



HAL
open science

Conceptual aircraft design: towards multiobjective, robust and uncertain optimisation

Céline Badufle

► **To cite this version:**

Céline Badufle. Conceptual aircraft design: towards multiobjective, robust and uncertain optimisation. Mathematics [math]. Université Paul Sabatier - Toulouse III, 2007. English. NNT: . tel-00367210

HAL Id: tel-00367210

<https://theses.hal.science/tel-00367210v1>

Submitted on 10 Mar 2009

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

UNIVERSITÉ PAUL SABATIER TOULOUSE III
UFR Mathématiques, Informatique et Gestion

THÈSE

présentée pour obtenir le titre de

Docteur en SCIENCES de l'Université Paul Sabatier

Spécialité : Mathématiques Appliquées

par

Céline BADUFLE

Définition conceptuelle d'avions : vers une optimisation multiobjectif, robuste et incertaine

Conceptual aircraft design: towards multiobjective, robust and uncertain optimisation

Soutenue publiquement le 4 mai 2007 devant le jury composé de :

BÈS Christian	Co-directeur de thèse Professeur de l'université Paul Sabatier Toulouse III
MOHAMMADI Bijan	Président du jury Professeur de l'université des Sciences et Techniques du Languedoc, Montpellier II
DRUOT Thierry	Examineur Ingénieur à Airbus France Toulouse
HIRIART-URRUTY Jean-Baptiste	Co-directeur de thèse Professeur de l'université Paul Sabatier Toulouse III
PÉRIAUX Jacques	Rapporteur Finnish Distinguished Professor, University Jyvaskyla Senior Visiting Professor, CIMNE/UPC Barcelona
SIARRY Patrick	Rapporteur Professeur de l'université de Paris XII Val-de-Marne, Créteil

Mathématiques pour l'Industrie et la Physique
Institut de Mathématiques de Toulouse

Université Paul Sabatier - 118 route de Narbonne - 31062 Toulouse Cedex 4

Remerciements

La première personne que je tiens à remercier tout spécialement est Thierry Druot chez Airbus. Grâce à lui, j'ai beaucoup appris et beaucoup progressé, tant sur le niveau professionnel que personnel. Ses conseils et ses encouragements m'ont été précieux.

Je tiens aussi à exprimer ma sincère gratitude à Christian Bès ainsi qu'à Jean-Baptiste Hiriart-Urruty, qui ont dirigé ces travaux, et sans qui cette thèse n'aurait pas été possible. Par leurs qualités scientifiques et humaines, ils m'ont servi de guides chaque fois que nécessaire, tout au long du déroulement de ma thèse. Ce fut un réel plaisir et un honneur de travailler avec eux.

Je remercie également Yvon Vigneron, Philippe Escarnot, Serge Bonnet et Jacques Genty-de-la-Sagne, ainsi que Gerhard Wolf, Claus Brandl et Yannick Caillabet pour m'avoir intégré au sein de leur équipe aux Avant-Projets d'Airbus.

Je voudrais par ailleurs remercier chaleureusement Christophe Blondel, Philippe Mattei, et Marylène Duffau pour leurs soutiens technique et moral, et enfin Nicolas May, qui a participé à l'introduction de la notion d'incertitude.

J'ai aussi une pensée particulière pour Cécile Péransin, Claire Marcoux, Soledad Martin-Hernandez, Florence Barboule, Michel Maspimby, Matthieu Belleville, Christophe Cros, Christophe Escassut, Olivier Cazals, ainsi que toutes les personnes que j'ai côtoyées chez Airbus durant ces trois années passées aux Avant-Projets et que je ne peux pas citer ici, la liste serait trop longue. Merci pour votre accueil et pour m'avoir fait partager votre savoir.

Je tiens à remercier Jacques Périaux et Patrick Siarry, qui, malgré leur emploi du temps bien rempli, ont généreusement accepté d'examiner le contenu technique de ce mémoire en tant que rapporteurs, et ont ainsi permis son amélioration par leurs commentaires.

Je remercie aussi vivement Bijan Mohammadi pour s'être intéressé à mes travaux et avoir accepté de participer au jury de soutenance.

Merci également à Philippe Homsy et Mohamed Masmoudi, pour leur confiance lors de ce *workshop* totalement imprévu à Bangalore. Ce fut une expérience inoubliable.

J'ajoute un remerciement personnel à mes parents, qui ont toujours cru en moi, ainsi qu'à ma soeur, à mon compagnon et à mes amis pour leur soutien sans condition.

Résumé : La conception d'avions au stade avant-projet consiste à déterminer les principales caractéristiques d'un avion répondant à un cahier des charges donné. Ces études peuvent être résumées par des problèmes d'optimisation globale sous contraintes avec typiquement un millier de paramètres et presque autant de contraintes. Les contraintes expriment la faisabilité physique ainsi que le cahier des charges à respecter, et les objectifs sont des performances de l'avion guidées par des études de marché. De plus, la conception d'avions est un problème d'optimisation multicritère du fait de la présence de fonctions objectifs antagonistes.

L'objectif de cette thèse est d'introduire de nouvelles méthodes mathématiques qui peuvent être utiles dans un outil de dimensionnement avant-projet pour résoudre le problème d'optimisation d'une configuration d'avion. Nous avons contribué à l'amélioration des méthodes d'optimisation qui sont couramment utilisées au département des Avant-Projets d'Airbus. En utilisant les algorithmes génétiques, nous avons rendu le processus d'optimisation monocritère plus robuste. Ensuite, nous avons introduit des méthodes d'optimisation multicritère car nous avons plusieurs critères conflictuels à considérer. Comme les temps de calcul sont devenus importants, nous avons décidé de substituer au modèle d'avion un modèle approché. Nous avons implémenté les fonctions à base radiale pour approcher les contraintes et les fonctions objectifs. Enfin, nous avons propagé les incertitudes du modèle pour estimer la robustesse des résultats de l'optimisation et nous avons proposé un aboutissement possible de l'intégration de ces techniques : donner aux ingénieurs une perception opérationnelle de l'espace de définition.

Mots-clés : Conception d'avions, optimisation multidisciplinaire et multicritère, approximation par un modèle de substitution, propagation d'incertitudes, aide à la décision.

Abstract: Aircraft sizing studies consist in determining the main characteristics of an aircraft starting from a set of requirements. These studies can be summarized as global constrained optimisation problems with typically one thousand parameters and almost as many constraints. The constraints express physical feasibility and the requirements to be satisfied, and the objectives are market driven performances of the aircraft. Moreover, aircraft sizing is typically a multicriteria optimisation problem because of some competing objectives.

The aim of this thesis is to introduce new mathematical methods that can be useful in a future project sizing tool to treat the aircraft configuration optimisation problem. We contributed in improving the optimisation methods that are currently used in the Airbus Future Project Office. By using genetic algorithms, we made the mono-criterion optimisation process more robust. Then, we introduced multicriteria optimisation methods because we had several conflicting criteria to consider. As the calculation times became important, we decided to substitute the aircraft model by a surrogate model. We implemented radial basis functions to approximate the constraint and the objective functions. Finally, we propagated the model uncertainty to assess the robustness of the optimisation results and we proposed a possible outcome of the integration of these different techniques in order to yield the engineers a global and operational perception of the design space.

Keywords: Aircraft sizing, multidisciplinary and multicriteria optimisation, surrogate model approximation, uncertainty propagation, help in decision making.

Contents

Glossary	xvi
Introduction	1
1 Aircraft design in Future Project Office	5
1.1 Future Project Office role	6
1.1.1 Design objectives and requirements	9
1.1.1.1 Focus on Top Level Aircraft Requirements (TLARs)	9
1.1.1.2 Choosing TLARs, functional point of view	10
1.1.1.3 Choosing TLARs, airlines point of view	12
1.1.1.4 Choosing TLARs, manufacturer point of view	12
1.1.2 Multidisciplinary aspects	15
1.1.2.1 Multidisciplinary issues	17
1.1.2.2 Influence on design	18
1.1.3 Design iteration and optimisation	19
1.2 Needs and motivations	20
1.2.1 Why do most of aircraft look like each other?	20
1.2.2 Model description	24
1.2.2.1 Geometry model	24
1.2.2.2 Aerodynamic model	24
1.2.2.3 Weights model	26
1.2.3 Design processes	28
1.2.3.1 Weights calculation for structure design	28
1.2.3.2 Mass-Mission adaptation	30
1.2.3.3 Optimisation under operational constraints	30
1.2.3.4 Optimisation under flight handling quality constraints	32
1.2.3.5 General considerations on design processes	33
1.2.3.6 Conclusion on design processes	34
1.2.4 Known issues with the existing processes	34
1.2.4.1 Convergence of internal resolution processes	34
1.2.4.2 The incomplete information of the mono-criterion optimisation	36
1.2.4.3 Nothing implemented to assess the uncertainty	36
1.2.4.4 Nothing to help in choosing the best model of aircraft	37
1.2.4.5 Black box effect of processes included in specialised modules .	37
1.2.4.6 Non-homogeneity of the aircraft representation	38

1.2.4.7	Conclusion	38
1.2.5	Proposed evolutions and solutions	38
1.2.5.1	Proposed evolutions	38
1.2.5.2	Multicriteria optimisation	39
1.2.5.3	Uncertainty management	39
1.3	Global context	40
1.3.1	Other projects related to the thesis	40
1.3.1.1	Problems to be solved and opportunities	41
1.3.1.2	Potential benefits	42
1.3.1.3	Restructuration of design tools	42
1.3.2	Description of the models	43
1.3.2.1	Current Future Project Office model	43
1.3.2.2	USMAC (Ultra Simplified Model of AirCraft)	45
1.3.2.2.1	Inputs	46
1.3.2.2.2	Outputs	47
1.3.2.2.3	USMAC tuning	47
1.3.2.2.4	USMAC architecture	48
1.3.2.2.5	Basic functions	49
Definition		49
Regulation		49
Models		49
1.3.2.2.6	Domain-level functions	50
1.3.2.2.7	Solving functions for the Mass/Mission loop	52
Range constraint		52
Weight estimate		53
1.3.2.2.8	Study-level functions	53
1.3.2.3	SMAC (Simplified Model of AirCraft)	56
1.4	Global objectives of the thesis	57
1.4.1	Robust global optimisation	58
1.4.2	Multicriteria optimisation	58
1.4.3	Response surfaces	58
1.4.4	Uncertainty	58
1.4.5	Help in decision making	58
2	Robust global optimisation	59
2.1	Multidisciplinary Design Optimisation (MDO)	61
2.1.1	Industrial needs	61
2.1.2	General introduction	61
2.1.2.1	The Multidisciplinary Design Analysis (MDA)	66
2.1.2.2	The Multidisciplinary Feasible (MDF)	68
2.1.2.3	The Individual Discipline Feasible (IDF)	68
2.1.2.4	The Collaborative Optimisation (CO)	69
2.1.2.5	The All-At-Once (AAO)	69
2.1.2.6	The multi-level process	70

2.1.2.7	The multi-physics analysis	70
2.1.3	Current MDO process in Future Project Office	72
2.2	Introduction to Genetic Algorithms (GAs)	75
2.2.1	Principle	75
2.2.1.1	Initialisation	76
2.2.1.2	The fitness function	77
2.2.1.3	The reproduction, or cross-over operator	77
2.2.1.4	The mutation operator	77
2.2.1.5	The selection operator	78
2.2.1.6	Summary	79
2.2.2	Discussion	79
2.3	Ensuring constraint satisfaction	80
2.3.1	Problem description	81
2.3.2	Material	82
2.3.2.1	Architecture of the evaluation function	82
2.3.2.2	Numerical and computational aspects	83
2.3.3	Indirect constraint handling	83
2.3.4	Constraint Satisfaction Problem (CSP) of aircraft sizing studies	84
2.3.4.1	Existence of solutions	85
2.3.4.2	Methods	85
2.3.4.2.1	Problem formulation	85
2.3.4.2.2	Penalty function build-up	86
2.3.4.2.3	Genetic algorithm implementation	87
2.3.4.3	Numerical results	89
2.3.4.3.1	Tuning of the mutation function parameters	90
2.3.4.3.2	Success rate	90
2.3.4.4	Comparison with FSQP	90
2.3.5	Conclusion	92
2.4	Robust mono-criterion optimisation	92
2.4.1	USMAC implementation	93
2.4.1.1	Variable description	93
2.4.1.1.1	Degrees of freedom	93
2.4.1.1.2	Constraints	94
2.4.1.1.3	Criteria	94
2.4.1.2	Adaptation of the previous work to the new problem	96
2.4.2	Improvement of initial population repartition	96
2.4.2.1	Generating well-distributed population	98
2.4.2.2	Latin Hypercube Sampling (LHS)	98
2.4.3	Producing admissible points with USMAC	100
2.4.4	Robust mono-criterion optimisation	100
2.4.5	Going further	103
2.4.5.1	Variables description	103
2.4.5.2	Results	104
2.5	Conclusion	105

3	Multicriteria optimisation	109
3.1	Multicriteria optimisation in engineering	110
3.1.1	Introduction	110
3.1.2	Some definitions	111
3.1.2.1	Multiobjective optimisation problem	111
3.1.2.2	Notion of dominance	113
3.1.3	Resolution methods	113
3.1.3.1	Methods with no articulation of preferences	114
3.1.3.1.1	Min-Max method	114
3.1.3.1.2	Nash arbitration	114
3.1.3.2	Methods with <i>a priori</i> articulation of preferences	115
3.1.3.2.1	Weighted sum method	115
3.1.3.2.2	Lexicographic method	116
3.1.3.2.3	Goal Programming method	116
3.1.3.3	Methods with progressive articulation of preferences	117
3.1.3.3.1	Surrogate Worth Trade off method	117
3.1.3.3.2	Fandel method	118
3.1.3.3.3	Other methods	120
3.1.3.4	Methods for <i>a posteriori</i> articulation of preferences	121
3.1.3.4.1	Physical programming	121
3.1.3.4.2	Normal boundary intersection method	124
3.1.4	Resolution algorithms	127
3.1.4.1	Non-cooperative multiple objective scheme	127
3.1.4.1.1	Merging Nash Equilibrium and GAs	128
3.1.4.1.2	Merging Stackelberg Equilibrium and GAs	128
3.1.4.2	Multi-Objective Simulated Annealing	130
3.1.4.3	Multi-Objective Evolutionary Algorithms (MOEA)	131
3.1.4.3.1	Multiple Objective Genetic Algorithm (MOGA)	131
3.1.4.3.2	Niched Pareto Genetic Algorithm (NPGA)	132
3.1.4.3.3	Strength Pareto Evolutionary Algorithm (SPEA)	134
3.1.4.3.4	Hierarchical Asynchronous Parallel Evolutionary Algorithms (HAPEA)	135
3.1.4.3.5	Tests suites	136
3.1.4.4	Other multiobjective programming methods	137
3.2	Conflicting objectives in our problem	137
3.2.1	Aircraft sizing objectives	137
3.2.2	Conflicting objectives in our models	138
3.2.2.1	When using the USMAC model	138
3.2.2.2	When using the SMAC model	139
3.3	Refinement of the admissible set	139
3.3.1	Introducing the acceptable domain	139
3.3.2	Method to refine the admissible set	140
3.4	Pareto front	141
3.4.1	The selected method	142

3.4.1.1	The ranking function	142
3.4.1.2	The mutation function	143
3.4.1.3	The selection function	143
3.4.2	Test problems	143
3.4.2.1	A classical convex front	144
3.4.2.2	A non-convex front	145
3.4.2.3	A disconnected front	146
3.5	Results	147
3.6	Conclusion	151
4	Surrogate model of the evaluation function	153
4.1	Motivation	154
4.2	State-of-the-art of approximation techniques	156
4.2.1	Polynomial response surfaces	156
4.2.1.1	First order	156
4.2.1.2	Second order	157
4.2.1.3	Least squares method	157
4.2.1.4	Taylor series approximation	158
4.2.2	Kriging interpolations	158
4.2.3	Artificial neural network approaches	159
4.2.3.1	Multi-layer perceptrons	159
4.2.3.2	Support Vector Machines	161
4.2.3.3	Radial Basis Functions	162
4.3	Selected methods	165
4.3.1	Polynomials using Taylor series developments	165
4.3.2	First order polynomials	169
4.3.3	Second order polynomials	172
4.3.4	Radial Basis Functions networks	175
4.4	Comparison of the results	178
5	Mathematical help in decision-making	183
5.1	Introducing robustness	185
5.1.1	Short state-of-the-art	186
5.1.1.1	Worst case analysis	186
5.1.1.2	Extreme condition approach	186
5.1.1.3	Probabilistic formulation	187
5.1.1.4	Moment matching formulation	188
5.1.2	Uncertainty propagation, an emergent technique in FPO tools	188
5.1.3	Feasibility robustness	191
5.1.4	Robustness: a new criterion	206
5.2	Exhibiting model internal relations	207
5.2.1	FPO need	207
5.2.2	Why ellipsoids?	208
5.2.2.1	Ellipsoid with the maximum volume inscribed in a convex set	208
5.2.2.2	Fixed ellipsoid minimising a criterion	212

5.3	Conclusion	214
5.4	Going further	215
Conclusion		217
A	USMAC functions, detailed description	221
A.1	USMAC basic function signatures	221
A.1.1	Definition functions	221
A.1.1.1	Aerodynamics	221
A.1.1.2	Geometry	221
A.1.1.3	Lift over Drag	221
A.1.1.4	Criteria	221
A.1.2	Regulation functions	221
A.1.3	Models	222
A.1.3.1	Environment	222
A.1.3.2	Engine functions	222
A.1.3.3	Geometry functions	222
A.1.3.4	Aerodynamic functions	222
A.1.3.5	Mass functions	223
A.1.3.6	Mission Model functions	223
A.1.3.7	Handling Quality Model functions	224
A.1.3.8	Performance Model functions	224
A.2	USMAC domain-level functions	224
A.2.1	Model functions	224
A.2.2	Operational Performance functions	225
B	Initial optimisation problem description	227
B.1	Degrees of freedom	227
B.2	Constraints	227
B.3	Criterion	227
C	Graphical representations in three dimensions	231
References		234

List of Figures

1.1	Milestones of a project	6
1.2	Diagram of a future project study	8
1.3	A320 family in formation	10
1.4	Family positionning	11
1.5	Market positionning	13
1.6	Life cycle of an aircraft	14
1.7	Caricatural optimal configuration for each discipline [Nicolai, 1975]	16
1.8	Example of unconventional aircraft configurations	18
1.9	Examples of old aircraft configurations	20
1.10	Example of a classical configuration	21
1.11	Comparison between a classical configuration and a flying wing	22
1.12	Cabin definition of the A3XX project	23
1.13	Three view chart of the A3XX project	23
1.14	Three view graph	25
1.15	Aerodynamic model	25
1.16	Fluid dynamic light modeling with linear methods	26
1.17	Different parts of the structure of a wing	26
1.18	Wing shape with its characteristic data	27
1.19	Mass loop	29
1.20	Mass-Mission loop	30
1.21	Optimisation loop	31
1.22	Flight handling quality loop	32
1.23	Design process organisation	33
1.24	Mission profile of a supersonic aircraft	36
1.25	Deterministic and uncertain carpets	37
1.26	Iterative process illustration	45
1.27	Complete flow graph with the 100 basic functions	51
1.28	Flow graph using domain-level functions as boxes	54
1.29	Flow graph at study-level	55
1.30	SMAC 3D model	57
2.1	Illustration of MDO system	64
2.2	Illustration of permutations of independent variables	67
2.3	Illustration of the multiple levels of a project	71
2.4	Example of carpet plot	73

2.5	Illustration of Genetic Algorithm vocabulary	76
2.6	Illustration of reproduction in aircraft sizing studies	77
2.7	Illustration of individual selection	78
2.8	Illustration of mutation in aircraft sizing studies	78
2.9	Illustration of selection in aircraft sizing studies	78
2.10	Flow-chart of a typical genetic algorithm	79
2.11	Admissible set representation	81
2.12	Evaluation function architecture in AVION software	82
2.13	Elementary penalty function representation	87
2.14	Representation of the sharing function Sh	88
2.15	Mutation function representation	89
2.16	Comparison of mutation function parametrisation	90
2.17	Isocurves of the constraints	95
2.18	Constraint position in the search space	96
2.19	Isocurves of the criteria	97
2.20	Example of a Latin Hypercube sampling	99
2.21	Example of a Latin Hypercube sampling in our context	99
2.22	Comparison of initial populations	101
2.23	Comparison of admissible populations	102
2.24	Examples of results of the mono-criterion optimisations	102
2.25	Constraint position in the search space using the SMAC	104
2.26	Examples of results with the SMAC model	105
2.27	Example of results of the mono-criterion optimisation	106
3.1	Illustration of Pareto fronts in the objective space	113
3.2	Illustration of Fandel method	120
3.3	Example of preference regions on the class function 1-S	123
3.4	Illustration of Normal boundary intersection vectors	125
3.5	Illustration of the method Normal boundary intersection	126
3.6	Example of not Pareto optimal points produced by the NBI method	126
3.7	Illustration of the merging of Nash Equilibrium and Genetic Algorithms	129
3.8	Illustration of the merging of Stackelberg Equilibrium and Genetic Algorithms	129
3.9	Example of interpolation function	131
3.10	Representation of the NPGA sharing function	133
3.11	Examples of SPEA fitness assignment	134
3.12	Hierarchical topology of the algorithm HAPEA	136
3.13	Illustration of quality constraints	141
3.14	Two examples of an acceptable domain	142
3.15	Analytical solution of the convex Pareto front test case	144
3.16	Experimental result for the convex Pareto front test case	145
3.17	Analytical solution of the non-convex Pareto front	146
3.18	Experimental result for the non-convex Pareto front	147
3.19	Analytical solution of the disconnected Pareto front problem	148
3.20	Experimental results for a disconnected Pareto front	148
3.21	Experimental result for the Pareto front using the USMAC model	149

3.22	Pareto front in the design space using the SMAC model	150
3.23	Experimental result for the Pareto front in the objective space using the SMAC model	150
4.1	General structure of an artificial neuron	160
4.2	Activation functions of a classical neural network	160
4.3	Three-layer neural network	161
4.4	Projection of points on the boundary	167
4.5	Graphical result of the linearisation of the constraints	168
4.6	Result of the optimisation of the MTOW using an LP optimiser	169
4.7	First order polynomial response surface using the LHS initial population	170
4.8	First order polynomial response surface using the admissible population	171
4.9	First order polynomial response surface using the projected population	171
4.10	Second order polynomial response surface using the LHS initial population	173
4.11	Second order polynomial response surface using the admissible population	173
4.12	Second order polynomial response surface using the projected population	174
4.13	First build of RBF approximation	176
4.14	Second build of RBF approximation	176
4.15	Result of the optimisation using the real functions	179
4.16	Result of the optimisation using the approximate functions	179
4.17	Result of the optimisation using the SMAC model	180
5.1	Influence of the bias of the model structure uncertainty	192
5.2	Influence of the standard deviation with $P_d = 80\%$	193
5.3	Influence of the standard deviation with $P_d = 90\%$	194
5.4	Influence of the model structure uncertainty on the constraints with $P_d = 80\%$	195
5.5	Influence of the model structure uncertainty on the constraints with $P_d = 90\%$	196
5.6	Illustration of the barycentre becoming non-admissible	197
5.7	Initial distribution of points using a Latin Hypercube Sampling	198
5.8	Robust admissible population with $P_d = 80\%$	199
5.9	Projection of points on the boundary of the admissible set	200
5.10	RBF network of the constraint when we consider the bias	201
5.11	RBF network of the constraint when we consider the standard deviation	201
5.12	RBF network of the constraint when we consider the two moments	202
5.13	Mono-criterion optimisations using RBF networks	202
5.14	Influence of the uncertainty on the degrees of freedom	204
5.15	Representations of the points in the design space that are on the Pareto front.	205
5.16	Representations of the points on the Pareto front.	205
5.17	Illustration of the linearisation of the constraints	209
5.18	Ellipsoid of maximum volume included in the set of our interest	211
5.19	Illustration of the principle of the method to find the centre of the ellipsoid	213
5.20	Ellipsoids with fixed main axes which centres minimise the criteria	214
B.1	Illustration of current degrees of freedom	227
B.2	Illustration of current constraints	228

B.3	Illustration of current value of the project criterion	228
C.1	Illustration of the initial population in 3 dimensions	231
C.2	Illustration of admissible population	232
C.3	Illustration of frontier of the admissible domain	232
C.4	Illustration of the result of the mono-criterion optimisations	233

List of Tables

1.1	USMAC input variables	46
1.2	USMAC output variables	47
1.3	Basic identification data	47
1.4	USMAC package	56
2.1	Test case numerical results	92
2.2	Definition of the search space of our problem	93
2.3	Fixed values for secondary-level degrees of freedom	94
2.4	Definition of the constraint bounds of our problem	94
2.5	Test case numerical results	101
2.6	Definition of the search space of our problem using SMAC	103
2.7	Definition of the admissible domain of our problem using SMAC	104
3.1	Preference function classification for Physical programming	122
4.1	Classical RBF functions	164
4.2	Comparison of the evaluation of the constraints in the barycentre	172
4.3	Relative errors in the approximation of the constraints	172
4.4	Comparison of the evaluation of the constraints in the barycentre	174
4.5	Relative errors in the approximation of the constraints	174
4.6	Comparison of the evaluation of the constraints in the barycentre	175
4.7	Relative errors in the approximation of the constraints	177
4.8	Number of centres needed to interpolate each constraint	177
4.9	Comparison of the results of the optimisations	178
4.10	Comparison of the results of the optimisations using the SMAC model	180
5.1	Effect of the bias of the model structure uncertainty on the constraints	192
5.2	Effect of the standard deviation on the constraints with $P_d = 80\%$	194
5.3	Effect of the standard deviation on the constraints with $P_d = 90\%$	194
5.4	Effect of the model structure uncertainty on the constraints with $P_d = 80\%$	196
5.5	Effect of the model structure uncertainty on the constraints with $P_d = 90\%$	196
5.6	Values of the degrees of freedom	200
5.7	Values of the constraints	203
5.8	Comparison of the optimisation results	203
B.1	Degrees of freedom	228

B.2 Constraints	229
B.3 Criterion	229

Glossary

AAO	All-At-Once, 66, 69
alt	Altitude, 46
ANN	Artificial Neural Networks, 157, 159
AR	Aspect Ratio, 94, 170, 172, 173
AVION	The current FPO tool, 43, 44, 53, 57, 82, 92, 100, 103, 137
Awing	Wing area, 27, 46, 93–97, 147, 176, 178, 200–202, 204
BPR	Engine Bypass ratio, a measurement that compares the amount of air blown past the engine to that moving through the core, 46, 47, 93, 94
CFD	Computational Fluid Dynamics, 18, 70
CO	Collaborative Optimisation, 65, 69, 72, 153
COC	Cash Operating Cost, 104, 136, 137
CSM	Computational Structural Mechanics, 70
CSP	Constraint Satisfaction Problem, 60, 80, 84, 86, 106
DE	Differential Evolution, 135
disa	Delta ISA, the temperature shift in reference to the International Standard Atmosphere representation, 46, 47
DOC	Direct Operating Cost, 12, 17, 34, 36, 58, 104, 109, 136
DoE	Design of Experiments, 98, 175
EA	Evolutionary Algorithm, 75, 129
FAR	Federal Air Regulations, 13
FNslst	Sea level static thrust, a reference thrust used for take off, 46, 93–97, 147, 176, 200, 201, 204
FoPoNe6	Fuel Efficiency, 96, 136
ForTab	Fortran-Tableur, AVION calculation kernel, 44
FOSM	First Order Second Moment, 187

FPO	Future Project Office, 7, 23, 26, 30, 32, 37, 39–41, 43, 60, 70, 72, 74, 82, 105, 108, 109, 135, 136, 149, 182, 183, 186, 205
FSQP	Feasible Sequential Quadratic Programming, 57, 60, 80, 84, 85, 90–92, 100, 102, 103, 105, 106, 108, 137, 138, 141, 196, 201, 208, 211
Fuel	Nominal Fuel, 94, 136
GA	Genetic Algorithms, 75, 77, 79, 80, 93, 98, 108, 126, 129, 137, 164, 196, 197
HAPEA	Hierarchical Asynchronous Parallel Evolutionary Algorithm, 133, 205
IDF	Individual Discipline Feasible, 65, 68
JAR	Joint Airworthiness Requirements, 13
Kff	Fuselage fuel ratio, 47, 94, 170, 172, 173
Kfn	Scaling factor on engine thrust, 47, 94, 170, 172, 173, 190, 192, 194, 195, 201
krub	a rubbering coefficient of the engine size, 103, 147, 178
LHS	Latin Hypercube Sampling, 98, 167, 168, 196
Lref	Reference length, 27
MDA	Multidisciplinary Design Analysis, 65, 66
MDF	Multidisciplinary Feasible, 44, 65, 68
MDO	Multidisciplinary Design Optimisation, 61, 62, 112
MLW	Maximum Landing Weight, 53
MOEA	Multi-Objective Evolutionary Algorithms, 129, 134
MOGA	Multiple Objective Genetic Algorithm, 129, 130
MOPSO	Multiple Objective Particle Swarm Optimization, 135
MOSA	Multiple Objective Simulated Annealing, 128
MTOW	Maximum Take-Off Weight, 15, 27, 28, 30, 31, 33, 37, 53, 72, 94, 104, 109, 136, 137, 164, 166, 196, 200, 201, 204
MWE	Manufacturer Weight Empty, 28, 94, 104, 136, 137
MWE_oMTOW	Weight Efficiency, 94, 136
Mwing	Wing weight, 27
MZFW	Maximum Zero Fuel Weight, 53
Naisle	Number of aisles, 46

NBI	Normal Boundary Intersection, 122
ne	Number of engines, 46
NLP	Nonlinear programming, 63
NM	Nautical Mile, 1NM=1852m, 17
Npax	Number of passengers, 46, 49
NpaxFront	Front passenger number, 46, 49
NPGA	Niched Pareto Genetic Algorithm, 130, 131
NRC	Non Recurring Costs, 12
NSGA	Non dominated Sorting Genetic Algorithm, 87, 131, 135, 164
ODISA	Object Driven Integrated Sizing and Analysis, 41
OWE	Operational Weight Empty, 35, 47, 53, 104, 137
PASA	Pareto Archived Simulated Annealing, 128
phi	Wing sweep angle, 27, 46
PL	Payload, 47, 52, 53
RA	Range, 47, 52
RBF	Radial Basis Functions, 152, 160, 173, 197
RC	Recurring Costs, 12
SMAC	Simplified Model of AirCraft, 43, 55, 57, 103, 178
SOM	Self-organising maps, 213
span	Wing span, 18, 27, 31, 46
SPEA	Strength Pareto Evolutionary Algorithm, 132, 133
SV	Support Vector, 159
SVM	Support Vector Machine, 159, 160
TLARs	Top Level Aircraft Requirements, 9, 19, 182, 205, 225
TOFL	Take-off field length, 31, 37, 44, 47, 94, 170, 172, 173, 190, 192, 194, 201
tuc	Thickness upon chord ratio, 27, 46
USMAC	Ultra Simplified Model of AirCraft, 43, 45–47, 49, 57, 93, 100, 102–105, 136, 137, 147, 149, 178, 182, 188
Vapp	Approach speed, 31, 37, 47, 50, 94, 137, 170, 172, 173, 190–192, 194–196, 200–202, 204
VEGA	Vector Evaluated Genetic Algorithm, 135
VIVACE	Value Improvement through Virtual Aeronautical Collaborative Enterprising, 40

V_z	Climb speed, 47, 94, 170, 172, 173, 190, 192–195, 200–202
ZFW	Zero Fuel Weight, 52

Introduction

This Ph.D. work (called “thèse CIFRE¹”) has been performed in collaboration with Airbus France, at the Future Project Office. The role of the Future Project Office is to define the conceptual and the preliminary design of military or civil transport aircraft. The main objective of this phase is to define and study an aircraft configuration, which will be frozen prior to pursuing more detailed studies. To summarize, the role of the Future Project Office is to perform aircraft sizing.

Generally, this first phase in designing an aircraft starts by a series of iterations between target market predictions, strategic choices, and proposed technical solutions. These iterations allow to determine the requirements and the objectives that the future aircraft will have to fulfill. The role of the Future Project Office consists in providing the general characteristics of an aircraft configuration that meets these requirements and objectives at best. After this first phase, the configuration is given to the specialised departments that work on the detailed design of the aircraft.

The difficulties that occur when designing an aircraft are that aircraft sizing is intrinsically:

- an inverse problem. The performances, given in the requirements and the objectives, can be calculated once the aircraft geometry and characteristics are known.
- a multidisciplinary problem. There are several disciplines involved in the definition of the aircraft, like the aerodynamics, the structure, the weights, the noise, the flight handling qualities, ...
- a multiobjective optimisation problem. Indeed, the ideal best aircraft configuration is the one having the minimum weight, while ensuring the best flight performances and being the less expensive to produce.

Moreover, future project studies aim at defining not only one new configuration of aircraft, but several configurations of aircraft that are sharing some components, like the wing for instance. This means that aircraft sizing studies lead to the definition of an aircraft family, the aim being to reduce development costs, or to increase the covering of the market shared by wider products.

The Future Project tool used for sizing is able to handle some of these difficulties. It is an open environment that allows to plug models that represent the different disciplines needed

¹CIFRE means in french Convention Industrielle de Formation par la REcherche

to calculate the performances of the aircraft, and it is also able to manage any direct or reverse calculation of any input or output scalar variable. Thus, this tool is able to deal with the multidisciplinary aspect and the inverse problem of aircraft sizing studies.

The objective of this study is to introduce new mathematical methods that can be useful in a future project design tool. These methods mainly aim at:

- making the current optimisation process more robust,
- solving the multiobjective part of the aircraft sizing problem,
- reducing the calculation times by replacing the functions by a surrogate model,
- introducing the notion of uncertainty in the design process,
- and introducing mathematical tools that can be helpful to make decisions.

The first chapter of this manuscript consists in describing in more details what is the role of the Future Project Office, the process of designing a new configuration of aircraft and the models that are currently used in that purpose.

The second chapter describes the method we developed to improve the global optimisation processes in the current FPO tool. Indeed, it offers the possibility to use commercial optimisation methods, but most of the time, the optimisation problem is too complicated to be solved in one optimisation calculation. The problem has to be simplified before being solved by the commercial optimisers. For instance, only one objective is optimised, and this objective is a weighted sum of all the objectives that are needed to be optimised concurrently. Moreover, the constraints reduce the design space in such a way that the optimisation frequently fails because of non-admissible points that the process is not able to calculate. Thus, the optimisation as it is currently performed is not robust.

We decided to first work on the mono-objective problem, the one currently treated by the Future Project Office engineers, by introducing a more robust process. The aim is to implement evolutionary algorithms to enhance the current global multidisciplinary optimisations. Indeed, we identify that the failures of the optimisation processes are due to the difficulty to find feasible points. Thus, before performing any optimisation, we want to identify a large amount of points satisfying the constraints, and then to reduce the design space to the admissible space, which will improve the convergence robustness.

After improving the robustness of the mono-objective problem, we decided to treat the problem as a multiobjective problem. Most of the time, the criterion to optimise in future project studies is a function that is well-representative of the aircraft quality and which integrates the impact of the operating costs. But the drawback in using this weighted sum as the criterion of the optimisation is that the requirements can evolve during the aircraft development. Thus, we also want to perform and validate multicriteria optimisation, still using evolutionary algorithms. The aim is to be able to produce a set of compromise solutions that are equivalent in solving the aircraft sizing process, to give the possibility to

the decision-makers to make their choice on additional criteria that are more qualitative. This is the topic of the third chapter.

When dealing with the multiobjective problem, we notice that the calculation time can easily become too important, especially for a typical future project study, which has to be reactive. As the whole problem is heavy and complex, we decided to introduce surrogate models to replace the evaluation function in the optimisation process. These techniques enable us to have an approximation of solutions in a shorter computation time. Once the design point is close enough to the solutions, it is possible to have a better assessment by using back the initial evaluation function. In chapter four, we present a short state-of-the-art of approximation techniques before explaining the way we use some of them.

Concurrently to the introduction of multiobjective optimisation techniques, the other aim of this study is to introduce the notion of uncertainty in the design process. We want to assess the technical risks related to one particular configuration, and to improve the way to take margins to ensure that the operational constraints are satisfied. Currently, the risk is assessed thanks to the sensitivity analysis manually performed through Jacobian.

After defining precisely what we intend with these notions of risk, robustness and uncertainty, because they do not exist in the vocabulary of designers of the Future Project Office yet, we introduce in the chapter five some methods to propagate the model structure uncertainty to the output of the system. Then, we tested one of these methods and compared the results obtained in a deterministic study with the results when we take the uncertainty into account.

In this last chapter, we also introduced a mean to help the users in understanding and dealing with the results that the methods we developed can produce. The idea was to inscribe an ellipsoid inside the feasible set to assess the volume and the shape of this set, and to determine the centre of a given ellipsoid that minimises one criterion. These ellipsoids give some information about the correlations between the design variables, or they model the freedom or the uncertainty that we allow around design points.

Chapter 1

Aircraft design in Future Project Office

Contents

1.1	Future Project Office role	6
1.1.1	Design objectives and requirements	9
1.1.2	Multidisciplinary aspects	15
1.1.3	Design iteration and optimisation	19
1.2	Needs and motivations	20
1.2.1	Why do most of aircraft look like each other?	20
1.2.2	Model description	24
1.2.3	Design processes	28
1.2.4	Known issues with the existing processes	34
1.2.5	Proposed evolutions and solutions	38
1.3	Global context	40
1.3.1	Other projects related to the thesis	40
1.3.2	Description of the models	43
1.4	Global objectives of the thesis	57
1.4.1	Robust global optimisation	58
1.4.2	Multicriteria optimisation	58
1.4.3	Response surfaces	58
1.4.4	Uncertainty	58
1.4.5	Help in decision making	58

1.1 Future Project Office role

Defining a commercial transport aircraft is a complex process (described in [Pacull, 1990], and in [Torenbeek, 1982]) because of industrial aspects, of the increasing number of involved partners in a concurrent engineering process, or because of safety constraints.

The development of new airliners has always been stimulated mainly by the growth of the traffic volume and the improvement of technical and operational standards [Torenbeek, 1982]. Growth in air traffic stems from:

- reduction of fares,
- improved quality of the aircraft (speed, comfort),
- increased business activity and growth of private incomes,
- aircraft capacity growth,
- increasing number of routes,
- increasing frequency on existing routes,
- greater utilization of aircraft and ground facilities.

Aircraft design process can be split into two main phases, the “future project” phase, going from milestone M1 to M5 in the figure 1.1, and the “project” phase, from M5 to M13.

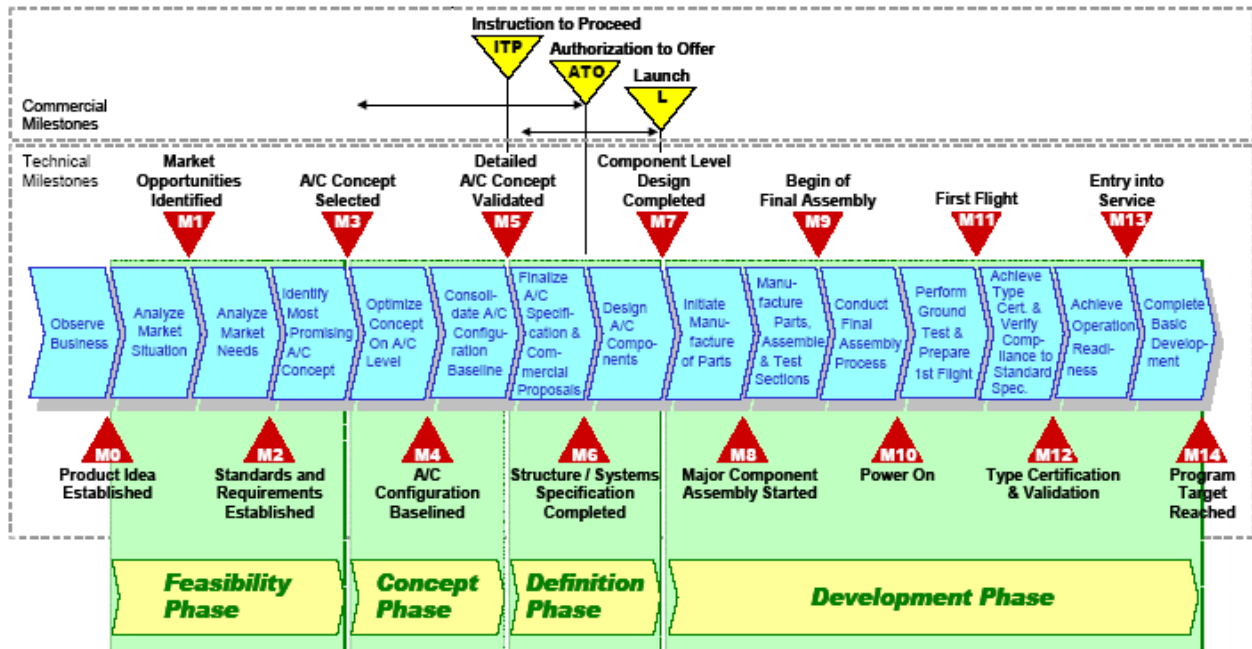


Figure 1.1: Milestones of a project

Definition 1.1. [Schumann-Hindenberg, 2001]

A **milestone** is a significant event in a project, usually the completion of a major deliverable (in any case an end event of an activity). It has the character of a decision point until the end of the definition phase (or Launch if before) and a target date subsequently.

The main objective of the first phase is to define and study an aircraft configuration, which will be, more or less, frozen prior to pursuing more detailed studies.

Definition 1.2. [Torenbeek, 1982]

The expression **configuration** refers to the general layout, the external shape, dimensions and other relevant characteristics.

Thus, the “future project” phase consists in a series of iterations between target market predictions, strategic choices, and proposed technical solutions.

When a project gets mature enough from a commercial and technical point of view, but also from an industrial and financial point of view, the “project” phase, or development phase, starts. More detailed studies are performed to obtain the complete definition of the product, and also the industrial means necessary for its manufacturing are determined.

This second phase is ended by flight tests, certification and finally by the delivery of the aircraft to the airlines.

Remark 1.1. Any modifications in the “project” phase have a high impact on costs. This emphasizes the role of the earlier phases to collect airlines needs and to propose an aircraft configuration answering these requirements.

In the thesis, we will focus on **conceptual and preliminary technical studies which contribute in defining a future project of aircraft.**

The conceptual design consists in investigating the viability of the project and obtaining a first impression of its most important characteristics [Torenbeek, 1982]. The main products of conceptual design are geometric description, configuration layout and drawings.

Then, the design obtained in conceptual studies that has scored the highest rating will be elaborated in greater detail in the preliminary design phase [Torenbeek, 1982]. This phase will detail research on the global aircraft, and will end when the configuration is sufficiently accurate for pursuing developments.

Future Project Office, further denoted as FPO, plays an important role **in the first phase** of an aircraft definition process, mainly working on the conceptual and the preliminary design of the configuration. It provides the general characteristics of the aircraft to meet requirements at best, it gives an initial configuration as a common starting point to the specialised departments for the detailed studies, and then, during the development phase, it links the different departments to ease collaborative work.

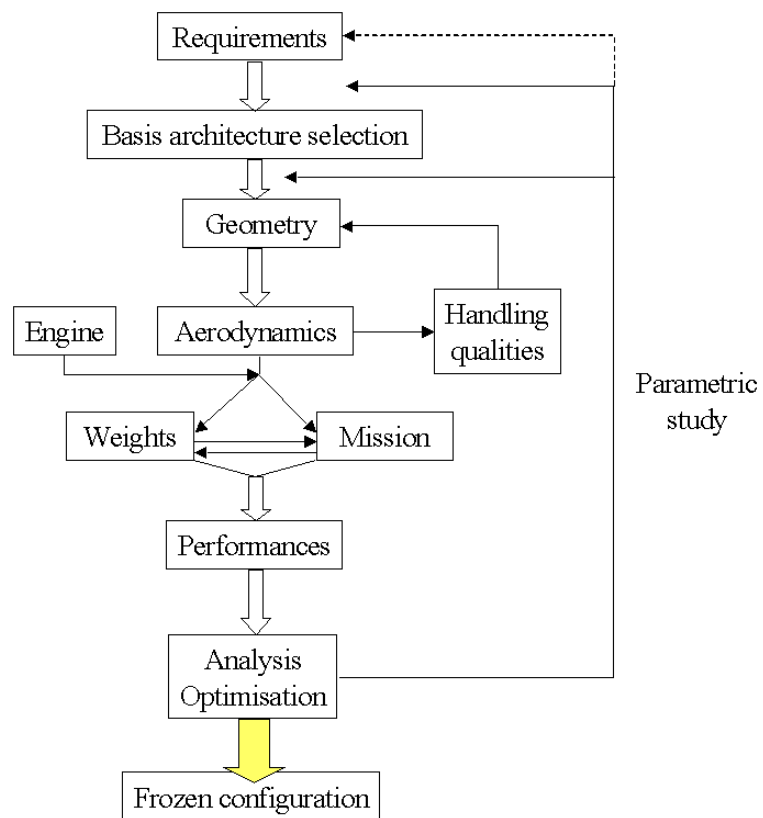


Figure 1.2: Diagram of a future project study

The general process leading to the definition of future project of aircraft is described on figure 1.2.

The objective is to define the main characteristics of an aircraft to meet the already established set of requirements (range, number of passengers, etc.). Obviously, several configurations can satisfy a given set of requirements. Thus, it is necessary to define a criterion to determine which solution is the best-adapted to the problem.

Engineers have to find the *best* configuration in some sense, but what does *best* configuration mean? This question is very important because the final result strongly depends on this notion of optimality. So one of the first task when designing a new aircraft is the establishment of design requirements and objectives.

1.1.1 Design objectives and requirements

The objectives are defined by the Marketing and the Product Strategy departments, and by the aircraft program manager. They are for instance weights, fuel burn, operating costs, etc.

The design requirements are collected in the Top Level Aircraft Requirements, further called TLARs. Specific design requirements are based on airline requirements, certification requirements, and aircraft manufacturer policy (including competitors).

Detailed sets of requirements and objectives include specification of aircraft performance, safety, reliability and maintainability, systems properties and performance, etc. TLARs and objectives are used to formally document the project goals, to help designers in ensuring that the final design meets the requirements, and to aid in future product development.

1.1.1.1 Focus on Top Level Aircraft Requirements (TLARs)

Choosing the TLARs is a very important step. All the aircraft characteristics result from these requirements, thus it has consequences on its economics, on the acceptance of airlines, and on its possibilities of future evolution. An example of evolution is the definition of new members in a family of aircraft, based on one aircraft configuration. For instance, the A320 family:

- it started with the A320, the first flight was in 1987,
- then, the A321, launched in 1993,
- the A319, launched in 1995,
- the Airbus Corporate Jetliner, ACJ, launched in 1997,
- finally, the A318, launched in 1999.

All these aircraft were defined based on the A320 configuration (see figure 1.3).

Today, development costs are so expensive that, usually, a new study leads to the definition of not only one new aircraft, but to a new aircraft family. Indeed, deriving an aircraft from a basic one, depending on the market evolution, is much less expensive than defining each



Figure 1.3: A320 family in formation

time an entirely new configuration, due to commonality of some components, like the wing aerodynamic shape, the forward and rear part of fuselage, the systems, etc. The price to pay is that the derived aircraft is theoretically sub-optimal from a pure performance point of view. But a loss on one performance criterion will maybe provide a gain on another criterion, development cost for instance, with common components of several configurations.

The figure 1.4 is an illustration of member positioning of a family to cover a large range of needs of the airliners.

All these considerations should be taken into account while elaborating requirements (see in [Pacull, 1990], chapter XI).

1.1.1.2 Choosing TLARs, functional point of view

The functional objective of a commercial aircraft is to transport some payload (cargo or passengers) from one point to another on the earth. The difference with other transportation means is that it is through the air. The main reason is the higher travelling speed than the one permitted on the ground, because of friction removal, and also because of the decrease of the air density with altitude. Travelling in the air has some consequences, like noise, costs, safety complexity, or complicated physical phenomena to take into account when designing a product.

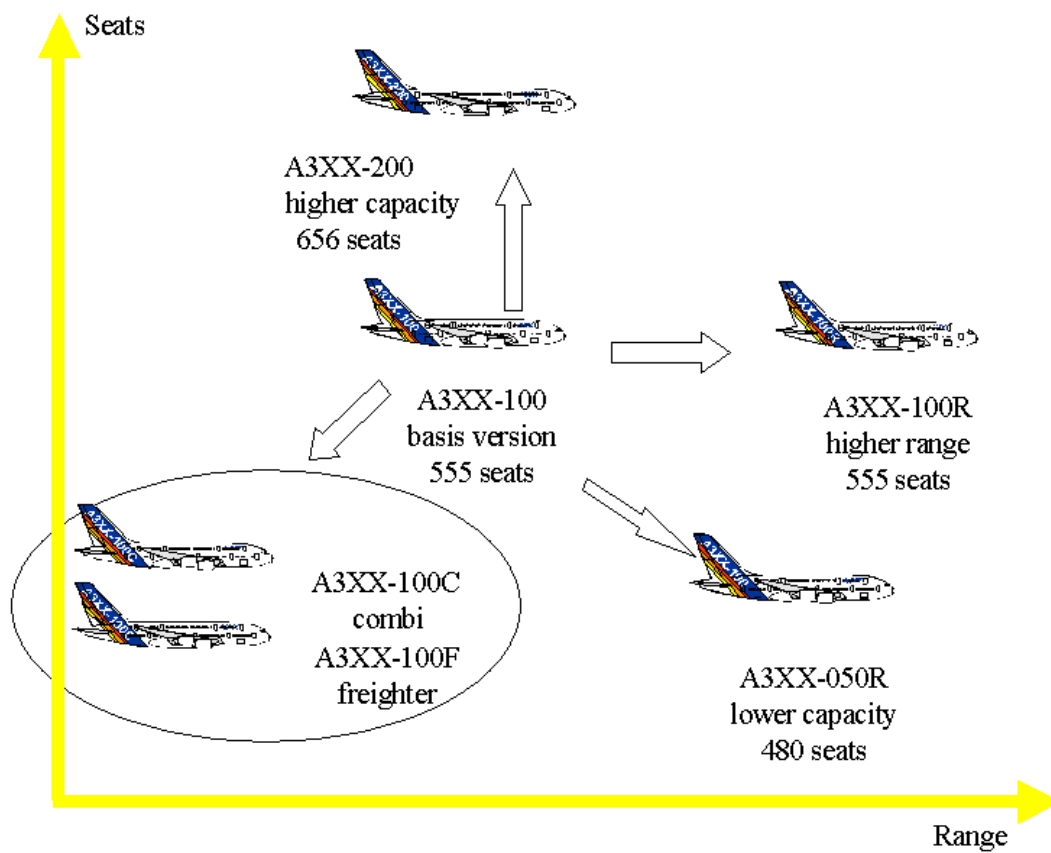


Figure 1.4: Family positioning

1.1.1.3 Choosing TLARs, airlines point of view

Each airline has its own needs, but the common ones are related to the transportation of passengers. They are for instance:

- Number of passengers,
- Comfort standard,
- Cargo capacity,
- Range,
- Cruise speed,
- Specific routes, linking cities pairs,
- Direct Operating Cost, (further denoted as DOC)
- Noise,
- Maintenance, etc.

Airlines are looking for aircraft satisfying their needs at best and allowing them to make the maximum profits, which means both minimise operational costs and justify a ticket price as high as possible with good services (new technologies to ensure maximum comfort, maximum speed, etc).

The aim of the airlines is to be competitive, thus they want to fill a short fall of their operational network, in range or in payload, in a short-middle term.

1.1.1.4 Choosing TLARs, manufacturer point of view

The problem is quite different from the manufacturer point of view. Short term has its importance because it is directly related to the program launch and to first aircraft sales.

The manufacturer has always to optimise the product, which has to contain some of the latest technologies, and to be the less expensive, thus it means minimising both Non Recurring Costs (further called NRC), which come from the engineering, and Recurring Costs (further denoted RC), coming from the manufacturing.

Moreover, the manufacturer has to find a configuration that can be suitable for a large number of customers, by diversifying the products, and widening the customer network (see figure 1.5).

Furthermore, long term studies carry more difficulties. Indeed, typical time durations for a new aircraft program are:

- Five or six years for preliminary studies,
- Four years for the development phase,

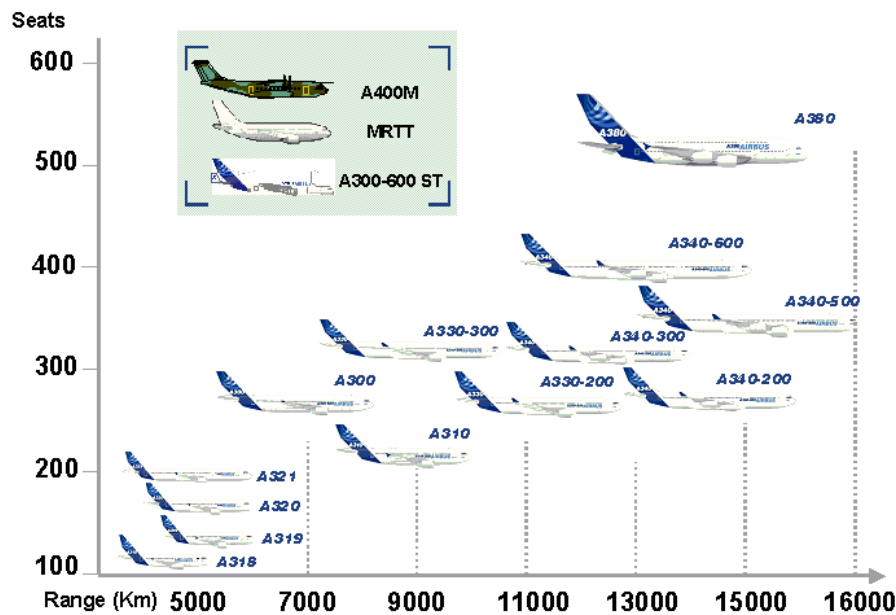


Figure 1.5: Market positioning

- Five years of production for first aircraft,
- Thirty years of operating.

So the total life time for an aircraft is close to fifty years (see figure 1.6).

For the manufacturer, the requirements are to sell a maximum of aircraft, and to produce them with minimum costs.

More requirements, used to translate the airline requirements, are added to those listed previously. They are related to the aircraft operational performances:

- Low speed performances (for instance, take-off performance sometimes limits payload),
- Time duration of a typical mission,
- Operational limitations,
- Number of engines, etc.

This kind of requirements is used to translate the requirements of the airlines, which are to transport the maximum payload with minimum costs. This means for instance to decrease operational costs in minimising the flight time, or maintenance costs, two engines are less expensive to maintain than four engines, etc.

Many other design requirements, related to safety, are also specified by the Federal Air Regulations (FAR) in the U.S. or the Joint Airworthiness Requirements (JAR) in Europe.

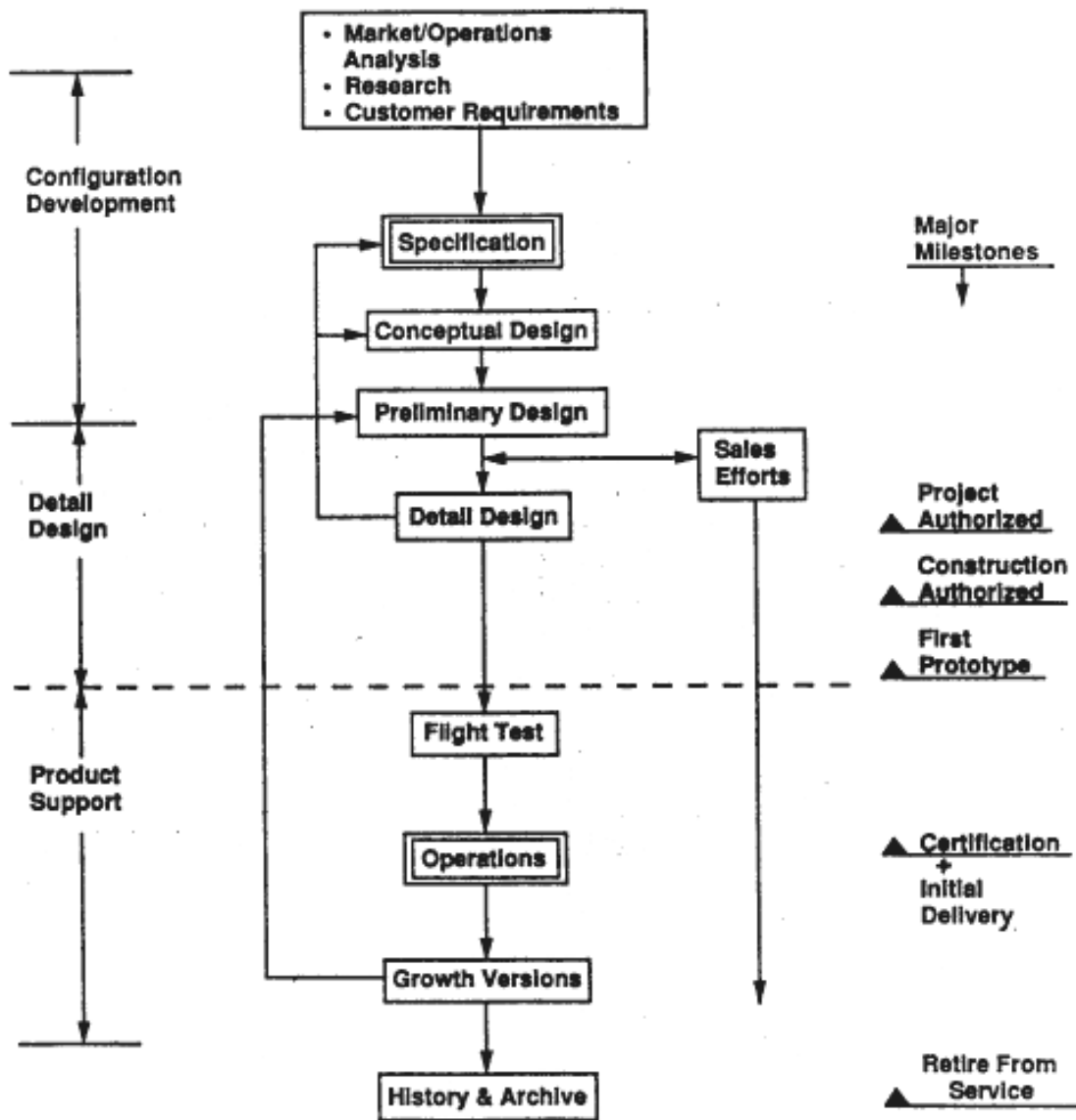


Figure 1.6: Life cycle of an aircraft

When engineers face these considerations, they have two complementary ways of working to define the first new configuration:

- Iterations considering market studies and contact with airlines is one way, oriented towards short-term studies to target the airline future needs.
- The other is to take into account future evolutions of the requirements in the first version of the configuration specifications, to allow family development.

1.1.2 Multidisciplinary aspects

A multidisciplinary problem appears when two, or more, disciplines are involved to collaborate on achieving a common goal, with their own know-how and skills. The final result involving all different disciplines is globally better than when disciplines are considered sequentially, because interactions between disciplines are taken into account.

When defining a new aircraft, the manufacturer wants it to be the less expensive (in producing and operating), the fastest, the most comfortable and also the simplest, the lightest, having the best aerodynamic design. But one cannot make everything best at once. The less expensive aircraft would surely not be the fastest, the most efficient would not be the most comfortable. Similarly, the best aerodynamic design is rather different from the best structural design, so that the best overall airplane is always a compromise in some sense (see figure 1.7).

The compromise is made in a rational way, provided the right measure of the aircraft quality is used. This choice allows to select the aircraft fundamental parameters. Various quantities have been used for this purpose, including those listed below:

- Empty weight,
- Maximum take-off weight, (further denoted MTOW)
- Fuel consumption,
- Cash operating cost,
- Direct operating cost,
- Return on investment, etc.

MTOW has always been a significant parameter to consider to represent the aircraft quality because it has a direct impact on costs. But with the increase of fuel price, MTOW is not as relevant as it used to be. Indeed, to minimise the fuel consumption, a decrease of 2% on mass has roughly the same impact than a decrease of 1% on drag. But engineers have to yield a great effort to improve aerodynamic properties because they are already well-optimised, whereas a decrease mass is easier for the engineers, by using composite for instance. Thus, the right measure of the aircraft quality has to take all these characteristics

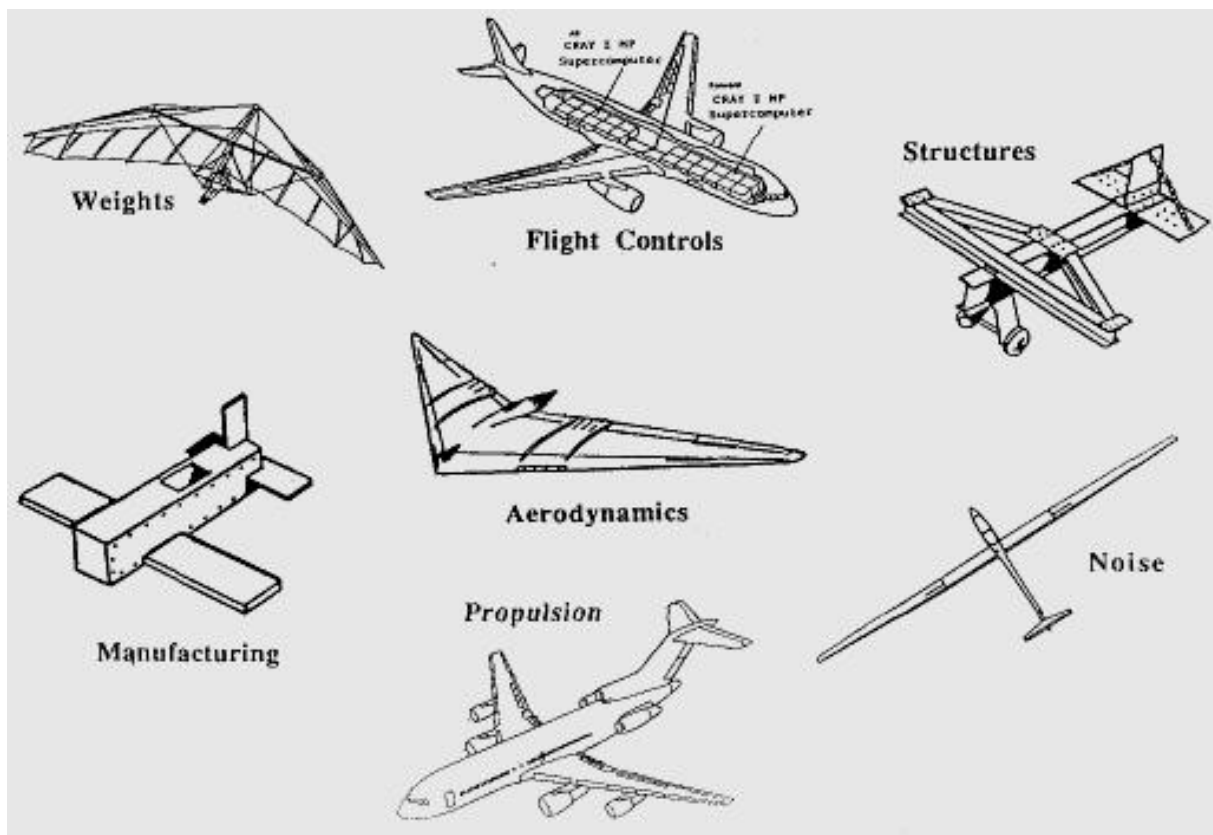


Figure 1.7: Caricatural optimal configuration for each discipline [Nicolai, 1975]

into account.

Currently, direct operating costs, DOCs, are considered as more representative of the aircraft quality because they integrate in the same variable the impact of fuel cost, maintenance, crew or taxes.

DOCs are expressed for a given mission in dollars per flight, or in dollars per nautical mile, further denoted NM, per passenger (one nautical mile is equal to 1852 meters). The main data contributing to the DOCs are (see in [Pacull, 1990], chapter IX):

- Fuel burn,
- Maintenance (of components and engines),
- Crew (technical crew and commercial crew),
- Depreciation,
- Interest,
- Insurance,
- Taxes (landing taxes and traffic taxes).

DOCs are a weighted sum of the data listed above, the weights depend on the airlines, on the aircraft and its operation, on fuel price, etc.

As there are several ways to define a DOC, the optimisation of mission variables leads to different results depending on the chosen DOC.

Today, direct operating costs are widely used as a measure of the aircraft quality.

1.1.2.1 Multidisciplinary issues

Once we have agreed on the definition of optimality, we must find a way of linking the design variables to the goals. For aircraft design, this process is extremely complex. The number of parameters needed to completely specify an aircraft like the A380 is astronomical and difficult to assess, somewhere between 10^7 and 10^9 . So one uses a combination of experience, approximation, and statistical information on similar aircraft, to reduce the number of design variables to a reasonable number. This may range from 1 or 2 for back-of-the-envelope feasibility studies to hundreds or even thousands of variables in the case of computer-aided optimisation studies.

Even when the studied situation, mission simulation for instance, is simplified, the earth is considered as flat, the model is usually complicated. One must generally use a hierarchy of analysis tools ranging from the simplest to some rather detailed methods. For instance, concerning aerodynamic efficiency, you can have three levels of formulation:

- a single line formula,

- a lifting line in 2 dimensions,
- a Computational Fluid Dynamics, further called CFD (for steady and unsteady simulations).

1.1.2.2 Influence on design

One of the most important part of the design process is the conceptual design phase. This involves deciding on just what topology will be used to describe the design. Will this be a flying wing? A twin-fuselage airplane? Often, designers develop several competing concepts and try to develop each in some details. The final concept is “down-selected” and studied in more details (see figure 1.8).



Figure 1.8: Example of unconventional aircraft configurations

For a given concept, in addition to the discrete parameters, the aircraft can be represented through a reduced number of continuous parameters, generally related to the main characteristics of the aircraft: the engine size and thrust, the wing area, the wing sweep angle, the wing span, the fuselage length, or the empennage area.

Geometry is then specified in the preliminary design phase, allowing to assess aerodynamical characteristics of the aircraft. In parallel, weight estimation can be performed. Finally, low speed performances, for take-off or landing phases, and operating cost are calculated.

1.1.3 Design iteration and optimisation

There are several methods used by engineers to choose the design variable values leading to an “optimal” design. In a general way, designing and defining an initial topology of an economical and physical system is a five step process. In this process, engineers take into account that this topology will be the basis of its own detailed definition and will lead to its realisation:

1. defining fundamental variables modeling the essential behaviour of the system,
2. building a model accurate enough to allow a reliable representation of the variable influence on the overall system,
3. defining a mean to measure the quality of the system which depends on these variables,
4. translating TLARs into operational constraints, or into imposed values of design parameters,
5. choosing variable values which lead to an “optimal” behaviour of the system, while ensuring the system meets requirement constraints.

All of these methods require that many elementary analyses are carried out (often thousands of times). The steps 1 and 2 are related to the complexity of the models. This requires that the model is simplified to the point that it is fast enough, but still being meaningful. The number of variables is generally reduced as much as possible to keep the understanding of physical phenomena, while being sophisticated enough to have a reliable representation of reality. Compromise between simplicity and representativeness is difficult to find, and is significant of the overall model quality.

Future project objective is to fix the main characteristics of the configuration which will impact the final product, not to obtain a detailed definition of the system. This is a justification to use simplified models, since they are fast in calculation time.

The steps 3, 4 and 5 lead the engineers to a new problem because finding a good general criterion representating the system quality is difficult, and ensuring constraints satisfaction is not always possible. Moreover, some considerations of importance are not always modeled nor quantified. For example, designing an aircraft is made more complicated by considering an aircraft family at once. This part of the work can be modeled as concept constraints or operating specifications, but it is difficult to quantify.

Nevertheless, evaluations of a future project configuration are still analytic studies where aircraft parameter values are modified. During this process, changing in general architecture choices can be considered, or even changing in some constraints defined in requirements, the aim being to assess their impact on the aircraft definition.

1.2 Needs and motivations

Future project designs are the basis of every new Airbus product. The knowledge on these new products at this stage is not detailed, but technical choices have a great influence on operational and economical performances of the future product and also on difficulties that may occur during the development phase.

1.2.1 Why do most of aircraft look like each other?

At the beginning of Aeronautic History, different kinds of aircraft configurations were built, like on the figure 1.9.



Biplane



The H-4 Hercules, a military seaplane roughly as big as the A380, 1947



The Capronissimo, a nine wing seaplane, able to transport 100 passengers, 1921



The Dornier DO-X, a 12 engine seaplane, able to transport 169 passengers, 1929

Figure 1.9: Examples of old aircraft configurations

To ensure its function of transporting payload from one point to another point of the earth, an aircraft requires the cooperation of four basic functions (see in [Badufle, 2003]):

- The **transport function**, ensured by the **fuselage** which is more or less a pressurised tube with some accommodations inside to allow human life and activity during the journey.
- The **lift function**, ensured by the **wing** which is a sort of plate beam whose position in reference to the air velocity allows to create and control a force that will be used to compensate for gravity and to achieve a desired trajectory.
- The **propulsion function**, ensured by the **engines**, is mainly dedicated to compensate the drag created by the movement in the air and the lift force itself. Aside of this, engines will also generate almost all forms of energy used in the aircraft, as the electricity inside the cabin, or the air heating.
- The **control function**, mainly ensured by the **tail surfaces** attached to the rear part of the fuselage, and also by aileron, etc, will provide an adequate control of the position of the wing in reference to the air velocity. The aim is to control the force generated by the wing that will be used to allow and control the airborne trajectory.

We can see that all functions are ensured by physically different parts of the aircraft. This particular topology is the main characteristic of what is called a “classical configuration” (see figure 1.10). It provides some uncoupling between the four functions, even though, some interactions still exist.



Figure 1.10: Example of a classical configuration

Thus, the classical configuration is characterised by a cylindrical fuselage, a wing under the fuselage, classical horizontal and vertical empennages, and two or four engines under the wing. Actually, this configuration is the one providing the less couplings between the different physics existing around an aircraft. There are other existing configurations answering the four transportation needs described above. The flying wing is one example. The fuselage is included in the wing, which is a delta wing, and there is no horizontal empennage. Thus, the transport function, the lift function and the control function are all ensured by the same

part of the aircraft: the “body” (see figure 1.11).

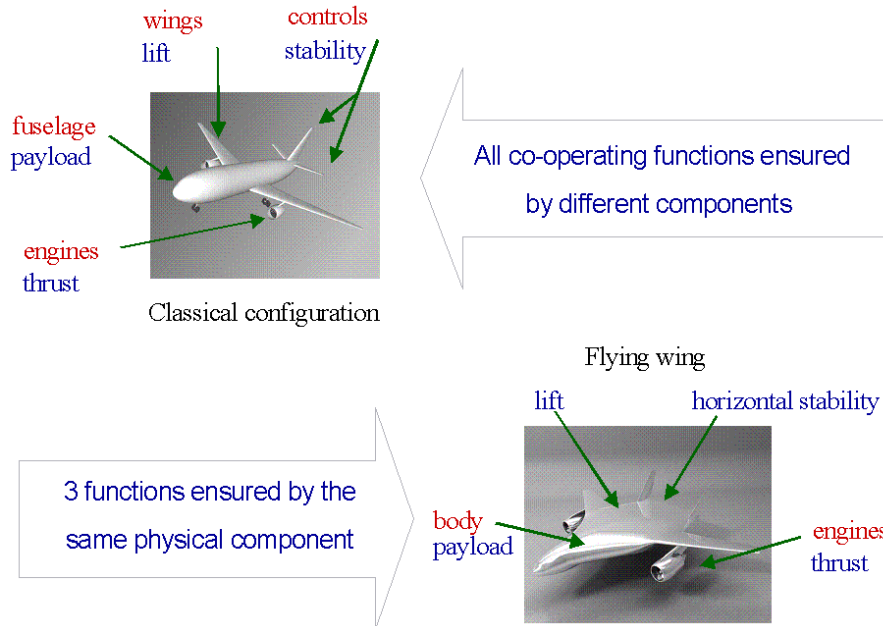


Figure 1.11: Comparison between a classical configuration and a flying wing

Interactions can be seen from two different points of view: operational and design. The operational point of view deals with a fixed aircraft, it enlightens:

- Mechanical interactions: all parts are supposed to be rigidly linked so they exchange mechanical energy,
- Aerodynamic interactions: tail surfaces are in the downwash of the wing.

The design point of view deals with a scalable “rubber” aircraft. Couplings are not only operational but conceptual. Fuel volume and wing structural weight depend on the global weight of the aircraft, more generally: most of internal weights are interdependent. Wing and engine sizes also depend on the global weight through engine thrust and aerodynamic efficiency.

A great proportion of the recent flying civil transport aircraft look like each other because the classical configuration is supposedly the best, and it is the one which has been the most studied, we have a better knowledge on it, we have more experimental data and also a better know-how coming from experiments.

This configuration is the most studied because it is the known one that minimises interactions between the four functions ensured by an aircraft. Thus, the multidisciplinary problem is simpler to study for the classical configuration.

The design process in FPO is based on a multidisciplinary optimisation which brings in interaction physical aspects like Geometry, Aerodynamics, Weights, Structures, Acoustics, etc. The available characteristics at that phase are about one hundred variables, the main parameters are related to:

- wing area and span,
- fuselage length and diameter,
- engine maximum thrust,
- landing gear size and location,
- tail area, etc.

These parameters represent a classical configuration, like on figures 1.12 and 1.13, a cabin layout description and a three view chart.

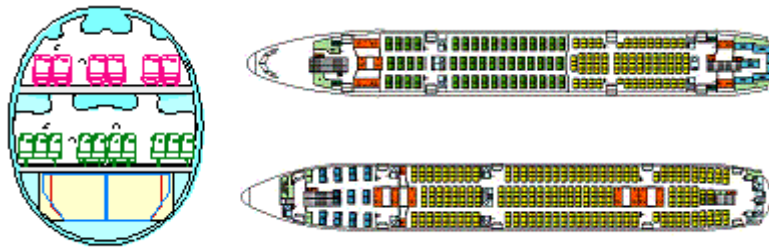


Figure 1.12: Cabin definition of the A3XX project

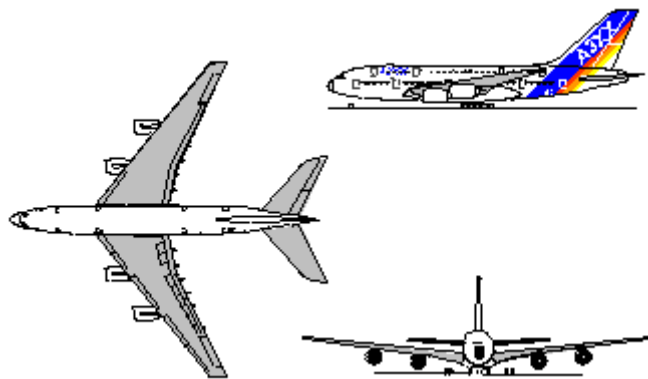


Figure 1.13: Three view chart of the A3XX project

An aircraft is not only represented through its geometry, but also with its properties, calculated thanks to models. These properties are mainly:

- element weights (wing, fuselage, pylons, etc.),
- aerodynamic values (lift, drag, lift over drag, etc.),
- operational performances (range, take-off field length, landing field length, etc.),
- economical performances (costs, product positioning related to already existing aircraft, etc.).

The following section is a description of these models.

1.2.2 Model description

Among all different physics interacting around an aircraft, the main two used to describe an aircraft at future project phase are Aerodynamics and Weights. Thus, we need to introduce the two following models:

- Aerodynamic model provides a relation between dynamic state of the aircraft and forces applied on its surface by the ambient air.
- Weights model provides a relation between external forces and the aircraft structure able to support these forces, like the aerodynamic forces, the ground reaction or the aircraft weight itself.

A complete description of all these models can be found in [Pacull, 1990; Torenbeek, 1982].

First, the data mainly needed to calculate these models are given through a geometrical description of the aircraft.

1.2.2.1 Geometry model

The geometry parameterisation at future project phase is a reduced set of scalars (around 100) and also some discrete values (like the number of engines). This is valid supposing some regularities on the shape of the configuration.

The geometry is basically what can be represented on a three view drawing, like on figure 1.14, containing all what can be determined at future project phase, except the seat layout, which is not represented on the figure.

1.2.2.2 Aerodynamic model

For aerodynamic assessment, only the geometry of the external skin of the aircraft is significant.

Aerodynamic phenomena are extremely sensitive to tiny geometry variations. Simplified models cannot completely describe the external skin with the required accuracy. Hence, strong hypotheses are integrated in the parametrical description of the geometry.

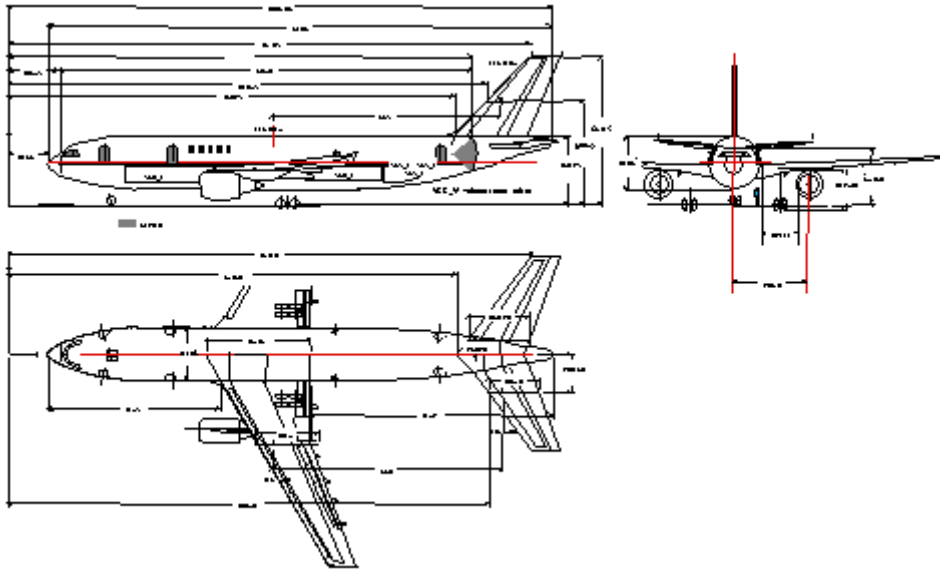


Figure 1.14: Three view graph

Moreover, future project design cannot allow using heavy mesh-based methods to solve the Navier-Stokes equations. Statistical methods are used because of time constraints and of a low granularity of data. These semi-empirical methods are more precise if the studied configuration is close to a known one, included in the statistical data base (see figure 1.15). These statistical data bases contain a set of aircraft whose aerodynamic model is known through flight tests, allowing for aerodynamic model matching.

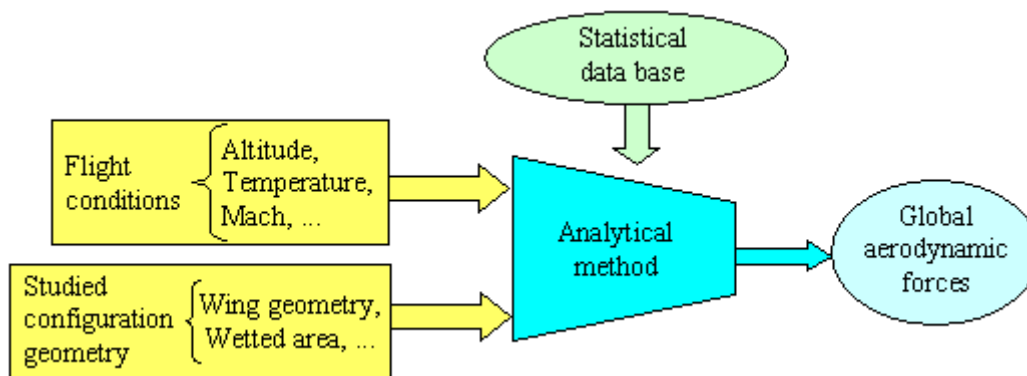


Figure 1.15: Aerodynamic model

Concerning unconventional configurations, simple mesh-based methods, in 2 dimensions or based on linear theory (see figure 1.16) can be used because they are more representative of physical reality and because we do not have any statistical data base.

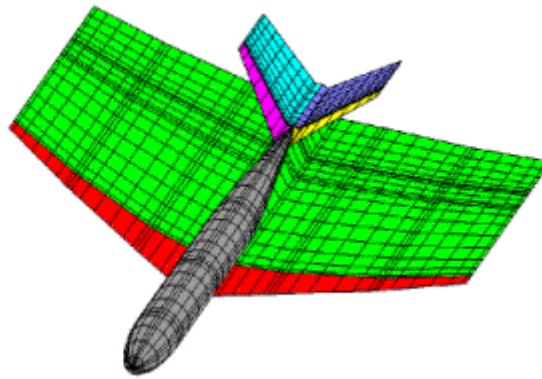


Figure 1.16: Fluid dynamic light modeling with linear methods

1.2.2.3 Weights model

This model is based on structure description. We distinguish two kinds of structures, the primary structure, which ensures the stiffness of the aircraft, and the secondary structure, which contains all structural elements not necessary participating in the stiffness, but hanging on the primary structure. The secondary structure participates in the external shape of the aircraft, to ensure its aerodynamic performances for instance, or its controlling, like the slats or the ailerons.

The figure 1.17 represents the different kinds of structures composing the wing. The grey parts are included in the secondary structure while white parts are in the primary structure. From the FPO point of view, the wing box is globally considered as a beam.

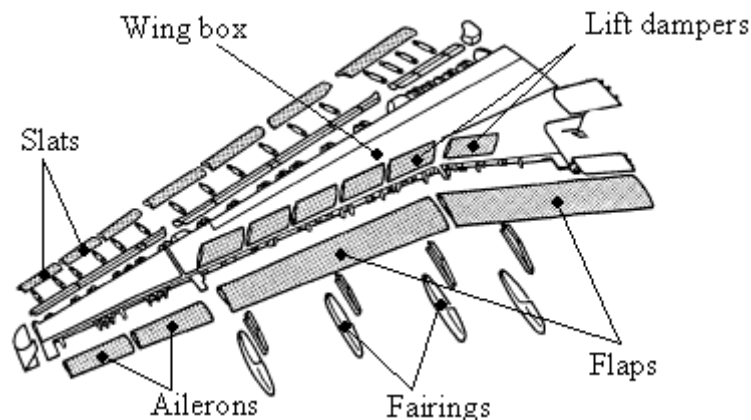


Figure 1.17: Different parts of the structure of a wing

In a general way, wing and fuselage primary structures are evaluated taking into account the envelope of global efforts (weight, inertia, aerodynamic, propulsion) acting on a simplified physical model of the structure. To get real weights, engineers use formulas translating some particular industrial know-how. Secondary structures are directly assessed using statistical

data base depending on relevant parameters. For instance, the following formula is used to calculate the wing weight:

$$M_{wing} = (K_1 A_{wing} + K_2) \cdot A_{wing} + \left(K_3 \cdot \frac{1}{L_{ref} \cdot tuc} \cdot \frac{MTOW \cdot span}{\cos(\Phi)} + K_4 \right) \cdot \frac{MTOW \cdot span}{\cos(\Phi)}$$

where:

- M_{wing} is the wing weight,
- A_{wing} is the wing area,
- L_{ref} is the reference length,
- tuc is the thickness upon chord ratio,
- $MTOW$ is the maximum take-off weight,
- $span$ is the wing span,
- ϕ is the wing sweep angle.

See on figure 1.18 for a graphical representation of these data:

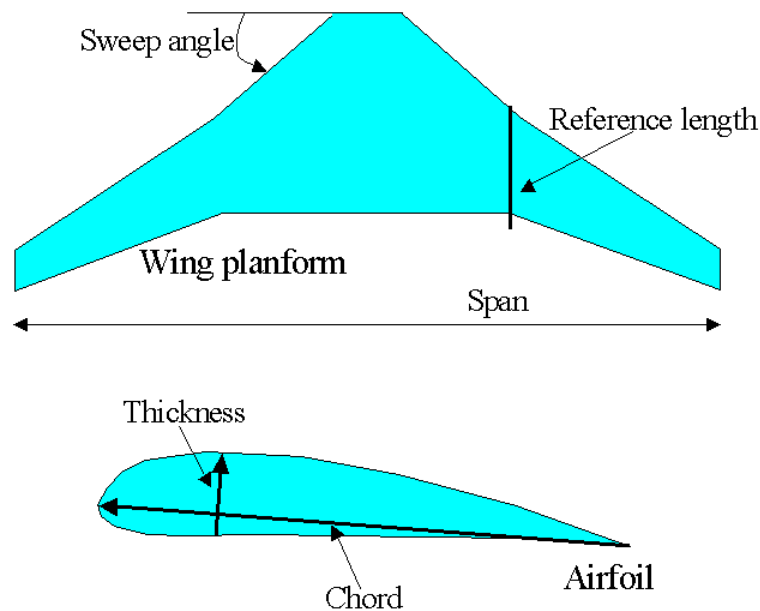


Figure 1.18: Wing shape with its characteristic data

We can see that the wing weight increases when the maximum take-off weight, the wing area, the wing span and the wing sweep angle increase, and when the reference length and the thick upon chord ratio decrease.

The parameter aggregation is coming from engineers know-how, and the coefficients K_i are determined from statistical regressions.

1.2.3 Design processes

The design processes have been established in order to ensure the main constraint satisfaction. Their origin is strongly linked to skill domains of specialised departments. They are all based on iterative research of a unique solution.

The four following processes are considered as the main ones in future project design. They are:

- the Weights calculation for structure design,
- the Mass-Mission adaptation,
- the Optimisation to meet requirement constraints,
- the Optimisation constrained by flight handling quality.

They have been progressively nested into each other as computational performance increased.

1.2.3.1 Weights calculation for structure design

This process is the central one. The origin of this process is the structural design domain. In past studies, it used to be a statistical method in preliminary design to simplify the calculations. Currently, the complete process which answers future project needs, involves specific aerodynamic, kinematic and dynamic models.

The aim of the process is to **minimise the material mass which is necessary to ensure a sufficient stiffness to the structure**, whatever flight conditions are, for a configuration which maximum take-off weight is given (see figure 1.19).

Difficulty comes from aeroelastic phenomena, the structure geometry under loads is modified through deformations which depend on its internal stiffness, and these deformations modify loads repartition in return.

Moreover, this structure has to be strong enough, and thus thick enough, to support loads during the aircraft operation, knowing that these loads depend on the structure weight.

As an illustration of this snowball effect, we can say that an increase of the MTOW makes the MTOW increase. Indeed, we have:

$$\nearrow MTOW \underset{\text{loads}}{\implies} \nearrow M_{wing} \underset{\text{structure}}{\implies} \nearrow MWE \underset{\text{mission}}{\implies} \nearrow MTOW$$

where *MWE* is the manufacturer weight empty.

This coupling, deformation on one side and weight on the other side, is treated as two iterative loops, one inside the other, acting on a flight simulation under high loads. The whole process is included inside a systematic exploration of different flight cases aiming at producing the loads envelope.

The same kind of process exists concerning the fuselage, without the loop on deformation.

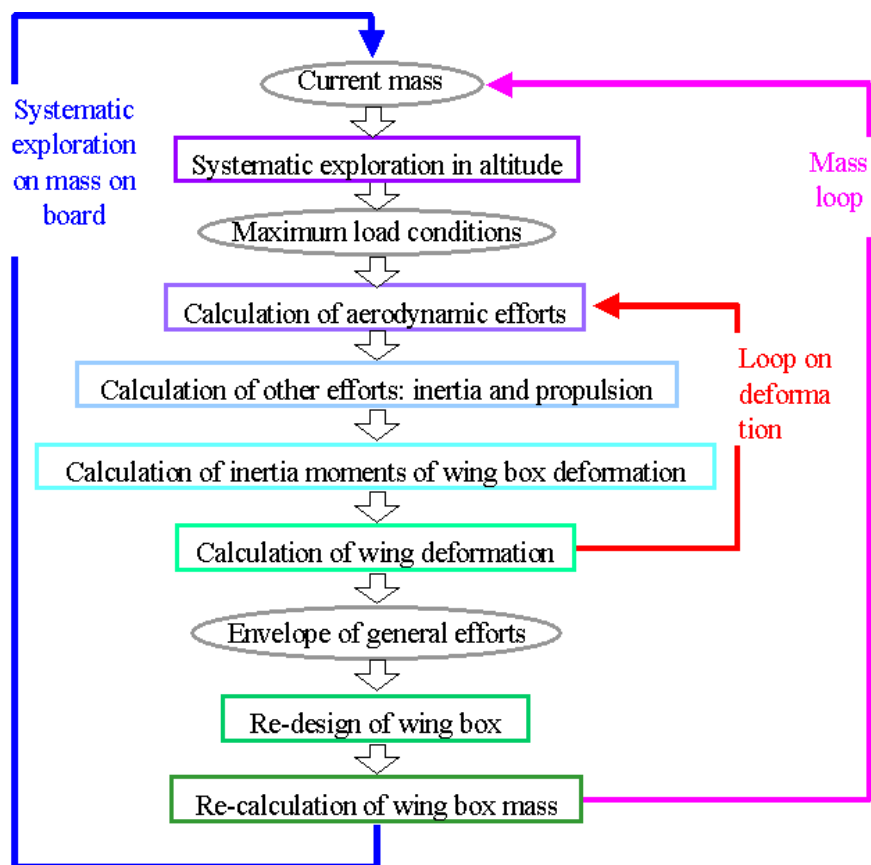


Figure 1.19: Mass loop

1.2.3.2 Mass-Mission adaptation

This process is under FPO responsibility. It is traditionally called the “Mass-Mission loop”.

The aim is to **define the maximum take-off weight of the configuration according to nominal payload and operational range, given by requirements.**

This process combines for two aspects. Mission simulation for a given distance needs some fuel quantity on board, which impacts the maximum take-off weight. And a given take-off weight leads to a certain structure weight, which impacts take-off weight.

This problem is traditionally solved iteratively, as shown on figure 1.20 by the black arrow. The Mission line represents the relation between range and MTOW, and the Structure line stands for the relation between payload and MTOW. The solution is situated where the two linear operational constraints are satisfied simultaneously. There is no need inside this process to perform an optimisation, like a minimisation of errors, to find the solution, a formal iterative resolution of a system is enough.

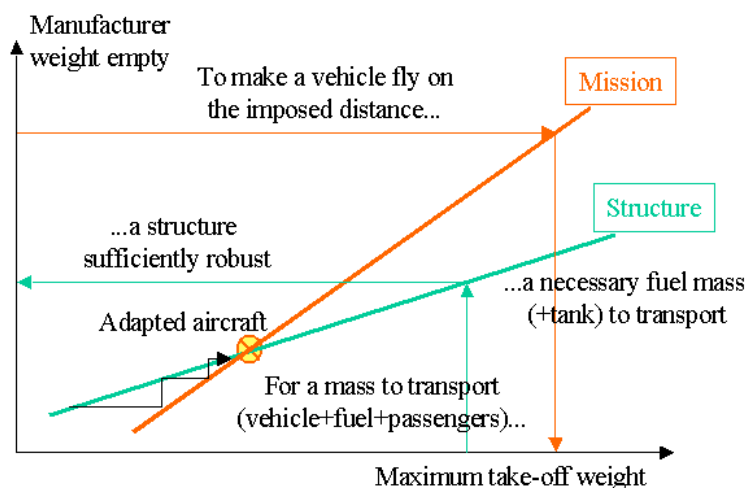


Figure 1.20: Mass-Mission loop

1.2.3.3 Optimisation under operational constraints

This process is based on know-how of the Operational Performance Department.

The aim is to **define at the same time at least the wing area and the engine thrust, while ensuring that the aircraft fulfills the required operational performances.** Some other parameters can be optimised within this process, such as landing gear or control surfaces.

This process is based on a constrained optimisation, because most of the requirements are expressed as operational constraints and involve both wing area and engine thrust (see figure 1.21). Operational constraints are calculated as mission trajectories, or through flight

simulations where the aircraft is represented by a point located at the aircraft centre of gravity. An example of operational constraint is the approach speed, calculated through a landing simulation, and it should be lower than the required value.

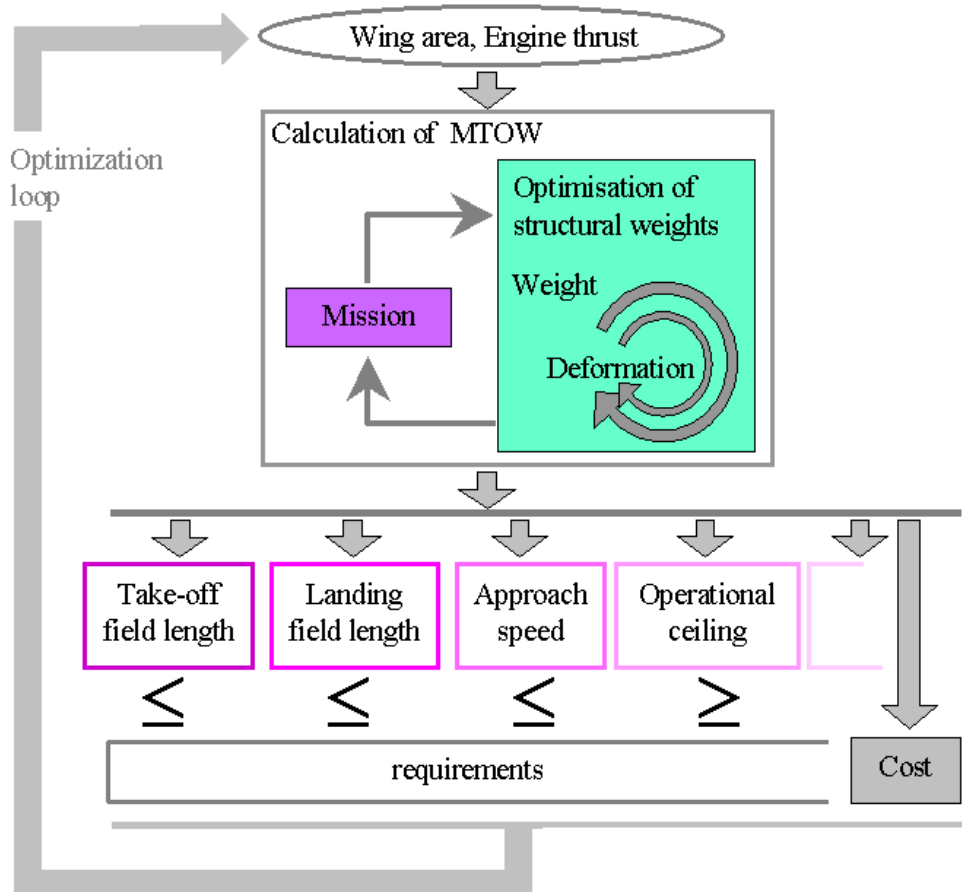


Figure 1.21: Optimisation loop

The main difficulty in this design process is to ensure constraint satisfaction. Indeed, some constrained variables cannot be calculated while some constraints are not satisfied, there is sometimes no numerical solution. For instance, for some large MTOW, it can even be impossible to take-off, thus the take-off field length, TOFL, performance is not even evaluable.

This kind of issue appeared when engineers first built their multidisciplinary integrated tool. To avoid it, engineers have to find an initial point close enough to the solution, or to reduce the research space.

This optimisation loop contains the mass-mission loop, which also contains the structure double loop.

Remark 1.2. The wing area and the engine thrust are the most sensitive parameters acting on the aircraft performances. They are always degrees of freedom in any constrained optimisation. But some other parameters of importance, as the wing sweep angle or the wing span,

can be degrees of freedom of the optimisation. Thus, with an increasing number of degrees of freedom, the convergence may be more difficult, slower, or can even fail.

1.2.3.4 Optimisation under flight handling quality constraints

This process is based on know-how of Handling Quality Department.

The aim in FPO is to **define at the same time empennage areas and longitudinal position of the wing, ensuring the aircraft equilibrium and that the aircraft is able to manoeuvre safely.**

Most of handling quality constraints are simulations with strong dynamics, which need a large amount of data. It is generally not available in future project studies. Thus, this optimisation is based on knowledge of a particular kind of configuration.

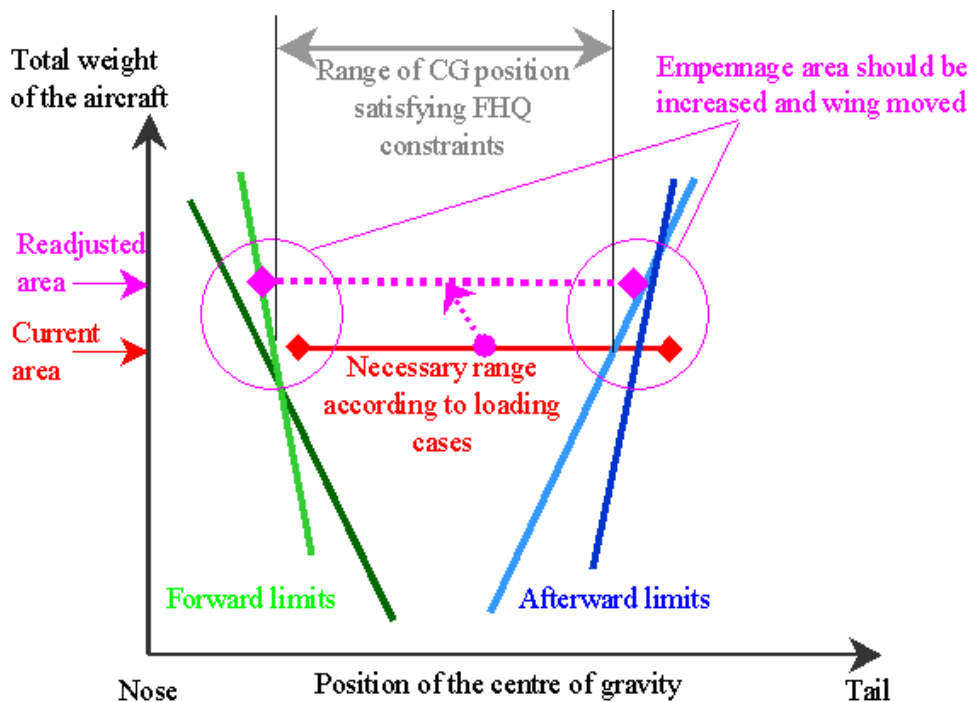


Figure 1.22: Flight handling quality loop

The figure 1.22 is an illustration of this process. It is called a “cissor diagram”. As the position of the centre of gravity is changing with the different way to load the aircraft, each linear constraint expresses a limited position for the aircraft centre of gravity, beyond which the constraints of equilibrium or of manoeuvrability, are not satisfied anymore.

All the loadings cases give the range of the centre of gravity. If the required range is bigger than the possible one, the empennage area should be increased and the wing moved. These changes modify the diagram and repeating this step will make the process converge.

The aircraft is optimised when the cissors on the figure are tangent to all the necessary position range of the centre of gravity. Moreover, this overall process is performed under the condition to minimise the MTOW, *i.e.* empennage size in this context.

1.2.3.5 General considerations on design processes

The figure 1.23 is an illustration of the way the processes are nested. To calculate the aerodynamic variables, we mainly need to know the geometry. To calculate the structure weights, we need to know the areodynamics, but it is not enough. We also need to know the MTOW, which is calculated in the Mass-Mission loop.

The same scenario is repeated in the last two loops (as shown in figure 1.23).

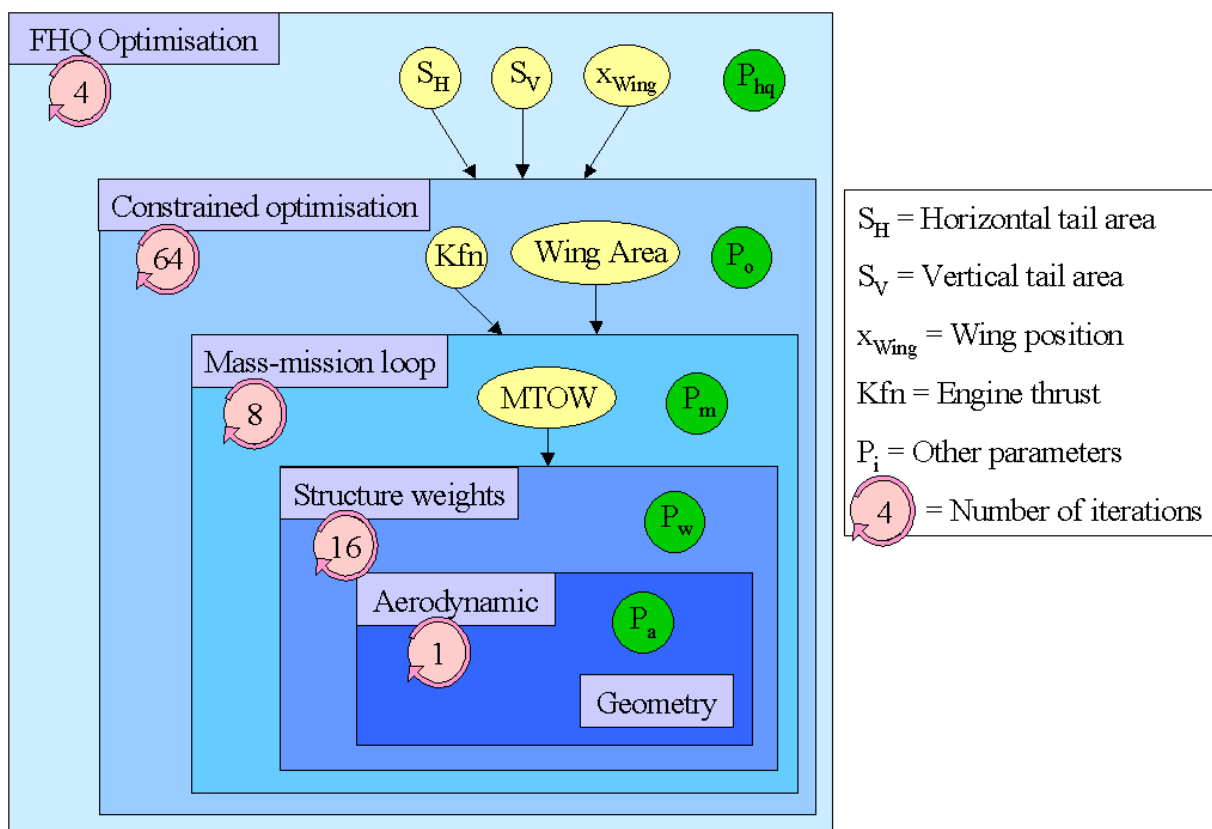


Figure 1.23: Design process organisation

To summarize, the geometry is necessary to calculate aerodynamic properties, which allow then to assess weights when you suppose the MTOW known. Then, you can calculate the performances of the aircraft and verify constraints are satisfied, and finally, the configuration is optimised under flight handling quality constraints.

Each stage of this process has to iterate several times before convergence, for instance on figure 1.23, are estimated the iteration times of each loop, with 8 times for the mass-mission loop for example. Finally, if the complete process is launched, it will require about 2^{15} calculations of the geometry, which means more than 32700 calculations. But this estimation

of the number of calculations is a very rough order of magnitude.

This organisation has emerged from the knowledge engineers had on the problem. It has been developed because of the need to answer the problem of designing an aircraft, but if you only consider the mathematical equations, the complete system can be solved using a different organisation of the equations. This organisation is an inheritance of the skill and knowledge cut-out in the design office, of the models construction and of current mathematical tools. But it may not be adapted for an exotic configuration, like the flying wing, because more couplings appear (like between the performance optimisation and the handling quality optimisation).

1.2.3.6 Conclusion on design processes

The four processes that we explained previously enable engineers to find values of the main parameters of the aircraft configuration, like maximum take-off weight, lifting surfaces areas or engine thrust. Many combinations of values for the design parameters are tested, and the frozen configuration, prior to more detailed studies, will be the one optimising the chosen criterion, or will be a compromise between several criteria.

Remark 1.3. The interweaving of the four processes leads quickly to a huge number of calculations of the most internal process. The main consequence is an increase of calculation time which is not admissible from the designer point of view.

Remark 1.4. The global resolution is not very stable because of the evaluation of configurations which do not satisfy intrinsic constraints of some methods. Thus, these configurations have some variables that are not calculable and in these cases, the global resolution fails. For any optimisation loop, the information is lost because the current optimisation process is a simple point iteration.

Remark 1.5. Experiment shows that the current criterion, a DOC, does not vary much around optimal points, the optima are flat, thus, engineers are looking for complementary means to classify solutions.

1.2.4 Known issues with the existing processes

1.2.4.1 Convergence of internal resolution processes

The first source of problems comes from the internal resolutions contained in the design processes, which are used to ensure the satisfaction of an internal equality constraint for example.

The numerical exploration of the design space leads to try to evaluate some properties of configurations which are meaningless physically speaking, for instance when the engine size and the lifting area are too small to be able to take off at a given weight. The consequence is to make the convergence fail in internal processes.

An example of an internal process is the mission calculation in the Mass-Mission adaptation. It is an important part of some constraint assessment. A mission contains the following phases:

- taxi-out,
- take-off,
- initial climb,
- first climb segment,
- second climb segment,
- cruise at different flight levels,
- descent,
- approach and landing,
- taxi-in.

The figure 1.24 illustrates a typical mission profile for a supersonic aircraft.

In a mission simulation, engineers calculate the necessary fuel and the payload, knowing the take-off weight, the range and the OWE.

$$\left. \begin{array}{l} \text{Take-Off Weight} \\ \text{Range} \end{array} \right\} \longrightarrow \left. \begin{array}{l} \text{Fuel} \\ \text{OWE} \end{array} \right\} \longrightarrow \text{Payload}$$

If the imposed parameter in a mission calculation is the payload, then the entire mission is included in a resolution loop to calculate the take-off weight, if the range and the OWE for instance are given. But other resolutions are hidden inside some mission phases.

Each phase can possibly contain complex resolutions, for instance the initial cruise altitude is the altitude minimising fuel consumption per distance unit, further called *specific air range*, while satisfying climb rate and flight ceiling constraints. Hence, this simple climb segment is obtained through a constrained optimisation.

This internal process example illustrates that the current design process can spend many time in optimising some internal properties of a configuration that finally will not succeed in satisfying some operational constraints.

Besides stability and calculation time problems, the analysis of the numerical environment raises other problems, like:

- the resulting information of the current mono-criterion optimisation as a central process of the overall aircraft design is incomplete. In future project phase, engineers have to provide a set of solution design points.

uncertainty on the results. Our knowledge on the internal mechanism of the multidisciplinary optimisation does not allow us to have a vision on the uncertainty propagation through the global process. This uncertainty information would help us improving the solution robustness.

Let x the design parameter vector, f the model functions, y the evaluation outputs and ϵ the uncertainty, then we have

$$x + \epsilon_x \xrightarrow{f + \epsilon_f} y + \epsilon_y = (f + \epsilon_f)(x + \epsilon_x)$$

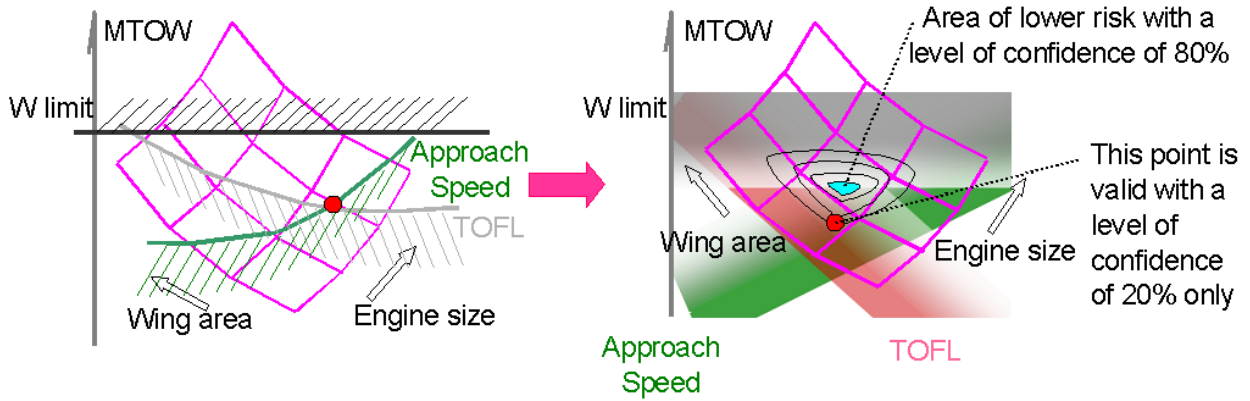


Figure 1.25: Deterministic and uncertain carpets

On the figure 1.25, two carpets are represented. The degrees of freedom are the wing area and the engine size, and the constraints are the TOFL, the approach speed and a maximum weight. The aim is to minimise the MTOW.

On the left carpet, the constraints are deterministic, thus the optimum is situated at the intersection of the constraints. On the right figure, uncertainties are taken into account, thus constraints are not straight lines anymore, they are blurred, and the optimum the more robust is not situated at the intersection of the previous deterministic constraints neither.

The aim of this part is to introduce tools aiming at introducing this kind of information in the early phase of the design process.

1.2.4.4 Nothing to help in choosing the best model of aircraft

The current evolution in FPO is to develop models that are more and more detailed, but the reliability of results coming from these complex models has not been assessed. The main advantage of this increased complexity is to have access to different modeling levels.

We claim that the rule “The more complex the model, the more accurate” is not true everytime. Model complexity often hides our lack of knowledge on the way of working of the global design process.

1.2.4.5 Black box effect of processes included in specialised modules

All calculation modules of operational constraints and, most of the time, of structural design, have their own resolution algorithm (like a fixed point method, Newton, etc.). Their stopping

criterion differs from one to another, and are not well-adapted for the use of these modules as evaluation function inside a higher process. Moreover, the understanding of the overall process evolution is impossible because of the module integration as black boxes in a multidisciplinary context.

1.2.4.6 Non-homogeneity of the aircraft representation

The splitting in modules is defined by know-how on specialised fields. But it leads to the reproduction of some models and of some important functions in a non-homogeneous way. Following are two examples to give an illustration:

- The Aerodynamics and Weights models integrate both their own geometric representation of the configuration. Their parameterisations of the geometry are different from the geometric representation used in the process of the configuration optimisation.
- All calculation modules of operational performances duplicate a calculation of flight equilibrium coming from different sources.

1.2.4.7 Conclusion

The first point leads to search for new criteria to perform optimisation, and to try new mathematical tools to help in decision making.

The other two points are related to general problems of uncertainty control in future project design. Understanding of uncertainty propagation in design process is also linked to the notion of technical risk. In this case, we have to select a solution which minimises the consequences that may occur when the configuration does not achieve the performance goal.

The last points listed above are structural problems, the design tool has to be recoded to overcome them.

1.2.5 Proposed evolutions and solutions

1.2.5.1 Proposed evolutions

We identified three main axes of evolution to improve general design process and to answer the issues listed previously:

1. Introducing new mathematical tools like **multicriteria optimisation**, uncertainty management, constraints propagation, **robustness assessment**, risk quantification, etc.
2. Processing of **unconventional configurations**, like the flying wing. Particularities of these configurations modify the architecture of the general design process, like taking into account some handling qualities during the cabin definition.
3. Designing an **aircraft family** derived from a central product. Members of a family share some components like the wing size, the forward fuselage part or the empennages. Thus, some degrees of freedom disappear when designing each aircraft of the family,

but an active constraint on one member of the family can have an influence on performances and costs of the other aircraft. The consequence of deriving aircraft from a basic one is the suboptimality of the performances versus optimising all aircraft separately, but which is much more costly.

In a general way, the evolution of design methods tends to increase complexity in the general calculation process, so we currently need an important rationalisation of models and of calculation processes.

1.2.5.2 Multicriteria optimisation

The research of new optimisation criteria evolves with development of new design methods, so we decide to introduce new techniques of multicriteria optimisation based on already existing criteria which seem to be relevant during the study:

- Maximum take-off weight,
- Fuel burn,
- Cash operating cost,
- Direct operating cost,
- Recurrent cost,
- Non-recurrent cost,
- Robustness,
- Family extension capability, etc.

1.2.5.3 Uncertainty management

In this domain, two main axes have been defined. The first one aims at introducing statistical tools of uncertainty calculations, which do not currently exist in FPO. The second axis aims at taking advantage of the different available models with different levels of complexity, when they exist.

Analysis functions will be introduced, coming from statistical tools. They will be used when knowing the uncertainty on different data or intermediate variables of the general design process, to deduce the uncertainty that we can expect on the results.

- Collecting information will be done to provide enough reference data on model and input data uncertainty.

- The design process will be improved to perform uncertainty propagation. Thus the uncertainty on results can be translated into robustness of the design points. This information can be used for instance as a new criterion of the optimisation problem to decide between the admissible configurations, which one minimises the global uncertainty.
- A specific user interface will be associated with this new function handling.

The general calculation process will be rationalised in such a way that several levels of models may be used concurrently according to several criteria, as proximity of the solution, or to compare uncertainty. A typical example is the multiscale process, simple models will be used when the current configuration is “far” from the admissible solutions, and then, when the configuration gets closer to the admissible solutions, more complex models will automatically be used to refine the solution. It is a coarse-to-fine approach.

- This important adaptation of the process will exploit the developing know-how in the uncertainty management.
- We shall integrate the notion of linked constraints to ensure that a constraint will be calculated only if its hypotheses coming from other constraints are already satisfied.
- A mean of intervention for the user will be introduced, for cases like skipping on outlier points in the optimisation process.
- Adapted tools of visualization will be defined and developed to help the user keeping control on the calculation process.

1.3 Global context

1.3.1 Other projects related to the thesis

There are several Airbus-internal or european projects interacting with the thesis subject.

The thesis participates in the european project *VIVACE*, *Value Improvement through Virtual Aeronautical Collaborative Enterprising*, and it has the same main objective, adapted to the FPO needs. The aim of *VIVACE* is to reduce global costs of an aircraft program by improving the quality and by a deeper exploration of preliminary studies. The main result will be an aeronautical collaborative design environment and associated processes, models and methods.

The aims of the thesis related to *VIVACE* are to:

- Improve technical risk perception linked to a particular configuration, and implications of technical choices on a given configuration.
- Compare in a homogeneous way conventional and unconventional configurations which answer the same requirements.

- Reduce maintenance and development costs of design tools thanks to an object driven approach of the aircraft model structure.
- Reduce time scale.

The thesis is also related to the Airbus internal project *ODISA*, *Object Driven Integrated Sizing and Analysis*. Basically, the objectives of the thesis are the same as the internal project *ODISA*.

The aim of *ODISA* is the creation of an **enhanced aircraft conceptual design environment**. *ODISA* will enhance the Airbus transnational consistent approach for a future **integrated sizing and analysis environment**, to finally have a common design tool shared by the different national FPO groups in England (Filton), France (Blagnac, S^t Martin) and Germany (Hambourg).

1.3.1.1 Problems to be solved and opportunities

Currently, for aircraft conceptual design and analysis, limitations are due to:

- Simplification of decision criteria, leading to application of margins for risk management;
- Difficulty to introduce unconventional design criteria and target functions;
- Complex but static design process, again leading to application of margins for risk management;
- Confinement in known cases which were used for validation, and thus, risks are increased in case of extrapolations;
- Difficulty to evaluate unconventional configurations at consistent quality with conventional configurations.

The new approach expected to be developed through *ODISA* will allow to:

- Introduce new technologies (IT, mathematics, etc.);
- Investigate different concepts consistently and at a higher level of detail;
- Introduce new aircraft development scenarios through recognition of more design parameters for a more robust design;
- Better evaluate risks before programme launch;
- Reduce technical risk after project milestone M5 (see figure 1.1) by providing decision consistency before M5;
- Cut back the global aircraft cost (development time and entry in-service maturity).

1.3.1.2 Potential benefits

At aircraft level:

- Potentially shorter and certainly higher quality concept phase,
- Higher maturity and solution robustness at start of development phase,
- Quicker identification of challenges and/or showstoppers.

In terms of product strategy:

- Support strategic decision process,
- Reduce aircraft development risk (and consequently cost),
- Improve aircraft environmental performance (as new design criteria).

Creation of a new design environment providing:

- Flexibility to cover design of new aircraft concepts (unconventional configurations),
- Capability to cover more complex design processes,
- Provisions to adapt to changing organisations including needs for extended enterprise operations. The extended enterprise is in fact the main enterprise, here Airbus, which is decomposed in several skill groups working together, and also the subcontractor enterprises, or the collaborative laboratories,
- Better requirements capture and tracking with proposed solutions.

1.3.1.3 Restructuration of design tools

In current tools, software modules are representing specialised domains, like the Aerodynamics, or Handling Qualities. We want to change this organisation by considering needs in calculation concerning the aircraft model, or design constraints. Future choices taken to make changes in this direction will lead to a complete restructuration of integrated design environment.

The **aircraft model** will be an explicit and centralised module to avoid multiple models in different specialised modules with a different parameterisation to represent the same physical quantity (like the geometry for instance).

- The general organisation will be decomposed according to the aircraft components and sub-components, for instance fuselage, nose, rear, wing, etc.
- Each component model will have the structure of an object [Druot, 2002].

- Each object will manage all physical aspects associated with the component: geometry, structure, weight, loads, etc.
- Calculation functions associated with models will be evolutive in case of configuration modifications, etc.

All calculations of **design constraints** will be treated as a simulation applied to the Aircraft object. The calculation functions of these simulations will be organised following four levels:

- Static simulations or geometric simulations (for instance, taxiway width needed for a turn)
- Static or dynamic simulations of the operating aircraft, where the aircraft is considered as a material point, like in mission calculations. For instance, to calculate a flight equilibrium in a stabilised step cruise.
- Static or dynamic simulations of the aircraft, where it is considered as a widen solid, when the influence of deformations is unknown or ignored. For instance, to calculate the dynamic equilibrium around pitch axis because we cannot calculate inertia momentum around pitch axis with a material point.
- Static or dynamic simulations of the flexible aircraft, the most complicated case, the flexible aircraft is modeled as a set of solids. For instance, to simulate a step cruise change, the acceleration increases, thus the wing is bending due to the increase of aerodynamic loads.

1.3.2 Description of the models

Three different models of aircraft were used during this study. The first one was the implicit model included in *AVION*, the current FPO tool, and then, as soon as they were available, the study was based on the *USMAC*, *Ultra Simplified Model of AirCraft* (from June 2005), and on the *SMAC*, *Simplified Model of AirCraft* (from September 2005).

1.3.2.1 Current Future Project Office model

One of the current models used in FPO is an implicit model which is integrated in a specific platform developed in FPO. The main modules composing this software are the four skill design processes described previously (like the Aerodynamic model, the Weight model, etc.), plus some additional modules for a better description of the whole problem to be solved.

Whenever it is possible, redundant variables are gathered as one, and some equations are manually added to link some variables which have the same physical meaning, but differ from a linear relation.

This model has no particular structure. All variables coming from different physical models are included in the same list without any skill attribute. The number of variables in this model

can differ according to the complexity of the modules that engineers use, but generally, it is around 10^3 variables, and the calculation time to evaluate one design point is roughly 5 seconds.

This implicit model is included in *AVION*.

AVION is the current tool used in FPO. Its architecture is centralised around a calculation kernel, *ForTab* (*Fortran-Tableur*) (see in [Druot, 1993]), and composed by a cluster of tools including a graphical interface (see in [Chabot and Obin, 2003]).

The main characteristics of *AVION* are:

- It is a structurally open environment that allows new models to be plugged in, avoiding development task outside the new models themselves.
- It does not contain any built-in process to let the user free to define its own processes.
- It allows any **direct or reverse calculation of any input or output scalar variables of any linked module or interpreted relation**. The user defines freely the problem to be solved or optimised, based on available modules. Thus, with the same set of equations given to *ForTab*, you can make a calculation in the direct mode, you know the geometry of the aircraft and you want an assessment of its properties. Or you can make a calculation in the reverse mode, you know the performances the aircraft should achieve, and you want to know its geometrical description.

Within *AVION*, it is possible to handle multidisciplinary optimisation process as soon as this process can be formalised as a sequence of system solvings embedded into an optimisation loop. Indeed, the formulation of the problem is focused on disciplines analysis, by solving their equation systems and intrinsic equality constraints, and on interdisciplinary consistency constraints, and an optimisation algorithm ensures the design constraints to be satisfied at the system level.

This formulation is called Multidisciplinary Feasible, further denoted as MDF (see in [Alexandrov and Lewis, 1999; Cramer et al., 1994; Sobieszczanski-Sobieski and Haftka, 1997]). We will explain its principle in more details in the following chapter, section 2.1, page 61.

The figure 1.26 is an illustration of this formulation process within *AVION*. The degrees of freedom, like the wing area on the figure, are given to the disciplines as inputs. The MDF formulation ensures that disciplinary intrinsic constraints and interdisciplinary consistency constraints are satisfied, on the figure in the green box. Finally, the next iteration is conducted by the optimiser, in the biggest blue box on the figure, to ensure design constraints satisfaction and to minimise one objective.

One feature of this particular design problem is the possible presence of local optimisation loops in some discipline modules, like in the calculation module used to evaluate the TOFL.

The process is also containing what is called a “smart reanalyser” in [Sobieszczanski-Sobieski and Haftka, 1997]. Only part of the original analysis affected by the design changes

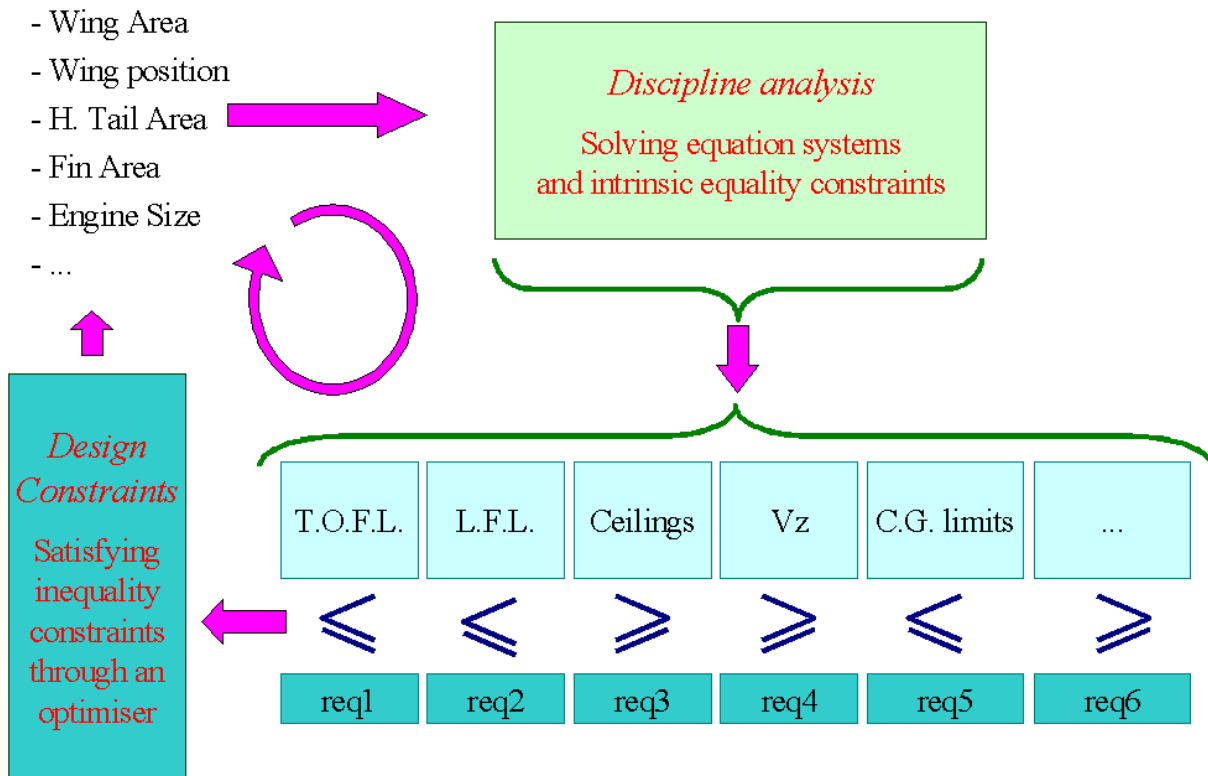


Figure 1.26: Iterative process illustration

is repeated, and thus, only variables affected by these changes are again computed.

The second model of aircraft which was used during this study is the *USMAC*, the Ultra Simplified Model of AirCraft.

1.3.2.2 USMAC (Ultra Simplified Model of AirCraft)

The *USMAC* was defined to answer one specific need, being as simple as possible in term of computations, but being sufficiently complex to keep illustrating the complexity of a multi-disciplinary design problem.

Thus, simplifications could only be done on the models themselves, not on the relations and couplings between the disciplines.

The *USMAC* is a set of *Scilab* (see in [INRIA Copyright, 1989]) functions that answers the following requirements:

1. Capture most of the relevant aspects of preliminary aircraft design:
 - Multidisciplinary approach,
 - Heterogeneous data,
 - Non linear functions,

- Formulated as an inverse problem.
2. Be as simple and fast as possible in term of computation;
 3. Be tunable in reference to a given set of data.

The *USMAC* Package provides a multilevel set of functions that is able to run most important computation processes currently used in preliminary design:

- Given configuration analysis,
- Mass-Mission adaptation,
- Equality constraint satisfaction,
- Optimisation of configuration.

The *USMAC* is simple, computation time efficient and easy to manipulate. Therefore, as it is a too simple modeling, and there are too few parameters to represent the aircraft, it does not fit aircraft studies used in an industrial context. But it is an adapted research platform, and it allows to test some existing mathematical methods which had never been tried in a future project design tool.

1.3.2.2.1 Inputs The *USMAC* has around a dozen of input parameters that are design parameters for fuselage, propulsion, wing geometry and flight conditions (see table 1.1).

Fuselage data	Npax NpaxFront Naisle	Number of passengers Front passenger number Number of aisles
Propulsion data	FNslst BPR ne	Sea level static net thrust of one single engine Engine bypass ratio Number of engines
Wing geometry	Awing phi span tuc	Wing area Wing sweep angle Span Thick upon chord ratio
Flight conditions	disa mach alt	Delta ISA Mach Altitude

Table 1.1: USMAC input variables

FNslst is a reference thrust used for take off.

BPR is a measurement that compares the amount of air blown past the engine to that moving through the core. Higher bypass ratios generally infer better specific fuel consumption as an increasing amount of thrust is being generated without burning more fuel.

disa is the temperature shift in reference to the International Standard Atmosphere representation.

1.3.2.2.2 Outputs The *USMAC* calculates the performances listed in table 1.2.

Vapp	Approach speed
TOFL	Take-Off Field Length
Kfn	Cruise thrust
Vz	Climb speed
Kff	Fuselage fuel ratio
MTOW	Maximum Take-Off Weight
OWE	Operational Weight Empty
PL	Payload
Fuel	Fuel
RA	Range

Table 1.2: USMAC output variables

Kfn is a scaling factor on engine thrust

Kff is the fuselage fuel ratio related to the fuel contained in the wing

1.3.2.2.3 USMAC tuning The *USMAC* can be tuned versus any aircraft of the Airbus fleet database. The database contains two categories of data: **basic identification** data, used to identify an aircraft and **technical data** used to characterize an aircraft.

Basic identification data describe an item of the aircraft model or aircraft configuration. An aircraft is fully identified by a complete set of identification data (see table 1.3).

Model options	Configuration options
Manufacturer: Airbus	Fuel option: 1 ACT
A/C Type: A340	Layout: 298 pax
Version: A340-600	Under floor: LD3 + pallets
Engine: Trent 500	Weight rules: AEG LR
Weight variant: 275 t	Mission rules: Marketing LR

Table 1.3: Basic identification data

ACT is the Auxiliary Centre Tank

pax is the number of passengers

LD3 is a standard Load Device, or container

AEG LR are Airbus Evaluation Guidelines for Long Range

LR means Long Range

An aircraft is described by a set of technical data. Technical data items have been split into 6 categories:

- Geometry,
- Weights and fuel,
- Payload,
- Aerodynamics,
- Engine,
- Performance.

1.3.2.2.4 USMAC architecture There are several levels of functions in the USMAC architecture, **basic functions** are called by **domain-level functions**. **Solving functions** are used for the mass-mission loop and there are also **optimisation functions**.

The first level of functions are the basic functions. They can be split into three main categories:

- Definition,
- Regulation,
- Models.

Definition functions are stable and not sensitive to uncertainty.

Regulation functions are less stable than definitions, since regulations may change, but it is very rare.

Models are composed of simple equations from physic laws, and from statistical regression.

- Physic laws are very stable, but models that are deduced from these physic laws are based on hypotheses and approximations. That is why these models are subject to changes.
- Statistical regressions are quite sensitive to changes. If the database is modified, statistical functions are modified.

The models are the most sensitive functions to uncertainty.

1.3.2.2.5 Basic functions The *USMAC* is a set of about 100 basic functions. Most of them are not longer than a single line expression, for instance, range estimation is based on Breguet-Leduc formula. A third of these equations come from statistical regressions computed out of Airbus fleet data (see in [May, 2005]).

Following are examples of basic functions. An exhaustive list of basic function signatures can be found in appendix A.1, page 221.

Definition *lod* is the lift over drag ratio. The function signature is:

```
function [lod] = lift_to_drag_#0(cz,cx)
```

and the function contains one line of formula, which is:

$$lod = \frac{cz}{cx}$$

where *cz* is the lift and *cx* is the drag.

Regulation *kvs_TO* is a security coefficient with respect to stall speed at take-off. The function signature is:

```
function [kvs_TO] = Kvs_Take_Off_#0()
```

and the function contains one line of formula, which is:

$$kvs_TO = 1.13$$

Models Model basic functions are composed by physic laws and statistical regressions. We give here three examples of physic law basic functions, and one example of statistical regression basic functions:

1. Physic laws

vsnd is the sound velocity, equal to $\sqrt{\gamma RT}$, where γ is the adiabatic index, or isentropic expansion factor, R is the universal gas constant, and T is the temperature.

The function signature is:

```
function [vsnd] = sound_velocity_#0(Tamb)
```

The following two examples are two model functions to assess geometry or mass variables. The aim is to show the coarse granularity of these models.

2. Geometry model example

This function allows to calculate the length of the fuselage, *lfus*, when knowing the number of passengers, *Npax*, the front passenger number, *NpaxFront* and the fuselage diameter, *dfus*.


```
function lfus = fuselage_length_(Npax,NpaxFront,dfus)
//=====
// Model: geometry
lfus = 5 * dfus + 0.75 * Npax/NpaxFront + 1;
endfunction
```

3. Mass model example

This function allows to calculate the weight of the fuselage, Mfus, when knowing the length of the fuselage, lfus, and the fuselage diameter, dfus.

```
function Mfus = fuselage_mass_(dfus,lfus)
//=====
// Model: weight
sfus = %pi*dfus*lfus ;
Mfus = ( 0.02 * sfus -2.6443584 ) * sfus + 6283.8838 ;
endfunction
```

4. Statistical regression

wAfus is the fuselage wetted area. There is a simple relation between the fuselage wetted area and the fuselage diameter and length, which are obtained with linear regressions.

```
function [wAfus] = fus_wetted_area_#0(dfus,lfus)
//=====
wAfus = 2.7 * dfus * lfus + (7.1) ;
endfunction
```

All these functions, definition, regulation and models, can be assembled to compute performances data of table 1.2 at given flying conditions.

As an illustration of the complexity of this assembling, the figure 1.27 is an extract of the complete flow graph with the 100 basic functions. The main comment to make about this figure is that it is too much fuzzy to be able to handle data thanks to it.

1.3.2.2.6 Domain-level functions These functions are called to calculate performances. They wrap the call to a series of basic functions.

Here is an example that calculates the approach speed:

```
function [vapp] = approach_speed_(alt_app,LDW,aircraft_data)
//non_scalar:aircraft_data
//=====
global UF
//-----
[Aref,phi] = aircraft_data(['Aref','phi']) ;
```

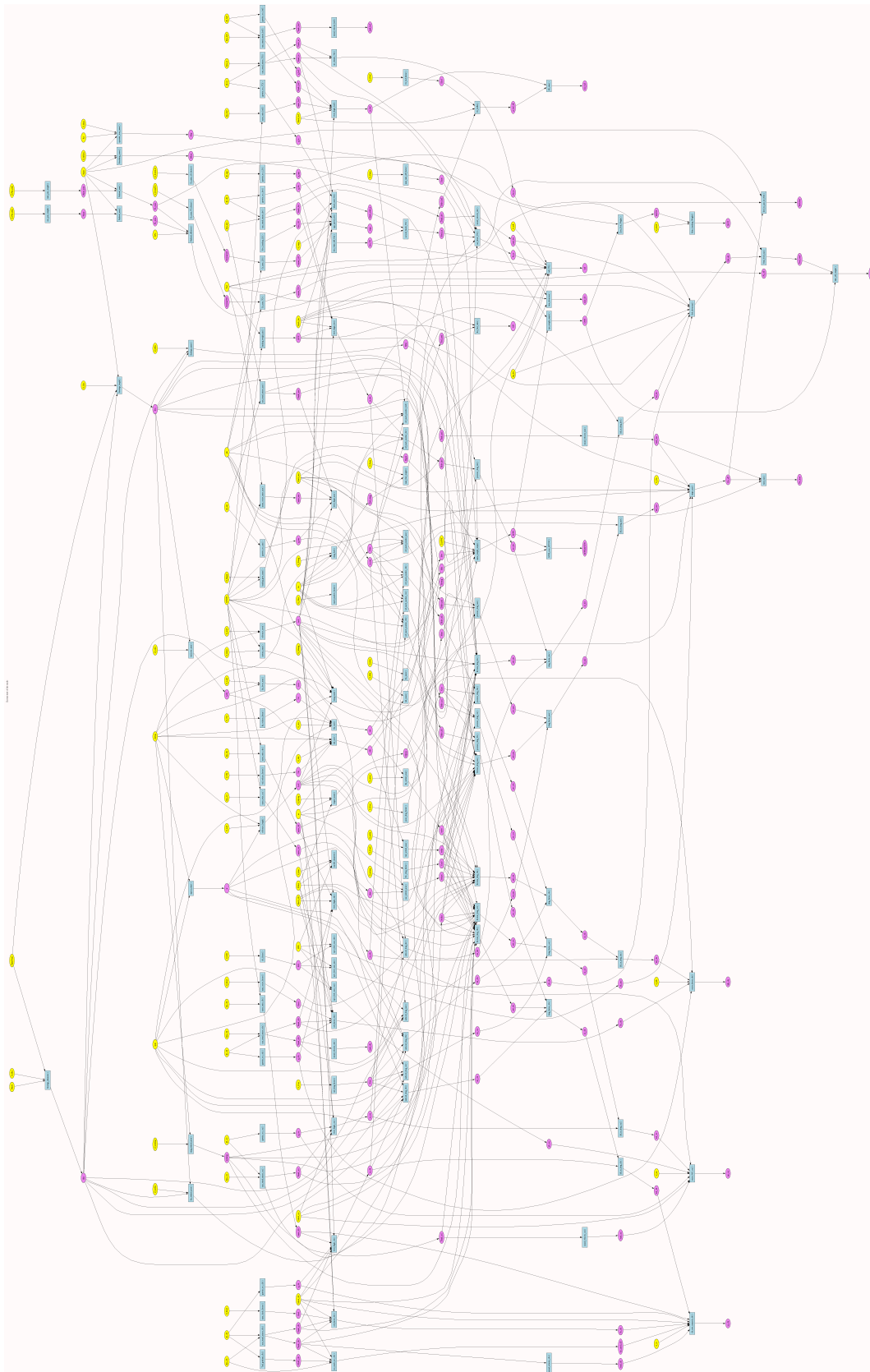


Figure 1.27: Complete flow graph with the 100 basic functions

```

kvs_LD = Kvs_Landing_() ; // Basic function
g = gravity_acc_(alt_app) ; // Basic function
[KczmaxLD] = Cz_max_LD_factor_(UF.KczmLD) ; // Basic function
[czmax_LD] = Cz_max_LD_(KczmaxLD,phi) ; // Basic function
vapp = app_speed_(LDW,czmax_LD,Aref,g,kvs_LD) ; // Basic function
endfunction

```

On the figure 1.27, instead of having six boxes to represent the basic functions used to calculate the approach speed, like `gravity_acc_` for instance, we would only have one box using domain-level functions (see on figure 1.28 page 51).

An exhaustive list of domain-level function signatures can be found in appendix A.2, page 224.

1.3.2.2.7 Solving functions for the Mass/Mission loop To evaluate the maximum take-off weight of an aircraft, we need to evaluate the weights of the aircraft components (wing, fuselage, landing gear, etc.). But these methods are estimation methods that need to know the maximum take-off weight.

On the other hand, to carry out the mission, the aircraft must take the necessary fuel, and the structure of the aircraft must be adapted to the fuel weight.

It appears that there is a system to solve to find the maximum take-off weight; this is a crucial point of the design.

Range constraint An aircraft is expected to perform a specified mission.

A nominal range, RA_{Nom} , is associated with a nominal payload, PL_{Nom} . This nominal payload corresponds to the maximum of passengers (and their luggages) in a typical configuration (80 kg/pax in short range, 90 kg/pax in medium range, 95 kg/pax in long range).

$$MTOW = ZFW_{Nom} + Fuel_{Nom}$$

where $Fuel_{Nom}$ is the necessary fuel quantity to perform the mission with the $MTOW$ and ZFW is the Zero Fuel Weight.

$$ZFW_{Nom} = OWE + PL_{Nom}$$

This means that

$$MTOW = OWE + PL_{Nom} + F_{Nom}$$

So, the OWE is

$$OWE = MTOW - PL_{Nom} - F_{Nom}$$

and

$$OWE = f(MTOW) \text{ for a given mission}$$

Weight estimate The structure weight, and then the *OWE* must be estimated to calculate the *MTOW*. The *OWE* depends on:

- *MTOW*: Maximum Take-Off Weight
- *MLW*: Maximum Landing Weight
- *MZFW*: Maximum Zero Fuel Weight
- *PL_{Max}*: Maximum Payload

PL_{Max} is linked to the fuselage volume that is a geometrical limitation, so it is a given value.

MLW is linked to *MZFW* ($MLW = k \cdot MZFW$, $k \sim 1.06$)

Thus,

$$OWE = g(MTOW, MLW, MZFW, PL_{Max}) = g(MTOW, MZFW)$$

Then, by definition,

$$MZFW = OWE + PL_{Max} = OWE(MTOW, MZFW) + PL_{Max}$$

For a given *MTOW*, we deduced that *MZFW* has to be adapted to get a *OWE* compatible with *PL_{Max}*. The following equation is thus deduced:

$$OWE = h(MTOW)$$

The resolution of this system is handled by the environment and thus, it has not been programmed in a dedicated function.

The figure 1.28 is a flow graph using domain-level functions as boxes to draw the graph, where the solving of the Mass/Mission loop is represented.

1.3.2.2.8 Study-level functions The study-level functions are the ones allowing the user to study a given aircraft configuration, or to perform an optimisation for instance. This level is directly implemented in a particular platform, *Odip*, *ODISA Integration Prototype* (see in [Mattei and Druot, 2005]), which is currently being developed, the aim is to create a new platform allowing some research on multidisciplinary problems, and then, to replace the platform of *AVION*.

An example of such functions is the `all_in_one` function, which performs a global analysis of the overall discipline equations.

```
function [Ar,Vht,Vvt,wAwing,wAht,wAvt,wAfus,wAnac, ...
         Mwing,Mht,Mvt,Mfus,Mgear,Mprop,Msys,Mfurn,Mop,Fuel_wing, ...
         Ksfc,Kmto,Kmcl,Kmcr,Kcx0,Kcxi,Kcxc,Kdiv,KczmTO,KczmLD, ...
         RA_eff,RA_time,LDW,Fuel_total,cz_mis,lod_mis,sfc_mis, ...
         Fuel_div,time_div,cz_div,lod_div,sfc_div,MTOW_eff, ...
         MLW,MZFW,PL_eff,PL_max,PL_nom,OWE,MWE,Fuel_max, ...
```

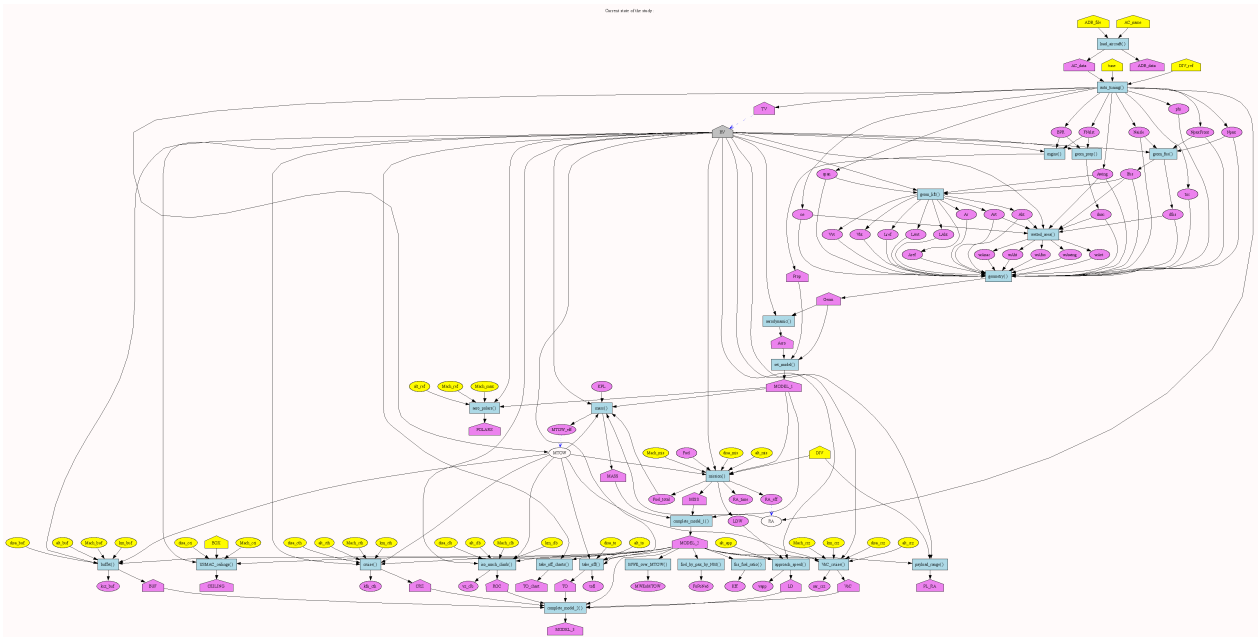


Figure 1.28: Flow graph using domain-level functions as boxes. Three main regions appear on this figure, the first one in the upper-right corner contains functions of the model construction, like the geometry, the engines and the aerodynamic models. The second region is in the middle of the figure, it concerns the Mass/Mission loop. The last region, the long line of functions in the lower side of the figure, contains analysis functions of the performances.

```

mass_clb,vz_clb,mass_cth,kfn_cth, ...
mass_crz,sar_crz,tofl,vapp] = all_in_one_(Npax,NpaxFront,...
Naisle,lfus,dfus,ne,BPR,FNslst,dnac,AWing,span,phi,tuc,Aref,...
Lref,LAht,Aht,LAVt,Avt,MTOW,Fuel,KPL,RA,disa_mis,alt_mis,...
Mach_mis,leg_div,alt_div,Mach_div,km_clb,disa_clb,alt_clb,...
Mach_clbkm_cth,disa_cth,alt_cth,Mach_cth,km_crz,disa_crz,...
alt_crz,Mach_crz,disa_to,alt_to,alt_app,RV)
    
```

The *Odip* platform also allows to draw flow graphics like the ones on figures 1.27, 1.28 and 1.29.

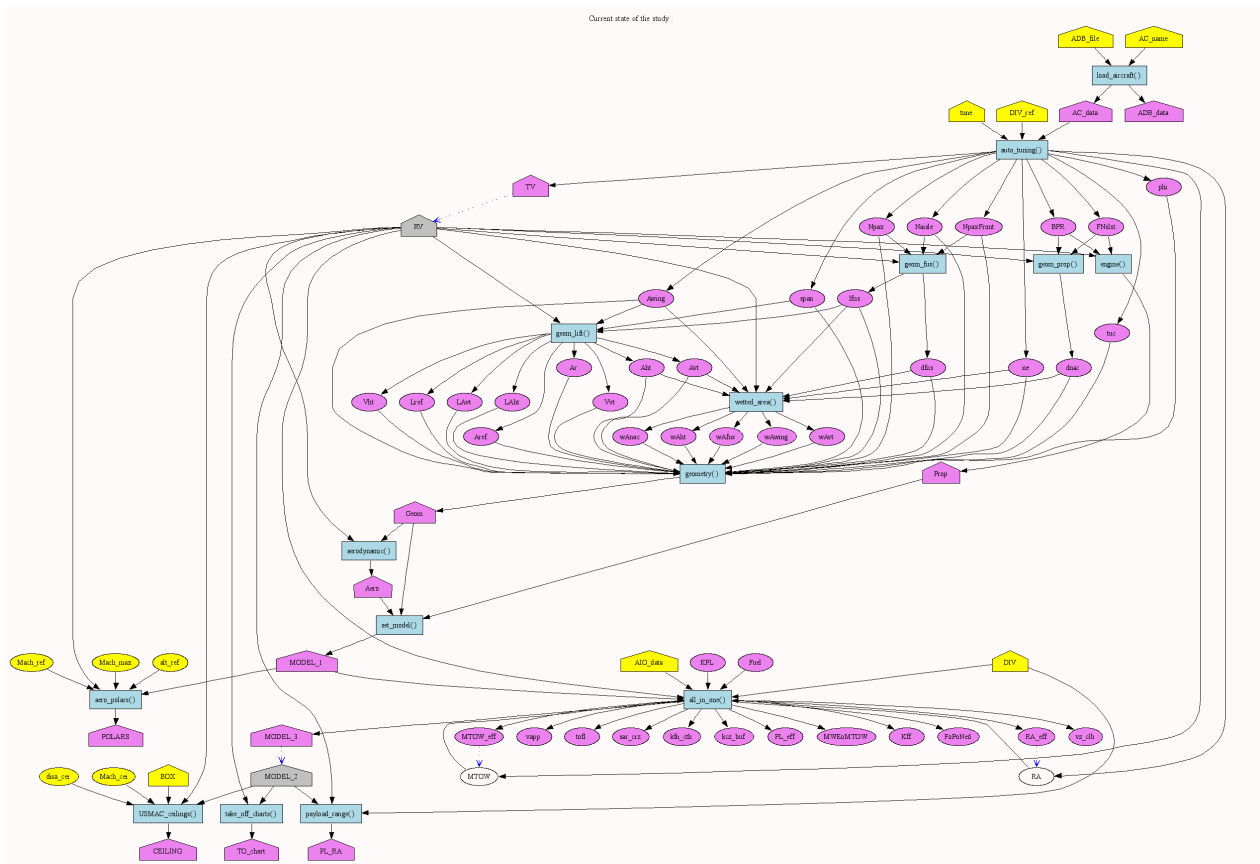


Figure 1.29: Flow graph at study-level. Again, two main regions appear on this figure, the first one in the upper-right contains functions of the model construction, with a detailed geometry. The second region is long line of functions in the lower side of the figure, containing the Mass/Mission loop and analysis functions of the performances.

The table 1.4 summarizes all the information concerning the different levels of function.

The third model of aircraft which was used during this study is the *SMAC*, the Simplified Model of AirCRAFT.

Categories	Basic	Domain	Study
Environment	6		
Geometry assessment	13	4	
Aerodynamic	14	1	
Engine	6	1	
Mass Estimation	23	1	
Performance estimation	22	9	
Configuration analysis			1
Mass-Mission loop			1
Generic solver			1
Optimisation			1
Design uncertainty solver			1

Table 1.4: USMAC package

1.3.2.3 SMAC (Simplified Model of AirCRAFT)

The *SMAC* has been developed to answer three main issues:

- the consistency between modules in the multidisciplinary design process,
- the maintenance cost,
- the lack of flexibility of the design process for the treatment of unconventional configurations.

Indeed, concerning the consistency problem, in the current FPO model, each discipline has its own representation of the aircraft geometry. The consequence is a lack of consistency between the discipline geometry representations, which may imply some difficulties to connect them. For instance, the aircraft geometry in the aerodynamic discipline only contains the external skin and shape of the aircraft, while the mass discipline needs an internal description of the aircraft components to assess weights.

The chosen solution was to change the overall structure of the model, by creating an independent geometry module, rich enough to contain all the information the disciplines need.

The *SMAC* is an object oriented model. As for the *USMAC*, it is coded, executed and launched in *Scilab* (see in [INRIA Copyright, 1989]). The core set of data is the geometry. According to [Sobieszczanski-Sobieski and Haftka, 1997], mathematical modeling of an aerospace vehicle critically depends on an efficient and flexible description of geometry, and concerning this model, an important part of the development work has been yield to define the geometry module.

In this structure, there are several levels of objects:

1. the first one is the global aircraft,

2. the second level contains objects representing functional part of the aircraft, like the propulsion or the control functions,
3. the third one is the aircraft description according to components, for instance the number of engines for the propulsion part.

Another characteristic of the *SMAC* is a built-in 3D model (see figure 1.30) to first and foremost model and represent the studied aircraft, and then, visualise the configuration that the designer is developing.

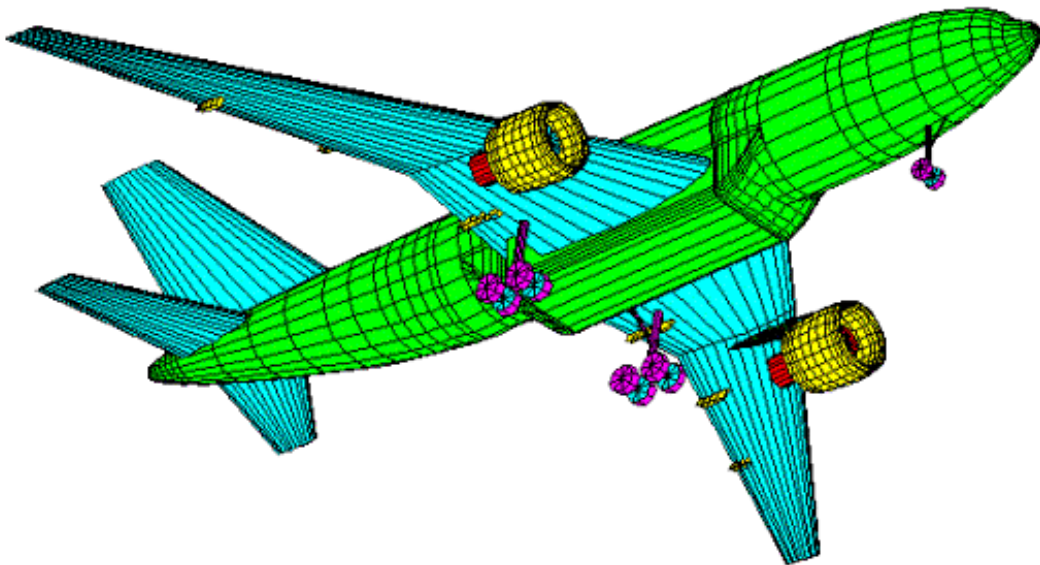


Figure 1.30: *SMAC* 3D model

The main difference between the *SMAC* and the *USMAC* is a higher complexity in the *SMAC* disciplines, which are the same than in *AVION* intrinsic model. And the main difference between the *SMAC* and the intrinsic model of *AVION* is the built-in 3D model in the architecture.

1.4 Global objectives of the thesis

The global objectives of the thesis are to:

1. **introduce new mathematical methods** that can be useful in a future project design tool and improve aircraft preliminary design,
2. **contribute in defining, testing and validating a new architecture of aircraft model** to perform multidisciplinary design based on a component cut-out approach.

1.4.1 Robust global optimisation

The main objective of the thesis is to improve global optimisation processes in current FPO tools. *AVION* offers the possibility to use commercial optimisation methods, like *Dot* (see description in [VR&D Copyright, 1995]), *FSQP* (see details in [Zhou et al., 1997]) or a simplex method (see in reference [Nelder and Mead, 1965]).

During the preparation of the thesis, **we decided to implement, develop and test evolutionary algorithms to enhance current global multidisciplinary optimisations** by introducing a more robust process. Indeed, before performing any optimisation, we want to identify a large amount of points satisfying constraints, and then to reduce the design space to the admissible space, which will improve the convergence robustness.

1.4.2 Multicriteria optimisation

Most of the time, the optimised criterion in future project studies is the DOC. But the problem is that there are as many DOCs as airlines. Moreover, requirements can evolve during the aircraft development. Thus, **we also performed and validated multicriteria optimisation, still using evolutionary algorithms.**

1.4.3 Response surfaces

Then, as the whole problem is heavy and very complex, **we decided to introduce a response surface to replace the evaluation function in the optimisation process to have an approximation of solutions in a shorter computation time.** Once the design point is close enough to the solutions, it is possible to have a better assessment by using back the initial evaluation function.

1.4.4 Uncertainty

Another part of the objectives of the thesis is **to introduce the notion of uncertainty in the numerical design process.** The aim is to assess the technical risks and to improve the way to take margins to ensure that the operational constraints are satisfied. Indeed, there is no ideal mean to define a robust configuration, thus, risk is assessed thanks to the sensitivity analysis manually performed through finite differences.

But a first task is to define precisely these notions of risk, robustness and uncertainty as they do not have been implemented in the current vocabulary of designers yet.

1.4.5 Help in decision making

Once we have implemented all the previously described mathematical methods, we have to introduce something to help the user in understanding results.

Chapter 2

Robust global optimisation

Contents

2.1	Multidisciplinary Design Optimisation (MDO)	61
2.1.1	Industrial needs	61
2.1.2	General introduction	61
2.1.3	Current MDO process in Future Project Office	72
2.2	Introduction to Genetic Algorithms (GAs)	75
2.2.1	Principle	75
2.2.2	Discussion	79
2.3	Ensuring constraint satisfaction	80
2.3.1	Problem description	81
2.3.2	Material	82
2.3.3	Indirect constraint handling	83
2.3.4	Constraint Satisfaction Problem (CSP) of aircraft sizing studies	84
2.3.5	Conclusion	92
2.4	Robust mono-criterion optimisation	92
2.4.1	USMAC implementation	93
2.4.2	Improvement of initial population repartition	96
2.4.3	Producing admissible points with USMAC	100
2.4.4	Robust mono-criterion optimisation	100
2.4.5	Going further	103
2.5	Conclusion	105

We explained in the previous chapter, in section 1.1 page 6, that aircraft sizing studies consist in determining characteristics of an aircraft, starting from a set of requirements. These studies can be summarized as global constrained optimisation problems with typically one thousand parameters. Actually, the constraints express physical feasibility and the requirements to be satisfied, and the objectives are market driven characteristics of the aircraft.

With this kind of problem, the first difficulty engineers are facing is to satisfy simultaneously all the constraints. Indeed, one of the most important part of the resolution is to reach the admissible domain before going on with further optimisations.

The objective of the work described in this chapter is to **improve the mono-criterion and constrained optimisation currently performed in FPO, by introducing a new structure to the problem, and new resolution methods coming from the Multidisciplinary Design Optimisation methodology.**

It appears to us that it could be interesting to uncouple the research of optimum solutions from the problem of admissible set extraction. Thus, to solve the aircraft sizing problem, our method consists in decomposing it to first focus on constraint satisfaction, and then, perform an optimisation without considering constraints anymore, starting from feasible points found in the first step.

In this chapter, we mainly consider the constraint satisfaction problem, further denoted CSP. **Our aim is to automatically produce large amounts of design points satisfying all the constraints, despite frequent evaluation failures.** Indeed, as the explored search space is vast, it contains meaningless design points (physically speaking), so the evaluation function we use fails typically 50% of times.

This CSP can potentially be solved in two ways:

1. gradient-based methods, which are fast but non-robust to evaluation failures,
2. stochastic methods, which are slow but allow for failure handling.

Since we wanted to favor robustness, we decided to implement and test a stochastic method. We showed that our dedicated implementation of genetic algorithms exhibits good results in terms of robustness and convergence speed. We compared it with a gold standard method in constraint satisfaction, FSQP [Lawrence and Tits, 1998] on the same constraint satisfaction problem.

Before introducing the method we implemented to solve this optimisation problem, we will introduce some general notions on the state-of-the-art methods developed to solve a multidisciplinary optimisation problem. Then, we will briefly describe the way it is currently solved in FPO. And finally, we will comment the results obtained with our genetic algorithm implementation for solving the CSP.

2.1 Multidisciplinary Design Optimisation (MDO)

2.1.1 Industrial needs

In the literature, several papers describe some industrial needs in Multidisciplinary Design Optimisation, further denoted MDO, particularly concerning aerospace design. A wide variety of industries are concerned with MDO processes, like airframe, automobile, rotorcraft, jet engine, space, dealing with problems such as feasible design, trade studies, structural sizing, sub-optimisation, dynamic response minimisation, and full configuration MDO.

The paper [Giesing and Barthelemy, 1998] presents a summary of ten papers dealing with industry design processes, experiences and needs, with emphasis on the needs of industry in the area of MDO. In particular, this paper summarizes the paper of [Wakayama and Kroo, 1998], which describes the optimisation of a detailed design of a flying wing, and illustrates the numerous challenges to MDO use in industry.

Many papers describe the needs in frameworks or environments allowing to perform MDO, like [MacMillin et al., 1995; Walsh et al., 2000a; Walsh et al., 2000b], or [Salas and Townsend, 1998], which makes a comparison of frameworks, like FIDO (Framework for Interdisciplinary Design Optimization), iSIGHT, LMS Optimus, DAKOTA (Design Analysis Kit for Optimization), etc.

According to [Bartholomew, 1998], although the software tools existing within individual disciplines may be reasonably mature, the challenge is now to provide the tools necessary to support such an integrated approach.

Another important need is about the problem formulation, *i.e.* posing the problem as a set of mathematical statements amenable to solutions, because it has a direct influence on methods and procedures for solving the problem once it has been posed.

Thus, after defining Multidisciplinary Design Optimisation, we will introduce some of the existing formulations of an MDO problem.

2.1.2 General introduction

The area of Multidisciplinary Design Optimisation has grown to the point of gaining near universal recognition in its ability to lead to “better” designs [AIAA White Paper, 1991].

Three definitions of MDO are given by the *AIAA MDO Technical Committee* (see [MDO Technical Committee, 2007]).

Definition 2.1. What is MDO?

1. A methodology for the design of complex engineering systems and subsystems that coherently exploits the synergism of mutually interacting phenomena.
2. Optimal design of complex engineering systems which requires analysis that accounts for interactions amongst the disciplines (or parts of the system) and which seeks to synergistically exploit these interactions.

3. How to decide what to change, and to what extent to change it, when everything influences everything else.

MDO in an aerospace context can be described as a methodology for the design of systems where the interaction between several disciplines must be considered, and where the designer is free to significantly affect the system performance in more than one discipline [Sobieszczanski-Sobieski and Haftka, 1997]. Thus, it embodies a set of methodologies, mathematical, numerical and organisational technics, which provide a mean of coordinating efforts and possibly conflicting recommendations of various disciplinary with well-established analytical tools and expertise.

Multidisciplinary design of aerospace systems is a computationally intensive process that combines discipline analyses with design space search and decision making. Optimal design of complex systems, more specifically aerospace systems, is increasingly becoming a geographically distributed activity, involving multiple decision teams and heterogeneous computing environments.

Thus, one of the aim of MDO is to meet the needs for increased interdisciplinary interaction and communication, and for reduced design cycle-time.

In engineering design problems, one attempts to improve or optimise several objectives, frequently competing and conflicting measures of the system performance, subject to satisfying a set of design and physical constraints (see in [Alexandrov and Lewis, 1999]). MDO enables the efficiency of designs to be optimised and supports trade-off studies between the design objectives of diverse disciplines (see in [Bartholomew, 1998]).

Currently, numerical optimisation is often applied sequentially, with certain parameters set by one discipline, and others assigned by the next discipline (see in [Kroo, 1995]). Generally, this approach does not lead to the optimal design of the complete system.

According to [Kroo, 1995], Prandtl solved a problem of wing design using an MDO procedure that treated structural sizing and aerodynamics concurrently. His solution yields 11% less drag at any selected structural weight than could be achieved using the sequential procedure (see in [Prandtl, 1933]).

According to [Bartholomew, 1998], MDO is seen as providing the means to avoid the fragmentation inherent in established methods which extends the time required for the design cycle and limits the efficiency of final designs. MDO permits the constraints of a diverse range of disciplines to be reflected from the earliest stages of the design process. This approach will facilitate the design of higher performance products with improved cost, structural integrity and maintainability. The methods will also offer the opportunity to maximise the exploitation of new materials technology within designs while minimising risk, and will have significant impact on project design times and cost.

The conceptual components of MDO are given in [Sobieszczanski-Sobieski and Haftka, 1997]:

- Mathematical modeling of a system,

- Design-oriented analysis,
- Approximation concepts,
- Optimisation procedures with approximations and decompositions,
- System sensitivity analysis,
- Human interface,
- Design space search.

Due to the extreme complexity of most MDO problems, [Cramer et al., 1994] believe it is necessary to focus on problem formulation methods and their interdependence with nonlinear programming algorithms, further denoted as NLP algorithms. According to [Alexandrov and Lewis, 1999], it is the nature of some of the MDO problem constraints that distinguishes the engineering design optimisation problem from the conventional NLP problem. The method of treating the problem constraints provides the defining characteristics for various approaches to solving MDO problems.

A conceptual or preliminary design problem can be formulated through an NLP formulation of the form:

$$\begin{aligned} \min \quad & f(X, Y, Z) \\ \text{s.t.} \quad & g(X, Y, Z) \leq 0 \end{aligned} \quad (2.1)$$

where:

- X is the vector of design variables,
- Y is the vector of shared variables between the disciplines,
- Z is the vector of discipline outputs.

Y is a set containing variables calculated by one discipline and needed as input by another discipline. This is the reason why it is distinguished from the set of design variables X . Some of the shared variables can be contained in the set of the system outputs, Z .

The figure 2.1 is a graphical representation of the different sets we have introduced, and their connections with the disciplines.

Thus, for one discipline D_i , we have the relation

$$\begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} = D_i(x_i, y_{i_{in}}) \quad (2.2)$$

where:

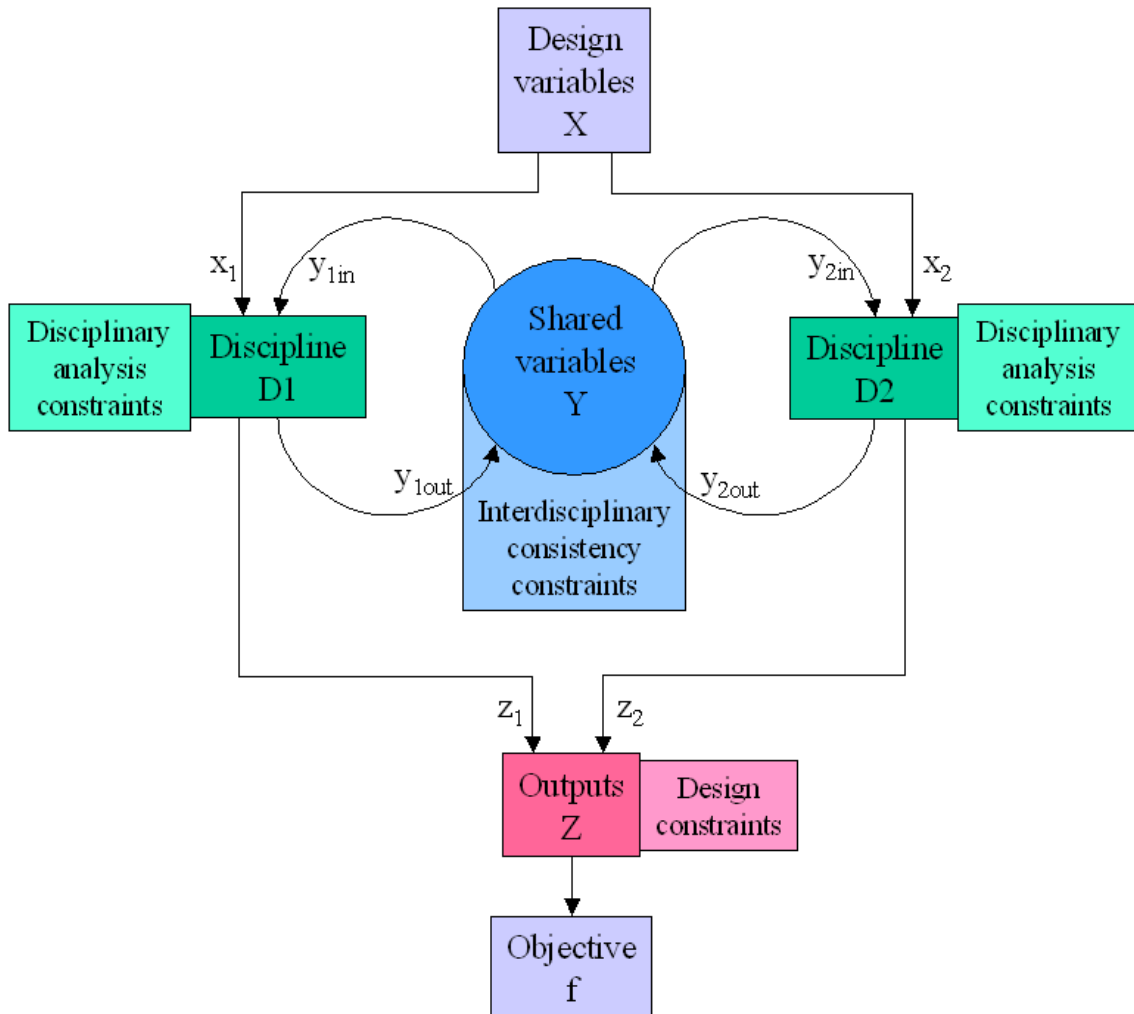


Figure 2.1: Illustration of MDO system: Y is a set containing variables calculated by one discipline and needed as input by another discipline.

- $x_i \subset X$ and $y_{i_{in}} \subset Y$ are the subsets of X and Y which contain the needed inputs for the discipline i ,
- $z_i \subset Z$ and $y_{i_{out}} \subset Y$ are the subsets of Z and Y which contain the outputs of the discipline i .

The classification of MDO problem formulation depends on the kind of feasibility that must be maintained at each optimisation iteration. In our context, we distinguish three kinds of constraints:

1. the **Disciplinary analysis constraints**, which are equality constraints implicit in disciplinary analyses,

$$\begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} - D_i(x_i, y_{i_{in}}) = 0 \quad (2.3)$$

2. the **Interdisciplinary consistency constraints**, which state for interdisciplinary couplings,

$$y_{ij} - y_{ji} = 0 \quad (2.4)$$

where y_{ij} are the interdisciplinary variables of the discipline i (inputs or outputs) which are common to the discipline j , and the values of y_{ij} are the ones considered for the discipline i .

3. the **Design constraints**, which are given by the requirements.

$$g(X, Y, Z) \leq 0 \quad (2.5)$$

Based on this constraint distinction, we can classify the MDO formulations, according to their treatment of these constraints. Following are examples of most common MDO formulations:

- the **Multidisciplinary Design Analysis**, further called MDA, ensures that the system of all equality and inequality equations, reaches an equilibrium state, all kinds of constraints must be satisfied, the optimisation is a distinct operator, which only has to manage the objective, and which is not a part of the MDA formulation,
- the **Multidisciplinary Feasible**, further called MDF, manages the disciplinary analysis and the interdisciplinary consistency constraints during the analysis, and the design constraints are handled by the optimiser,
- the **Individual Discipline Feasible**, further called IDF, handles the disciplinary analysis constraints during the analysis, and the interdisciplinary consistency constraints are added to the design constraints as optimisation constraints,

- The **Collaborative Optimisation**, further called CO, treats the disciplinary analysis and the design constraints during the analysis, and the interdisciplinary consistency constraints are handled by the optimiser,
- the **All-At-Once**, further called AAO, considers that all of the analysis variables are optimisation variables, and all of the analysis discipline equations are optimisation constraints.

The MDO problem formulation has a great impact on the algorithm required to solve it [Alexandrov and Lewis, 2000].

We now give more details about each of these MDO formulations, which are fully described in [Alexandrov and Lewis, 1999; Cramer et al., 1994; Dennis and Lewis, 1994; Masmoudi and Auroux, 2005]

2.1.2.1 The Multidisciplinary Design Analysis (MDA)

The MDA treats the way to link shared variables coming from different disciplines to be able to solve a global system including all these discipline equations. As an illustration, the following is a simple example using equations balancing the weight of an aircraft with its lift (see in [Buckley et al., 1992]):

$$W_S = q C_L \quad (2.6)$$

$$q = \frac{1}{2} \rho v^2 \quad (2.7)$$

$$\rho = 1.225 \quad (2.8)$$

These equations relate five parameters:

- W_S is the wing loading, in N.m^{-2} ,
- q is the dynamic pressure, in Pa,
- C_L is the lift coefficient, non-dimensional variable,
- v is the velocity, in m.s^{-1} ,
- ρ is the air density at sea level, in kg.m^{-3} .

To solve this system of equations, we have to make choices about which equations will enable us to evaluate which variables. We have no choice for the equation (2.8), it directly determines the air density ρ . Thus, we are left with four variables and two equations. This implies that we need to specify two independent variables and let the system of equations determine the other two. With this example, we have five possibilities of choosing independent variables (see in figure 2.2).

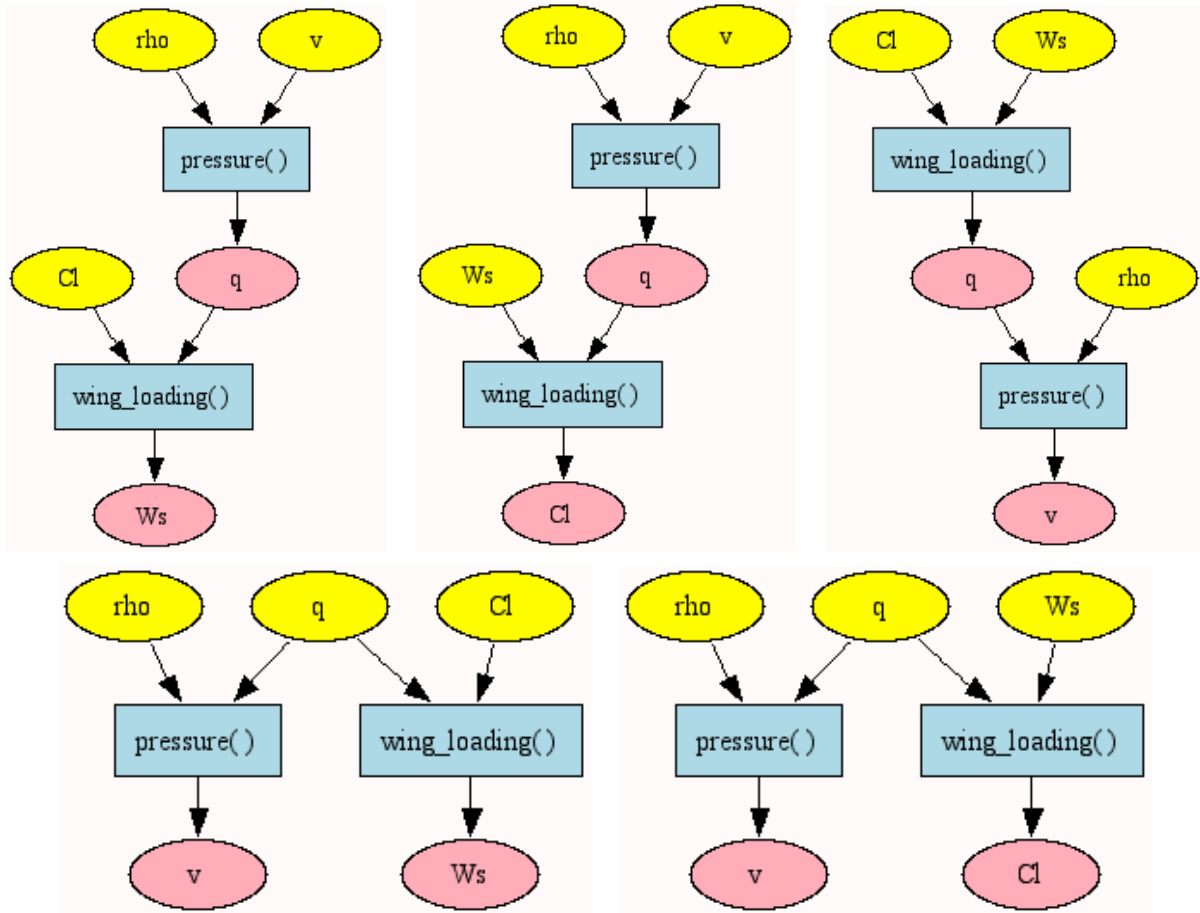


Figure 2.2: Illustration of permutations of independent variables corresponding to the equation system (2.6, 2.7, 2.8). The variables in yellow are the independent variables of the system. The variable ρ is determined by the equation 2.8, thus it is always an input of the two left equations.

This aspect of the problem, system analysis and resolution, is not a part of this work because it is directly managed in the Future Project Office tools thanks to bipartite graphs (see in [Druot, 1994]).

When dealing with large-scale systems, the analysis is not so straightforward because of the amount of variables shared by different disciplines. In the previous example, only one

variable was common to two equations.

MDA can be very costly because of the expense incurred by an MDA algorithm in repeatedly executing the analysis code. One way to avoid some of this cost is not to require feasibility until convergence to optimality. However, there will be approaches in which we will require partial feasibility for some good reasons. We are now going to describe other formulations to treat an MDO problem, which vary according to the way they handle constraints.

2.1.2.2 The Multidisciplinary Feasible (MDF)

This formulation is the most common way of posing an MDO problem, a complete multidisciplinary feasibility is required.

The vector of design variables, X , is provided by the optimiser to the coupled system of disciplines. A complete analysis is performed to solve the entire system, to obtain the shared variable Y and output Z values. Then, the objective function and the design constraints values can be evaluated and given back to the optimiser:

$$\begin{aligned}
 & \text{minimise} && f(X, Y, Z) \\
 & \text{with respect to} && X \\
 & \text{subject to} && g(X, Y, Z) \leq 0 \\
 & \text{where} && \begin{cases} \begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} = D_i(x_i, y_{i_{in}}), \forall i \in 1, \dots, N \\ y_{ij} = y_{ji}, \forall i, j \in 1, \dots, N, i \neq j \end{cases}
 \end{aligned} \tag{2.9}$$

More details can be found in [Cramer et al., 1994].

2.1.2.3 The Individual Discipline Feasible (IDF)

One way to avoid a complete analysis of the coupled system of disciplines is to use a formulation like the IDF formulation, where a specific decomposition of the work between analysis and optimisation is done.

IDF maintains individual discipline feasibility while allowing the optimiser to drive the individual disciplines toward interdisciplinary feasibility and optimality. In this approach, the shared variables Y become degrees of freedom of the optimiser:

$$\begin{aligned}
 & \text{minimise} && f(X, Y, Z) \\
 & \text{with respect to} && X, Y \\
 & \text{subject to} && \begin{cases} g(X, Y, Z) \leq 0 \\ y_{ij} - y_{ji} = 0, \forall i, j \in 1, \dots, N, i \neq j \end{cases} \\
 & \text{where} && \begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} = D_i(x_i, y_{i_{in}}), \forall i \in 1, \dots, N
 \end{aligned} \tag{2.10}$$

More details can be found in [Cramer et al., 1994].

2.1.2.4 The Collaborative Optimisation (CO)

Another way to decompose the problem is developed in Collaborative Optimisation formulation. It consists of a bi-level optimisation architecture in which individual disciplinary is charged with satisfying local constraints with the use of local optimisers on its own set of local design variables. The goal of each local optimiser is to agree with the other groups on values of the interdisciplinary variables, while a system-level optimiser provides coordination and minimises the overall objective:

$$\begin{aligned}
 & \text{minimise} && f(X, Y, Z) \\
 & \text{with respect to} && X, Y \\
 & \text{subject to} && y_{ij} - y_{ji} = 0, \forall i, j \in 1, \dots, N, i \neq j \\
 \\
 & \text{where} && \begin{cases} g(X, Y, Z) \leq 0 \\ \begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} = D_i(x_i, y_{i_{in}}), \forall i \in 1, \dots, N \end{cases}
 \end{aligned} \tag{2.11}$$

More details and examples can be found in [Kroo et al., 1994; Braun and Kroo, 1995; Kroo and Manning, 2000].

2.1.2.5 The All-At-Once (AAO)

In this formulation, we do not seek to obtain feasibility for the analysis problem for any kind of constraint until optimisation convergence is reached. We spend no time to achieve feasibility when we are far from an optimum. Moreover, the disciplines are treated explicitly as equality constraints:

$$\begin{aligned}
 & \text{minimise} && f(X, Y, Z) \\
 & \text{with respect to} && X, Y, Z \\
 & \text{subject to} && \begin{cases} g(X, Y, Z) \leq 0 \\ y_{ij} - y_{ji} = 0, \forall i, j \in 1, \dots, N, i \neq j \\ \begin{pmatrix} y_{i_{out}} \\ z_i \end{pmatrix} - D_i(x_i, y_{i_{in}}) = 0, \forall i \in 1, \dots, N \end{cases}
 \end{aligned} \tag{2.12}$$

More details can be found in [Cramer et al., 1994; Alexandrov, 1995].

2.1.2.6 The multi-level process

The multi-level process for aircraft design can be represented as a three level process [Coleman et al., 2006]:

1. the **future project level**: the optimisation is conducted at future project level to identify roughly the optimum configurations associated with the requirements. This can include the global topology (e.g. engines on fuselage or on wing) or values for continuous parameters (e.g. wing sweep angle).
2. the **discipline level**: the optimisation goes further in the design process by conducting systematic single discipline optimisation with heavier analysis (Computational Fluid Dynamics, CFD, and Computational Structural Mechanics, CSM) for these configurations. They are now all optimised with respect to lower level design variables and the optimisation results are used to produce surrogate models or response surfaces that can be further re-introduced in the FPO code (or similar formulations) for solving a MDO problem. This time the MDO problem is not solved based on semi-empirical methods but based on optimum results found with advanced numerical analysis.
3. the **component level**: it concerned the detailed design of the aircraft (refined aerodynamic shape, structural sizing). The same kind of process than in the second step, can be applied here.

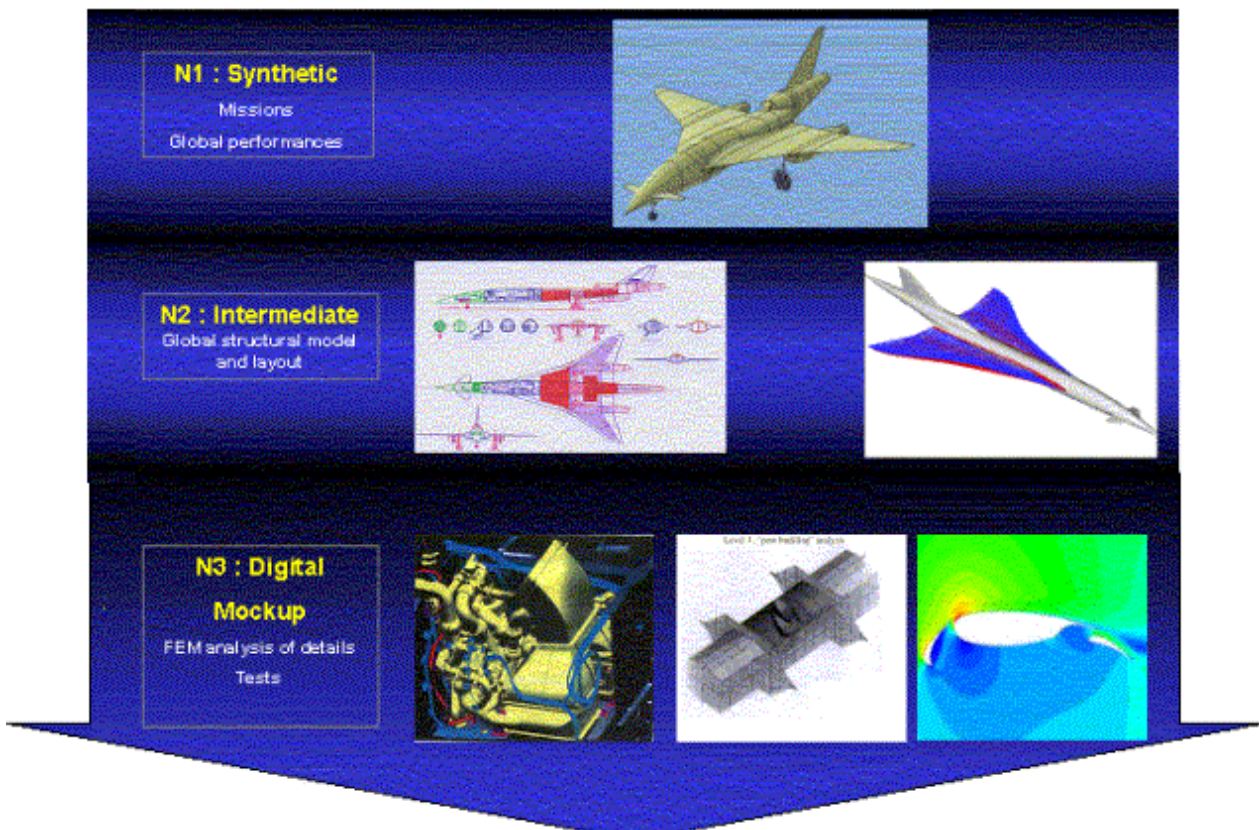


Figure 2.3: Illustration of the multiple levels of a project

Multi-level description introduces a hierarchy of variables with a growing level of detail and also a hierarchy of methods of growing fidelity and complexity.

2.1.2.7 The multi-physics analysis

Multi-physics analysis is another emerging method to treat design problems. These approaches propose to integrate the disciplines at analysis level, for example mixing CFD and CSM in a single mesh for aeroelastic studies based on Navier-Stokes.

Multidisciplinary optimisation is more intended to address optimisation based on the today design process and discipline methods and tools. Multi-physics is probably the future, but it is changing the process and the link with the company experience through the history, state of the art, experimental data. It is true that for totally innovating configurations like a flying wing, these methods can be very useful. So it is probable that these approaches have to be worked out in parallel with integration of more standard approaches [Coleman et al., 2006].

[Wakayama and Kroo, 1998] pointed out that currently some of the more successful approaches use close-coupled, all-at-once procedures, however, their success depends, in part, on the fact that automated, fast-running analysis codes are used.

After this short introduction to some of the different existing formulations of an MDO problem, we now describe the way this problem is currently managed in FPO.

2.1.3 Current MDO process in Future Project Office

The general problem of aircraft sizing in FPO can be summarized as an optimisation problem with typically:

- 15 degrees of freedom, like the engine size, the wing area, etc.
- 20 inequality constraints, like on the take-off field length, etc.
- and generally one criterion, like the MTOW or a DOC.

An example of a typical problem of aircraft sizing is described in appendix, the degrees of freedom in appendix B.1, page 227, the constraints in appendix B.2, page 227 and the criterion in appendix B.3, page 227. This problem is the basic one that we want to solve during this work.

This general problem is not currently treated as an NLP problem, its generic formulation. Indeed, the discipline equations and interconnections make it so complex that it is not possible for classical optimisation algorithms to treat it directly numerically. Moreover, optimisation is often used within these disciplines, because they need the satisfaction of some local constraints.

However, combining the computational tasks into a single optimisation problem is impractical, because of the large number of variables and the amount of time required to execute the analyses.

Furthermore, even if this would be possible, it would not be used by engineers, because:

- they are cautious with the results produced by the optimisers; since the evaluation function is not robust and sometimes fails, it makes the optimisers fail too,
- they need some more information than just finding an optimum, like knowing the results of sensitivity analysis around the optimum point on the objectives and the design variables.

Thus, optimisation in FPO is currently manually conducted by engineers, based on experience and knowledge acquired during previous studies.

The MDO formulation of the problem in the FPO design tool has been formulated as an intermediary formulation between MDF and CO, formulations described previously. Indeed, the tool is able to perform an entire analysis on the discipline constraints and the interdisciplinary consistency constraints, and it performs also some local optimisation inside disciplines, like for the calculation of the take-off field length.

The FPO tool helps also engineers to find design points satisfying constraints. With its ability to transform the problem by choosing freely the status of variables, fixed, dependent or free, the design constraint values can be imposed to the process, which solves the inverse problem to find admissible design variable values.

Once a design point satisfying TLARs is found, the design study can continue using a direct resolution of the aircraft sizing system. Most design studies rely on sequential parametric trade studies in which one or two, sometimes three, but rarely more, design variables are changed to examine the effect on the design. Carpet plots (like on figure 2.4) can be presented, showing the effect of these variables on each of the system constraints or objectives.

On the figure 2.4, two degrees of freedom are considered, the engine size and the wing area. The MTOW, the approach speed and the TOFL are calculated for several values of these degrees of freedom. The limit value of the constraints, here the approach speed and the TOFL, are interpolated to be represented on the graph. Then, the minimum value of the criterion, here the MTOW, is found graphically, and the corresponding values of the degrees of freedom are deduced.

The number of parameters in this type of study is limited by the dimensionality that can be perceived graphically or by the complexity to collect all information coming from such samplings when there are more than two parameters. When the number of parameters to be optimised is too large such that trade studies on the interesting variables are not possible, some kind of alternated optimisation is performed. Values of some important parameters are manually changed and then, trade studies are performed again on the modified designs.

This kind of trade studies is not performed so frequently, because of the preparation work that must be done before each sampling. Indeed, some *a priori* choices are made on which trade variables to select, and some evolution rules are imposed to some other degrees of freedom of the study, which are second order parameters. Because it is not possible to make

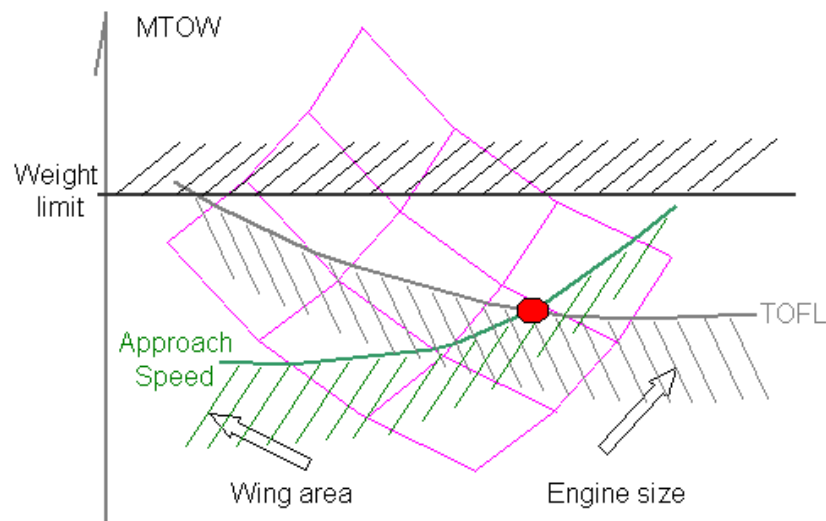


Figure 2.4: Example of carpet plot with two degrees of freedom, the engine size and the wing area, and two constraints, the approach speed and the TOFL, which delimits the admissible domain. The aim of the process described here is to minimise the MTOW.

vary all the design parameters, the research cannot be exhaustive, which limits the generality of the results.

Sometimes, these trade studies are replaced by inverse resolutions because, most of the time, optimum solutions are located at the intersection of some constraints. Thus, thanks to this knowledge engineers have on this problem, they do not spend time performing trade studies, but directly calculate the configuration which is of their interest. For instance, instead of minimising the MTOW according to the wing area, with a constraint on the approach speed, like on the figure 2.4, they fix the approach speed at its limit value, and they deduce the wing area.

At a preliminary stage, the sampled variables are the engine size and the wing area, because it is known that these variables are the most important driving parameters of a configuration. The other design variables are then fixed or linked to the scanning parameters through some rules, imposed by the designer. For instance, the empennage area and leverarm depends on the wing area, or the landing gear size depends directly on the engine size, at given engine ground clearance.

The most important and difficult part of this work is to define the evolution rules, to make them the most relevant and representative possible. This is the reason why such scannings are not performed frequently.

This process progresses step by step, each configuration satisfying some particular requirements is kept as a reference configuration, and then, some modifications are applied to it, according to new informations on technologies, or on customer needs. This leads to new trade studies, until a new reference configuration is found, and so on.

Convergence of the overall process has not a real sense for this kind of study, because there are always new technologies coming from research on the current aircraft program, that can be added to the current future project of aircraft. Thus, it is to the decision-maker to stop the study when he considers it is mature enough to begin the development phase.

Concerning multilevel, there are generally two levels of modelisation in FPO, but they are rarely represented together in the design environment, mainly because these models do not share the same geometrical representation of the aircraft.

Models with different grained precisions are not mixed in an integrated model of aircraft. It can be technically possible, but the lack of information on the uncertainty related to these models makes the engineers be not confident in the results obtained by coarse-grained models.

The general thinking is that coarsed-grained models may produce erroneous results, but they can have a good representation of the variations of the model according to the variable modifications, and sensitivity analyses can give good results using these models.

Thus, when multilevel formulation is used in FPO, it is done with a lot of precautions. A really simplified model of aircraft is used to calculate the main parameters of the configuration, and these values are given to a more sophisticated model of aircraft, but the transition is made with caution.

2.2 Introduction to Genetic Algorithms (GAs)

Genetic Algorithms, further denoted GA, are stochastic search algorithms based on principles of natural selection and recombination. They are one part of Evolutionary Algorithms, further denoted EAs.

EAs indicate a subset of Evolutionary Computation, which is a part of Artificial Intelligence [Schwefel, 1981]. It is a generic term used to indicate any population-based metaheuristic optimisation algorithm that uses mechanisms inspired by biological evolution, such as reproduction, mutation, recombination, natural selection and survival of the fittest, or the luckiest [Roudenko, 2004]. Randomness has an important role in these algorithms. Candidate solutions to the optimisation problem play the role of individuals in a population, and the cost function determines the environment within which the solutions "live". Evolution of the population then takes place after the repeated application of the above operators.

EAs contains also [Roudenko, 2004]:

- Evolutionary Programming,
- Evolution Strategy (see in [Bäck et al., 1991; Rechenberg, 1973]),
- Genetic Programming (see in [Koza, 1992]).

They are used in diverse fields such as engineering, biology, economics, genetics, operations research, robotics, and others.

There are others metaheuristics that can solve these problems. As examples, we have [Dréo et al., 2003]:

- Particle swarm optimisation, based on the ideas of animal flocking behaviour. Also primarily suited for numerical optimisation problems.
- Ant colony optimisation, based on the ideas of ant foraging by pheromone communication to form path. Primarily suited for combinatorial optimisation problems.
- Taboo search, a single-point metaheuristic, described page 137 in a multiobjectif context.
- Simulated annealing, also a single-point metaheuristic, described in paragraph 3.1.4.2 page 130 in a multiobjectif context.

2.2.1 Principle

Genetic Algorithms are inspired by the natural selection principle, as elaborated by Charles Darwin. They were created by J. Holland [Holland, 1975], and used by D. Goldberg [Goldberg, 1989] as the optimisation algorithm we currently know. Specific vocabulary is directly coming from evolution and genetic theories. Following is a description of this vocabulary, before explaining the way GA work.

We call (see figure 2.5):

- a **population** a set of points inside the design space,
- an **individual** is one point of the population,
- a **genotype**, or a **chromosome**, is another way of speaking about an individual, taken as a set of values,
- a **gene** is what composes a chromosome, it is one degree of freedom of the optimisation process,
- a **phenotype** is the result of the evaluation of one genotype, it contains the performance, the fitness or the properties of the individual.

Each individual is coded as a set of genes. In most implementation of genetic algorithms, the individuals are composed of a binary string, a gene can only be 0 or 1. Thus one degree of freedom is represented by a group of genes. But a degree of freedom can be either a discrete or a continuous variable.

As we are dealing with discrete and continuous variables, in our formulation, one gene represents one degree of freedom, and can take all possible values inside the definition space of the variable it encodes.

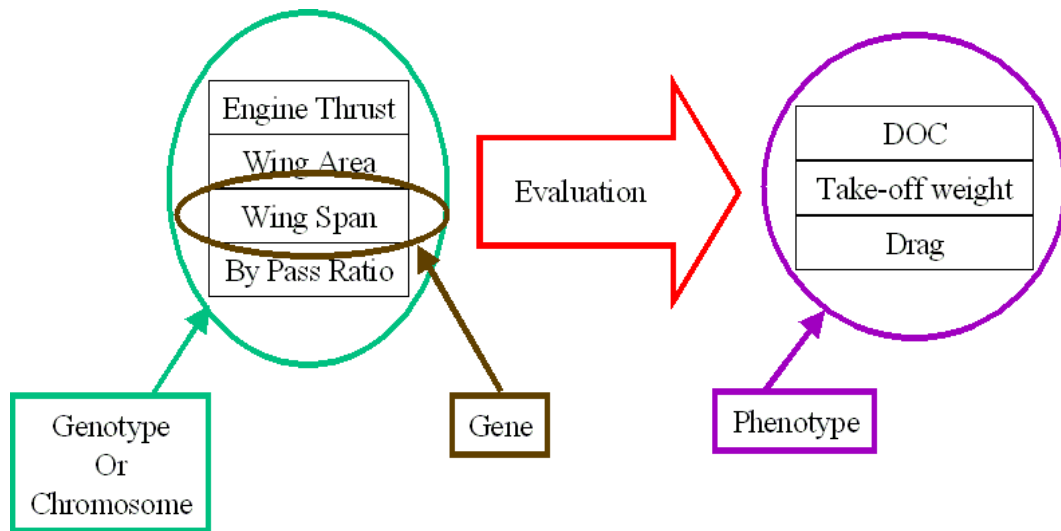


Figure 2.5: Illustration of Genetic Algorithm vocabulary

2.2.1.1 Initialisation

The first step of the algorithm is the definition of the initial population. There are mainly two ways to create the initial population:

- points are randomly sampled inside the design space,
- points are well-distributed inside the design space, using methods like Sobol, Cross-Validation or Latin Hypercube [Poles et al., 2006].

Generally, well-distributed samplings have best performances on the final convergence of the algorithm [Poles et al., 2006].

The size of the initial population is an important point, and it is usually chosen to ensure a compromise between calculation time and quality of the final results [Gao, 2003].

2.2.1.2 The fitness function

An efficiency is associated with each individual. This efficiency corresponds to the ability of the individual to answer the problem. It is evaluated through the fitness function. For instance, in a minimisation problem, the efficiency of an individual will increase with its capacity to minimise the objective function. Generally, the fitness function is proportional to the objective function of the optimisation problem.

Then, three genetic operators are applied to each individual, in different ways, depending on the GA that you consider.

2.2.1.3 The reproduction, or cross-over operator

The reproduction is applied on a pair of individuals. It consists in mixing the genes of these two individuals to produce new points, called the children (see in figure 2.6).

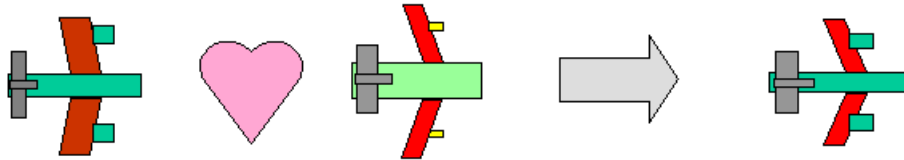


Figure 2.6: Illustration of reproduction in aircraft sizing studies

The way to select the individuals in the reproduction process depends on the algorithm one runs. If the algorithm is eugenicist, the individuals are chosen amongst the best points, according to the fitness function. In other kinds of algorithms, the individuals are chosen randomly, or you can create two sub-populations containing for the first, best individuals, and for the second, points chosen randomly. There are other ways in between to select the individuals for the reproduction, it depends on the choice to favor the convergence speed or the globality of the optimum solution (vs locality). A comparison of the performance of four commonly used selection schemes can be found in [Goldberg and Deb, 1991].

The figure 2.7 is an illustration of one way of individual selections for the reproduction operator.

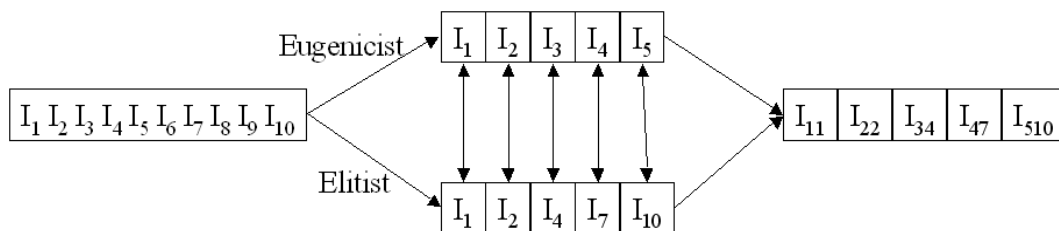


Figure 2.7: Illustration of individual selection

The cross-over operator favors the exploration of the design space by creating new points to evaluate, the children, which are most of the time different from their parents.

2.2.1.4 The mutation operator

The mutation consists in changing randomly the value of one gene of an individual according to a given probability. Generally, the probability for an individual to mutate is very small, and if it happens, a new value for the concerned gene is randomly sampled inside the range of definition of the corresponding variable (see in figure 2.8).

The mutation operator ensures as well as the reproduction a global search inside the design space. It helps finding the global optimum. Moreover, if the population size is too small, the Mutation operator prevent from the homogenization of the population.

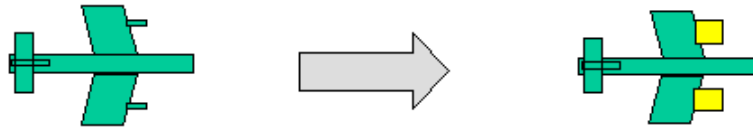


Figure 2.8: Illustration of mutation in aircraft sizing studies

2.2.1.5 The selection operator

After the reproduction, the new population size is larger than the initial size. In most Genetic Algorithms, the population size has to remain globally constant. From one iteration to another, the selection operator enables to get back to the initial population size, and the selected individuals are the best ones amongst the parents and the children, according to the fitness function (see in figure 2.9).

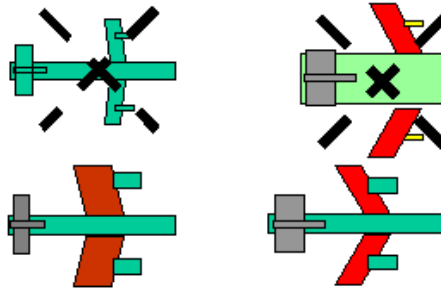


Figure 2.9: Illustration of selection in aircraft sizing studies

As for the selection of individuals in the reproduction operator, there are several ways to select individuals for the next generation of the algorithm, the design points can be selected according to a compromise of optimality and variability defined by the user.

2.2.1.6 Summary

There are as many Genetic Algorithms as users of such algorithms, as you can:

- combine the operators in different orders,
- tune the operators as you want, depending on your problem, like for instance the probability for one gene to mutate,
- fix the population size to have a good influence whether the algorithm can find good solutions in an acceptable computational time, which strongly depends on the problem,
- adapt your selection operator to favor convergence speed on one hand, or space search on the other hand, to find the global optimum.

In figure 2.10 is an example of a typical genetic algorithm.

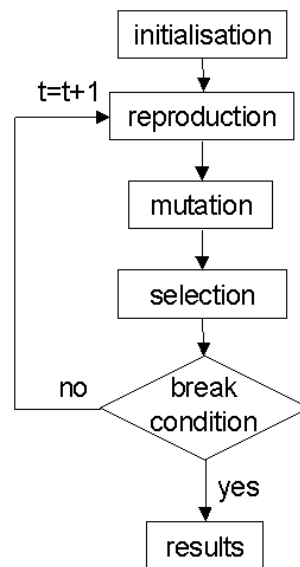


Figure 2.10: Flow-chart of a typical genetic algorithm

2.2.2 Discussion

Genetic Algorithms do not need *a priori* knowledge on the problem. The main part of the work is to formalise the data as chromosomes to be treated by the algorithm.

Calculation times do not increase with the number of variables but with the size of the population, and strongly depend on the evaluation function, for instance on its shape, if it is a convex function or not, etc.

This kind of algorithms can easily be parallelised, which promises a substantial gain in performance. When there is an increase in the time required to find adequate solutions, one of the most promising choice to make GA faster is to use parallel implementations.

The most popular parallel GA consists in multiple populations that evolve separately most of the time and exchange individuals occasionally. The paper of [Cantú-Paz, 1997] is a survey of the most representative publications on parallel genetic algorithms.

An important part of a genetic algorithm implementation is to choose the right values of the control parameters, like the mutation rate, to find a good compromise between optimality and variability according to the studied problem. This choice appears also when defining the population size or the way to select individuals, for the reproduction or the next generation.

Most of the time, these tuning parameters are handled as global, external parameters, which remain constant over time. Other approaches, described in [Bäck, 1992; Beyer and Deb, 2001] consist in environment-dependent self-adaptation of appropriate settings for the mutation distributions or the recombination rates.

Thus, the convergence of the algorithm, and the time it will spend to find optimum solution depends on the choice of the control parameters. Some demonstrations on GA convergence properties exist, like in [Srivastava and Goldberg, 2001], but the hypotheses

taken in such papers are most of the time impossible to verify in an industrial context like ours.

Constraint handling is not so straightforward in evolutionary algorithms because the search operators mutation and recombination are “blind” to constraints. Hence, there is no guarantee that if the parents satisfy some constraints, children will satisfy them as well [Eiben, 2001; Michalewicz, 1992]. No standard evolutionary algorithm takes constraints into account, they perform unconstrained search.

As we want to use Genetic Algorithms to solve our problem of aircraft sizing, we have to find a way to handle constraints. There are many ways to overcome this question, like indirect constraint handling, where constraints are incorporated in the fitness function, or direct handling, where constraints are left as they are and the GA is adapted to enforce them. Thus the following section is focused on the way we chose to handle constraints in our implementation of Genetic Algorithms.

2.3 Ensuring constraint satisfaction

Constraint satisfaction in our initial aircraft sizing problem (described in appendix B, page 227) proved to be hard to ensure. To explain the reasons of such a difficulty, we now describe the mathematical characteristics of the problem of aircraft sizing, and the material used to perform it, with a description of the architecture of the evaluation function and its numerical and computational aspects.

Then, we will detail the methodology we decided to test, and the main reason why we decided to uncouple the Constraint Satisfaction Problem, CSP, to the optimisation problem.

After that, we will present the results we obtained with our specific implementation of a Genetic Algorithm, and we will compare them with the results obtained with FSQP on the same problem. All these results have been presented in [Baduffe et al., 2005].

2.3.1 Problem description

Aircraft Sizing activity is basically an inverse problem. We know the performance that the system should achieve and we look for its physical characteristics. The studied system is an aircraft evolving in its environment. It is complex enough to make it impossible to express directly characteristics as a function of performance. For this reason, the core of Aircraft Sizing is necessarily an iterative process driving an evaluation function that quantifies the performance of a given aircraft configuration.

The design space is determined by validity intervals on the design parameters and the property space is determined by the set of constraints. The domain we call the admissible set is the antecedent of the intersection between the image of design space and the property space (see in figure 2.11).

The necessity to find a simple answer a complex problem has led many engineers to aggregate their evaluation function into a constrained mono-criterion optimisation process. The main drawback of this approach is that it gives a limited view on compromise solutions, since it is obvious that the final product is never a mathematical optimum but an *alchemic* compromise.

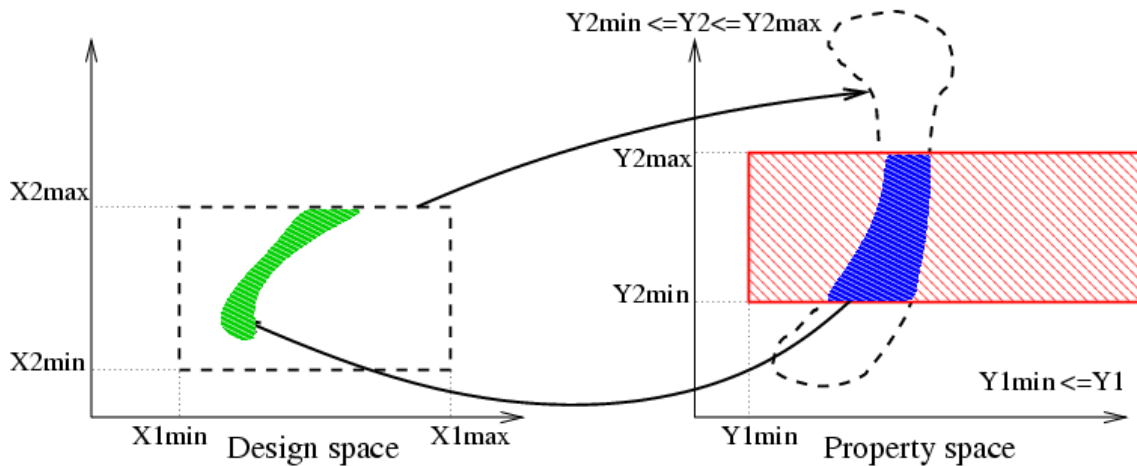


Figure 2.11: Admissible set representation: the admissible set (in green) is the antecedent of the intersection (in blue) between the image of design space (dashed domain on the right) and the admissible property space (in red)

The multi-criteria approach is an answer to the research of compromise, but requirements and performance formulations may change, or at least evolve, along the study duration. Finally, it appears to us that it could be interesting to uncouple the problem of research of compromise solutions from the problem of ensuring constraint satisfaction.

Thus in the following sections, we will focus on ensuring constraint satisfaction. We will come back to the optimisation in section 2.4, page 92.

2.3.2 Material

2.3.2.1 Architecture of the evaluation function

To perform aircraft sizing, *i.e.* to assess the main characteristic parameters defining an aircraft, we use a complex and dense Fortran code, coming from the current tool in FPO: *AVION*. This code is described in the previous chapter, section 1.3.2.1, page 43. It is composed of a set of modules, each one specific to a particular physical domain or to an aircraft element, for instance aerodynamics, structure, performance, nominal mission, or engines. The equations from Physics are formulated in their natural direction: from descriptive data to properties. Until the end of this section, we will refer to this Fortran code as our evaluation function.

In the framework of our problem, we have 16 scalar inputs, with 2 discrete variables, representing the Design Parameters, which are our degrees of freedom, defined onto intervals. We call X the design space, which is extended to a wider space, \hat{X} , to get all necessary input values to run the calculation. \hat{X} , the Detailed Parameters space, is typically of dimension 2 000. Then, we compute the properties values on which our constraints are applied. We call the property space Y . The calculation flow is represented in figure 2.12.

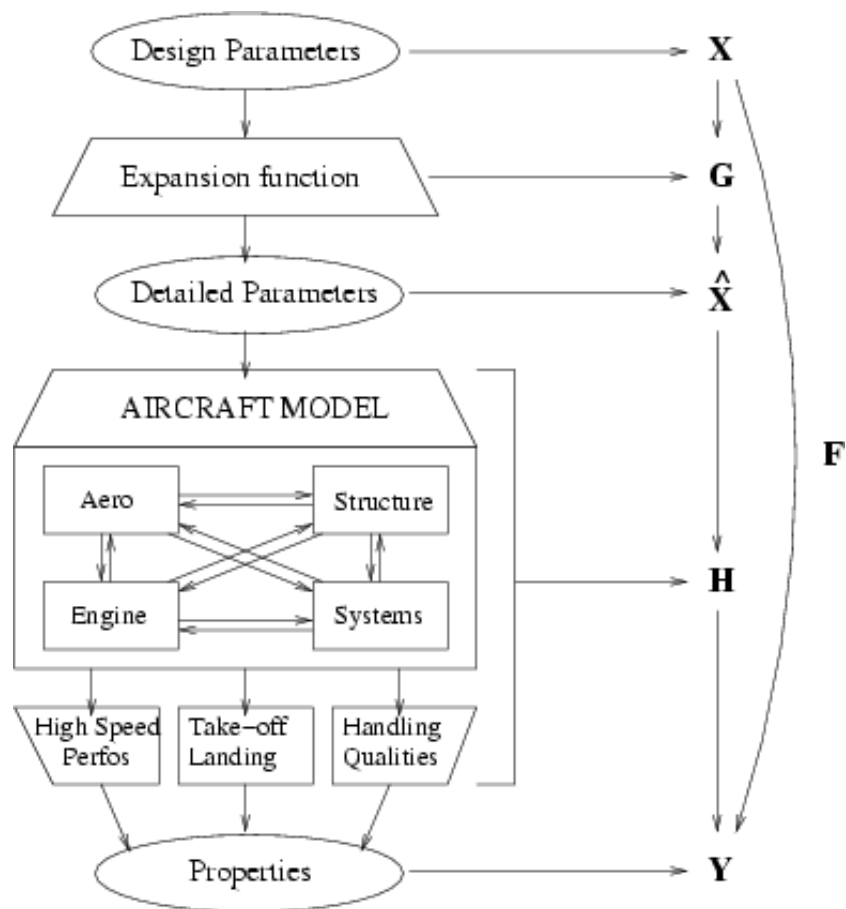


Figure 2.12: Evaluation function architecture in AVION software

2.3.2.2 Numerical and computational aspects

The practical implementation of the evaluation function brings one major issue. The iterative resolutions, present in some modules to solve their internal equation systems, introduce numerical noise. The consequence is that repeating the evaluation of one fixed point produces results with small differences.

Another important problem is linked to the dimension of the design search space. Values given to the evaluation function can be out of the definition space of some modules, particularly if close to the “corners” of the domain. The evaluation function makes possibly meaningless extrapolations, thus we cannot be confident in these results.

Because of the numerical complexity of the evaluation function, we have only little information on its mathematical properties. But, at least the function is not continuous, because of discrete variables, of possible evaluation failures and of the ForTran code itself (which contains a lot of *goto*). And even if it is continuous, it may be a piecewise linear function, as defined through the model. Consequently, we do not have any reliable gradient information.

Concerning the shape of the admissible set, we have no *a priori* information, but the design space and the evaluation function are so complex that we cannot assume that the admissible set is convex or connected.

Until the end of this section, we will consider the evaluation function as a black box.

The problem we are going to treat in the following parts is the one described in appendix B, page 227. It contains 16 degrees of freedom, 2 of them are discrete variables. For each input parameter, we define a search interval. Our search space is thus a 16-dimensional hyperbox. Among all the calculated properties, we consider 16 of them, all continuous variables, on which we apply inequality constraints. Finally, we have 21 constraints.

In the following section, we are going to explain the way we decided to handle constraints.

2.3.3 Indirect constraint handling

As we wanted to use Genetic Algorithms to solve our optimisation problem, the first idea we had to manage constraints was to indirectly handle them, *i.e.* to add a penalty function to the objective function to define the genetic algorithm fitness function [Eiben, 2001; Craenen et al., 2001; Craenen et al., 2003]. This penalty function would quantify the violation of the constraints.

This way to handle constraints means transforming the constrained optimisation problem into an unconstrained problem. Our aim was to apply such technics as the one described in [Craenen and Eiben, 2001]. The basic idea is that constraints that are not satisfied after a certain number of search steps must be hard and therefore, be given more attention. This is realized by using a weighted sum of constraint violations as fitness function and varying these weights to direct the search.

Thus, we decided to transform our initial problem, described in equation 2.13,

$$\begin{aligned} \min \quad & f(x) \\ \text{s.t.} \quad & g(x) \leq 0 \end{aligned} \quad \text{where } f : \mathbb{R}^n \mapsto \mathbb{R}^m \text{ and } g : \mathbb{R}^n \mapsto \mathbb{R}^q, \quad (2.13)$$

into the new unconstrained optimisation problem 2.14,

$$\min h(x) = f(x) + \gamma \sum_{i=1}^n \lambda_i \cdot S(g_i(x)), \quad i = 1, \dots, q \quad (2.14)$$

where γ is used to favor the objective or the constraint aggregation during the optimisation, λ_i are constant parameters to homogenize the sum of constraints, and S is the penalty function, which is zero when the constraint is satisfied, and the absolute value of the violation of the constraint, if not satisfied. The figure 2.13 page 87 is an illustration of such a penalty function like S .

Starting from this new formulation, we decided to tune the parameters γ and λ_i before the beginning of the optimisation. We performed a uniform random sampling inside our Design Space of about 10 000 points, and we calculated the values of the constraints and of the objective function for all these points.

Then, we noticed that among all the points we calculated, none of them satisfied the constraints, and the variations of the objective function were small. Thus, as soon as we noticed this, we envisaged another method to ensure the constraint satisfaction than the indirect constraint handling, because finding an admissible point inside the vast Design Space appears to be too difficult.

We decided then to uncouple the constraint satisfaction problem, further denoted CSP, from the research of optimal solutions. Our aim is to produce large amounts of design points satisfying all the constraints before considering the objective function.

2.3.4 Constraint Satisfaction Problem (CSP) of aircraft sizing studies

CSPs are particular mathematical problems where one must find states or objects that satisfy a number of constraints. CSPs are the subject of intense research in both artificial intelligence and operational research. Many CSPs require a combination of heuristics and combinatorial search methods to be solved in a reasonable time.

The paper [Dechter and Rossi, 2000] is a survey which describes all CSP categories, and the existing methods to solve them.

We decide to treat this CSP as an optimisation problem to give an initial structure for the next steps of the global optimisation. To choose the optimisation process, we have been inspired by the FSQP principle, namely to reach the admissible set before performing any optimisation.

Our aim is to automatically produce large amounts of design points satisfying the requirements, despite the numerical noise introduced by the evaluation function. Moreover, as the

design space is vast, due to the large number of degrees of freedom, it contains meaningless design points (physically speaking), making the evaluation function sometimes fail.

Two optimisation method classes can be envisaged to solve this problem of constraint satisfaction:

1. gradient methods, which are supposedly fast but non-robust against evaluation failures,
2. stochastic methods, which are supposedly slow but can allow failure handling, through some adaptations.

We implemented a genetic algorithm method dedicated to our context, with a specific penalty function deduced from the constraints, applying linear penalties with physically meaningful thresholds.

We compared our method with the FSQP gold standard in constraint satisfaction. We performed this comparison under the same constraint satisfaction problem.

2.3.4.1 Existence of solutions

Before starting our study, we knew the admissible set was not empty because engineers currently working on the problem exhibited such a point, satisfying all the requirements.

To collect more information about the admissible set shape, we decided to try finding other admissible points without using any constraint satisfaction method. This would also give us a first guess of the difficulty to reach the admissible set.

We used a Monte Carlo method by performing a uniform random sampling inside the search interval of each variable. Then, we simply applied the evaluation function on these design parameters and checked the constraint values.

Finally, among 10^6 calculated points,

- 52% were evaluable,
- 5 were admissible,
- 190 satisfied all the constraints but 1.

Thus, finding admissible points is not trivial, and the constraints satisfaction method will have to be adequately fitted.

2.3.4.2 Methods

2.3.4.2.1 Problem formulation We decided to solve the constraint satisfaction problem as an optimisation problem using a penalty method as the objective function. To give a general

structure to our formulation, we called X our design search space of dimension n . Suppose we have p continuous variables and $n - p$ discrete variables. We denote:

$$X = [l_1, u_1] \times \cdots \times [l_p, u_p] \times \{l_{p+1}, l_{p+1} + 1, \dots, u_{p+1}\} \times \cdots \times \{l_n, l_n + 1, \dots, u_n\} \quad (2.15)$$

Let G be the expansion function, to get from the Design Parameters to all the Detailed Parameters, and let H be the practical evaluation function.

$$G : X \rightarrow \hat{X}; \quad H : \hat{X} \rightarrow Y. \quad (2.16)$$

Let F be our evaluation function (see in figure 2.12):

$$\begin{aligned} F : X \subset \mathbb{R}^n &\rightarrow Y \subset \mathbb{R}^q \\ x &\mapsto F(x) = H(G(x)). \end{aligned} \quad (2.17)$$

Then, we introduce our penalty function. Let $\varepsilon : Y \rightarrow \mathbb{R}$ be a criterion on constraint satisfaction: ε has global minima at all admissible design points.

The optimisation problem to be actually solved is:

$$\min_{x \in X} \varepsilon(F(x)) \quad \text{where} \quad \begin{cases} x \in X \subset \mathbb{R}^n \\ F : X \rightarrow Y \subset \mathbb{R}^q \\ \varepsilon : Y \rightarrow \mathbb{R} \end{cases} \quad (2.18)$$

We had three main requirements to choose the optimisation method to solve our CSP:

- it had to be robust against possible evaluation failures,
- it had to manage discrete variables,
- at last, we wanted a homogeneous sampling of the admissible set, preferably with a quite uniform distribution of feasible points in the admissible set.

Consequently, we chose to implement genetic algorithms, because they allow to manage populations of design points, which could fulfill our needs.

2.3.4.2.2 Penalty function build-up As we said previously, classical genetic algorithms cannot manage constraints. Thus we had to implement a penalty function whose minimisation would lead to the admissible set.

First, as constraints were simple or double inequalities, we transformed each constraint to give them the same formulation.

Let $(F(x))_i$ be the i^{th} component of the vector $F(x) \in Y \subset \mathbb{R}^q$, n_1 the number of double bound inequalities, n_2 the number of upper bound inequalities and n_3 the number of lower bound inequalities, then $n_1 + n_2 + n_3 = q$. Let $\xi(x)$ be the vector containing in its components

ξ_i the new expressions of the constraint applying on $F(x)$. $\xi \in \mathbb{R}^s$ where $s = 2n_1 + n_2 + n_3$. Indeed, we have:

$$\begin{aligned}
 \bullet \quad l_i \leq (F(x))_i \leq u_i &\Leftrightarrow \begin{cases} \xi_{2i-1} = l_i - (F(x))_i \leq 0 \\ \xi_{2i} = (F(x))_i - u_i \leq 0 \end{cases} & \text{for } i = 1, \dots, n_1 \\
 \bullet \quad (F(x))_i \leq u_i &\Leftrightarrow \xi_{n_1+i} = (F(x))_i - u_i \leq 0 & \text{for } i = n_1 + 1, \dots, n_1 + n_2 \\
 \bullet \quad l_i \leq (F(x))_i &\Leftrightarrow \xi_{n_1+i} = l_i - (F(x))_i \leq 0 & \text{for } i = n_1 + n_2 + 1, \dots, q
 \end{aligned} \tag{2.19}$$

Then, we introduced the function S to be applied to ξ_i , $i = 1, \dots, s$:

$$S(\xi_i) = \begin{cases} 0 & \text{if } \xi_i \leq 0, \\ \xi_i & \text{otherwise.} \end{cases} \tag{2.20}$$

If a constraint ξ_i is not satisfied, then $S(\xi_i)$ returns an estimation on how much the constraint is violated, else it is null. The figure 2.13 is an illustration of the penalty function S .

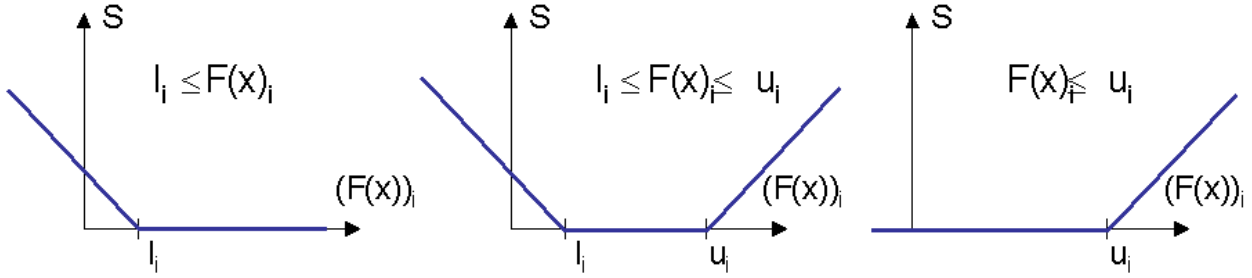


Figure 2.13: Elementary penalty function representation

The smoothness of S is not required as genetic algorithms do not regard gradients.

Finally, the global penalty function is a weighted sum of componentwise penalties. These weights, $\lambda_i \geq 0$, $i = 1, \dots, s$ are set to ensure homogeneous contributions of all constraints. Our final penalty function is:

$$\varepsilon(f(x)) = \sum_{i=1}^s \lambda_i \cdot S(\xi_i(x)). \tag{2.21}$$

2.3.4.2.3 Genetic algorithm implementation For the sake of the good exploration of the admissible set and of the variety of admissible points, we decided to implement the *Non dominated Sorting Genetic Algorithm* (NSGA) method, developed by Srinivas and Deb [Srinivas and Deb, 1994]. In addition to the standard fitness, this method sorts the population depending on the distance to the closest design point in the population. Consequently, the individuals do not tend to aggregate.

Our genetic algorithm implementation was done with the following specifications:

- Each gene represents one degree of freedom, thus each individual is composed of 16 genes.
- The initial population is composed of 50 individuals, whose genes are obtained from uniform randomized sampling inside respective bounds.
- Reproduction is done by mixing genes of two population subsets: the first one is containing the best individuals, the second one is containing a random sample of the current population. As a result, we get one child per crossing of two individuals. The number of children is also 50, the crossed population size is thus between 50 and 100 individuals once clones are removed.
- Selection of individual x_j is done according to its fitness \mathcal{F}_j , inversely proportional to its penalty value $\varepsilon(F(x_j))$. If several points have the same penalty value, they are allocated the same fitness.

Then, as we want to explore uniformly the admissible set, the fitness is weighted according to the distance between points in the population. Thus the final fitness f_j for the j^{th} individual is:

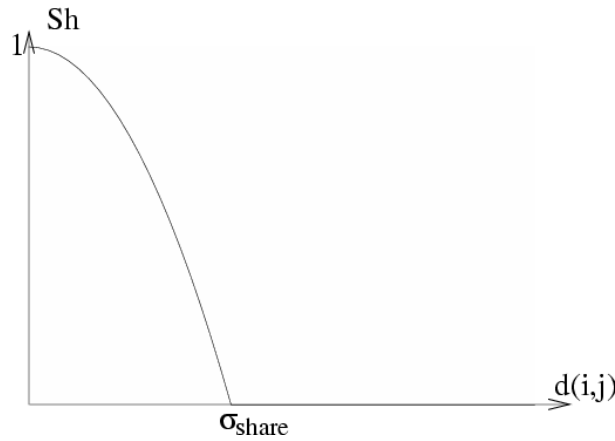
$$f_j = \frac{\mathcal{F}_j}{m_j} \quad \text{where} \quad m_j = \sum_{k=1}^K Sh(d(j, k)), \quad (2.22)$$

where K is the number of individuals in the crossed population, $d(j, k)$ is the distance between individuals indexed by j and k , Sh is a function depending on the distance between two individuals and on a reference distance σ_{share} defining an estimate of the minimum distance we want between individuals.

$$Sh(d(i, j)) = \begin{cases} 1 - \left(\frac{d(i, j)}{\sigma_{share}}\right)^2 & \text{if } d(i, j) < \sigma_{share} \\ 0 & \text{otherwise.} \end{cases} \quad (2.23)$$

The function Sh is represented in figure 2.14.

- Mutation strategy has to be done by a specific function. Considering that $d_x = \varepsilon(F(x))$ is an estimate of the distance to the admissible set, the mutation probability is calculated through a function P , depending on d_x . The function P has to satisfy 3 basic properties:
 1. $P(0) = 0$: if the distance to the admissible set is null, then the point has reached it, there is no reason to mute any gene. The mutation probability is then 0.
 2. $P(d_x) > 0$ when $d_x \rightarrow 0^+$: if d_x is not null, even in the neighbourhood of 0, we have to mute to increase chances to reach the admissible set.

Figure 2.14: Representation of the sharing function Sh

3. P constant for one given d_{\max} : The mutation probability must increase towards a threshold because the probability to mute shall not increase once a certain distance from the admissible set is reached.

We choose a piecewise linear function satisfying these three propositions, which is continuous, except in zero. This function is represented in figure 2.15.

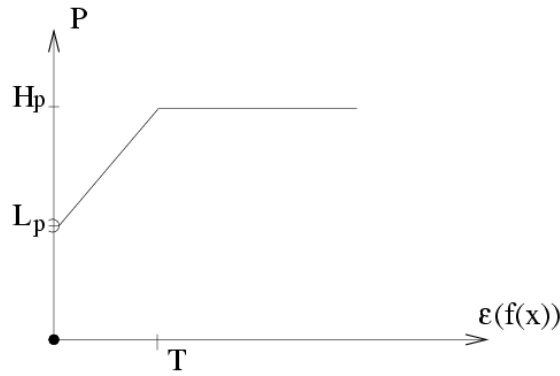


Figure 2.15: Mutation function representation

This mutation function is defined with three parameters, a distance threshold T above which the mutation probability remains constant, and a low and a high probability parameters, L_p and H_p , which define the slope of the function. Thus, we have:

$$P(d_x) = \begin{cases} 0 & \text{if } d_x = 0 \\ L_p + \frac{H_p - L_p}{T} d_x & \text{if } 0 < d_x \leq T \\ H_p & \text{if } d_x \geq T \end{cases} \quad \text{with } \begin{cases} L_p \leq H_p \\ H_p \leq 1 \end{cases} \quad (2.24)$$

- Algorithm convergence is obtained when all population points have reached the admissible set, *i.e.* $\max_{x \in \text{Population}} \varepsilon(f(x)) = 0$.

2.3.4.3 Numerical results

In test phases, the calculations were stopped after 50 generations. It is worth noting that the evaluation of one design point takes typically 5 seconds. The calculation time for one generation was roughly five minutes and it took globally six hours for the overall calculation. Calculations were computed on an Ultra SPARC III machine at 1.2 GHz.

2.3.4.3.1 Tuning of the mutation function parameters We chose two values for the threshold applied on the constraint satisfaction criterion ε , T : 1 and 3. Then, we chose three different values for the high probability parameter H_p , 50%, 25% and 5%. The low probability parameter L_p depended on the high probability parameter, its value was either $\frac{1}{5}H_p$ or $\frac{1}{10}H_p$. Thus, we performed a comparison, based on convergence speed, on twelve different configurations for the mutation function parameters, all calculations started with the same initial population.

The figure 2.16 illustrates the convergence speed of the algorithm for two different sets of the mutation function parameters, exhibiting completely different behaviours, depending on the parameters choice.

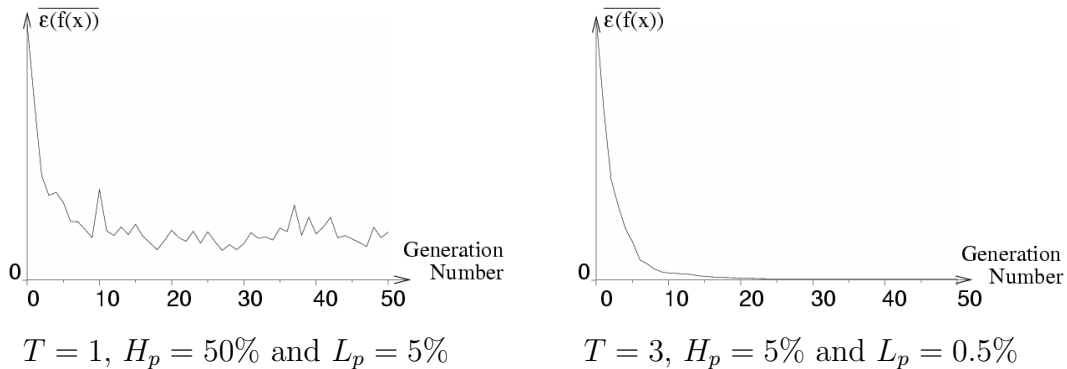


Figure 2.16: Comparison of convergence depending on mutation function parametrisation. For two distinct parametrisations, we plotted the evolution of the average of the distance to the admissible set, showing highly differing behaviours of the optimisation process.

Among all these possible parametrisations, we had to find a trade-off between optimality and variability. We chose the parametrisation showing the fastest convergence speed, with a sufficient global search inside the design space. The selected mutation function parameters were:

$$\begin{cases} T & = & 3 \\ H_p & = & 5\% \\ L_p & = & 1\% \end{cases}$$

2.3.4.3.2 Success rate With this set of mutation function parameters, we tested our algorithm on 10 different initial populations.

After 50 generations, in 8 cases, at least one point reached the admissible set, and all points in the population were admissible in 5 cases.

2.3.4.4 Comparison with FSQP

We compared genetic algorithms with a gradient based-method: FSQP. We chose FSQP among different softwares performing global optimisation (some of them are described in [Mongeau et al., 2000]) because it is dedicated to problems where the objective evaluation is difficult out of the admissible domain. Thus, this choice was natural as FSQP was designed to first exhibit an admissible point, before optimising the criterion.

We used FSQP with its constraint satisfaction mode [Zhou et al., 1997], *i.e.* when there is no objective function.

The main problem we had to deal with was about discrete degrees of freedom. As the algorithm could not manage them, we split the process into two steps.

First, FSQP considered that all degrees of freedom are continuous. This process did not make the calculation fail. Indeed, all the equations contained in our model of aircraft consider that all the variables are continuous. When engineers are dealing with discrete variables, they are always inputs of the equations, like the number of passengers or the number of engines.

Thus it allowed also FSQP to get a gradient relative to these discrete degrees of freedom using finite differences.

After this first pass results, we rounded the discrete degrees of freedom values to the nearest integers, and then, ran another FSQP optimisation, but this time, discrete variables were fixed to these values and not optimised anymore.

To make an unbiased comparison between our genetic algorithm and FSQP methods on their efficiency to reach the admissible set, first, we built a random initial population with 50 design points to run genetic algorithms. Then, we performed successively as many FSQP constraint satisfaction using as starting point each individual of the initial genetic algorithm population.

Genetic Algorithm results:

- It took nearly 2h10 for the genetic algorithms to produce one admissible point and 3h15 to converge, *i.e.* having 50 admissible points.
- If we consider the success rate as the number of admissible points over the number of points to optimise, it was of 100% for genetic algorithms.
- If we consider the efficiency as the ratio of the number of admissible points over the time needed to produce them, we had an efficiency of 15.4 points per hour for genetic algorithms, concerning this test case.

FSQP results:

- In the first pass results, FSQP produced 11 admissible points in nearly 3h50. With these 11 points, we built 16 new starting points for the second step of our FSQP constraint satisfaction method. And finally we got 16 points satisfying the constraints with fixed discrete degrees of freedom values obtained in the first step. So, FSQP produced 16 admissible points out of 50 starting design points in 5h40 of calculation.
- The success rate was 32% for FSQP.
- We had an efficiency of 2.8 points per hour for FSQP, concerning this test case.

In this context, genetic algorithms showed more robustness and efficiency than FSQP, which is not really surprising. All results are summarised in table 2.1.

	Genetic algorithms	FSQP
Convergence time (h)	3.25	5.66
Admissible points (out of 50)	50	16
Success rate (adm point/initial point)	100%	32%
Efficiency (adm point/h)	15.4	2.8

Table 2.1: Test case numerical results

2.3.5 Conclusion

We presented a novel method to automatically produce large amounts of admissible design points in the context of aircraft sizing. Our dedicated implementation of genetic algorithms to solve a constraint satisfaction problem succeeds in 80% of the cases to produce roughly 50 admissible points, which correspond to the size of the population. Moreover, in this aircraft sizing context, genetic algorithms show more robustness but also, more surprisingly, higher productivity than a gold standard dedicated method like FSQP, producing in average one admissible point every 15 minutes.

In addition, this is done in a relatively short time frame, compared with the time engineers need to exhibit a single admissible point.

To explain the 20% of failure cases, we notice that the number of individuals in the current population, nearly 100 individuals including the parents and the children, is small relatively to the 16-dimensional design space. So the genetic algorithm can find a minimum which is local.

Moreover all the calculations were stopped after 50 generations, they could surely have converged if we had let them have enough time. But due to the limited time we had, we could not afford to let the calculations go on until the convergence.

Our next step will be to improve the success rate of our algorithm, by increasing population size and maximising population dispersion. The step after is to perform a mono-criterion

optimisation under constraints, *i.e* simultaneously global optimisation of one figure of merit with constraint satisfaction.

2.4 Robust mono-criterion optimisation

At this stage of the study, we decided to change the evaluation function, because the current one, coming from *AVION*, is too heavy. It takes 5 seconds to calculate one point, when it is evaluable, and the evaluation failures slow down the calculations of population of points, thus the overall GA process can be launched only at night. For these reasons, development and research were not easy to perform.

We decided to change the evaluation function to use a simplified model of aircraft which has to be more research-friendly than the previous one. Thus, we decided to use the USMAC model, which is described in details in the previous chapter, section 1.3.2.2, page 45. This function is more robust and rarely fails, and the evaluation of one point lasts 0.11s of CPU on a P4 with 1.70 GHz. The calculation times on this machine are comparable to those previously obtained.

2.4.1 USMAC implementation

2.4.1.1 Variable description

2.4.1.1.1 Degrees of freedom The USMAC has possibly up to 6 degrees of freedom:

- the sea level static reference thrust, $FNslst$ (daN),
- the engine by-pass ratio, BPR (no dimension),
- the wing area, $Awing$ (m²),
- the wing span, $span$ (m),
- the wing reference sweep angle, phi (deg),
- the wing reference thickness to chord ratio, tuc (no dimension).

The table 2.2 summarises the intervals of definition of the search space of our problem.

10000	\leq	$FNslst$	\leq	20000
5	\leq	BPR	\leq	8
40	\leq	$Awing$	\leq	400
16	\leq	$span$	\leq	80
15	\leq	phi	\leq	40
0.08	\leq	tuc	\leq	0.12

Table 2.2: Definition of the search space of our problem

Most of the time, the methods we developed were tried on the two main ones, *FNslst* and *Awing*, to be able to draw graphics, and visualise the results. The other four variables are fixed to meaningful values, see table 2.3.

BPR	=	6
span	=	34 m
phi	=	25 deg
tuc	=	0.11

Table 2.3: Fixed values for secondary-level degrees of freedom

2.4.1.1.2 Constraints We are dealing with 6 constraints:

- the take-off field length, *TOFL* (m),
- the approach speed, *Vapp* (kt),
- the aspect ratio, *AR* (no dimension),
- the fuselage fuel ratio, *Kff* (no dimension),
- the climb speed, *Vz* (ft/min)
- the cruise thrust, *Kfn* (no dimension).

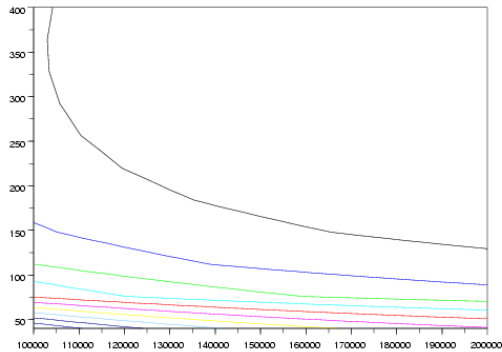
The figure 2.17 is a representation of ten isocurves of the constraints according to the two main degrees of freedom *FNslst* and *Awing*.

The 6 constraints of this new aircraft sizing problem are defined by the following relations:

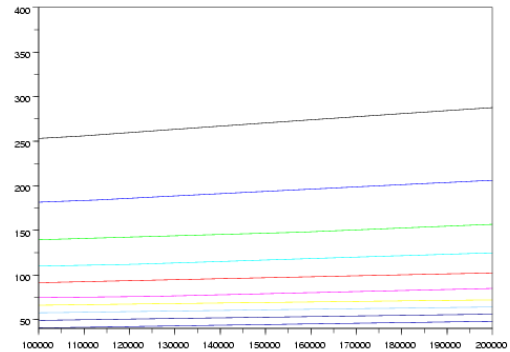
$$\begin{array}{rcl}
 \text{TOFL} & \leq & 2000 \text{ m} \\
 \text{Vapp} & \leq & 120 \text{ kt} \\
 \text{AR} & \leq & 11 \\
 0.75 & \leq & \text{Kff} \\
 500 \text{ ft/min} & \leq & \text{Vz} \\
 & & \text{Kfn} \leq 1
 \end{array}$$

Table 2.4: Definition of the constraint bounds of our problem

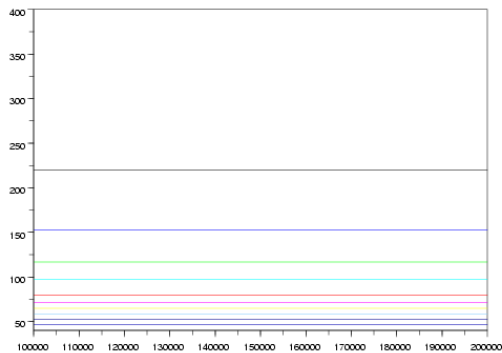
The boundary of the admissible set and the related constraints are represented on figure 2.18.



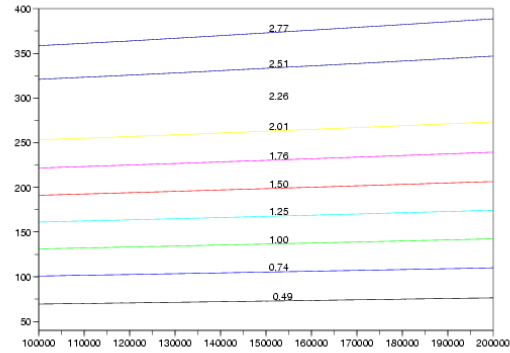
$TOFL$



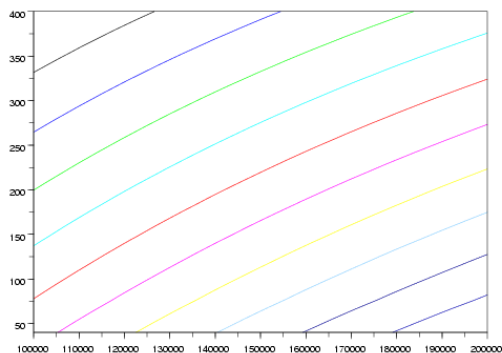
V_{app}



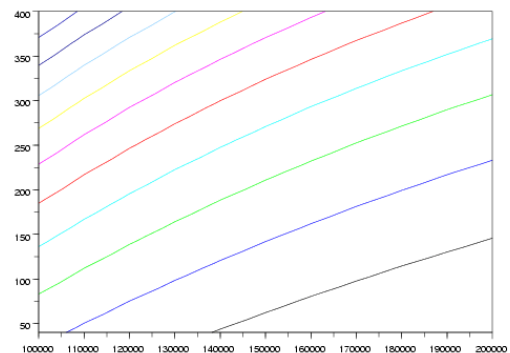
AR



K_{ff}



V_z



K_{fn}

Figure 2.17: Isocurves of the constraints according to the two main degrees of freedom $FNslst$ and $Awing$

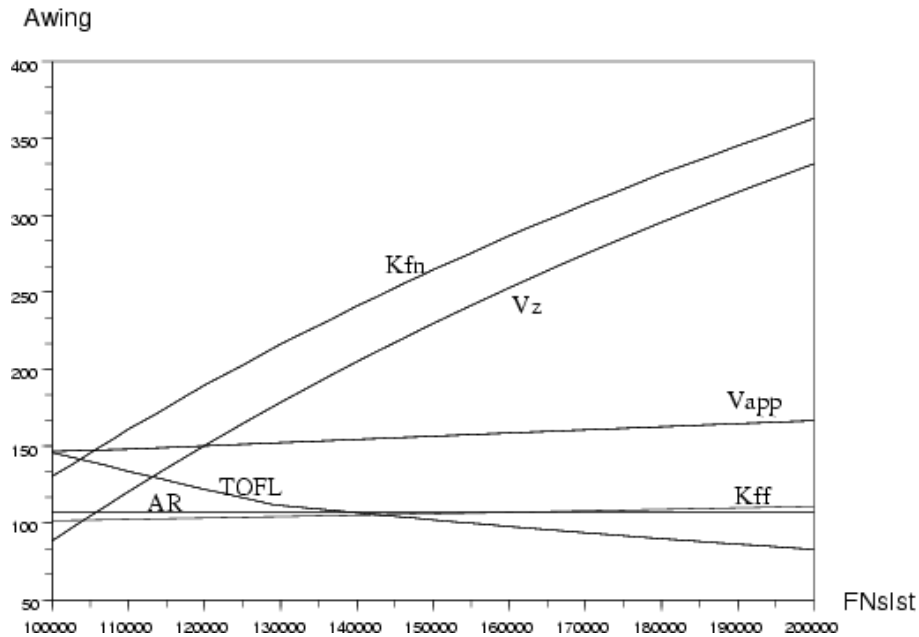


Figure 2.18: Constraint position in the search space

2.4.1.1.3 Criteria Again, in this new problem, there are 4 criteria to minimise:

- the maximum take-off weight, $MTOW$ (kg),
- the nominal fuel, $Fuel$ (kg),
- the weight efficiency, the ratio of MWE , the manufacturer weight empty, over the $MTOW$, $MWEoMTOW$ (no dimension)
- the fuel efficiency, the ratio of the fuel over the number of passengers over the range, $FoPoNe6$ ($10^6\text{kg}/\text{m}/\text{Pax}$).

The figure 2.19 is a representation of the isocurves of the criteria according to the two main degrees of freedom $FNslst$ and $Awing$ with the limit curves of the constraints.

2.4.1.2 Adaptation of the previous work to the new problem

Once the framework has been established, and the new evaluation function plugged to our implementation of genetic algorithms, the same work than in the previous section has been done to calculate the parameters which define the mutation distribution. Finally, as the evaluation function is more robust, as the calculation time is shorter and the design space smaller, the values of the parameters changed to:

$$\begin{cases} T & = 3 \\ H_p & = 50\% \\ L_p & = 10\% \end{cases}$$

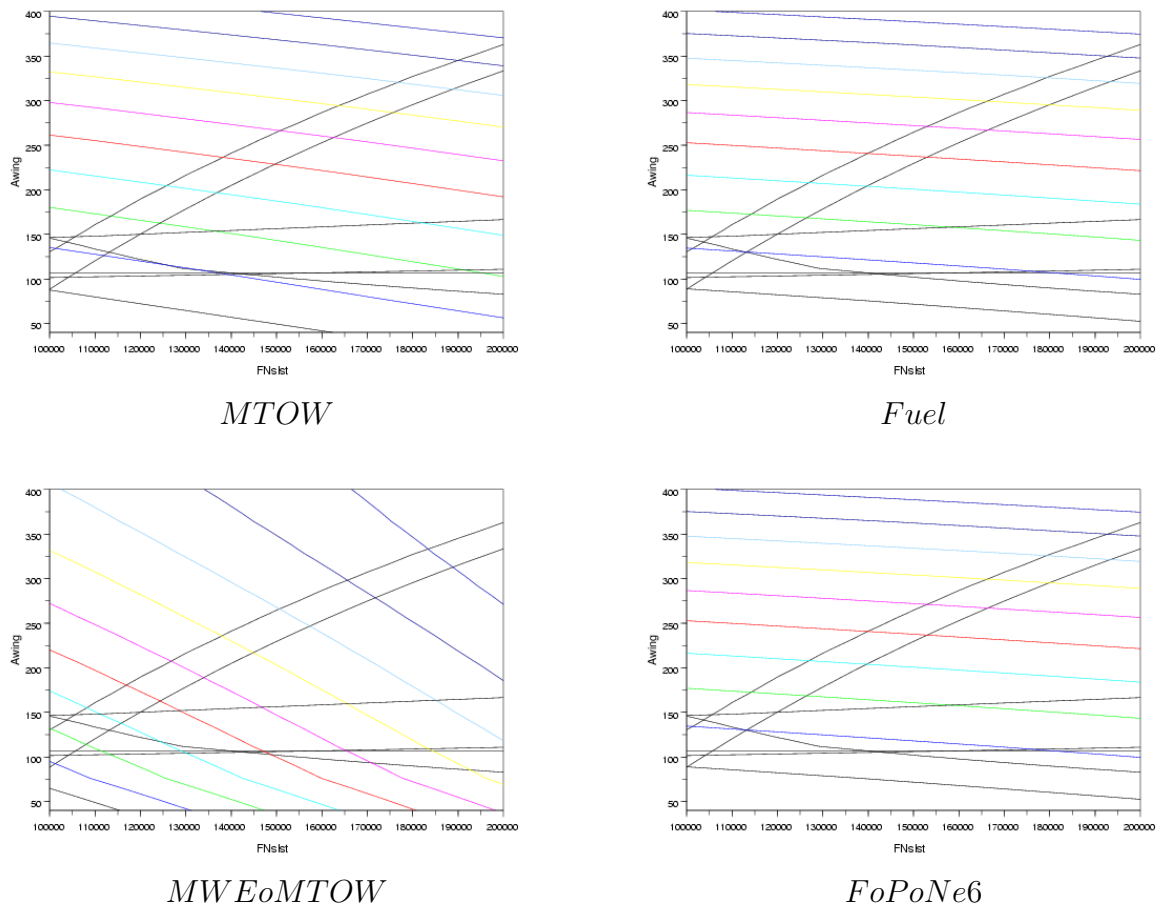


Figure 2.19: Isocurves of the criteria according to the two main degrees of freedom $FNslst$ and $Awing$ with the limit curves of the constraints

2.4.2 Improvement of initial population repartition

Until now, the generation of the initial population was done randomly in previous section 2.3.4.2.3, page 87, using a Monte Carlo sampling method with uniform distributions on the definition intervals of the degrees of freedom.

According to [Poles et al., 2006], well-distributed samplings allow to gain more information on the problem, and thus increase the robustness and the convergence speed of a generic genetic algorithm. Indeed, if the information contained in the initial population is not sufficient, the algorithm can suffer from premature convergence and get stuck in local optimal solutions.

Poles et al. prove that a well-distributed population speeds up the convergence to the correct Pareto frontier independently from the GA and the problem. Thus, we decided to change the way we generate the initial population to improve the convergence of our algorithm.

2.4.2.1 Generating well-distributed population

Among the techniques to generate well-distributed population, there are Design of Experiments, further denoted DoE.

DoE are a series of tests organised to determine the influence of multiple parameters on one or several responses in a minimum number of tests and with the maximum of precision [Orsi, 1994].

DoE are widely used, not only in numerical applications. These methods were first elaborated to reduce the number of real experiments, they are for instance widely used in agronomy or oil field detection. The manuscript [Van Grieken, 2004] describes a systematic approach to build DoE.

There are several other methods to generate well-distributed samplings, like:

- Sobol, which is a deterministic algorithm, also known as quasi-random. Its aim is to obtain a uniform sampling of the design space.
- Cross-Validation, which distributes the points uniformly on the basis of the Kriging algorithm used for response surfaces. This method estimates the error of the model and then chooses a good new set of points in order to make the response surface more reliable.
- Latin Hypercube Sampling, which is one form of stratified sampling that can reduce the variance in the Monte Carlo estimate of the integrand.

We decided to implement a Latin Hypercube sampling method because it is simple to manipulate in our context, and we don't want to introduce any *a priori* knowledge on the problem.

2.4.2.2 Latin Hypercube Sampling (LHS)

The statistical method of Latin Hypercube Sampling, further denoted LHS, was developed to generate a distribution of plausible collections of parameter values from a multidimensional distribution.

In the context of statistical sampling, a square grid containing sample positions is a Latin square if (and only if) there is only one sample in each row and each column. A Latin hypercube is the generalisation of this concept to an arbitrary number of dimensions, whereby each sample is the only one in each axis-aligned hyperplane containing it.

When sampling a function of N variables, the range of each variable is divided into M equally probable intervals. M sample points are then placed to satisfy the Latin hypercube requirements. Note that this forces the number of divisions, M , to be equal for each variable. Note also that this sampling scheme does not require more samples for more dimensions (variables). This independence is one of the main advantages of this sampling scheme. Another advantage is that random samples can be taken one at a time, remembering which samples were taken so far.

The figure 2.20 is an example of a Latin Hypercube sampling in two dimensions, and the figure 2.21 is an example of a Latin Hypercube sampling in our context, with the representation of the constraint bounds. The admissible set is the smaller triangle containing the red point.

	X			
			X	
				X
X				
		X		

Figure 2.20: Example of a Latin Hypercube sampling

In two dimensions, the difference between random sampling and LHS can be explained as follows:

- In random sampling, new sample points are generated without taking into account the previously generated sample points. Thus, one does not necessarily need to know beforehand how many sample points are needed.
- In Latin Hypercube sampling, one must first decide how many sample points to use and for each sample point remember in which row and column the sample point was taken.

According to [Wang, 2003], LHS has desirable features, like:

- providing more information within a design space,

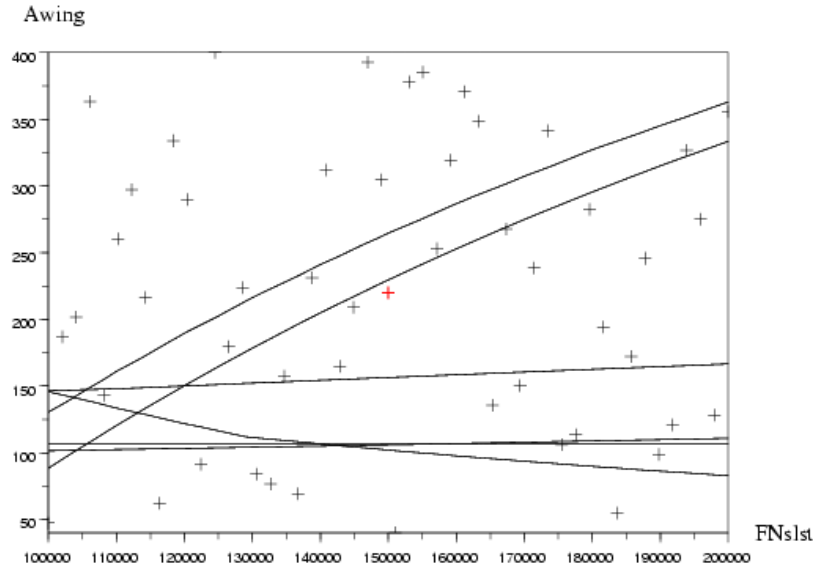


Figure 2.21: Example of a Latin Hypercube sampling in our context

- no interrelationship is pre-assumed between design variables,
- the size of a sample is controllable and determined by the designer.

2.4.3 Producing admissible points with USMAC

Since we work with the USMAC, our dedicated implementation of a genetic algorithm never fails producing admissible points, each time the entire population reaches the feasible set.

This is mainly because the admissible set is comparatively greater than with the *AVION* evaluation function. When 5 points among 1 000 000 of randomly sampled points were admissible with the *AVION* code, with the USMAC model, 15% of randomly sampled points are admissible.

We can now compare the efficiency of producing admissible points according to the initial population. The figure 2.22 is an example of two initial populations produced by a random sampling on the left, and a LHS on the right.

Calculations on 100 initial populations for both cases were launched. The resolution of the constraint satisfaction problem using these two kinds of initial populations succeeds in each case in producing 50 admissible points, the size of the population. The convergence speed is higher for the LHS sampling, the algorithm converges in nearly 26 seconds instead of 27 seconds starting with the random initial population.

A comparison of the evolution of the population is detailed in table 2.5.

Then, we can see on the figure 2.23 the resulting populations inside the admissible set.

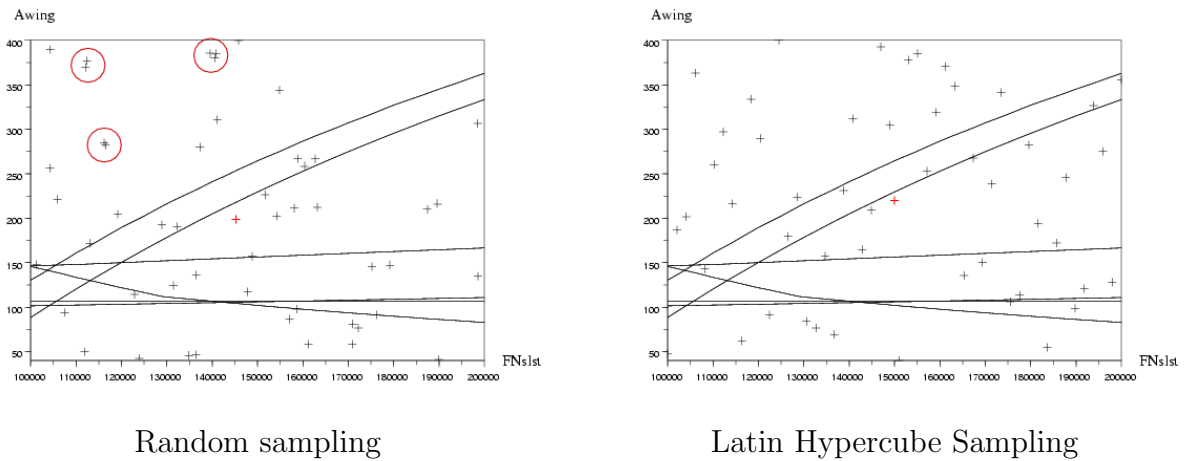
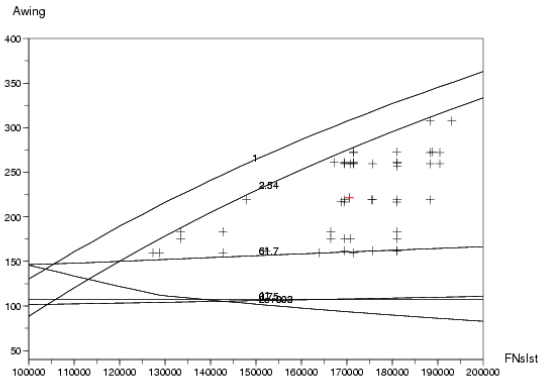


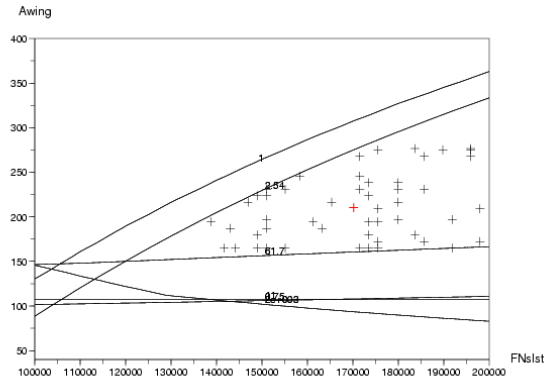
Figure 2.22: Comparison of initial populations: the red circles enlighten some of the clusters of randomly sampled points on the left figure. Such clusters cannot be found on the right figure. The red point in the middle of the figure is not one particular point of the population, but its centre of gravity.

Number of admissible points at generation	Random sampling	Latin Hypercube Sampling
Sampled generation	9.91	10.075
N°1	17.49	18.36
N°2	28.03	29.13
N°3	40.04	41.36
N°4	47.32	48.98
N°5	49.67	49.98
N°6	49.99	50
N°7	50	
Convergence time	27.13s	26.16s

Table 2.5: Test case numerical results



Random sampling



Latin Hypercube Sampling

Figure 2.23: Comparison of admissible populations produced with the two kinds of initial populations. The results look like quite the same, but the dispersion of points inside the admissible domain seems better using a Latin Hypercube Sampling.

2.4.4 Robust mono-criterion optimisation

Robust mono-criterion optimisation is done using FSQP, starting from an initial point located at the centre of gravity of the admissible population found thanks to genetic algorithms (the red point on the figures). Thus the initial point is already admissible, FSQP does not have to put any initial point inside the admissible set.

The optimisation process is done for each criterion we have. FSQP always manages to converge to the right optimum, which is located at the intersection of two constraints (see on figure 2.24).

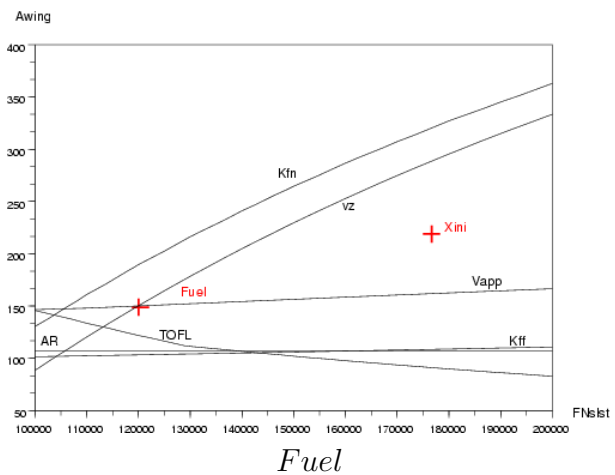
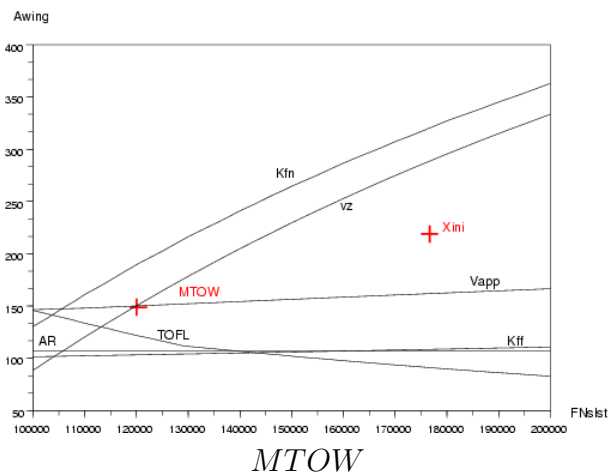


Figure 2.24: Examples of results of the mono-criterion optimisations

By combining our dedicated genetic algorithm to find admissible points to

the standard optimiser **FSQP**, we build a robust process to find global optimal solutions of our aircraft sizing problem.

2.4.5 Going further

This robust mono-criterion optimisation has been successfully tested with a simplified model of aircraft, USMAC. What would be interesting now is to test it on more realistic models of aircraft like the one included in *AVION*.

As an intermediary stage, we made some tests on a more complex model of aircraft, the SMAC, which is described in section 1.3.2.3 page 56. Compared with *AVION* intrinsic model, the methods are almost the same, the platform to plug the disciplines is the main difference.

With the SMAC, our dedicated implementation of a genetic algorithm never fails producing admissible points, each time the entire population reaches the feasible set. Like with the USMAC, this is because the model is robust, thus the evaluation never fails, and the admissible set with the SMAC model is greater than the one using *AVION* model.

The main difference with the USMAC is the calculation time, which lasts 36 seconds for one point, instead of 0.11 seconds for the USMAC using the same machine. Thus calculation were not launched very often.

2.4.5.1 Variables description

Like with the USMAC, we worked on two variables, the main driving characteristics in defining an aircraft configuration:

- a rubbering coefficient of the engine size, *krub* (no dimension),
- the wing area, *wing_area* (m^2).

Our search domain was defined as described in the table 2.6.

$$\begin{array}{rcccl} 0.5 & \leq & \text{krub} & \leq & 2 \\ 350 & \leq & \text{wing_area} & \leq & 600 \end{array}$$

Table 2.6: Definition of the search space of our problem using SMAC

The problem we defined with the SMAC model contained four constraints:

- the Take-Off Field Length, TOFL (m),
- the approach speed, *app_speed* (kt),
- two climb rates, the top-of-climb at max climb rating *cl_rate_1* and the top-of-climb at max cruise rating *cl_rate_2* (ft per min).

The admissible domain was thus defined as described in the table 2.7.

Finally, we computed a mono-criterion using **FSQP** on four criteria:

$$\begin{aligned}
 \text{TOFL} &\leq 1800 \\
 \text{app_speed} &\leq 155 \\
 500 &\leq \text{cl_rate_1} \\
 50 &\leq \text{cl_rate_2}
 \end{aligned}$$

Table 2.7: Definition of the admissible domain of our problem using SMAC

- the Maximum Take-Off Weight, MTOW (kg),
- the Cash Operating Cost, Cash_cost (dollars per passenger per nautical mile),
- the Manufacturer Weight Empty, MWE (kg),
- the Operational Weight Empty, OWE (kg).

The cash operating cost is another expression of costs calculated like the DOC (see in chapter 1, section 1.1.2, page 17), except that the depreciation, the interest and the insurance are not taken into account in its calculation.

2.4.5.2 Results

First, to have an idea of the difficulty of producing admissible points using the SMAC model, we started by interpolating the boundary of the admissible domain defined by the constraints. The result can be seen on the figure 2.25.

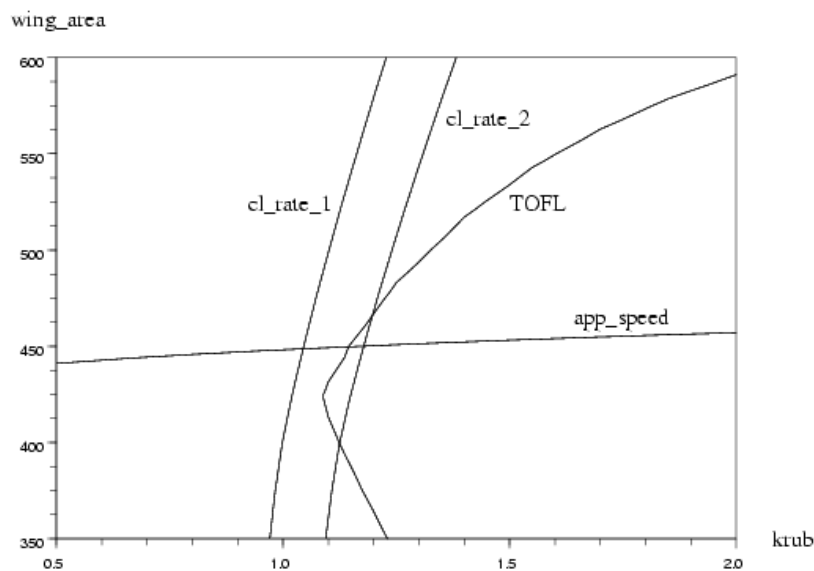


Figure 2.25: Constraint position in the search space using the SMAC

Then, we sampled the initial population using the Latin Hypercube Sampling, and we tested our genetic algorithm dedicated to produce admissible points. Each time the calculation was launched, the algorithm produced as many admissible points as the size of the initial population. Thus, as we cannot afford to perform as many tests as when using the USMAC model, we cannot evaluate the success rate of the algorithm, but we can say that it is good enough for our general problem of aircraft sizing.

The figure 2.26 is an illustration of the kind of results we obtained with this model, still working with a scaling factor of the engine size and the wing area.

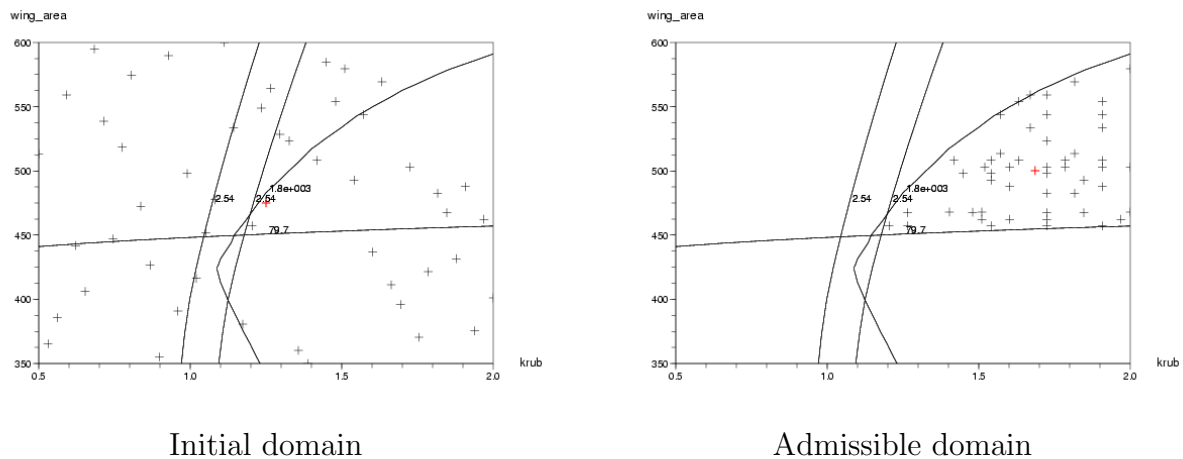


Figure 2.26: Examples of results with the SMAC model

Finally, we launched again FSQP starting from the centre of gravity of the admissible population as initial point. We obtained the same kind of results as for the USMAC. The mono-criterion optimisation using FSQP is robust, and succeeds each time in finding the global optimum of the optimisation problem.

The figure 2.27 shows an illustration of the results we obtained with FSQP, still working with a scaling factor of the engine size and the wing area, and the criterion represented here is the MTOW.

2.5 Conclusion

In the first chapter, we explained that aircraft sizing studies consist in determining characteristics of an aircraft, starting from a set of requirements. These studies can be summarized as global constrained optimisation problems with typically one thousand parameters.

Our aim in the work presented in this chapter was to improve the mono-criterion and constrained optimisation currently performed in FPO, by introducing a new structure to the problem, and new resolution methods coming from the Multidisciplinary Design

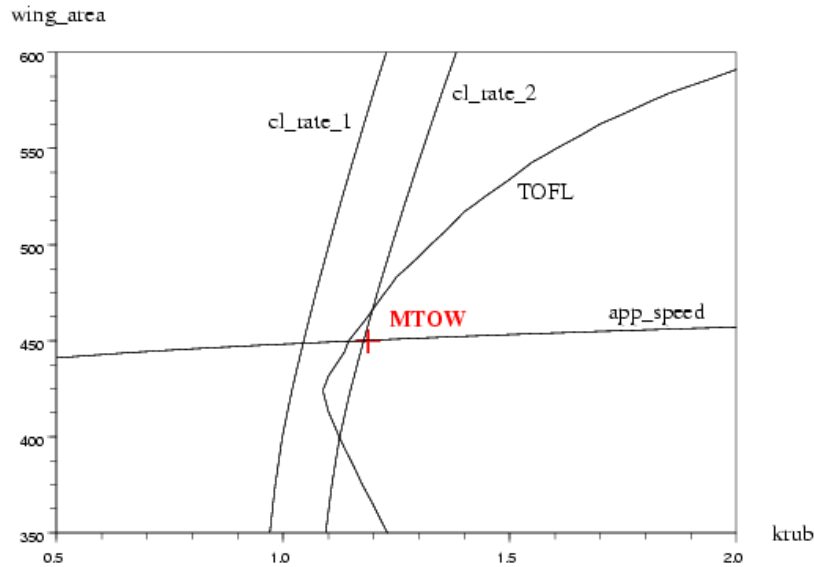


Figure 2.27: Example of results of the mono-criterion optimisation

Optimisation methodology.

It appeared to us that it could be interesting to uncouple the research of optimum solutions from the problem of admissible set extraction. Thus, to solve the aircraft sizing problem, we proceeded in two steps:

1. We first focused on the constraint satisfaction problem, our aim was to automatically produce large amounts of design points satisfying all the constraints, despite frequent evaluation failures, to help the optimisation algorithm being more robust.
2. Then, we performed a global and mono-criterion optimisation with FSQP, starting from feasible points found in the first step.

In this chapter, we presented a new method to automatically produce large amounts of admissible design points in the context of aircraft sizing. Our dedicated implementation of genetic algorithms to solve the CSP has a high success rate, depending on the model used to represent the aircraft. In addition, this is done in a relatively short time frame, compared to the time engineers need to exhibit a single admissible point.

Then, we applied FSQP on the reduced set of the search domain, the admissible domain. Each calculation started from the centre of gravity of the population of admissible points produced by the genetic algorithm and succeeded in finding the global optimum of the mono-criterion optimisation problem.

In our aircraft sizing study, we have several criteria to consider, which are most of the time competing and conflicting measures of the system performance. In this chapter, we performed mono-criterion optimisations, but these processes are not efficient when dealing with conflicting objectives. Thus, one part of this work was to introduce some methods of multicriteria optimisation, which is described in the following chapter.

Chapter 3

Multicriteria optimisation

Contents

3.1	Multicriteria optimisation in engineering	110
3.1.1	Introduction	110
3.1.2	Some definitions	111
3.1.3	Resolution methods	113
3.1.4	Resolution algorithms	127
3.2	Conflicting objectives in our problem	137
3.2.1	Aircraft sizing objectives	137
3.2.2	Conflicting objectives in our models	138
3.3	Refinement of the admissible set	139
3.3.1	Introducing the acceptable domain	139
3.3.2	Method to refine the admissible set	140
3.4	Pareto front	141
3.4.1	The selected method	142
3.4.2	Test problems	143
3.5	Results	147
3.6	Conclusion	151

As we said previously, aircraft sizing studies consist in determining the main characteristic parameters of an aircraft, like its fuselage length or its wing reference area. In mathematical words, aircraft sizing studies can be summarized as constrained global and multicriteria optimisation problems, with typically one thousand variables and parameters. Actually, the constraints express physical feasibility and the requirements to be satisfied, and the objectives are market driven characteristics of the aircraft. Moreover, this is typically a multicriteria optimisation problem because of some conflicting objectives.

In the previous chapter, we treated separately the constraint satisfaction problem related to our aircraft sizing problem. We produced large amounts of admissible design points using a dedicated implementation of genetic algorithm, further denoted GA. Now, having enough design points satisfying the requirements, we can reintroduce the objectives to perform a global optimisation, starting from admissible points and ensuring that the constraints are not violated during the progress of the optimiser.

In section 2.4 page 92, when we reintroduced the objectives, we performed a mono-criterion optimisation, trying to make it more robust than it is currently in FPO. Thus, after finding large amounts of admissible points, we launched an optimisation using the optimiser FSQP starting from the centre of gravity of this set of points. And most of the time, this method to solve the mono-criterion optimisation managed to find the global optimum, making our process more robust.

Before performing any multiobjective optimisation, as one of the main difficulty we faced, especially with the initial aircraft sizing problem, was the huge design space, we decided to refine once again the search space. Indeed, the search space was initially a 16-dimensional hyperbox containing 5 admissible points among 1 000 000 of randomly sampled design points.

When solving the constraint satisfaction problem, we refined the design space a first time to the admissible set, and now, what we intend to do is to refine it once more to a smaller domain which is of the interest of engineers, and finally to focus on the smaller interesting part of the domain, the compromise solution surface.

Thus, in this chapter, in a first part, we are going to explain what are the characteristics of multicriteria optimisation, and what are the FPO needs and the conflicting objectives in aircraft sizing studies. Then, we detail the methodology we developed to refine successively the search space to finally obtain the compromise solution surface.

3.1 Multicriteria optimisation in engineering

3.1.1 Introduction

In most real-life engineering optimisation problems, one attempts to improve or optimise several objectives, frequently competing and conflicting measures of system performance, subject to satisfying a set of design and physical constraints [Alexandrov and Lewis, 1999]. In these cases, it is unlikely that the same values of design variables will result in the best

optimal values for all the objectives. And the main difficulty with monocriterion optimisation is to model the problem with a unique equation and with a unique objective which translates the designer needs with the required accuracy. Multiobjective optimisation techniques are means to find solutions to overcome this difficulty.

Examples of multiobjective problems are:

- in bridge construction, a good design is characterized by simultaneously low total mass and high stiffness,
- in economics, the traditional portfolio optimisation problem attempts to simultaneously minimize the risk and maximize the fiscal return.

[Nakayama, 2005] applies its *Satisficing Trade-off Method* to several real problems, like blending plastic materials, cement production, portfolio, etc.

In these and most other cases, it is unlikely that the different objectives would be optimised by the same alternative parameter choices. Hence, some trade-off between the criteria is needed to ensure a satisfactory design.

Currently in FPO, the optimisation is done on one criterion, like the MTOW, or when engineers want to consider several objectives, they aggregate them into another variable, like a DOC, which is in fact a weighted sum of several costs concerning the aircraft operating or maintenance (see a detailed description of DOC in section 1.1.2, page 17).

What we intend to do here is to separate the different objectives contained in such a variable like the DOC, to perform multicriteria optimisation.

In the following part, we introduce some definitions and we survey different methods to perform multiobjective optimisation.

3.1.2 Some definitions

3.1.2.1 Multiobjective optimisation problem

Multiobjective optimisation, also called multicriteria or vector optimisation, can be defined as follows [Coello Coello, 2000]:

Definition 3.1 (Multiobjective optimisation). Multiobjective optimisation is the problem of finding a vector of decision variables which satisfies constraints and optimises a vector function whose elements represent the objective functions. These functions form a mathematical description of performance criteria which are usually in conflict with each other. Hence, the term “optimise” means finding such a solution which would give the values of all the objective functions acceptable to the designer.

The general multiobjective optimisation problem is posed as follows:

$$\begin{aligned} \min \quad & \begin{cases} f_1(x) \\ f_2(x) \\ \vdots \\ f_k(x) \end{cases} \\ & x \in X \\ \text{subject to} \quad & g_j(x) \leq 0, j = 1, 2, \dots, m \\ & h_l(x) = 0, l = 1, 2, \dots, e \end{aligned} \quad (3.1)$$

where:

- $X \subset \mathbb{R}^n$, n being the number of design parameters,
- k is the number of objective functions,
- m is the number of inequality constraints, e the number of equality constraints.

Let

$$F(x) = \begin{cases} f_1(x) \\ f_2(x) \\ \vdots \\ f_k(x) \end{cases}, G(x) = \begin{cases} g_1(x) \\ g_2(x) \\ \vdots \\ g_m(x) \end{cases}, H(x) = \begin{cases} h_1(x) \\ h_2(x) \\ \vdots \\ h_e(x) \end{cases}$$

and

$$\mathcal{F} = \{x \in X : G(x) \leq 0 \text{ and } H(x) = 0\}.$$

Then our general multiobjective optimisation problem is:

$$\min_{x \in \mathcal{F}} F(x). \quad (3.2)$$

Solutions of a multiobjective problem are rarely a unique solution, but a set of alternative solutions. After finding these solutions, another question arises, **how to select one particular solution among this set**. This depends on the way the decision-maker intervenes in the process.

Multiobjective optimisation methods are divided into four major categories [Marler and Arora, 2004]:

1. methods with **no articulation of preferences**,
2. methods with *a priori* **articulation of preferences**, which implies that the user indicates the relative importance of the objective functions or desired goals before running the optimisation algorithm,
3. methods with a **progressive articulation of preferences**, in which the decision-maker is continually providing input during the running of the algorithm,

4. methods with *a posteriori* articulation of preferences, which means selecting a single solution from a set of mathematically equivalent solutions.

3.1.2.2 Notion of dominance

Once the multiobjective optimisation problem has been solved, we can obtain a great amount of solutions. But only a limited number of solutions is effectively interesting. To refine the set of solutions to this interesting part, we have to introduce the notion of efficient solutions and dominance [Ehrgott, 2005].

Definition 3.2. (Efficient solutions, dominance).

- A point \hat{x} is called **efficient** or **Pareto optimal** if and only if it is a feasible solution and there is no other feasible x such that $F(x) \leq F(\hat{x})$, (*i.e.* $f_i(x) \leq f_i(\hat{x})$ for all $i = 1, \dots, k$) and $f_i(x) < f_i(\hat{x})$ for at least one function.
- If \hat{x} is efficient, $F(\hat{x})$ is called **non-dominated value point**.
- If x^1, x^2 are feasible, and if $F(x^1) \leq F(x^2)$, we say x^1 dominates x^2 and $F(x^1)$ dominates $F(x^2)$.

The Pareto front corresponds to the set of all efficient points [Pareto, 1896]. The image of all Pareto optimal points lies on the boundary of the image of the feasible set (see on figure 3.1). Pareto solutions are such that any improvement in one objective can occur only if at least another objective is degraded. Therefore, it can be stated that no Pareto point is objectively better than another, the choice of one versus another can only be made on the basis of subjective human judgement.

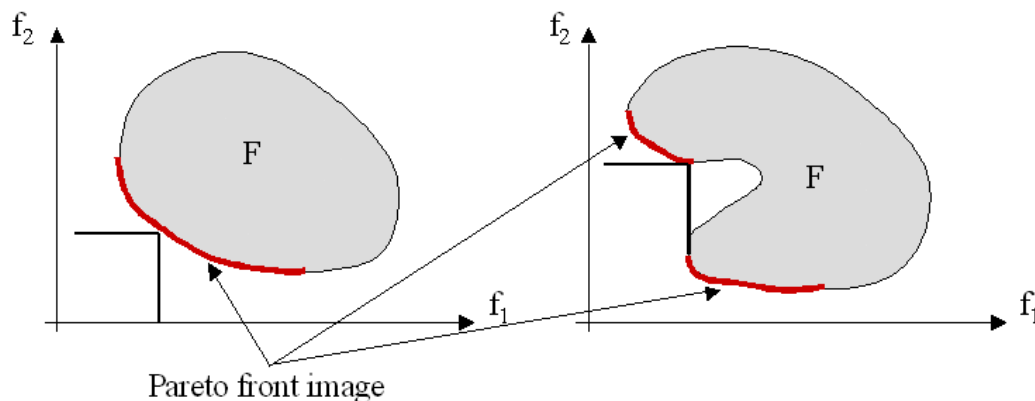


Figure 3.1: Illustration of Pareto fronts, in a convex and a non-convex case, in the objective space

3.1.3 Resolution methods

In this section, we explain some selected methods to solve a multiobjective optimisation problem, according to the way the decision-maker intervenes in the process. This survey is not exhaustive, more detailed descriptions on resolution methods can be found in [Collette and Siarry, 2002; Ehrgott, 2005; Marler and Arora, 2004; Miettinen, 1999].

3.1.3.1 Methods with no articulation of preferences

Methods with no articulation of preferences are used in cases when it is not possible for the decision-maker to express what he/she prefers. We select two examples of methods with no articulation of preferences, the Min-Max method and the Nash arbitration.

3.1.3.1.1 Min-Max method The basic min-max formulation is posed as follows:

$$\min_{x \in \mathcal{F}} \left\{ \max_i f_i(x) \right\}. \quad (3.3)$$

It is the same kind of method as the general weighted sum method, explained in paragraph 3.1.3.2.1, page 115, where:

- there is no weight because the decision-maker does not have to express his/her preferences,
- the norm used here is the L_∞ norm,
- the difference with the weighted sum method is that there is no need to introduce one goal to achieve the objectives.

In order to use this resolution method, objectives have to represent the same kind of quantities, or to be normalised, especially in an MDO context. Indeed, how to compare mass and aerodynamics, for instance, without normalising the criteria?

If the minimum of the optimisation problem (3.3) is unique, then it is Pareto optimal. Otherwise, there are additional conditions for the algorithm to provide a Pareto optimal point, but they are impractical in terms of computational applications [Marler and Arora, 2004].

3.1.3.1.2 Nash arbitration Nash arbitration [Nash, 1951] is a particular case of the *objective product methods*, which consist in minimising the product of the objective functions.

The Nash arbitration approach consists in maximising the objective function F_g in the equation (3.4):

$$F_g(x) = \prod_{i=1}^k (s_i - f_i(x)) \quad (3.4)$$

where s_i is selected as an upper limit value on each objective function.

The solution to this approach depends on the values of s_i , and tends to improve most significantly the objectives the farthest from s_i .

If $s_i \geq f_i(x)$ is guaranteed for each objective function, then the solution of the maximisation of the function F_g is a Pareto optimal point.

The method described in this section is derived from *game theory*. Based on predetermined axioms of fairness, Nash suggests that the solution to an arbitration problem be the maximum of the product of the player's degrees of contentment. When no agreement is reached, in absence of cooperation, the product is zero [Marler and Arora, 2004].

3.1.3.2 Methods with *a priori* articulation of preferences

The most common way of conducting multiobjective optimisation is by *a priori* articulation of the decision-maker preferences. This means that the user has to specify its preferences in terms of goals or relative importance of the different objectives. This method is also called naive approach [Collette and Siarry, 2002; Coello Coello, 2000] because it is the most evident one to solve a multiobjective problem.

Most of these methods incorporate information, which are coefficients, exponents, etc., that can be set by the decision-maker to reflect his/her preferences. Methods with *a priori* articulation of preferences often use scalarised functions or aggregation functions as for the effective objective function.

3.1.3.2.1 Weighted sum method This method is a particular case of weighted global criterion methods, which consist in minimising the following function:

$$U = \left\{ \sum_{i=1}^k w_i (f_i(x) - f_i^o)^p \right\}^{\frac{1}{p}} \quad (3.5)$$

where:

$$\bullet F^o = \begin{cases} f_1^o(x) \\ f_2^o(x) \\ \vdots \\ f_k^o(x) \end{cases} \text{ is called the } \textit{utopia point}, \textit{ i.e. for each } i = 1, 2, \dots, k, f_i^o = \min f_i(x),$$

- w is a vector of weights set by the decision-maker, such that $\sum_{i=1}^k w_i = 1$ and $w_i > 0$, for all $i = 1, 2, \dots, k$,
- the exponent p is also defined by the user, depending on the distance measure he wants to use.

In a typical weighted sum method, $p = 1$ and the utopia point in the definition of the utility function U is assigned to zero, $f_i^o = 0$ for all $i = 1, 2, \dots, k$:

$$U = \sum_{i=1}^k w_i f_i(x) \quad (3.6)$$

The condition in hypotheses that all of the weights have to be positive ensures that the minimum of the equation (3.6) is Pareto optimal [Marler and Arora, 2004].

There are mainly three difficulties in using a weighted sum method:

1. *a priori* selection of weights does not necessarily guarantee that the final solution will be acceptable from the decision-maker point of view, he may have to solve the problem once more with new weights,
2. it is impossible to obtain points on non-convex portions of the Pareto front,
3. varying the weights consistently and continuously may not necessarily result in an even distribution of Pareto optimal points and an accurate complete representation of the Pareto front.

3.1.3.2.2 Lexicographic method This method consists in first classifying the objectives by order of importance, and then successively minimising the following optimisation problem for each objective function:

$$\begin{aligned} \min \quad & f_i(x) \\ \text{s.t.} \quad & f_1(x) = f_1^*, \dots, f_{i-1}(x) = f_{i-1}^* \\ & G(x) \leq 0 \\ & H(x) = 0 \end{aligned} \quad (3.7)$$

where i represents the position of the function in the classification, f_j^* is the optimum of the j^{th} function in the process iteration, $j < i$.

The last value of x is the one minimising the last objective function of the classification, while the other objectives are fixed to the previously found minima. This last value of x is the solution of the optimisation problem when using the lexicographic method.

This method is intuitive, but like for the weighted sum method, it requires an arbitrary ordering by preference of the objective functions. Different ordering of the functions leads generally to different solutions [Collette and Siarry, 2002].

3.1.3.2.3 Goal Programming method The first step of this method is for the decision-maker to fix goals, or targets, for each criterion, denoted here T_j for the j^{th} function.

These values are incorporated into the problem as additional constraints [Coello Coello, 2000]. The optimisation problem is now to minimise the sum of the absolute deviations from

the targets to the actually achieved values of the objectives. The simplest form of this method may be formulated as follows:

$$\begin{aligned}
 & \min \quad (d_1, \dots, d_k) \\
 & \text{s.t.} \quad f_1(x) = T_1 + d_1 \\
 & \quad \quad \quad \vdots \\
 & \quad \quad \quad f_k(x) = T_k + d_k \\
 & \text{and} \quad x \in \mathcal{F}
 \end{aligned} \tag{3.8}$$

where d_i is the deviation from the target T_i , $d_i = f_i(x) - T_i$.

3.1.3.3 Methods with progressive articulation of preferences

Methods with progressive articulation of preferences are interactive methods. They produce only one solution, allowing the decision-maker to adapt his/her preferences during the optimisation process.

3.1.3.3.1 Surrogate Worth Trade off method This method is based on compromise methods, where an interactive process is added to converge towards the preferences of the decision-maker [Collette and Siarry, 2002].

This method is composed of several steps. The first one consists in minimising each objective function separately, except the first one:

$$\begin{aligned}
 & \min \quad f_i(x) \quad \text{for } i = \{2, \dots, k\} \\
 & \text{s.t.} \quad x \in \mathcal{F}.
 \end{aligned} \tag{3.9}$$

Let \bar{x}_i denote one solution of the equation (3.9) and $\bar{f}_i = f_i(\bar{x}_i)$, for $i = \{2, \dots, k\}$.

Then, the next step consists in introducing some coefficients ε_i , for $i = \{2, \dots, k\}$, and the decision-maker has to fix adequate values for these coefficients such that:

- $\varepsilon_i \geq \bar{f}_i$,
- the optimisation problem (3.10) has admissible solutions.

$$\begin{aligned}
 & \min \quad f_1(x) \\
 & \text{s.t.} \quad f_2(x) \leq \varepsilon_2, \dots, f_k(x) \leq \varepsilon_k \\
 & \quad \quad \quad G(x) \leq 0 \\
 & \quad \quad \quad H(x) = 0.
 \end{aligned} \tag{3.10}$$

The next step is now for the decision-maker to give his/her opinion whether he/she wants to increase, or not, the objective function f_i of the value λ_{1i} to obtain a relative gain of 1 on the first objective function f_1 , where:

$$\lambda_{1i} = -\frac{\partial f_1}{\partial f_i}. \quad (3.11)$$

The opinion of the decision-maker is expressed through coefficients W_{1i} , for $i = \{2, \dots, k\}$, varying from -10 to $+10$:

- -10 means unfavourable,
- $+10$ means favourable,
- 0 means that it does not matter.

This process is repeated until all values of W_{1i} , for $i = \{2, \dots, k\}$, are zero during the iterations of the expression of the opinion.

Once satisfaction has been reached, the solution is determined by solving the problem (3.12):

$$\begin{aligned} \min \quad & f_1(x) \\ \text{s.t.} \quad & f_2(x) = f_2^*, \dots, f_k(x) = f_k^* \\ & G(x) \leq 0 \\ & H(x) = 0 \end{aligned} \quad (3.12)$$

where f_i^* is the value of the i^{th} objective function corresponding to the case when the coefficient W_{1i} is zero.

3.1.3.3.2 Fandel method In Fandel method, the preference information from the decision-maker is used to refine step by step the research space. The aim is to help the decision-maker in defining the weights of an aggregation method.

The initial problem (3.2) (described page 112) is modified as follows:

$$\begin{aligned} \min \quad & \sum_{i=1}^k w_i f_i(x) \\ \text{s.t.} \quad & x \in \mathcal{F} \end{aligned} \quad (3.13)$$

where w is a vector of weights such that $\sum_{i=1}^k w_i = 1$ and $w_i > 0$ for all $i = 1, \dots, k$.

Like for the weighted sum method, described in paragraph 3.1.3.2.1 page 115, the variation of the coefficients w_i allows to find points on the compromise surface. But the difference with this method is that these coefficients are not known *a priori*. The aim of the Fandel method is to find a solution which is as close as possible to an “ideal” solution, according to the decision-maker. *This method is a guidance for assigning numerical weights*

to the decision-maker mind-oriented weights.

The first step of the method consists in minimising each objective function separately, while ensuring constraint satisfaction:

$$\begin{aligned} \min \quad & f_i(x) \quad \text{for } i = \{1, \dots, k\} \\ \text{s.t.} \quad & x \in \mathcal{F}. \end{aligned} \quad (3.14)$$

We denote:

- \bar{x}_i a solution of the equation (3.14),
- $\bar{f}_i = f_i(\bar{x}_i)$,
- $F_i^* = F(\bar{x}_i) = (f_1(\bar{x}_i), \dots, f_{i-1}(\bar{x}_i), \bar{f}_i, f_{i+1}(\bar{x}_i), \dots, f_k(\bar{x}_i))$,
- $\bar{F} = (\bar{f}_1, \dots, \bar{f}_k)$. \bar{F} is the “ideal” vector, but it is impossible to get it.
- $B = (F_1^*, \dots, F_k^*)$. B is a matrix containing the values \bar{f}_i on its diagonal, and $f_j(\bar{x}_i)$, with $i \neq j$, elsewhere.

Then, we calculate a vector F^M which will be used to refine the search space:

$$F^M = \frac{1}{k} \sum_{i=1}^k F_i^*. \quad (3.15)$$

This vector is represented on the figure 3.2 page 120.

Finally, the search space will be refined by adding a constraint to the problem (3.14), which depends on the new vector F^M that we have introduced:

$$\begin{aligned} \min \quad & f_i(x) \quad \text{for } i = \{1, \dots, k\} \\ \text{s.t.} \quad & G(x) \leq 0 \\ & H(x) = 0 \\ & F(x) \leq F^M. \end{aligned} \quad (3.16)$$

We note:

- \hat{x}_i a solution of the equation (3.16),
- $\hat{f}_i = f_i(\hat{x}_i)$,
- $\hat{F}_i = F(\hat{x}_i) = (f_1(\hat{x}_i), \dots, f_{i-1}(\hat{x}_i), \hat{f}_i, f_{i+1}(\hat{x}_i), \dots, f_k(\hat{x}_i))$,
- $\hat{B} = (\hat{F}_1, \dots, \hat{F}_k)$.

\hat{B} defines a hyperplane of equation:

$$\hat{B}.a = c.e \quad (3.17)$$

where :

- $a = (a_1, \dots, a_k)^t$,
- $a > 0, \sum_{i=1}^k a_i = 1$,
- $c = \text{constant}$,
- $e = (1, \dots, 1)^t$.

The last step of the method consists in finding the value of the constant c so that the hyperplane is tangent to the admissible domain. The intersection point is the compromise solution (see an illustration on figure 3.2).

If this compromise point is not convenient to the decision-maker, then he/she has to modify the added constraints through the vector F^M to refine the search space to another domain, and thus, he/she has to define a different problem (3.16) to finally solve it again.

3.1.3.3 Other methods There are several other methods with progressive articulation of preferences than the two described previously. There are for instance:

- the STEP method,
- the Jahn method,
- the Geoffrion method,
- a simplex method, etc.

These methods will not be described here. Further details on the methods listed above can be found in [Collette and Siarry, 2002].

3.1.3.4 Methods for *a posteriori* articulation of preferences

These methods generally help in producing a set of alternative design points, allowing the decision-maker to make his/her choice among a set of possible solutions. They are also called *Generate First–Choose Later*.

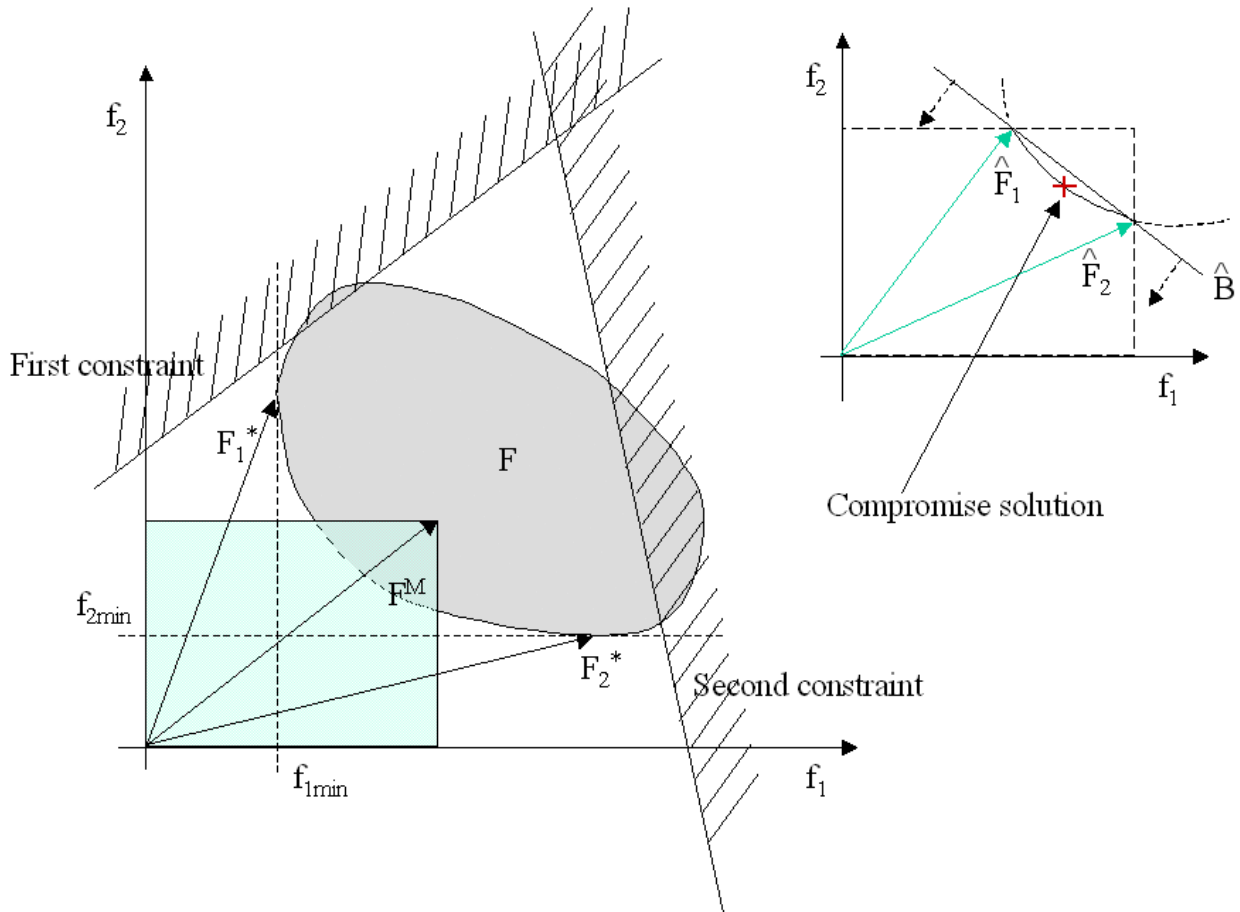


Figure 3.2: Illustration of Fandel method on an optimisation problem with 2 objective functions to minimise and with 2 constraints.

F is the set of all coupled points $(f_1(x), f_2(x))$, for x in the search space, $F_1^* = (f_{1min}, f_2(\bar{x}_1))$ where $\bar{x}_1 = argmin(f_1)$, $F_2^* = (f_2(\bar{x}_2), f_{2min})$ where $\bar{x}_2 = argmin(f_2)$ and $F^M = \frac{1}{2}(F_1^* + F_2^*)$. The search space is refined to the intersection of the green and the grey sets on the figure. The compromise solution is the tangent point to the plane defined through the new extremum vectors \hat{F}_1 and \hat{F}_2 .

3.1.3.4.1 Physical programming This method was first developed for *a priori* articulation of preferences [Messac, 1996] but it can be effectively adapted as well for progressive articulation of preferences [Tappeta et al., 2000], or for *a posteriori* articulation of preferences [Messac and Mattson, 2002].

Physical programming method consists in minimising the function $g(x)$ defined in equation (3.18).

$$g(x) = \log_{10} \left(\frac{1}{n_{sc}} \sum_{i=1}^n \bar{g}_i(g_i(x)) \right) \quad (3.18)$$

where:

- g_i represents an objective to be optimised, or a constrained quantity to satisfy,
- \bar{g}_i is a preference function introduced according to the decision-maker preferences, and defined like in the table 3.1. Hard class-functions reflect the presence of constraints, and soft class-functions reflect the preferences settled for objective functions.
- n_{sc} is the number of *soft* criteria.

In Physical programming, some region boundaries are introduced to translate the degrees of desirability of the decision-maker. They are composed of six ranges:

- unacceptable,
- highly-undesirable,
- undesirable,
- tolerable,
- desirable,
- highly-desirable.

An illustration of these regions can be found on the figure 3.3 in the case of the minimisation of the objective function g_i .

The interest in introducing these regions is to compare the evolution of objective functions in a qualitative way. For instance, it can be better for one criterion to travel across the *tolerable* region than for all the others to travel across the *desirable* region.

For each criterion, one must first define the class-type among those described on table 3.1. This is followed by the prescription of scalar values that quantitatively define the preferred behaviour of the criterion via region limits, g_{ik} on the figure 3.3.

The values of the class function \bar{g}_i are the same regardless of class-type or criterion. As a consequence, as one travels across a given region-type, the change in the class-function will

	SOFT	HARD
SMALLER IS BETTER (Class-1)		
LARGER IS BETTER (Class-2)		
CENTER IS BETTER (Class-3)		

Table 3.1: Preference function classification for Physical programming

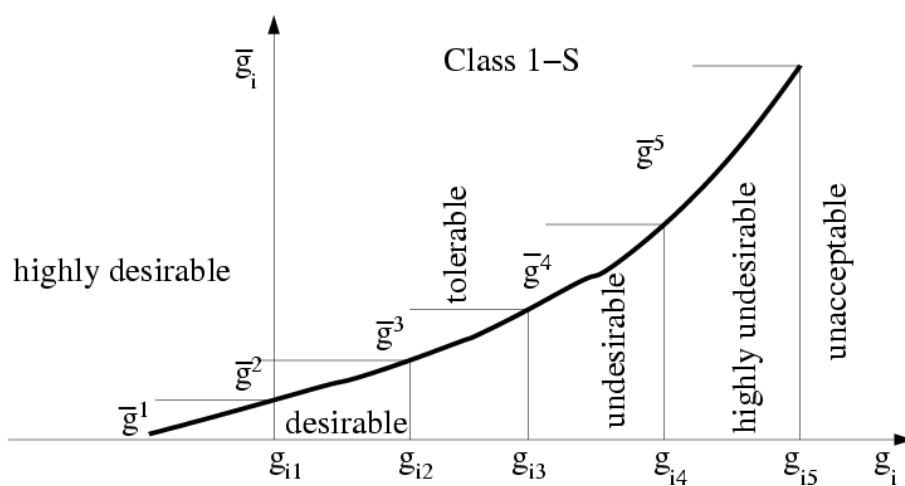


Figure 3.3: Example of preference regions on the class function 1-S

always be of the same magnitude, \bar{g}^k on the figure 3.3. This property makes each generic region-type have the same numerical consequence for different criteria. This same behaviour also has a normalizing effect, and results in favorable numerical conditioning properties [Messac, 1996].

The use of Physical programming in the case of *a posteriori* articulation of preferences is made possible by using pseudo-preferences. Indeed, as the decision-maker has no *a priori* preference, he/she is not intended to give particular scalar values to the variables g_{ik} to define his/her regions of interest. Thus, this method is used in this case to generate points on the Pareto frontier.

Pseudo-preferences are generic numbers generated to represent likely preference values of the decision-maker. Their values vary in the entire objective space. They depend on extrema values of the objective functions, or on the boundaries of the desired region of investigation. Each time pseudo-preferences are fixed, Physical programming computes a point of the Pareto frontier.

The way to generate points of the Pareto frontier using Physical programming is interesting because it enables the decision-maker to have a well-distributed set of points, facilitating the *a posteriori* decision.

To have more informations, the basis principles of Physical programming are described in [Messac, 1996], and more details to use this method to generate well-distributed points on the Pareto frontier are given in [Messac and Mattson, 2002].

An interesting work has been done in Cranfield university, which is described in [Guenov et al., 2005; Utyuzhnikov et al., 2005]. They modify the Physical programming method to make it simpler and more efficient for generating an evenly distributed Pareto set. Their algorithm does not provide non-Pareto solutions while local Pareto solutions may be easily recognised and removed in the framework of this method.

3.1.3.4.2 Normal boundary intersection method The Normal Boundary Intersection method, further denoted NBI, is another method for generating Pareto optimal points of a general nonlinear multicriteria optimisation problem [Das and Dennis, 1998]. NBI is a technique intended to find the portion of the objective set frontier which contains the Pareto optimal points.

Again, to explain the general notions of this method, we call:

- x_i^* a solution of the equation (3.14), and $f_i^* = f_i(x_i^*)$,
- $F_i^* = F(x_i^*)$,
- $F^* = \begin{pmatrix} f_1^* \\ f_2^* \\ \vdots \\ f_k^* \end{pmatrix}$ is the utopia vector.

These vectors are represented on the figure 3.4 for an example in two dimensions.

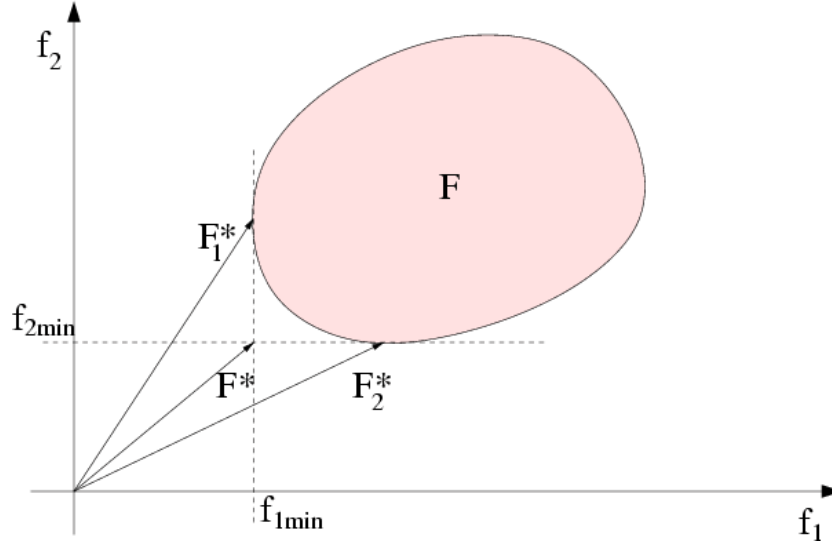


Figure 3.4: Illustration of Normal boundary intersection vectors in two dimensions with no constraint

Then we introduce the matrix $\Phi = [F_i^* - F^*]$, for $i = 1, \dots, k$, which has the following properties:

- $\Phi(i, i) = 0$, for all $i = 1, \dots, k$,
- $\Phi(i, j) > 0$, for all $i, j = 1, \dots, k$ and $i \neq j$.

With this matrix, we define the *Convex Hull of Individual Minima*, *CHIM*:

$$CHIM = \{\Phi\beta : \beta \in \mathbb{R}^k, \sum_{i=1}^k \beta_i = 1, \beta_i \geq 0\}. \quad (3.19)$$

The set of attainable objective vectors, $\{F(x) : G(x) \leq 0, H(x) = 0\}$ is denoted by \mathcal{F} , its boundary by $\partial\mathcal{F}$.

The idea behind the NBI approach is that the intersection points between the boundary $\partial\mathcal{F}$ and the normal vector pointing towards the origin emanating from any point in the *CHIM* is a point on the portion of $\partial\mathcal{F}$ containing the efficient points.

Such boundary points can be found by solving the following optimisation problem:

$$\begin{aligned} \max \quad & t \\ & x, t \\ \text{s.t.} \quad & \Phi\beta + t\hat{n} = F(x) \\ & G(x) \leq 0 \\ & H(x) = 0 \end{aligned} \quad (3.20)$$

where \hat{n} is the unit normal vector to the *CHIM* pointing towards the origin and t is a scalar. The figure 3.5 is an illustration of this process.

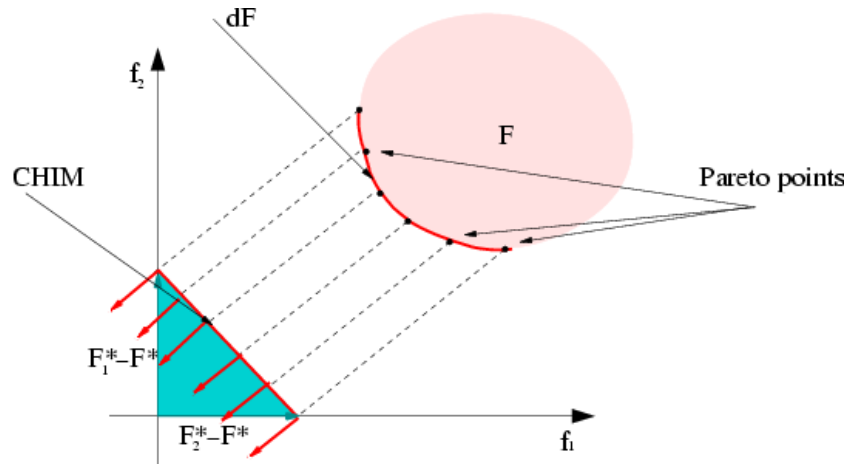


Figure 3.5: Illustration of the method Normal boundary intersection

All NBI points are not Pareto optimal. However, according to [Das and Dennis, 1998], the components of the utopia vector F^* being global minima of the objectives and the convexity of the Pareto surface are sufficient, though far from necessary, conditions for the NBI points to be globally Pareto optimal.

An example of a not Pareto optimal point produced by the NBI method can be found on figure 3.6. This point lies on the “concave part” of the boundary of the objective set.

Nevertheless, these points can be useful since they help in constructing a smoother approximation of the Pareto boundary.

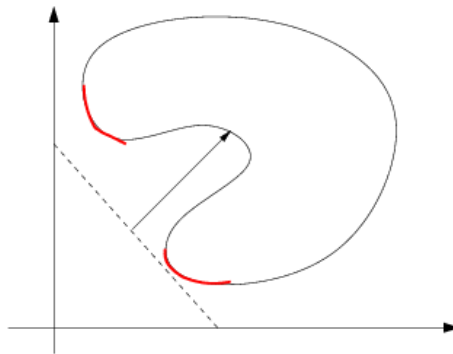


Figure 3.6: Example of not Pareto optimal points produced by the NBI method. The Pareto frontier is represented in red. The “concave part” of the boundary is not part of the Pareto frontier but it can be easily *a posteriori* removed from the results.

3.1.4 Resolution algorithms

After this not so short introduction to some existing methods to solve multicriteria optimisation problems, we now describe some available algorithms based on game strategies, on simulated annealing or on evolutionary algorithms.

3.1.4.1 Non-cooperative multiple objective scheme

Until now, we only considered Pareto optimality and dominance notions to solve our general multiobjective optimisation problem. Pareto ranking and sharing can be considered as a cooperative game which computes the set of non-dominated solutions.

We now introduce other multiple objective schemes, Nash Equilibrium and Stackelberg Equilibrium, which are respectively a non-cooperative and a hierarchical, or competitive, game. These approaches introduced the notion of player and aimed at solving multiobjective optimisation problems originating from Game Theory and Economics [Périaux et al., 2006].

Definition 3.3 (Nash Equilibrium). Nash strategy consists in having k players, each optimising its own criterion while all the other criteria are fixed by the rest of the players. When no player can further improve his criterion, it means that the system has reached an equilibrium state called Nash Equilibrium [Périaux et al., 2006].

As an illustration of this definition, let us consider two criteria and two sets of variables $x \in X \subset \mathbb{R}^n$ and $y \in Y \subset \mathbb{R}^{k-n}$, a strategy pair $(\bar{x}, \bar{y}) \in X \times Y$ is said to be in Nash equilibrium if and only if:

$$\begin{aligned} f_X(\bar{x}, \bar{y}) &= \inf_{x \in X} f_X(x, \bar{y}) \\ f_Y(\bar{x}, \bar{y}) &= \inf_{y \in Y} f_Y(\bar{x}, y). \end{aligned} \quad (3.21)$$

Definition 3.4 (Stackelberg Equilibrium). In a Stackelberg game, there are two kinds of players, the leaders and the followers. The followers have to optimise their criterion using fixed values imposed by the leaders on the variables they manage. Then the leaders get back the results of the follower optimisations, and optimise their criterion using best values coming from the followers. The Stackelberg Equilibrium is achieved when the leaders cannot improve their criterion.

Again, as an illustration of this definition, let us consider two players, let X denote the search space of the leader, and Y the search space of the follower; a Stackelberg equilibrium is characterized as follows:

$$f_X(\bar{x}, \bar{y}) = \inf_{x \in X} f_X(x, \bar{y}^*) \quad (3.22)$$

where f_X denotes the gain of the first player, \bar{y}^* is the solution of the following minimisation problem with respect to the decision variable y :

$$f_Y(x^*, \bar{y}^*) = \inf_{y \in Y} f_Y(x^*, y) \quad (3.23)$$

where f_Y denotes the gain of the second player, and x^* is the design variable value received by the first player.

A Stackelberg game has a non-symmetrical structure with completely different roles of players. During a Nash game, the two players choose their best strategies according to the one decided by the other player to improve their own criterion. The players associated to a Nash game have a symmetric role, while a Stackelberg game has a hierarchical definition of the roles.

If we consider a multiobjective optimisation problem with k criteria, each player manages a subset of the design variables. For the repartition of the variables between the players, *i.e.* which player should optimise which variable, it depends on the structure of the problem. In most real-life problems, the structure of the problem is likely to suggest a way to divide those variables.

Nash and Stackelberg equilibria are very difficult to find with classical approaches [Sefrioui and P eriaux, 2000]. It is generally easier to prove that a given solution is an equilibrium, but exhibiting such a solution may reveal to be a hard task. And it becomes even harder if the criteria are non-differentiable functions. The idea developed in [Sefrioui and P eriaux, 2000; Gonzalez et al., 2005; P eriaux et al., 2006] is to merge game strategy and genetic algorithms to make GA build the equilibria.

In the following, we consider two players to clarify the explanations, but everything is easily applicable to more than two players.

Let X denote the subset of variables handled by *Player-1* and optimised along the first criterion, let Y denote the subset of variables handled by *Player-2* and optimised along the second criterion.

3.1.4.1.1 Merging Nash Equilibrium and GAs The next step consists in building two different populations, one for each player. Let X_{k-1} be the best value found by *Player-1* at generation $k - 1$, and Y_{k-1} be the best value found by *Player-2* at generation $k - 1$. At generation k , the first player optimises its criterion using the values Y_{k-1} and at the same time, the second player optimises its criterion using the values X_{k-1} . After the optimisation process, *Player-1* sends the best value X_k to *Player-2* who will use it at generation $k + 1$, and similarly, *Player-2* sends the best value Y_k to *Player-1* who will use it at generation $k + 1$ [P eriaux et al., 2006].

Nash equilibrium is reached when neither *Player-1* nor *Player-2* can further improve their criteria. The figure 3.7 is an illustration of this process.

3.1.4.1.2 Merging Stackelberg Equilibrium and GAs In this game strategy, we consider that the first player is the leader, and the second player is the follower.

For each individual \bar{X} frozen of the leader's decision, the follower searches the corresponding Y^* to improve his criterion. Once all individuals of the leader's decision set have received the corresponding Y^* values, then the leader changes X to improve his criterion.

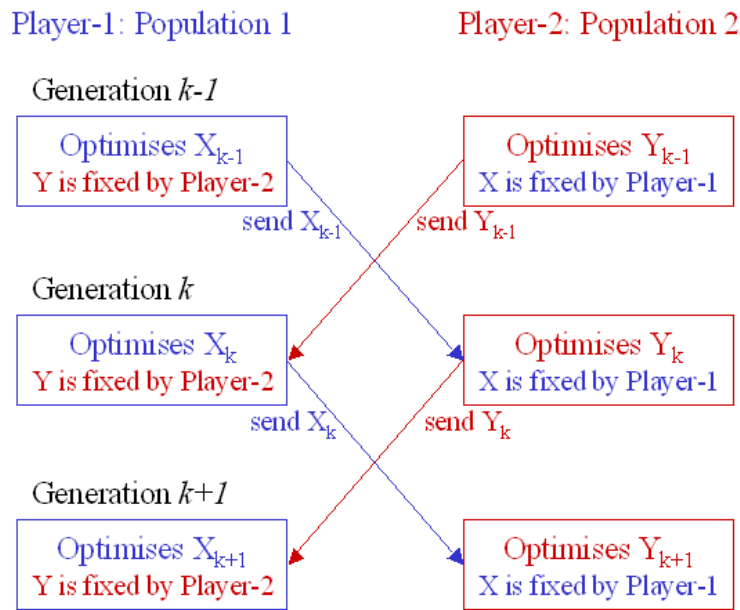


Figure 3.7: Illustration of the merging of Nash Equilibrium and Genetic Algorithms

This process is repeated until the leader could not further improve his criterion. The figure 3.8 is an illustration of this process.

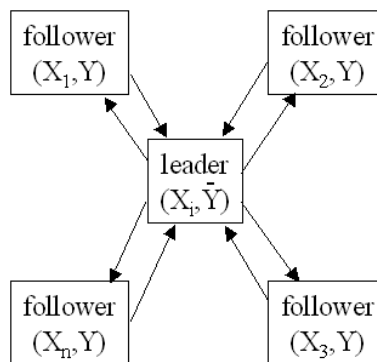


Figure 3.8: Illustration of the merging of Stackelberg Equilibrium and Genetic Algorithms

3.1.4.2 Multi-Objective Simulated Annealing

Simulated Annealing is based upon a metallurgical principle [Collette and Siarry, 2002]: to find an internal arrangement close to perfect crystal, *i.e.* of minimal energy, the temperature is increased and then decreased slowly to give enough time for the atoms to get the regular ordering.

In mathematical words, let T denote the temperature. Starting from a randomly sampled point x_0 , F is evaluated at a point y in its neighbourhood. This difference between the two

evaluations, $\Delta F = F(y) - F(x_0)$, is assessed. If ΔF is negative, y becomes the new starting point, if it is positive, it can also be the new starting point, with the probability $e^{-\frac{\Delta F}{T}}$.

During this process, the temperature decreases step by step until it reaches a given minimum.

To perform multiobjective optimisation using a simulated annealing method, the following can be done:

1. to aggregate the objective functions to build a new criterion G such as:

$$G(x) = \sum_{i=1}^k \ln(f_i(x)). \quad (3.24)$$

The optimisation process is applied directly on the function G such as described previously and it is repeated for a given number of initial points. Each time the process is run, the solutions are compared to the ones obtained previously. Non-dominated solutions are stored in a set of non-dominated solutions, while new dominated points are removed from this set.

This method is called *Pareto Archived Simulated Annealing*, P.A.S.A.

2. to adapt the probability of acceptance of one point, depending on the relative importance of the criteria, by introducing the variable Π_i for each objective function:

$$\Pi_i = \begin{cases} \exp\left(\frac{-\Delta f_i}{T_n}\right) & \text{if } \Delta f_i > 0 \\ 1 & \text{if } \Delta f_i < 0. \end{cases} \quad (3.25)$$

Then the probability of acceptance is defined through the formula:

$$p = \prod_{i=1}^k \Pi_i^{w_i}, \quad (3.26)$$

where w_i are the weights of the aggregate function f_{eq} to be minimised:

$$f_{eq} = \sum_{i=1}^k w_i f_i(x) \quad (3.27)$$

This method is called *Multiple Objective Simulated Annealing*, M.O.S.A.

3.1.4.3 Multi-Objective Evolutionary Algorithms (MOEA)

It was recognized early in their development that Evolutionary Algorithms, further denoted EAs, were possibly well-suited to multiobjective optimisation [Fonseca and Fleming, 1995]. Multiple individuals can search for multiple solutions in parallel, eventually taking advantages of any similarities available in the solution set. The ability to handle complex problems,

involving features such as discontinuities, disjoint feasible spaces and noisy function evaluations, reinforces the potential effectiveness of EAs in multiobjective search and optimisation.

The general principle of genetic algorithms has been described in the section 2.2.1 page 75. The method we now describe shortly can be found in the following overviews of genetic algorithms in multiobjective optimisation: [Fonseca and Fleming, 1995; Coello Coello, 2000; Van Veldhuizen and Lamont, 2000; Coello Coello, 2001; Deb, 2001]

3.1.4.3.1 Multiple Objective Genetic Algorithm (MOGA) This method has been developed by Fonseca and Fleming [Fonseca and Fleming, 1993]. The approach consists in a scheme in which the rank of an individual is given by the number of individuals that dominate the considered point. For instance, if the number of points which dominate the considered individual x_i is $p_i^{(t)}$, then we have:

$$\text{rank}(x_i, t) = 1 + p_i^{(t)}.$$

All non-dominated individuals are assigned rank 1.

Fitness value is calculated following these steps:

1. Sort population according to rank,
2. Assign fitness to individuals by interpolating from the best (rank 1) to the worst (rank n) according to some function, usually affine like in the figure 3.9.

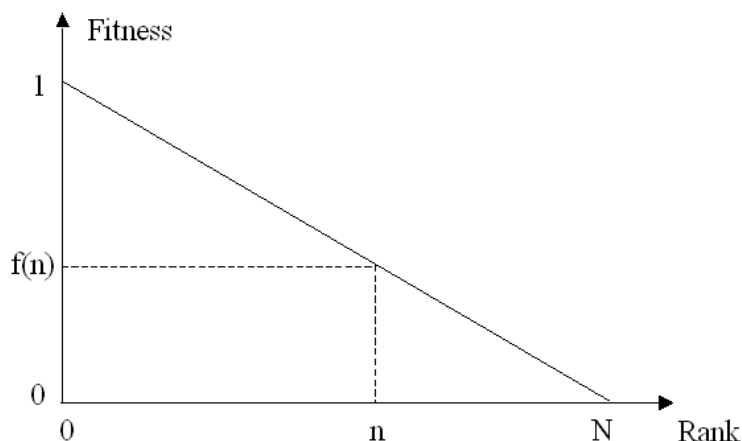


Figure 3.9: Example of interpolation function to calculate the fitness in the MOGA method

More details can be found in the algorithm 3.1.

Algorithm 3.1. MOGA

```

Initialisation of the population
Evaluation of objective functions
Rank affectation based on dominance
Fitness affectation based on rank
For  $i = 1$  to  $G$ 
    Random selection proportional to the fitness values
    Crossover
    Mutation
    Evaluation of objective functions
    Rank affectation based on dominance
    Fitness affectation based on rank
End For

```

3.1.4.3.2 Niched Pareto Genetic Algorithm (NPGA) The specifics of this method are the implementation of the selection process [Horn et al., 1994]. Instead of having a classic tournament selection, which consists in keeping for the next generation the best individual among a sub-group of the population, the selection is based on Pareto dominance tournaments.

In a classic tournament selection, the population generally converges towards a uniform solution. The Pareto dominance tournament selection is used to maintain multiple Pareto optimal solutions. Two candidates for selection are picked at random from the population. A set of individuals is also picked, for comparison, randomly from the population, its size is denoted t_{dom} in the literature. Each candidate is then compared against each individual in the comparison set. If one candidate is dominated by the comparison set, and the other is not, the latter is selected for reproduction. If neither or both are dominated by the comparison set, then we must use sharing to choose a winner.

Fitness sharing is used to distribute the population over a number of different peaks of the search space [Horn et al., 1994]. Each peak receives a fraction of the population in proportion to the height of this peak.

This distribution is obtained by degrading an individual's fitness f_i by a niche count m_i . This degradation is obtained by simply dividing the fitness by the niche count to find the shared fitness: $\frac{f_i}{m_i}$. The niche count m_i is an estimate of how crowded is the neighbourhood, or niche, of the individual i . It is calculated over all individuals in the current population:

$$m_i = \sum_{j \in Pop} Sh(d(i, j))$$

where:

- $d(i, j)$ is the distance between individuals i and j ,
- $Sh(d)$ is the *sharing function*. It is a decreasing function of $d(i, j)$ such that $Sh(0) = 1$ and $Sh(d \geq \sigma_{share}) = 0$, σ_{share} being the minimal separation desired or expected

between the goal solutions. Typically, the sharing function is a piecewise affine function, like on the figure 3.10.

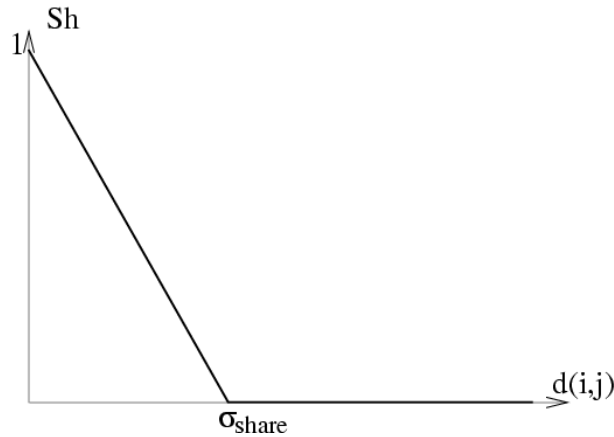


Figure 3.10: Representation of the NPGA sharing function

Another example of the calculation of the niche count can be seen in paragraph 2.3.4.2.3, page 88, concerning the fitness sharing evaluation of the method NSGA, with a different sharing function than the triangular.

More details can be found in the algorithm 3.2.

Algorithm 3.2. NPGA

Initialisation of the population

Evaluation of objective functions

For $i = 1$ to G

 Pareto dominance tournament selection

 If candidate 1 is dominated, candidate 2 is selected

 If candidate 2 is dominated, candidate 1 is selected

 If both candidates are dominated:

 Perform sharing fitness

 Candidate with smaller number of individuals
 in its neighbourhood is selected

 Crossover

 Mutation

 Evaluation of objective functions

End For

The convergence behaviour of this method strongly depends on the parameters t_{dom} and σ_{share} , which have to be well-tuned according to the optimisation problem. Explanations and discussions on this topic can be found in [Coello Coello, 2000; Fonseca and Fleming, 1993; Horn et al., 1994] for both NPGA and MOGA methods.

3.1.4.3.3 Strength Pareto Evolutionary Algorithm (SPEA) SPEA uses a mixture of established and new techniques in order to find multiple Pareto-optimal solutions in parallel [Zitzler and Thiele, 1999]. It combines the three following techniques in a single algorithm:

- Non-dominated solutions found so far are stored in an external population,
- Pareto dominance is used in order to assign scalar fitness values to individuals,
- Clustering is performed to refine the number of non-dominated solutions stored without destroying the characteristics of the trade-off front.

The fitness assignment is a two-stage process. First the individuals in the external non-dominated set P' are ranked. Afterwards, the individuals in the population P are evaluated:

1. Each solution $i \in P'$ is assigned a real value $s_i \in [0, 1)$, called *strength*. s_i is proportional to the number of individuals $j \in P$ for which i dominates j . Let n denote this number, then the fitness f_i is equal to $\frac{n}{N+1}$, where N is the size of the population.
2. The fitness of an individual $j \in P$ is calculated as follows:

$$f_j = 1 + \sum_{i \text{ dominating } j} s_i.$$

The figure 3.11 is an illustration of the calculation of the fitness of the individuals in two examples.

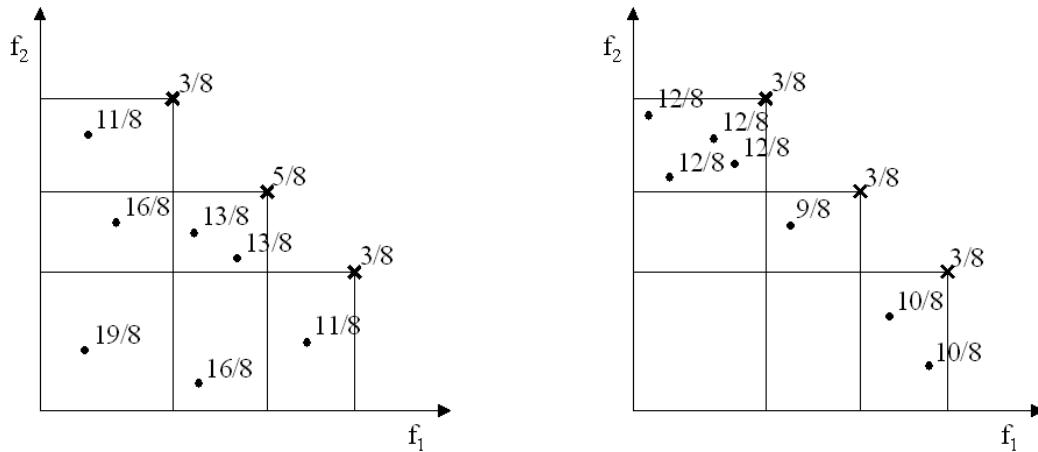


Figure 3.11: Examples of SPEA fitness assignment: Individuals represented with a cross are non-dominated points of the set P' . Individuals represented with a dot are points of the current population P . All the numbers on these figures are the fitness of the corresponding individuals.

More details can be found in the algorithm 3.3.

Algorithm 3.3. SPEA

```

Initialisation of the population  $P$ 
Creation of an empty external non-dominated set  $P'$ 
For  $i = 1$  to  $G$ 
    Copy non-dominated points of  $P$  to  $P'$ 
    Remove solutions within  $P'$  which are dominated by any other member of  $P'$ 
    If the number of externally stored non-dominated solutions exceeds a given
    maximum, prune  $P'$  by means of clustering
    Calculate the fitness of each individual in  $P$  as well as in  $P'$ 
    Select individuals from  $P + P'$  using a binary tournament
    Crossover
    Mutation
End For

```

For more explanations on the SPEA method, refer to [Zitzler and Thiele, 1999; Sbalzarini et al., 2000].

3.1.4.3.4 Hierarchical Asynchronous Parallel Evolutionary Algorithms (HAPEA) The HAPEA method is described in [Périaux et al., 2006; Gonzalez et al., 2005]. It has two main characteristics:

1. it is a hierarchical algorithm,
2. the calculations are paralleled asynchronously, which is effective in calculation time because any machine does not have to wait for all the population point evaluations before starting a new calculation.

Hierarchical genetic algorithms use multi-populations on different granularity of models of the same physics. If we consider a three level hierarchical algorithm, we have:

1. The bottom layer is devoted to the exploration of the design space. Thus, the algorithm can make large leaps in the search space.
2. The intermediate layer is a compromise between exploration and exploitation.
3. The top layer concentrates on refining solutions. This can be achieved by tuning the algorithm in a way that takes very small steps between successive points.

The main feature is the interaction between the given layers. The best solutions progress from the bottom layer to the top layer where they are refined.

Bottom layer can use a less accurate, faster model to compute the fitness functions of the individuals. Even though these solutions may be evaluated rather roughly, the hierarchical topology allows their information content to be used. During the migration phase, they are re-evaluated by more precise models until reaching the top layer.

The figure 3.12 is an illustration of the hierarchical concept and the interaction between the different layers.

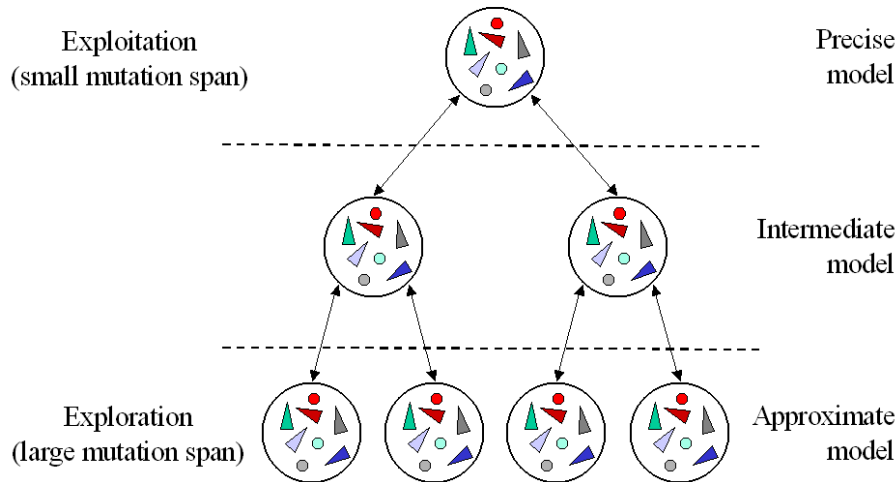


Figure 3.12: Hierarchical topology of the algorithm HAPEA

Parallelisation strategies are common with genetic algorithms. In the case of the HAPEA algorithm, solvers do not need to run at the same speed. This is allowed by the modification of the population evaluation, which is usually performed simultaneously for each individual of the population. The distinctive method of an asynchronous approach is that it generates only one candidate solution at a time and only re-incorporates one individual at a time, rather than an entire population at every generation.

Applications of HAPEA are described in [Gonzalez et al., 2005; Gonzalez et al., 2006] on the design and the optimisation of UAV, Unmanned Air Vehicle, systems.

3.1.4.3.5 Tests suites Various and numerous MOEA implementations and applications are published in scientific papers. [Deb, 1999; Van Veldhuizen and Lamont, 1999] define a set of test problems with known and controlled difficulty measure for systematically testing the performance of an optimisation algorithm. They also show procedures for constructing multiobjective test problems and offer a methodology for quantitatively comparing MOEA performances.

According to [Deb, 1999], there are two tasks that a MOEA should accomplish in solving multiobjective optimisation problems:

1. guide the search towards the global Pareto-optimal region,
2. maintain population diversity (in the function space, parameter space or both) in the current non-dominated front.

Discussions about these two tasks can be found in [Deb, 1999] and in [Van Veldhuizen and Lamont, 1999]; an explicit description of test functions can be found in [Collette and Siarry, 2002].

3.1.4.4 Other multiobjective programming methods

There are other genetic algorithms used to solve multiobjective problems, like:

- Vector Evaluated Genetic Algorithm, VEGA [Schaffer, 1985],
- Non dominated Sorting Genetic Algorithm, NSGA [Srinivas and Deb, 1994; Deb et al., 2000] (already described in paragraph 2.3.4.2.3 page 87)
- Multiple Objective Particle Swarm Optimization, MOPSO [Coello Coello and Lechunga, 2002], ...

There are further methods to solve multiobjective optimisation problems, using evolutionary algorithms or not. For instance:

- Taboo search. This method consists in evaluating a set of points in the neighbourhood of the current point x_n and to keep the point which minimises the objective function for the next step. Taboo search allows to keep a point x_{n+1} such that $f(x_n) < f(x_{n+1})$ but x_{n+1} has to be different from previously selected points, $x_{n+1} \neq x_k$, for $k = n, \dots, n-i$, i being generally equal to 7 [Collette and Siarry, 2002].
- Differential Evolution, DE [Rai, 2006a]. DE is a population-based method for finding global optima. As with other evolutionary algorithms, the three main ingredients are mutation, recombination and selection. The main characteristic of this method is the mutation operator. Mutations are obtained by computing the difference between two randomly chosen parameter vectors in the population, and adding a portion of this difference to a third randomly chosen parameter vector to obtain a candidate vector. Thus, the mutation operator adapts to the particular objective function, and this results in an automatically reduction of the magnitude of mutation as the optimisation process converges.
- Adjoint approaches. [Gauger, 2006] uses this method in aerodynamic shape optimisation, which is a single discipline optimisation, and in an aero/structure test case as an illustration of an MDO study. But this method has not been extended to more than two disciplines in the quoted paper.

After this introduction to different methods and algorithms existing to solve a multiobjective optimisation problem, **we now describe the multiobjective optimisation problem that FPO engineers are dealing with, and the way we treat it using our dedicated method.**

3.2 Conflicting objectives in our problem

3.2.1 Aircraft sizing objectives

The multiple objectives in aircraft sizing studies were described in section 1.1.2 page 15 as a measure of the aircraft quality.

MTOW has always been a significant parameter to consider to represent the aircraft quality because it has a direct impact on costs. But with the increase of fuel price, MTOW is not as relevant as it used to be.

Currently, DOCs or COC, Cash Operating Cost, are considered as more representative of the aircraft quality because they integrate in the same variable the impact of fuel cost, maintenance, crew or taxes. A detailed definition of the DOC is given in section 1.1.2 page 17.

One of the aims of our work was to perform multiobjective optimisation with no *a priori* articulation of preferences, as it is currently done with a DOC. Indeed, with the DOC, the multiobjective optimisation method used in FPO is a weighted sum method. Thus, one of the aims is to implement an *a posteriori* articulation of preferences. The objectives we want to consider are for instance:

- MWE, Manufacturer Weight Empty,
- MTOW,
- Fuel consumption,
- Maintenance costs (of components and engines),
- Depreciation,
- Taxes (landing taxes and traffic taxes),
- Return on investment, etc.

The four last items of objectives listed above, related to economics, are not available in the models of aircraft we developed and on which we tested our methodology. Thus, we now describe the different objectives we considered during our implementation work of a method with *a posteriori* articulation of preferences.

3.2.2 Conflicting objectives in our models

3.2.2.1 When using the USMAC model

The optimisation problem treated with the USMAC model has already been described in paragraph 2.4.1.1.3 page 94. To summarize, there are 4 criteria to minimise:

- MTOW, the maximum take-off weight (kg),

- Fuel, the nominal fuel (kg),
- MWEoMTOW, the weight efficiency, *i.e.* the ratio of MWE, the manufacturer weight empty, over the MTOW (no dimension)
- FoPoNe6, the fuel efficiency, *i.e.* the ratio of the fuel over the number of passengers over the range (10^6kg/m/Pax).

The figure 2.19 page 97 represents the iso-curves of the criteria according to the two main degrees of freedom of the USMAC.

In the previous chapter, we performed a robust mono-criterion optimisation using GA to refine the search space to the admissible space, and then we used FSQP to perform the optimisation.

Thanks to these first results, we noticed that the optimum of the four criteria were all located at the intersection of two constraints (see on figure 2.24 page 102). Thus, there was no interest in performing multicriteria optimisation with these four criteria, because there is no compromise to define.

As no other criterion of the previous list was available with this model, we decided to introduce the approach speed as a new criterion, even if it is not used in practice as a criterion. The technical interest in introducing the approach speed as a criterion is to be able to land on a short landing field length, like for instance for military aircraft, or business jet willing to land in London City.

3.2.2.2 When using the SMAC model

The criteria used with the SMAC model are also described in paragraph 2.4.5.1 page 103. There are 4 criteria to minimise:

- MTOW, the Maximum Take-Off Weight (kg),
- Cash_cost, the Cash Operating Cost (dollars per passenger per nautical mile),
- MWE, the Manufacturer Weight Empty (kg),
- OWE, the Operational Weight Empty (kg).

When dealing with our first model of aircraft, the one contained in *AVION*, the first problem we faced was the huge design space, which is a sixteen-dimension hypercube. Thus, our methodology to perform robust mono-criterion optimisation was to refine the search space to the admissible space.

What we intend to do now is to apply the same kind of methodology to refine once more the design space to what we call the acceptable domain.

3.3 Refinement of the admissible set

3.3.1 Introducing the acceptable domain

Our aim is to define a method to refine the design space to its most interesting part, increasing chances to find best aircraft configurations.

Our method is composed of three steps:

1. First, find the admissible set, because it is difficult to produce feasible points inside the design space. In a real case study, the proportion of admissible points is about 5 among 1 000 000 points.
2. Then introduce a new kind of constraints, named quality constraints, which are actually acceptable margins applied on each criterion, and based on the optimal values of these criteria through acceptable degradation ratios.
3. Finally refine the admissible set to an acceptable domain, where not only operational constraints are satisfied but also quality constraints.

We are thus progressing step by step to reach smaller and smaller subspaces contained in the original search space, the final aim being to find the Pareto front, which most interesting part is included in the acceptable domain.

The first step of this method was the topic of the second chapter. We implemented a genetic algorithm to solve the constraints satisfaction problem related to our aircraft sizing problem. We focused on the mutation operator of the GA to make the design points of the population gather step by step in the requested domain, *i.e.* the admissible set.

The second objective of the chapter 2 was to perform a robust optimisation. Thanks to FSQP, we get the optimum values of each of our criteria, optimised separately.

3.3.2 Method to refine the admissible set

To refine the admissible set to a smaller set, we decided to introduce the notion of quality of the results. Quality is based on allowed degradation of the best values of criteria that can be found inside the admissible set. Quality is thus based on the decision-maker needs, wishes and experiments.

Let C_i^{max} denote the best value of the i^{th} criterion, and R_i denote the degradation ratio, R_i^{max} being the maximum degradation value allowed for the i^{th} criterion. The degradation ratio is defined as in the formula (3.28):

$$R_i = \frac{|C_i^{max} - C_i|}{C_i^{max}} \leq R_i^{max} \quad (3.28)$$

where C_i is the current value of the i^{th} criterion.

The relation (3.28) is similar to a constraint equation in an optimisation problem. We want the degradations of the values of each criterion of the current point to be less than the upper bound given by R_i^{max} . The quality of a point depends on the position of the values of its criteria related to the values of each degradation ratio. Thus, we can consider that a new kind of constraints has been added to our initial problem, and we call these constraints quality constraints.

Definition 3.5 (Acceptable Domain). What we called the Acceptable Domain is defined by the subset of the admissible set where degradation ratios are less than R_i^{max} , for all criteria, *i.e.* where quality constraints are satisfied.

The figures 3.13 and 3.14 are illustrating quality constraints based on different criteria. In the first test case (on figure 3.13), the four optima are located at the same point. But in a more general case (like on figure 3.14), the optima are not obtained at the same point. Thus a compromise has to be defined.

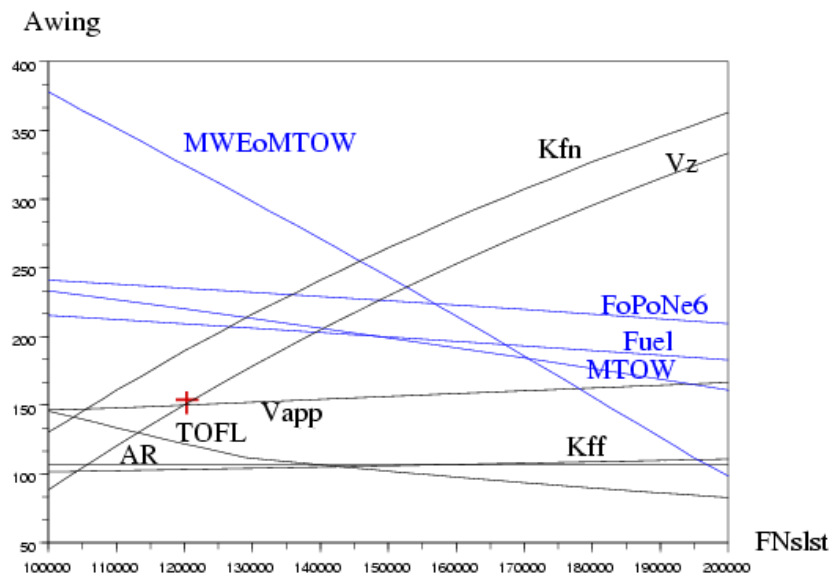


Figure 3.13: Illustration of quality constraints. The black lines are the operational constraints, the blue ones are the quality constraints. In this example, the four optima are located at the red point.

Remark 3.1. A part of the Pareto front is included in the boundary of the acceptable domain.

Remark 3.2. The definition of the acceptable domain requires that the optimal values of each criterion be known.

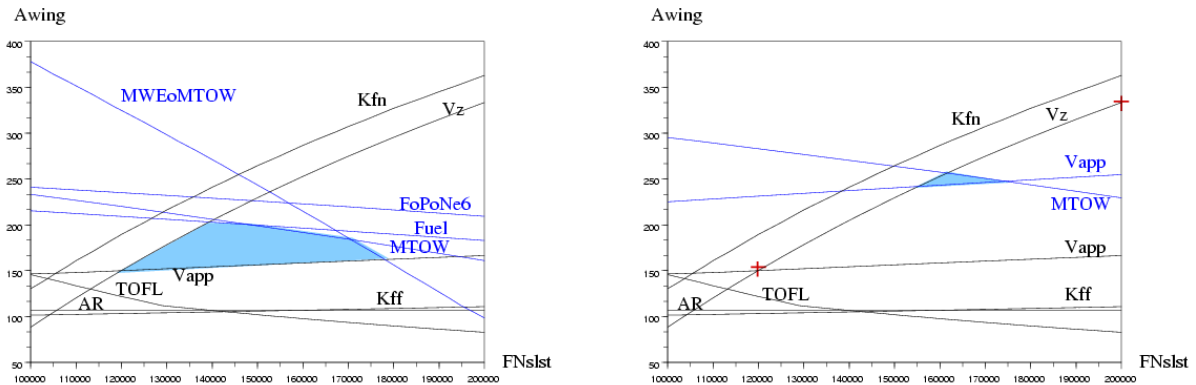


Figure 3.14: Two examples of an acceptable domain. The black lines are the operational constraints, the blue ones are the quality constraints and the red points are the location of the optimal points for each criterion. In these examples, the acceptable domain is the polygon in blue.

All this work has been presented in [Badufle et al., 2006a].

3.4 Pareto front

The Pareto front is the last subset of the admissible domain we want to reach. It contains all the Pareto points, as they are defined in the definition 3.2, page 113. It is the final answer of our multicriteria optimisation problem.

3.4.1 The selected method

To produce some points on the Pareto front, we decided to keep on working with the GA we implemented, NSGA, with some modifications to adapt it to our problem. The basic principles of the NSGA method are described in section 2.2.1 page 75.

We now give some general principles on the adaptations we made on this algorithm to produce the Pareto front of our aircraft sizing problem, refined to the domain which is of our interest, the acceptable domain.

3.4.1.1 The ranking function

The ranking function is used to classify the points according to their Pareto rank, *i.e.* if they are non-dominated, their rank is the smallest, it is assigned to 1, but if they are dominated, their rank is higher than 1. To calculate it, non-dominated points are removed from the current population, and then, the non-dominated points among the remaining points have their rank assigned to 2, and so on.

In our problem, ensuring constraint satisfaction is a complex task, especially with the introduction of the quality constraints. Thus, ranking is performed in two steps. The population is split into two subpopulations, depending on constraints. If one point satisfies all the operational and quality constraints, it is included in the first population, if it does not satisfy one of these constraints, it is included in the second population.

Then, ranking is performed as described above, the best rank is set to the non-dominated points of the population which satisfy all the constraints, and the non-dominated points of the second population have their rank assigned to the last rank value of the first population, plus 1.

The other ranking function, the fitness function, is the same as already described in section 2.2.1 page 75. It depends on the distance between points in the population.

3.4.1.2 The mutation function

As we know the optimal values of each criterion, we decided to influence the mutation operator by imposing a direction to the mutation. Thus, the resulting point is obtained as follows:

$$Mutated_Point = Original + \frac{\alpha}{\sum_{i=1}^k \beta_i} \sum_{i=1}^k \beta_i \left(Optimum_i - Original_i \right) \quad (3.29)$$

where

- *Mutated_Point* is the point produced by the mutation operator,
- *Original* is the point which mutates,
- α and β_i for $i = 1, \dots, k$ are randomly sampled scalars, their values vary between 0 and 1,
- $Optimum_i$ are the optimal points found with FSQP at each criterion.

Dividing by the sum of β_i , for $i = 1, \dots, k$, ensures that the movement is contained in the convex hull defined by the original point and the extremum points given by the optimisation run with FSQP.

3.4.1.3 The selection function

At each generation of the process, the currently produced non-dominated points are saved in another population containing already produced non-dominated points. The new ones are compared to the others, and only the non-dominated points of this pool will remain for the next generation. This method is inspired by the SPEA algorithm, which is described in paragraph 3.1.4.3.3, page 134.

Another condition is added in the selection function. One point will remain in the non-dominated population if it is at a distance to its neighbours greater than the minimum one imposed by the decision-maker. This condition ensures that the points of the Pareto front are well-distributed in the design space.

3.4.2 Test problems

To assess the capability of our adaptation of the NSGA algorithm to produce Pareto points, we decided to test it on some of the test problems we found in the literature, like in the papers already mentioned in paragraph 3.1.4.3.5, page 136.

We decided to test the algorithm on three typical multicriteria problems, which are bi-objective problems with 2 degrees of freedom. The first one is a classical multiobjective optimisation problem, where the Pareto front is convex. Thus, it enables us to see if our algorithm can produce a well-distributed set of points of the front.

The second problem is a more difficult test case where the Pareto front is not convex. It enables us to know if our algorithm can find Pareto points of any kind of multiobjective optimisation problem, where the shape of the Pareto front has no particular property, like convexity.

The last one is an optimisation problem where the Pareto front is not connected. Once again, this problem enables us to assess the robustness of our algorithm in finding the Pareto front, whatever shape it has.

3.4.2.1 A classical convex front

The first test problem is to find a convex front by minimising the two following objectives:

$$\min_{x_1, x_2 \in [-5; 10]} \begin{cases} f_1(x) = x_1^2 + x_2^2 \\ f_2(x) = (x_1 - 5.0)^2 + (x_2 - 5.0)^2 \end{cases} \quad (3.30)$$

The exact solution of this problem can be seen on figure 3.15.

The graphical result of the numerical optimisation of this problem can be seen on figure 3.16. The calculations were stopped after 100 generations. We see that the points on the Pareto front are quite well-distributed, and cover almost all the objective space. Thus, we observe that our algorithm is able to produce well-distributed Pareto points.

3.4.2.2 A non-convex front

The second test problem consists in finding a non-convex Pareto front, because we know that some multiobjective optimisation methods can fail finding a non-convex Pareto front, like the weighted sum methods. This optimisation problem was defined by Deb in [Deb, 1999]:

$$\max_{x_1, x_2 \in [0; 1]} \begin{cases} f_1(x) = f(x_1) \\ f_2(x) = g(x_2) \cdot h(f, g) \end{cases} \quad (3.31)$$

where:

$$f(x_1) = 4 \cdot x_1$$

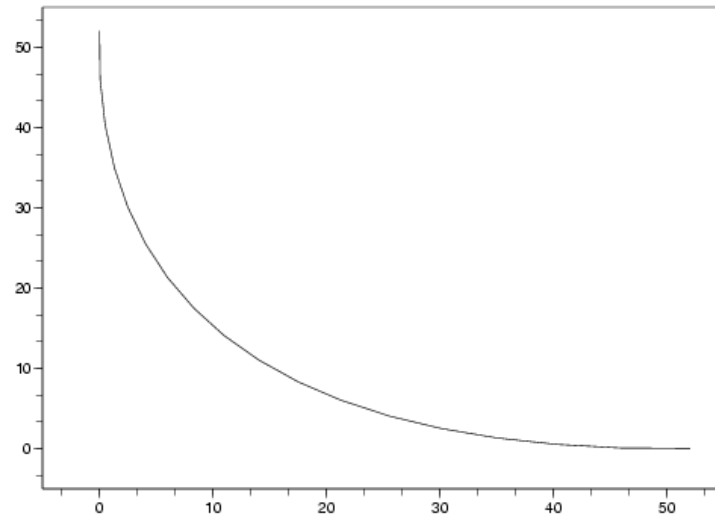


Figure 3.15: Analytical solution of the convex Pareto front test case

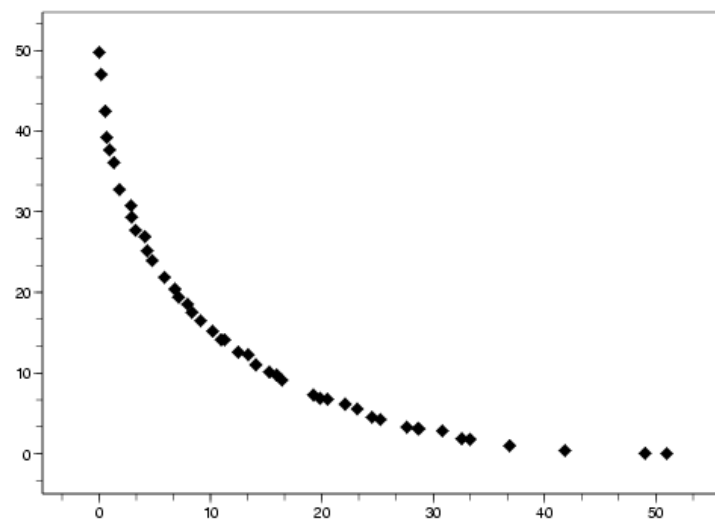


Figure 3.16: Experimental result for the convex Pareto front test case

$$g(x_2) = \begin{cases} 4 - 3 \cdot \exp\left(-\left(\frac{x_2 - 0.2}{0.02}\right)^2\right) & \text{if } 0 \leq x_2 \leq 0.4 \\ 4 - 2 \cdot \exp\left(-\left(\frac{x_2 - 0.7}{0.2}\right)^2\right) & \text{if } 0.4 \leq x_2 \leq 1 \end{cases}$$

and

$$h(f, g) = \begin{cases} 1 - \left(\frac{f}{\beta \cdot g}\right)^\alpha & \text{if } f \leq \beta \cdot g \\ 0 & \text{otherwise} \end{cases}$$

with

$$\alpha = 0.25 + 3.75 \cdot (g(x_2) - 1) \text{ and } \beta = 1$$

The exact solution of this problem can be seen on figure 3.17.

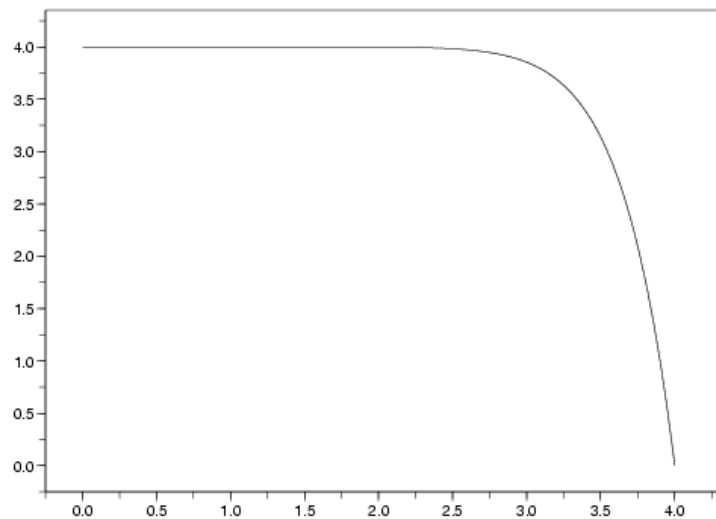


Figure 3.17: Analytical solution of the non-convex Pareto front

The graphical result of the optimisation of this problem can be seen on figure 3.18. Again, the calculations were stopped after 100 generations. We observe that the points on the Pareto front are quite well-distributed, and cover almost all the front.

In conclusion with this test case, we observe that our algorithm is able to produce Pareto points of a non-convex front.

3.4.2.3 A disconnected front

The third test problem is a biobjective optimisation problem where the Pareto front is not connected. According to [Collette and Siarry, 2002], most of the multiobjective optimisation methods are unable to manage disconnected Pareto fronts. Thus we want here to assess

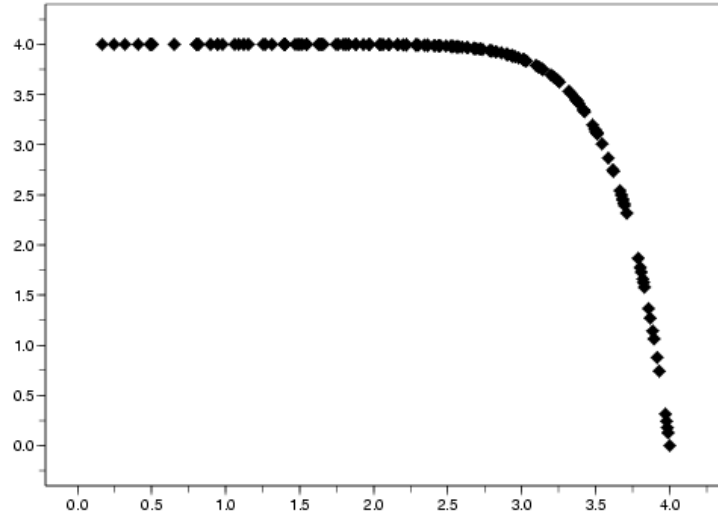


Figure 3.18: Experimental result for the non-convex Pareto front coming from a multiobjective optimisation problem defined by Deb.

the robustness of our algorithm. This problem is also one from the test case proposed by Deb [Deb, 1999]:

$$\min_{x_1, x_2 \in [0; 1]} \begin{cases} f_1(x) = f(x_1) \\ f_2(x) = g(x_2) \cdot h(f, g) \end{cases} \quad (3.32)$$

where:

$$f(x_1) = x_1$$

$$g(x_2) = 1 + 10 \cdot x_2$$

and

$$h(f, g) = 1 - \left(\frac{f}{g}\right)^\alpha - \left(\frac{f}{g}\right) \cdot \sin(2\pi \cdot q \cdot f)$$

with

$$\alpha = 2 \text{ and } q = 5.$$

The parameter q defines the number of disconnected regions on the front.

The analytical solution of this problem is represented on figure 3.19, with the Pareto front in red; the graphical result of the optimisation process on this problem can be seen on figure 3.20.

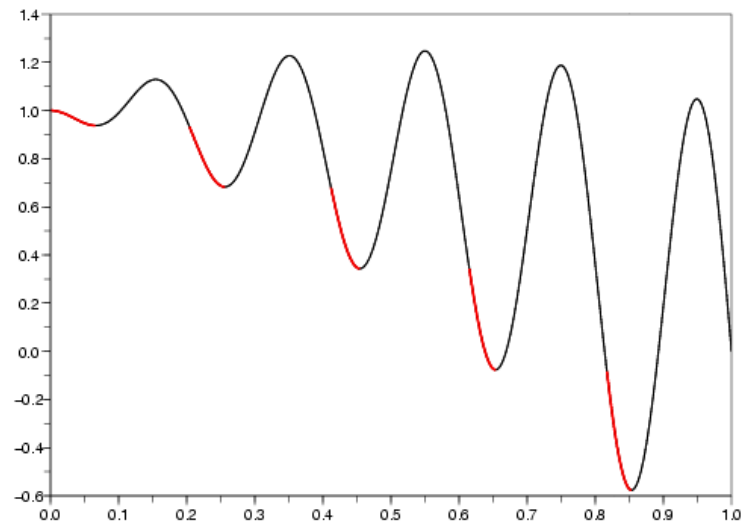


Figure 3.19: Analytical solution of the disconnected Pareto front problem

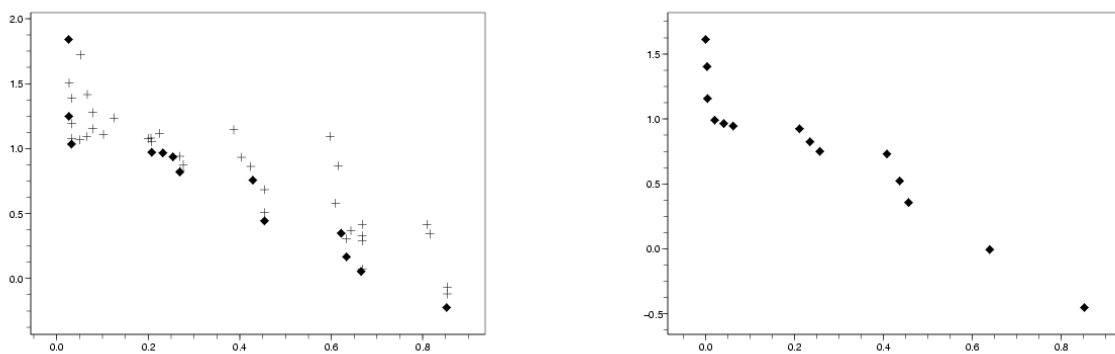


Figure 3.20: Experimental results for a disconnected Pareto front. The Pareto points are the bold diamonds.

On the figure 3.20, the left-part is a representation of the evolution of the front during the optimisation process. The Pareto points are represented by bold diamonds. The figure on the right represents the Pareto points resulting from the optimisation. We see that our algorithm manages to find the five regions of the Pareto front, which is the main difficulty related to this problem. Points are not well-distributed, but the conclusion is that our algorithm is robust enough to treat this kind of problem.

3.5 Results

After satisfactory results for our algorithm to find discontinuous Pareto fronts, we can use it on our aircraft sizing problem.

We still work on the aircraft sizing problem where we have two objectives, the maximum take-off weight and the approach speed.

The first result is obtained using the USMAC model with two degrees of freedom, the sea level static reference thrust, FN_{slst} , which varies between 10000 and 20000 daN, and the wing area, A_{wing} , which varies between 40 and 400 m². The figure 3.21 is a graphical representation of one of the results we obtained.

We see that the points are well-distributed along the front, but some of them do not reach exactly the Pareto front. The main reason is that a minimal distance between points in the design space has been imposed and the calculations were stopped after 100 generations. But it is not a problem that all the points do not reach exactly the Pareto front. Indeed, it brings some additional information on the neighbourhood of the Pareto front.

The second result we show is obtained using the SMAC model with two degrees of freedom, a rubbering coefficient of the engine size, denoted $krub$, which varies between 0.5 and 2, and the wing area, denoted $wing_area$, which varies between 350 and 600 m². The figures 3.22 and 3.23 are graphical representations of one of the results we obtained. The black lines still represent the operational constraints, the blue lines represent the quality constraints.

Again, on the figure 3.22, we can see that the points are quite well-distributed along the Pareto front, but some of them do not reach exactly the front, which is on the boundary of the admissible set near the red points on the figure. The main reason is the same as with the USMAC model: a minimal distance has been imposed between points in the design space, and the calculations were stopped after 100 generations. This time, using the SMAC model, the calculation of the 100 generations lasts 18.5 hours of CPU on a bi-pentium M at 1.8 GHz. Thus, some improvement has to be done on the imposed minimal distance to get a more homogeneous front, because we cannot afford spending more time in calculating more generations.

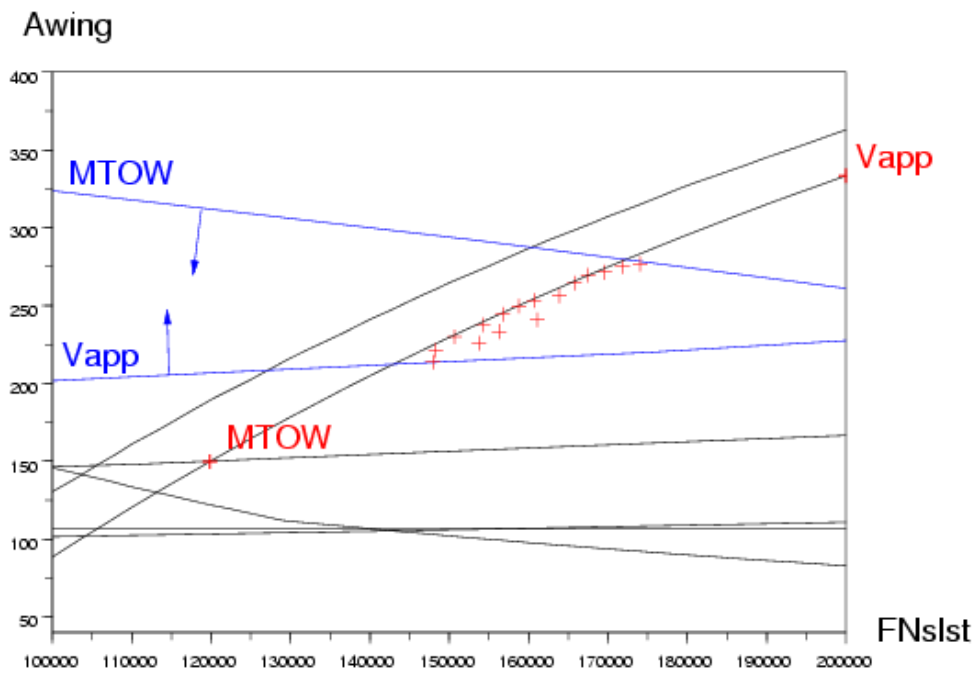


Figure 3.21: Experimental result for the Pareto front using the USMAC model. The black lines represent the operational constraints, the blue lines represent the quality constraints and the blue arrows the descent direction. The acceptable domain is located between the two quality constraints, one operational constraint and the upper bound of the abscissa. The red points are Pareto points produced by our adaptation of the algorithm NSGA. The real Pareto front is the part of the operational constraint near the red points and contained between the blue lines.

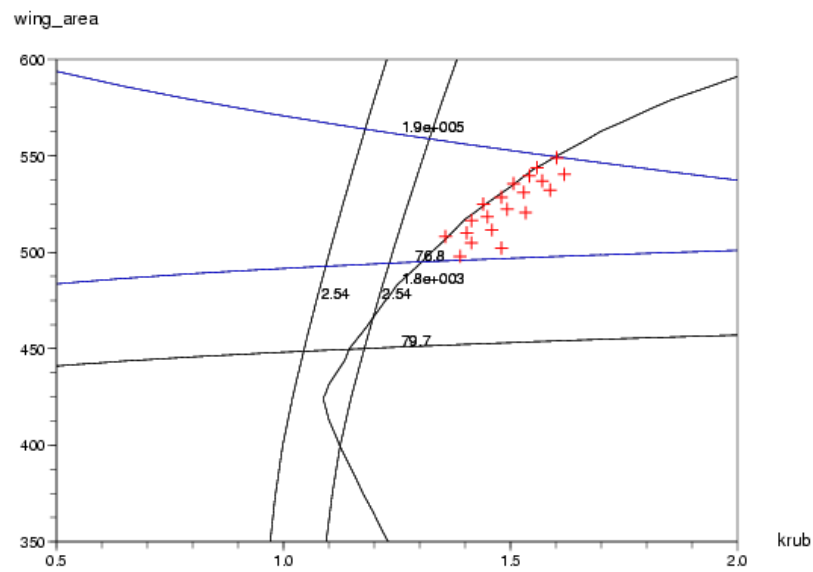


Figure 3.22: Pareto front in the design space using the SMAC model. The black lines still represent the operational constraints, the blue lines represent the quality constraints.

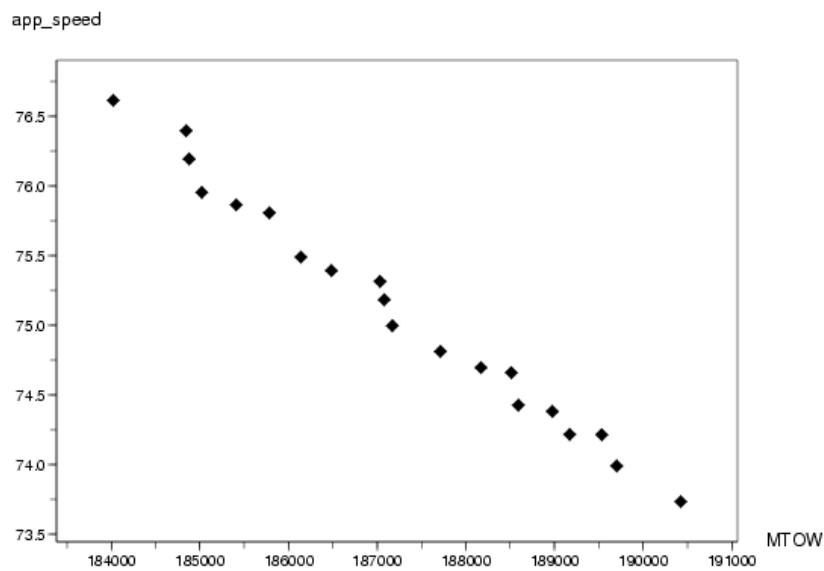


Figure 3.23: Experimental result for the Pareto front in the objective space using the SMAC model

3.6 Conclusion

As for a conclusion of this chapter, we say that we managed to extract some points of the search space that enjoy the following properties:

- they satisfy the constraints defined by the requirements related to the aircraft sizing problem, *i.e.* the operational constraints,
- they lie at a given distance of the optimal values of all the criteria of the multiobjective optimisation problem, *i.e.* they satisfy the quality constraints,
- they are located on, or near the Pareto front of the multiobjective optimisation problem.

Thus, our multiobjective optimisation problem has been solved. Moreover, some knowledge on the problem has been added by the decision-maker to define the quality constraints. The resulting points of our methodology are thus located in a subset of interest for the engineers.

But our methodology enlightens another kind of problem. Calculation times are too important to widely use our algorithm on the models currently used in FPO. These models are of the same complexity as the SMAC model, and calculations using the SMAC model were very time consuming.

Thus, we decided to consider further another approach, by substituting our model by a response surface, which is the topic of the following chapter.

Chapter 4

Surrogate model of the evaluation function

Contents

4.1	Motivation	154
4.2	State-of-the-art of approximation techniques	156
4.2.1	Polynomial response surfaces	156
4.2.2	Kriging interpolations	158
4.2.3	Artificial neural network approaches	159
4.3	Selected methods	165
4.3.1	Polynomials using Taylor series developments	165
4.3.2	First order polynomials	169
4.3.3	Second order polynomials	172
4.3.4	Radial Basis Functions networks	175
4.4	Comparison of the results	178

The aim of this study was to solve a typical aircraft sizing problem, which consists in determining characteristics of an aircraft, starting from a set of requirements. These studies can be summarized as constrained, global and multiobjective optimisation problems, where constraints express requirements and physical feasibility, and objectives are market-driven characteristics of the aircraft.

One of the issues we face in aircraft sizing studies comes from the expensive computation time. Indeed, optimisation problems that arise in engineering design are often characterised by objective functions that are computationally intensive [Torczon and Trosset, 1998].

An attractive alternative to such prohibitive computational costs is **to make use of approximations within an optimisation context**, since approximations are far less expensive to compute. Moreover, the way to construct these approximations can be chosen and directed by the decision-maker.

In the framework of our study, we decided **to implement and test concurrently four approximation techniques**: two linear and one quadratic response surfaces and an artificial neural network model, based on Radial Basis Functions, further denoted RBF. Response surfaces are the most used approximation techniques, because they are easy to implement and naturally understandable. But one advantage in using RBF is that a minimal error can be fixed *a priori* by the user to fix the accuracy of the approximation.

Then, we compared the results that we obtained in approximating with the different approximation techniques we developed. Finally, we replaced the initial function by the approximations to perform the mono-criterion optimisation before defining the quality constraints introduced in the previous chapter.

Thus, in this chapter, we survey in a first part some approximation techniques, then, we detail the different methods we developed and tested, and finally, we show the compared results we obtained using these approximation techniques to replace our computationally expensive evaluation function.

4.1 Motivation

According to [Dennis and Torczon, 1995], a consistent theme in the engineering optimisation literature is that the time and cost required for the detailed analysis of a single design is often so great that it becomes prohibitive to implement a “black-box” optimisation approach to the design problem. The point of using approximation techniques is to reduce the number of full or detailed analyses required during the course of the optimisation iterations.

Typically, the objective function is expensive to evaluate because there are large numbers of system variables that must be determined for each choice of design parameters before the criteria can be evaluated [Booker et al., 1999].

Approximation models have several properties that make them attractive for use with optimisation [Simpson, 1998; Sobieski and Kroo, 2000]:

- They yield insight into the relationship between input design variables and output responses,
- they provide fast analysis tools for optimisation and design space exploration,
- they avoid potential numerical difficulties that some have experienced near the solution,
- they represent noisy analysis with an inherently smooth model,
- they provide a natural way of implementing coarse-grained parallelisation,
- they facilitate the integration of discipline dependent analysis codes.

Some multidisciplinary optimisation methods like collaborative optimisation, introduced in section 2.1.2 page 65 and denoted CO, need such properties like coarse-grained parallelisation to execute the optimisation of discipline subproblem on multiple processors. Moreover, in this method, some optimisation algorithms have difficulties with interdisciplinary constraints. Thus, the use of approximation models avoids the computational expense associated with successive evaluations of the interdisciplinary constraints.

There is a standard engineering practice for attacking multidisciplinary optimisation problems with expensive evaluation functions f [Booker et al., 1999]:

1. Choose a surrogate s for f that is either
 - (a) a simplified physical model of f ; or
 - (b) an approximation of f obtained by evaluating f at selected design sites and interpolating or smoothing the function values thus obtained.
2. Minimise the surrogate s on its definition space to obtain x_s .
3. Compute $f(x_s)$ to determine if improvement has been made over the best x found to date.

Many papers deal with the open question of how to manage the interplay between the optimisation and the fidelity of the approximation models, the aim being to insure that the process converges towards a solution of the original problem.

Some methods like those described in [Booker et al., 1999; Dennis and Torczon, 1995; Alexandrov et al., 2001; Wang, 2003; Cramer et al., 2006] increase the fidelity of the approximation during the optimisation, getting it more precise in the vicinity of a minimiser, or when it becomes clear that the approximation is not doing a good job of identifying trends in the objective [Torczon and Trosset, 1998]. The aim is to compromise between the research of a minimiser and the construction of an approximation that gives a reasonable estimation of the behaviour of the objective.

In the following section, we deal with the first point of the standard practice described previously when working with expensive evaluation functions, *i.e.* choosing a surrogate for our objective and constraint functions. Thus, we survey some approximation techniques, among which we have chosen to compare the properties and the results of two main techniques, response surface approximations and artificial neural network models, by replacing our expensive evaluation function during the optimisation.

4.2 State-of-the-art of approximation techniques

According to [Giunta et al., 1998], the use of approximation models has grown in popularity, and a variety of modeling methods have been employed. In this section, we describe some of the most known approximation techniques:

1. the polynomial response surfaces,
2. kriging interpolations,
3. artificial neural network models.

4.2.1 Polynomial response surfaces

The most popular techniques involve polynomial models, typically linear or quadratic functions, created by performing a least squares surface fit to a set of data, or with Taylor series developments.

Polynomial-based modeling methods have come to be known as response surface models.

Response surface modeling postulates a model of the form:

$$f(x) = P(x) + \varepsilon \quad (4.1)$$

where [Simpson, 1998]:

- $f(x)$ is the unknown function of interest,
- $P(x)$ is a known polynomial function of x ,
- ε is random error which is assumed to be normally distributed with mean zero and variance σ^2 .

4.2.1.1 First order

The general form of an affine response surface in n variables is:

$$P(x) = c_0 + \sum_{i=1}^n c_i x_i \quad (4.2)$$

The number of unknown coefficients in this equation is $n + 1$. Determining these coefficients requires at least $n + 1$ different values of f , using least squares method.

Early work in the development of approximation concepts concentrated on using linear approximations so that mathematical programming techniques such as linear programming could be employed [Alexandrov et al., 1998].

4.2.1.2 Second order

The general form of a quadratic response surface in n variables is:

$$P(x) = c_0 + \sum_{i=1}^n c_i x_i + \sum_{1 \leq i < j \leq n} c_{ij} x_i x_j \quad (4.3)$$

The number of unknown coefficients in this equation is $\frac{(n+2)(n+1)}{2}$.

4.2.1.3 Least squares method

One of the reasons of the popularity of polynomial models is the relatively computationally inexpensive and straightforward use of least squares method. Suppose these polynomial models are written in matrix notation as follows:

$$P(x) = c^T \bar{x} \quad (4.4)$$

where:

- $c = [c_0, c_1, \dots, c_{t-1}]^T$, t being the number of unknown coefficients,
- $\bar{x} = [1, x_1, \dots, x_n, x_1^2, x_1 x_2, \dots, x_i x_j, \dots, x_n^2]^T$ is a vector of size t , with $i \leq j$.

Here, we consider the equations for a quadratic response surface, but for a linear response surface, the equations are the same, without any term of second order.

Estimating the unknown coefficients requires to make a number of analyses greater or at least equal to t .

If we denote X the matrix containing on its rows the t sampled data \bar{x} ,

$$X = \begin{pmatrix} 1 & x_1^{(1)} & \dots & (x_n^{(1)})^2 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(t)} & \dots & (x_n^{(t)})^2 \end{pmatrix}$$

and y the vector of the results of the t analyses of all the design sites contained in the rows of X ,

$$y = [f(x^{(1)}), \dots, f(x^{(t)})]^T$$

then, we have the relation:

$$y \approx Xc \quad (4.5)$$

If the rank of X is t , *i.e.* if the rows of X are linearly independent, the least squares solution of 4.5 is unique and is given by:

$$c = (X^T X)^{-1} X^T y. \quad (4.6)$$

Then, we can predict the values of f at any point x in the design space using the approximation given in equation 4.4.

4.2.1.4 Taylor series approximation

Another method to approximate a function with a polynomial is to use Taylor series developments.

Linear approximations are given through the following equation:

$$\tilde{f}(x) = f(x_k) + \sum_{i=1}^n \frac{\partial f}{\partial x^i} (x^i - x_k^i) \quad (4.7)$$

and quadratic approximations are given by:

$$\tilde{f}(x) = f(x_k) + \sum_{i=1}^n \frac{\partial f}{\partial x^i} (x^i - x_k^i) + \sum_{i=1}^n \sum_{j=1}^n \frac{\partial^2 f}{\partial x^i \partial x^j} (x^i - x_k^i) (x^j - x_k^j) \quad (4.8)$$

where x^i is the i^{th} component of the vector x of dimension n .

According to [Alexandrov et al., 1998], a first-order Taylor series approximation could be computationally effective but such approximations are not always accurate. And second-order approximation is efficient as a local approximation, but some other existing approximations are better on a large neighbourhood.

Moreover, gradient vectors and Hessian matrices are not always available. It is rare to have them in their analytical expression, and it is expensive to compute them using finite differences.

4.2.2 Kriging interpolations

Among the large number of interpolation techniques (like Legendre polynomials, Newton polynomials, splines, etc.) are the class of interpolation techniques based on Bayesian statistics termed kriging models [Rodriguez et al., 2000].

Kriging model is an interpolation model to approximate response data obtained from deterministic computer simulations [Giunta et al., 1998]. This interpolation model was originally developed in the fields of spatial statistics and geostatistics.

Kriging model expresses the unknown function as:

$$f(x) = P(x) + Z(x) \quad (4.9)$$

where [Simpson, 1998]:

- $f(x)$ is the unknown function of interest,
- $P(x)$ is a known polynomial function of x ,
- $Z(x)$ is the realisation of a normally distributed Gaussian random process with mean zero, variance σ^2 and non-zero covariance.

The $P(x)$ term in equation 4.9 is similar to the polynomial model in a response surface and provides a “global” model of the design space. In many cases, it is simply taken to be a constant β as an estimate of the mean of the data.

While $P(x)$ “globally” approximates the design space, $Z(x)$ creates “localised” deviations so that the kriging model interpolates the sampled data points. The construction of the interpolation is based on spatial covariance between points in the sampled data sites: $\text{cov}(Z(x^i), Z(x^j))$. The correlation function which defines this covariance is specified by the user. See [Giunta et al., 1998; Simpson, 1998] or [Rodriguez et al., 2000] for a short and clear description of the principle of construction of a kriging interpolation.

The aim of the papers of [Giunta et al., 1998; Simpson, 1998; Simpson et al., 1998] is to compare the performance and the accuracy of a kriging interpolation model with polynomial response surfaces.

Kriging models are suitable for many engineering applications because they can be calibrated, *i.e.* additional values can be incorporated in the model to improve its prediction capacity [Cramer et al., 2006]. Moreover, they can capture oscillatory (multi-modal) response trends whereas quadratic polynomials are by definition unimodal.

According to [Simpson, 1998], the use of response surfaces for approximating deterministic computer analyses is statistically questionable due to the lack of random error in the computer model. The use of kriging models is more appropriate and perhaps more justified from a statistical point of view for approximating deterministic computer experiments. Indeed, kriging models interpolate between data points which may yield more accurate results since computer experiments typically do not contain random errors. Moreover, an important by-product of kriging models construction processes is an estimate of the uncertainty in their prediction.

4.2.3 Artificial neural network approaches

The last category of approximation techniques we survey is artificial neural network, further denoted ANN. The concept of ANN is inspired by human brain behaviour. It consists of multiple simple computation units, neurons, which nonlinearly transform their input signal [Giannakoglou and Karakasis, 2006; Giannakoglou, 2004]. At each neuron, the input signal is the weighted sum of the outputs of all the neurons connected to it (see figure 4.1). The entire network is stored in the form of the weights associated with the synaptic connections.

Advantages of neural networks are their ability to approximate arbitrarily well any function and, in particular, to successfully model abrupt changes in the function.

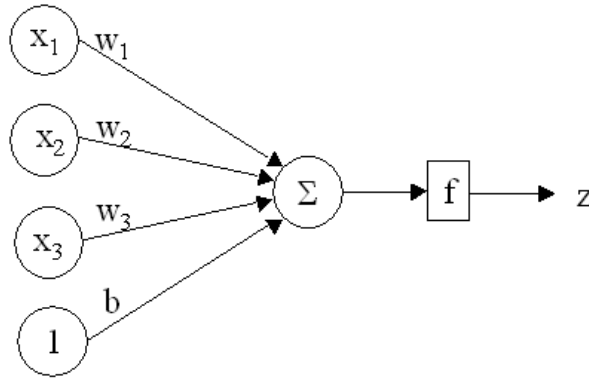


Figure 4.1: General structure of an artificial neuron

4.2.3.1 Multi-layer perceptrons

In a multi-layer perceptron, we can identify the following elements [Van Grieken, 2004]:

- a set of inputs, x_i , which represent the independent parameters of the problem,
- a set of synaptic weights, w_i , which represent the connection between the input x_i and the neuron,
- a bias, b ,
- a sum operator, Σ ,
- an activation function f .

A real value, z , is associated with each neuron and is calculated through the formula:

$$z = f\left(\sum_{i=1}^k w_i x_i + b\right). \quad (4.10)$$

The most used activation function is the sigmoidal function:

$$f(x) = \frac{1}{1 + e^{-(x-\theta)/\tau}}. \quad (4.11)$$

See the figure 4.2 for different representations of activation functions.

The choice of the multi-layers structure is known as choosing the architecture in the neural network framework and is analogous to model selection in the regression framework. The figure 4.3 is an illustration of a three layer structure.

The user needs to decide the number of input nodes, the number of hidden layers and hidden nodes, the number of output nodes and the activation functions [Balkin and Lin, 2000]. The number of input nodes corresponds to the number of variables to consider for the model. The hidden layer and node parameter selection is very important in that it is this feature that allows the ANN to perform the nonlinear mapping between inputs and outputs.

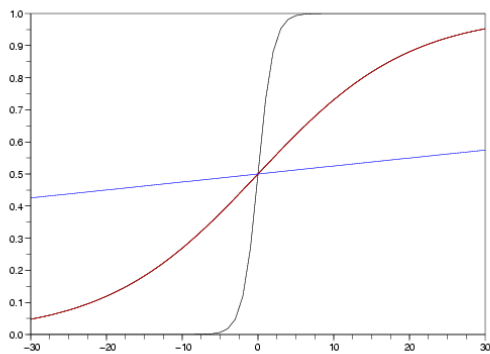


Figure 4.2: Activation functions of a classical neural network. Here, $\theta = 0$ and $\tau = 1$ for the black function, $\tau = 10$ for the red function and $\tau = 100$ for the blue function.

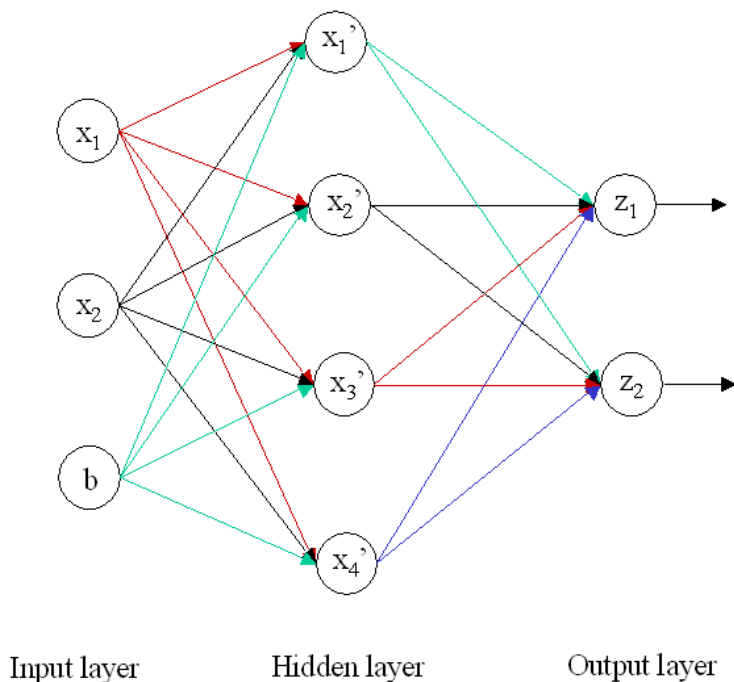


Figure 4.3: Three-layer neural network

The number of output nodes is specified directly by the problem.

According to [Balkin and Lin, 2000], there is currently no widely accepted method for making these model design decisions, but any standard three-layer feed-forward neural network is capable of approximating any function from one finite dimensional space to another to any desired degree of accuracy, provided sufficiently many hidden units are available.

4.2.3.2 Support Vector Machines

Another kind of ANN that we introduce here is Support Vector Machine, further denoted SVM. It is grounded in the framework of statistical learning theory [Smola and Schölkopf, 2004]. It was initially designed to solve pattern recognition problems [Vapnik et al., 1997], where, in order to find a decision rule with a good generalisation ability, one selects some (small) subset of the training data, called Support Vectors (SVs). Optimal separation of the SVs is equivalent to optimal separation of the entire data.

Suppose we are given training data $\{(x_1, y_1), \dots, (x_\ell, y_\ell)\} \in \mathbb{R}^d \times \mathbb{R}$. In SVM regression, the goal is to find a function $f(x)$ that has at most ε deviation from the currently obtained targets y_i for all the training data, and at the same time is as flat as possible.

SVMs approximate the function by a linear regression. The general SVM formulation is given in the following equation:

$$f(x) = \langle w, \Phi(x) \rangle + b \text{ with } x \in \mathcal{X}, w \in \mathcal{F} \text{ and } b \in \mathbb{R} \quad (4.12)$$

where $\Phi : \mathcal{X} \rightarrow \mathcal{F}$ maps x into some feature space \mathcal{F} and $\langle \cdot, \cdot \rangle$ denotes the inner product in \mathcal{F} .

As an example, we describe the case of linear functions as in [Smola and Schölkopf, 2004]:

$$f(x) = \langle w, x \rangle + b \text{ with } x, w \in \mathcal{X} \text{ and } b \in \mathbb{R}. \quad (4.13)$$

In this case, ensuring the *flatness* means to minimise the norm of w ; thus, this problem can be written as a convex optimisation problem:

$$\begin{aligned} \min \quad & \frac{1}{2} \|w\|^2 \\ \text{subject to} \quad & \begin{cases} y_i - \langle w, x_i \rangle - b \leq \varepsilon \\ \langle w, x_i \rangle + b - y_i \leq \varepsilon. \end{cases} \end{aligned} \quad (4.14)$$

[Smola and Schölkopf, 2004] gives an overview of the basic ideas underlying SVMs for function estimation. It begins by explaining the principles in a linear case, in using a Lagrangian formulation to solve the optimisation problem stated in equation 4.14, and it continues in the general case, with nonlinear algorithms.

Note that what is interesting with SVMs is that the complete algorithm can be described in terms of inner products between the data x_i . Hence, in the nonlinear case, it suffices to know $k(x, x') := \langle \Phi(x), \Phi(x') \rangle$ rather than Φ explicitly. An implicit mapping is done via

kernels which corresponds to an inner product in some feature space \mathcal{F} .

According to [Vapnik et al., 1997], the complexity of the solution of the function approximation problem using SVM representation depends on the complexity of the desired solution rather than on the dimensionality of the space.

See [Barzilay and Brailovsky, 1999] for applications in pattern recognition, [Shin and Chob, 2006] for applications in direct marketing or [Fan et al., 2005] for applications in aerodynamic data modeling.

4.2.3.3 Radial Basis Functions

The last kind of artificial neural networks we survey here are radial basis functions, further denoted RBF. This kind of network is formed from three layers of neurons [Giannakoglou and Karakasis, 2006]. The activation function depends on the distance of the input x from the corresponding centre c :

$$h(x, c) = h\left(\|x - c\|_2\right). \quad (4.15)$$

The network output is obtained by linearly combining the responses of the hidden neurons to the input:

$$y = \sum_{i=1}^k w_i h\left(\|x - c_i\|_2\right). \quad (4.16)$$

Finding an approximation using RBF can be viewed as a minimisation problem, where the objective function is the error as it is defined in the following equation:

$$E(\tilde{f}) = \frac{1}{2} \sum_{i=1}^k (y_i - \tilde{f}(x_i))^2 + \frac{1}{2} \lambda \|D\tilde{f}\|^2 \quad (4.17)$$

where the regularisation parameter λ controls the tradeoff between attaching strictly to the training patterns and reconstructing a smooth mapping as reflected in the measure of derivatives $\|D\tilde{f}\|$.

The simplest RBF network is obtained by setting the inputs of all training patterns as centres, $c_i = x_i$, and $\lambda = 0$.

[Krishnamurthy, 2003] describes the construction of the approximation based on classical RBF, compactly supported RBF and augmented RBF. Here, we only consider classical RBF.

In classical RBF based methods, the interpolation of a surface is performed as a linear combination of radial functions as:

$$\tilde{f}(x) = \sum_{i=1}^k w_i h\left(\|x - c_i\|_2\right) \quad (4.18)$$

where:

- the radial functions h are functions of the radial distance $\|x - c_i\|_2$ from node i . The table 4.1 lists the most commonly used classical RBF.

- the w_i are interpolation constants to be determined,
- k is the number of sample or data points with known function values y_i such that $\tilde{f}(x_i) = y_i$.

The constant σ in basic functions is adjusted to obtain the best fit. A way to adjust both the centres of the functions, c_i , and the width, σ_i for all $i \in 1, \dots, k$, can be found in [Orr, 1998; Benoudjit et al., 2002].

According to [Fritzke, 1994], RBF networks offer an interesting alternative to multi-layer perceptrons since they can be trained much faster. Training involves the placement of the localised units in input vector space. The difficulty and critical part with RBF networks is the placement and parameterisation of the local units as well as the choice of their number.

[Carr et al., 2001] employ RBF networks in a different way, not to build a surrogate model. They use RBFs to reconstruct smooth, manifold surfaces from point-cloud data and to repair incomplete meshes.

There are other approximation techniques that are not described here, like for instance splines (that are extensively used at Boeing according to [Grandine, 2005]). The aim here was not to make an exhaustive review of all existing approximation techniques, but to survey the ones that can be the most interesting to be used in our problem.

After this survey, we now explain the choices we made among all these possibilities, and we detail the results we obtained by replacing our objective and constraint functions by different surrogate models.

4.3 Selected methods

In this section, we describe the way we implemented the different surrogate models we decided to test and use in our aircraft sizing problem, in the chronological order in which we develop them. In each subsection, we explain the general problem of the interpolation, or learning, data basis, and the choice we made to fill this data basis.

4.3.1 Polynomials using Taylor series developments

The first idea came from the observation that most of the time, the admissible set is convex and the different optima of the mono-criterion optimisation are located on the boundary of the admissible space, more precisely at points where some constraints are binding.

Thus, to simplify the expression of the constraints in the general optimisation problem, *i.e.* $G(x) \leq 0$ in its nonlinear form, we decided to replace them by a linear inequation of the form:

$$Ax \leq b \tag{4.19}$$

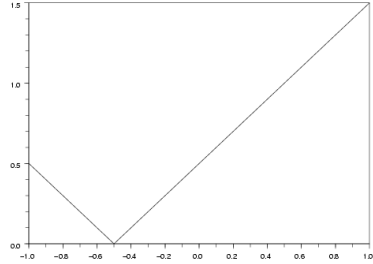
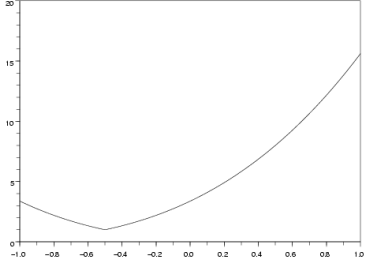
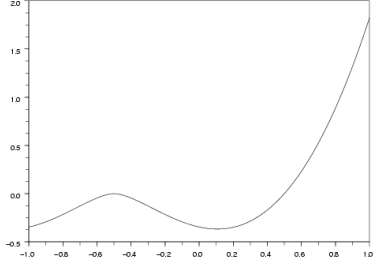
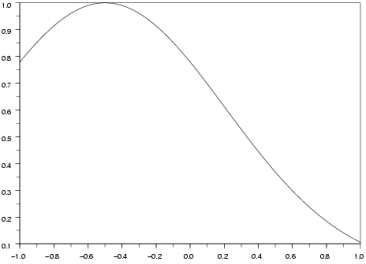
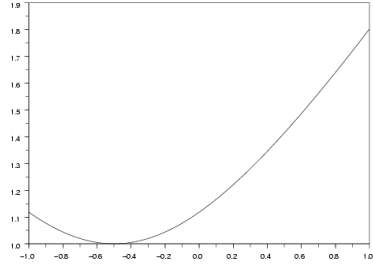
Basis function name	Linear
Equation	$\varphi(r) = \sigma r$
Graphical Representation	
Cubic	Thin plate spline
$\varphi(r) = (r + \sigma)^3$	$\varphi(r) = r^2 \log(\sigma r^2)$
	
Gaussian	Multi-quadratic
$\varphi(r) = e^{-(\frac{r}{\sigma})^2}$	$\varphi(r) = (r^2 + \sigma^2)^{\frac{1}{2}}$
	

Table 4.1: Classical RBF functions, where $r = \|x - c_i\|_2$ and $c_i = -0.5$

Then, the general optimisation problem

$$\begin{aligned} \min \quad & F(x) \\ \text{s.t.} \quad & G(x) \leq 0 \end{aligned} \quad (4.20)$$

becomes:

$$\begin{aligned} \min \quad & F(x) \\ \text{s.t.} \quad & Ax \leq b \end{aligned} \quad (4.21)$$

where only the constraints are linearised.

Our aim is to be able to use optimisation techniques coming from Linear Programming or Quadratic Programming.

To find this approximate expression of the constraints, we have to linearise the constraints around the boundary of the admissible set. Thus, we have to obtain more information on this boundary. To do this, we decided to project the admissible points, obtained when solving the constraint satisfaction problem in the previous step of our optimisation process, on the boundary of the admissible space.

We still worked with Genetic Algorithms. The algorithm described in the previous chapter has been adapted to project points on the boundary of the admissible space.

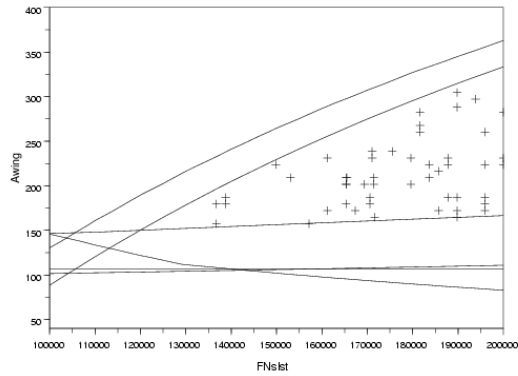
This time, the fitness function of the GA depends on the distance of the current point from the boundary of the admissible set, thus from the closest constraint to it. The way to create children is unchanged. Only the mutation operator has been modified. The points mutate in the direction of the constraints that move them the farthest away from the barycentre. The distance of the displacement is randomly sorted between zero and the distance to this closest constraint.

A minimum distance is also imposed between points in the design space, like it is done in NSGA (described in in paragraph 2.3.4.2.3, page 88). The aim is to obtain a well-distributed repartition of the points along the boundary of the admissible space.

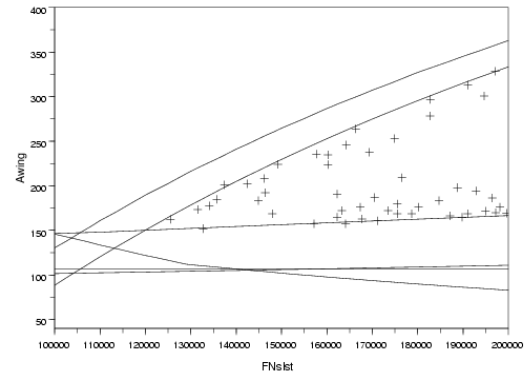
Because of the intrinsic randomness of GA operators, like the mutation operator, it is difficult for the algorithm to find points that exactly reach the boundary of the admissible set. Thus, we decided to fix a small distance such that when the distance of a particular point from the constraint is smaller than this minimum distance, it is considered that the process has converged. When all points have converged this way, they are projected on the boundary by using a Newton projection method.

This minimum distance is a compromise between convergence and well-distributed repartition of points. It has to be not too small to allow the algorithm to converge, but also not too large. Otherwise, the algorithm would manage to converge too fast and would not have enough time to distribute the points along the boundary of the admissible space.

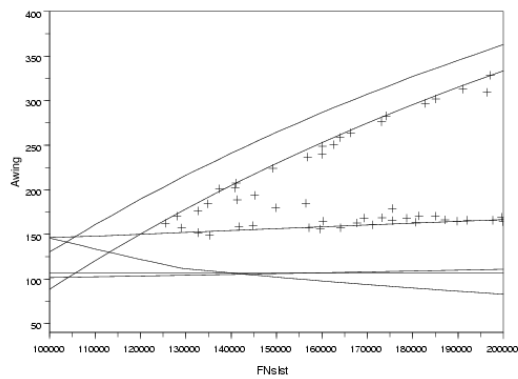
The figure 4.4 represents each step of the Genetic Algorithm when projecting the points on the boundary of the admissible set.



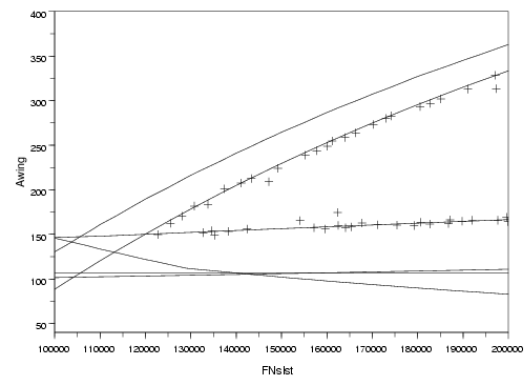
Admissible population obtained when solving the constraint satisfaction problem



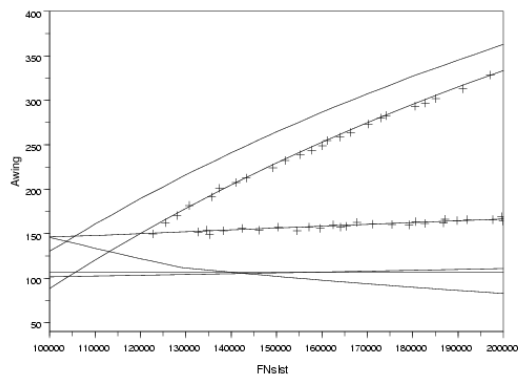
First iteration, 2 points among 50 have reached the boundary



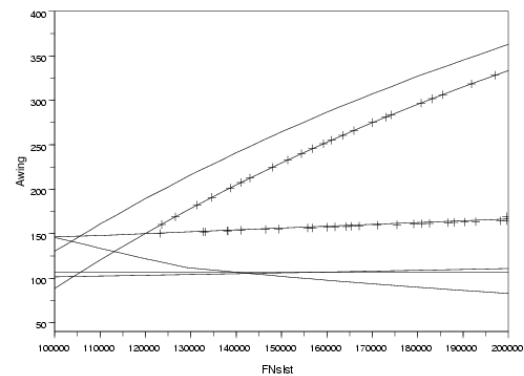
Second iteration, 15 points among 50 have reached the boundary



Third iteration, 27 points among 50 have reached the boundary



Fourth iteration, 45 points among 50 have reached the boundary



Fifth iteration, 50 points among 50 have reached the boundary

Figure 4.4: Evolution of the points during their projection on the boundary of the admissible set. We notice that the points go far from the barycentre during the evolution of the population. The last step is just before the projection of the points on the boundary by using a Newton projection method.

Now that we have well-distributed points on the boundary of the admissible set, we can linearise the constraints around these points. We calculate the gradient using finite differences, but since this operation is expensive in general, we decided not to linearise systematically around every point that has reached a constraint.

Thus, for each active constraint, we identified the points located on the extremities of this particular constraint, in each direction. We linearise the constraint around these points. Then, we calculate the values of the other points that have reached the same constraint using the linearised expressions. If this value is not sufficiently precise, *i.e.* the difference between this value and the exact value of the constraint is too important according to the decision-maker, then the constraint is linearised once again around this particular point.

The figure 4.5 is the graphical representation of the result of the linearisation process in a design space of two dimensions.

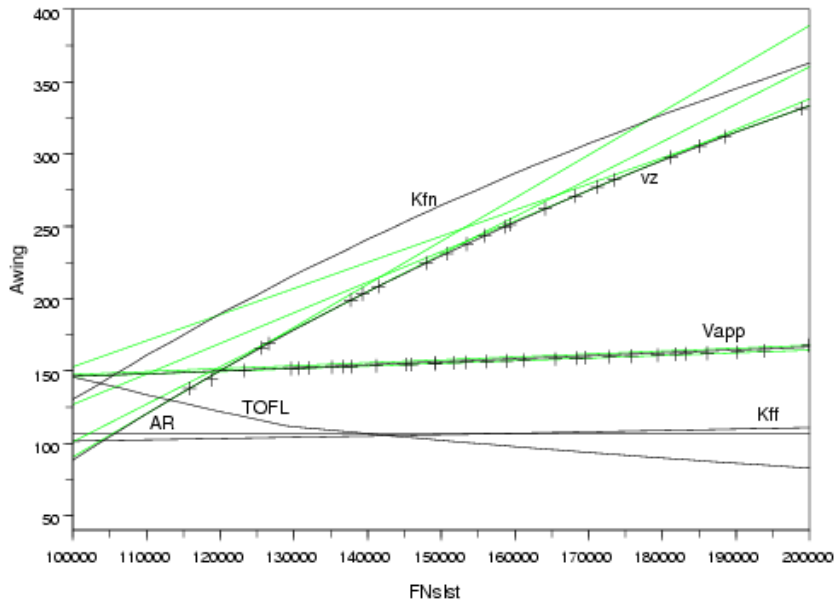


Figure 4.5: Graphical result of the linearisation of the constraints. The constraints *vz* is linearised around four points, and the constraint *Vapp* is linearised around two points.

The figure 4.6 represents the result of the optimisation of the MTOW in the same design space using an optimisation technique coming from Linear Programming. This method is an intrinsic function in *Scilab* [INRIA Copyright, 1989], named *linpro*. This method solves the following optimisation problem:

$$\begin{aligned}
 \min \quad & c^T x \\
 \text{s.t.} \quad & Ax \leq b \\
 & x_{min} \leq x \leq x_{max}
 \end{aligned} \tag{4.22}$$

As we have linearised only the constraints, we called this function by using the gradient of

the criterion at the barycentre of the admissible population. Moreover, we use the barycentre as the initial point for the algorithm, because we know that this particular point is feasible, and it is the one located the farthest from the boundary of the admissible set, hence it would not influence the result of the optimisation. Thus, we called the function by the following command:

```
linpro(grad(criterion), A, b, xmin, xmax, Barycentre)
```

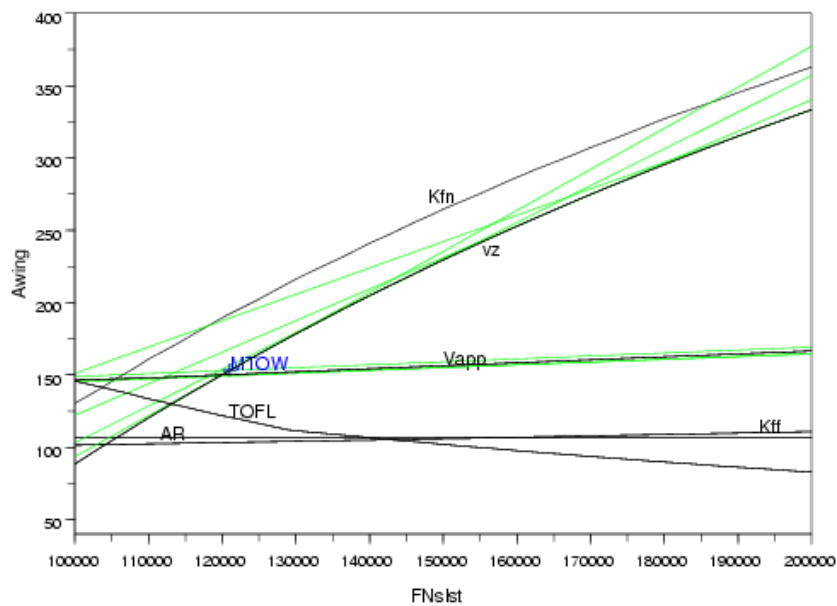


Figure 4.6: Result of the optimisation of the MTOW using a Linear Programming optimiser

As we can see on the figure 4.6, the Linear Programming optimiser manages to find the right optimum.

This method could not be applied to the second criterion because *linpro* fails in finding the right step to make the process converge towards the optimum. We suppose this failure is due to the gradient values of the criterion. They are too close to zero, so the function *linpro* gives the error message that the domain is not bounded.

But as the method works for the MTOW, we can say that the failure of the process is not imputable to our methodology.

Since it seems difficult to use a linearisation of the constraints to simplify the optimisation formulation, we decided to test other approximation methods, the first one being polynomial response surfaces.

4.3.2 First order polynomials

The first step in using polynomial response surfaces is to establish a data basis on which the regression can be applied to determine the coefficients of the polynomial.

We have already calculated some points in the previous steps of our general process:

1. the initial population which was given by a Latin Hypercube Sampling, noted LHS,
2. the admissible population which was produced by solving the constraint satisfaction problem related to our optimisation problem,
3. and now the population of points which were projected on the boundary of the admissible set (see the previous paragraph).

We decided to test the results of the regression when we use the different populations. The coefficients of the polynomial are determined by using the Least Squares method such as described in the paragraph 4.2.1.3 page 157.

The figures 4.7, 4.8 and 4.9 are the graphical representations of the results of the different approximations depending on the population used for the regression.

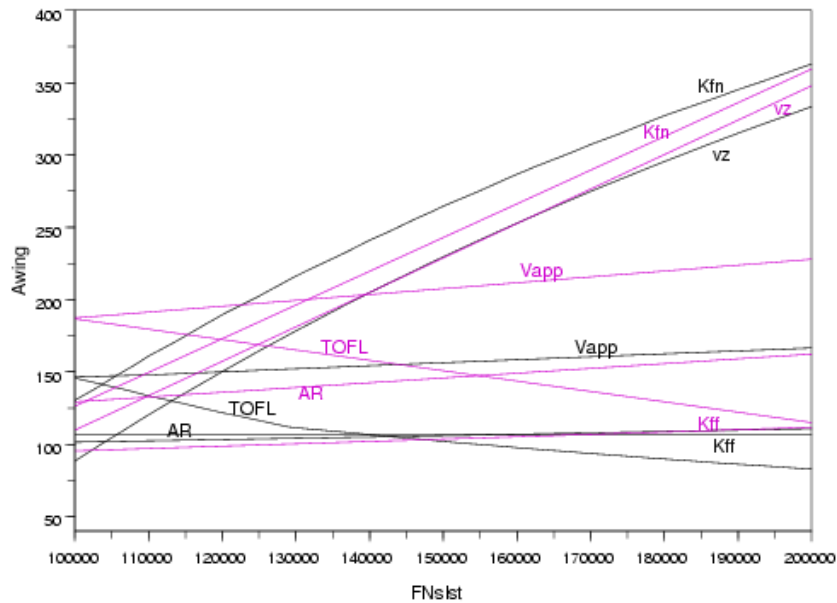


Figure 4.7: First order polynomial response surface using the LHS initial population for the regression

The first figure was obtained by using only the LHS initial population for the regression. The second one was obtained by using the LHS initial population and the admissible

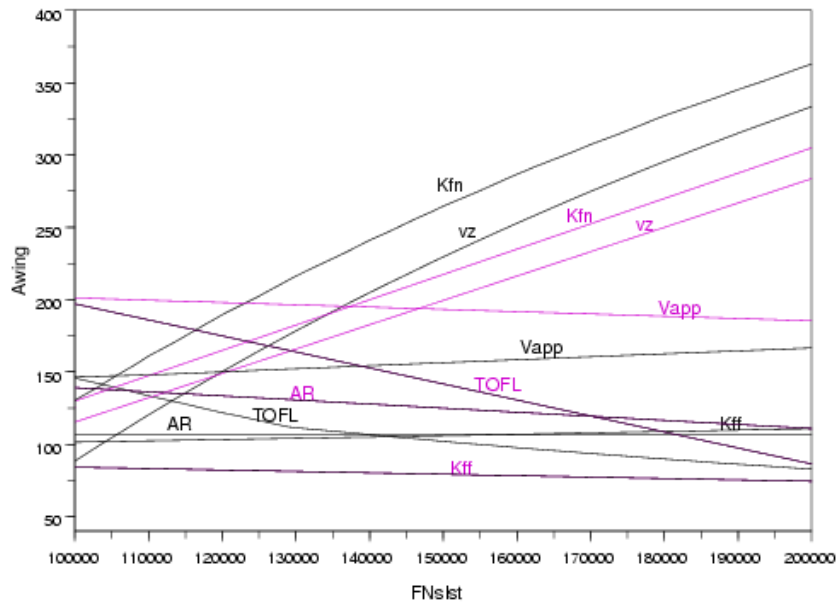


Figure 4.8: First order polynomial response surface using the LHS initial population and the admissible population for the regression

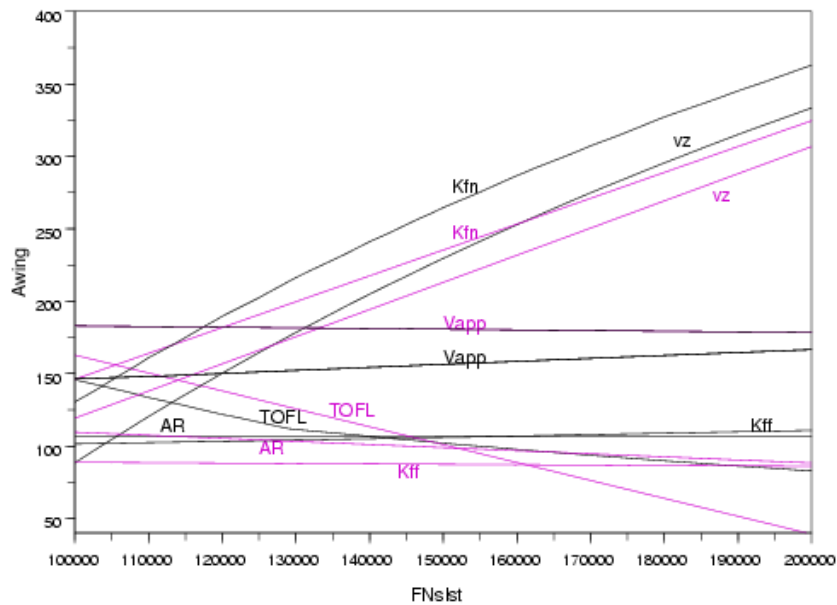


Figure 4.9: First order polynomial response surface using the LHS initial population, the admissible population and the projected population for the regression

population, and the last one was obtained by using all the populations we had, *i.e.* the LHS initial population, the admissible population and the projected population.

It is difficult to assess graphically which solution is the best. We decided to compare the approximations at a point that was not used for the regressions, the barycentre. This comparison gives us a more objective mean to make a choice between the three approximations.

The result of this comparison can be seen in the table 4.2 for the exact values of the constraints against the approximate values, while the relative errors in percentage are written in the table 4.3.

Constraint name	Real values	First population	Second population	Last population
TOFL (m)	1079.71	1522.83	1455.92	1330.38
Vapp (kt)	106.95	121.72	114.20	112.25
AR (no dim)	5.45	8.33	7.09	6.58
Kff (no dim)	1.57	1.54	1.66	1.62
vz (ft/min)	1127.95	1131.89	712.59	860.23
Kfn (no dim)	0.76	0.77	0.89	0.84

Table 4.2: Comparison of the evaluation of the constraints in the barycentre

Constraint name	Regression on the first population	Regression on the second population	Regression on the last population
TOFL	41%	34%	23%
Vapp	13%	6%	4%
AR	52%	30%	20%
Kff	2%	5%	3%
vz	0.35%	36%	23%
Kfn	1.7%	17%	11%

Table 4.3: Relative errors in the approximation of the constraints

As we can see in these two tables, the results we obtained with the three data bases we used to construct a linear response surface of our constraints are not satisfactory. Errors in the three approximations are too important to be neglected. Thus, we decided to increase the degree of the approximating polynomial and to make the same kind of comparison.

4.3.3 Second order polynomials

We now perform the same study as for first order polynomial response surfaces, here for a second order polynomial response surface. We use the same three populations as data bases on which we perform a Least Squares method to determine the coefficients of the polynomial. We made the same kind of graphical representations, as can be seen in the figures 4.10, 4.11 and 4.12.

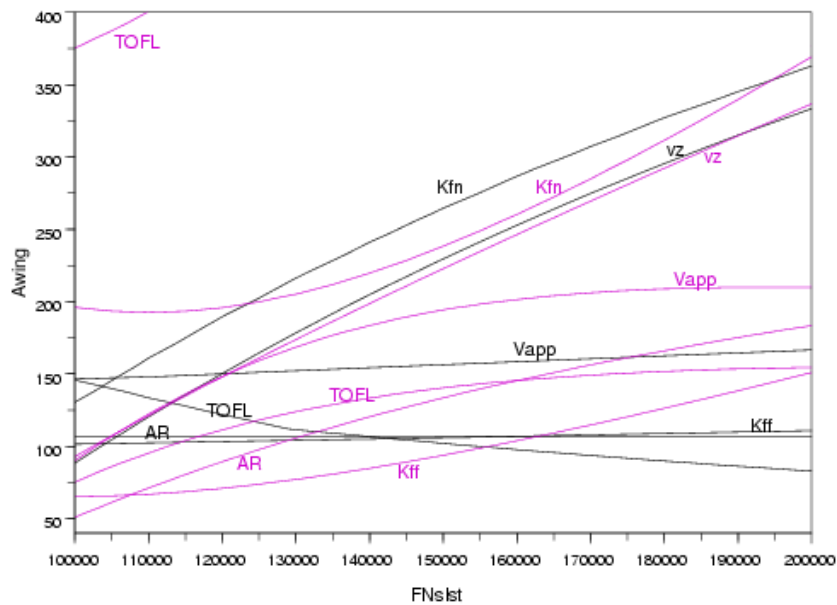


Figure 4.10: Second order polynomial response surface using the LHS initial population for the regression

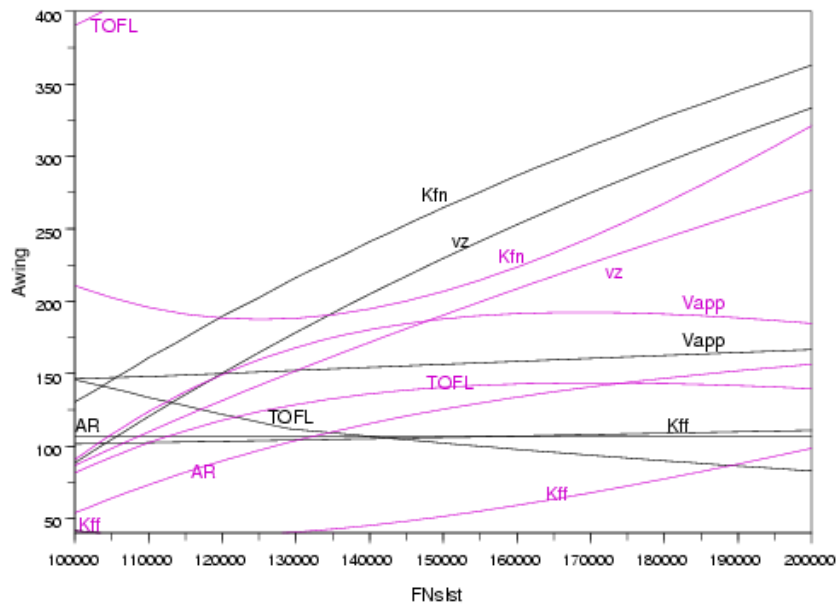


Figure 4.11: Second order polynomial response surface using the LHS initial population and the admissible population for the regression

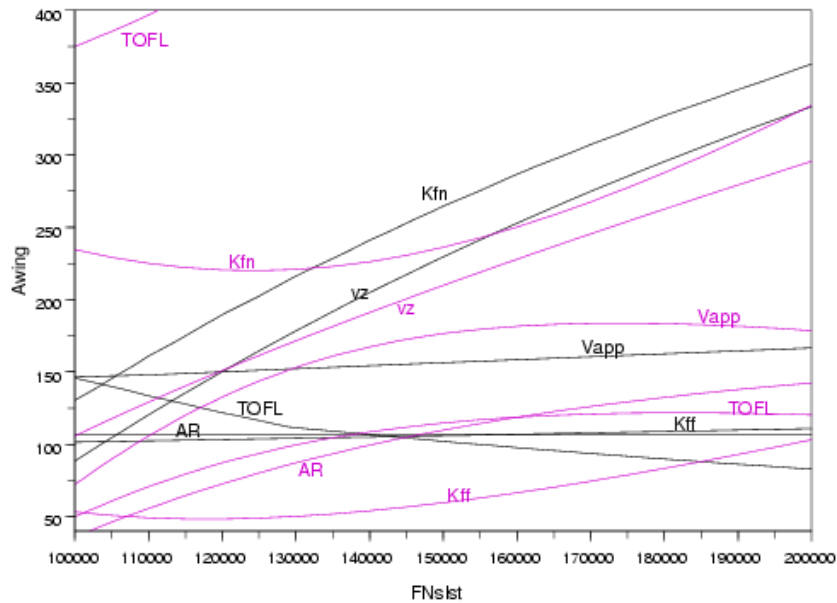


Figure 4.12: Second order polynomial response surface using the LHS initial population, the admissible population and the projected population for the regression

Considering these graphical representations, we are lead to the same conclusion as previously: it is difficult to assess the quality of the approximations by looking at these figures. The only comment we can make is that some approximate constraints do not have the same curvature than the real constraints. The most evident difference can be seen for the TOFL or the Kfn constraints.

To assess the quality of the approximations, we compare them at the barycentre as it was done in the previous paragraph. The result of this comparison can be seen in the table 4.4 for the exact values of the constraints against the approximate values, and the relative errors in percentage are written in the table 4.5.

Constraint name	Real values	First population	Second population	Last population
TOFL (m)	1079.71	1385.76	1408.02	1278.96
Vapp (kt)	106.95	118.36	113.11	111.05
AR (no dim)	5.45	7.24	6.57	6.11
Kff (no dim)	1.57	1.47	1.65	1.61
vz (ft/min)	1127.95	1167.32	685.04	844.48
Kfn (no dim)	0.75	0.74	0.89	0.84

Table 4.4: Comparison of the evaluation of the constraints in the barycentre

When we compare these results with those obtained with a first order polynomial response

Constraint name	Regression on the first population	Regression on the second population	Regression on the last population
TOFL	28%	30%	18%
Vapp	10%	5%	3%
AR	32%	20%	12%
Kff	6%	4%	2%
vz	3%	39%	25%
Kfn	2%	18%	11%

Table 4.5: Relative errors in the approximation of the constraints

surface, we see that they are globally much less important, except in a few cases. Thus a second order polynomial response surface is more efficient in approximating the constraints of our problem.

But errors are still too important to be acceptable. Thus, we decided to test a new kind of approximation technique than response surfaces. We decided to test Radial Basis Functions networks, noted RBF, because they are easy to implement and to adapt to our particular problem.

4.3.4 Radial Basis Functions networks

For the simplicity of the implementation, we decided that each node of our RBF network will be one of the points of all the populations we have at our disposal, *i.e.* $c_i = x_i$, and the parameter λ , defined in equation 4.17 page 163, that ensures the compromise between the interpolation and the smoothness of the approximation, is assigned to zero, to avoid calculating any gradient of the function.

To construct the network, we proceed as [Fritzke, 1994], we add the network one more basis function until the mean error of the approximated function calculated on the known points has reached the decision-maker threshold. And the centre of this new basis function is localised at the point of largest error in the learning data basis.

Remark 4.1. The required precision of the RBF network should not be higher than what allows the intrinsic uncertainty of the model.

As in the previous paragraphs, we construct two different learning data bases with the populations we have already calculated. The first learning data basis contains the initial population, coming from a LHS, and the points projected on the constraints. The second population contains the first one plus the population of the admissible points.

The figures 4.13 and 4.14 are graphical representations of the results.

We can see graphically on the figure 4.13 a main difference with the response surface approximations: the RBF approximations have at least the same curvature as the real constraints. On the other hand, the figure 4.14 shows that the network does not manage to approximate the variation direction of the constraints, it is oscillating.

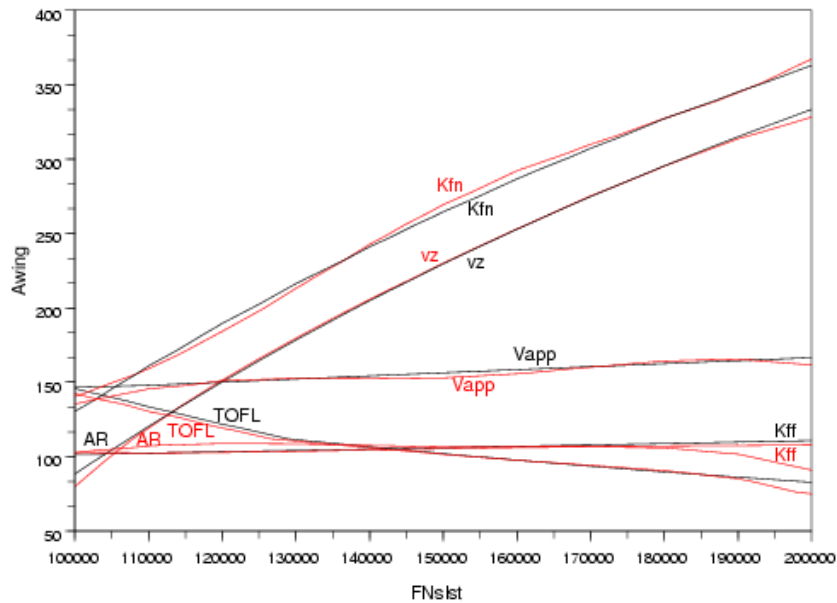


Figure 4.13: RBF approximation using the LHS initial population and the projected population as the learning data basis

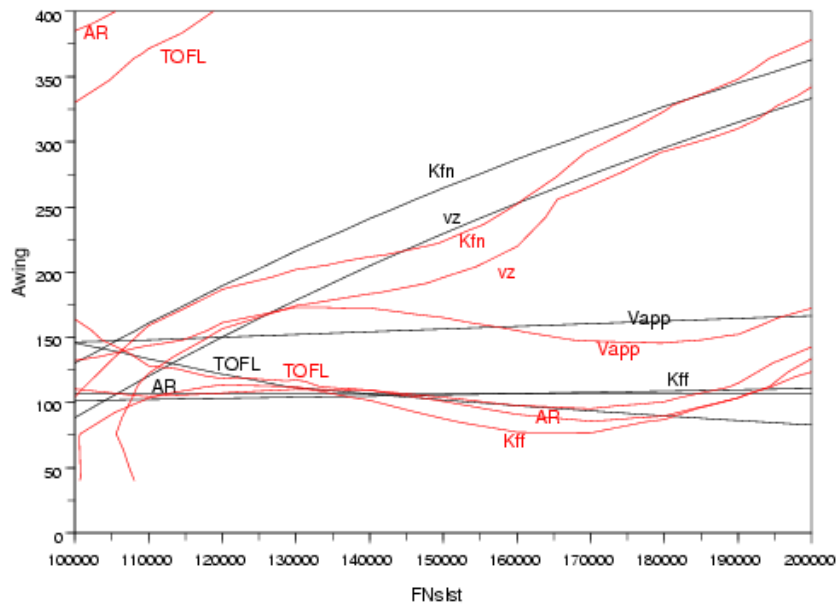


Figure 4.14: RBF approximation using the LHS initial population, the admissible population and the projected population as the learning data basis

Again, to have a more objective comparison of the two learning data bases, we calculate the differences they produce related to the real function at the barycentre of the admissible population. The result of these comparisons can be seen in the tables 4.6 and 4.7.

Constraint name	Real values	First population	Second population
TOFL (m)	1079.71	1108.19	1259.64
Vapp (kt)	106.95	107.04	108.62
AR (no dim)	5.45	5.75	5.80
Kff (no dim)	1.57	1.57	1.76
vz (ft/min)	1127.95	1181.10	618.11
Kfn (no dim)	0.75	0.75	0.92

Table 4.6: Comparison of the evaluation of the constraints in the barycentre

Constraint name	First population	Second population
TOFL	2.6%	16.6%
Vapp	0.08%	1.5%
AR	5.5%	6.4%
Kff	0.3%	11.5%
vz	4.6%	45.1%
Kfn	0.4%	21.9%

Table 4.7: Relative errors in the approximation of the constraints

In all the approximation techniques described in the section 4.2 page 156, one of the important issues for the construction of accurate models is to choose a proper set of initial design sites.

[Kodiyalam, 2001] compares two sampling methods, which are variations of DoE, a uniform dispersal of design points, and a hyper-sphere method. His conclusion is that uniform distribution of points inside the design space maximises the amount of information extracted from the design space using a fixed number of samples.

When we analyse the comparison results in the tables 4.6 and 4.7, we can say that the number of learning design points is also important. Indeed, the first learning data basis contains the LHS initial population, thus it is distributed in all the design space, as advised by [Kodiyalam, 2001]. It contains also the points projected on the boundary of the admissible set, because it is the region where we want the approximation to be the more precise. According to the results in the table 4.6 and on the figure 4.13, we can say that this learning data basis is efficient.

The second learning data basis contains the first one, and we add to it the points of the admissible population. In this case, there are much more points to perform the learning of the network, and according to the results described in table 4.6 and on the figure 4.14, we can say that this learning data basis is not efficient.

The table 4.8 allows us to confirm what we just said. It lists the number of nodes each network needs to approximate the constraints at a given precision. For the second network, this precision had never been reached, and the calculations were stopped because the mean error on the design space did not decrease when adding more nodes.

Constraint name	Number of centres for the first network	Number of centres for the second network
TOFL	20	35
Vapp	13	21
AR	17	27
Kff	9	28
vz	5	28
Kfn	9	23

Table 4.8: Number of centres needed to interpolate each constraint

Our implementation of the RBF network can be improved. The first way is by assigning the nodes to other points than the learning points. The second, according to [Benoudjit et al., 2002], is that identical widths σ for all Gaussian kernels, as we did, should be avoided, since their width should depend on the positions of the centres, which in turn depend on the data distribution in the input space. And, unfortunately, most real-life problems show non-uniform data distributions. The option is to estimate the width of each Gaussian functions independently.

The aim of this chapter was to test and compare the different results we obtain with different approximation techniques. When analysing all the results we get, we decided to continue to use RBF networks. The way we implemented them is sufficient for our study. We know it could be improved, but we decided not to spend more time on it.

Thus, we now compare the results of the mono-objective optimisation on the real evaluation function and when we use RBF networks to approximate the constraints and the objective functions of our problem.

4.4 Comparison of the results

We built two RBF networks to approximate the criteria in the same way we constructed them for the constraints.

According to the table 4.9 and the figures 4.15 and 4.16, we see that the results concerning the two criteria, using the real functions or the approximations, are similar. The maximum error is of 2.14% for the Vapp. This error is acceptable in our context of preliminary design of an aircraft, but depending on the kind of study we want to perform, the network may have to be refined for more precise studies.

Just as a remark, we can see on the figure 4.16 that the quality constraint Vapp (in blue on the figure) is worse approximated than the operational constraint Vapp (in black on the

		FNs1st (daN)	Awing(m ²)	Optimum value	Relative error
MTOW (kg)	Real function	119818.99	149.72	80130.60	0.16%
	Approximate function	119644.13	150.36	80003.61	
Vapp (kt)	Real function	200000.00	333.38	91.63	2.14%
	Approximate function	200000.00	328.27	89.67	

Table 4.9: Comparison of the results of the optimisation using the real function and the approximation

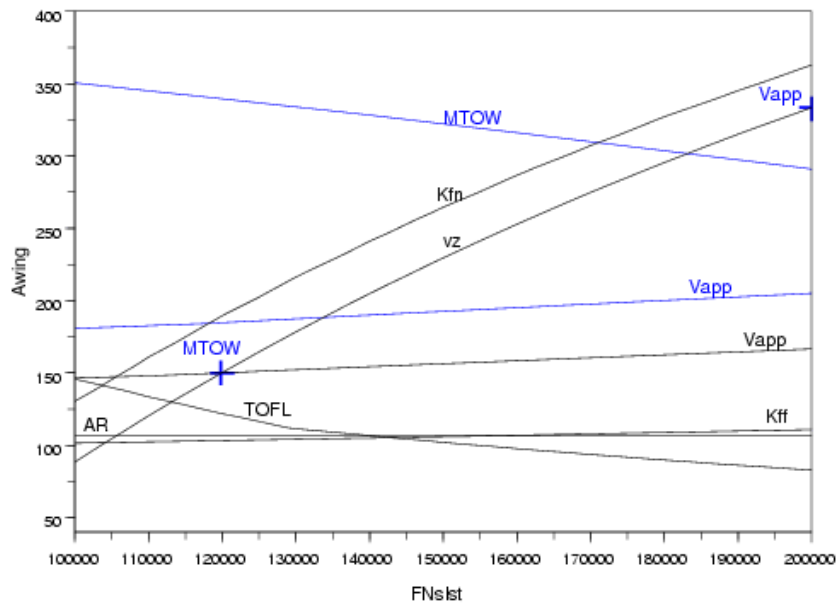


Figure 4.15: Result of the optimisation using the real functions

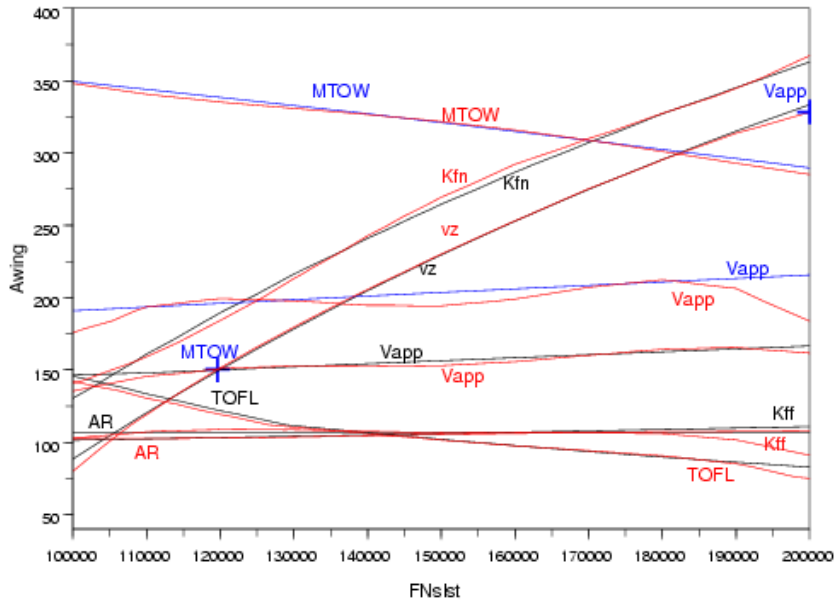


Figure 4.16: Result of the optimisation using the approximate functions

figure). The RBF network is more oscillating for the quality constraint. This observation allows us to confirm that the learning population must contain the points we projected on the boundary of the admissible set, the approximation is then more precise on this region, which is exactly what we intended to obtain.

Thus, we validate our approach for the next steps of our process, any optimisation of one or several criteria will be done using the approximation functions built with RBF networks.

We now test the approach on our more complex model of aircraft, the SMAC. We produced the same graphical and numerical results, which are shown in the table 4.10 and on figure 4.17.

The calculation times to compute the optimisation when using the RBF network on the SMAC model were exactly the same as when using them on the USMAC model, and the errors introduced by the approximation are quite negligible. Thus, the RBF network plays its role of approximating the constraint and criteria functions with the required accuracy, and of making the optimisation faster.

In the next chapter, we will introduce some means for the decision-maker to understand the properties of a typical aircraft sizing problem, like the uncertainty related to the model of aircraft he/she uses.

		krub (no dimension)	wing_area (m ²)	Optimum value	Relative error
MTOW (kg)	Real function	1.17	450.05	178017.49	0.4%
	Approximate function	1.17	444.46	177320.42	
app_speed (kt)	Real function	2	591.52	139.43	3.4%
	Approximate function	1.99	600	134.67	

Table 4.10: Comparison of the results of the optimisation using the real function and the approximation using the SMAC model

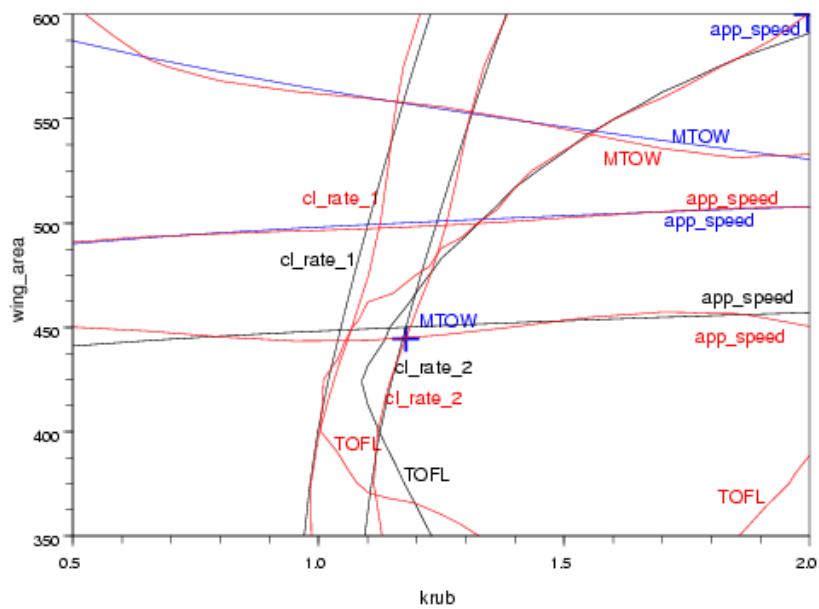


Figure 4.17: Result of the optimisation using the approximate functions and the SMAC model

Chapter 5

Mathematical help in decision-making

Contents

5.1	Introducing robustness	185
5.1.1	Short state-of-the-art	186
5.1.2	Uncertainty propagation, an emergent technique in FPO tools	188
5.1.3	Feasibility robustness	191
5.1.4	Robustness: a new criterion	206
5.2	Exhibiting model internal relations	207
5.2.1	FPO need	207
5.2.2	Why ellipsoids?	208
5.3	Conclusion	214
5.4	Going further	215

Until now, we solved our aircraft sizing problem as a constrained, global and multiobjective optimisation problem. As for the need for compromise solutions, the risk quantification has become more and more necessary. The chosen configuration must be stable to small disturbances of the considered hypotheses, coming from changes in the needs of the customers, in the requirements, from uncertainty on the intrinsic parameters of the evaluation function or from the model itself. By managing uncertainty in the early phases of the sizing process, we can improve both the quality of the results and the overall response time.

Nowadays, the typical approach is deterministic and the output quality is not quantified. Risk is covered with margins that are figured out using sensitivity studies. Robustness is estimated manually, calculating partial derivatives with finite difference methods around design points.

As this study is a preliminary stage in introducing uncertainty in aircraft sizing studies, we use a simplified model of aircraft to simulate aircraft sizing, the USMAC. This model, as it is described in the first chapter in paragraph 1.3.2.2 page 45, has been adapted to perform uncertainty propagation.

Today, the properties acting as cost functions are related to weights, operating costs or fuel consumption. From a global point of view, all these criteria tend at pushing some design characteristics in the same direction, namely minimising the wing area for instance. Thus, most of the time, the optimum of the deterministic optimisation is located on the boundary of the admissible set. In practice, if we take the uncertainty into account, we find that this optimum has a weak probability to really satisfy these constraints. Consequently, a pure cost-optimised design point may present a high industrial risk.

Introducing robustness will help us to increase likelihood of meeting requirements. In fact, robustness tends in a global point of view at pushing the design characteristics in the opposite direction than the classical criteria. For instance, the design points will have typically larger lifting areas and higher thrust.

Knowing that the results of the performance evaluation contain uncertainty, we need to be confident in the design point resulting from these calculations. Thus, we aim at introducing the robustness as a new criterion to handle risks.

Until now, in the previous chapters, we supported engineers in defining a preliminary aircraft design that answers requirements like those we can find in TLARs (see the definition in chapter 1, paragraph 1.1.1 page 9). During this study, first we found feasible designs of aircraft that answer the initial question of fulfilling TLARs, then we optimised criteria with a robust method, and we produced compromise solutions that are equivalent in solving the aircraft sizing problem.

In this chapter, after introducing uncertainties in the FPO processes, we try to go further to support engineers making decisions, thanks to the methods and tools we can implement and test during this study. The FPO role, after defining a preliminary design of aircraft, is to help the specialised departments which work on the detailed design, to be consistent when modifying their part of the aircraft.

The idea is to represent the correlations between design parameters, inside

the design space. For that purpose, we work with ellipsoids, whose main axes represent these correlations. This way, engineers can control and bound the different variations that they can allow the specialised departments to use. They can also identify which disciplines are the most coupled.

The objectives of this last chapter are, in a first part, to introduce uncertainty in FPO tools and in a second part, to yield means to help the FPO engineers controlling the modifications they can allow the specialised departments to perform on the detailed design of the aircraft.

5.1 Introducing robustness

Up to now, data are represented with deterministic values. Deterministic optimisation techniques have been successfully applied to our aircraft sizing problem, in a simplified test case and on a real problem. However, it is recognised that there always exist uncertainties in any engineering system due to variations in design conditions, such as loading, material properties, physical dimensions of parts, and operating conditions.

An issue in simulation-based multidisciplinary design is that the uncertainties of one discipline may be propagated to another discipline through the linking variables, so that the final output from the integrated multidisciplinary system gets an uncertainty that is the sum of the uncertainties coming from all the disciplines [Du and Chen, 2000a].

[Du and Chen, 2000a] list three sources of uncertainties in simulation predictions. Suppose we have a simulation model F which maps the input variables x to the output z , *i.e.*

$$z = F(x, p)$$

where p is the model parameter vector. The sources of uncertainties are then coming from:

- the variability of input x , called **input parameter uncertainty**,
- the uncertainty due to limited information in estimating the characteristics of model parameters p , called **model parameter uncertainty**, and
- the uncertainty in the model structure F itself, referred to as **model structure uncertainty**.

There are other kinds of uncertainties that we will not consider here, like the representation of real figures thanks to a finite quantity of digits, or the algorithmic errors related to computer implementations, for instance stop conditions in iteration loops.

Deterministic approaches do not consider the impact of such variations. Moreover, deterministic optimisation lacks the ability to achieve specified levels of constraint satisfaction [Du and Chen, 2000c].

There are several methods to propagate and manage the effect of uncertainties, which are mainly classified in two categories:

- methods that require probability and statistical analyses, and
- methods that do not require such analyses.

We are now describing some of these methods, beginning by the easiest to employ.

5.1.1 Short state-of-the-art

5.1.1.1 Worst case analysis

Worst case analysis is a simplistic approach, which do not need the distribution of random variables to be given. This method assumes that all fluctuations may occur simultaneously in the worst possible combinations.

Assuming that uncertainty of all variables is represented by intervals,

$$[\bar{x} - \Delta x, \bar{x} + \Delta x,]$$

the effect of variations on a function F is estimated from a first order Taylor's development as follows:

$$\Delta F \simeq \sum_{i=1}^n \left| \frac{\partial F}{\partial x_i} \Delta x_i \right|. \quad (5.1)$$

According to [Du and Chen, 2000c], in most cases, the worst case analysis is conservative because it is unlikely that the worst cases of variable deviations will simultaneously occur.

But in the case of a function F that is non-monotonous inside the intervals $[\bar{x} - \Delta x, \bar{x} + \Delta x,]$, it is possible that ΔF does not cover the entire range of variation of the function. This method is certainly not adapted to our problem.

5.1.1.2 Extreme condition approach

The extreme condition approach was also developed to obtain an interval, or the extremes, of the final output from a chain of simulation models. The term *extreme* is defined as the minimum or the maximum value of the end performance corresponding to the given ranges of all the uncertainties of the simulation [Du and Chen, 2000b].

With this approach, the variable uncertainties are characterised by intervals,

$$[\bar{x} - \Delta x, \bar{x} + \Delta x].$$

Correspondingly, the outputs of the simulation models are described by intervals, $[z^{min}, z^{max}]$.

Optimisations are used to find the maximum and minimum of the outputs. For a given set of variables x , the function F such that $z = F(x)$ is minimised and maximised with x varying in its definition range:

$$\begin{aligned} \min & \quad F(x) \\ \text{s.t.} & \quad x \in [\bar{x} - \Delta x, \bar{x} + \Delta x] \end{aligned} \quad (5.2)$$

and

$$\begin{aligned} \max & \quad F(x) \\ \text{s.t.} & \quad x \in [\bar{x} - \Delta x, \bar{x} + \Delta x] \end{aligned} \quad (5.3)$$

Contrary to the worst case analysis approach, this method finds the variation range of the outputs for any kind of function, even the nonlinear ones. But to find these extreme values, we have to solve two optimisation problems, which may be computationally expensive.

5.1.1.3 Probabilistic formulation

Probabilistic formulations consider the probability of events to be satisfied. It should be greater than the user specified probability. A general probabilistic formulation can be expressed as follows:

$$P[F(x) \leq 0] \geq P_d \quad (5.4)$$

where P_d is the desired probability specified by the decision-maker.

If the distributions of all the variables x are known, the probability P of equation (5.4) can be obtained accurately by integrating the joint probability density function of the variables on the corresponding region.

But practically, it is very difficult, or even impossible, to get an analytical or numerical solution of this integration [Du and Chen, 2000c; Green et al., 2002] for several reasons:

- the joint probability density function is generally not known,
- the boundary over which the integral is to be evaluated is generally not known, and
- even when the previously listed points are known, the multidimensional integral itself is difficult to evaluate.

In the case where the analytical method is not applicable, simulation-based approaches, such as Monte Carlo simulation, are often used to obtain an estimation of the probability:

$$P[F(x) \leq 0] \approx \frac{1}{N} \sum_{i=1}^N I[-F(x_i)] \quad (5.5)$$

where:

- N is the number of sampled data,
- x_i are the sampled points over the range of the distribution of x ,

- $I[\cdot]$ is the Heaviside function, defined as

$$I[F] = \begin{cases} 1 & \text{if } F \geq 0 \\ 0 & \text{otherwise.} \end{cases}$$

Simulation methods are flexible for any type of distributions and the results are often of a high accuracy if a sufficient number of simulations is used.

5.1.1.4 Moment matching formulation

Probabilistic formulations can become computationally expensive because of the number of simulations needed to have an accurate estimation of the probability. The moment matching formulation is then used to reduce the computational burden [Du and Chen, 2000c].

The name of this method comes from the fact that it uses the first and second moments of statistical distributions. With this approach, if F is assumed to be normally distributed, the probability of the event $F \leq 0$ becomes:

$$P[F \leq 0] = \Phi\left(\frac{\mu_F}{\sigma_F}\right) \quad (5.6)$$

where:

- Φ is the cumulative distribution function of a standard normal distribution,
- μ_F and σ_F are respectively the mean value and the standard deviation of F .

The relation (5.4) is then equivalent to:

$$\mu_F + k \cdot \sigma_F \leq 0 \quad (5.7)$$

where $k = \Phi^{-1}(P_d)$. For example, when the desired probability P_d is equal to 0.90, the corresponding k is then equal to 1.2816.

There are other existing methods, like the most probable point, that are not described here but some details can be found in [DeLaurentis and Marvris, 2000; Du and Chen, 2002; Green et al., 2002; Du and Chen, 2005]; some of these papers focus on model structure uncertainty, while others only propagate parameter uncertainties. The last one deals with a multidisciplinary problem.

In the following, we focus on the model structure uncertainty; the impact of model parameter uncertainty has been studied during a training on the same USMAC-based problem using statistical approaches, Monte Carlo methods and analytical methods. Results can be found in [May, 2005].

The same kind of study can be done on input parameter uncertainty. All principles are the same as for model parameter uncertainty because model parameters p and input parameters x are playing the same role for our evaluation function $F = F(x, p)$.

5.1.2 Uncertainty propagation, an emergent technique in FPO tools

Robustness is a new notion in FPO. Currently, risk is covered with margins that are figured out using sensitivity studies. The propagation of input and model parameter uncertainty has been introduced during a master thesis [May, 2005], thus here, we will focus on model structure uncertainty.

Quantification of model structure uncertainty is more complicated compared to parameter, input and model, uncertainty [Du and Chen, 2000a]. Hence, for simplicity in this preliminary stage in introducing uncertainty, we decide to add to each deterministic variable a random variable ε representing the uncertainty. An additive error model is used to represent model structure uncertainty in this study, though its real form can be much more complicated.

The uncertainty variable ε is modeled all along the study thanks to normal distributions $\mathcal{N}(\mu, \sigma^2)$ with expectation μ and variance σ^2 . This assumption simplifies the implementations and helps quantifying the output uncertainty.

This uncertainty is propagated through the calculation process to compute the uncertainty on performances using the First Order Second Moment method, further denoted FOSM, like in [Green et al., 2002], who applied this method on aircraft analysis and preliminary design.

Thus, if in the deterministic process, to calculate the performance y knowing the design point x , we have:

$$y = F(x, p), \quad (5.8)$$

now considering all kinds of uncertainty, we have:

$$y + \varepsilon_y = F(x + \varepsilon_x, p + \varepsilon_p) + \varepsilon_F \quad (5.9)$$

considering x , p and y as deterministic variables.

Then, using first order Taylor's developments, we have:

$$y + \varepsilon_y = F(x, p) + \frac{\partial F}{\partial x}(x, p) \cdot \varepsilon_x + \frac{\partial F}{\partial p}(x, p) \cdot \varepsilon_p + \varepsilon_F. \quad (5.10)$$

Hence, we have

$$\varepsilon_y = \frac{\partial F}{\partial x}(x, p) \cdot \varepsilon_x + \frac{\partial F}{\partial p}(x, p) \cdot \varepsilon_p + \varepsilon_F. \quad (5.11)$$

If we consider that ε_y is modeled according to the normal distribution $\mathcal{N}(\mu_y, \sigma_y^2)$ with expectation μ_y and variance σ_y^2 , then we have:

$$\mu_y = \sum_i \frac{\partial F}{\partial x_i}(x, p) \cdot \mu_{x_i} + \sum_j \frac{\partial F}{\partial p_j}(x, p) \cdot \mu_{p_j} + \mu_F \quad (5.12)$$

and

$$\sigma_y^2 = \sum_i \left(\frac{\partial F}{\partial x_i}(x, p) \cdot \sigma_{x_i} \right)^2 + \sum_j \left(\frac{\partial F}{\partial p_j}(x, p) \cdot \sigma_{p_j} \right)^2 + \sigma_F^2. \quad (5.13)$$

To summarise, the following system is the formulation we will use to consider the uncertainty in our design process:

$$\begin{cases} y &= F(x, p) \\ \mu_y &= \sum_i \frac{\partial F}{\partial x_i}(x, p) \cdot \mu_{x_i} + \sum_j \frac{\partial F}{\partial p_j}(x, p) \cdot \mu_{p_j} + \mu_F \\ \sigma_y^2 &= \sum_i \left(\frac{\partial F}{\partial x_i}(x, p) \cdot \sigma_{x_i} \right)^2 + \sum_j \left(\frac{\partial F}{\partial p_j}(x, p) \cdot \sigma_{p_j} \right)^2 + \sigma_F^2 \end{cases} \quad (5.14)$$

Computationally speaking, uncertainty is propagated through the calculation process the same way that values are propagated through the process to obtain the output values.

Yet, when taking uncertainty into account, the input vector x is no longer containing deterministic values, but the first two moments are added to its components. The input is now the vector

$$\begin{pmatrix} x \\ \mu_x \\ \sigma_x^2 \end{pmatrix}.$$

The basic functions of the model of aircraft that we use here, the USMAC, are calibrated with statistical regressions, as it is explained in [May, 2005] and in the paragraph 1.3.2.2.3 page 47. Thanks to these regressions, we know the bias and the standard deviation of the basic functions of the model, which are propagated along the calculation process via the formulation in the equation (5.14).

For that purpose, the USMAC model has been adapted to have as inputs our three dimensional vectors, and to produce three dimensional vectors as outputs, with the same kind of components.

We want to consider both the robustness of design criteria and the robustness of design constraints, the aim is to make the product least sensitive to the potential variations without eliminating the sources of uncertainty. The robust optimisation objective is achieved by simultaneously "optimising the mean performance" and "reducing the performance variation", while maintaining design feasibility under variations, which is a critical part of the problem.

According to [Rai, 2006b], robust design is in essence multiobjective optimisation because of the presence of the additional objective, robustness, and the addition of the robustness criterion may result in an optimal solution that is substantially different from the one obtained without this criterion.

[Baghdasaryan et al., 2002] also use uncertainty propagation to perform model validation, *i.e.* they assess how accurately the mathematical model represents the real world. This is another application of uncertainty propagation that we will not use here, but that may be considered in the future.

According to [DeLaurentis and Marvris, 2000], from an industrial perspective, the goal of multidisciplinary design should primarily be to design a vehicle that satisfies the requirements, and then to determine the robustness of the design to changes in assumptions made along the way. In their paper, they focus on describing the formal design uncertainty model that has been developed for uncertainty propagation, and they take the example of a

supersonic transport aircraft.

Following the same methodology as when we were dealing with the deterministic multiobjective optimisation problem, we will focus first on the constraint satisfaction problem before introducing the criteria to optimise. Thus, we are now dealing with the feasibility robustness, before introducing the robustness as a new criterion in our multiobjective optimisation problem. All this work has been presented in [Badufle et al., 2006b].

5.1.3 Feasibility robustness

Feasibility robustness in design can be explained by the following definition:

Definition 5.1. [Du and Chen, 2000c]:

A design is described to have **feasibility robustness** if it can be characterised by a definable probability, set by the designer, to remain feasible relative to the nominal constraint boundaries as it undergoes variations.

With this definition, feasibility in robust design can be considered as the fact that the probability of constraints to be satisfied should be greater than the user specified probability.

According to [Rai, 2006b], the effect of uncertainties on constraint satisfaction is important from a reliability perspective. In this paragraph, we focus on reliability-based optimisation.

It is obvious that, compared with the deterministic feasible region, the size of the admissible set will be reduced under the robustness consideration. In addition, based on the above definition, we note that the degree of feasibility can be defined by the desired level of probability chosen by the decision-maker.

A general formulation for the feasibility robustness can be:

$$P[G(x) \leq 0] \geq P_d \quad (5.15)$$

where P_d is the desired probability for satisfying the constraint G set by the decision-maker.

Following the same procedure as in the deterministic optimisation, we split our problem to first focus on constraints, thus on feasibility robustness.

To reduce the computational burden, we use the **moment matching method** since we assume all variables to be normally distributed. The principle is that the decision-maker chooses the desired probability P_d for the satisfaction of the constraint G , and we deduce the following relation:

$$P[G(x) \leq 0] = P_d \iff G(x) + \mu_G + k \cdot \sigma_G \leq 0 \quad (5.16)$$

with $k = \Phi^{-1}(P_d)$, where Φ is the cumulative distribution function of a standard normal distribution.

We decided to test the impact of the different components of the uncertainty, the bias μ and the standard deviation σ , of the model structure uncertainty. By graphically representing these different components, we can observe the influence they have on each constraint.

The figure 5.1 shows the model structure uncertainty when we only consider the bias μ of the constraint uncertainty. To draw the constraints when considering the bias of the uncertainty, instead of representing

$$G(x) = 0,$$

as in the deterministic case, we draw

$$G(x) + \mu_G = 0.$$

The table 5.1 shows numerically the impact of the bias on the constraints calculated at the barycentre of the admissible population produced previously using Genetic Algorithms.

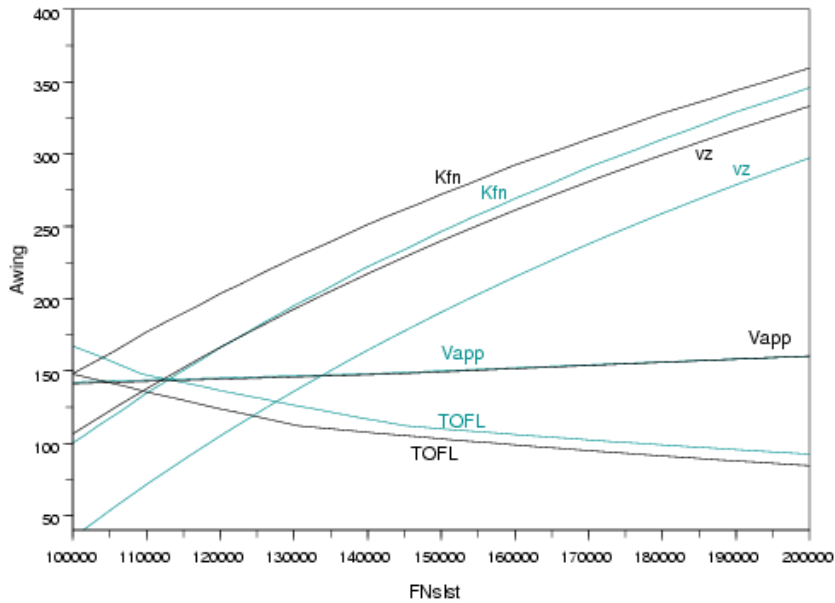


Figure 5.1: Influence of the bias of the model structure uncertainty. The deterministic constraints are represented in black, and the constraints considering the uncertainty are represented in blue.

We see in the table 5.1 that the bias has different influences on the constraints. The approach speed, V_{app} , has a bias close to zero, whereas the climb speed, v_z , has an important bias. This influence can be seen on the figure 5.1, where the deterministic constraints are represented in black, and the constraints considering the bias are represented in blue. We can observe on this figure that the two constraints representing the approach speed, V_{app} , have merged.

Constraint name	Deterministic values	Uncertain values	Relative differences
TOFL (m)	1127.14	1186.70	5.28%
Vapp (kt)	108.98	108.83	0.14%
vz (ft/min)	1501.47	1071.77	28.61%
Kfn (no dim)	0.68	0.74	9.88%

Table 5.1: Effect of the bias of the model structure uncertainty on the constraints calculated at the barycentre.

After considering the effect of the bias of the model structure uncertainty on the constraints of our problem, we now consider the effect of the standard deviation alone, without the bias. The figures 5.2 and 5.3 show the impact of the standard deviation caused by the model structure uncertainty on the constraints for the desired probabilities of 80% and 90%. To draw the constraints when considering the standard deviation of the uncertainty, we work like with the deterministic constraints, but instead of representing

$$G(x) = 0,$$

we draw

$$G(x) \pm k \cdot \sigma_G = 0,$$

with k defined depending on the desired probability set by the decision-maker, like in equation (5.7), and the sign of the relation is set according to the boundary of the constraint, if it is a minimal or a maximal bound.

The tables 5.2 and 5.3 show numerically the impact of the deviation on the constraints calculated at the barycentre, like in the previous table.

Constraint name	Deterministic values	Uncertain values	Relative differences
TOFL (m)	1127.14	1267.87	12.48%
Vapp (kt)	108.98	114.17	4.76%
vz (ft/min)	1501.47	957.72	36.21%
Kfn (no dim)	0.68	0.85	26.09%

Table 5.2: Effect of the standard deviation of the model structure uncertainty on the constraints calculated at the barycentre for a desired probability of 80% to reach the constraints.

Constraint name	Deterministic values	Uncertain values	Relative differences
TOFL (m)	1127.14	1341.43	19.01%
Vapp (kt)	108.98	116.88	7.25%
vz (ft/min)	1501.47	673.49	55.14%
Kfn (no dim)	0.68	0.95	39.72%

Table 5.3: Effect of the standard deviation of the model structure uncertainty on the constraints calculated at the barycentre for a desired probability of 90% to reach the constraints.

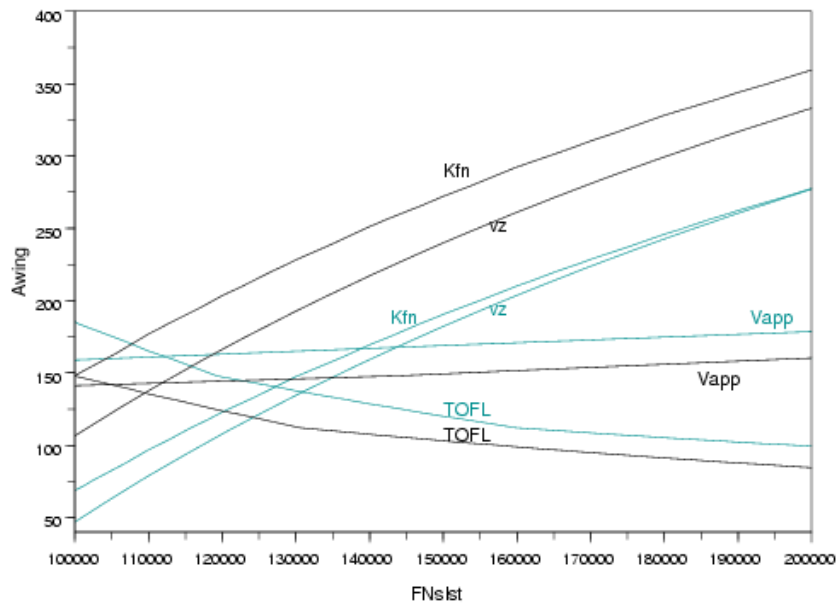


Figure 5.2: Influence of the standard deviation of the model structure uncertainty for a desired probability of 80% to reach the constraints. The deterministic constraints are represented in black, and the constraints considering the uncertainty are represented in blue.

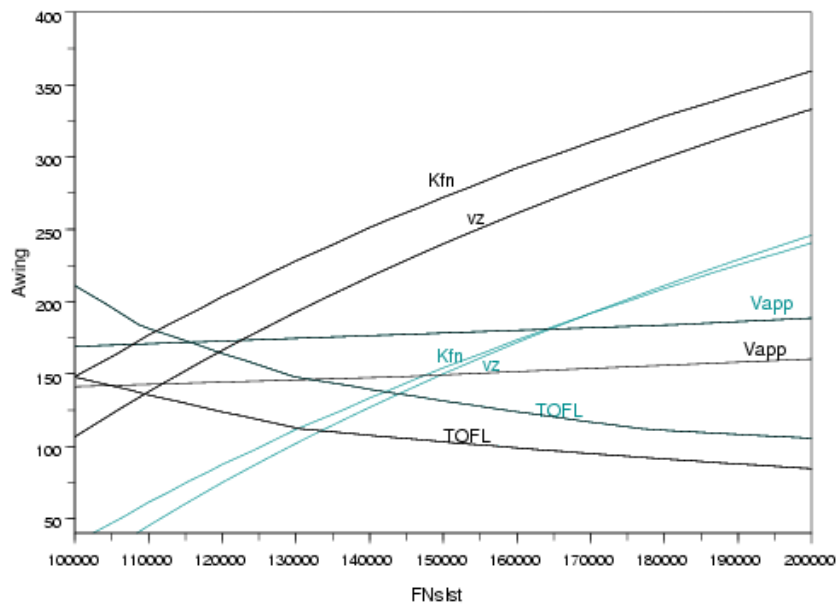


Figure 5.3: Influence of the standard deviation of the model structure uncertainty for a desired probability of 90% to reach the constraints. The deterministic constraints are represented in black, and the constraints considering the uncertainty are represented in blue.

We can see in the tables 5.2 and 5.3 that the standard deviation, like the bias, has different influences on the constraints. The approach speed, V_{app} , has a standard deviation relatively small compared with the standard deviation applied on the climb speed, v_z , which has finally an important bias and an important standard deviation. This constraint is very sensitive to uncertainty. This influence can be seen on the figures 5.2 and 5.3.

After taking into account separately the effects of the bias and of the standard deviation of the model structure uncertainty on the constraints of our problem, we now look at the effect of the two moments together. The figures 5.4 and 5.5 finally show the impact of uncertainty on the constraints for the desired probabilities of 80% and 90%. To draw the constraints when considering the bias and the standard deviation of the uncertainty, instead of representing

$$G(x) = 0,$$

as in the deterministic case, we draw

$$G(x) + \mu_G \pm k \cdot \sigma_G = 0,$$

with k defined like in equation (5.7), depending on the desired probability set by the decision-maker, and the sign of the relation is set according to the boundary of the constraint, if it is a minimal or a maximal bound.

The tables 5.4 and 5.5 show numerically the impact of the model structure uncertainty on the constraints calculated at the barycentre, like in the previous tables.

Constraint name	Deterministic values	Uncertain values	Relative differences
TOFL (m)	1127.14	1327.43	17.76%
V_{app} (kt)	108.98	114.02	4.61%
v_z (ft/min)	1501.47	528.02	64.83%
Kfn (no dim)	0.68	0.92	35.97%

Table 5.4: Effect of the model structure uncertainty on the constraints calculated at the barycentre for a desired probability of 80% to reach the constraints.

Constraint name	Deterministic values	Uncertain values	Relative differences
TOFL (m)	1127.14	1400.99	24.29%
V_{app} (kt)	108.98	116.73	7.10%
v_z (ft/min)	1501.47	243.79	83.76%
Kfn (no dim)	0.68	1.01	49.61%

Table 5.5: Effect of the model structure uncertainty on the constraints calculated at the barycentre for a desired probability of 90% to reach the constraints.

We can see in the tables 5.4 and 5.5 that the model structure uncertainty has also different influences on the constraints. The approach speed, V_{app} , has still a model structure

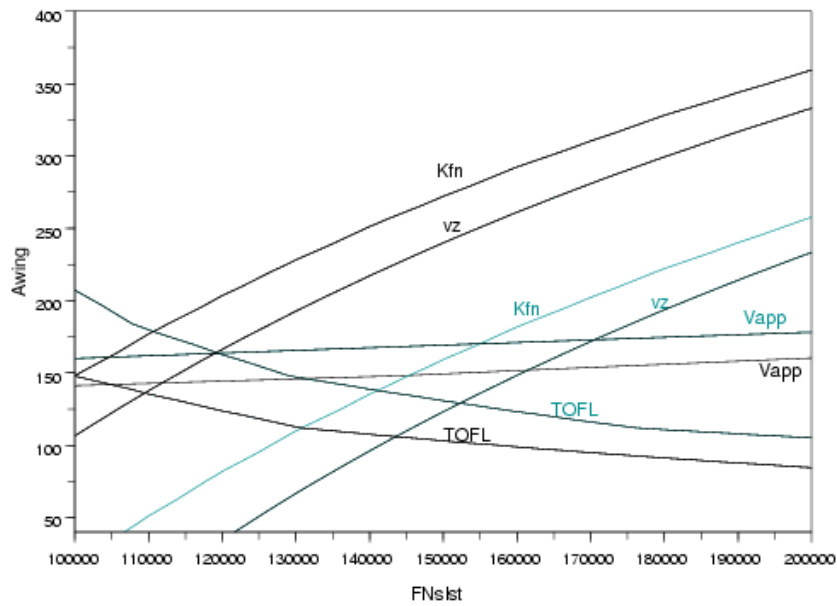


Figure 5.4: Influence of the model structure uncertainty on the constraints for a desired probability of 80% to reach the constraints. The deterministic constraints are represented in black, and the constraints considering the uncertainty are represented in blue.

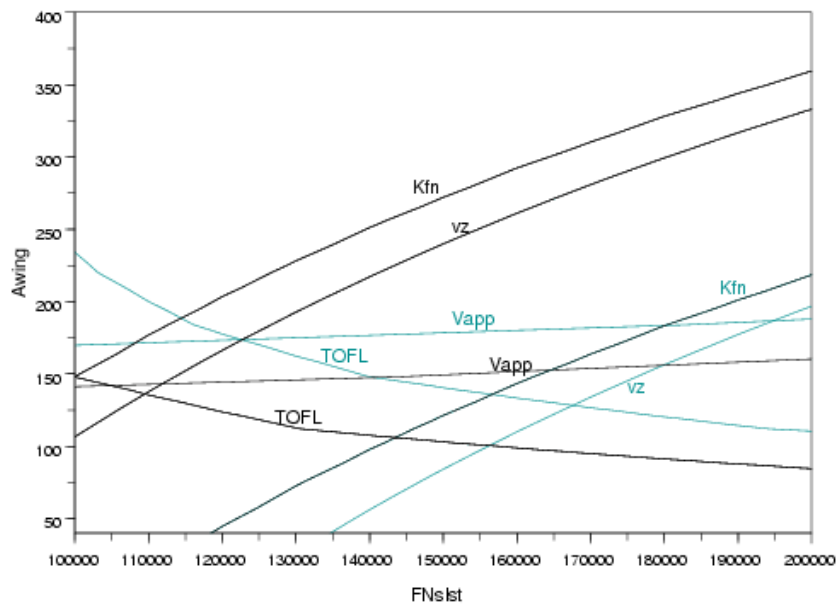


Figure 5.5: Influence of the model structure uncertainty on the constraints for a desired probability of 90% to reach the constraints. The deterministic constraints are represented in black, and the constraints considering the uncertainty are represented in blue.

uncertainty relatively small compared with the model structure uncertainty applied for instance on the climb speed, v_z , which has finally an important model structure uncertainty. This constraint is very sensitive to uncertainty, the two first moments are cumulating to give the final uncertainty for this constraint. This influence can also be seen on the figures 5.4 and 5.5.

Moreover, we observe on these figures that the admissible set is really reduced when considering the model structure uncertainty.

Remark 5.1. If we now consider the robustness with a desired probability of 90% as in the table 5.5, we see that the barycentre of our deterministic admissible population is not feasible anymore. Indeed, the constraints on the climb speed, v_z , and on the scaling factor on engine thrust, K_{fn} , are not satisfied. The value of the climb speed v_z should be greater than 500 ft/min, and the value of the scaling factor on engine thrust K_{fn} should be lower than 1.

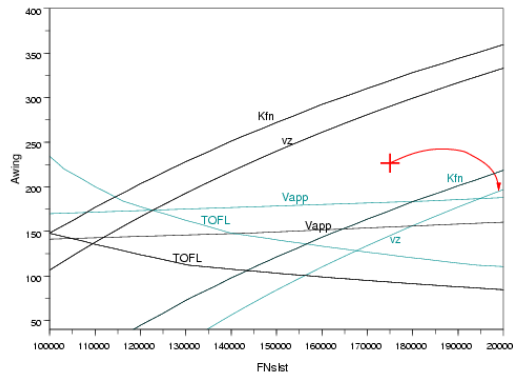


Figure 5.6: Illustration of the barycentre becoming non-admissible when willing to satisfy the constraints with the desired probability of $P_d = 90\%$

Remark 5.2. Considering the robustness has enlightened that some functions, here modeling constraints, are not precise, and that all the functions of the model have not the same precision. Thus, propagating the uncertainty allows us to know which are the functions that need to be reformulated to become of the same precision than the other functions.

Now that we have assessed the impact of the model structure uncertainty on the admissible set of our aircraft design problem, we work on the constraint satisfaction problem related to our optimisation problem, as we did in the chapter 2.

All that we have implemented with genetic algorithms to find admissible design points in the deterministic case can be re-used here to find the robust feasible set, according to the way the decision-maker defines it. We use the previous formula to build the penalty function that defines the criterion to optimise. The resulting set of admissible points is then robust.

Thus, as with the deterministic problem, the initial population is built using a Latin Hypercube Sampling, denoted LHS. An example of such an initial population can be seen in the figure 5.7.

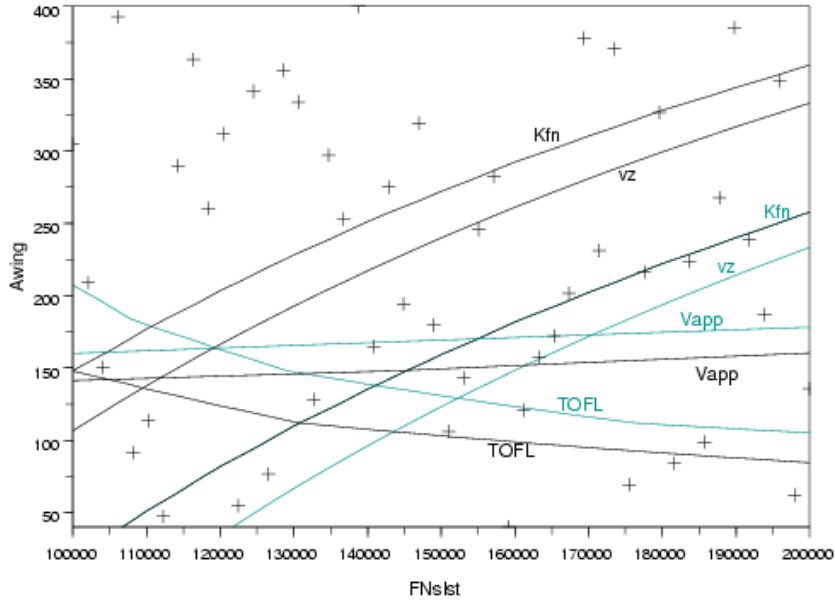


Figure 5.7: Initial distribution of points using a Latin Hypercube Sampling. One point is admissible when wanting the constraints to be satisfied with a probability of 80%.

Then, the points of the current population are driven to the admissible set using a penalty function of the violation of the constraints as the objective function of our dedicated implementation of a genetic algorithm, denoted GA. In the deterministic problem, this penalty function was built with the vector function $G(x)$, and now, when taking uncertainty into account, the penalty function is built using $G(x) + \mu_G \pm k \cdot \sigma_G$.

The admissible population produced for a desired probability of 80% can be seen in the figure 5.8.

The following step is now to test the mono-criterion and robust optimisation method we implemented in the chapter 2. We keep on working with the same problem, minimising the maximum take-off weight, MTOW, and the approach speed, Vapp.

Using the USMAC functions, the optimiser FSQP did not manage to calculate all the points it needed. FSQP has to calculate large amounts of points in the neighbourhood of the initial point before finding the descent direction of the gradient. The USMAC model adapted for uncertainty propagation is not robust enough, the calculations fail when trying to solve the Mass-Mission loop, which is described in paragraph 1.2.3.2 page 30. USMAC is not able to calculate all the points FSQP needs, thus we have to find another mean to make the process more robust.

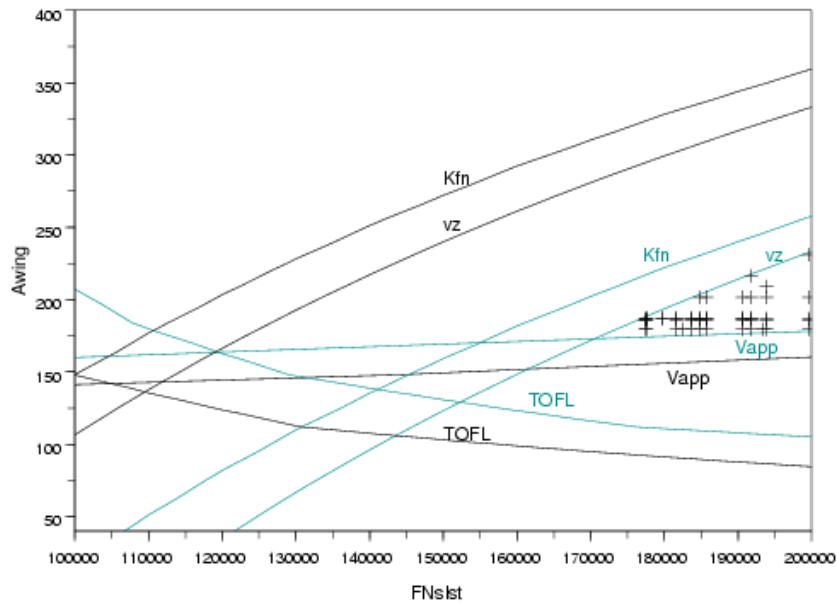


Figure 5.8: Admissible population using the model of aircraft which propagates uncertainty. The desired probability of reaching the constraints is of 80%.

To make the mono-criterion optimisation more robust, we decided to replace the objective and constraint functions by RBF networks as we did in the previous chapter. This time, the RBF networks replace the uncertain constraints, thus, we consider the constraint formulation $G(x) + \mu_G \pm k \cdot \sigma_G$ to be approximated.

To construct the RBF network, we project the admissible points we produced previously on the boundary of the feasible space when taking uncertainty into account. This is done using the GA which was adapted for that purpose in the chapter 4, this adaptation is described in paragraph 4.3.1 page 165.

A minimal distance is imposed between points when they reach the boundary of the admissible set. This minimal distance has to be adapted depending on the volume of the feasible set to have a well-distributed repartition of points on the boundary. The figure 5.9 shows the result we obtained when projecting points on the boundary of the admissible set when taking uncertainty into account.

Now that we have points on the boundary of the admissible set, we can construct the learning data basis to build the RBF network as it is done in the chapter 4.

The figures 5.10, 5.11 and 5.12 are illustrations of such RBF networks which will be used to replace the constraints, depending on the moment of the uncertainty we consider. We do not make the same numerical validation of the surrogate here because it was done in the previous chapter, we only see graphically that the results are reasonably accurate for our study, especially on the boundary of the admissible set, which is the most important.

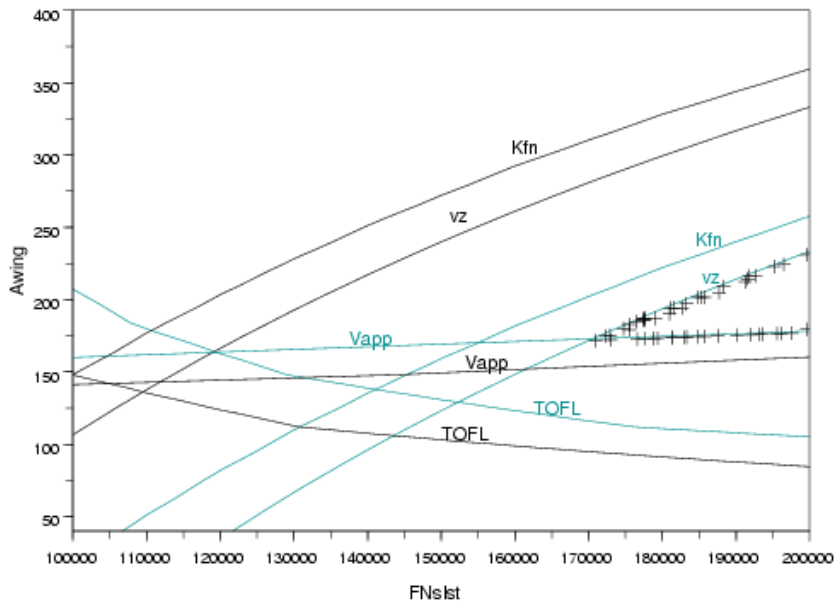


Figure 5.9: Projection of points on the boundary of the admissible set when taking uncertainty into account. The desired probability of reaching the constraints is of 80%.

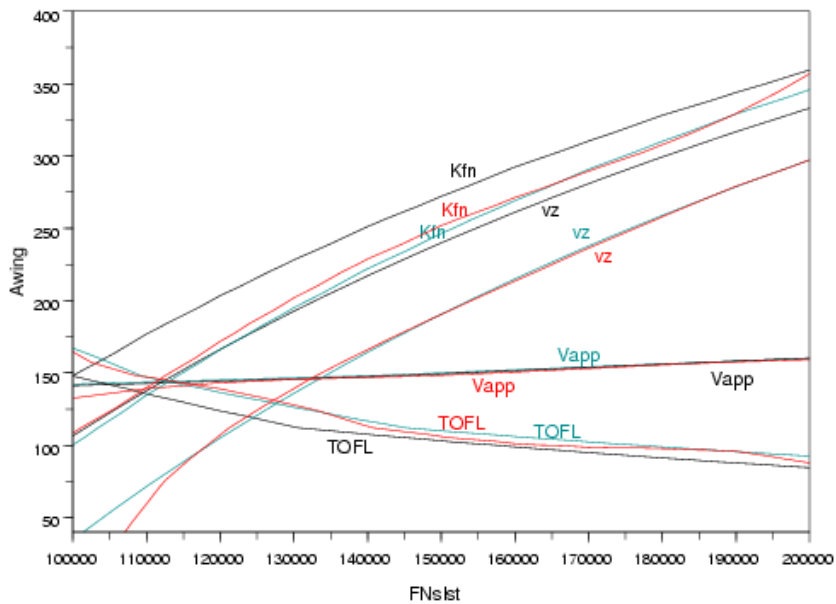


Figure 5.10: RBF network of the constraint when we consider the bias

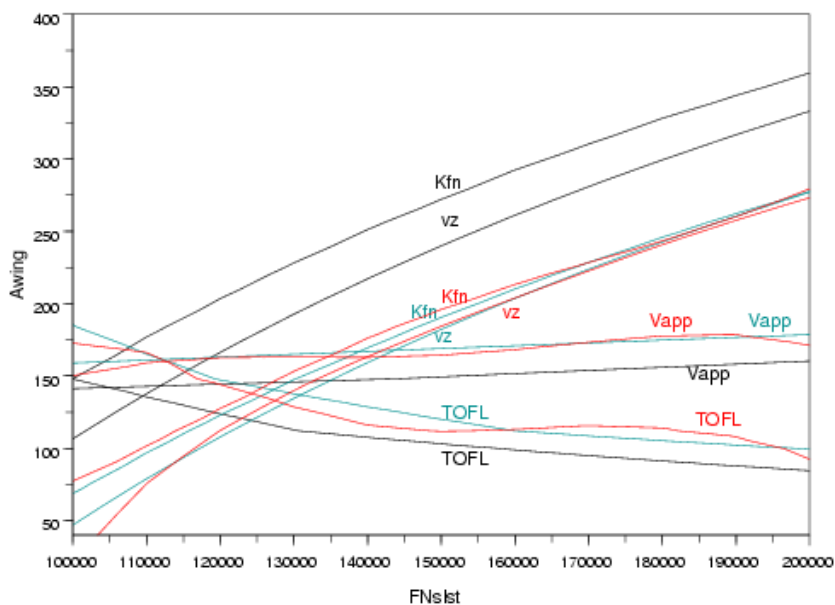


Figure 5.11: RBF network of the constraint when we consider the standard deviation, with a desired probability of 80% to satisfy the constraints.

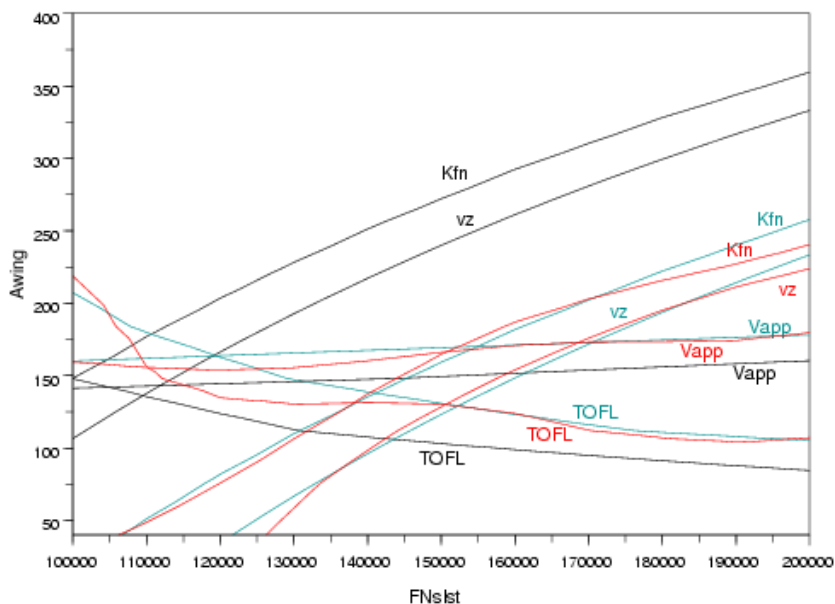


Figure 5.12: RBF network of the constraint when we consider the two moments, with a desired probability of 80% to satisfy the constraints.

Once the surrogate model of our evaluation function is built, we use it to perform the mono-criterion optimisation. The problem here is that, due to the lack of robustness of the adaptation of the USMAC model to handle uncertainties, we are not able to assess the quality of the approximated optima. These optima can be seen on the figure 5.13 and their numerical values in tables 5.6 and 5.7.

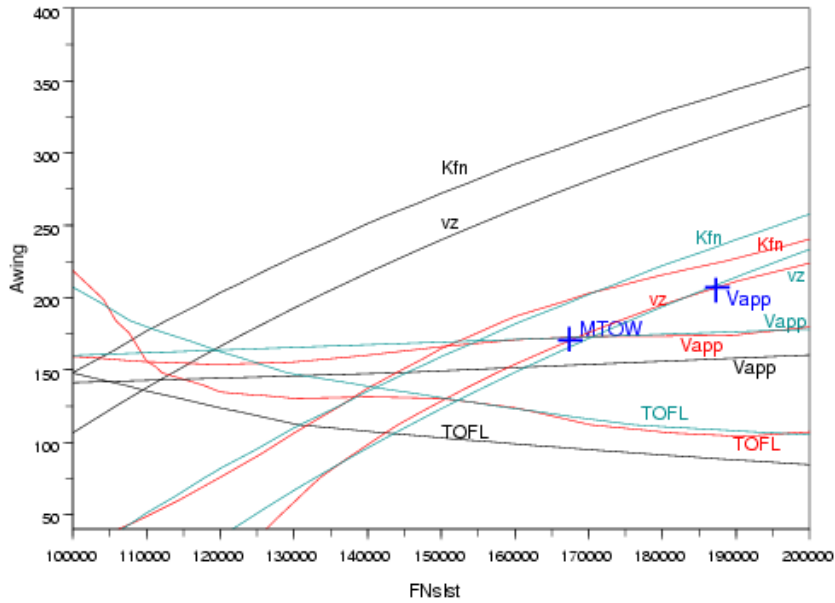


Figure 5.13: Mono-criterion optimisations of the maximum take-off weight, MTOW, and of the approach speed, Vapp, using RBF networks. The desired probability of reaching the constraints is of 80%.

	MTOW	Vapp
FNslst (daN)	16737.48	18726.65
Awing (m ²)	170.46	207.03

Table 5.6: Values of the degrees of freedom resulting from the optimisation considering uncertainty.

We observe graphically and numerically that the optimisation process reaches the right optimum for the MTOW criterion, which is located at the intersection of two constraints, the approach speed, Vapp, and the climb speed, vz.

Concerning the second criterion, each time we launched the optimisation process, it never succeeded in finding the right optimum, that we know to be located at the upper bound of the first degree of freedom, the sea level static thrust FNslst, and is limited by the constraint on the climb speed, vz.

Constraint name	MTOW		Vapp	
	Deterministic values	Uncertain values	Deterministic values	Uncertain values
TOFL (m)	1281.65	1525.02	1085.96	1274.3992
Vapp (kt)	114.54	120.14	106.99	111.83
vz (ft/min)	1445.40	464.32	1478.90	513.00
Kfn (no dim)	0.68	0.93	0.68	0.93

Table 5.7: Values of the constraints resulting from the optimisation considering uncertainty.

This result is not satisfying, especially as we wanted the optimisation to be more robust in finding the global optimum. We know that we have to work more on the tuning of the different parameters required in the implementation of FSQP, like the steps for searching the descent direction of the gradient.

Our aim here is not to perform robust mono-criterion optimisation as we did in the chapter 2, but to introduce the robustness as a new criterion in our multiobjective optimisation problem. Thus we decided not to spend time on this problem, but instead to go on with our multiobjective and robust optimisation problem.

Now that we found the optimum for the MTOW criterion in a deterministic case and when considering the uncertainty, we can compare the different results we obtained on the design parameter values and the difference on the criterion values. This comparison can be observed in the table 5.8.

	Deterministic case	Uncertain case	Relative differences
FNslst (daN)	11981.89	16737.48	39.68%
Awing (m ²)	149.72	170.46	13.85%
MTOW (kg)	80130.62	86336.35	7.74%

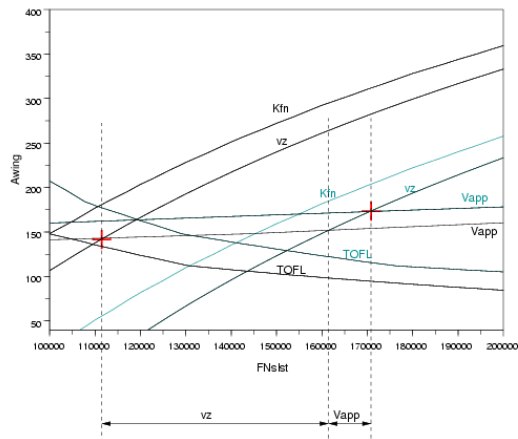
Table 5.8: Comparison of the results coming from the optimisation of the MTOW in a deterministic case and when considering uncertainty.

Ensuring the constraint satisfaction with a desired probability of 80% means increasing the optimal weight of roughly 8%, which is not so important compared to that, in the deterministic case, the probability of the operational constraints to be satisfied is weak. We did not calculate this probability in the deterministic case, but we can make the assumption that it is much lower than 50% because:

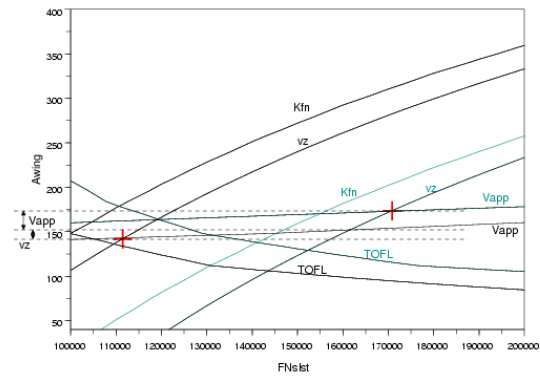
- the constraints are satisfied with the probability of 50% when the bias is considered,
- we can see graphically that the optimum in the deterministic case is located in a sharp angle of the admissible set.

On the figure 5.14, we can graphically make the observation of what causes the increase of the values of the degrees of freedom. Concerning the FNslst variable, the increase is

mainly due to the uncertainty on the climb speed constraint v_z , while concerning the A_{wing} variable, the increase is mainly due to the uncertainty on the approach speed constraint V_{app} .



Influence of the uncertainty on the FN_{slst} . It is mainly due to the uncertainty on the v_z



Influence of the uncertainty on the A_{wing} . It is mainly due to the uncertainty on the V_{app}

Figure 5.14: Influence of the uncertainty on the degrees of freedom

Before introducing the robustness as a new criterion, we tested our multiobjective optimisation process, as it is described in the paragraph 3.4 page 141, to find the Pareto front when we consider the feasibility robustness with a desired probability of 80% to satisfy the constraints.

The figures 5.15 and 5.16 are graphical representations of the results that we obtained after 30 generations when we work on our two-objective problem.

We can see that the points are quite well-distributed in the design and objective spaces. Our algorithm manages to find the Pareto front where points satisfy the constraints with the probability of 80%. The only issue is the calculation time. As expected, it is much longer than when we do not propagate uncertainty. It takes 1 second 15 to calculate one point when propagating uncertainty, instead of 0.05 second when we work on the deterministic problem. Thus, to find the Pareto front in an uncertain context, calculation time becomes really important. In our test case, calculating 30 generations took 1 hour and 30 minutes of CPU time on a Pentium 4 at 1.70 GHz.

We have tested all the processes that we implemented previously in a study case where we wanted to ensure that the constraints are satisfied with a desired probability of 80%. The next study will be to propagate the input and parameter uncertainty too, the aim being to obtain the total uncertainty of the final output. This problem is important because the design will be slightly modified during the detailed studies, but the constraints must be satisfied whatever the modifications are.

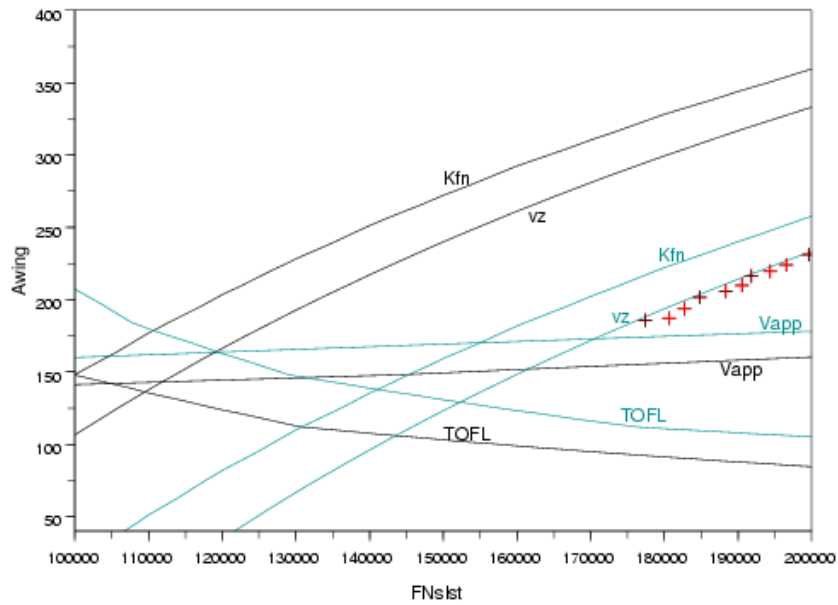


Figure 5.15: Representations of the points in the design space that are on the Pareto front.

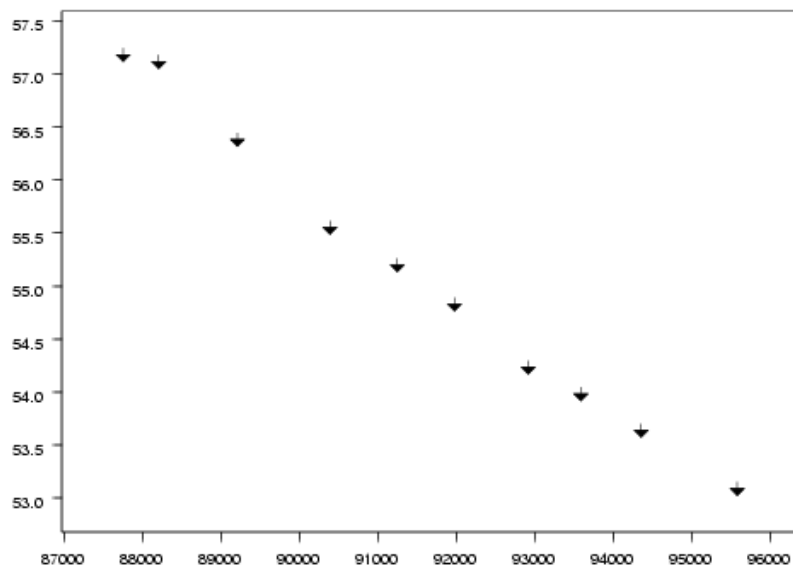


Figure 5.16: Representations of the points on the Pareto front.

We also want to take into account the uncertainty of the criterion, and this is the purpose of the following paragraph.

5.1.4 Robustness: a new criterion

Going back to the multiobjective optimisation problem, what we intend to do here is to introduce the robustness as a new criterion added to those we considered previously.

If the deterministic formulation is

$$\begin{aligned} \min \quad & F(x, p) \\ \text{s.t.} \quad & G(x, p) \leq 0 \end{aligned} \quad (5.17)$$

with $x \in \mathbb{R}^n$, $p \in \mathbb{R}^k$ and $F \in \mathbb{R}^p$, we now consider the optimisation problem

$$\begin{aligned} \min \quad & \begin{cases} F(x, p) + \mu_F \\ \sigma_F \end{cases} \\ \text{s.t.} \quad & G(x, p) + \mu_G + k \cdot \sigma_G \leq 0. \end{aligned} \quad (5.18)$$

Thus, we are simultaneously optimising the expected performance and minimising the performance variance, while ensuring robust feasibility.

Suppose we are still working on our two-objective optimisation problem. Considering the robustness as a new criterion, we have now four criteria, the two objectives which are the maximum take-off weight, MTOW, and the approach speed, V_{app} , and their variances, with still two degrees of freedom, the sea level static thrust, F_{Nslst} , and the wing area, A_{wing} . As expected, the problem we face now is that all the admissible points are part of the Pareto front.

Introducing the variances of the objectives as new criteria to be minimised means that we want to refine the precision of the results we obtain. In our test case, when dealing with two criteria (which means four criteria to optimise when considering the robustness), we need to have more than two degrees of freedom to have a meaningful problem.

Here, we are working with a simplified model of aircraft. By definition, refining the precision of the results obtained with a simplified model has no sense. The USMAC is thus not adapted for that purpose. Hence, we cannot perform this study with this model, it makes more sense to work on a realistic model of aircraft, which results are supposed to be less sensitive to structure model uncertainty. But such a model, allowing to propagate uncertainty in a more relevant way, is not available yet.

What we intend to do as a future work is to evaluate these methods on more complex models of aircraft, which are more realistic from geometrical and physical points of view, as soon as we will be able to propagate uncertainty with this kind of model. With these more complex models, we will be able to increase the number of degrees of freedom, of constraints and of criteria in the optimisation problem.

Performing optimisation when taking uncertainty into account was tested on a simplified model of aircraft in order to achieve a short calculation time. In the following, when dealing with realistic models, we will face the problem of important calculation time, and

certainly, we will have to generalise the idea of working with surrogate model and parallel implementation to perform the optimisation when propagating uncertainty.

During this first study in trying to introduce robustness in our design process, we decided to try and test Genetic Algorithms to solve this multiobjective optimisation problem, although we know there exist other multiobjective optimisation methods that can deal with robustness as well, like the evolutionary algorithm HAPEA [Lee et al., 2007], Physical programming [Chen et al., 1999; Messac and Ismail-Yahaya, 2002], or chaos polynomials [Molina-Cristobal et al., 2006]. As uncertainty management is gaining more interest in general design problems, these methods may be studied in further studies in FPO.

The aim of this last chapter of the manuscript is to help engineers in making decisions. The first part of this chapter was about uncertainty management, we wanted the preliminary aircraft design to be robust to uncertainties on the input variables, model parameters and the model structure itself.

The other point we want to treat here to help engineers in making decisions is to exhibit model internal links, which is the purpose of the next paragraph.

5.2 Exhibiting model internal relations

5.2.1 FPO need

Until now, all the processes we implemented help engineers in defining a preliminary aircraft design that answers requirements defined in TLARs (which are defined in paragraph 1.1.1 page 9). During this study, we first found feasible designs of aircraft that answer the initial question of fulfilling TLARs, then we optimised the criteria separately with a robust method, and we produced compromise solutions that are equivalent in solving the aircraft sizing problem. Finally in this chapter, we introduced uncertainties in the FPO processes to assess the robustness of the different solutions we produced.

In this paragraph, we go further to help engineers in driving the specialised departments, which work on the detailed design, when modifying their part of the aircraft, while ensuring that these modifications are consistent with the ones made by the other departments.

Currently, this is done by using sensitivity analyses on the main variables to know which are the ones having the most important influence on a criterion or a constraint. This kind of analyses can only be performed at a future project phase, as the models used in specialised departments are too complicated to make these analyses. Most of the time, it is approximated thanks to the engineers know-how on the problem.

We want here to make the process more systematic, by yielding a mean to help the FPO engineers in:

- anticipating the problems that may occur when working on the detailed design of the aircraft,

- determining the variables that are the most coupled, and
- controlling the modifications they can allow the specialised departments to perform on the detailed design of the aircraft.

The idea is to work with ellipsoids, whose main axes represent the correlations between design variables, inside the design space. This way, engineers can control and bound the different variations they can allow the specialised departments to use, and identify which disciplines are strongly coupled. Maybe other applications of ellipsoids will be found when working with them on this kind of problems.

5.2.2 Why ellipsoids?

There are two kinds of ellipsoids that we introduce:

1. the ellipsoid with the maximum volume that can be inscribed inside the admissible set, or the domain of our interest,
2. the ellipsoid whose main axes are *a priori* defined, and whose center minimises a criterion while its boundary is tangent to the admissible set boundary.

In both cases, determining these ellipsoids is an optimisation problem. For simplicity, we decided to approximate the set where the ellipsoids are fitted by a convex polyedral set. But all this can be generalised to any kind of convex set.

To approximate the convex set of our interest by a convex polyedral one, we use what we had already implemented in the previous chapter, in paragraph 4.3.1 page 165. We consider the points we projected on the boundary of the set of our interest, and with a Taylor's developments, we linearise the active constraints at some points. An illustration of what we obtain can be observed on the figure 5.17.

5.2.2.1 Ellipsoid with the maximum volume inscribed in a convex set

The first kind of ellipsoid we decided to inscribe inside the set of our interest is the one having the maximum volume.

This ellipsoid allows us to identify, in an easy manner, the inner shape and a rough approximation of the maximum volume of the set of freedom that is available. This can also be done more precisely thanks to multidimensional polyedral sets, but these polyedral sets are not easy and expensive to obtain, especially when willing to approximate a high-dimensional domain. Moreover, the needed geometrical information thus obtained with this approximation will remain difficult to be exploited.

In the case of ellipsoids, we know that some interesting theory exists to determine them in a convex set. The practical interest in determining a maximum volume ellipsoid that fits our feasible domain is that:

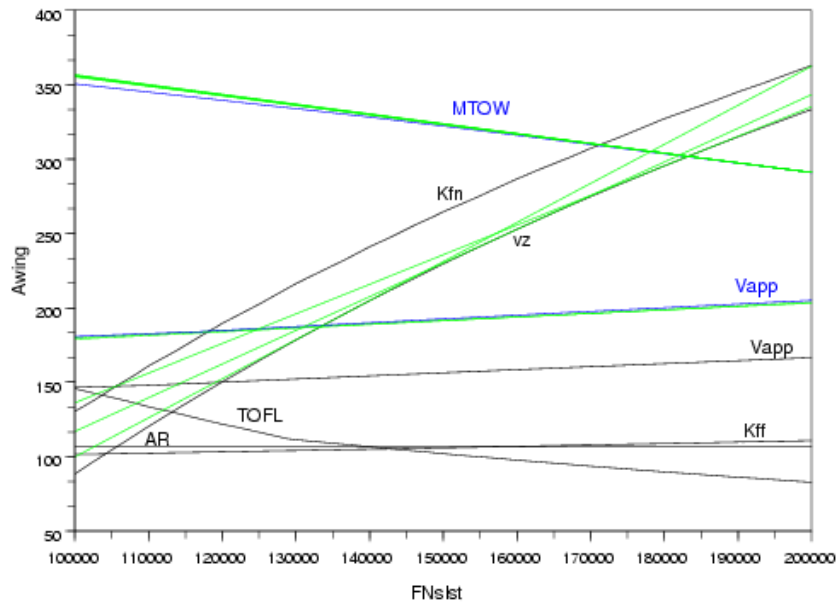


Figure 5.17: Illustration of the linearisation of the constraints. The envelop of the convex polyhedral set replacing the domain of our interest is represented in green.

- representations of our two-dimension problem seem trivial, but for more general studies with more than two dimensions, geometrical information is difficult to obtain. Ellipsoids give a geometrical information that can easily be exploited. For instance, the shape of the ellipsoid gives an idea of the correlations between design variables. If it is a sphere, the variables are relatively independent, and if it is flat in some directions, the corresponding parameters are correlated.
- It gives a first guess of the volume of the set of our interest, even if the angular parts of this domain are missing in the calculation of the volume.

Remark 5.3. In fact, the angular parts are of poor interest, they always correspond to non-robust part of the design space because of the proximity of at least two constraints. This lack of robustness in this part of the feasible set is due to probable changings in the initial aircraft sizing problem, or to the presence of uncertainties.

To determine this ellipsoid, we have to solve an optimisation problem, which formulation comes directly from the convex optimisation and analysis discipline [Hiriart-Urruty, 1998; Ben-Tal and Nemirovski, 2001].

Let P be a convex polyhedral set which is bounded and of a non-empty interior. We have:

$$P := \{x \in \mathbb{R}^n \mid \langle a_i, x \rangle \leq b_i \text{ for all } i = 1, \dots, m\} \quad (5.19)$$

where:

- a_i is a vector of \mathbb{R}^n , and
- b_i is a real number.

Then, let \mathcal{E} be an ellipsoid of \mathbb{R}^n . We can represent \mathcal{E} as

$$\mathcal{E} := c + \{Bu \mid \|u\| \leq 1\} \quad (5.20)$$

where:

- c is a vector of \mathbb{R}^n , and
- B is a (n, n) symmetric and positive definite matrix.

The volume of \mathcal{E} is proportional to the determinant of B , $\det(B)$.

The condition that the ellipsoid is included inside the convex polyedral set is a constraint of the optimisation problem. The formulation of this constraint is:

$$\begin{aligned} \mathcal{E} \subset P &\iff g_i(c, B) \leq 0 \\ &\iff \langle a_i, c \rangle + \|Ba_i\| - b_i \leq 0, \text{ because } B \text{ is symmetric.} \end{aligned} \quad (5.21)$$

With this definition, we know that g_i is convex.

Now, we have to define the objective function of our optimisation problem, which consists in maximising the volume of an ellipsoid. This volume is proportional to the determinant of the matrix defining the ellipsoid (this relation is deduced from the equation (5.20)). Thus, we have the following formulation to define the objective function, coming from the convex optimisation and analysis discipline [Hiriart-Urruty, 1998; Ben-Tal and Nemirovski, 2001]:

$$f(B) = -\ln(\det(B)). \quad (5.22)$$

Finally, the optimisation problem we have to solve (to find the ellipsoid having the maximum volume inside a given convex polyedral) is:

$$\begin{aligned} \min & \quad -\ln(\det(B)) \\ \text{s.t.} & \quad \langle a_i, c \rangle + \|Ba_i\| - b_i \leq 0 \text{ for all } i = 1, \dots, m \\ & \quad B \text{ definite positive.} \end{aligned} \quad (5.23)$$

To solve numerically this optimisation problem, we decided to work again with FSQP. The problem we met by directly solving the problem (5.23) with FSQP is that the logarithm makes the variations of the objective function too smooth.

Thus, we decided to replace the problem formulated in equation (5.23) by a simpler one, removing the logarithm:

$$\begin{aligned} \min \quad & -\det(B) \\ \text{s.t.} \quad & \langle a_i, c \rangle + \|Ba_i\| - b_i \leq 0 \text{ for all } i = 1, \dots, m. \end{aligned} \tag{5.24}$$

Nevertheless, this new problem still answers the question of finding the ellipsoid of maximum volume inside a convex polyhedra set, because the volume of the ellipsoid is proportional to the determinant of the matrix, with no logarithm.

Indeed, the logarithm of the determinant of the matrix defining the ellipsoid was introduced to make the objective function strictly convex. Thus, the problem formulated in equation (5.23) is a convex problem having a unique solution.

When removing the logarithm, we also removed the condition that B has to be definite positive, because this condition was set to ensure that the determinant had to be non-zero. Moreover, to describe numerically that the matrix B is definite positive is neither easy nor intuitive.

The figure 5.18 is an illustration of the result we obtained in our two-dimensional set bounded by the operational and the quality constraints.

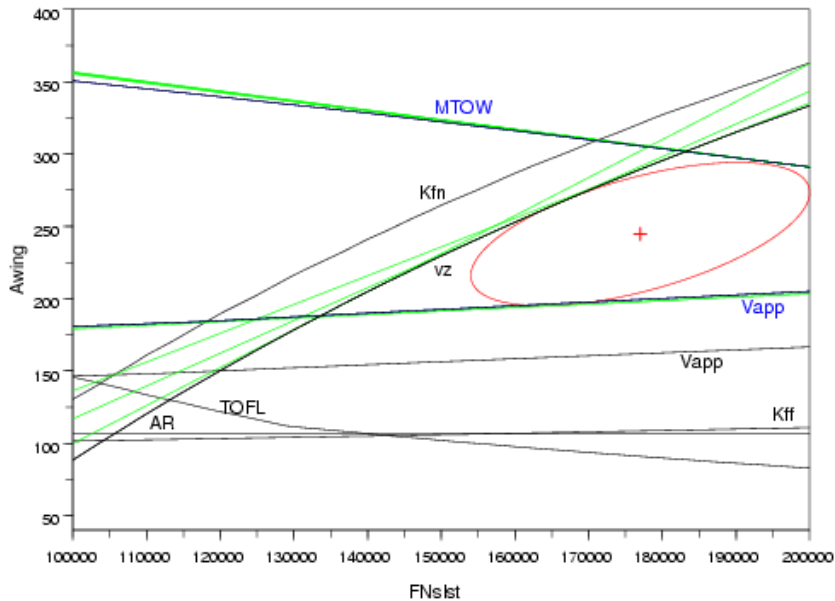


Figure 5.18: Ellipsoid of maximum volume included in the set of our interest

5.2.2.2 Fixed ellipsoid minimising a criterion

The second kind of ellipsoids we decided to include inside the set of our interest is an ellipsoid which main axes are *a priori* defined and fixed, and are also parallel to the directions of the axes of the degrees of freedom.

The practical interest of this second kind of ellipsoids we introduce here is that it allows us to *a priori* fix a set of freedom, or uncertainty, around design points to reduce the search space. Knowing the maximum volume of the set of our interest, as we defined it in the previous paragraph, the fixed ellipsoid should have a volume that is lower. Thus, a preliminary study with the ellipsoid of maximum volume that can be included inside the domain of our interest would be necessary for the decision-maker to choose if one wants to keep the entire domain, or to reduce it with the fixed ellipsoid.

In the case of the reduction of the domain with fixed ellipsoids, the main axes of these ellipsoids are set parallel to the axes of the degrees of freedom to avoid introducing artificial couplings between the parameters. It is of course possible to introduce these couplings, but we will not do it during this study to be able to understand the results, as it is a first attempt in dealing with ellipsoids.

This time, the optimisation problem consists in finding the centre of the ellipsoid such that it minimises one criterion, under the constraint that the boundary of this ellipsoid is tangent to the boundary of the set inside which it is included. The problem we face to solve numerically this optimisation problem is to ensure the constraint to be satisfied. Indeed, the direct method consists in discretising the boundary of the ellipsoid and to evaluate the aircraft sizing constraints at each point of the discretisation. If we want the results to be precise, the number of evaluations of points on the boundary of the ellipsoid becomes too large. Thus, we have to find another method to solve this problem, avoiding the evaluation of the aircraft sizing constraints.

To change this problem into the problem of just finding the location of the centre of the ellipsoid, the idea is to translate the constraints virtually in the direction of the centre of the ellipsoid when this fixed ellipsoid is tangent to this particular constraint. The length of this displacement is calculated to make the virtual constraint passing through the centre of the ellipsoid, in such a way that the optimised centre of the ellipsoid will be located on a displaced constraint, or at the intersection of displaced constraints. The figure 5.19 is an illustration to explain this method.

Moving the constraints in the direction of the centre of the ellipsoid when it is tangent to this constraint is not so easy. We do not know the direction of the displacement because we do not know at which point of the ellipsoid the constraint is tangent.

The idea is to transform the space where we work into a space where the ellipsoid becomes a circle of radius 1. Indeed, we want to use the fact that the vector normal to a plane that is tangent to a circle is also pointing towards the centre of the circle when it is located at the tangent point. Thus, when going back to the initial space, we will have the direction in which we have to move the constraint, and the length of this displacement is easy to calculate when having the direction.

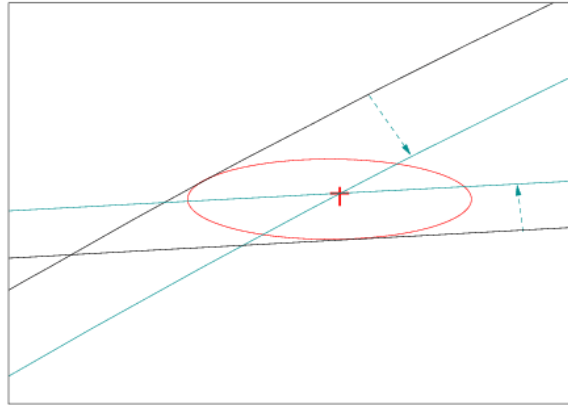
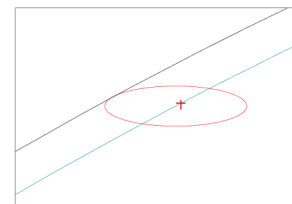
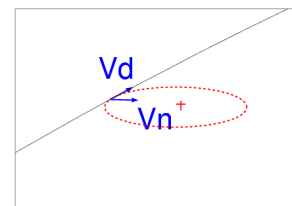
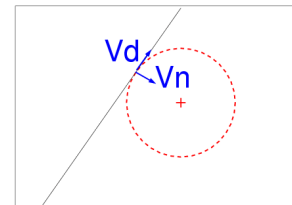
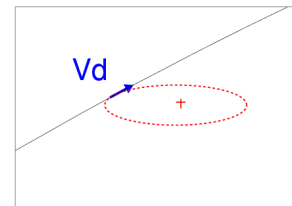


Figure 5.19: Illustration of the principle of the method to find the centre of the ellipsoid

The algorithm we implemented follows these steps:

1. Finding the directing vectors \vec{V}_d of the hyperplanes defining the convex polyedral set where we want to inscribe the ellipsoid.
2. Transforming the design space in such a way that the ellipsoid becomes a circle of radius 1.
3. Finding the vector \vec{V}_n that is normal to the transformed directing vectors \vec{V}_d of the constraints, \vec{V}_n is thus pointing towards the centre of the circle from the tangent point.
4. Making the inverse transformation to go back to the design space. The angles are thus changed, but \vec{V}_n is still pointing towards the center of the ellipsoid.
5. Calculating the length of the displacement. Repeat this operation for each hyperplane that linearises one constraint.
6. Optimising the criterion to find the centre of the ellipsoid.



Once again, we use FSQP to perform the optimisation. The figure 5.20 is an illustration

of the results we obtained on our two-objective optimisation problem.

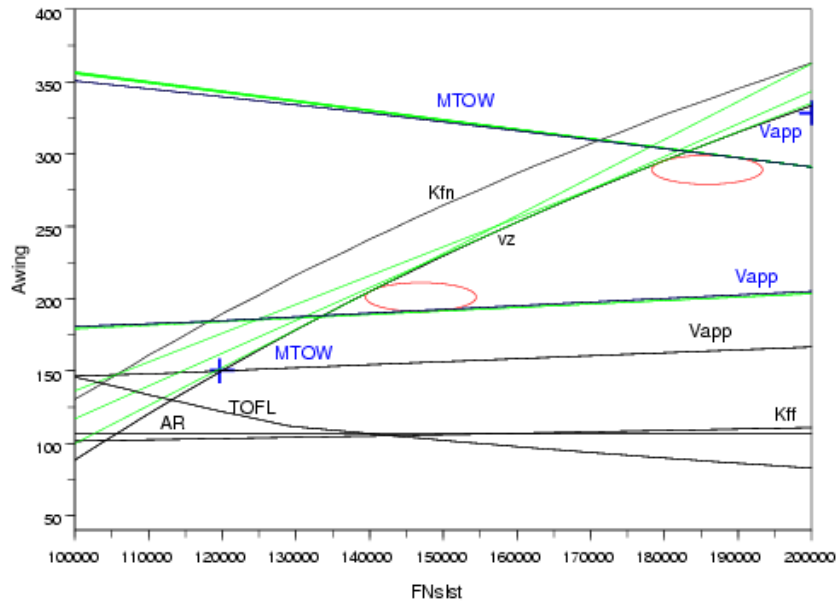


Figure 5.20: Ellipsoids with fixed main axes which centres minimise the criteria

To illustrate the use of these ellipsoids, we can imagine the following scenario: let's suppose that about fifteen degrees of freedom have been selected. Five of them are in the scope of Engine discipline, five others in the scope of Aerodynamic discipline, and the last five in the area of System architecture. Each of the disciplines are handled by different groups of people.

A first run has exhibited the inscribed ellipsoid of maximum volume and a second run, with targeted uncertainty, has exhibited a relevant design point. The characteristics of this design point are now supposed to be distributed to discipline departments to start the detailed design. Thanks to ellipsoid analysis, we can also provide guidelines for design parameter modifications. For instance, if the Engine group wants to move the value of one parameter which is strongly coupled with a parameter under the responsibility of the Aerodynamic group, the Engine group people already know what is their range of freedom if they want to play alone and they already know with which groups they have to play with if they need to go beyond their own independent limits.

5.3 Conclusion

In this chapter, we implemented in a first part a method that helps the engineers in quantifying the risk and in ensuring that the constraints are satisfied with the desired probability. This method uses the propagation of model structure uncertainty to assess the uncertainty on the constraint and criteria variables. What we intend to do as a future work is to ensure

the constraint satisfaction while optimising the criteria and minimising the uncertainty on these criteria.

In a second part, we introduced ellipsoids to represent the correlations between design parameters. The ellipsoid with the maximum volume that can be included inside the design space is a good approximation of the useful part of the search space. Obviously, some parts of the design space are not included in this ellipsoid, in particular sharp corners. But this is not synonymous to a loss of information because these sharp corners are not the parts of the design space where the optimum configuration has to be located. Indeed, we know that these sets are non robust parts of the search space.

5.4 Going further

Another tool that could be useful, to help engineers in using the methods we developed during this study, is a mean to understand the results obtained from the multicriteria optimisation. The result of this optimisation process is a set of multidimensional vectors that we need to represent in a two-dimensional map to extract all the available information it contains.

We know that such a tool can be developed by using self-organising maps, further denoted SOM, as it is described in [Pediroda and Poloni, 2006; Sasaki and Obayashi, 2004]. Such maps allow the decision-maker to identify groupings and relationships in the data, and further examinations may then reveal what features the members of a cluster have in common.

According to [Johnson and Rokhsaz, 2001], SOM is a neural network based on competition among neurons. It can be used in a pattern recognition system to order multiple input vectors according to their degrees of similarity. Such a map is a topology-preserving map, meaning that it associates each input vector with a single neuron and places similar input vectors near each other. The network is competitive in the sense that each neuron competes to be the one that corresponds to the input vector.

[Büche et al., 2002] uses SOM to adapt the mutation and the recombination operators in their implementation of a genetic algorithm. [Johnson and Rokhsaz, 2001] also use SOM for detecting airframe ice accretion.

Such a tool would be interesting to support engineers when they want to choose the best configuration, to their opinion, among the set of compromise solutions produced by the multicriteria optimisation process.

Conclusion

The aim of this study was to introduce new mathematical methods that can be useful in a future project sizing tool. During this study, we implemented methods like genetic algorithms, radial basis functions, moment matching formulation, and evaluated their utility in the context of aircraft sizing.

We contributed with this work in improving the optimisation methods that are currently used in the Future Project Office. Indeed, the lack of robustness of the optimisation processes is mainly due to a large design space that contains few feasible points. As the calculation of points that do not satisfy the constraints sometimes fails, the optimisation tools are not able to systematically find the optimum.

We first presented a new method, using genetic algorithms, to automatically produce large amounts of admissible design points, to be able to reduce the search space of the optimisation process to the admissible set. Our dedicated implementation of genetic algorithms solved the constraint satisfaction problem related to our aircraft sizing problem with a high success rate. Moreover, this was done in a short time frame, compared to the time engineers need to exhibit a single admissible point.

Then, we applied FSQP to solve the mono-criterion optimisation problem currently treated by the future project engineers. The research space was directly the admissible domain and each calculation succeeded in finding the global optimum. Hence, with our methodology, we managed to make the mono-criterion optimisation more robust.

In our aircraft sizing study, we have several criteria to consider, which are most of the time competing and conflicting measures of the system performance. Thus, one part of this work was to introduce some methods of multicriteria optimisation. We decided to keep working with genetic algorithms. We adapted the Non-dominated Sorting Genetic Algorithm to our problem and to make it answer our needs. With this algorithm, we managed to extract some points of the search space that were located on, or near the Pareto front of the multiobjective optimisation problem while satisfying the operational constraints, defined by the requirements related to the aircraft sizing problem, and the quality constraints, which were added to represent the knowledge of the engineers on this particular problem.

Thus, our multiobjective optimisation problem has been solved. Moreover, some knowledge on the problem has been added by the decision-maker to define the quality constraints. The resulting points of our methodology are thus located in a subset of interest for the future project engineers.

But our methodology revealed another kind of problem. Calculation times on a simplified model of aircraft were more important when we consider the uncertainty than in the deterministic studies. Thus, we supposed that the calculation times will be too long to widely use our algorithm on the models currently used in the Future Project Office to perform multi-criteria optimisation. Hence, we decided to consider another approach, by substituting the aircraft model by a surrogate model.

We identified several methods that can be useful to our aircraft sizing problem, like support vector machines or kriging interpolations. We decided to implement radial basis functions mainly because they are easy to implement and to adapt to a particular problem. Moreover, with radial basis function networks, we can easily fix a *a priori* given precision to reach for the approximation.

To construct the surrogate model, we fixed a precision to reach for the approximation, and we added the network one more basis function until this precision has been reached. The points we used as centres of the basis functions to build the approximation were the points obtained with the genetic algorithms, *i.e.* the initial population, the admissible population and the points that were projected on the constraints. Hence, we used the knowledge we obtained when making the previous studies.

The calculation times to compute the mono-criterion optimisations when using this approximation technique on a realistic model of aircraft were exactly the same as when using it on a simplified model of aircraft, and the errors introduced by the approximation were acceptable in the context of future project studies. Thus, the radial basis function network played its role of approximating the constraint and criteria functions with the required accuracy, and of making the optimisation faster.

The last part of this work consisted in introducing some mathematical tools in order to support the engineers of the Future Project Office when they have to give hints to the decision-makers. There are several methods that we wanted to implement and test during this work.

The first one was to introduce the notion of uncertainty related to the model of aircraft that is used to perform aircraft sizing. The aim was to assess the robustness of the design solutions of the aircraft sizing problem. Nowadays, the typical approach is deterministic. Risk is covered with margins and the robustness is estimated manually, by calculating partial derivatives with finite difference methods around design points. Most of the time, the optimum of the deterministic optimisation is located on the boundary of the admissible set.

In practice, when taking the uncertainty into account, we found that this deterministic optimum has a weak probability to really satisfy the active constraints. Consequently, introducing robustness helped us to increase likelihood of meeting the requirements. Finally, it appeared that the optimised maximum take-off weight had to be increased a bit to ensure that the constraints were satisfied with a high probability.

To be confident in the design point resulting from these calculations, we can also introduce the robustness as a new criterion to handle risks. This study has been envisaged in the framework of this study, but finally, it has not been performed here because the model of aircraft we used did not allow us to make this work. Indeed, introducing the robustness as a new criterion means minimising the variance of the criteria, and thus, refining the

precision of the results. The model of aircraft we used to propagate the uncertainty was a simplified model, thus refining its results had no sense and no realistic model of aircraft that propagates the uncertainty was available yet.

The second tool we wanted to implement was a mean which would support us to help the specialised departments which work on the detailed design, to be consistent when modifying their part of the aircraft. The idea was to represent the correlations between the design parameters, inside the design space, thanks to ellipsoids, whose main axes represented these correlations. This way, the engineers of the Future Project Office can control and bound the different variations they can allow the specialised departments to use, and identify which disciplines are the most coupled.

During this study, we experimented several techniques to treat the whole aircraft optimisation problem in the framework of the Future Project Office. We wanted the process to be more systematic in the exploration and the exploitation of the entire design space, but we also wanted to introduce multicriteria optimisation methods that are able to take the uncertainty of the models into account.

In the last part, we proposed a possible outcome of the integration of these different techniques. The aim was to yield the engineers a global and operational perception of the part of the design space that presents the greatest interest for them.

Of course, we know that we only explored a little part of all what it is possible to do in this context, but we worked on all the steps of the process, *i.e.* starting from a model of aircraft that is able to calculate its performance, we provided synthetic results of a global, multicriteria and robust optimisation.

In parallel to this work, another Ph.D. thesis is being performed on the same problematic of preliminary aircraft design. [Welcomme et al., 2006] presented a method based on self-regulating multi-agent system that answers the problem of multidisciplinary optimisation. They used the notions of cooperation and self-regulation offered by multi-agent systems to produce compromise solutions.

The next steps concerning this work is the industrialisation and the implementation. Step by step, all methods we tested during this study will be used by the engineers of the Future Project Office.

The first methodology we developed that will be industrialised is the research of the admissible space. Indeed, we showed in the second chapter that finding admissible design points was a difficult task. The aim is to use our method to generate the initial points starting classical mono-criterion optimisations.

The second industrialised method will be the approximation of the initial problem with surrogate models. In the chapter 4, we showed that using an optimiser on the entire aircraft sizing problem is costly in calculation time. The aim is to make the classical mono-criterion optimisations faster in approximating the optimum design point.

Another method that will be used by the engineers is the way we represent the correlations between design parameters inside the design space. This method will be a new tool

to support engineers in working with other specialised departments. Indeed, ellipsoids are a shared and an understandable representation of the correlations between design parameters, and also a good way to approximate the design space in which we work.

The advantage of all these methodologies is that the number of degrees of freedom can be increased up to 10, while it is currently about 3 or 4. This is an important improvement because the current way to perform the optimisation with 10 degrees of freedom is done manually by the engineers of the Future Project Office.

During this study, we implemented and tested some particular methods that can answer our aircraft sizing problem, like Genetic Algorithms or Radial Basis Functions. It would be interesting to compare the results we obtained here with what can produce other methods, like Particle Swarm optimisation, or Kriging interpolation. There are also some other tools that can be useful to support engineers in making decisions, like self-organising maps, as we mentioned in the chapter 5. This comparison could be done in a subsequent Ph.D. study.

The part of this work that is not planned to be industrialised at this stage is a part of the multicriteria optimisation. Initially, another part of the work that should have been done during this study was to perform aircraft sizing not only for one new configuration, but to design an aircraft family whose members share some components. Indeed, one way to reduce costs is to conceive a family of aircraft which share common parts and characteristics, satisfying different mission requirements [Willcox and Wakayama, 2002]. Traditionally, this has been achieved through the use of derivatives. The aim here is, as a future work, to optimise these shared components to design the aircraft family at once, in order to improve the common solution instead of having an optimum baseline aircraft and sub-optimum derivatives.

Using multicriteria optimisation takes fully its sense in this context of optimising an aircraft family at once. Indeed, in the chapter 3, it was not obvious to find relevant conflicting criteria to optimise. In this context, we would have one criterion to optimise, the direct operating cost, DOC, but for several aircraft. And minimising costs for one aircraft probably means degrading costs for the other members of the family. Then, working on optimising an aircraft family at once will be the main development that can be done on this study.

Appendix A

USMAC functions, detailed description

A.1 USMAC basic function signatures

A.1.1 Definition functions

A.1.1.1 Aerodynamics

```
function [Lref] = reference_length_(Awing,Ar,U_Lref)
function [Aref] = reference_area_(Awing,U_Aref)
```

A.1.1.2 Geometry

```
function [ar] = aspect_ratio_(span,Awing)
```

A.1.1.3 Lift over Drag

```
function [lod] = lift_to_drag_(cz,cx)
```

A.1.1.4 Criteria

```
function Kff = fus_fuel_ratio_(Fuel_nom,Fuel_max)
function FoPoNe6 = fuel_by_pax_by_NM_(Fuel,Npax,RA_eff)
function MWEoMTOW = MWE_over_MTOW_(MWE,MTOW)
```

A.1.2 Regulation functions

```
function kvs_T0 = Kvs_Take_Off_()
function kvs_LD = Kvs_Landing_()
```

A.1.3 Models

A.1.3.1 Environment

Earth function

```
function g = gravity_acc_(alt)
```

Atmosphere functions

```
function [Pamb,Tamb] = non_stand_atmos_(disa,alt)
function [Pstd,Tstd] = standard_atmosphere_(Alt)
function rho = air_density_(Pamb,Tamb)
function vsnd = sound_velocity_(Tamb)
function dTodZ = Tstd_gradiant_(alt)
```

A.1.3.2 Engine functions

```
function [Ksfc] = sfc_factor_(U_Ksfc)
function sfc = spec_fuel_cons_(BPR,Ksfc)
function [Kmt0] = max_take_off_factor_(U_Kmt0)
function [Kmc1] = max_climb_factor_(U_Kmc1)
function [Kmcrcr] = max_cruise_factor_(U_Kmcrcr)
function Fn = net_thrust_(Mach,rho,FNslst)
```

A.1.3.3 Geometry functions

```
function dnac = nacelle_diameter_(BPR,FNslst,U_dnac)
function dfus = fuselage_diameter_(NpaxFront,Naisle,U_dfus)
function lfus = fuselage_length_(Npax,NpaxFront,dfus,U_lfus)
function Npax = number_of_pax_(NpaxFront,lfus,dfus,U_lfus)
function [Aht] = tail_size_(Awing,Lref,LAht,Vht,U_Aht)
function [Avt] = fin_size_(Awing,span,LAvt,Vvt,U_Avt)
function [wAwing] = wing_wetted_area_(Awing,U_wAwing)
function [wAht] = tail_wetted_area_(Aht,U_wAht)
function [wAvt] = fin_wetted_area_(Avt,U_wAvt)
function [wAfus] = fus_wetted_area_(dfus,lfus,U_wAfus)
function [wAnac] = nac_wetted_area_(ne,dnac,U_wAnac)
function Fwing = wing_fuel_(Awing,tuc,U_Fwing)
function Fuel_max = max_fuel_weight_(Fwing,U_Fmax)
```

A.1.3.4 Aerodynamic functions

```
function [tuc] = thickness_o_chord_(tcr,tck,tct)
function [Kdiv] = mach_number_factor_(U_Kdiv)
function [Mdiv] = ref_mach_number_(cz,phi,tuc,Kdiv)
function [Kcxc] = press_drag_factor_(U_Kcxc)
```

```

function [cxc] = pressure_drag_(Mach,Mdiv,Kcxc)
function [Kcxi] = ind_drag_factor_(U_Kcxi)
function [cxi] = induced_drag_(cz,ar,span,dfus,Kcxi)
function [Kcx0] = fric_drag_factor_(U_Kcx0)
function [cx0] = friction_drag_(Mach,Pamb,Tamb,wAwing,wAht,wAvt,wAfus,...
    wAnac,lfus,Lref,Aref,Kcx0)
function [cx] = drag_factor_(cx0,cxi,cxc)
function [KczmTO] = Cz_max_TO_factor_(U_KczmTO)
function [czmax_TO] = Cz_max_TO_(phi,KczmTO)
function [KczmLD] = Cz_max_LD_factor_(U_KczmLD)
function [czmax_LD] = Cz_max_LD_(phi,KczmLD)

```

A.1.3.5 Mass functions

```

function Mwing = wing_mass_(MTOW,AWing,Lref,span,phi,tuc,U_Mwing)
function Mht = tail_mass_(Aht,U_Mht)
function Mvt = fin_mass_(Avt,U_Mvt)
function Mfus = fuselage_mass_(dfus,lfus,U_Mfus)
function Mgear = landing_gear_mass_(MTOW,U_Mgear)
function Mprop = engine_mass_(ne,FNslst,U_Mprop)
function Msys = system_mass_(MTOW,U_Msys)
function Mfurn = furnishing_mass_(Npax,U_Mfurn)
function Mop = operator_item_mass_(RA,Npax,U_Mop)
function MWE = manu_weight_empty_(Mfus,Mwing,Mht,Mvt,Mgear,Mprop,Msys,...
    Mfurn,U_MWE)
function OWE = ope_weight_empty_(MWE,Mop)
function LDW = landing_weight_(MTOW,Fuel)
function Wpax = one_pax_weight_(U_Pax_nom)
function Wpax = max_pax_weight_(U_Pax_max)
function PL = Payload_(Npax,Wpax)
function Fuel_total = Total_Fuel_(Fuel,Fuel_div)
function TOW = take_off_weight_(Fuel,PL,OWE)
function Payload = mission_payload_(TOW,Fuel_total,OWE)
function ZFW = Zero_Fuel_Weight_(PL,OWE)
function MLW = Max_Landing_Weight_(MZFW,U_MLW)
function mass = mean_cruise_mass_(MTOW,Fuel)
function [mass_ratio] = mass_ratio_(mass,ratio)
function PLeff = Payload_effective_(PLnom,PLmax,KPL) ;

```

A.1.3.6 Mission Model functions

```

function RA = range_(TOW,LDW,Mach,vsnd,g,lod,sfc,U_RA)
function fuel = fuel_(mass,leg,Mach,vsnd,g,lod,sfc)
function time = time_(RA,Mach,vsnd)

```

A.1.3.7 Handling Quality Model functions

```
function [Vht] = tail_volume_factor_(U_Vht)
function [LAht] = tail_lever_arm_(lfus,U_LAht)
function [Vvt] = fin_volume_factor_(U_Vvt)
function [LAvt] = fin_lever_arm_(LAht,U_LAvt)
```

A.1.3.8 Performance Model functions

```
function Mach = Mach_from_Vcas_(Pamb,Vcas)
function Vcas = Vcas_from_Mach_(Pamb,Mach)
function Veas = Veas_from_Vtas_(rho,Vtas)
function acc_mach = iso_mach_acc_(dTodZ,Tamb,disa,Mach)
function acc_cas = iso_cas_acc_(dTodZ,Tamb,disa,Mach)
function Mach_stall = Mach_stall_(mass,Aref,czmax_TO,Pamb,g)
function Mach = secured_Mach_(kvs_TO,Mach_stall)
function tofl = tofl_(Fn,Kmto,mass,czmax_TO,rho,ne,Aref,kvs_TO,U_tofl)
function vapp = app_speed_(LDW,czmax_LD,Aref,g,kvs_LD,U_vapp)
function pth = path_iso_cas_(mass,acc_cas,Fn,Kmcl,lod,g,ne,U_path)
function vz = clb_rate_iso_cas_(mass,Mach,acc_cas,Fn,Kmcl,lod,vsnd,g,...
    ne,U_vz)
function vz = clb_rate_iso_mach_(mass,Mach,acc_mach,Fn,Kmcl,lod,vsnd,...
    g,ne,U_vz)
function kfn = cruise_thrust_(mass,Fn,Kmcr,lod,g,ne,U_kfn)
function sar = cruise_VoC_(mass,sfc,lod,Mach,vsnd,g,U_sc)
function cz = level_flight_(mass,g,Mach,Pamb,Aref)
```

A.2 USMAC domain-level functions

A.2.1 Model functions

```
function [dfus,lfus] = geom_fus_(Npax,NpaxFront,Naisle,RV)
function [dnac] = geom_prop_(BPR,FNslst,RV)
function [Aht,LAht,Vht,Avt,LAvt,Vvt,Aref,Lref,ar] = geom_lift_(lfus,...
    Awing,span,RV)
function [wAwing,wAht,wAvt,wAfus,wAnac] = wetted_area_(dfus,lfus,dnac,...
    ne,Awing,Aht,Avt,RV)
function [Kmto,Kmcl,Kmcr,Ksfc] = engine_(BPR,RV)
function [Kcx0,Kcxi,Kcxc,Kdiv,KczmaxTO,KczmaxLD] = aerodynamic_(Awing,...
    span,phi,tuc,ar,RV)
function [MTOW_eff,MLW,MZFW,PL_eff,PL_max,PL_nom,OWE,MWE,Fuel_max,...
    Fuel_wing,bkdn] = mass_(MTOW,Fuel_total,KPL,Npax,RA,FNslst,...
    ne,dfus,lfus,Awing,span,phi,tuc,Aht,Avt,Lref,RV)
```

A.2.2 Operational Performance functions

```
function [RA_eff,RA_time,LDW,fuel_total,cz_mis,lod_mis,sfc_mis,fuel_div,...
        time_div,cz_div,lod_div,sfc_div] = mission_(TOW,Fuel,disa_mis,...
        alt_mis,Mach_mis,leg_div,alt_div,Mach_div,BPR,Ksfc,wAwing,wAht,...
        wAvt,wAfus,wAnac,lfus,dfus,ar,span,phi,tuc,Lref,Aref,Kcx0,Kcxi,...
        Kcxc,Kdiv,RV)
function [Y] = Point_A(Fuel)
function [Y] = Point_B(Fuel)
function [Y] = Point_C(X)
function [vz_clb] = iso_mach_climb_(mass_clb,disa_clb,alt_clb,Mach_clb,...
        FNslst,Kmcl,ne,wAwing,wAht,wAvt,wAfus,wAnac,lfus,dfus,ar,span,...
        phi,tuc,Lref,Aref,Kcx0,Kcxi,Kcxc,Kdiv,RV)
function [kfn_cth] = cruise_(mass_cth,disa_cth,alt_cth,Mach_cth,FNslst,...
        Kmcr,ne,wAwing,wAht,wAvt,wAfus,wAnac,lfus,dfus,ar,span,phi,tuc,...
        Lref,Aref,Kcx0,Kcxi,Kcxc,Kdiv,RV)
function [sar_crz,lod_crz,sfc_crz] = VoC_cruise_(mass_crz,disa_crz,...
        alt_crz,Mach_crz,BPR,Ksfc,wAwing,wAht,wAvt,wAfus,wAnac,lfus,...
        dfus,ar,span,phi,tuc,Lref,Aref,Kcx0,Kcxi,Kcxc,Kdiv,RV)
function [tofl] = take_off_(TOW,disa_to,alt_to,FNslst,Kmto,ne,phi,...
        KczmaxTO,Aref,RV)
function [vapp] = approach_speed_(LDW,alt_app,phi,KczmaxLD,Aref,RV)
```


Appendix B

Initial optimisation problem description

B.1 Degrees of freedom

Degrees of freedom of a civil aircraft configuration design are specified in the table B.1. They define the wing, the engines, the main landing gear and the control surfaces (horizontal and vertical). Most of them are continuous variables, and the variation intervals are given with lower and upper bounds. Variables annotated with the symbol * are discrete variables.

1	ER_EMP	0.135	sd	dn	[se]	Epaisseur relative a l emplanture	-1	422
1	ER_INT	0.0913	sd	dn	[se]	Epaisseur relative a la cassure 1	-1	425
1	ER_MED	0.0918	sd	dn	[se]	Epaisseur relative a la cassure 2	-1	427
1	YEXT	29	m	dn	[se]	Position en envergure de l extremite	-1	127
1	ER_EXT	0.0922	sd	dn	--	Epaisseur relative a l extremite	-1	
1	SREF	359.47	m2	dn	--	Surface de reference de la voilure	-1	
1	PHI25	34.41	deg	dn	--	Fleche de la voilure a 25% C.T.B.	-1	
2	YTPR	5.0001	m	dn	[se]	Position transversale du train principal	-1	197
1	K_FNC	0.85	sd	dn	--	Coefficient de caoutchoutage moteur	-1	
1	K_RTO	0.92	sd	dn	--	Coefficient de pousse sur le regime RTO	-1	
1	EINT_DYN	0.3448	uc	dn	[se]		430	430
1	EMED_DYN	0.65	uc	dn	[se]		-1	6
1	K_CBEC	0.71	uc	dn	--		-1	
0	NBCAD_XCEMP	0	uc	dn	--	.	-1	
0	NBCAD_CCEMP	10	uc	dn	--	.	-1	
0	EYM_INT_DYN	0.3793	uc	dn	[se]	.	-1	1

Figure B.1: Illustration of current degrees of freedom

B.2 Constraints

Constraints applied to the optimisation problem are mainly coming from the TLARs of the current study. They are operational or geometrical constraints. There are 21 constraints described in B.2

B.3 Criterion

The criterion to optimise is described in B.3

Description	Name	Lower bound	Current value	Upper bound
External wing sweep	PHI25	30°	34.4°	36°
Wing reference area	SREF	320	359.5	400
Wing half span	YEXT	25m	29m	33m
Internal kink spanwise relative position	EINT_DYN	29%	35%	40%
External kink spanwise relative position	EMED_DYN	60%	65%	70%
Driver for slat area	K_CBEC	0.5	0.71	1
Additional fuselage frames before the wing	NBCAD_XCEMP*	-2	0	+2
Number of frames corresponding to the wing box length	NBCAD_CCEMP*	8	10	12
Wing root tuc ratio	ER_EMP	12%	13.5%	15%
Internal kink tuc ratio	ER_INT	8%	9.1%	10%
External kink tuc ratio	ER_MED	8%	9.2%	10%
Wing tip tuc ratio	ER_EXT	8%	9.2%	10%
Engine rubbering factor	K_FNC	0.8	0.85	0.9
RTO throttle push	K_RTO	0.8	0.92	1
Engine spanwise relative position	EYM_INT_DYN	30%	38%	40%
Main landing gear spanwise position	YTPR	4m	5m	10m

Table B.1: Degrees of freedom

1	KLC25	0.515	sd	--	[se]	Proportion cabine a 25% CMA voilure	285	285
1	TM_CLIMB	29	min	--	[se]	Duree de la montee de la mission nominale	342	342
1	ALTP_PMM	37130	ft	--	[se]	Plafond de montee iso_MACH	383	383
1	ALTP_PBUF	43322	ft	--	[se]	Plafond de Buffeting	378	378
1	ALTP_PCR	37251	ft	--	[se]	Altitude plafond de croisiere	380	380
1	VCAS_VAPP	140.19	kt	--	[se]	Vitesse conventionnelle en approche	421	421
2	TETA_MAX	11.56	deg	--	[se]	Assiette maximale a cabre avion pose au sol	245	245
1	LP_DEC_RG_A	2773	m	--	[se]	Longueur de piste reglementaire au decollage	249	249
1	MARGE_FUEL	1.6151	uc	--	[se]		466	466
2	GARDE_SOL	0.8351	uc	--	[se]		436	436
1	JEU_EA_PORTE	2.5315	uc	--	[se]		438	438
0	BANK_ANGLE	7.9244	uc	--	[se]		437	437
1	SVBA_SREF	8.0437	uc	--	[se]		457	457
1	SAILRN_SREF	3.1509	uc	--	[se]		459	459
1	PINTLE_MARGIN	1.4541	uc	--	[se]		463	463
1	CBEC1E_CVOIL	0.12	uc	--	[se]		444	444

Figure B.2: Illustration of current constraints

1	DOC_AI	50722.4	\$/vol	--	[se]	Cout direct d exploitation / d.o.c. AI	226	226
---	--------	---------	--------	----	------	--	-----	-----

Figure B.3: Illustration of current value of the project criterion

Description	Name	Current value	Constraint
Cabine length ratio at 25% of MAC	KLC25	51.5 (%)	KLC25 =51.5
Time to climb for nominal mission	TM_CLIMB	29 (min)	TM_CLIMB \leq 29
Climb ceiling	ALTP_PMM	37130 (ft)	ALTP_PMM \geq 35000
Buffeting ceiling	ALTP_PBUF	43322 (ft)	ALTP_BUF \geq 35000
Cruise ceiling	ALTP_PCR	37251 (ft)	ALTP_PCR \geq 35000
Approach calibrated air speed	VCAS_VAPP	140.2 (kt)	VCAS \leq 140.2
Maximal pitch on ground	TETA_MAX	11.56 ($^{\circ}$)	TETA_MAX \geq 11.56
Regulatory take-off field length	LP_DEC_RG_A	2773 (m)	LP_DEC_RG_A $2750 \leq \cdot \leq 2800$
Fuel margin	MARGE_FUEL	1.6151 ()	MARGE_FUEL ≥ 1.6151
Ground clearance	GARDE_SOL	0.8351 (m)	GARDE_SOL ≥ 0.8351
Air inlet to door clearance	JEU_EA_PORTE	2.5315 (m)	JEU_EA_PORTE ≥ 2
Maximum bank angle on ground	BANK_ANGLE	7.9244 ($^{\circ}$)	BANK_ANGLE ≥ 6
Slat to wing area ratio	SVBA_SREF	8.0437 (%)	SVBA_SREF $8\% \leq \cdot \leq 9\%$
Aileron to wing area ratio	SAILRN_SREF	3.1509 (%)	SAILRN_SREF $3\% \leq \cdot \leq 4\%$
Rear spar clearance for landing gear integration	PINTLE_MARGIN	1.4541 (m)	PINTLE_MARGIN ≥ 0
Slat to wing chord ratio at internal kink position	CBEC1E_CVOIL	12 (%)	CBEC1E_CVOIL $12\% \leq \cdot \leq 14\%$

Table B.2: Constraints

Description	Name	Current value	Constraint
Direct operating cost	DOC_AI	50722\$ per flight	TO MINIMISE

Table B.3: Criterion

Appendix C

Graphical representations in three dimensions

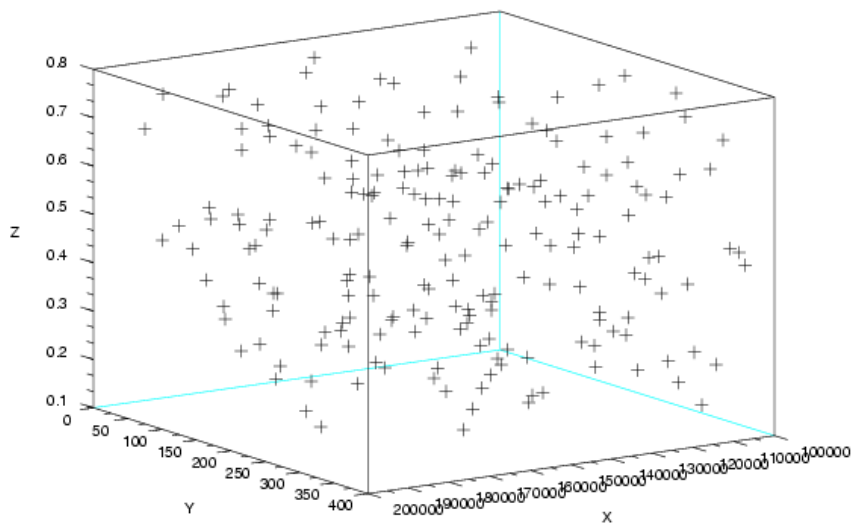


Figure C.1: Illustration of the initial population in 3 dimensions

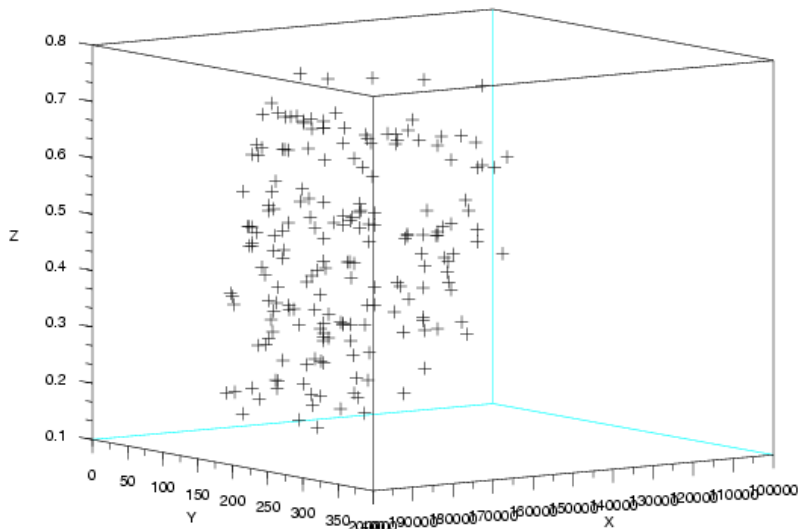


Figure C.2: Illustration of admissible population

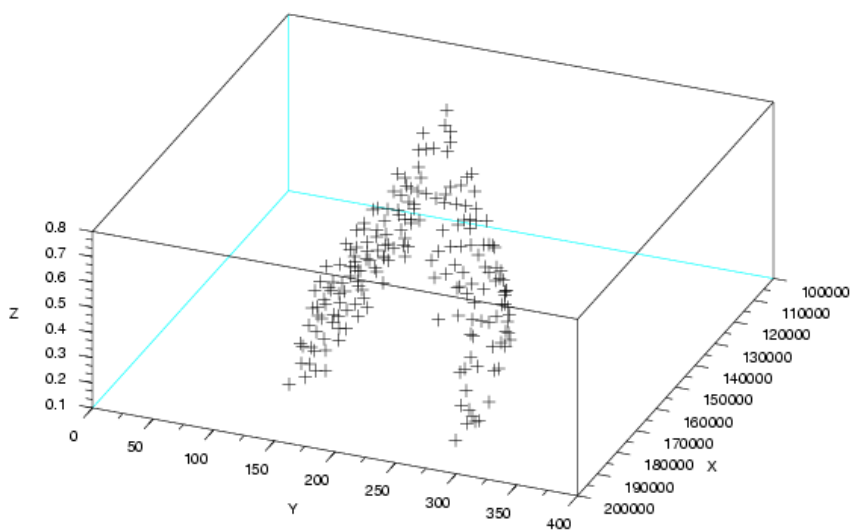


Figure C.3: Illustration of frontier of the admissible domain

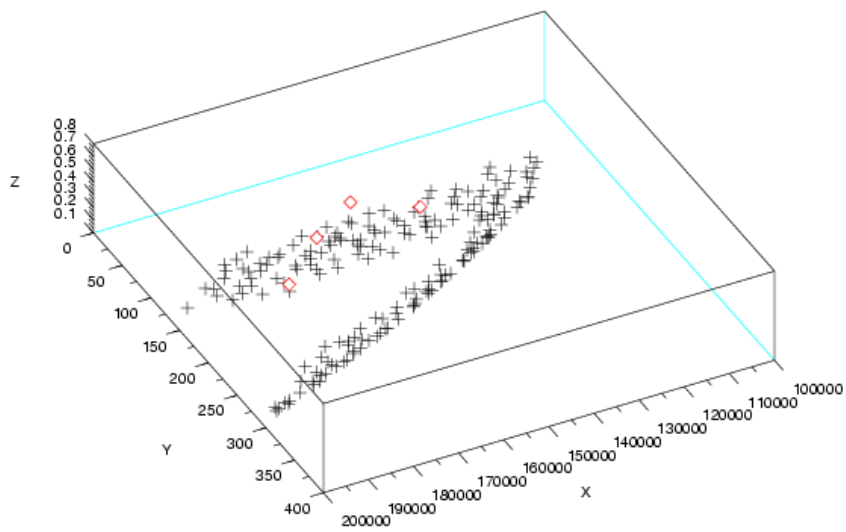


Figure C.4: Illustration of the result of the mono-criterion optimisations

References

- AIAA White Paper (1991). Current state-of-the-art in Multidisciplinary Design Optimisation. see also http://endo.sandia.gov/AIAA_MDOTC/sponsored/aiaa_paper.html. prepared by the AIAA Technical Committee for MDO, approved by AIAA Technical Activities Committee.
- Alexandrov, N., Dennis, J. E. J., Lewis, R. M., and Torczon, V. (1998). A Trust Region Framework for Managing the Use of Approximation Models in Optimization. *Structural Optimization*, 15(1):16–23.
- Alexandrov, N. M. (1995). Multilevel methods for MDO. In Alexandrov, N. M. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization, State of the art - Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, pages 79–89. SIAM.
- Alexandrov, N. M. and Lewis, R. M. (1999). Comparative properties of collaborative optimization and other approaches to MDO. Technical Report TR-99-24, NASA/ICASE.
- Alexandrov, N. M. and Lewis, R. M. (2000). Algorithmic Perspectives on Problem Formulations in MDO. In *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper 2000-4719.
- Alexandrov, N. M., Lewis, R. M., Gumbert, C. R., Green, L. L., and Newman, P. A. (2001). Approximation and Model Management in Aerodynamic Optimization With Variable-Fidelity Models. *Journal of Aircraft*, 38(6):1093–1101.
- Badulle, C. (2003). Simulation models of flight in future project conception. Master’s thesis, Université Paul Sabatier.
- Badulle, C., Blondel, C., Druot, T., and Duffau, M. (2005). Automatic satisfaction of constraints set in aircraft sizing studies. In Herskovits, J., Matorche, S., and Canelas, A., editors, *CD-Rom Proceedings of the 6th World Congress on Structural and Multidisciplinary Optimization*, Rio de Janeiro. ISSMO.
- Badulle, C., Blondel, C., Druot, T., and Duffau, M. (2006a). Refinement of Design Space in Aircraft Sizing Studies. In *Booklet of abstracts of the International Conference on Mathematics of Optimization and Decision Making*, Pointe-à-Pitre, Guadeloupe. SMAI, MODE, UAG.

- Badufle, C., Blondel, C., Druot, T., and Duffau, M. (2006b). Robustness Criterion in Aircraft Sizing Studies. In *Booklet of abstracts of the 7th International Conference devoted to Multi-Objective Programming and Goal Programming*, Tours, Loire Valley.
- Baghdasaryan, L., Chen, W., Buranathiti, T., and Cao, J. (2002). Model validation via uncertainty propagation using response surface models. In *Proceedings of DETC'2002, ASME 2002 Design Engineering Technical Conferences*.
- Balkin, S. D. and Lin, D. K. J. (2000). A neural network approach to response surface methodology. *Communications in statistics. Theory and methods*, 29(9&10):2215–2227.
- Bartholomew, P. (1998). The role of MDO within aerospace design and progress towards an MDO capability. In *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis anference*, pages 2157–2165. AIAA Paper 98-4705.
- Barzilay, O. and Brailovsky, V. (1999). On domain knowledge and feature selection using a support vector machine. *Pattern Recognition Letters*, 20(5):475–484.
- Ben-Tal, A. and Nemirovski, A. (2001). *Lectures on Modern Convex Optimization, Analysis, Algorithms, and Engineering Applications*. MPS-SIAM Series on Optimization.
- Benoudjit, N., Archambeau, C., Lendasse, A., Lee, J., and Verleysen, M. (2002). Width optimization of the Gaussian kernels in Radial Basis Function Networks. In *Proceedings of the 10th European Symposium on Artificial Neural Networks, ESANN'2002*.
- Beyer, H. and Deb, K. (2001). On self-adaptive features in real-parameter evolutionary algorithms. *IEEE Transactions on Evolutionary Computation*, 5(3):250–270.
- Booker, A. J., Dennis, J. E. J., Frank, P. D., Serafini, D. B., Torczon, V., and Trosset, M. W. (1999). A Rigorous Framework for Optimization of Expensive Functions by Surrogates. *Structural Optimization*, 17(1):1–13.
- Braun, R. and Kroo, I. (1995). Development and Application of the Collaborative Optimization Architecture in a Multidisciplinary Design Environment. In Alexandrov, N. M. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization, State of the art - Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, pages 98–116. SIAM.
- Buckley, M. J., Fertig, K. W., and Smith, D. E. (1992). Design Sheet: An environment for facilitating flexible trade studies during conceptual design. AIAA Paper 92-1191.
- Bäck, T. (1992). Self-adaptation in genetic algorithms. In Varela, F. J. and Bourgine, P., editors, *Towards a Prattice of Autonomous Systems: Proceedings of the First European Conference on Artificial Life*, pages 263–271, Cambridge, MA. MIT Press.
- Bäck, T., Hoffmeister, F., and Schwefel, H. (1991). A survey of evolution strategies. In Belew, R. K. and Booker, L. B., editors, *Proceedings of the 4th International Conference on Genetic Algorithms*, pages 2–9, San Diego, CA. Morgan Kaufmann.

- Büche, D., Milano, M., and Koumoutsakos, P. (2002). Self-Organizing Maps for Multi-Objective Optimization. In Barry, A. M., editor, *GECCO 2002: Proceedings of the Bird of Feather Workshops, Genetic and Evolutionary Computation Conference*, pages 152–155.
- Cantú-Paz, E. (1997). A Survey of Parallel Genetic Algorithms. Technical report, IlliGAL 97003, University of Illinois at Urbana-Champaign.
- Carr, J. C., Beatson, R. K., Cherrie, J. B., Mitchell, T. J., Fright, W. R., McCallum, B. C., and Evans, T. R. (2001). Reconstruction and representation of 3D objects with radial basis functions. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 67–76. ACM Press.
- Chabot, S. and Obin, P. (2003). AVION/FORTAB Reverse Engineering. Technical report, Airbus France.
- Chen, W., Sahai, A., Messac, A., and Sundararaj, G. J. (1999). Physical Programming for Robust Design. AIAA Paper-99-1206.
- Coello Coello, C. A. (2000). An updated survey of GA-based multiobjective optimization techniques. *ACM Computing Surveys*, 32(2):109–143.
- Coello Coello, C. A. (2001). A Short Tutorial on Evolutionary Multiobjective Optimization. In Zitzler, E., Deb, K., Thiele, L., Coello, C. A. C., and Corne, D., editors, *First International Conference on Evolutionary Multi-Criterion Optimization*, pages 21–40. Springer-Verlag. Lecture Notes in Computer Science No. 1993.
- Coello Coello, C. A. and Lechunga, M. (2002). MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization. In IEEE Press, editor, *Proceedings of the IEEE Congress on Evolutionary Computation, CEC '02*, volume 2, pages 1051–1056.
- Coleman, P., Gendre, P., Grihon, S., and Wacht, D. (2006). Conference feedback of the 2006 European-U.S. MDO colloquium. Technical report, Airbus.
- Collette, Y. and Siarry, P. (2002). *Optimisation multiobjectif*. Eyrolles.
- Craenen, B. G. W. and Eiben, A. E. (2001). Stepwise Adaption of Weights with Refinement and Decay on Constraint Satisfaction Problems. In Spector, L., Goodman, E. D., Wu, A., Langdon, W. B., Voigt, H.-M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M. H., and Burke, E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, pages 291–298, San Francisco, California, USA. Morgan Kaufmann.
- Craenen, B. G. W., Eiben, A. E., and Marchiori, E. (2001). How to handle constraints with evolutionary algorithms. In Chambers, L., editor, *The Practical Handbook of Genetic Algorithms: Applications, 2nd edition*, chapter 10, pages 341–361. Chapman & Hall/CRC, 2nd edition edition.

- Craenen, B. G. W., Eiben, A. E., and van Hemert, J. I. (2003). Comparing evolutionary algorithms on binary constraint satisfaction problems. *IEEE Trans. Evolutionary Computation*.
- Cramer, E. J., Dennis, J. E., J., Frank, P. D., Lewis, R. M., and Shubin, G. R. (1994). Problem formulation for multidisciplinary optimization. *SIAM Journal on Optimization*, 4(4):754–776.
- Cramer, E. J., Du, V. X., and Gablonsky, J. M. (2006). Multi-objective optimization for complex computer simulations. In *Proceeding of the 44th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA Paper 2006-729.
- Das, I. and Dennis, J. E. (1998). Normal-Boundary Intersection: A New Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems. *SIAM Journal on Optimization*, 8(3):631–657.
- Deb, K. (1999). Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230.
- Deb, K. (2001). *Multiobjective Optimization Using Evolutionary Algorithms*. Wiley.
- Deb, K., Agrawal, S., Pratab, A., and Meyerivan, T. (2000). A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In Schoenauer, M., Deb, K., Rudolph, G., Yao, X., Lutton, E., Merelo, J. J., and Schwefel, H.-P., editors, *Proceedings of the Parallel Problem Solving from Nature VI Conference*, volume 1917 of *Lecture Notes in Computer Science*, pages 849–858, Paris, France. Springer.
- Dechter, R. and Rossi, F. (2000). Constraint Satisfaction. *Encyclopedia of Cognitive Science*.
- DeLaurentis, D. and Marvris, D. (2000). Uncertainty Modeling and Management in Multidisciplinary Analysis and Synthesis. In *Proceedings of the 38th AIAA Aerospace Sciences Meeting*. AIAA Paper 2000-0422.
- Dennis, J. and Lewis, R. (1994). Problem Formulations and Other Optimization Issues in Multidisciplinary Optimization. Technical Report CRPC-TR94469, Center for Research on Parallel Computation, Rice University, Houston.
- Dennis, J. and Torczon, V. (1995). Managing approximation models in optimization. In Alexandrov, N. M. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization, State of the art - Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*. SIAM.
- Druot, T. (1993). Manuel de référence du logiciel fortab. Technical Report 474.0408/93, Airbus France.
- Druot, T. (1994). Aide à la formulation de problèmes numériques par l'utilisation de graphes bipartis. Technical Report 474.0357/94, Airbus France.

- Druot, T. (2002). Design oriented multidisciplinary integrated models of aircraft and object meshing modelling. Technical Report TN 0401949, Airbus France.
- Dréo, J., Pétrowski, A., Siarry, P., and Taillard, E. (2003). *Métaheuristiques pour l'optimisation difficile*. Eyrolles.
- Du, X. and Chen, W. (2000a). An efficient approach to probabilistic uncertainty analysis in simulation-based multidisciplinary design. In *Proceedings of the 38th AIAA Aerospace Sciences Meeting and Exhibit*. AIAA Paper 2000-0423.
- Du, X. and Chen, W. (2000b). Methodology for managing the effect of uncertainty in simulation-based design. *AIAA Journal*, 38(8):1471–1478.
- Du, X. and Chen, W. (2000c). Towards a better understanding of modeling feasibility robustness in engineering design. *ASME Journal of Mechanical Design*, 122(4):385–394.
- Du, X. and Chen, W. (2002). Efficient uncertainty analysis methods for multidisciplinary robust design. *AIAA Journal*, 40(3):545–552.
- Du, X. and Chen, W. (2005). Collaborative reliability analysis under the framework of multidisciplinary systems design. *Optimization and Engineering*, 6(1):63–84.
- Ehrgott, M. (2005). *Multicriteria Optimization*. Springer.
- Eiben, A. E. (2001). Evolutionary algorithms and constraints satisfaction: Definitions, survey, methodology, and research directions. In Kallel, L., Naudts, B., and Rogers, A., editors, *Theoretical Aspects of Evolutionary Computing*, Natural Computing series, pages 13–30. Springer, Berlin.
- Fan, H.-Y., Dulikravich, G. S., and Han, Z.-X. (2005). Aerodynamic data modeling using support vector machines. *Inverse Problems in Science and Engineering*, 13(3):261–278. AIAA Paper 2004-280.
- Fonseca, C. M. and Fleming, P. J. (1993). Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Genetic Algorithms: Proceedings of the Fifth International Conference*, pages 416–423. Morgan Kaufmann.
- Fonseca, C. M. and Fleming, P. J. (1995). An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation*, 3(1):1–16.
- Fritzke, B. (1994). Fast learning with incremental RBF networks. *Neural Processing Letters*, 1(1):2–5.
- Gao, Y. (2003). Population Size and Sampling Complexity in Genetic Algorithms. In *Proceedings of the Bird of a Feather Workshops(GECCO2003)—Learning, Adaptation, and Approximation in Evolutionary Computation*.
- Gauger, N. R. (2006). Adjoint approaches in aerodynamic shape optimization and MDO context. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.

- Giannakoglou, K. C. (2004). Neural Network assisted evolutionary algorithms in Aeronautics and Turbomachinery. In *VKI lecture series: Optimization Methods & Tools for Multicriteria/Multidisciplinary Design, Applications to Aeronautics and Turbomachinery*, Rhode-Saint-Genese, Belgium.
- Giannakoglou, K. C. and Karakasis, M. K. (2006). Hierarchical and Distributed Metamodel-Assisted Evolutionary Algorithms. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Giesing, J. P. and Barthelemy, J.-F. (1998). A summary of industry MDO applications and needs. In *Proceeding of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper 98-4737.
- Giunta, A., Watson, L., and Koehler, J. (1998). A Comparison of Approximation Modeling Techniques: Polynomial Versus Interpolating Models. In *Proceeding of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis & Optimization*. AIAA Paper 98-4758.
- Goldberg, D. and Deb, K. (1991). A comparison of selection schemes used in genetic algorithms. In Rawlings, G., editor, *Foundations of Genetic Algorithms*, volume 1, pages 69–93, Redwood City, CA. Addison-Wesley.
- Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley.
- Gonzalez, L. F., Périaux, J., DongSeop, L., Whitney, E. J., and Srinivas, K. (2006). MOO Methods for Multidisciplinary Design Using Parallel Evolutionary Algorithms, Game Theory and Hierarchical Topology: Practical Application to the Design and Optimisation of UAV Systems. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Gonzalez, L. F., Srinivas, K., Périaux, J., and Wong, K. C. (2005). Capture of Pareto Fronts and Nash Equilibrium Solutions of Multi-Criteria/Multidisciplinary Optimisation Problems in Aeronautics Using Evolutionary Computing and Game Strategies. In Uthup, B., Koruthu, S. P., Sharma, R. K., and Priyadarshi, P., editors, *Recent Trends in Aerospace Design and Optimization, Proceedings of SAROD-2005, December 2005*, pages 438–484, Hyderabad, India. Tata McGraw Hill.
- Grandine, T. A. (2005). The Extensive Use of Splines at Boeing. *SIAM News*, 38(4):3–6.
- Green, L. L., Lin, H.-Z., and Khalessi, M. R. (2002). Probabilistic methods for uncertainty propagation applied to aircraft design. In *Proceedings of the 20th AIAA Applied Aerodynamics Conference*. AIAA Paper 2002-3140.
- Guenov, M. D., Utyuzhnikov, S. V., and Fantini, P. (2005). Application of the modified physical programming method to generating the entire pareto frontier in multiobjective optimization. In Schilling, R., Haase, W., Périaux, J., Baier, H., and Bugeda, G.,

- editors, *Proceedings of EUROGEN 2005, 6th Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems*, Munich, Germany.
- Hiriart-Urruty, J.-B. (1998). *Optimisation et analyse convexe*. Presses universitaires de France.
- Holland, J. (1975). *Adaptation in Natural and Artificial Systems*. The University of Michigan Press.
- Horn, J., Nafpliotis, N., and Goldberg, D. E. (1994). A Niche Pareto Genetic Algorithm for Multiobjective Optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, IEEE World Congress on Computational Intelligence*, volume 1, pages 82–87, Piscataway, New Jersey. IEEE Service Center.
- INRIA Copyright (1989). Scilab, a free scientific software package. Description on <http://scilab.org/>.
- Johnson, M. D. and Rokhsaz, K. (2001). Using Artificial Neural Networks and Self-Organizing Maps for Detection of Airframe Icing. *Journal of Aircraft*, 38(2):224–230.
- Kodiyalam, S. (2001). Multidisciplinary Aerospace Systems Optimization – AeroSciences (CAS) Project. Technical Report CR-2001-211053, NASA.
- Koza, J. (1992). *Genetic Programming : on the Programming of computers by means of natural selection*. MIT Press.
- Krishnamurthy, T. (2003). Response surface approximation with augmented and compactly supported radial basis functions. In *Proceeding of the 44 th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*. AIAA Paper 2003-1748.
- Kroo, I. (1995). MDO for Large-Scale Design. In Alexandrov, N. M. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization, State of the art - Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, pages 22–44. SIAM.
- Kroo, I., Altus, S., Braun, R., Gage, P., and Sobieski, I. (1994). Multidisciplinary optimization methods for aircraft preliminary design. In *Proceedings of the 5th AIAA/NASA/USAF/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, pages 697–707. AIAA Paper 94-4325.
- Kroo, I. and Manning, V. (2000). Collaborative optimization: status and directions. AIAA Paper 2000-4721.
- Lawrence, C. T. and Tits, A. L. (1998). A computationally efficient feasible sequential quadratic programming algorithm. *SIAM Journal on Optimization*, 11(4):1092–1118.

- Lee, D. S., Gonzalez, L. F., Srinivas, K., and Périaux, J. (2007). Multi-objective robust design optimisation using hierarchical asynchronous parallel evolutionary algorithms. In *45th AIAA Meeting, Reno, Nevada*.
- MacMillin, P. E., Huang, X., Dudley, J., Grossman, B., Haftka, R. T., and Mason, W. H. (1995). Multidisciplinary Optimization of the High-Speed Civil Transport. In Alexandrov, N. M. and Hussaini, M. Y., editors, *Multidisciplinary Design Optimization, State of the art - Proceedings of the ICASE/NASA Langley Workshop on Multidisciplinary Design Optimization*, pages 153–171. SIAM.
- Marler, R. T. and Arora, J. S. (2004). Survey of Multi-objective Optimization Methods for Engineering. *Structural and Multidisciplinary Optimization*, 26(6):369–395.
- Masmoudi, M. and Auroux, D. (2005). The State-of-the-Art in Collaborative Design. In Uthup, B., Koruthu, S. P., Sharma, R. K., and Priyadarshi, P., editors, *Recent Trends in Aerospace Design and Optimization, Proceedings of SAROD-2005, December 2005*, pages 411–425, Hyderabad, India. Tata McGraw Hill.
- Mattei, P. and Druot, T. (2005). General description of odip platform. Technical report, Airbus. Ref FM0200016.
- May, N. (2005). Quantifying uncertainty in conceptual aircraft design. Master’s thesis, ENSICA.
- MDO Technical Committee (2007). Some popular definitions for Multidisciplinary Design Optimisation. see also <http://www.aiaa.org/portal/index.cfm?Getcomm=80>. Copyright 2007 AIAA.
- Messac, A. (1996). Physical Programming: Effective Optimization for Computational Design. *AIAA Journal*, 34(1):149–158.
- Messac, A. and Ismail-Yahaya, A. (2002). Multiobjective Robust Design Using Physical Programming. *Structural and Multidisciplinary Optimization*, 23(5):357–371.
- Messac, A. and Mattson, C. A. (2002). Generating Well-Distributed Sets of Pareto Points for Engineering Design Using Physical Programming. *Optimization and Engineering*, 3(4):431–450.
- Michalewicz, Z. (1992). Genetic Algorithms + Data Structures = Evolution Programs.
- Miettinen, K. (1999). *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers.
- Molina-Cristobal, A., Parks, G. T., and Clarkson, P. J. (2006). Finding robust solutions to multi-objective optimisation problems using polynomial chaos. In *Proceedings of the 6th ASMO UK/ISSMO Conference on Engineering Design Optimization*.
- Mongeau, M., Karsenty, H., Rouzé, V., and Hiriart-Urruty, J.-B. (2000). Comparison of public-domain software for black-box global optimization. *Optimization Methods and Software*, 13(3):203–226.

- Nakayama, H. (2005). Multi-objective Optimization and its Engineering Applications. In Branke, J., Deb, K., Miettinen, K., and Steuer, R. E., editors, *Practical Approaches to Multi-Objective Optimization*. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany.
- Nash, J. F. (1951). Non cooperative games. *Annal of Mathematics*, 54(2):286–295.
- Nelder, J. A. and Mead, R. (1965). A simplex method for function minimization. *Computer Journal*, 7(4):308–313.
- Nicolai, L. M. (1975). *Fundamentals of Aircraft Design*. METS Inc.
- Orr, M. J. L. (1998). Optimising the widths of radial basis functions. In *Proceedings of the Fifth Brazilian Symposium on Neural Networks*, Belo Horizonte, Brazil.
- Orsi, G. (1994). Les plans d’expériences. *Technologie*, 66:17–31.
- Pacull, M. (1990). *Conception avant-projet des avions de transport commerciaux*. AIRBUS internal document.
- Pareto, V. (1896). *Cours d’économie politique, Rouge*. Lausanne, Switzerland.
- Pediroda, V. and Poloni, C. (2006). Robust design, approximation methods and self organizing map, techniques for MDO problems. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Poles, S., Fu, Y., and Rigoni, E. (2006). The effect of initial population sampling on the convergence of multi-objective genetic algorithms. Congress abstract. 7th international conference on MultiObjective Programming and Goal Programming.
- Prandtl, L. (1933). Uber tragflügel des kleinsten induzierten widerstandes. *Zeitschrift für Flugtechnik und motorluftschiffahrt*.
- Périaux, J., Gonzalez, L. F., Whitney, E. J., and Srinivas, K. (2006). MOO Methods for Multidisciplinary Design Using Parallel Evolutionary Algorithms, Game Theory and Hierarchical Topology: Theoretical Background. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Rai, M. M. (2006a). Single- and Multiple-Objective Optimization with Differential Evolution and Neural Networks. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Rai, M. M. (2006b). Towards Robust Designs Via Multiple-Objective Optimization Methods. In *VKI lecture series: Introduction to Optimization and Multidisciplinary Design*, Rhode-Saint-Genese, Belgium.
- Rechenberg, I. (1973). *Evolutionstrategie : Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart : Fromman-Holzboog.

- Rodriguez, J. F., Renaud, J. E., Wujek, B. A., and Tappeta, R. V. (2000). Trust region model management in multidisciplinary design optimization. *Journal of Computational and Applied Mathematics*, 124(1-2):139–154.
- Roudenko, O. (2004). *Application des Algorithmes Evolutionnaires aux Problèmes d'Optimisation Multi-Objectif avec Contraintes*. PhD thesis, Ecole Polytechnique.
- Salas, A. O. and Townsend, J. C. (1998). Framework requirements for MDO application development. In *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper 98-4740.
- Sasaki, D. and Obayashi, S. (2004). Adaptive range Multi objective genetic algorithms and self-organizing map for multi objective optimization problem. In *VKI lecture series: Optimization Methods & Tools for Multicriteria/Multidisciplinary Design, Applications to Aeronautics and Turbomachinery*, Rhode-Saint-Genese, Belgium.
- Sbalzarini, I. F., Müller, S., and Koumoutsakos, P. (2000). Multiobjective Optimization Using Evolutionary Algorithms. In *Proceedings of the Summer Program*. Center of Turbulence Research.
- Schaffer, J. D. (1985). Multiple objective optimization with vector evaluated genetic algorithms. In *Genetic Algorithms and their Applications: Proceedings of the First International Conference on Genetic Algorithms*, pages 93–100. Lawrence Erlbaum.
- Schumann-Hindenberg, U. (2001). Ap2054 module 1. Technical report, AIRBUS INDUSTRIE Procedure, Develop new A/C (DnA) - Business Process Definition - Milestones Model.
- Schwefel, H. P. (1981). *Numerical Optimization of Computer Models*. J. Wiley & Sons, NY.
- Sefrioui, M. and Périaux, J. (2000). Nash Genetic Algorithms: Examples and Applications. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00*, pages 509–516. IEEE Press.
- Shin, H. and Chob, S. (2006). Response modeling with support vector machines. *Expert Systems with Applications*, 30(4):746–760.
- Simpson, T. W. (1998). Comparison of Response Surface and Kriging Models in the Multidisciplinary Design of an Aerospike Nozzle. Technical Report TR-98-16.
- Simpson, T. W., Mauery, T. M., Korte, J. J., and Mistree, F. (1998). Comparison of Response Surface and Kriging Models for Multidisciplinary Design Optimization. In *Proceedings of the 7th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper 98-4755.
- Smola, A. J. and Schölkopf, B. (2004). A tutorial on support vector regression. *Statistics and Computing*, 14(3):199–222.
- Sobieski, I. P. and Kroo, I. M. (2000). Collaborative Optimization Using Response Surface Estimation. *AIAA Journal*, 38(10):1931–1938.

- Sobieszczanski-Sobieski, J. and Haftka, R. T. (1997). Multidisciplinary aerospace design optimization: survey of recent developments. In *Structural and Multidisciplinary Optimization*, volume 14, pages 1–23. Springer Berlin/Heidelberg.
- Srinivas, N. and Deb, K. (1994). Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms. *Evolutionary Computation*, 2(3):221–248.
- Srivastava, R. and Goldberg, D. (2001). Verification of the Theory of Genetic and Evolutionary Continuation. In Spector, L., Goodman, E., Wu, A., Langdon, W., Voigt, H. M., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M., and Burke, E., editors, *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*, pages 623–630, San Francisco, CA. Morgan Kaufmann.
- Tappeta, R., Renaud, J., Messac, A., and Sundararaj, G. (2000). Interactive Physical Programming: Tradeoff Analysis and Decision Making in Multicriteria Optimization. *AIAA Journal*, 38(5):917–926.
- Torczon, V. and Trosset, M. W. (1998). Using approximations to accelerate engineering design optimization. In *Proceedings of the AIAA/USAF/NASA/ISSMO 7th Symposium on Multidisciplinary Analysis and Optimization*. AIAA Paper 98-4800.
- Torenbeek, E. (1982). *Synthesis of subsonic airplane design*, chapter General aspects of aircraft configuration development. Delft University Press, Kluwer Academic Publishers.
- Utyuzhnikov, S. V., Fantini, P., and Guenov, M. D. (2005). Numerical method for generating the entire pareto frontier in multiobjective optimization. In Schilling, R., Haase, W., Périaux, J., Baier, H., and Bugeda, G., editors, *Proceedings of EUROGEN 2005, 6th Conference on Evolutionary and Deterministic Methods for Design, Optimization and Control with Applications to Industrial and Societal Problems*, Munich, Germany.
- Van Grieken, M. (2004). *Optimisation pour l'apprentissage et apprentissage pour l'optimisation*. PhD thesis, Université Paul Sabatier.
- Van Veldhuizen, D. A. and Lamont, G. B. (1999). Multiobjective Evolutionary Algorithm Test Suites. In Carroll, J., Haddad, H., Oppenheim, D., Bryant, B., and Lamont, G. B., editors, *Proceedings of the 1999 ACM Symposium on Applied Computing*, pages 351–357, San Antonio, Texas. ACM.
- Van Veldhuizen, D. A. and Lamont, G. B. (2000). Multiobjective Evolutionary Algorithms: Analyzing the State-of-the-Art. *Evolutionary Computation*, 8(2):125–147.
- Vapnik, V., Golowich, S., and Smola, A. (1997). Support vector method for function approximation, regression estimation and signal processing. *Advances in Neural Information Processing Systems*, 9:281–287.
- VR&D Copyright (1995). DOT, Design Optimization Tools, users manual, version 4.20. Description on <http://www.vrand.com>.

- Wakayama, S. and Kroo, I. (1998). The challenge and promise of blended-wing-body optimization. AIAA Paper 98-4736.
- Walsh, J. L., Townsend, J. C., Salas, A. O., Samareh, J. A., Mukhopadhyay, V., and Barthelemy, J.-F. (2000a). Multidisciplinary high-fidelity analysis and optimization of aerospace vehicles, part 1: Formulation. In *Proceedings of the 38th Aerospace Sciences Meeting and Exhibit*. AIAA Paper 2000-0418.
- Walsh, J. L., Weston, R. P., Samareh, J. A., Mason, B. H., Green, L. L., and Biedron, R. T. (2000b). Multidisciplinary high-fidelity analysis and optimization of aerospace vehicles, part 2: Preliminary results. In *Proceedings of the 38th Aerospace Sciences Meeting and Exhibit*. AIAA Paper 2000-0419.
- Wang, G. G. (2003). Adaptative Response Surface Method Using Inherited Latin Hypercube Design Points. *Journal of Mechanical Design, Transactions of the ASME*, 125:210–220.
- Welcomme, J.-B., Gleizes, M.-P., Redon, R., and Druot, T. (2006). Self-Regulating Multi-Agent System for Multi-Disciplinary Optimisation Process. In *Proceedings of the EUMAS 4rd European Workshop on Multi-Agent Systems*.
- Willcox, K. and Wakayama, S. (2002). Simultaneous optimization of a multiple-aircraft family. AIAA Paper 2002-1423.
- Zhou, J. L., Tits, A. L., and Lawrence, C. T. (1997). User's guide for ffsqp version 3.7: A fortran code for solving constrained nonlinear (minimax) optimization problems, generating iterates satisfying all inequality and linear constraints. Technical Report TR-92-107r2, Systems Research center, University of Maryland, College Park, MD 20742.
- Zitzler, E. and Thiele, L. (1999). Multiobjective Evolutionary Algorithms: A Comparative Case Study and the Strength Pareto Approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271.

Résumé : La conception d'avions au stade avant-projet consiste à déterminer les principales caractéristiques d'un avion répondant à un cahier des charges donné. Ces études peuvent être résumées par des problèmes d'optimisation globale sous contraintes avec typiquement un millier de paramètres et presque autant de contraintes. Les contraintes expriment la faisabilité physique ainsi que le cahier des charges à respecter, et les objectifs sont des performances de l'avion guidées par des études de marché. De plus, la conception d'avions est un problème d'optimisation multicritère du fait de la présence de fonctions objectifs antagonistes.

L'objectif de cette thèse est d'introduire de nouvelles méthodes mathématiques qui peuvent être utiles dans un outil de dimensionnement avant-projet pour résoudre le problème d'optimisation d'une configuration d'avion. Nous avons contribué à l'amélioration des méthodes d'optimisation qui sont couramment utilisées au département des Avant-Projets d'Airbus. En utilisant les algorithmes génétiques, nous avons rendu le processus d'optimisation monocritère plus robuste. Ensuite, nous avons introduit des méthodes d'optimisation multicritère car nous avons plusieurs critères conflictuels à considérer. Comme les temps de calcul sont devenus importants, nous avons décidé de substituer au modèle d'avion un modèle approché. Nous avons implémenté les fonctions à base radiale pour approcher les contraintes et les fonctions objectifs. Enfin, nous avons propagé les incertitudes du modèle pour estimer la robustesse des résultats de l'optimisation et nous avons proposé un aboutissement possible de l'intégration de ces techniques : donner aux ingénieurs une perception opérationnelle de l'espace de définition.

Mots-clés : Conception d'avions, optimisation multidisciplinaire et multicritère, approximation par un modèle de substitution, propagation d'incertitudes, aide à la décision.

Abstract: Aircraft sizing studies consist in determining the main characteristics of an aircraft starting from a set of requirements. These studies can be summarized as global constrained optimisation problems with typically one thousand parameters and almost as many constraints. The constraints express physical feasibility and the requirements to be satisfied, and the objectives are market driven performances of the aircraft. Moreover, aircraft sizing is typically a multicriteria optimisation problem because of some competing objectives.

The aim of this thesis is to introduce new mathematical methods that can be useful in a future project sizing tool to treat the aircraft configuration optimisation problem. We contributed in improving the optimisation methods that are currently used in the Airbus Future Project Office. By using genetic algorithms, we made the mono-criterion optimisation process more robust. Then, we introduced multicriteria optimisation methods because we had several conflicting criteria to consider. As the calculation times became important, we decided to substitute the aircraft model by a surrogate model. We implemented radial basis functions to approximate the constraint and the objective functions. Finally, we propagated the model uncertainty to assess the robustness of the optimisation results and we proposed a possible outcome of the integration of these different techniques in order to yield the engineers a global and operational perception of the design space.

Keywords: Aircraft sizing, multidisciplinary and multicriteria optimisation, surrogate model approximation, uncertainty propagation, help in decision making.
