2017). Sombrero: Som bound to realize euclidean and relational outputs [Computer software manual]. (R package version 1.2-2)

Walwer, D., Calais, E., & Ghil, M. (2016). Data-adaptive detection of transient deformation in geodetic networks. *Journal of Geophysical Research: Solid Earth*, *121*(3), 2129–2152.

Warren Liao, T. (2005). *Clustering of time series data - A survey* (Vol. 38) (No. 11).
Wu, L. L. (2000). *Some comments on "sequence analysis and optimal matching methods in sociology: Review and prospect"* (Vol. 29) (No. 1).
Xiong, Y., & Yeung, D. Y. (2004). Time series clustering with ARMA mixtures. *Pattern Recognition*, *37*(8), 1675–1689.
Yang, J., & Leskovec, J. (2011). Patterns of temporal variation in online media. In *Wsdm* (pp. 177–186).

9 Annexes

9.1 Annexe du Chapitre 3

Dans la Figure 80, on représente les résultats du PELT-BIC sur les mêmes séries temporelles restreintes qu'à la phase de croisière.

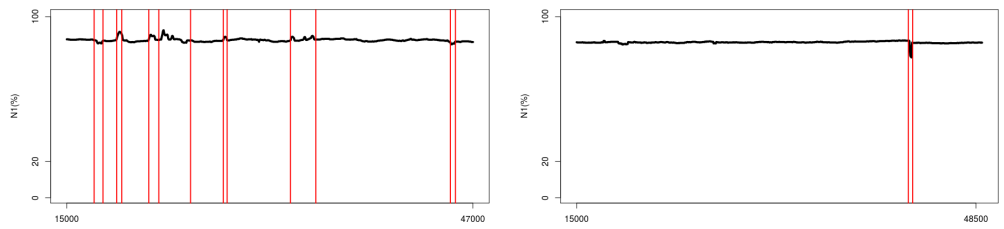


Figure 80: Résultats du PELT-BIC sur les vitesses de rotation N1 restreintes à la croisière (Exemple 1 à gauche et Exemple 2 à droite

Les variations de faibles amplitudes sont bien détectées. Cela est dû à la longueur de la série temporelle (on pénalise moins car la longueur est plus petite).

9.2 Annexes des Chapitres 4 et 5

9.2.1 Difficultés informatiques

Contexte univarié

Pour réaliser les cartes de Kohonen, nous avons utilisé initialement les fonctions graphiques du package SOMbrero (Villa-Vialaneix et al., 2017). Plusieurs types de cartes sont disponibles : densité du nombre d'observations par classe, affichage des prototypes centrés, etc... Cependant le package ne gère pas la visualisation de séries temporelles de longueurs inégales. Or pour les experts, il est important d'identifier l'allure des courbes dans chaque classe de la carte de Kohonen. Un important travail de visualisation des cartes de Kohonen a donc été effectué.

Pour créer ces nouvelles visualisations, nous avons mis en place une fonction graphique permettant de les construire. Cette fonction se base sur la fonction déjà existante dans le package SOMbrero afin que les classes soient construites dans le même ordre (et soient ainsi comparables aux classes d'un autre format de carte).

Les difficultés rencontrées :

- Le premier point important à gérer est l'échelle. Elle doit être adaptée à la longueur maximale des séries temporelles tout en restant limitée pour conserver la visualisation des séries temporelles de longueurs plus petites.
- Pour les experts, il est intéressant d'observer à quel moteur appartient les phases transitoires d'une même classe. Pour cela, nous avons mis en place un code couleur pour chaque moteur (8 au total) et chaque phase transitoire est associée à une couleur.
- Dans chaque classe, les phases transitoires sont calées à 1. Initialement, il était question de synchroniser chaque phase transitoire d'une même classe afin d'avoir une meilleur visualisation de l'allure générale des courbes (dans ce cas, les courbes ne sont pas calées à 1). Cependant, ce format est plus adapté lorsqu'on utilise l'algorithme SOM relationnel plutôt que l'algorithme SOM numérique (car la mesure de dissimilarité permet d'effectuer la synchronisation).

Contexte bivarié

Pour la représentation graphique des superclasses de niveau 2, les besoins visuels ont été différents. En effet, étant donné le contexte bivarié, on peut observer l'allure générale des courbes dans chaque superclasse. Pour un expert, il est intéressant de pouvoir zoomer sur les graphiques et avoir l'identifiant des phases/courbes qu'il juge intéressant à analyser.

C'est pourquoi nous nous sommes aidés du package plotly (Sievert et al., 2017) pour créer des graphiques interactifs et pour faire apparaître l'identifiant à chaque passage de la souris sur une phase/courbe du graphique.

Ce package permet de réaliser des graphiques avec l'option zoom pour chaque superclasse de niveau 2. Ces graphiques peuvent également se sauvegarder tout en conservant ses propriétés interactives. L'inconvénient est la mémoire de tels objets qui sont parfois très volumineux pour une superclasse (jusqu'à 2 Go). Par ailleurs, il reste possible d'avoir des crash de session R lorsque l'allocation mémoire est trop importante. Ces problèmes n'ont pas encore été résolus.

Descriptions des superclasses de phases transitoires croissantes

L'identification des 7 superclasses se fait ensuite au moyen des étiquettes ID_FM des phases transitoires croissantes qu'elles contiennent.

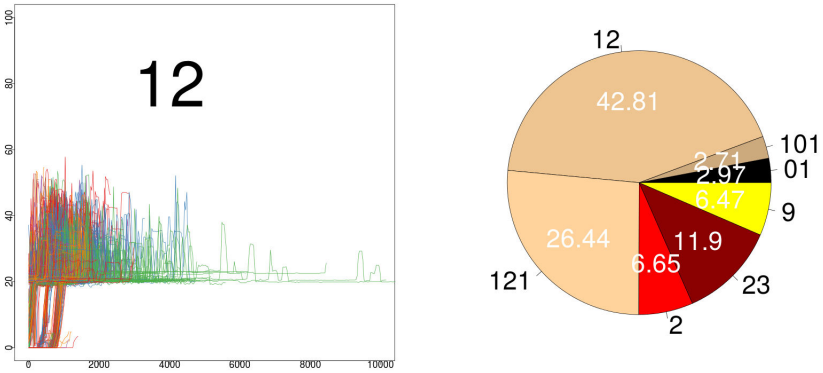


Figure 81: Représentation du contenu de la superclasse 1 et pourcentages des modes de vol.

La première superclasse est composée essentiellement de phases de démarrage moteur (101,01,12 et 121), de taxi (22 et 99) et de taxi précédant le décollage (23). Ceci est cohérent avec les statistiques présentées dans la Table 7 puisque à ce stade du vol l'avion se trouve entre le sol et 600 mètres d'altitude.

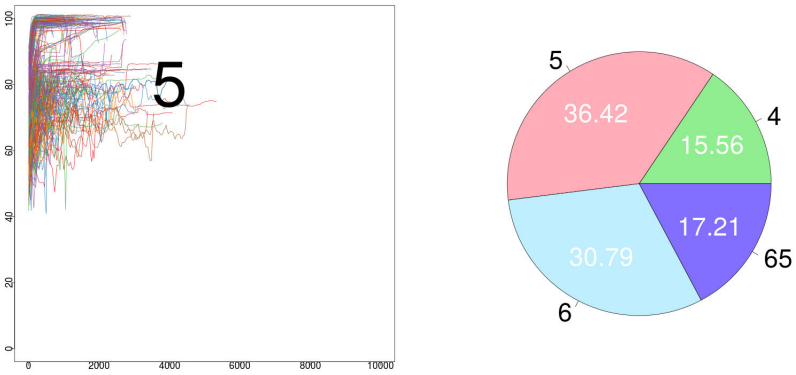


Figure 82: Représentation du contenu de la superclasse 2 et pourcentages des modes de vol.

Dans la deuxième superclasse, on retrouve des phases transitoires se déroulant pendant les modes de (début de) croisière (4 et 5) et de descente (65 et 6). Les phases transitoires croissantes détectées pendant une croisière proviennent en général d'évènements particuliers comme par exemple des turbulences. Le mode croisière est compatible avec l'altitude élevée qu'on a constaté dans la Table 7.

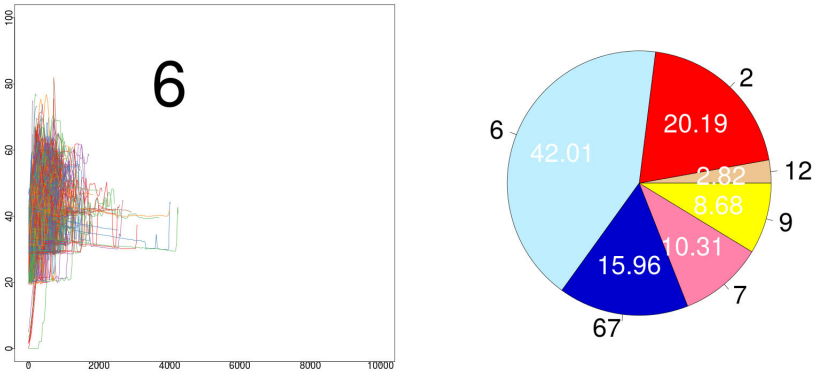


Figure 83: Représentation de la superclasse 3 des phases croissantes et le diagramme en camembert des modes de vols la composant.

La troisième superclasse est assez hétérogène du point de vue des modes de vol. On y trouve des phases de descente et d'approche (66, 67 et 77) ainsi que des phases de taxi (12,22 et 99). Ceci s'explique parce que le comportement de ces phases est semblable, elles sont plutôt courtes avec une forte variabilité (oscillant entre 20% à 60%).

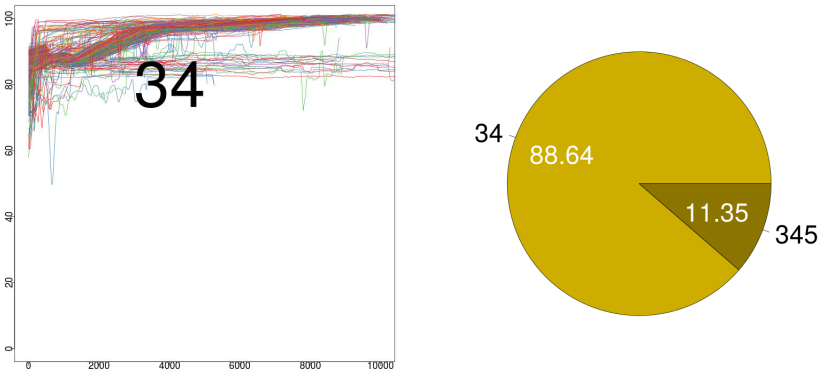


Figure 84: Représentation de la superclasse 4 des phases croissantes et le diagramme en camembert des modes de vols la composant.

Au contraire, la quatrième superclasse ne contient des phases de climb (34) et de fin de *climb*/début de croisière (345). Ces modes de vol correspondent à des valeurs très élevées de N1 et de N2, ce qui est cohérent avec les résultats de la Table 7.

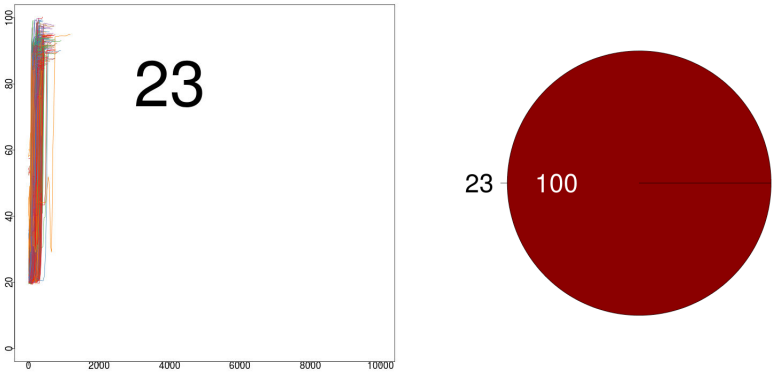


Figure 85: Représentation de la superclasse 5 des phases croissantes et le diagramme en camembert des modes de vol la composant.

La superclasse 5 n'est composée que de phases de décollage (23).

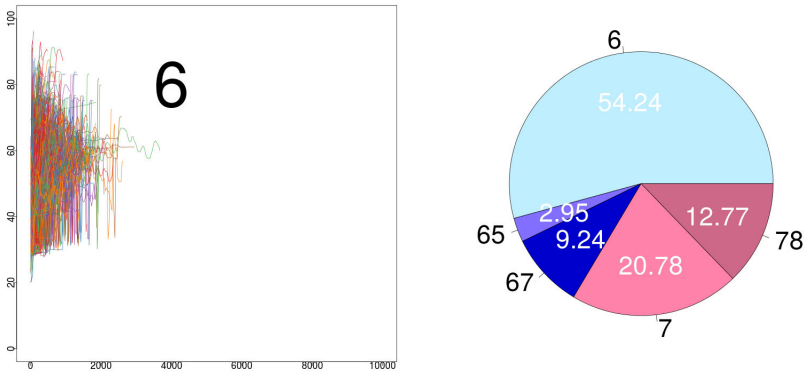


Figure 86: Représentation de la superclasse 6 des phases croissantes et le diagramme en camembert des modes de vol la composant.

Dans la superclasse 6, on retrouve des phases de descente (65, 66 et 67) ainsi que des phases d'approche (77 et 78). Elle ressemble à la superclasse 3 mais l'amplitude des variations est plus importante (entre 20% et 80%).

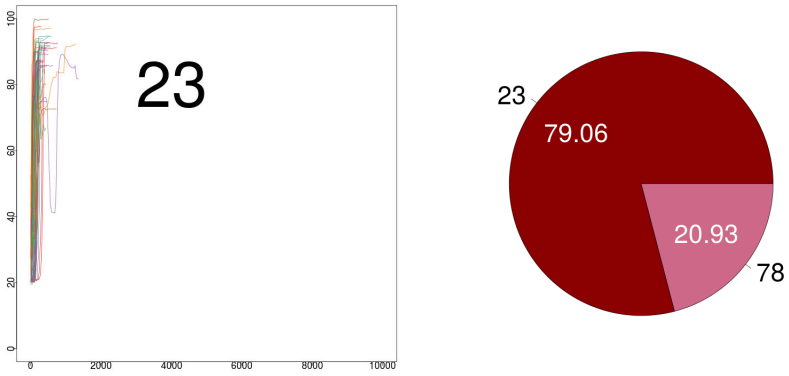
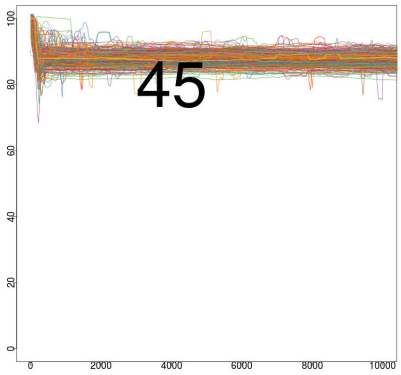


Figure 87: Représentation de la superclasse 7 des phases croissantes et le diagramme en camembert des modes de vol la composant.

La dernière superclasse des phases transitoires croissantes contient 80% de phases de décollage. Les phases d'approche de cette classe correspondent à des instants où le pilote remet les gaz.

Applications aux phases transitoires décroissantes

L'identification des 8 superclasses se fait au moyen des étiquettes ID_FM des phases transitoires décroissantes qu'elles contiennent.



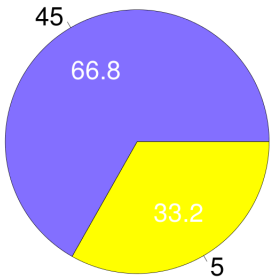
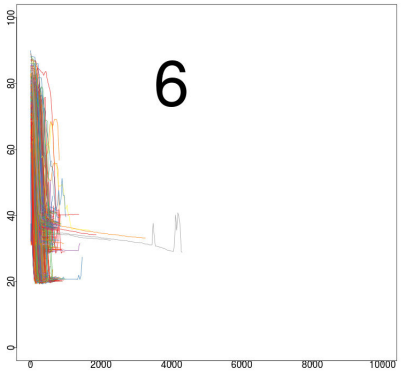


Figure 88: Représentation de la superclasse 1 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La première superclasse est composée de phases de croisière (45 et 5). C'est pourquoi dans la Table 8, l'altitude est élevée.



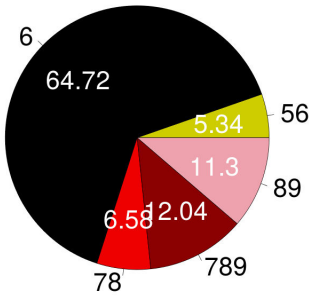


Figure 89: Représentation de la superclasse 2 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La composition des modes de vol de la superclasse 2 est similaire à celle de la superclasse CC_6. Elle est composée de phases ayant des changements abrupts (de 80% à 20%). Ces phénomènes ont généralement lieu durant la descente (56 et 66) ainsi que durant l'approche (78 et 789) et les phases d'atterrissage (89). D'après la Table 8, l'altitude n'est pas très élevée ce qui signifie que ces phases se déroulent non loin du sol.

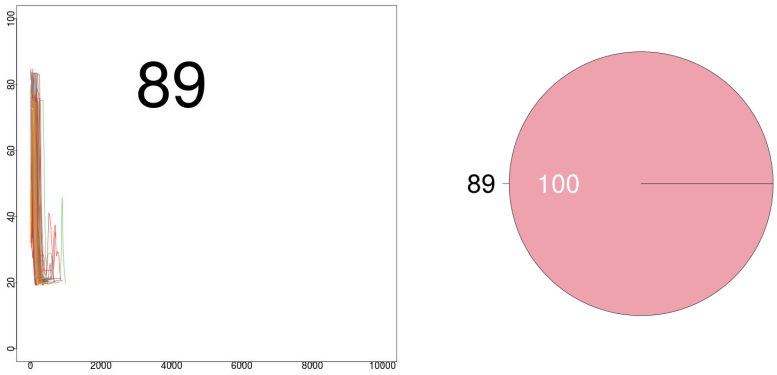


Figure 90: Représentation de la superclasse 3 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La superclasse 3 n'est composée que de phases d'atterrissage.

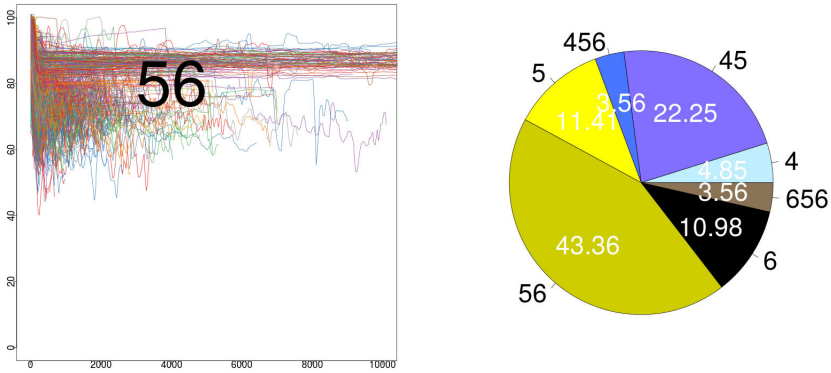


Figure 91: Représentation de la superclasse 4 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La composition de la superclasse 4 est quant à elle plus hétérogène: *climb* (44, 45 et 456), croisière (55, 56) et descente (66 et 656). D'après la Table 8, elles se déroulent pour la plupart en haute altitude.

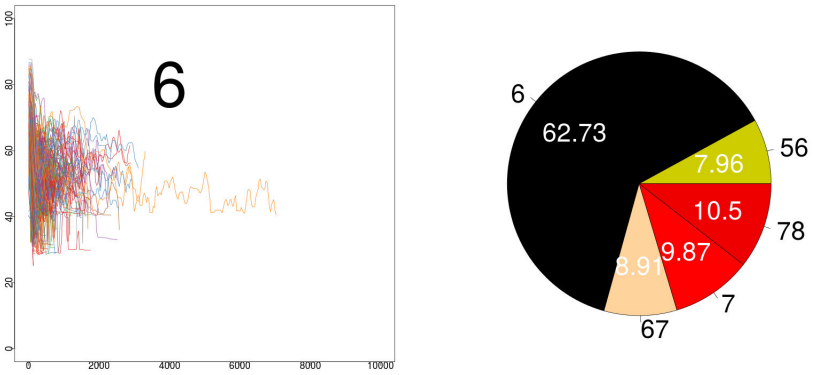


Figure 92: Représentation de la superclasse 5 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La superclasse 5 est composée de phases de descente (56, 66 et 67) et d'approche (77 et 78). Elle est similaire à la superclasse 2 mais on observe que l'amplitude des variations est moins élevée dans la deuxième moitié des phases.

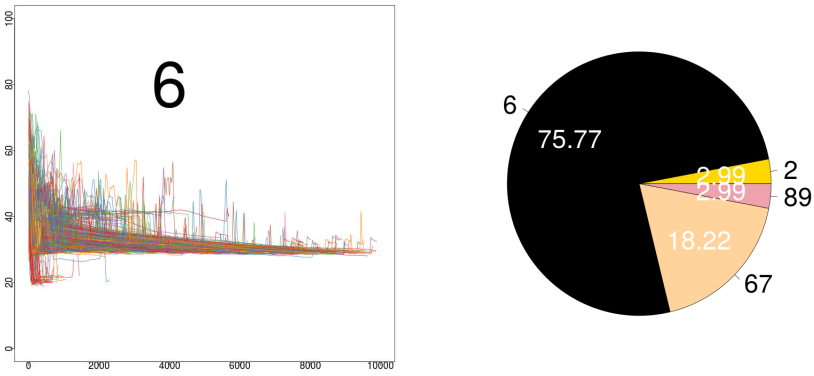


Figure 93: Représentation de la superclasse 6 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

Dans la superclasse 6, on observe des phases de descente (66 et 67) et quelques phases de taxi (22 et 89). L'évolution de ces phases est particulière, elles commencent par un changement abrupt (de 80% à 40%) et elles diminuent progressivement vers 20%.

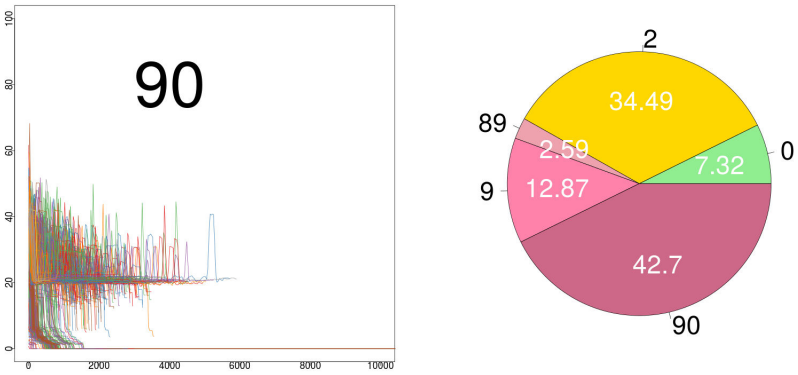


Figure 94: Représentation de la superclasse 7 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

Il s'agit de la seule superclasse composée de taxi (22, 89 et 99) et d'arrêt moteur (90 et 0). Les valeurs de N1 sont comprises entre 0% et 60%.

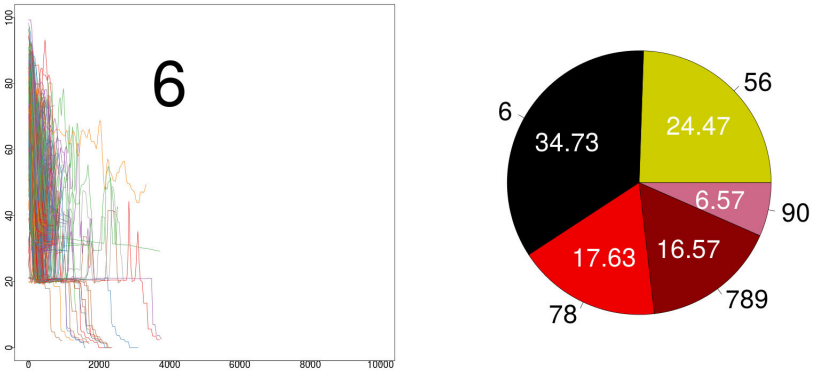


Figure 95: Représentation de la superclasse 8 des phases décroissantes et le diagramme en camembert des modes de vol la composant.

La superclasse 8 est très proche de la superclasse 2 que ce soit au niveau du comportement global des phases ou de sa composition. Toutefois, on remarque que les phases de la superclasse 8 sont plus longues et que l'amplitude des variations vers la fin des phases est plus importante.

9.3 Annexes du chapitre 6

Cartes de Kohonen

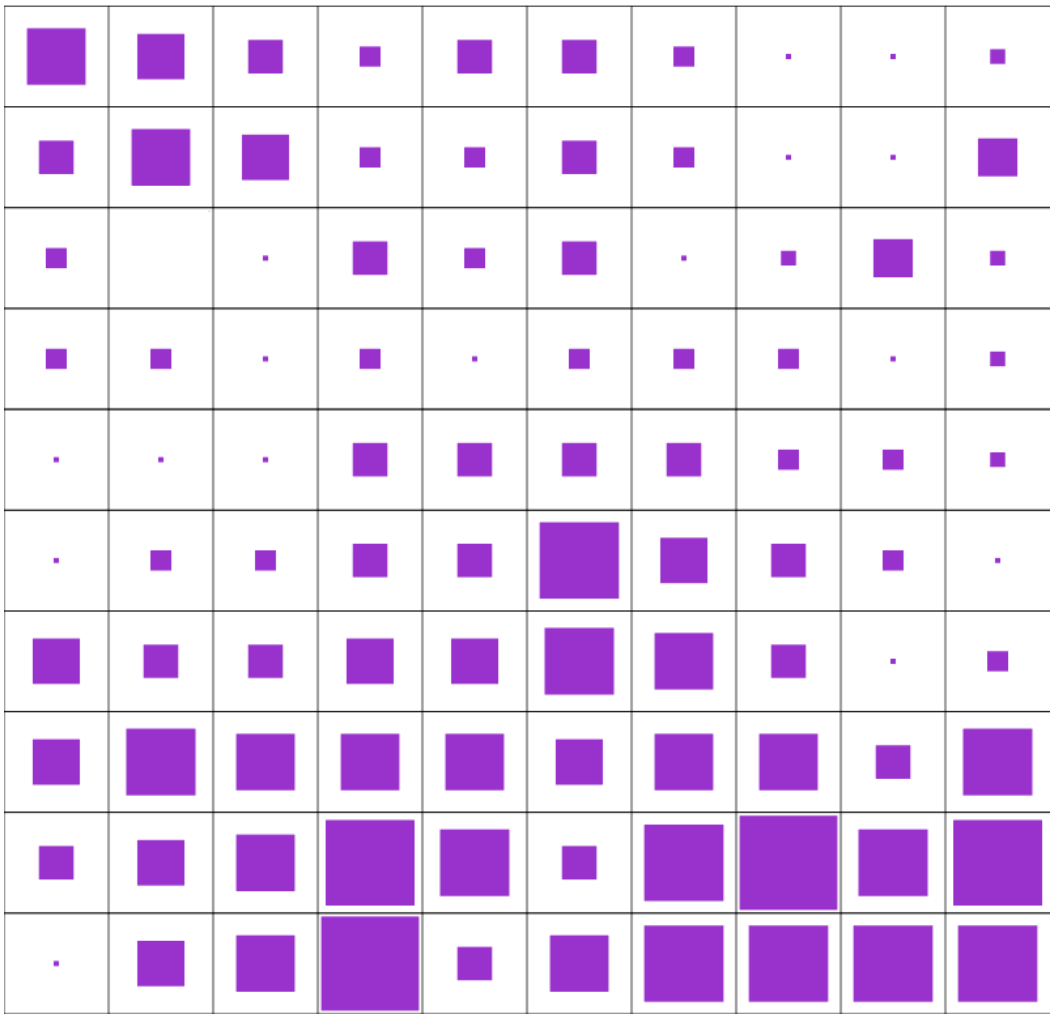


Figure 96: Distribution de la moyenne de la fréquence du label S par séquence labellisée pour chaque classe pour une carte 10x10

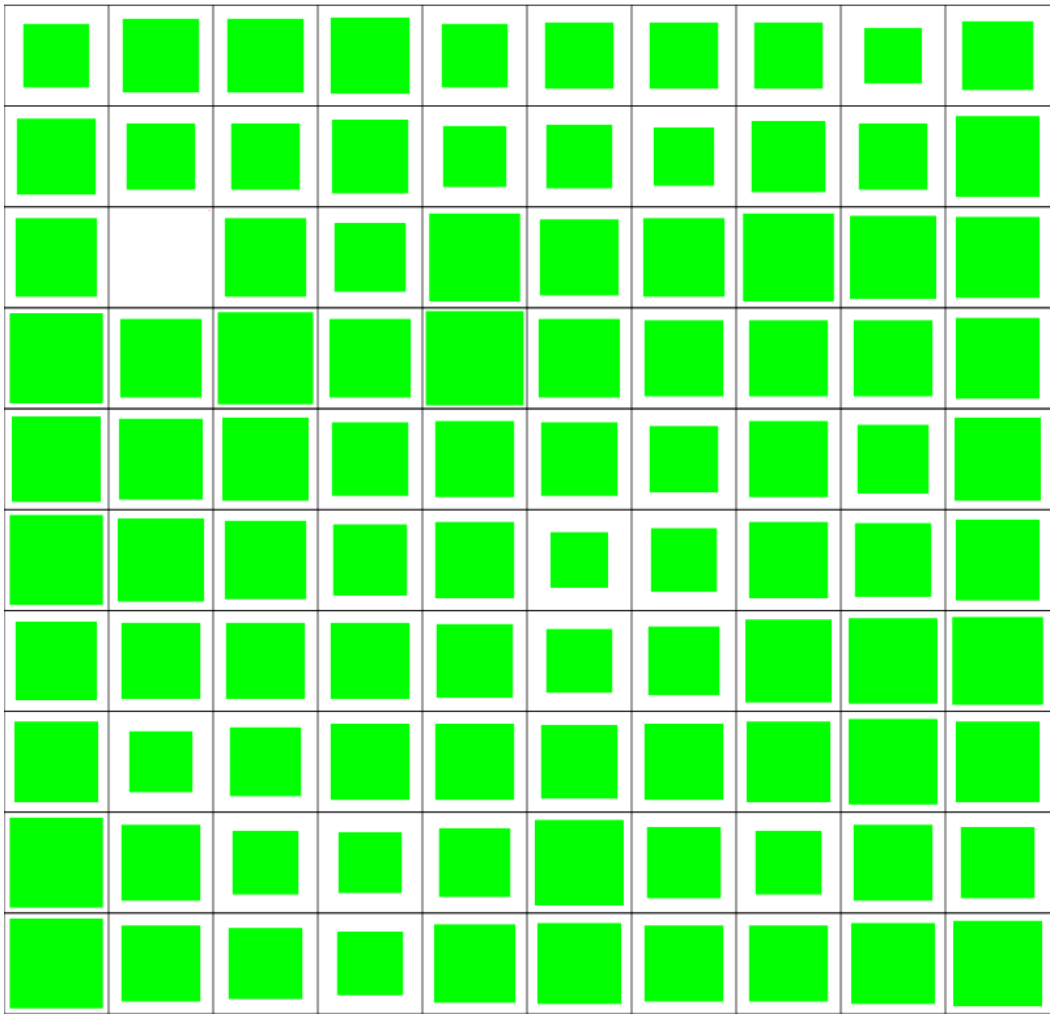


Figure 97: Distribution de la moyenne de la fréquence du label A par séquence labellisée pour chaque classe pour une carte 10x10.

Autres vols volages

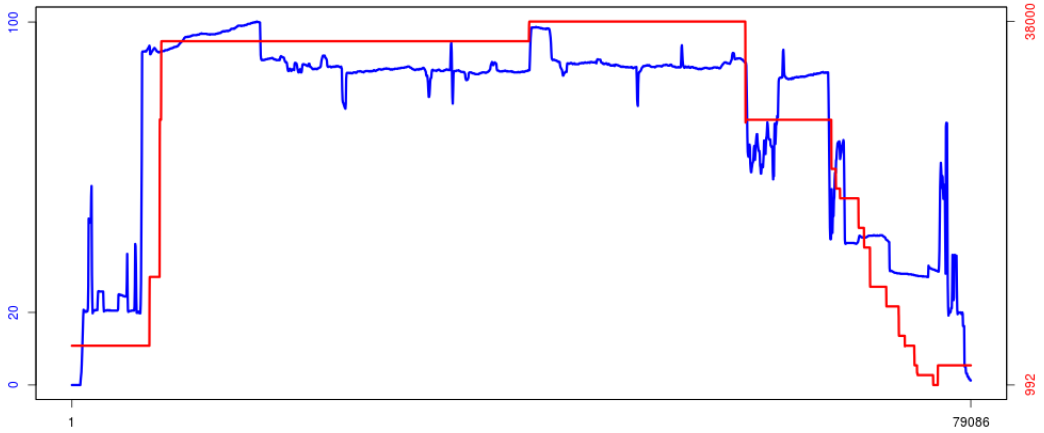


Figure 98: Vol volage

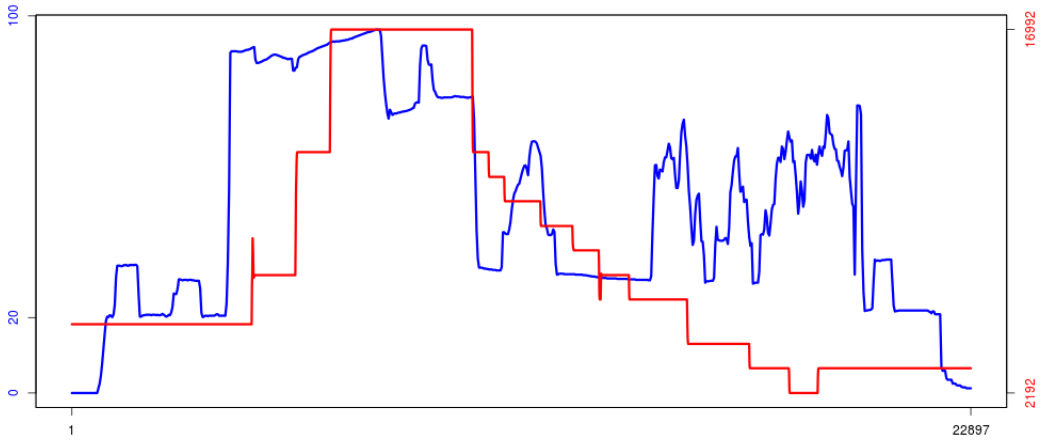


Figure 99: Vol volage

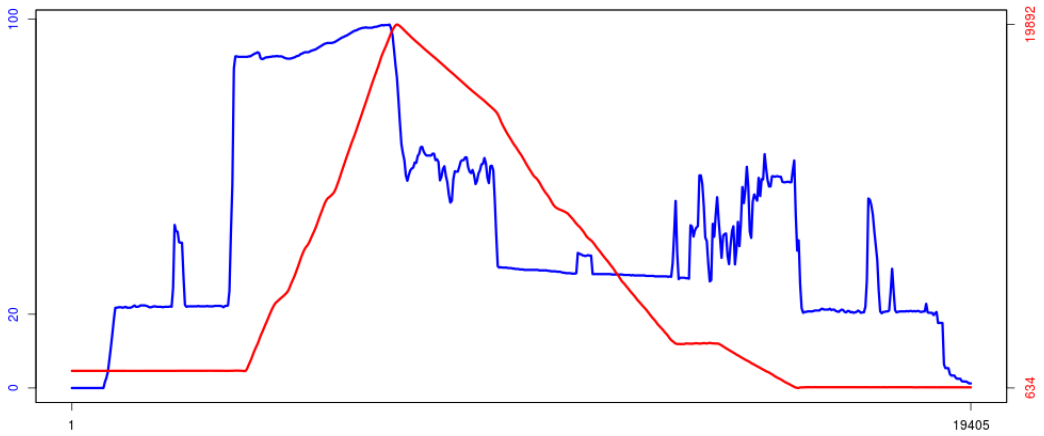


Figure 100: Vol volage

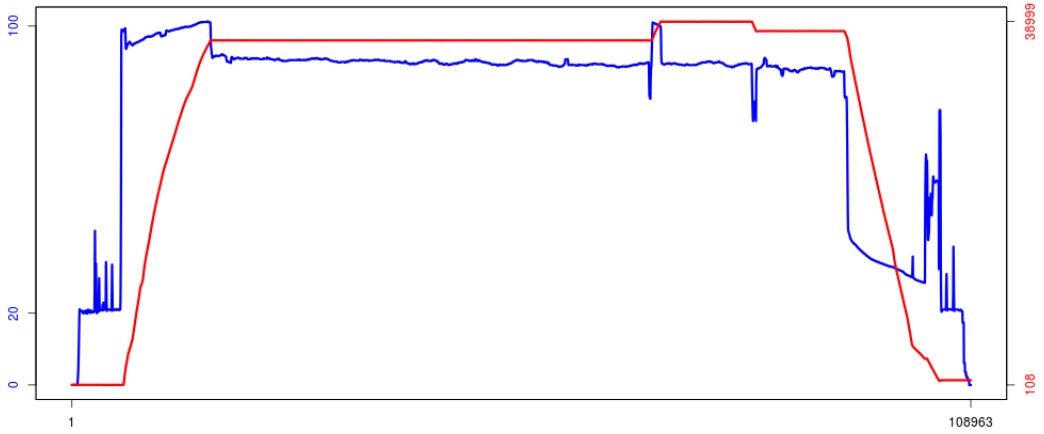


Figure 101: Vol volage

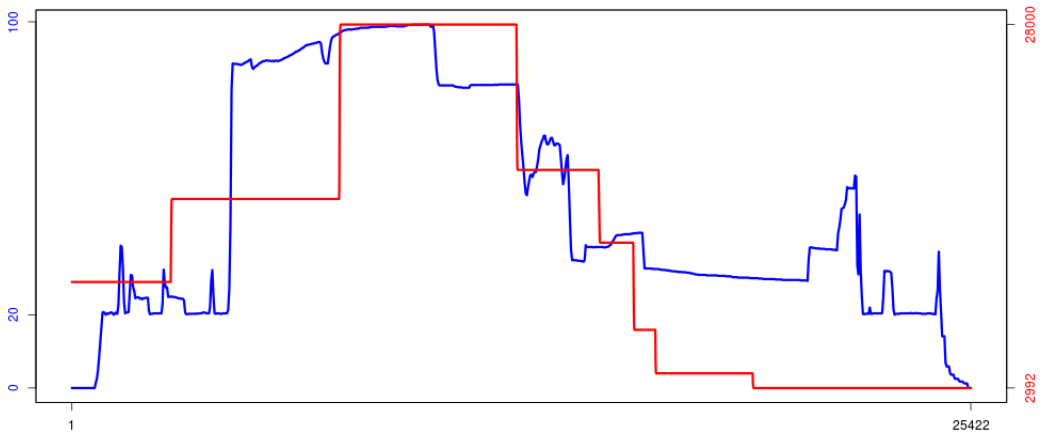


Figure 102: Vol volage

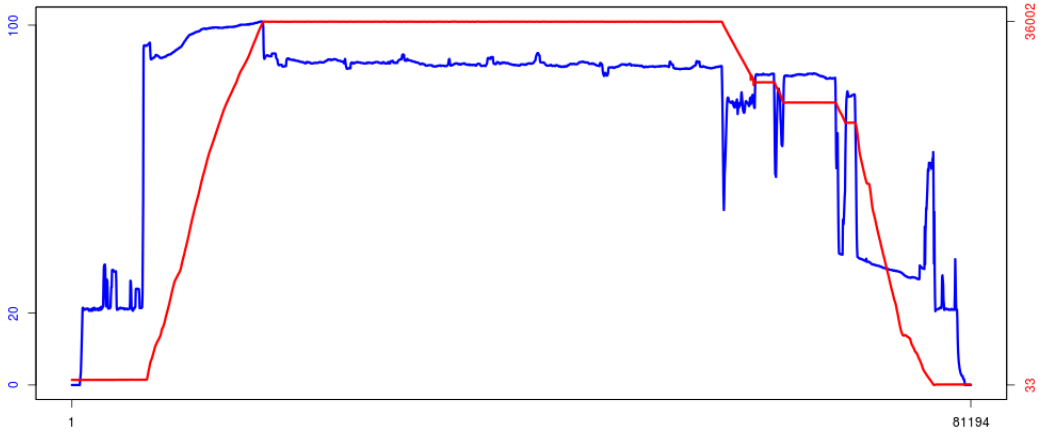


Figure 103: Vol volage

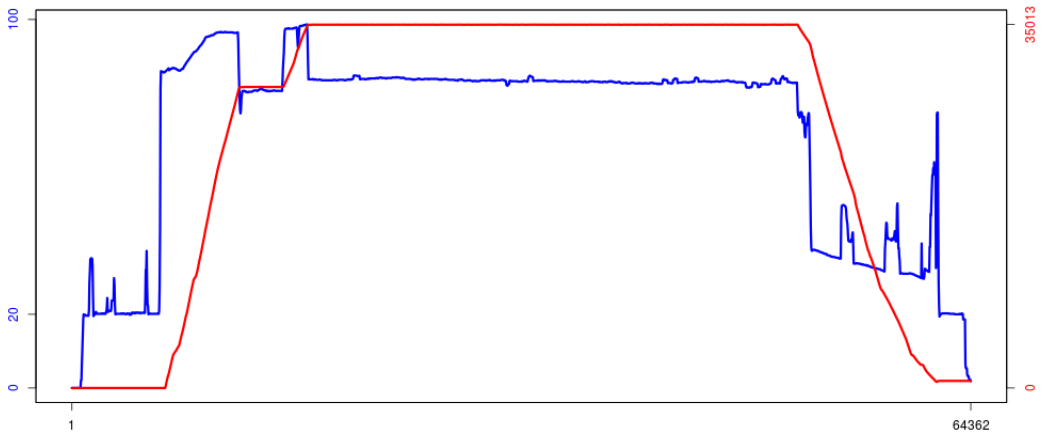


Figure 104: Vol volage

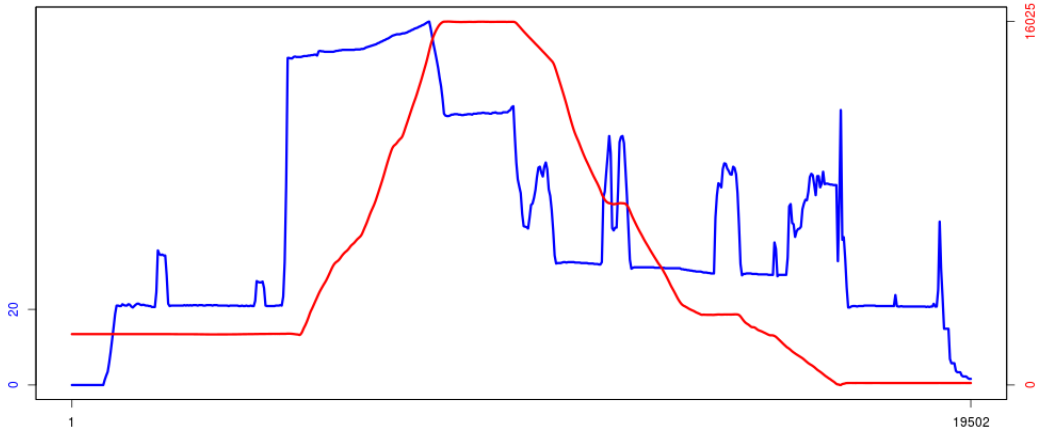


Figure 105: Vol volage

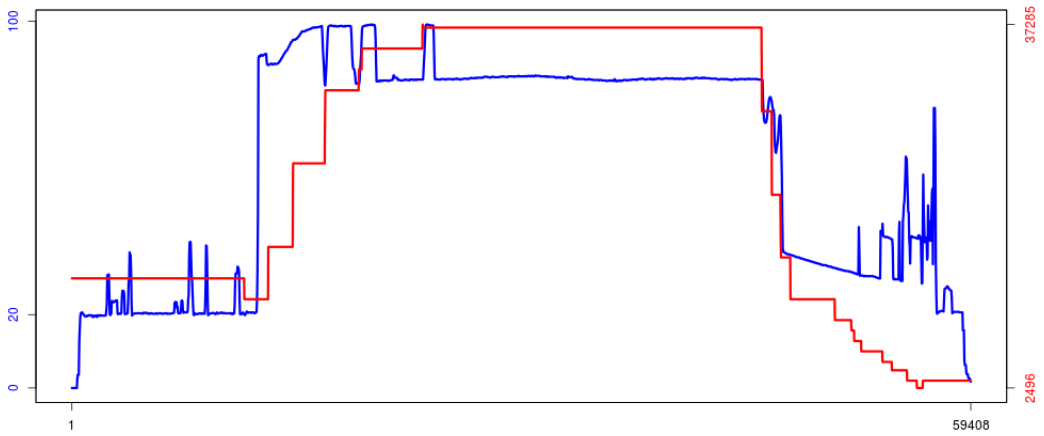


Figure 106: Vol volage

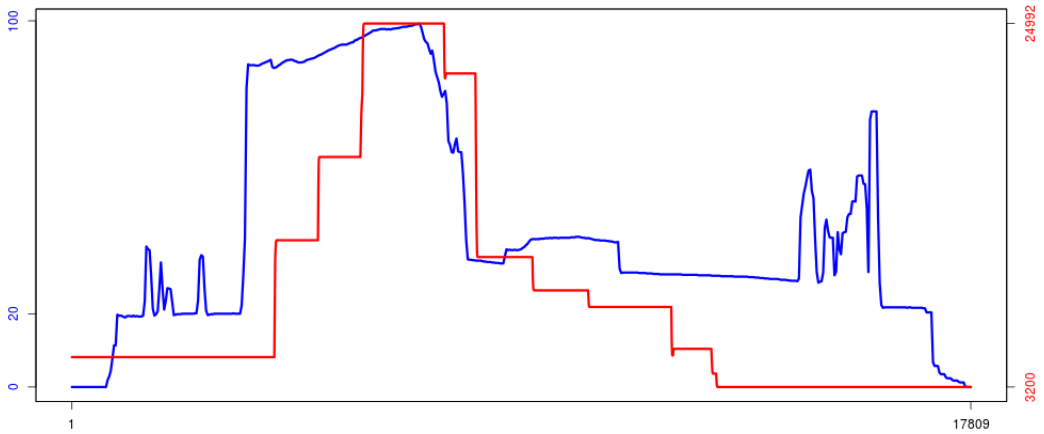


Figure 107: Vol volage