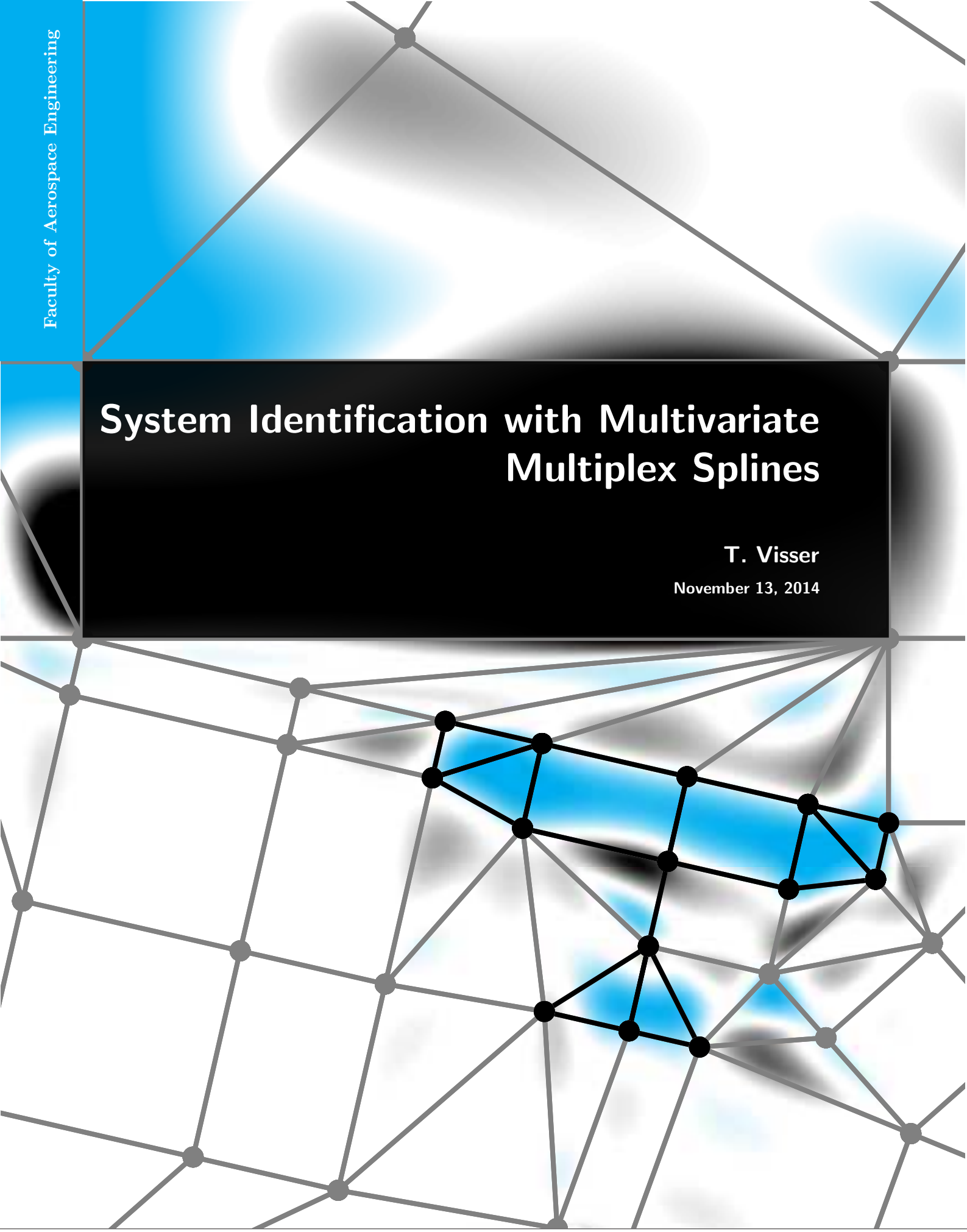
Faculty of Aerospace Engineering

System Identification with Multivariate Multiplex Splines

T. Visser

November 13, 2014



TU Delft Delft University of Technology

Challenge the future

System Identification with Multivariate Multiplex Splines

Master of Science Thesis

For obtaining the degree of Master of Science in Aerospace Engineering
at Delft University of Technology

T. Visser

November 13, 2014

Faculty of Aerospace Engineering · Delft University of Technology

Delft University of Technology

Delft University Of Technology
Department Of
Control and Simulation

The undersigned hereby certify that they have read and recommend to the Faculty of Aerospace Engineering for acceptance a thesis entitled **"System Identification with Multivariate Multiplex Splines"** by **T. Visser** in partial fulfillment of the requirements for the degree of **Master of Science**.

Dated: November 13, 2014

Readers:

Prof. dr. ir. J. A. Mulder

Dr. ir. C. C. de Visser

Dr. ir. E. van Kampen

Dr. Z. Erkin

Contents

Preface

In this thesis the results of my final masters project are presented. This includes an extensive mathematical definition of the multiplex itself and B-splines defined on tessellations of these polytopes. The ability of these splines to fit scattered data is investigated in a verification process. Finally a system identification algorithm is defined and validated on DelFly flight test data.

The thesis is split in three parts. The first part contains the article that was accepted for the 2014 AIAA SciTech conference. This article covers the most important parts of the multiplex spline definition, as well as part of the verification process that was performed on a quadrotor analytic model. The second part is the actual report, including an executive summary and introduction. The report includes all phases of the research. Finally the third part contains a draft mathematical paper describing the definition of continuity conditions between multiplices. This is seen as the mathematical innovation of this work. The goal is to submit this paper to the Journal of Approximation Theory within a few months, as soon as the connection to existing literature is made more clear. All parts of the report can be read independent of each other and may use different notation.

I would like to take this opportunity to express my gratitude for the support I got in the pursuit of my MSc title. First of all, I would like to thank my daily supervisors dr. ir. Coen de Visser and dr. ir. Erik-Jan van Kampen for fueling my curiosity with new insights and providing useful comments on my written work. My thanks also go out to the chair of my graduation committee prof. dr. ir. J. A. Mulder and the external member dr. Z. Erkin for reading and assessing this thesis. For their help in my attempts to attain and use a data set for validation purposes, I thank João Caetano and Erik van der Horst. Finally, special thanks go out to my fellow students in room SIM 0.08, aka the Lower House, for providing the essential distractions in the form of coffee breaks and geopolitical discussions.

Part I

AIAA SciTech Conference Paper

Quadrotor System Identification using the Multivariate Multiplex B-Spline

T. Visser, * C. C. de Visser † and E. van Kampen ‡

Delft University of Technology, P.O. Box 5058, 2600GB Delft, The Netherlands

A novel method for aircraft system identification is presented that is based on a new multivariate spline type; the multivariate multiplex B-spline. The multivariate multiplex B-spline is a generalization of the recently introduced tensor-simplex B-spline. Multivariate multiplex splines obtain similar or better approximation accuracy using less parameters (B-coefficients) than standard multivariate simplex B-splines which are currently used for aircraft system identification. The multiplex spline allows the user to incorporate a-priori knowledge of the modelled system in the definition of the model structure. In particular, while the standard simplex B-splines use a multi-dimensional triangulation in which all dimensions are coupled, the multiplex spline enables the user to decouple specific model dimensions based on expert knowledge of the system. The new method is used to approximate a 4-dimensional nonlinear quadrotor inflow dataset. The results show that the multiplex B-spline obtains a relative root mean square error of 0.672% using 1440 B-coefficients. This compares favorably to results obtained with the standard 4-dimensional simplex B-spline on the same dataset, which resulted in an relative root mean square error of 1.608% using 1540 B-coefficients.

Nomenclature

B	Matrix form of basis polynomials of a spline
$\boldsymbol{B}_{\kappa}^{d}$	Vector of basis polynomials of a simplex spline
$\boldsymbol{b}$	Vector of barycentric coordinates in a simplex
$\boldsymbol{\mathcal{B}}_{\lambda}^{d}$	Vector of basis polynomials of a multiplex spline
$\boldsymbol{c}$	Vector of B-coefficients
d	Polynomial degree
H	Smoothness matrix
K_{emf}	Constant related to back EMF in engine
ℓ	Number of layers in a multiplex spline
$\mathcal{M}$	Tessellation consisting of multiplices
m	Continuity order or dimension smaller than n
N	Number of simplices or multiplices in a tessellation
n	Dimension, number of variables
P	Engine power, W
p	Simplex polynomial
R_a	Motor armament resistance, Ω
r	Total continuity order
T	Thrust, N
$\mathcal{T}$	Triangulation consisting of simplices
U	Input voltage, V
$\boldsymbol{v}$	Vertex of a simplex
v	Velocity, m/s

*Master Student, Control and Simulation Department, t.visser-1@student.tudelft.nl.

†Assistant Professor, Control and Simulation Department, c.c.devisser@tudelft.nl, AIAA Member.

‡Assistant Professor, Control and Simulation Department, e.vankampen@tudelft.nl, AIAA Member.

$\boldsymbol{w}$ Vertex of a multiplex layer
X Input matrix
$\boldsymbol{y}$ Output vector
α Angle of attack, deg
$\boldsymbol{\beta}$ Vector of barycentric coordinates in a multiplex
Γ Multiplex
γ Multi-index used in simplex continuity conditions
Δ Simplex
ε Multi-index used in multiplex continuity conditions
η Total motor and rotor efficiency
κ Simplex spline multi-index
λ Multiplex spline multi-index
μ Multi-index containing dimensions of multiplex layers
ν Multi-index of dimensions of a multiplex
π Multiplex polynomial
ω Rotor rotational rate, rpm

Subscript
h Induced in hover
i Induced
z Vertical
∞ Freestream

I. Introduction

Due to the increased popularity of model-based control techniques such as nonlinear dynamic inversion, the need for accurate non-linear models has grown. In many industrial applications data tables are used, whereas the research community has embraced neural networks as a function approximator. Recently the multivariate simplex B-spline was presented as an alternative.[1–3] The main benefit of this technique over neural networks is the fact that it is linear in the parameters. This allows for the use of efficient least squares solvers.

The major downside of the simplex spline is its triangulation. Although it provides the user with a lot of flexibility in spanning the domain of interest, it is unclear how this flexibility can be exploited. Most triangulation optimization algorithms are developed for two-dimensional domains.[4] In higher-dimensional domains these algorithms cannot be used. Current practice is to split up the domain in rectangular shapes first. These boxes are then split up using a standard triangulation.[1] Another option that is currently being investigated is the simultaneous optimization of the triangulation and the parameters.[5] Although a global optimum can be found with both the parameters and the triangulation as arguments, its high computational complexity currently forbids the use of this technique on large triangulations.

On top of the excess flexibility in defining a triangulation, the shape of the high-dimensional simplex also gives reason for concern. As the dimension increases, the content of the simplex moves towards the vertices, away from the center. In practical terms, it becomes increasingly hard to fill the entire simplex with data.

In this paper a new spline is proposed that can immediately be defined on orthotopes (or boxes), without triangulating them. This has significant benefits over standard simplex B-spline methods in terms of the distribution of data over a domain element. The flexibility in defining the triangulation is reduced, but the technique employed by De Visser can still be used,[1] with the difference that it is no longer required to triangulate the orthotopes.

The definition of the orthotope spline is based in the tensor-product simplex spline, which was recently introduced by Govindarajan et al. to solve the Hamilton-Jacobi-Bellman equation, central in reinforcement learning and optimal control.[6,7] In this framework the B-coefficients of a multivariate simplex spline are defined using a univariate spline function of a different variable. This framework was used by Govindarajan et al. to estimate autopilot safety margins,[7] and by Sun et al. in an aircraft system identification framework.[8]

The multivariate multiplex spline used in the current research is a direct generalization of the tensor-product simplex spline. It allows for the use of multiple multivariate layers. Each layer represents a multivariate simplex spline that describes the coefficients in the previous layer. Fitting data on non-rectangular

domains is facilitated by studying the combination of a single simplex from each layer: a multiplex.

The goal of this paper is to present and test an algorithm for aircraft system identification based on this multivariate multiplex spline. As a test case, the thrust and torque of a quadrotor MAV are modeled, taking engine dynamics and nonuniform inflow into account. The advantage of the multiplex spline over the standard simplex B-spline is demonstrated; a similar or better approximation accuracy is obtained with a lower total number of parameters (B-coefficients) than the standard simplex B-splines.

Because this is the first introduction of the multiplex spline in an engineering application, a short introduction is presented here as section II. A more extensive discussion, as well as many of the proofs of presented theory, can be found in Ref. 9. In section III the main contribution of this article is presented. An algorithm is proposed for the use of the multiplex spline in system identification. The application of the algorithm to a quadrotor modeling task is described in section IV. Finally in section V the performance of the algorithm is discussed and conclusions are drawn regarding its applicability.

II. The multiplex spline framework

The origin of the multiplex spline lies in the tensor-product simplex spline of Govindarajan.[6] That is, B-coefficients of one spline are represented by a spline in other variables. The multiplex spline results when this process is repeated and B-coefficients are reordered. The process and the effect on the defining elements of a simplex spline are discussed in this section.

To properly introduce the multiplex spline, first a short explanation of the simplex spline framework is presented. A more elaborate discussion of the bivariate case is given by Lai and Schumaker.[10]

A. The multivariate simplex spline

A multivariate simplex spline is a piecewise polynomial defined on a geometric basis of simplices. Continuity of arbitrary order can be defined between simplices if they share an edge. The polynomials on individual simplices are weighted sums of Bernstein basis polynomials defined in barycentric coordinates. The most important facets of such a spline are discussed in this subsection.

1. Simplices, barycentric coordinates and triangulations

An n-simplex is defined as the convex hull of $n+1$ non-degenerate vertices $\mathcal{V} = \{\boldsymbol{v}_0, ..., \boldsymbol{v}_n\}, \boldsymbol{v}_i \in \mathbb{R}^n$. Any point in $\mathbb{R}^n$ can be uniquely defined using a vector of $n+1$ barycentric coordinates $\boldsymbol{b} := (b_0, ..., b_n) \in \mathbb{R}^{n+1}$ relative to this set of vertices, such that $\sum_{i=0}^{n} b_i = 1$. For a general point $\boldsymbol{x} \in \mathbb{R}^n$ we have

$$\boldsymbol{x} = \sum_{i=0}^{n} b_i \boldsymbol{v}_i \tag{1}$$

If $b_i \geq 0, \forall i \in [0, n]$, then $\boldsymbol{x}$ lies in the simplex described by $\mathcal{V}$.

Multiple n-simplices Δ can be combined to form a triangulation $\mathcal{T}$ of a domain. We define a triangulation as the collective of simplices, such that

$$\mathcal{T} = \bigcup_{i=1}^{N} \Delta_i, \qquad \Delta_i \cap \Delta_j \in \{\emptyset, \bar{\Delta}\}, \forall i \neq j \tag{2}$$

with N the total number of simplices and $\bar{\Delta}$ an m-simplex, $0 \leq m < n$. That is, in a triangulation simplices do not overlap but may share complete edges of dimension lower than n.

2. B-form polynomial

Polynomials are defined on a simplex by taking the weighted sum of Bernstein basis polynomials, defined in barycentric coordinates. The B-form of such a polynomial is[11]

$$p(\boldsymbol{b}) = \sum_{|\kappa|=d} c_\kappa B_\kappa^d(\boldsymbol{b}), \qquad B_\kappa^d(\boldsymbol{b}) := \frac{d!}{\kappa!} \boldsymbol{b}^\kappa \tag{3}$$

where c_κ are B-coefficients, discussed in the next subsubsection, and $\kappa = (\kappa_0, ..., \kappa_n) \in \mathbb{N}^n$ is a multi-index with $|\kappa| := \sum_i \kappa_i$ its one-norm. Also, $\kappa! := \prod_i \kappa_i!$ and $\boldsymbol{b}^\kappa := \prod_i b_i^{\kappa_i}$. Every possible permutation of κ results in a separate Bernstein basis polynomial. The complete set forms a basis for the space of all polynomials of degree d in n variables.

To define a complete set of basis polynomials on each simplex in a triangulation, the vector form of the basis polynomials is defined as

$$\boldsymbol{B}_\kappa^d(\boldsymbol{b}(\boldsymbol{x})) := \begin{cases} B_\kappa^d(\boldsymbol{b}^{\Delta_i}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \in \Delta_i \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

where $\boldsymbol{b}^{\Delta_i}(\boldsymbol{x})$ signifies that $\boldsymbol{x}$ is described in barycentric coordinates with respect to the vertices of simplex Δ_i. The vector form is thus a collection of the basis polynomials in one simplex, padded with zeros for all other simplices.

3. B-coefficients and the B-net

One important feature of the simplex spline is that the B-coefficients c_κ, introduced in Eq. (3) as the weights of the basis polynomials, have a spatial location in the simplex. They coincide with control points, such that c_κ lies at $\boldsymbol{x}_\kappa \in \mathbb{R}^n$

$$\boldsymbol{x}_\kappa = \frac{1}{d} \sum_{i=0}^{n} \kappa_i \boldsymbol{v}_i \tag{5}$$

The collective of these locations is called the B-net. It plays a central role in the definition of continuity conditions. An example B-net is shown in figure 1.

Much like the basis polynomials, the B-coefficients are collected in a vector. We write $\boldsymbol{c}^{\Delta_i} = (c_\kappa^{\Delta_i})_{|\kappa|=d}$, where the coefficients are ordered according to the multi-index in lexicographical order, and $\boldsymbol{c} = (\boldsymbol{c}^{\Delta_1}, ..., \boldsymbol{c}^{\Delta_N})$. The superscript Δ_i indicates that the coefficients lie in that simplex.

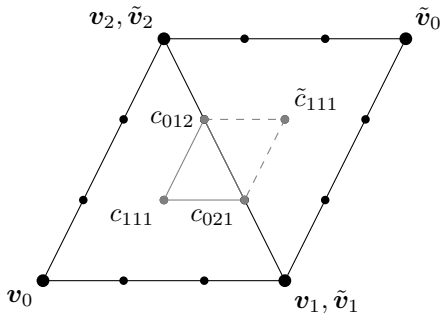


Figure 1: Two 2-simplices sharing an edge, and their B-net (small black circles). In gray an example first order continuity condition is given. Note how one coefficient in the right simplex is related to three coefficients in the left simplex.

4. Continuity conditions

Two polynomials p and $\tilde{p}$ defined on n-simplices Δ and $\tilde{\Delta}$ that share an $n-1$-edge can be joined with arbitrary continuity order $r < d$. This is done by imposing the following constraints[1, 10]

$$\tilde{c}_{(m,\kappa_1,...\kappa_n)} = \sum_{|\gamma|=m} c_{(0,\kappa_1,...\kappa_n)+\gamma} B_\gamma^m(\boldsymbol{b}^{\Delta}(\tilde{\boldsymbol{v}}_0)), \qquad \forall m \in [0, r] \tag{6}$$

where γ is again a multi-index of the same size as κ and the sum between κ and γ is element-wise. It is assumed that $\boldsymbol{v}_0$ and $\tilde{\boldsymbol{v}}_0$ are the out-of-edge vertices. An example of a first order (that is $m = 1$) continuity condition is given for the bivariate case in figure 1.

Because the conditions in Eq. (6) are linear in c_κ, they can be collected in a smoothness matrix H, such that we obtain[12]

$$\mathrm{H}\boldsymbol{c} = \boldsymbol{0} \tag{7}$$

H is a very sparse matrix, because each row describes a single constraint between two simplices. That is, in each row a maximum of $1 + \frac{(d+n-1)!}{(d-1)!n!}$ elements are nonzero.

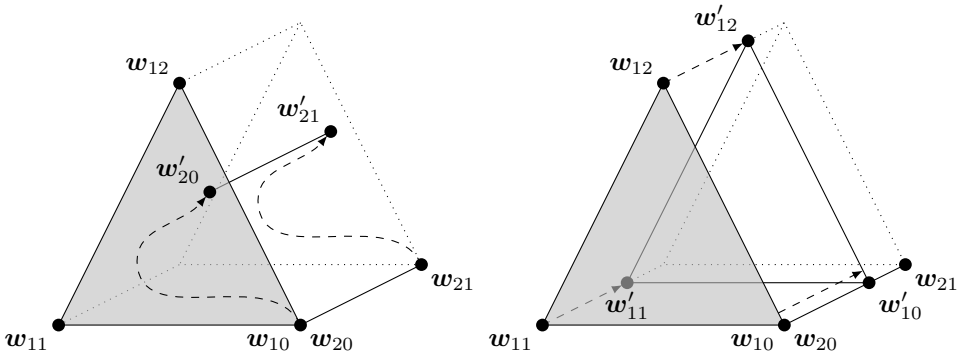


Figure 2: The construction of the (2,1)-multiplex in different order. In the left figure, a 2-simplex $\{\boldsymbol{w}_{10}, \boldsymbol{w}_{11}, \boldsymbol{w}_{12}\}$ is combined with a 1-simplex $\{\boldsymbol{w}_{20}, \boldsymbol{w}_{21}\}$, indicating that the 1-simplex is appended to any point on the 2-simplex. On the right a 2-simplex is appended to any point on the 1-simplex. Both approaches result in the same prism: the dotted (2,1)-multiplex with layers $\{\boldsymbol{w}_{10}, \boldsymbol{w}_{11}, \boldsymbol{w}_{12}\}$ and $\{\boldsymbol{w}_{20}, \boldsymbol{w}_{21}\}$.

5. Fitting scattered data

The above can be combined into a single equality constrained least squares problem.

$$\begin{aligned} &\min ||\boldsymbol{y} - \mathrm{B}(\boldsymbol{b}(\mathrm{X}))\boldsymbol{c}|| \\ &\text{subject to: } \mathrm{H}\boldsymbol{c} = \boldsymbol{0} \end{aligned} \tag{8}$$

In the above, $||\bullet||$ is the 2-norm, X is a matrix containing the input data, and B is the matrix of basis polynomials. The output vector is given as $\boldsymbol{y}$.

The solution of Eq. (8) can be found using Lagrangian multipliers. This is discussed in detail by De Visser et al.[12] Because the basis polynomial matrix B is block diagonal and H is sparse, distributed solvers can be employed.

B. The multivariate multiplex spline

The multiplex spline is a generalization of the tensor-product simplex spline presented by Govindarajan et al.[6] In the latter framework the B-coefficients c_κ are represented by a univariate simplex spline. In the current discussion c_κ is instead described using a multivariate spline. On top of that, the coefficients in this second layer can again be described using a spline. The result is a spline defined on nested triangulations. In the multiplex spline framework one simplex from each triangulation is considered at a time. The combination of these simplices is called a multiplex.

1. Multiplices, barycentric coordinates and tessellations

A ν-multiplex Γ, with ν a multi-index with ℓ entries and $|\nu| = n$, is an n-dimensional polytope that is constructed by, starting from the ν_1-simplex, iteratively combining the $(\nu_1, ..., \nu_j)$-multiplex with a ν_{j+1}-simplex. Combining layers in this case means to append the ν_{j+1}-simplex to every point in the $(\nu_1, ..., \nu_j)$-multiplex (see figure 2). The ν_{j+1}-simplex is called the $(j+1)^{\text{th}}$ layer of the multiplex. In order for the multiplex to be n-dimensional, each ν_i-dimensional layer should introduce ν_i new variables. In our current discussion this means that the layers are mutually perpendicular. Note that the order in which layers are combined does not affect the end result, as illustrated in figure 2.

Any point in the multiplex can be uniquely described using barycentric coordinates $\boldsymbol{b}_i$ in each layer. By collecting the vectors $\boldsymbol{b}_i$ we obtain $\boldsymbol{\beta} = (\boldsymbol{b}_1, ..., \boldsymbol{b}_\ell) \in \mathbb{R}^{n+\ell}$, where $\boldsymbol{b}_i = (b_{i0}, ..., b_{i\nu_i}) \in \mathbb{R}^{\nu_i+1}$ with constraints $\sum_j b_{ij} = 1, \forall i \in [1, \ell]$. Instead of solving Eq. (1) for $\boldsymbol{b}_i$ in each layer, a direct solution can be found by considering the vertices of layers. For this we select a copy of each layer such that they share a vertex $\boldsymbol{w}_0$. That is, we select vertex sets $\mathcal{W}_i = \{\boldsymbol{w}_0, \boldsymbol{w}_{i1}, ..., \boldsymbol{w}_{i\nu_i}\}$, as indicated in figure 2 for $i \in \{1, 2\}$. Then we have for any point $\boldsymbol{x} \in \mathbb{R}^n$ in the multiplex

$$\boldsymbol{x} = (1-\ell)\boldsymbol{w}_0 + \sum_{i=1}^{\ell}\sum_{j=0}^{\nu_i}\beta_{ij}\boldsymbol{w}_{ij} \tag{9}$$

where $\boldsymbol{w}_{i0} := \boldsymbol{w}_0, \forall i \in [1,\ell]$. Equation (9) is a direct result of adding the origin location $\boldsymbol{w}_0$ to the locations of the projections of $\boldsymbol{x}$ in the layers with respect to the same origin point $\boldsymbol{w}_0$.

A tessellation of multiplices is easily constructed from triangulations in each layer of the spline. For example, a tessellation of (2,1)-multiplices can be constructed by making a triangulation of 2-simplices and one of 1-simplices, then constructing the multiplices from any combination of simplices from both triangulations. The resulting tessellation $\mathcal{M}$ consisting of multiplices Γ_i is defined as

$$\mathcal{M} = \bigcup_{i=1}^{N}\Gamma_i, \qquad \Gamma_i \cap \Gamma_j \in \{\emptyset, \bar{\Gamma}\}, \forall i \neq j \tag{10}$$

where N is the number of multiplices. The multiplex $\bar{\Gamma}$ is a $(\nu-\mu)$-multiplex, where μ is a multi-index like ν with $0 \leq |\mu| < n$ and $\nu - \mu > 0$.

2. B-form polynomial

If c_κ in Eq. (3) is replaced by a polynomial in the B-form, a tensor-product between the basis polynomials results. The multi-indices κ from the layers are renamed and combined to form one multi-index $\lambda = (\lambda_1, ..., \lambda_\ell)$, where $\lambda_i = (\lambda_{i0}, ..., \lambda_{i\nu_i})$ the multi-index in the i^{th} layer with $|\lambda_i| = d_i$. The degree is defined per layer, leading to a degree vector $\boldsymbol{d} = (d_1, ..., d_\ell)$. Applying these changes to Eq. (3) we obtain

$$\pi(\boldsymbol{\beta}) = \sum_{\substack{|\lambda_i|=d_i,\\ \forall i\in[1,\ell]}} c_\lambda \mathcal{B}^{\boldsymbol{d}}_\lambda(\boldsymbol{\beta}), \qquad \mathcal{B}^{\boldsymbol{d}}_\lambda(\boldsymbol{\beta}) := \prod_{i=1}^{\ell} B^{d_i}_{\lambda_i}(\boldsymbol{b}_i) = \frac{\prod_i d_i!}{\lambda!}\boldsymbol{\beta}^\lambda \tag{11}$$

Again, the barycentric coordinate vector $\boldsymbol{\beta} = (\boldsymbol{b}_1, ..., \boldsymbol{b}_\ell)$ is a collection of barycentric coordinates in the layers. The resulting basis polynomials have a degree of $\prod_i d_i$, but not all basis polynomials of this degree are present.

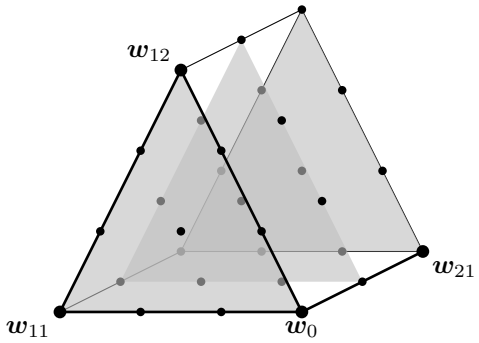


Figure 3: A (2,1)-mulitplex and its B-net for a multiplex spline of degree (3,2). The B-net is formed by three parallel copies of the simplex spline B-net (see figure 1).

In the vector form this procedure results in a tensor-product per multiplex. The vector form of $\boldsymbol{\mathcal{B}}^{\boldsymbol{d}}_\lambda$ can be obtained by taking the tensor-product of the subsets $\boldsymbol{B}^{d_i}_{\lambda_i}$ and then reordering, but obtaining them directly is preferable.

$$\boldsymbol{\mathcal{B}}^{\boldsymbol{d}}_\lambda(\boldsymbol{\beta}(\boldsymbol{x})) := \begin{cases} \mathcal{B}^{\boldsymbol{d}}_\lambda(\boldsymbol{\beta}^{\Gamma_i}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \in \Gamma_i \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Note the similarity with Eq. (4).

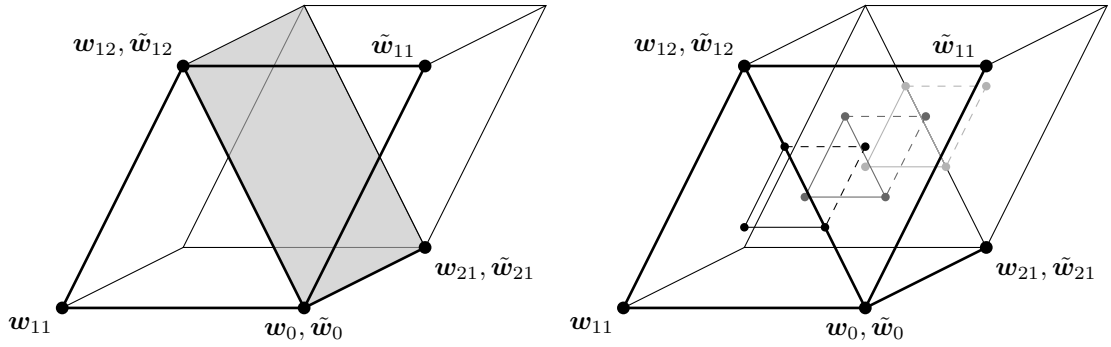
3. B-coefficients and the B-net

The B-coefficients c_λ are generated by collecting the multi-indices in the layers. Their location is again determined by the control points $\boldsymbol{x}_\lambda \in \mathbb{R}^n$

$$\boldsymbol{x}_\lambda = (1-\ell)\boldsymbol{w}_0 + \sum_{i=1}^{\ell}\frac{1}{d_i}\sum_{j=0}^{\nu_i}\lambda_{ij}\boldsymbol{w}_{ij} \tag{13}$$

Note the similarity between Eq. (9) and Eq. (13). The B-net of the (2,1)-multiplex is given in figure 3.

The coefficients can again be collected in a vector $\boldsymbol{c} = (\boldsymbol{c}^{\Gamma_1}, ..., \boldsymbol{c}^{\Gamma_N})$ with $\boldsymbol{c}^{\Gamma_i} = (\boldsymbol{c}^{\Gamma_i}_\lambda)_{|\lambda_i|=d_i, \forall i\in[1,\ell]}$. The coefficients are in lexicographical order based on the multi-index λ.



(a) Shared edge **(b) Continuity conditions**

Figure 4: The shared edge and related continuity conditions between (2,1)-multiplices $\Gamma = \{\boldsymbol{w}_{ij}\}$ and $\tilde{\Gamma} = \{\tilde{\boldsymbol{w}}_{ij}\}$, $i \in \{1, 2\}$. Since the out-of-edge vertex lies in the two-dimensional layer, the shared edge is a (1,1)-multiplex (shaded). Much like the B-net (in figure 3), the continuity conditions are formed by making parallel copies of 2-simplex continuity for each B-coefficient of the second layer. Note that for clarity much of the B-net is left out and each example first order continuity condition is given a different color.

4. Continuity conditions

Continuity between two polynomials π and $\tilde{\pi}$ on ν-multiplices Γ and $\tilde{\Gamma}$ is defined per layer.[9] The conditions in Eq. (6) are applied to the triangulations in each layer first. Then they are copied for each parallel copy of the B-net. In the end the conditions for r^{th} order continuity can be written as

$$\tilde{c}_{(m,\lambda_1,\ldots\lambda_n)} = \sum_{|\varepsilon|=m} c_{(0,\lambda_1,\ldots\lambda_n)+\varepsilon} \mathcal{B}^m_\varepsilon(\boldsymbol{\beta}^\Gamma(\tilde{\boldsymbol{w}}_{10})), \qquad \forall m \in [0, r] \tag{14}$$

where it is assumed that $\tilde{\boldsymbol{w}}_{10}$ is the out-of-edge vertex. That is, the out-of-edge vertex lies in the first layer. Note that $\varepsilon = (\varepsilon_1, ..., \varepsilon_\ell)$ contains many zeros by default, because any adjustment to λ that violates $|\lambda_i + \varepsilon_i| = d_i$ results in a zero B-coefficient. In other words, the degree cannot be changed in other layers than the one containing the out-of-edge vertex. In the case that the out-of-edge vertex lies in the first layer, we therefore have $\varepsilon_i = (0, ..., 0), \forall i \in [2, \ell]$.

The conditions are linear in the B-coefficients, and can therefore again be written as

$$\mathrm{H}\boldsymbol{c} = \boldsymbol{0} \tag{15}$$

The matrix H contains the same conditions as the one in Eq. (6). Each condition (or row) is duplicated multiple times, such that the conditions hold in each parallel copy of the B-net in a layer. This is illustrated in figure 4b.

5. Fitting scattered data

If Eq. (12) and (15) are compared to Eq. (4) and (7) respectively, the conclusion can be drawn that the multiplex spline fitting problem is the same as that of the simplex spline. That means that Eq. (8) can be reused if $\boldsymbol{b}$, B, and $\boldsymbol{c}$ are replaced with $\boldsymbol{\beta}$, $\mathcal{B}$, and $\boldsymbol{c}$.

C. Comparison

The simplex and multiplex spline can be compared on the basis of their defining elements. First the geometric implications of using a multiplex are discussed, followed by the polynomial consequences of choosing layers.

1. The n-simplex versus the ν-multiplex

A major downside of the use of simplices is the distribution of the content (or high-dimensional volume). As the dimension n increases, the content of the n-simplex moves towards the vertices, yielding a very thin polytope. This can be illustrated by plotting the ratio of the content of the n-simplex and its inscribed n-sphere. The same can be done for the n-cube, which is the (1,...,1)-multiplex. Both are plotted in figure

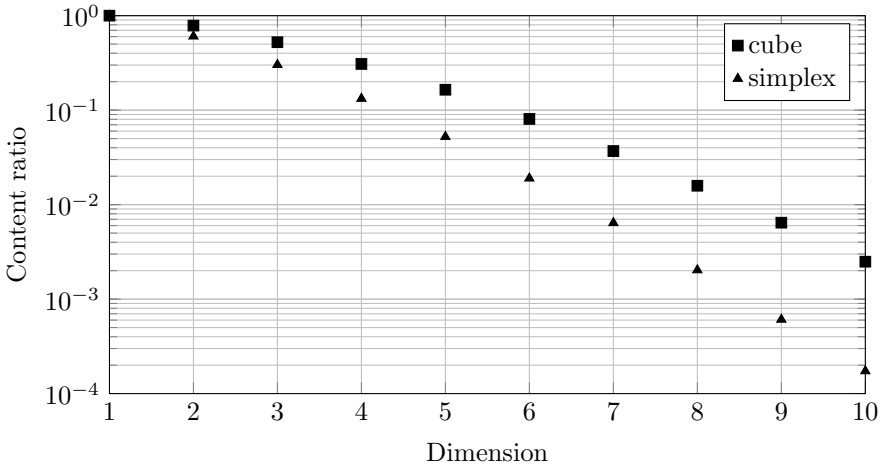


Figure 5: The ratio of the content of the n-simplex and the n-cube to their inscribed circle as a function of dimension n. The n-cube retains more of its content near the center as dimension increases, meaning it is less difficult to fill an n-cube with data.

5. As the ratio decreases, content is moving towards the vertices, meaning the polytope becomes more stretched. In practical terms, it becomes increasingly hard to properly fill each polytope with data. The simplex is the polytope that suffers most from this effect. Using multiplices will thus always improve the content distribution.

A second advantage of the multiplex spline is that a high-dimensional tessellation can be defined by multiple low-dimensional triangulations. If groups can be made of less than four variables, all triangulations can be plotted for visual inspection and optimal triangulation algorithms can be employed.

2. The use of layers

From the polynomial point of view, the multiplex spline presents the opportunity to choose the form of coupling between variables. When two variables lie in the same layer, they become part of a complete multivariate polynomial, resulting in nonlinear (d^{th} degree) cross-coupling between the variables. Separating variables results in a multilinear, tensor product coupling. In a simplex spline all variables will always be coupled.

In many cases different variables require different polynomial degrees for appropriate modeling. In the multiplex spline it is possible to make a subdivision of the variables in layers based on this required degree. This may prevent overfitting or divergence near the edges of the domain of variables for which few data are available.

III. System identification algorithm

The great promise of the multiplex spline lies in the fact that variables can be decoupled within a single spline model. In practice this means that we have a lot more freedom to incorporate our knowledge of the system in the model structure. In this section an approach to choosing the structure is presented as an algorithm.

In short, the algorithm contains the following steps.

1. Group variables and project the data set;
2. Triangulate each projection;
3. Combine triangulations into a tessellation;
4. Fit a multiplex spline;
5. Iterate on step 1 or 2.

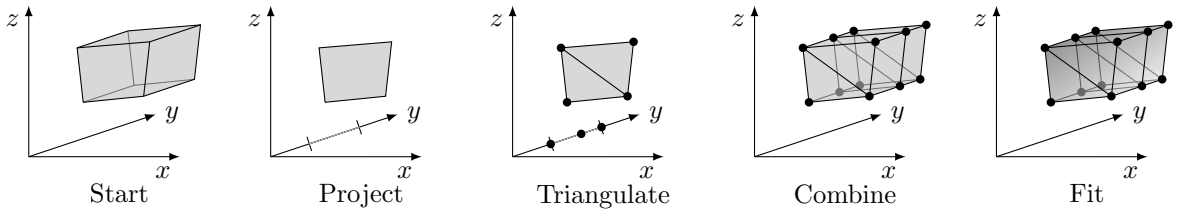


Figure 6: Schematic overview of the steps in the system identification algorithm. The data (transparent gray) are projected onto the chosen axes. The domain in y is bounded with normal lines on the axis for clarity. Next the triangulations are defined and combined. The black circles indicate vertices of simplices. Finally a spline model (fading gray) is fitted.

The complete process is illustrated in figure 6. The individual steps are discussed in more detail in dedicated subsections.

A. Group variables

Groups of variables define layers in the multiplex spline. If an analytic model is available, the coupling between variables should form the basis for this grouping. Variables that require similar degrees and are strongly coupled should together form a layer. If no analytic model is available, the shape of the data set can play a leading role. The goal would then be to make sure that the projections can be triangulated efficiently in the next step.

B. Design triangulations

A triangulation is made in each layer based on the shape of the projection of the data set and the desired degree. With the dimension of the layers the complexity of making triangulations can be scaled. One could even decide to use only one- and two-dimensional layers to allow for the use of simple triangulation optimization algorithms.

C. Generate a tessellation

Combining the triangulations in each layer into a single tessellation is mostly a formal procedure. In many applications it is preferable to consider separate triangulations, e.g. converting input data to barycentric coordinates. If a multiplex contains less data points than B-coefficients, it should be removed from the tessellation.

D. Fit a spline model

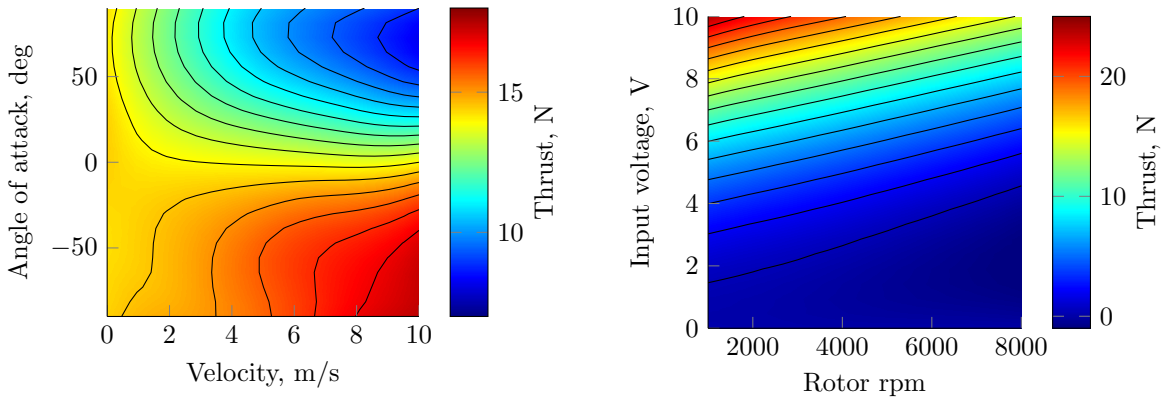
The knowledge of the system is used to choose an appropriate degree and continuity order. This is combined with the tessellation to form a spline. The B-coefficients can be found using the method described in Ref. 12.

E. Iterate

Based on the results obtained in the previous step it is decided if and how the model can be improved. If locally there is a lack of approximation power, simplices can be added in the layers that contain the relevant variables. If there are signs of unmodeled dynamics one can choose to either raise the degree in certain layers or move variables from one layer to the other.

IV. Quadrotor modeling

To proof the applicability of the multiplex spline, it was used to model the thrust of a quadrotor as a function of free-stream velocity, angle of attack, rotor rotational rate, and input voltage to the engine. The analytic model presented in subsection A was used to generate a data set.



(a) Cut at 3000 rpm, 8 V

(b) Cut at $v_\infty = 5$ m/s, $\alpha = 20$ deg

Figure 7: Two-dimensional cuts of the simplex spline model of the quadrotor. The well-known dependence of the thrust on angle of attack and freestream velocity is clearly visible.

A. Analytic model

The analytic model used to generate a data set is mostly based on helicopter theory that has been applied to quadrotors before.[13,14] The core of the model is the ideal thrust

$$T = \frac{\eta P}{v_\infty \sin\alpha + v_i} \tag{16}$$

P signifies the electric engine power input, which is converted to aerodynamic power using a single efficiency η. For the electric power we use[13]

$$P = \frac{U - K_{emf}\omega}{R_a} U \tag{17}$$

The induced velocity v_i is given by the well-known equation

$$v_i = \frac{v_h^2}{\sqrt{(v_\infty \cos\alpha)^2 + (v_\infty \sin\alpha + v_i)^2}} \tag{18}$$

where v_h is the induced velocity in hover that is required to support one quarter of the assumed weight of the quadrotor. In the range of vertical velocity $-2v_h \leq v_z < 0$, where $v_z = v_\infty \sin\alpha$, it is assumed that the rotor is in the vortex ring state. Its induced velocity is then approximated by[15]

$$v_i = v_h \left(k_0 + k_1 \frac{v_z}{v_h} + k_2 \left(\frac{v_z}{v_h}\right)^2 + k_3 \left(\frac{v_z}{v_h}\right)^3 + k_4 \left(\frac{v_z}{v_h}\right)^4 \right) \tag{19}$$

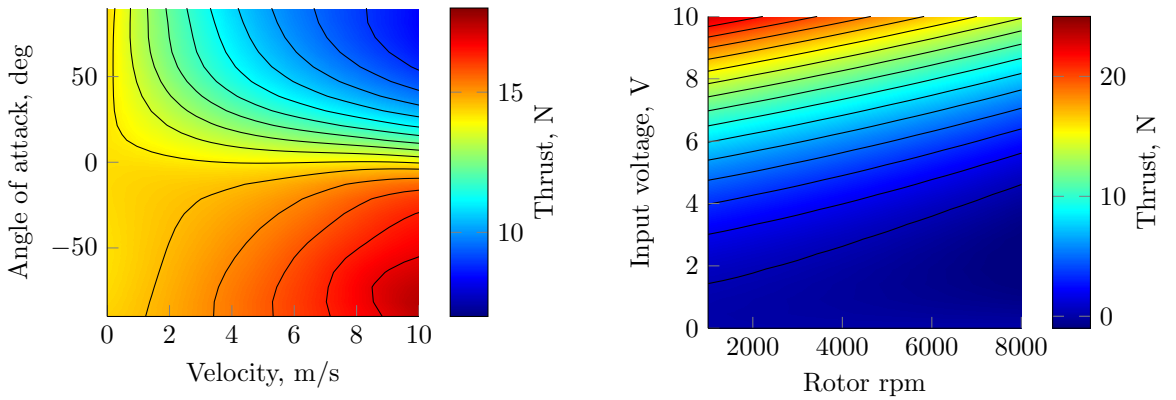
with $k_0 = 1, k_1 = -1.125, k_2 = -1.372, k_3 = -1.718$, and $k_4 = -0.655$.

The data set is generated using a set of 5000 random inputs within the operating range of the motor. Pseudo-random noise bounded to 1% of the maximum thrust is added to the simulated output.

B. Simplex spline model

To fit the data set generated using the analytic model of the previous subsection, a simplex spline is employed first. The construction of the model is inspired by De Visser's approach.[1]

The first step in the system identification process is deciding on the model structure. By filling in Eq. (17), (18), and (19) into Eq. (16) we find a system of equations with four variables: v_∞, α, ω, and u. The spline model will thus be four-dimensional. Because Eq. (18) is a quartic equation, a fourth degree polynomial is required. To limit the freedom of the separate polynomials, first order continuity constraints are imposed. To preserve the simplicity of the model, a standard type-I triangulation of the rectangular domain is used. This results in a total of 22 simplices. The model structure is summarized in table 1.



(a) Cut at 3000 rpm, 8 V

(b) Cut at $v_\infty = 5$ m/s, $\alpha = 20$ deg

Figure 8: Two-dimensional cuts of the multiplex spline model. The well-known relation of thrust, freestream velocity and angle of attack is clearly visible, as well as the quadratic nature of the engine dynamics.

The second step is to use the model structure in the defining equations of subsection A to arrive at the convex problem of Eq. (8). The most time-consuming part of this process is defining the smoothness matrix.

Finally the problem can be solved using the iterative solver described before. A total of 1540 B-coefficients are estimated using the data set of 5000 points.

Several cuts of the spline are shown in figure 7. When a value for rotor rpm and input voltage is chosen, the well-known dependence of thrust on velocity and angle of attack is clearly visible. In the other cuts the quadratic relation between the thrust and the input voltage and rotor rpm can be observed. Especially at higher input voltages the parabolic shape is adhered to. Close inspection however shows that in some cases small non-quadratic variations can be observed.

C. Multiplex spline model

The multiplex spline model is constructed along the lines of section III. An attempt is made to arrive at a model with a comparable number of B-coefficients as in the simplex spline model.

The model should again include the four variables v_∞, α, ω, and u, so it will be four-dimensional. From Eq. (17) and Eq. (18) it is clear that although the induced velocity requires a polynomial model of degree 4, this is not the case for the power. In fact, a quadratic model should suffice to perfectly model the power as a function of rotor rpm and input voltage. In the multiplex spline framework this difference in degrees can be adhered to by choosing appropriate layers.

In the first layer the freestream velocity v_∞ and the angle of attack α are combined. From early iteration of the algorithm it was found that in angle of attack extra approximation power was required. Therefore a triangulation of four simplices was chosen in this layer, generated by splitting the range of angle of attack in half before making a minimal type-I triangulation. In the end this procedure also results in a similar amount of B-coefficients as in the simplex spline model. First order continuity is used to limit the spline freedom, as before.

In the second layer the remaining two variables ω and u are taken together to model the engine dynamics. The simple model allows for a quadratic spline on a minimal triangulation of two simplices. First order continuity is again imposed, even though this is rather high for a quadratic spline. In practice this means that the two piecewise polynomials will almost form a single parabola. The resulting structure of the multiplex spline is compared with the simplex spline structure in table 1.

In combining the triangulations in a tessellation, no extra operations are required. A rectangular data set is used, meaning no multiplices will be (too) empty. The tessellation thus consists of 8 multiplices, together containing 1440 B-coefficients.

The resulting spline model is shown in figure 8. The relation between thrust, freestream velocity and angle of attack is again properly modeled. The main difference is the fact that the engine dynamics are now quadratic by definition. This removes the unwanted higher degree effects in this part of the model without affecting the approximation power in the other layer.

Table 1: Comparison between the simplex and multiplex spline structure used to model the quadrotor thrust as a function of air inflow and engine dynamics. The term 'inflow layer' refers to the layer of variables v_∞ and α.

	Model design parameters				Model characteristics	
	n	d	r	N	B-coefficients	Constraints
Simplex spline	4	4	1	22	1540	1760
Multiplex spline	4	6*	1	8	1440	1548
Inflow layer	*2*	*4*	*1*	*4*	*60*	*27*
Engine layer	*2*	*2*	*1*	*2*	*24*	*15*

* No complete basis (see section II).

Table 2: The relative root mean square error (RRMS), mean square error (MSE, normalized with maximum thrust), and maximum error (ME, in percent of maximum thrust) for both the modeling data set (with noise) and the verification data set. The differences between the two data sets show that no significant overfitting to noise takes place. The overall performance of the multiplex spline model is better than that of the simplex spline model.

	Modeling			Verification		
	RRMS, %	MSE, 10^{-5}	ME, %	RRMS, %	MSE, 10^{-5}	ME, %
Simplex spline	2.397	5.012	4.494	1.608	2.153	3.487
Multiplex spline	1.934	3.269	1.998	0.672	0.376	1.879

D. Model quality assessment and comparison

The comparison between the simplex and multiplex spline models focuses on efficiency. The central question is what approximation power is achieved at what cost.

It was noted before that both models were designed to use similar amounts of B-coefficients. In the end, the simplex spline uses 1540 B-coefficients, whereas the multiplex spline contains 1440. These are constrained using a total of 1760 and 1548 continuity conditions respectively, of which 403 and 405 are redundant. The simplex spline thus requires more continuity constraints per B-coefficient, resulting in a larger smoothness matrix.

The smoothness matrix is not just smaller in the multiplex spline framework, but also less demanding to compute. Whereas the complete smoothness matrix for the simplex spline must be generated based on relations between simplices, the matrix is constructed of copies of smaller matrices in the multiplex case. This leads to a significant reduction in computation time.

To measure the quality of the fit, the relative root mean square error (RRMS, RMS of the error divided by RMS of the data), the mean square error (MSE), and maximum error (ME) are used. The results are presented in table 2. Note that due to the nature of the model, it is possible to generate a verification data set that does not contain noise. The RRMS and MSE values are given for both data sets. In the first place these figures show that no significant overfitting of the noise in the modeling data takes place, since the errors are smaller for the crisp verification data set. Most important is that in every metric the multiplex spline model performs better than the simplex spline.

The differences between the two splines become clear when looking at the error metrics. The relative root mean square error shows that both splines have a small error relative to the system dynamics and noise. The multiplex spline however is almost three times as accurate in predicting system outcome as the simplex spline. The mean square error for the simplex spline is more than a factor 8 higher, indicating that this spline is less capable of modeling the complete dynamics in the way the multiplex spline does. The same can be observed from the difference in maximum error and when comparing figure 7a and 8a. One explanation for this may be that the continuity constraints in the simplex spline are not in line with the system dynamics. That is, they relate B-coefficients that lie at control points that differ in all variables, whereas the constraints in the multiplex spline lie in the chosen layers.

V. Conclusion

In this paper a new multivariate spline was introduced and an algorithm for system identification using this spline was proposed. The thrust of a quadrotor was modeled to show the applicability of the spline and the algorithm.

The major benefit of the multiplex spline over the simplex spline is the fact that it allows for including knowledge of the system in the model. Any subdivision of the variables in layers can be made, depending on the expected type of coupling between them. Due to the nature of the data set used in this paper, the choice for layers was rather straightforward. If no analytic model of the system is present, many iterations of the proposed algorithm are required.

In modeling the quadrotor thrust, the multiplex is able to take full advantage of the decoupling between engine dynamics and aerodynamics. This results in a better fit to the data set than in the case of the simplex spline. Although the differences are small in this specific case, it is expected that especially the option of choosing multiple degrees will introduce significant improvements in the overall model fit. One part of the model can be given high approximation power by using a high degree while simultaneously preventing overfitting in other layers.

An extra benefit of the multiplex spline is the reduced computation time for the smoothness matrix. Because continuity constraints are defined in layers and then copied, the total amount of operations is significantly smaller than in the case of the simplex spline. The current research implies that in many cases the multiplex spline will also require less continuity conditions in general, resulting in faster computations overall.

References

[1]de Visser, C. C., *Global nonlinear model identification with multivariate splines*, Ph.D. thesis, Delft Technical University, July 2011.

[2]de Visser, C. C., Chu, Q. P., and Mulder, J. A., "Global Nonlinear Aerodynamic Model Identification with Multivariate Splines," *Proceedings of AIAA Atmospheric Flight Mechanics Conference*, August 2009.

[3]Tol, H. J., de Visser, C. C., van Kampen, E., and Chu, Q. P., "Nonlinear Multivariate Spline-Based Control Allocation for High-Performance Aircraft," *Journal of Guidance, Control, and Dynamics*, 2014, in press.

[4]Bern, M., Eppstein, D., and Gilbert, J., "Provably good mesh generation," *Journal of Computer and System Sciences*, Vol. 48, No. 3, 1994, pp. 384–409.

[5]de Visser, C. C., van Kampen, E., Chu, Q. P., and Mulder, J. A., "Intersplines: a new approach to globally optimal multivariate splines using interval analysis," *Reliable Computing*, Vol. 17, December 2012, pp. 153–191.

[6]Govindarajan, N., de Visser, C. C., and Krishnakumar, K., "A sparse collocation method for solving time-dependent HJB-equations using multivariante B-splines," *Automatica*, 2014, in press.

[7]Govindarajan, N., de Visser, C. C., van Kampen, E., Krishnakumar, K., Barlow, J., and Stepanyan, V., "Optimal Control Framework for Estimating Autopilot Safety Margins," *Journal of Guidance, Control, and Dynamics*, 2014, in press.

[8]Sun, L. G., de Visser, C. C., and Chu, Q. P., "A new multivariate tensor-product simplex B-spline with its application in real-time modular adaptive flight control," *Proceedings of AIAA Guidance, Navigation and Control Conference*, January 2014.

[9]Visser, T., "A new proof of continuity between tensor-product Bernstein polynomials," *In preperation*, 2014.

[10]Lai, M. and Schumaker, L. L., *Spline functions on triangulations*, Encyclopedia of mathematics and its applications, Cambridge University Press, 2007.

[11]de Boor, C., "B-form basics," *G. Farin, editor, Geometric Modeling: Algorithms and New Trends*, 1987.

[12]de Visser, C. C., Chu, Q. P., and Mulder, J. A., "A new approach to linear regression with multivariate splines," *Automatica*, Vol. 45, No. 12, December 2009, pp. 2903–2909.

[13]Hoffmann, G., Huang, H., Waslander, S., and Tomlin, C., "Quadrotor helicopter flight dynamics and control: theory and experiment," *Proceedings of AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007.

[14]Zawiski, R. and Blachuta, M., "Dynamics and optimal control of quadrotor platform," *Proceedings of AIAA Guidance, Navigation and Control Conference*, August 2012.

[15]Leishman, J. G., *Principles of Helicopter Aerodynamics*, Cambridge University Press, New York, NY, 2000.

Part II

Report

Acronyms

DoF	Degrees of Freedom
MAV	Micro Aerial Vehicle
ME	Maximum Error
MSE	Mean Square Error
NDI	Nonlinear Dynamic Inversion
RMSE	Root Mean Square Error
RRMS	Relative Root Mean Square

List of Symbols

Greek Symbols

α	Angle of attack
$\boldsymbol{\beta}, \beta$	(Vector of) barycentric coordinates in a multiplex
Γ	Multiplex
γ	Layer
$\bar{\Gamma}$	Multiplex shared edge
Δ	Simplex
δ	Layer set
$\bar{\Delta}$	Simplex shared edge
δ_f	DelFly right wing angle
$\dot{\delta}_f$	DelFly right wing angular rate
ζ	Flap phase angle
θ	Body pitch angle
$\boldsymbol{\kappa}, \kappa$	(Set of) multi-index permutations for a simplex basis polynomial
$\boldsymbol{\lambda}, \lambda$	(Set of) multi-index permutations for a multiplex basis polynomial
μ	Multi-index describing a deviation from the dimensions in ν
ν	Multi-index of dimensions in the multiplex
$\boldsymbol{\xi}, \xi$	(Vector of) weights for the biased multiplex
π	Multiplex polynomial
ρ	Air density
ρ	Multi-index describing a deviation from λ

σ	Multi-index describing a deviation from κ
τ	Engine loading (torque)
χ^i	Special case of ϕ with all zero entries except for the i^{th} element.
ϕ	Vector of indices of the vertices taken from layer sets
ω	Rotor rpm

Roman Symbols

A	Rotor disc area
$\boldsymbol{b}, b$	(Vector of) barycentric coordinates in a simplex
B	Matrix of Bernstein basis polynomials on a simplex
$\mathbb{B}^n$	Set of barycentric coordinates in n dimensions
$\boldsymbol{\mathcal{B}}^d_\nu$	Vector of Bernstein basis polynomials of degree d on a ν-multiplex
$\boldsymbol{B}^d_n$	Vector of Bernstein basis polynomials of degree d on an n-simplex
$\mathcal{B}^d_\lambda$	Multiplex Bernstein basis polynomial of degree d with multi-index λ
B^d_κ	Simplex Bernstein basis polynomial of degree d with multi-index κ
$\boldsymbol{\mathcal{C}}, c$	(Set of) B-coefficients of a multiplex spline
C^r	Space of r^{th} order continuous functions
$\boldsymbol{C}, c$	(Set of) B-coefficients of a simplex spline
C_T	Thrust coefficient
d	Degree of basis polynomials, or multi-index of degrees
e	Degree of the equivalent simplex spline
H	Smoothness matrix
I	Engine input current
ℓ	Number of layers in a multiplex
$\mathcal{M}$	Tessellation consisting of multiplices
m	Deviation from the dimension n; dimension of the equivalent simplex spline
N	Number of elements in a tessellation or triangulation
n	Total dimension
$\mathbb{N}^n$	Set of natural numbers up to n
P	Power
p	Simplex polynomial
q	Body pitch rate
q	Control point
R	Rotor radius

r	Continuity order
$\mathbb{R}^n$	Set of n-dimensional real vectors
s_ν^d	Multiplex spline of degrees d and dimensions ν
$\mathcal{S}_n^d$	Space of polynomials of degree d and dimension n
s_n^d	Simplex spline of degree d and dimension n
T	Rotor thrust
$\mathcal{T}$	Triangulation consisting of simplices
$\mathcal{U}, u$	(Set of) vertices of a multiplex in the top-down definition
U	Engine input voltage
u	Body x-velocity
w	Body z-velocity (positive down)
$\mathcal{V}, v$	(Set of) vertices of a simplex
v_∞	Freestream velocity
v_h	Induced velocity in hover
v_i	Induced velocity
v_z	Vertical velocity component, $v_\infty \sin\alpha$
$\mathcal{W}, w$	(Set of) vertices of a multiplex in the bottom-up definition
X	Matrix of input data
$\boldsymbol{y}$	Vector of output data
$\mathbb{Z}_+^n$	Set of positive integers up to n
$\mathbb{Z}^n$	Set of non-negative integers up to n

Summary

In recent years the introduction of model-based control techniques has lead to an increased need for accurate, smooth, non-linear models. Rather than using data tables, many research groups employ neural networks for such purposes. Recently a multivariate simplex spline-based approach to system identification was introduced. Although these simplex splines show great promise, it is still unclear how a high-dimensional triangulation consisting of simplices can be shaped to incorporate knowledge of the system. In general high-dimensional triangulations consist of standard triangulations of stacked hypercubes.

In this work a tensor-product spline is developed and made ready for scattered data fitting. It is based on previous efforts to combine multiple simplex splines to simplify computing partial derivatives. The goal of this work is to expand the theory of these tensor-product splines such that system identification can be performed and to develop and validate an algorithm for such applications.

To allow for covering non-rectangular domains the generalization starts at the geometric basis. Instead of defining a spline for each B-coefficient the concept of a multiplex is introduced. This polytope is formed by combining simplices that lie in mutually orthogonal spaces. It is found that this multiplex is also a slice of a higher-dimensional simplex. Both the simplex and the hypercube are special cases of the multiplex.

The definition of basis polynomials in the multiplex spline framework can be derived from the literature. Basis polynomials are formed by taking the tensor-product of Bernstein polynomials in barycentric coordinates. Together they form a basis for the tensor-product polynomial space. The basis polynomials preserve the non-negativity and partition of unity properties. It is found that the set of basis polynomials in the multiplex spline is also a scaled subset of that of a higher-dimensional simplex spline.

The B-net can be considered to be a Cartesian product of individual B-nets or as a slice of the B-net of a simplex spline. This insight, combined with the discoveries with respect to basis polynomials and the multiplex itself, leads to the concept of the equivalent simplex spline. This is a simplex spline of such degree and dimension that the corresponding multiplex spline is a slice of it in all its facets: geometric basis, basis polynomials and B-net.

The equivalence between simplex spline and multiplex spline allows for a very lean proof of continuity between multiplices. Because of the equivalence between the B-nets, continuity conditions in the equivalent simplex spline can be used directly in the multiplex spline framework. It is found that an arbitrary continuity order can be imposed over a complete shared $(n-1)$-edge, irrespective of the shape of the out-of-edge layers, as long as the multiplices are both n-dimensional.

The multiplex spline was tested on a set of test functions to get a view on its characteristics and behavior. On the one hand this behavior is strongly influenced by the content distribution in the multiplex. Content of this polytope is more concentrated in the center of the polytope than in the simplex, which makes it easier to fill a multiplex with data points. The ability to choose different degrees for variables in different layers proved to be beneficial when a function is more complex to approximate in some variables than in others. The total degree however rises quickly and may cause large excursions along the diagonal of the domain.

Based on the characteristics of the spline a system identification algorithm is constructed. The algorithm requires a choice of global layers, in which groups of variables are combined. The layers are triangulated and then combined to form a tessellation. In each layer a different degree and continuity order can be chosen, according to the need. The algorithm was verified on an analytic model of a quadrotor, where the multiplex showed superior performance over the simplex spline.

The validation of the algorithm was performed on DelFly flight test data. This data was first analyzed to find that it contains many voids and is poorly distributed over the state space. Several regions can be identified in which the data is closely packed in all states. In most cases these regions can be linked to flapping stages during maneuvers. As a result of the poor distribution of the data, the performance of the multiplex spline is poor. It does however perform better than the simplex spline in terms of the mean error and the amount of coefficients required. This confirms the validity of the multiplex spline and the system identification algorithm.

Because of the novelty of the spline in engineering applications, many recommendations can be given for future research. With respect to fundamental research, it is interesting to investigate whether each multiplex can be triangulated and under what conditions a circumscribed triangulation can be defined. To improve the performance of the spline, differential constraints can be introduced to reduce the degree along the diagonal and mixed grid partitions can developed. Finally recommendation regarding the global modeling of the DelFly are given, as well as the suggestion to use the multiplex spline in nonlinear dynamic inversion control.

Chapter 1

Introduction

With the increased use of model-based control techniques and the increased complexity of the systems to be controlled, the need for nonlinear function approximators has grown. Although industry still prefers data tables, many research groups have switched to neural networks. The benefit of the latter lies both in the fact that they require less computer memory and are smooth by nature. A major downside however is the fact that basis functions are global, complicating local updates and the use of efficient solution techniques.

Recently the multivariate simplex spline was added to the system identification field [16, 17]. This technique has local basis functions and is linear in the parameters. Therefore many efficient solution techniques can be employed, also for making local updates. As basis functions are defined on simplices, the choice of a triangulation plays a major role in the definition process. There is however no consensus yet on how to define such a triangulation for system identification purposes. In two or three dimensions shape criteria have been formulated for simplices [3, 37, 38], but in more than three dimensions the best solution is generally to use a standard split of the hypercube [16].

To circumvent the problem of high-dimensional triangulation, a new spline is introduced in this thesis. The multivariate multiplex spline allows for combining multiple lower-dimensional triangulations to span a high-dimensional state space. This approach was first introduced by Govindarajan et al. and used in a system identification application by Sun et al. [22, 23, 40]. The novelty of this thesis lies in a redefinition of the combination of triangulations. Instead of describing B-coefficients with multivariate simplex splines, multiplices are formed by combining one simplex from each triangulation. This results in a tessellation of the high-dimensional space consisting of polytopes that can be deleted or added at will. A generalization effort shows that both tensor-product Bernstein polynomials and the simplex spline are special cases of the multiplex spline.

The goal of this thesis is to introduce the mathematical basis of the multiplex spline and to verify and validate a system identification method using these splines. The verification is performed both on test functions and on an analytic model of a quadrotor. The quadrotor is used here because it is largely understood and many analytic models are available. Validation

of the proposed system identification algorithm is done on a DelFly data set. The DelFly is a flapping wing Micro Aerial Vehicle (MAV) developed at the Delft University of Technology with has a span of 27.4cm and weight of 18 grams. In the process the data set is analyzed in the state-space, resulting in recommendations regarding future system identification efforts and flight tests.

The remainder of this report is structured as follows. First the research goals and questions are formulated in chapter 2. An extensive overview of the literature used for defining the spline, setting up the analytic quadrotor model and the analysis of the DelFly data can be found in chapter 3. To support the reader with limited knowledge of splines this chapter includes a concise discussion of the simplex spline and tensor-product Bernstein polynomials in section 3-1. The multiplex spline itself is introduced in great detail in chapter 4. In chapter 5 the multiplex spline is verified as a system identification tool by testing its performance on several test functions and data distributions. The system identification algorithm for the multiplex spline is validated using DelFly flight test data in chapter 6. This chapter also includes an analysis of the data set itself in the state-space in section 6-1-2. In chapter 7 the conclusions and recommendations regarding the research described in this report are given. Finally in chapter 8 several possible future research topics are presented. They range from fundamental mathematical topics to engineering applications.

Chapter 2

Problem statement

In this chapter the goals and research questions for the current research are presented and discussed. They were formulated by combining the research proposal and an initial literature study. Because of the nature of the research there is a clear division in two parts. On the one hand a fundamental mathematical discussion should lead to a description of the newly developed spline. On the other hand the spline should be made ready for and applied in a system identification task.

Although there are clearly two sides to the research, it is useful to formulate a single goal for the entire research. The goal that encompasses the entire research is

to present a solution to the multivariate spline structure definition problem by developing and validating a new multivariate spline and related system identification algorithm that allow for the use of multiple coupled multivariate spline spaces.

In this goal the two parts of the research are clearly identifiable.

The remainder of this chapter is split in multiple sections to clearly distinguish between the two parts of the research. First in section 2-1 the goals and research questions of the investigation into multivariate multiplex spline theory are discussed. Then in section 2-2 the goals are set for the application to system identification.

2-1 Multivariate multiplex spline theory

The first goal of this research is the development of the multivariate multiplex spline's mathematical basis. The spline needs to be properly defined before it can be used in a system identification procedure. Therefore the first subgoal is

to facilitate the use of the multivariate multiplex spline as a function approximator by deriving its defining elements, characteristics and limitations.

In practice this goal will lead to an investigation of the basis polynomials and continuity conditions in the multiplex spline, as well as more fundamental research into the shape of the multiplex. This is reflected in research question 2.

Another important aspect of the first subgoal is the dependence on the application. Because the aim is to introduce a new system identification tool, a limited set of properties and characteristics needs to be derived. The above considerations lead to the following list of research questions.

1. **What defining elements and characteristics of a spline are required for approximating scattered data?**
   Fitting scattered data does not require a complete definition of the properties of a function approximator. A list of required definitions is used to provide the right focus. This list can be constructed through a literature study.

2. **What is the effect of taking the tensor product of multivariate simplex B-splines on the spline's defining elements and characteristics?**
   Based on the list of characteristics resulting from the previous question, the actual derivation of the multiplex spline properties should be performed. Note that the formulation of this question already points towards a solution procedure. The observation that the multiplex spline consists of tensor products of simplex splines should provide insight in its properties. The defining elements will follow from analysis, whereas properties such as the approximation power will be estimated using test functions.

From the answers to the above questions it should become clear what the limitations are to the use of the multiplex spline. This knowledge is essential in the second part of the research, in which the spline is applied in a system identification procedure.

2-2 Application to system identification

The second part of this research focuses on the application of the multiplex spline to system identification problems. The goal therefore is

to facilitate the use of the multiplex spline for system identification problems by formulating and validating an algorithm for choosing the multiplex spline structure.

For testing purposes the thrust and torque of a quadrotor will be modeled. The validation process is performed on DelFly flight test data. This change in platform is made because the DelFly is not yet completely understood. Few analytic models exist, which makes it hard to generate a data set and analyze the results. The quadrotor on the other hand has been modeled often, and is more closely related to the well-studied helicopter field. Many analytic models are available and the outcome of these models is largely known. This facilitates the verification process.

Clearly the goal is twofold. An algorithm for system identification should be formulated first. This requires knowledge of high dimensional data, quality triangulations and many other related topics. Below the relevant research questions related to the construction of the algorithm are listed.

3. **Can a set of rules be formulated for choosing layers based on an analytic model structure?**
   It is expected that when variables are clearly not coupled, they should be distributed over different layers. This hypothesis is tested both on test function designed specifically for this goal, and during the verification process on the quadrotor analytic model.

4. **What methods exist for representing high-dimensional data?**
   This is an important question when constructing the tessellation. If a high-dimensional data set is available it is not trivial to visually inspect it, and consequently to construct a triangulation in a layer. A superficial literature study will be performed to inspire techniques of visualizing the flight test data of the DelFly.

5. **What quality criteria exist for low-dimensional triangulations in data approximation applications?**
   If triangulations are to be constructed in the layers, it is valuable to use quality measures in the process. Because layers can be chosen at will, it is interesting to look at quality measures for low-dimensional triangulation. These measures can be found in the literature.

Note that the above questions mostly cover the structure of the model. It is expected that the methods for coefficient estimations follow directly from simplex spline theory.

The second part of the goal focuses on testing the algorithm. The research questions below are formulated such that they can be applied to any desired application.

6. **Can the algorithm be automated?**
   This is especially interesting if the system under investigation is not well known, as is the case for the DelFly. Automation allows for generating a large set of models, which provides insight into the required variables and approximation power.

7. **How does the performance of the multiplex spline in the system identification problem depend on the layer structure?**
   To answer this question a list of criteria for assessing the approximation performance is required. The simplex spline is used as the benchmark because it is so similar in structure. This allows for a straight forward comparison of efficiency instead of power. That is, the approximation power can be assessed with respect to the number of coefficients and the size of the problem. In general a large set of layer structures is tested on multiple test functions to come to a complete overview.

Apart from the research questions that aim directly at reaching the research goal, it is important to get a view of the systems modeled. This is essential for applying the algorithm. In summary this amounts to identifying the relevant phenomena, variables and analytic models for the system under consideration. This holds for both the quadrotor and the DelFly.

8. **What phenomena significantly affect the aerodynamic forces on the system?**
   Both dynamic and aerodynamic phenomena should be considered. For the quadrotor it is useful to select a dominant set of phenomena by taking into account that it is light and

small. For example, blade flapping is expected to be small under normal flight conditions compared to most other effects. For the DelFly the information regarding aerodynamic effects is scarce as it is a very new and not yet completely understood platform.

9. **What variables should at least be included in a model for these aerodynamic forces?**
   From the list of phenomena it should be clear what variables are relevant for the modeling task.

10. **In what domain are the phenomena active or dominant?**
    The domains of interest are important in constructing a tessellation. In domains where highly nonlinear effects are dominant, a more dense tessellation can be employed to locally boost the approximation power.

The questions 8, 9 and 10 can all be answered using a literature study for the quadrotor. Regarding the DelFly, these questions can only be partially found in the literature. Many dynamic and aerodynamic effects are still unknown, but those that are available are also discussed in the literature review. Any further investigation is performed in data analysis.

Chapter 3

Literature review

In chapter 2 it was noted that some of the research questions could be answered with a review of the literature only. On top of that it was indicated that an orientational review was performed to arrive at many of the research questions. The results of both these reviews are collected in this chapter.

In section 3-1 the research into multivariate splines is considered. A general introduction is provided in which many different types of splines are presented. Then the focus shifts towards the simplex spline and types of tensor-product splines. In the discussion of the simplex spline several triangulation quality criteria are also discussed. In section 3-2 the available literature on analytic quadrotor modeling is reviewed. Recent efforts of modeling the DelFly flapping wing MAV are discussed in section 3-3. On top of that, a subsection is devoted to describing some existing methods for interpreting high-dimensional data sets.

3-1 Multivariate spline theory

In the most general sense, a spline is a piecewise polynomial function in which continuity between the pieces can be defined. It presents an alternative to high-degree polynomials that suffer from Runge's phenomenon near the edges. By using multiple low-degree polynomials the approximation power can increased without increasing the polynomial degree.

Univariate splines have been a well-defined mathematical concept for many years. The generalization to the multivariate case however knows many forms. The polyhedral spline of Neamtu is considered the true multivariate generalization [34], but is impractical for most engineering applications. This is because it requires the definition of an $(n + d)$-dimensional tessellation, with n the dimension and d the degree of the spline. Even though multiple simplifications have been introduced [14, 33], no algorithms exist for the construction of these tessellations based on scattered data.

More practical multivariate splines are the simplex spline and several types of splines constructed as tensor-products of univariate splines. They are discussed in more detail, starting

with the simplex spline in subsection 3-1-1. The defining elements discussed in that subsection were taken from the work of De Visser, so they are assumed to be required for system identification [16]. In subsection 3-1-2 the concept of the tensor-product spline is introduced and the most relevant types are discussed.

3-1-1 The multivariate simplex spline

The multivariate simplex spline, a collection of triangular B-patches, was first introduced by de Casteljau to be used in the design of car parts [20]. An important contribution was given by De Boor, who introduced the B-form [13]. Only recently was the simplex spline used for system identification purposes by De Visser [16] and in nonlinear control by Tol et al. [42]. Below the most important results of these developments are summarized. Many can be found in the work of Lai and Schumaker [30], in which the bivariate simplex spline is considered.

The simplex spline is discussed from the bottom up. That is, the geometric basis and coordinate system are discussed first. Then the basis polynomials are described, followed by the corresponding B-coefficients and their spatial location. On the basis of this so called B-net, continuity conditions can be defined. Finally the scattered data approximation problem is formulated and a solution method is proposed.

Simplices and barycentric coordinates

The multivariate simplex spline consists of Bernstein polynomials defined on triangular patches, or more generally simplices. In n dimensions an n-simplex is defined as the convex hull of $n+1$ non-degenerate points. In two dimensions this results in a triangle, which is the most studied case of the simplex.

The advantage of using simplices over other polytopes lies in the fact that any point $\boldsymbol{x} \in \mathbb{R}^n$ can be described uniquely with respect to the vertices $\mathcal{V} = \{v_0, ..., v_n\}$ of a simplex. Such a description takes the form of a linear combination.

$$\boldsymbol{x} = \sum_{i=0}^{n} b_i v_i \tag{3-1}$$

In the above equation the vector of weights $\boldsymbol{b} = (b_0, ..., b_n) \in \mathbb{R}^{n+1}$ are called the barycentric coordinates. These coordinates always sum to one.

Barycentric coordinates also present a way of determining if a point lies inside a given simplex. If all elements of $\boldsymbol{b}$ relative to the vertices of a simplex Δ are non-negative, the point lies inside or on the boundary of Δ [16].

Triangulation and quality criteria

By combining multiple simplices of the same dimension, a triangulation can be constructed. When collecting all barycentric coordinates with respect to a triangulation, only the simplex in which the point lies is considered. The other elements of the total coordinate vector are

set to zero. This also leads to a global coordinate vector which consists of the appropriate local coordinate vector, padded with $n+1$ zeros for each other simplex.

In the current discussion a triangulation consisting of n-simplices is considered valid if none of the simplices overlap, or in other words share n-dimensional content, and simplices share complete m-edges, where $0 \leq m < n$, or are completely disjoint. These conditions can also be described mathematically for a triangulation consisting of N simplices Δ_i as [16]

$$\mathcal{T} = \bigcup_{i=1}^{N} \Delta_i, \qquad \Delta_i \cap \Delta_j \in \{\emptyset, \bar{\Delta}\}, \forall i \neq j \tag{3-2}$$

In this description $\bar{\Delta}$ is an m-simplex with $0 \leq m < n$. In two dimensions the first condition means that no part of the domain with nonzero area can lie inside two simplices at the same time. The second condition indicates that simplices either share a complete side (1-simplex), a vertex (0-simplex) or nothing at all.

The two conditions for a valid triangulation leave a lot of freedom in triangulating a domain. The number of simplices can be chosen freely, as well as their shape. These two parameters however have a great influence on the quality of the resulting spline model. Therefore a set of quality requirements should be formulated to allow for consistent construction of appropriate triangulations.

In the literature many different quality requirements can be found, mostly for two-dimensional triangulation. In finite element applications the number of simplices should be limited to bound the computation power and time [3, 37]. On the other hand the mesh should be sufficiently fine to allow for the appropriate approximation power.

For the shape of the individual simplices, the following list of criteria can be found.

- The minimum in-plane angle between sides should not be too small [3, 38];
- The largest in-plane angle between sides should not be obtuse [3];
- The aspect ratio of a simplex (ratio of longest edge to shortest altitude for triangles) should be bounded [37];
- The center of the circumscribed circle should lie inside the simplex [3, 16, 38];
- The ratio of the radius of the circumscribed circle to the shortest simplex ridge should be small [16, 38];
- The simplex should contain at least as many data points as there are B-coefficients in a simplex [16].

Note that the second and fourth criteria are different formulations of the same condition [3].

In system identification tasks the last criterion, concerning available data, is most important. Without meeting this criterion it is not possible to find a best estimate for the spline parameters. On the other hand, so called slither simplices (with small or obtuse angles) may cause numerical problems [16].

It is also possible to define criteria for an entire triangulation. The most notable criterion is the one presented by De Visser et al., stating that the best triangulation is the one that results in the smallest overall approximation error [19]. To change the location of vertices of a triangulation they use interval analysis, resulting in long computation times. This possibility is therefore not considered in this work.

B-form polynomial

Polynomials can be defined on a simplex in many ways. In the current discussion the B-form introduced by De Boor is used [13]. In this form the polynomial is a weighted sum of Bernstein basis polynomials defined in barycentric coordinates.

$$p(\boldsymbol{b}) = \sum_{|\kappa|=d} c_\kappa B_\kappa^d(\boldsymbol{b}), \qquad B_\kappa^d(\boldsymbol{b}) := \frac{d!}{\kappa!}\boldsymbol{b}^\kappa \tag{3-3}$$

The weights c_κ are called B-coefficients and are discussed in the next part. The multi-index $\kappa = (\kappa_0, ..., \kappa_n), \kappa_i \in \mathbb{Z}^d$ is the driving force behind the definition of the Bernstein polynomials. Every permutation of this multi-index for which $|\kappa| = d$ leads to a Bernstein basis polynomial. Together the complete set of polynomials forms a basis for all polynomials of degree d in n variables, $\mathcal{S}_n^d$.

Like the barycentric coordinates, the basis polynomials can be collected in a global vector. This is done in the same way as before, by padding the local vector of basis polynomials with sufficient zeros for each simplex. That is

$$\boldsymbol{B}_n^d(\boldsymbol{b}(\boldsymbol{x})) := \begin{cases} B_\kappa^d(\boldsymbol{b}^{\Delta_i}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \in \Delta_i \\ 0 & \text{otherwise} \end{cases} \tag{3-4}$$

In the above equation $\boldsymbol{b}^{\Delta_i}(\boldsymbol{x})$ indicates that the local coordinates with respect to simplex Δ_i are used.

The advantage of Bernstein polynomials is that they form a stable local basis. That is, they form a basis for all polynomials of a particular degree and do so within finite bounds set by the B-coefficients [30]. A major contributor to this stability is the partition of unity property. This means that the sum of the complete set of basis polynomials of a given degree is always one over the entire simplex. On top of that the basis polynomials are non-negative over their basis.

B-coefficients and the B-net

The weights c_κ in equation (3-3) are called B-coefficients. Contrary to many other definitions of polynomials, these coefficients have a spatial location in the simplex in which they are defined. These locations are the control points $q_\kappa \in \mathbb{R}^n$ defined as

$$q_\kappa = \frac{1}{d}\sum_{i=0}^{n} \kappa_i v_i \tag{3-5}$$

Note how the location is completely defined by the corresponding multi-index κ. More importantly, the location of a B-coefficient c_κ coincides with that of the maximum of the corresponding Bernstein basis polynomial B^d_κ.

The collective of all B-coefficients that lie in a simplex and correspond to a certain polynomial degree, is called the B-net. An example for two neighboring simplices is given in Figure 3-1. This concept will proof to be a useful tool in defining continuity between simplices.

The B-coefficients are generally collected in a global vector $\boldsymbol{C}$ with equal length as the basis polynomial vector. This allows for replacing the sum in equation (3-3) with an inner product of the vectors $\boldsymbol{C}$ and $\boldsymbol{B}^d_n$.

Continuity conditions

Continuity conditions can be defined between simplices that share a complete $(n-1)$-dimensional edge. In many cases only geometric continuity is considered. This is a type of continuity popular in structural design, because it relates directly to the visual continuity between two elements. To be more precise, first order geometric continuity (G^1) is defined as the situation in which two patches have a continuously varying tangent plane along the shared boundary [20]. This form of continuity is less restrictive than normal continuity (C^r), in which the derivatives up to an including order r on either side of the shared edge are set equal [30]. Because continuity is used in system identification applications to ensure a sufficiently smooth function, geometric continuity is not investigated any further.

To define continuity between two polynomials p and $\tilde{p}$ on simplices Δ and $\tilde{\Delta}$, the derivatives should be equal on either side of the shared edge in n linearly independent directions. The first $n-1$ directions can be covered by setting all B-coefficients c_κ in Δ that lie on the shared edge equal to their counterparts in $\tilde{\Delta}$. By doing this, the polynomial on the shared edge is equal on both sides, leading to full continuity in that edge [30].

The last independent direction is best chosen from one vertex of the shared edge to the out-of-edge vertex $\tilde{v}_{ooe}$. By doing this, the following continuity conditions result for C^r joins of simplex polynomials [30].

$$\tilde{c}_{(m,\kappa_1,\dots\kappa_n)} = \sum_{|\sigma|=m} c_{(0,\kappa_1,\dots\kappa_n)+\sigma} B^m_\sigma(\boldsymbol{b}^\Delta(\tilde{v}_{ooe})), \qquad \forall m \in [0,r] \tag{3-6}$$

Note that here it is assumed that in both Δ and $\tilde{\Delta}$ the first vertex is the out-of-edge vertex. If the naming of vertices is different, the m and 0 entries in κ shift along. In the conditions σ is again a multi-index with the same number of entries as κ.

Equation (3-6) clearly indicates that continuity conditions are defined almost purely in terms of B-coefficients. In fact, one can visualize the continuity conditions in the B-net. In Figure 3-1 an example first order continuity condition is drawn for a bivariate cubic B-net. The condition forms a miniature version of the simplex, also in more dimensions.

Because of the linearity in the parameters of the continuity conditions, they can be collected in a smoothness matrix H such that $\mathrm{H}\boldsymbol{C} = \boldsymbol{0}$ [17]. The resulting matrix is very sparse, with in each row a maximum of $1 + \frac{(d+n-1)!}{(d-1)!n!}$ nonzero elements, which is the amount of coefficients used in a single condition given in equaton (3-6).

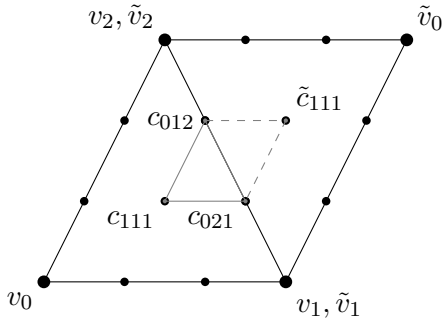


Figure 3-1: A first order continuity condition in a bivariate cubic simplex spline. Note that the condition forms a miniature of the simplex in the B-net of Δ, whereas in $\tilde{\Delta}$ only a single B-coefficient is affected.

The approximation problem

The problem of finding an optimal approximation of the data is a constrained least squares problem.

$$\begin{aligned} &\min ||\boldsymbol{y} - \mathrm{B}(\boldsymbol{b}(\mathrm{X}))\boldsymbol{C}|| \\ &\text{subject to: } \mathrm{H}\boldsymbol{C} = \mathbf{0} \end{aligned} \tag{3-7}$$

In this expression X is a matrix containing the input data and $\boldsymbol{y}$ is the output vector. This problem was solved by Awanou et al. [2] and made fit for the use of standard parameter estimation techniques by De Visser et al. [17]. They rewrite the problem to a Karush-Kuhn-Tucker system as

$$\begin{bmatrix} \mathrm{B}^\top\mathrm{B} & \mathrm{H}^\top \\ \mathrm{H} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{C} \\ \boldsymbol{\psi} \end{bmatrix} = \begin{bmatrix} \mathrm{B}^\top\boldsymbol{y} \\ 0 \end{bmatrix} \tag{3-8}$$

where ψ are the Lagrangian multipliers. De Visser et al. go on to prove that this system is solvable if in each simplex there are at least as many data points available as there are coefficients to estimate.

Solving the system involves taking the pseudo inverse of the left matrix in equation (3-8). This is rather time consuming if many data points and simplices are considered. Iterative solvers can however be employed to speed up the process [16]. Because of the sparsity of both H and B it is also possible to use distributed solution methods. Boyd et al. provide a review of such methods [7].

The model that results can be tested in many ways for consistency and performance. An extensive discussion is provided by De Visser [16]. Most notable are the maximum error to find outliers, the parameter covariance to detect overfitting and the root mean square error to assess overall approximation performance.

3-1-2 Tensor-product splines

In the tensor-product framework the general idea is to create a surface by displacing a univariate curve while changing its shape [20]. Even though this concept seems simple enough, at least two different forms of tensor-product spline can be distinguished. The most important difference between the types lies in the use of knots.

The standard tensor-product spline is defined using knots. Every piecewise polynomial has a nonzero value over a predefined domain bounded by the amount of interior knots. By making the polynomials overlap over the stretch of certain amounts of knots, continuity can be introduced into the spline. This spline however is inherently unable to fit scattered data [1].

A more suitable approach for function approximation is the tensor-product Bézier patch. This patch comes in two forms, depending on the type of coefficient. In most computer-aided design applications the coefficients are control points with a spatial location in $\mathbb{R}^3$ that can be freely chosen. However, in function approximation and scattered data fitting applications the coefficients are best considered single variables, in which case we may speak of tensor-product Bernstein polynomials [21]. The location of the coefficients is then dictated by the form of the corresponding polynomials, just like in the case of the simplex spline.

In the remainder of this section some of the important properties of these tensor-product Bernstein polynomials are discussed. This is combined with recent efforts by Govindarajan et al., who generalized such splines to a more flexible form [22].

Orthotopes and coordinate systems

When univariate polynomials are combined they together form orthotopes, or rectangles in two dimensions. Because tensor-product Bernstein polynomials are most often used in computer-aided design applications, the bivariate case is discussed the most in the literature.

An orthotope in general is most easily defined using an iterative definition, inspired by the definition of a parallelotope of Kendall [28]. Starting from a point as the 0-orthotope, the $(n+1)$-orthotope is the polytope traced out by moving the n-orthotope in a direction perpendicular to all its edges. So, by moving the point we obtain a line, by moving the line we obtain a rectangle, than a box, and so forth.

Describing a point in a rectangle is best done by defining the locations of its perpendicular projections onto the sides. This is often done using barycentric coordinates in the separate directions [20, 21]. The resulting description is not as lean as the one for points in a simplex, as it requires more variables and constraints. For an n-orthotope $2n$ coordinates are required and n equality constraints, all from the definition of barycentric coordinates.

Instead of combining univariate polynomials, one can also combine multivariate polynomials. Govindarajan et al. observe that this results in polytopes other than simplices [22]. Lai takes a similar approach, but does not investigate general, n-dimensional cases [31]. His discussion is therefore limited to bases with a maximum of one multivariate polynomial. The implications of the more general approach are a major topic of this report, and more specifically chapter 4.

Tessellation and quality criteria

The triangulation discussed in section 3-1-1 is a specific form of tessellation. In general a tessellation is any collection of polytopes that do not overlap and together cover some domain. That is, the definition in expression (3-2) can be reused, but now Δ_i can be any n-dimensional polytope. $\bar{\Delta}$ then signifies any lower-dimensional edge.

To constrain the current discussion a set of properties of tessellations is derived from Grünbaum [24]. All tessellations are assumed to be packings, meaning they show no overlaps. Although gaps are generally allowed, for now we only discuss covering tessellations. In most cases we will consider monohedral tessellations, meaning all elements (or prototiles) are of the same shape, with the relaxation that not all elements are of the same size and proportions. The only exceptions to this rule are section 4-4 on continuity between unequal polytopes and the recommendation for future research in section 8-2.

Contrary to triangulation, few quality criteria exist for tessellations of rectangles. This is not only due to the difference in popularity, but also due to the inherent characteristics of the rectangle. After all, the angles need not be constrained, nor does its inscribed circle. The only reasonable criterion would be that it contains enough data points to define a polynomial of desired degree.

One benefit of using other domain elements than the simplex, is that the simplex has a very unfavorable content distribution. As the dimension increases, the content of the polytope tends towards the vertices, away from the center. This fact can be verified mathematically by comparing the content of the polytope and its inscribed circle [41]. With the help of standard works of Coxeter and Kendall, these calculations can be made for both the n-cube and the n-simplex [12, 28]. The results are presented in subsection 5-1-1.

Basis polynomials

The basis polynomials for the tensor-product spline follow directly from the tensor-product of Bernstein basis polynomials [20]. In the bivariate case we find

$$\pi(\boldsymbol{x}) = \sum_{\lambda_0+\lambda_1=d_1} \sum_{\lambda_2+\lambda_3=d_2} c_\lambda \mathcal{B}^{d_1}_{(\lambda_0\lambda_1)}(x_1)\mathcal{B}^{d_2}_{(\lambda_2\lambda_3)}(x_2) \tag{3-9}$$

with the Bernstein polynomials

$$\mathcal{B}^{d_1}_{(\lambda_0\lambda_1)} = \frac{d_1!}{\lambda_0!\lambda_1!}\beta_0^{\lambda_0}\beta_1^{\lambda_1} \tag{3-10}$$

By multiplying the Bernstein polynomials we arrive at a set of tensor-product basis polynomials. It can be shown that this basis preserves the partition of unity property [20] and many other stability related properties [21] and forms a basis for the tensor-product polynomial space [39].

There is no theoretical restriction to using multivariate instead of univariate Bernstein polynomials in equation (3-9). For example, Strøm derives the representation of such polynomial models with respect to a polynomial basis of total degree in a very general manor [39]. In

engineering applications however such an approach was first taken by Govindarajan et al. [22].

Coefficients and the control net

The coefficients of the bivariate tensor-product spline are often chosen to be vectors in $\mathbb{R}^3$. This results in a lot of design freedom, but is undesirable in scattered data approximation tasks [21]. Therefore we use the same scalar coefficients as in the simplex spline.

That the coefficients are defined as scalars does not mean that they have no spatial location, but rather that this position is fixed. Much in the same way as the simplex spline, the locations are purely defined by the corresponding multi-index λ [21]. Together the coefficients form a control net, that may be defined as a product of two univariate control point sets.

Continuity

Continuity can be defined between two neighboring tensor-product patches. In the majority of the literature only geometric continuity is considered.

Farin however gives the following condition for continuity between two bivariate tensor-product patches [20]: two patches are C^r across their boundary if and only if the vertices of the control net can be interpreted as polygons of C^r Bézier curves. In other words, if the coefficients around the boundary define continuous piecewise polynomials, then the join of surfaces is also smooth. This implies that one can reuse univariate continuity conditions and apply them perpendicular to the shared edge.

In his PhD dissertation, Lai takes it a step further and derives continuity conditions between both two- and three-dimensional tensor-product polynomials [31]. Also the continuity between different shapes, such as rectangle to triangle and tetrahedron to prism are considered. His approach however is to set the derivatives on both sides of the shared edge equal. This method is therefore hard to generalize to higher dimensions. On top of that the view on these conditions is distorted in the sense that their generalization is unclear.

The approximation problem

Scattered data can be fitted with tensor-product Bernstein polynomials in much the same way as with simplex splines. Pereyra and Scherer describe a method based on Singular Value Decomposition [35]. They also propose iterative and distributive methods to decrease calculation time. The main disadvantage of their work is that they do not include continuity conditions in their approximation scheme. By comparing the simplex spline framework and the tensor-product Bernstein polynomials however, there are clear indications that problem (3-7) can also be formulated for the latter.

Indeed this approach was taken by Sun et al. [40]. They used the method of Govindarajan et al. directly to estimate a model for the F-16 fighter aircraft. A disadvantage of this approach is the fact that only rectangular domains can be spanned.

3-2 Quadrotor analytic modeling

Verification of the ability of the multiplex spline to fit scattered data will be performed using, among others, a quadrotor data set. Rotor inflow will be modeled as a function of total velocity, angle of attack and engine state (power and rpm). In this section an overview is provided of the available analytic models for quadrotors. The focus lies on modeling phenomena that affect the thrust directly. Most of the quadrotor models are derived from helicopter aerodynamics, implying this field can also be investigated [4, 26].

In the models for thrust there is a clear subdivision in two types. The first type, discussed in subsection 3-2-1, is what we may call the dynamic model: a model in which rotor rpm is the main variable and all aerodynamic effects are collected in a thrust coefficient C_T [4]. The other type, discussed in subsection 3-2-2, is an aerodynamic model in which an attempt is made to model all aerodynamic effects [26]. Finally the well-known electric motor model is presented in subsection 3-2-3 and blade flapping is briefly mentioned in subsection 3-2-4.

3-2-1 Dynamics thrust models

In the first extensive discussion of quadrotor modeling and control, Bouabdallah proposes a flight dynamics model for the thrust as it is also found in helicopter theory. This expression for thrust is used often for modeling purposes [4, 26, 36].

$$T = C_T \rho A (\omega R)^2 \tag{3-11}$$

The air density ρ, rotor disc area A and the tip speed ωR are used as variables such that C_T is relatively constant in nominal flight. There exist analytic equations for this thrust coefficient in which shape parameters of the rotor take a central role [4].

One way of modeling quadrotor thrust would be to assume the model structure of equation (3-11). By estimation of the rotor rpm and direct measurement of the other parameters, a numeric model can be made of the thrust coefficient [16].

3-2-2 Aerodynamic thrust model

An analytic model that is deemed more interesting in the current research is the model for ideal thrust [26].

$$T = \frac{P}{v_\infty \sin\alpha + v_i} \tag{3-12}$$

This is the thrust that would be generated if the entire supplied power would be converted to aerodynamic force. The freestream velocity v_∞, angle of attack α and power P are the true variables in this equation. The induced velocity v_i can instead be described as a function of the same variables and itself [26].

$$v_i = \frac{v_h^2}{\sqrt{(v_\infty \cos\alpha)^2 + (v_\infty \sin\alpha + v_i)^2}} \tag{3-13}$$

Equation (3-13) is only valid in a limited vertical speed regime. When the quadrotor is descending with at most two times the hover induced speed v_h the rotor may enter a vortex ring state. This is an uncertain aerodynamic state in which air is recycled through the rotor. Although the vortex ring shows a periodic nature, it can be approximated using an empirical polynomial model from helicopter theory [26, 32].

$$\frac{v_i}{v_h} = k_0 + k_1\frac{v_z}{v_h} + k_2\left(\frac{v_z}{v_h}\right)^2 + k_3\left(\frac{v_z}{v_h}\right)^3 + k_4\left(\frac{v_z}{v_h}\right)^4 \tag{3-14}$$

The values of the coefficients are $k_0 = 1, k_1 = -1.125, k_2 = -1.372, k_3 = -1.718, k_4 = -0.665$. At higher descent rates the rotor enters a windmill brake state, in which equation (3-13) is again valid. In that state the rotor is driven mainly by the incoming air, but still produces lift.

The set of equations presented above shows that the thrust is an appropriate candidate for the validation of a system identification procedure. After all, equation (3-13) cannot be solved analytically and there are many cross-couplings between variables. The effects are rather well-known and sufficient dynamics can be included in a data set by only studying nominal flight.

More extensive rotor inflow models are available in helicopter literature. An overview is presented by Chen [11]. The use of these sophisticated models is not anticipated, because an attempt is made to keep the analytic simple. This will simplify the verification process for which the model is used.

3-2-3 Electric motor model

For the small DC motors installed on quadrotors only simple motor models are relevant. The standard is a second order differential system, which is often simplified to a first order system by assuming that inductance is low [4, 5, 6].

$$\dot{\omega} = -k_\omega\omega - \tau + k_U U \tag{3-15}$$

The coefficients k_ω and k_U related to rotor rpm and input voltage can be found in the thesis of Bouabdallah [4]. The motor load is indicated with τ and has a direct linear relation to the current I.

It is also possible to ignore all dynamic effects, which results in a static model [26]. It can be attained by setting $\dot{\omega} = 0$ in equation (3-15)

3-2-4 Blade flapping

The importance of blade flapping in quadrotor micro air vehicles depends on the maneuvers performed. In aggressive maneuvers blade flapping may have significant effect on the thrust produced [27]. In nominal flight however, this aerodynamic phenomenon is often neglected [4]. All flapping models for quadrotors are taken from helicopter literature.

3-3 DelFly numeric modeling

The validation of the system identification method is done using a set of flight test data of the DelFly. This process serves not only as a test for the multiplex spline, but also to investigate whether global system identification is possible based on the currently available data sets. This process thus requires knowledge of the system, as well as methods of analyzing high-dimensional data. Both these subjects are investigated in this section.

The state of the art of the research into the DelFly is presented in subsection 3-3-1. In subsection 3-3-2 a short overview is provided of the research into analyzing high-dimensional data.

3-3-1 DelFly modeling

In the validation process the flight test data of the DelFly II obtained by Caetano et al. is used [10]. The flight tests they performed are discussed in some detail in subsection 6-1-1. In this subsection the focus lies on the numeric models that were developed thus far. These models can be split in two groups. The first can best be described as linear flight dynamics models, whereas the latter are purely aerodynamic.

The flight dynamics models are a result of the same data set that is used in this thesis. From the initial data analysis it was already found that eigenmodes can be identified that resemble those of fixed-wing aircraft [10]. These modes were further investigated to find that the DelFly at least has a longitudinal, phugoid-like mode and a lateral mode [9]. In the process the states were reconstructed using standard system identification techniques. Finally a full, linear flight dynamics model was constructed and validated [8]. It is especially useful to note the model structure that was adopted. A simple model, using only the pitch angle and rate, the flap frequency and the elevator input to model the Z-force, reached a similar performance as a complete model in static tests. This shows that decoupling of the longitudinal behavior of the DelFly is valid under some circumstances.

Whereas flight dynamics models tend to use the flap average performance, aerodynamic models focus specifically on modeling the vortex states due to flapping. It was found that these vortices affect the overall performance significantly [15]. Especially the so called clap-and-peel mechanism is studied in detail. When the wing closes, the actual foil making up the wing is trailing behind. These parts of the wing only clap together when the leading edge is already open again. Thereby a chord location can be identified at which the wings touch, that runs down to the trailing edge as the leading edges move apart again. In other words, the wings clap and peel at the same time. This effects was found to positively affect thrust generation. New wing designs were incorporated by De Croon et al. to find a 5% increase in power-efficiency.

The results from the literature show that both the flap cycle and the maneuvers have significant effects on the forces on the DelFly. In flight dynamics models the effects of the flap cycle are often neglected, because they can be averaged over the relatively long specific time of the eigenmodes. Because the distribution of different variables over different layers is the great innovation presented by the multiplex spline, it is deemed interesting to generate a model that combines both maneuvers and the flapping cycle.

3-3-2 Interpreting high-dimensional data

To allow for system identification of the DelFly the analysis of the data is crucial in two ways. First of all a state vector should be deduced from the data. This amounts to assessing the relevance of different states in terms of predicting system outcome. On the other hand a high-dimensional tessellation will be constructed based on the projection of the data set onto lower-dimensional planes. Both these aspects are kept in mind when investigating the broad research field of big data.

It was discussed before that choosing layers may well be done based on an analytic model of the system under consideration. In some cases however such a model may not be available or sufficient and the data set itself should be used to make subsets of variables. From the review of dimensionality reduction and variable selection by Guyon and Elisseeff, two methods can be deduced [25]. The first is to use variable ranking methods to deduce the effect different individual variables have on the output. A ranking parameter they propose is the coefficient of determination. With such a ranking it may be possible to deduce what polynomial degree is required or how fine the triangulation should be to prevent overfitting.

On top of ranking individual variables, one can also perform variable subset selection by using wrappers or filters. The difference between the two is that wrappers use an induction algorithm (from state to output), whereas filters require only a data set [29]. This implies that the use of filters for layer selection is preferred. On the other hand, the brute force approach of fitting a spline using all possible subdivisions could be called a wrapper approach.

When the layers are chosen, a triangulation should be defined in each of them. This process can be simplified when insight is provided into the position of data points in the entire state space. In high-dimensional data sets it is impossible to plot all variables, and thus a method of dimensionality reduction can be helpful. These methods however generally construct a new, smaller set of variables (often referred to as features) to represent the most important directions [25]. Examples of such methods are principal component analysis and singular value decomposition. In system identification it is more important to keep known states in, such that the models can be interpreted more easily. In the case of the DelFly it may be useful to combine the flap states to come to a single flap phase.

Chapter 4

Multiplex spline theory

In this chapter the theoretical framework of the multiplex spline is presented. It will become clear that the multiplex spline bridges the gap between triangular and tensor-product Bernstein polynomials by presenting a generalization that describes both. On top of that, the generalization is extended into n dimensions. The chapter is written in such a form that it can be read separately from the rest of this report.

Before the remainder of this chapter is outlined, the notation for several important sets is introduced. The first is $\mathbb{Z}^n$, which signifies the non-negative integers up to n. Similarly, the positive integers up to n are indicated with $\mathbb{Z}_+^n$, such that $\mathbb{Z}^n = \{0, \mathbb{Z}_+^n\}$. For the set of n-dimensional barycentric coordinates we use the notation $\mathbb{B}^n = \{\boldsymbol{b} \in \mathbb{R}^{n+1} | \sum b_i = 1\}$.

For the definition of the multiplex and Bernstein basis polynomials a multi-index is used. In general a bold symbol is used for the complete set of valid permutations, whereas a normal symbol signifies a single permutation.

This chapter is subdivided into four sections. The first introduces the geometric basis: the multiplex. Multiple definitions are provided, as well as algorithms for construction and a discussion of the shared edge in tesselations. In section 4-2 the basis polynomials defined on a multiplex are discussed. This discussion leads to a description of the B-net in section 4-3. Finally in section 4-4 the different parts are combined to come to continuity conditions between multiplices, both equal and unequal in layer structure.

4-1 The multiplex

In this section the multiplex is described. Central is the introduction of two different definitions of this polytope, the bottom-up definition in subsection 4-1-1 and the top-down definition 4-1-2. The equivalence of these two definitions is shown in subsection 4-1-3. This property plays an essential role in the remainder of this chapter.

4-1-1 Bottom-up definition

The multiplex spline is a direct generalization of the tensor-product simplex spline introduced by Govindarajan et al. [22]. In this framework two layers of splines are used. In the first a multivariate simplex spline is used to describe the state space. The second layer contains a univariate simplex spline to describe the evolution of the B-coefficients of the first layer in time.

Although Govindarajan et al. indicated that this nesting of splines has a geometric effect, they do not provide a general description. In this section these geometric consequences are investigated, even for the more general case of including more than two layers of splines, and allowing all layers to be of arbitrary dimension. To properly do this, we define a layer to be a simplex instead of a complete spline.

Definition 1 (Layer)**.** *Consider an n-dimensional polytope Γ in $\mathbb{R}^n$ and a separate m-dimensional space $\mathcal{R}$ such that $\mathbb{R}^n \times \mathcal{R} = \mathbb{R}^{n+m}$. A* layer *is an m-simplex Δ in $\mathcal{R}$ positioned such that Γ and Δ share exactly one vertex.*

Note that the type of a layer is completely defined by its dimension. This means that if Γ consists entirely of layers, its shape can be defined using the dimensions of the separate layers. In fact, in that case Γ is the ν-multiplex, where $\nu = (\nu_1, ..., \nu_\ell)$, with ℓ the number of layers, is a multi-index of which each element indicates the dimension of a layer.

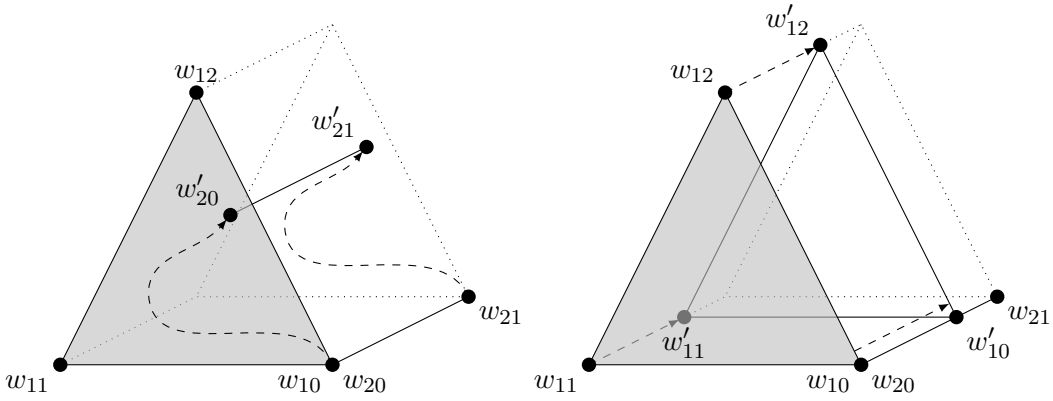
The definition of the ν-multiplex is much like an algorithm for construction. This algorithm is inspired by the definition of a parallelotope as presented by Kendall [28]. The starting point for the definition is the 0-multiplex, a point in zero dimensions.

Definition 2 (Bottom-up multiplex)**.** *Consider a ν-multiplex Γ of ℓ layers, with $|\nu| = n$, and a new m-dimensional layer Δ. Now consider the motion of the vertex w_0 of Δ such that it traces out Γ. The* (ν, m)-multiplex *is the polytope traced out by Δ as it moves along with w_0 in his motion while staying parallel to the original orientation of Δ.*

Before considering a set of examples, the notation of vertices in the layers is introduced. Since a vertex belongs to a certain layer, this should be reflected in the name of the vertex. Therefore we define vertices w_{ij}, where $i \in \mathbb{Z}_+^\ell$ is the index of the layer and $j \in \mathbb{Z}^{\nu_i}$ is the index within the layer. Some of the vertices of the multiplex will thus have multiple names, whereas others will not be labeled according to this definition.

The practical use of the above definition is best illustrated using a couple of examples. First the definition itself is clarified by showing two ways of constructing the (2,1)-multiplex. Example 2 focuses on the complete set of polytopes that can be constructed using definition 2.

Example 1 ((2,1)-Multiplex construction)**.** In this example we investigate the construction of the ν-multiplex, where $\nu = (2, 1)$. According to definition 2 this means we start with a 2-simplex with vertices $\{w_{10}, w_{11}, w_{12}\}$, then add the second, one-dimensional layer $\{w_{20}, w_{21}\}$. According to definition 2 the second layer should move along with w_{20} as it traces out the entire first layer. This is illustrated in Figure 4-1a as a random motion of the vertex w_{20}. The resulting multiplex is outlined using dotted lines.



(a) (2,1)-multiplex **(b)** (1,2)-multiplex

Figure 4-1: The construction of a (2,1)-multiplex according to definition 2. The order of the layers can be changed without affecting the shape of the multiplex. The orientation of the polytopes can be changed freely, and is only considered when defining continuity conditions.

A second, more intuitive way of constructing the same multiplex is by changing the order of the layers. This is shown in Figure 4-1b. The motion of the vertex w_{10} along the 1-simplex results in the same prism-shape.

By combining different layers, many different polytopes can be generated. This is illustrated using the following example.

Example 2 (Bottom-up multiplex)**.** As pointed out before, the 0-multiplex is a point. Let's look at our options to arrive at higher dimensions with this multiplex as a starting point. Note that the multiplices discussed are shown in Figure 4-2.

We start with the one-dimensional multiplex. There is only one way to arrive at a one-dimensional polytope, namely by using a single one-dimensional layer. The result is the 1-multiplex, which is equivalent to the 1-simplex: a line segment.

In two dimensions things become more interesting. There are two ways to arrive at two-dimensional polytopes using definition 2. The first is by using a single two-dimensional layer. This results in the (2,0)-multiplex: the 2-simplex. The 0 is added in the notation to keep the number of layers equal to the number of dimensions. The true significance of this 0 will become clear in the next subsection. The second way of arriving at a two-dimensional polytope is by combining two one-dimensional layers. This results in the (1,1)-multiplex: a parallelogram. In Figure 4-2 a square (or 2-cube) is drawn, which is a special case of the parallelogram.

In three dimensions one again has the option to use a single full-dimensional layer, which results in the 3-simplex. On the other end of the spectrum, three one-dimensional layers can be used to construct a (3-)cube. The third option is to use a two-dimensional and a one-dimensional layer. This results in the (2,1,0)-multiplex: a prism with parallel triangular sides.

If we look at even more dimensions, the number of different options continues to increase. Independent of the dimension we can always construct the n-simplex by using a single layer,

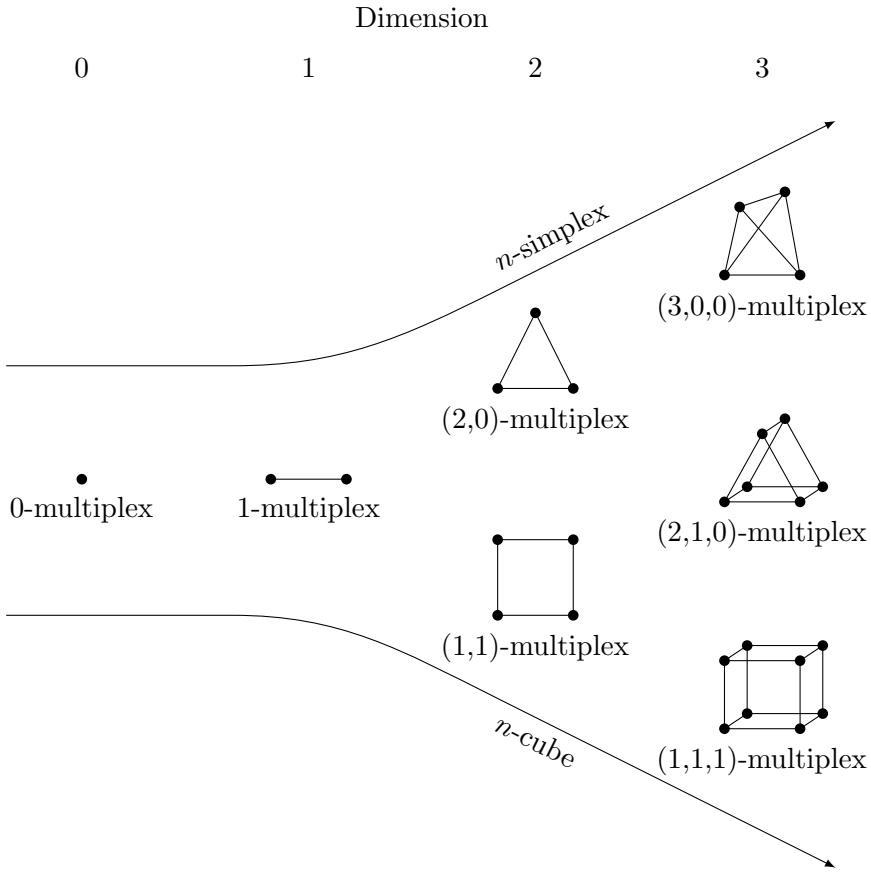


Figure 4-2: All multiplices for 0 up to 3 dimensions. Along the top, the n-simplices are lined up. They are formed using a single layer in definition 2. On the other end of the spectrum lie the n-cubes, that are formed using n iterations of definition 2 using only one-dimensional layers.

and the n-cube (or more generally the n-parallelotope) by using n one-dimensional layers. The exact amount of multiplices in n dimensions is given by the integer partition function $P(n)$, for which only generating functions and recursive formula's exist.

At this point it is useful to discuss why the multiplex is preferred over more general polytopes. First note that in an n-simplex one can uniquely describe a point as the weighted sum of the vertices of the simplex. The weights are called barycentric coordinates, denoted as $\boldsymbol{b} \in \mathbb{B}^n$.

Now note that a point in a ν-multiplex (with $|\nu| = n$) can be uniquely described by its projection onto each layer. After all, each layer lies in a different set of dimensions, such that together they span a region in $\mathbb{R}^n$. Because each layer is a simplex by definition 1, the projection of a point on a layer can be described using barycentric coordinates in that layer. For the multiplex this results in a vector of barycentric coordinates $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_\ell)$, with subvectors $\boldsymbol{\beta}_i \in \mathbb{B}^{\nu_i}, \forall i \in \mathbb{Z}_+^l$ and ℓ the number of layers. Each element of this vector is indicated as β_{ij} and refers to vertex w_{ij}, that is the j^{th} vertex in the i^{th} layer.

4-1-2 Top-down definition

To arrive at the second definition of the multiplex, we closely investigate the barycentric coordinates in the multiplex. As pointed out at the end of the previous subsection, any point in the multiplex can be defined using a vector of barycentric coordinates $\boldsymbol{\beta}$, with subsets $\boldsymbol{\beta}_i \in \mathbb{B}^{\nu_i}, i \in \mathbb{Z}_+^\ell$. This results in a total of $n+\ell$ barycentric coordinates. Apart from a unique description of a point in the multiplex, this vector also describes a point in the $(n+\ell-1)$-simplex.

Adjusting the multiplex barycentric coordinates $\boldsymbol{\beta}$ to simplex coordinates $\boldsymbol{b}$ amounts to a simple scaling operation. In the end we should have $\boldsymbol{b} \in \mathbb{B}^{n+\ell-1}$ which can be achieved by setting $\boldsymbol{b} = \frac{1}{\ell}\boldsymbol{\beta}$. This results in an extra set of constraints on $\boldsymbol{b}$, namely $\boldsymbol{b}_i \in \frac{1}{\ell}\mathbb{B}^{\nu_i}, \forall i \in \mathbb{Z}_+^\ell$. In these constraints $\boldsymbol{b}_i$ corresponds to the subset $\boldsymbol{\beta}_i$ in the multiplex coordinates.

In words this means that the layers can still be identified in the coordinates in the $(n+\ell-1)$-simplex. Now however these subsets corresponds to groups of vertices named layer sets.

Definition 3 (Layer set)**.** *Consider a simplex with vertices $\mathcal{V}$. The simplex is said to be subdivided in ℓ* layer sets *δ_i defined by sets of vertices $\mathcal{V}_i$ if $\mathcal{V}_i \subset \mathcal{V}, \forall i \in \mathbb{Z}_+^\ell$ and $\mathcal{V} = \bigcup_{i=1}^{\ell} \mathcal{V}_i$ but $\mathcal{V}_i \cap \mathcal{V}_j = \emptyset, \forall i \neq j \in \mathbb{Z}_+^\ell$.*

It is clear that the polytope described by $\boldsymbol{b}$ is convex, so defining it amounts to finding the extreme points. In order to comply with the conditions on $\boldsymbol{b}$ it is clear that in each subset $\boldsymbol{b}_i$ at least one element should be nonzero. If in each subset only one element is nonzero we arrive at the extreme points. Their locations in the simplex can be easily described by using the following concept.

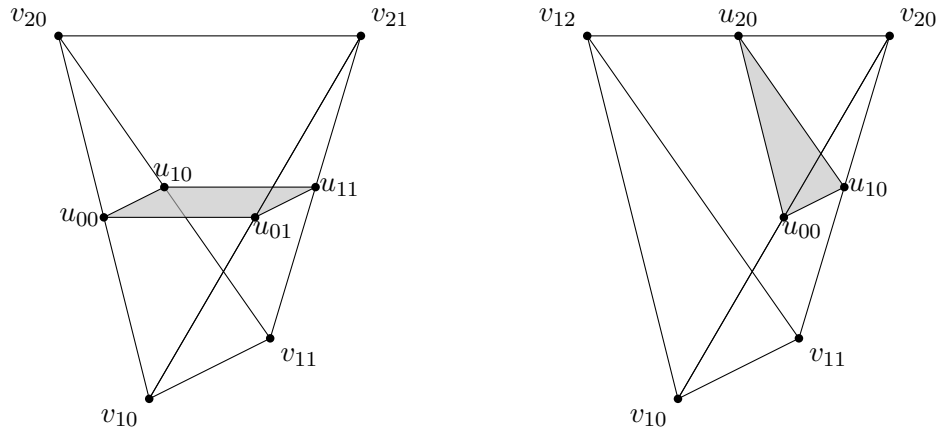
Definition 4 (Simplex link)**.** *Consider an $(n+\ell-1)$-simplex with vertices $\mathcal{V}$ subdivided in layer sets with vertices $\mathcal{V}_i = \{v_{i0}, ..., v_{i\nu_i}\}, i \in \mathbb{Z}_+^\ell$. A* simplex link *is an ℓ-simplex with vertices $\mathcal{W}_\phi = \{v_{1\phi_1}, ..., v_{\ell\phi_\ell}\}$ with ϕ a vector with $\phi_i \in \mathbb{Z}^{\nu_i}$, such that $\mathcal{V}_i \cap \mathcal{W}_\phi = \{v_{i\phi_i}\}, \forall i \in \mathbb{Z}_+^\ell$.*

In practical terms a simplex link is thus an ℓ-simplex spanned by a set of vertices obtained by taking one arbitrary vertex from each layer set. When comparing simplex links, it is often useful to say in what layer set they overlap. If two simplex links use the same vertex from a layer set, they are set to agree in that layer set. On the other hand, if they use different vertices from a layer set, they are said to disagree there.

By setting all but one element of each subset $\boldsymbol{b}_i$ zero, we arrive at the barycenter of a simplex link. This fact leads to a concise definition of the multiplex.

Definition 5 (Top-down multiplex)**.** *Consider an $(n+\ell-1)$-simplex Δ subdivided in ℓ layer sets δ_i defined by ν_i+1 vertices. The* ν-multiplex*, with $\nu = (\nu_1, ..., \nu_\ell)$ a multi-index, is the convex hull of the barycenters of all possible simplex links in Δ based on the subdivision in layer sets δ_i.*

A barycenter of a simplex link can be defined using the vector ϕ. Therefore the following notation is used for these points.



(a) (1,1)-multiplex **(b)** (2,0)-multiplex

Figure 4-3: Two 3-simplices $\{v_{10}, v_{11}, v_{20}, v_{21}\}$ and $\{v_{10}, v_{11}, v_{12}, v_{20}\}$ with inscribed multiplices. On the left the subdivision in layer sets is such that a (1,1)-multiplex is defined. On the right one vertex is moved from the second to the first layer set, which results in the (2,0)-multiplex. Both multiplices are colored transparent gray.

$$u_\phi = \frac{1}{\ell} \sum_{i=1}^{\ell} v_{i\phi_i} \tag{4-1}$$

To clarify the process of constructing the multiplex using the top-down definition, let's look at an example.

Example 3 (Top-down multiplex)**.** In this example we construct the two two-dimensional multiplices presented in example 2 from the 3-simplex. The result is displayed in Figure 4-3.

We start from the 3-simplex defined by the set of vertices $\mathcal{V} = \{v_{10}, v_{11}, v_{20}, v_{21}\}$ as displayed in Figure 4-3a. To construct the (1,1)-multiplex, we choose two layer sets δ_1 and δ_2 with vertices $\mathcal{V}_1 = \{v_{10}, v_{11}\}$ and $\mathcal{V}_2 = \{v_{20}, v_{21}\}$. The four $(\ell - 1)$-dimensional simplex links that can be constructed between δ_1 and δ_2 are the line segments described by $\{v_{10}, v_{20}\}$, $\{v_{10}, v_{21}\}$, $\{v_{11}, v_{20}\}$ and $\{v_{11}, v_{21}\}$. The barycenters of these simplex links are indicated as u_{00} up to u_{11}, using expression (4-1). Together these four points span the (1,1)-multiplex.

To construct the (2,0)-multiplex using the top-down definition we can reuse the 3-simplex of the previous example. The vertices are renamed for clarity. Now we choose one two-dimensional layer set and one zero-dimensional one: $\{v_{10}, v_{11}, v_{12}\}$ and $\{v_{20}\}$ as shown in Figure 4-3b. This choice only results in three simplex links, being every combination of a vertex from the first layer set with v_{20}. This results in one new barycenter, on the line $\{v_{12}, v_{20}\}$, indicated with u_{20}. Again we see that the barycenters span the multiplex. In this case we find the (2,0)-multiplex as the convex hull of $\{u_{00}, u_{10}, u_{20}\}$.

Note that this example shows why it can be useful to add zero-dimensional layer sets. Although a triangle can also be obtained by defining one layer set in a 2-simplex, now the (1,1)- and the (2,0)-multiplex are both inscribed in a 3-simplex. This will prove to be important when defining continuity between multiplices of different types in section 4-4.

4-1-3 Multiplex definition equivalence

In the previous subsections two definitions for the multiplex were presented. Both have their own use in the remainder of this chapter, so it is essential to show that the two definitions are equivalent. This is done in this subsection. The proof is rather long and is therefore split up into several parts and lemmas.

We start this subsection with a general lemma that will be used extensively. It was pointed out before that the multiplex layers and the simplex layer sets are related in many ways. The most important relation is the following.

Lemma 1 (Parallelism layers and layer sets)**.** *Let Δ be an $(n+\ell-1)$-simplex with layer sets $\delta_i, i \in \mathbb{Z}_+^\ell$. Consider the ν-multiplex Γ inscribed in Δ with $|\nu| = n$. A polytope $\gamma \subseteq \Gamma$ that is formed according to definition 5 using all simplex links that disagree in one and the same layer set δ_i, forms a parallel copy of that layer set.*

Proof. Assume that a layer set δ_i is defined by vertices $\mathcal{V}_i = \{v_{i0}, ..., v_{i\nu_i}\}$. Without loss of generality we can consider the simplex links that disagree in this layer set, and in the other layer sets use $v_{k0}, \forall k \neq i$. That is, in equation (4-1) we use only $\phi = \chi^i$ where $\chi_i^i = j$ and all other elements are zero.

$$u_{\chi^i} = \frac{1}{\ell}\left(v_{ij} + \sum_{k\neq i} v_{k0}\right), \qquad i, k \in \mathbb{Z}_+^\ell, \forall j \in \mathbb{Z}^{\nu_i}$$

Since the sum in the above expression is independent of χ_i^i, it is equal for all simplex links that disagree only in the i^{th} layer. Collecting all u_{χ^i} with $\chi_i^i = j$ and $\chi_k^i = 0, \forall k \neq i$ we find the same set of vertices $\mathcal{V}_i$, yet scaled with $\frac{1}{\ell}$ and translated along the vector $\frac{1}{\ell}\sum_{k\neq i} v_{k0}$. □

An example of this parallelism can be found in Figure 4-3a. In the multiplex $\{u_{10}, u_{11}\}$ is for example parallel to the layer set $\{v_{20}, v_{21}\}$.

The second building block that is required for the proof is a description of a point in the multiplex, according to the bottom-up definition. As pointed out earlier, the idea is to describe the projection of a point onto each layer using barycentric coordinates in each layer. These descriptions can be combined to find the following.

Lemma 2 (Point in bottom-up multiplex)**.** *Consider a ν-multiplex Γ consisting of ℓ layers γ_1 up to γ_ℓ. If each layer is defined using a set of vertices $\mathcal{W}_i = \{w_{i0}, ..., w_{i\nu_i}\}$, a general point $\boldsymbol{x} \in \Gamma$ can be defined as*

$$\boldsymbol{x} = (1-\ell)w_0 + \sum_{i=1}^{\ell}\sum_{j=0}^{\nu_i} \beta_{ij} w_{ij} \tag{4-2}$$

where β_{ij} are the barycentric coordinates in the i^{th} layer and w_0 the origin point such that $w_{i0} = w_0, \forall i \in Zp\ell$.

Proof. Using the vertices of the i^{th} layer the projection of $\boldsymbol{x}$ onto this layer can be defined as

$$\boldsymbol{x}_i = \sum_{j=0}^{\nu_i} \beta_{ij} w_{ij}$$

Now we choose an origin point w_0 that all layers share. We can define any point by adding to this origin point the vectors from w_0 to the projections $\boldsymbol{x}_i$.

$$\boldsymbol{x} = w_0 + \sum_{i=1}^{\ell} (\boldsymbol{x}_i - w_0)$$

By combining the above two expressions and collecting terms for w_0 we arrive at equation (4-2). □

Note that equation (4-2) provides us with a description of the extreme points of the multiplex. They are found when in each layer $\boldsymbol{x}_i$ corresponds to a vertex in that layer. This results in the expression

$$u_\phi = (1-\ell) w_0 + \sum_{i=1}^{\ell} w_{i\phi_i} \tag{4-3}$$

The above lemmas provide the tools to show that a multiplex defined using definition 2 has an equivalent multiplex according to definition 5.

Lemma 3 (Bottom-up to top-down equivalence)**.** *Consider a ν-multiplex Γ with ℓ layers γ_1 up to γ_ℓ and $|\nu| = n$. A set of $n+\ell-1$ vertices $\mathcal{V}$ can be defined such that they form an $(n+\ell-1)$-simplex Δ with layer sets v_i such that the inscribed multiplex in Δ is equal to Γ.*

Proof. We can invert lemma 1 to read that layer sets in Δ should be scaled (with ℓ) parallel copies of the layers of Γ. This can be achieved by choosing vertices v_{ij} in the i^{th} layer set of Δ as follows.

$$v_{ij} = \ell \begin{bmatrix} w_{ij} \\ g_{ij} \end{bmatrix} + (1-\ell) \begin{bmatrix} w_0 \\ \mathbf{0}_{\ell-1} \end{bmatrix}$$

where $g_{ij} \in \mathbb{R}^{\ell-1}$ completes the vector such that $v_{ij} \in \mathbb{R}^{n+\ell-1}$. Note that $\mathbf{0}_n$ indicates a vector of n zeros.

Filling in the above choice for v_{ij} in definition (4-1) of u_ϕ with $j = \phi_i$ we find

$$\begin{aligned} u_\phi &= \frac{1}{\ell} \sum_{i=1}^{\ell} v_{i\phi_i} \\ &= (1-\ell) \begin{bmatrix} w_0 \\ \mathbf{0}_{\ell-1} \end{bmatrix} + \sum_{i=1}^{\ell} \begin{bmatrix} w_{i\phi_i} \\ g_{i\phi_i} \end{bmatrix} \end{aligned}$$

which is immediately equivalent to equation (4-3) if $\sum g_{i\phi_i} = \mathbf{0}_{\ell-1}$. This introduces $(n+1)(\ell-1)$ constraints on $(n+\ell)(\ell-1)$ degrees of freedom, which always has infinitely many solutions if we count the zero dimensional layers (that is $\ell = n$). □

One way of constructing a circumscribed simplex is as follows. First we set $g_{i\phi_i} = e_i, \forall\phi_i, \forall i \in \mathbb{Z}_+^{\ell-1}$, with e_i the i^{th} unit vector. Then in the final ℓ^{th} layer we set $g_{\ell\phi_\ell} = \mathbf{-1}$. This effectively amounts to pushing each layer into its own extra dimension, then pushing the last one in the other direction in all extra dimensions.

The final step towards full equivalence of the multiplex definitions is proving that the multiplex inscribed in a simplex can also be described using a set of layers.

Lemma 4 (Top-down to bottom-up equivalence)**.** *Let Δ be an $(n+\ell-1)$-simplex with vertices $\mathcal{V}$ and ℓ layer sets δ_i defined. In the inscribed ν-multiplex Γ, where $|\nu| = n$, ℓ layers can be identified such that they define the same multiplex when used in definition 2.*

Proof. First it is noted that every u_ϕ is a vertex of one copy of each layer set. That is, we can choose any u_ϕ as origin point.

From lemma 1 we have that the multiplex consists of parallel copies of layer sets. We choose one copy of each layer set as our candidate layers. To simplify the notation we choose those copies of which u_ω, with $\omega = (0, ..., 0)$ is a vertex. The other vertices of the candidate layers are then defined by equation (4-1) where $\phi = \chi^i$, with χ_i^i the only nonzero element.

Now consider the top-down definition of points in the multiplex

$$\boldsymbol{x} = \sum_{i=1}^{\ell} \sum_{j=0}^{\nu_i} b_{ij} v_{ij} \tag{4-4}$$

with constraints $\boldsymbol{b}_i \in \frac{1}{\ell}\mathbb{B}^{\nu_i}$. The above can be rewritten by expanding v_{ij} as

$$v_{ij} = \sum_{k=1}^{\ell} (v_{k\chi_k^i} - v_{k0}) + v_{i0}$$

where $\chi_i^i = j$ and all other elements of χ^i are zero, as prescribed by the definition of our candidate layers. Filling this into equation (4-4) and taking out a factor ℓ we find

$$\boldsymbol{x} = \sum_{i=1}^{\ell} \sum_{j=0}^{\nu_i} \ell b_{ij} \left(\frac{1}{\ell} \sum_{k=1}^{\ell} (v_{k\chi_k^i} - v_{k0}) + \frac{1}{\ell} v_{i0} \right)$$

Combining the above with equation (4-1) this reduces to

$$\boldsymbol{x} = \sum_{i=1}^{\ell} \sum_{j=0}^{\nu_i} \ell b_{ij} \left(u_{\chi^i} - u_\omega + \frac{1}{\ell} v_{i0} \right)$$

Because u_ω is independent of i, and using the definition of the barycentric coordinates, we can rewrite this to

$$\begin{aligned}\boldsymbol{x} &= \sum_{i=1}^{\ell}\sum_{j=0}^{\nu_i} \ell b_{ij} u_{\chi^i} - \left(\sum_{i=1}^{\ell}\sum_{j=0}^{\nu_i} \ell b_{ij}\right) u_\omega + \sum_{i=1}^{\ell} v_{i0} \left(\sum_{j=0}^{\nu_i} b_{ij}\right) \\ &= \sum_{i=1}^{\ell}\sum_{j=0}^{\nu_i} \ell b_{ij} u_{\chi^i} - \ell u_\omega + u_\omega\end{aligned}$$

Finally, by collecting the terms related to u_ω, setting $\beta_{ij} = \ell b_{ij}$ and choosing $u_\omega = w_0$ and $u_{\chi^i}|_{\chi_i^i=j} = w_{ij}$ we find equation (4-2). That is, by choosing u_ω as the origin point and the connected copies of layer sets as layers, we can describe all points in Γ using the bottom-up definition. □

Now we finally arrive at the goal of this section.

Theorem 1 (Multiplex equivalence)**.** *The bottom-up definition 2 and the top-down definition 5 of the multiplex are equivalent.*

Proof. From lemma 3 and lemma 4 we have that any multiplex defined using one of the definitions can also be defined using the other. □

4-2 Basis polynomials

In this section the Bernstein basis polynomials of the multiplex spline are defined. It will become clear that the set of polynomials can be either defined using a tensor-product of lower dimensional basis polynomials or as a subset of a higher dimensional simplex spline. The spatial location of the polynomials with respect to the multiplex is discussed in more detail in section 4-3.

The remainder of this section is structured as follows. In subsection 4-2-1 the tensor product definition of the basis polynomials is presented. Then in subsection 4-2-2 it is shown that these basis polynomials can also be found in a higher dimensional, higher degree simplex spline. Finally the most important properties of the polynomials are collected in subsection 4-2-3.

4-2-1 Tensor-product basis polynomials

The first definition of the Bernstein basis polynomials follows directly from the definition of the tensor-product simplex spline. Govindarajan introduced this spline as a multivariate simplex spline of which the B-coefficients are described using a univariate simplex spline [22]. Generalizing this concept to multiple multivariate layers we again arrive at a tensor-product definition.

Definition 6 (Tensor-product basis polynomials)**.** *Consider a ν-multiplex as defined in definition 2 or 5 and write the set of basis polynomials in the i^{th} layer of dimension ν_i and degree d_i as*

$$\begin{aligned}\boldsymbol{B}_{\nu_i}^{d_i} &= \left\{ B_{\lambda_i}^{d_i} \right\}_{|\lambda_i|=d_i} \\ &:= \left\{ \frac{d_i!}{\lambda_i!} \boldsymbol{\beta}_i^{\lambda_i} \right\}_{|\lambda_i|=d_i}\end{aligned} \tag{4-5}$$

where $\lambda_i = (\lambda_{i0}, ..., \lambda_{i\nu_i})$ is a multi-index used in the multinomial definitions. The set of multiplex Bernstein basis polynomials *$\boldsymbol{\mathcal{B}}_{\nu}^{d}$ is defined as*

$$\begin{aligned}\boldsymbol{\mathcal{B}}_{\nu}^{d} &= \left\{ \mathcal{B}_{\lambda}^{d} \right\}_{|\lambda_i|=d_i, \forall i \in \mathbb{Z}_+^{\ell}} \\ &:= \boldsymbol{B}_{\nu_1}^{d_1} \otimes ... \otimes \boldsymbol{B}_{\nu_\ell}^{d_\ell}\end{aligned} \tag{4-6}$$

with $d = (d_1, ..., d_\ell)$ a multi-index containing the degrees and $\lambda = (\lambda_1, ..., \lambda_\ell)$ the collection of multi-indices in the layers.

Taking a single element from the set $\boldsymbol{\mathcal{B}}_{\nu}^{d}$ we find

$$\mathcal{B}_{\lambda}^{d} := \prod_{i=1}^{\ell} B_{\lambda_i}^{d_i} = \frac{d!}{\lambda!} \boldsymbol{\beta}^{\lambda} \tag{4-7}$$

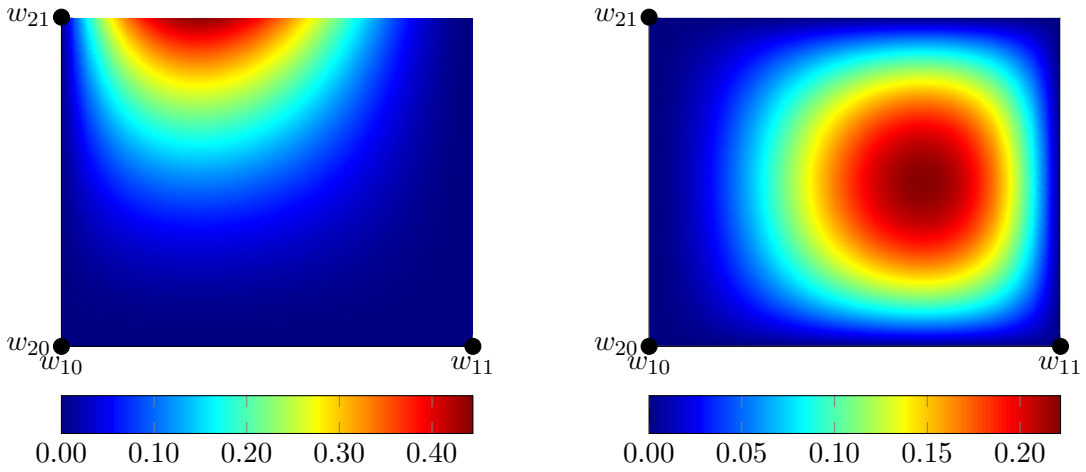
In the case of a simplex spline $\boldsymbol{B}_{\nu_i}^{d_i}$ contains all possible permutations of the multi-index λ_i such that $|\lambda_i| = d_i$. In the multiplex spline these conditions on λ_i should still hold, yielding a set of basis polynomials where not all permutations of $\lambda : |\lambda| = |d|$ are included. More precisely

$$\boldsymbol{\lambda} := \boldsymbol{\lambda}_1 \times ... \times \boldsymbol{\lambda}_\ell \tag{4-8}$$

Or in words, the set of multi-index permutations occurring in $\boldsymbol{\lambda}$ is defined as the Cartesian product of the sets $\boldsymbol{\lambda}_i$ in the layers.

Example 4 (Tensor-product basis polynomials)**.** In this example we consider the (1,1)-multiplex spline (or tensor-product Bézier patch). We choose polynomial degrees of 3 and 2 in the first and second layer respectively (that is, $d = (3, 2)$). Since both splines are univariate, we find all permutations of the multi-indices to be

$$\begin{aligned}\boldsymbol{\lambda}_1 &= \{(3,0), (2,1), (1,2), (0,3)\} \\ \boldsymbol{\lambda}_2 &= \{(2,0), (1,1), (0,2)\}\end{aligned}$$



(a) $\mathcal{B}^{3,2}_{2,1,0,2}(\boldsymbol{\beta}) = 3\beta_{10}^2\beta_{11}\beta_{21}^2$ **(b)** $\mathcal{B}^{3,2}_{1,2,1,1}(\boldsymbol{\beta}) = 6\beta_{10}\beta_{11}^2\beta_{20}\beta_{21}$

Figure 4-4: Two example basis polynomials of the (1,1)-multiplex spline with degree $d = (3, 2)$. Note the two different color scales used.

The Cartesian product in (4-8) implies that any combination of two entries from $\boldsymbol{\lambda}_1$ and $\boldsymbol{\lambda}_2$ occurs in $\boldsymbol{\lambda}$. One entry may for example be $\lambda = ((2, 1), (0, 2)) = (2, 1, 0, 2)$, if the second and third entry of the two sets are combined. This leads to the basis polynomial

$$\begin{aligned}\mathcal{B}^{(3,2)}_{(2,1,0,2)}(\boldsymbol{\beta}) &:= B^3_{(2,1)}(\boldsymbol{\beta}_1)B^2_{(0,2)}(\boldsymbol{\beta}_2)\\ &= \frac{3!}{2!1!}\beta_{10}^2\beta_{11}^1\frac{2!}{0!2!}\beta_{20}^0\beta_{21}^2 = 3\beta_{10}^2\beta_{11}\beta_{21}^2\end{aligned}$$

Together with the above example, another basis polynomials of the same degrees is plotted in Figure 4-4.

4-2-2 A subset of simplex spline basis polynomials

In subsection 4-1-2 it was shown that the barycentric coordinates $\boldsymbol{\beta}$ in the multiplex also describe locations in a higher-dimensional simplex. A similar equivalence can be found in the basis polynomials.

This equivalence is best made clear using the multi-index λ. It was noted before that it consists of several parts that stem from the different layers in the multiplex. This means that, like with the barycentric coordinates, there are $n + \ell$ entries, where a multivariate simplex spline needs only $n + 1$. On top of that, if the separate constraints on the subsets of λ are taken together, we find $|\lambda| = |d|$. This implies that $\boldsymbol{\lambda} \subseteq \boldsymbol{\kappa}$ with $\boldsymbol{\kappa}$ the multi-index permutations of an $(n + \ell - 1)$-simplex spline of degree $|d|$. Taking together the equivalence in barycentric coordinates and in multi-index, the concept of polynomial equivalence can be derived.

Proposition 1 (Polynomial equivalence)**.** *Let Δ be an $(n+\ell-1)$-simplex and Γ a ν-multiplex inscribed in Δ. Consider the sets of Bernstein basis polynomials $\boldsymbol{\mathcal{B}}^d_\nu$ defined on Γ and $\boldsymbol{B}^{|d|}_{n+\ell-1}$ defined on Δ. $\boldsymbol{\mathcal{B}}^d_\nu$ forms a scaled subset of $\boldsymbol{B}^{|d|}_{n+\ell-1}$, or*

$$\boldsymbol{\mathcal{B}}_{\nu}^{d}(\boldsymbol{\beta}) \subseteq \left(\ell^{|d|}\frac{d!}{|d|!}\right)\boldsymbol{B}_{n+\ell-1}^{|d|}(\boldsymbol{b}) \tag{4-9}$$

where the scaling factor is only determined by the degree d and the number of layers ℓ.

Proof. We take one general element $\mathcal{B}_{\lambda}^{d}(\boldsymbol{\beta}) \in \boldsymbol{\mathcal{B}}_{\nu}^{d}$ and the element $B_{\lambda}^{|d|}(\boldsymbol{b}) \in \boldsymbol{B}_{n+\ell-1}^{|d|}$. Dividing the two and filling in the barycentric coordinate scaling $\boldsymbol{b} = \frac{1}{\ell}\boldsymbol{\beta}$, we obtain

$$\frac{\mathcal{B}_{\lambda}^{d}(\boldsymbol{\beta})}{B_{\lambda}^{|d|}(\boldsymbol{b})} = \frac{\frac{d!}{\lambda!}\boldsymbol{\beta}^{\lambda}}{\frac{|d|!}{\lambda!}\left(\frac{1}{\ell}\boldsymbol{\beta}\right)^{\lambda}} = \ell^{|d|}\frac{d!}{|d|!}$$

Since the above holds for any $\lambda \in \boldsymbol{\lambda}$ and $\lambda \subseteq \kappa$, we arrive at equation (4-9). □

The equivalence of the multiplex and simplex basis polynomials leads to a second definition of the basis polynomials.

Definition 7 (Subset basis polynomials)**.** *Consider an $(n+\ell-1)$-simplex spline with basis polynomials $\boldsymbol{B}_{n+l-1}^{|d|}$ and a ν-multiplex Γ according to definition 5. The* multiplex Bernstein basis polynomials *$\boldsymbol{\mathcal{B}}_{\nu}^{d}$ defined on Γ form the scaled subset*

$$\boldsymbol{\mathcal{B}}_{\nu}^{d} := \left\{\mathcal{B}_{\lambda}^{d}(\boldsymbol{\beta}) = \left(\ell^{|d|}\frac{d!}{|d|!}\right)B_{\lambda}^{|d|}(\boldsymbol{b}) \in \left(\ell^{|d|}\frac{d!}{|d|!}\right)\boldsymbol{B}_{n+l-1}^{|d|} : |\lambda_i| = d_i, \forall i \in \mathbb{Z}_{+}^{\ell}\right\} \tag{4-10}$$

of the set of simplex basis polynomials $\boldsymbol{B}_{n+l-1}^{|d|}$.

Definition 7 is a direct result of rephrasing proposition 1.

4-2-3 Properties of basis polynomials

In this subsection both definitions of the basis polynomials in the multiplex spline are used to find their most important properties. Based on the collection of propositions in this subsection it can be shown that the basis polynomials form a stable local basis [30].

In many proofs of stability of a polynomial basis, the partition of unity property and non-negativity of the basis play a central role.

Proposition 2 (Partition of unity)**.** *The set of basis polynomials $\boldsymbol{\mathcal{B}}_{\nu}^{d}$, with ν and d the dimensions and degrees of the layers respectively, has the partition of unity property and all elements are non-negative.*

Proof. From among others Lai and Schumaker we have that the Bernstein basis polynomials of a simplex spline are non-negative and have the partition of unity property [30]. Because the basis polynomials of the multiplex spline are a tensor-product of simplex spline polynomials (equation (4-6)), we have for the sum of the basis polynomials

$$\sum_{\lambda \in \boldsymbol{\lambda}} \prod_{i=1}^{\ell} B_{\lambda_i}^{d_i} = \prod_{i=1}^{\ell} \sum_{\lambda \in \boldsymbol{\lambda}} B_{\lambda_i}^{d_i} = 1$$

Note that the order of summation and multiplication can be inverted as the product does not depend on λ.

Also, if we fill in $B_{\lambda_i}^{d_i} \geq 0$ in equation (4-6), we find that the same holds for $\mathcal{B}_{\lambda}^{d}$. Therefore both the partition of unity and the non-negativity properties are preserved. □

A remaining question is what the Bernstein polynomials form a basis of. This can be deduced from the definition of the basis polynomials in equation 4-6.

Proposition 3 (Basis for polynomial space)**.** *The set of basis polynomials $\boldsymbol{\mathcal{B}}_{\nu}^{d}$ of degree d and layer dimensions ν forms a basis for the Cartesian product of the layer polynomial spaces $\mathcal{S}_{\nu}^{d} = \mathcal{S}_{\nu_1}^{d_1} \times ... \times \mathcal{S}_{\nu_\ell}^{d_\ell}$.*

Proof. From among others Lai and Schumaker we have that the simplex spline basis polynomials $\boldsymbol{B}_{\nu_i}^{d_i}$ form a basis for the polynomial space $\mathcal{S}_{\nu_i}^{d_i}$ [30]. The tensor-product of the bases, as described in definition 6, trivially results in a basis for the Cartesian product of the individual spaces. □

The implications of proposition 3 are not investigated further for the time being. It is accepted that since the Bernstein polynomials form a basis in each individual layer, the approximation power will be satisfactory. On top of that, it is expected that the approximation power in different layers will not affect each other.

A final note is placed here to couple the basis polynomials to the B-net, discussed in the next section. In the simplex spline framework the B-coefficients lie at control points, that align with the locations of maxima of basis polynomials. Therefore it is of interest to find the locations of these maxima.

Proposition 4 (Maximum value)**.** *The maximum of a basis polynomial $\mathcal{B}_{\lambda}^{d}(\boldsymbol{\beta}) = \prod_{i=1}^{\ell} B_{\lambda_i}^{d_i}(\boldsymbol{\beta}_i)$ with multi-index $\lambda = (\lambda_1, ..., \lambda_\ell)$ and degrees $d = (d_1, ..., d_\ell)$ lies at the location where the vector $\boldsymbol{\beta} = (\boldsymbol{\beta}_1, ..., \boldsymbol{\beta}_\ell)$ results in a maximum in each basis polynomial $B_{\lambda_i}^{d_i}(\boldsymbol{\beta}_i)$.*

Proof. From definition 1 of a layer it is clear that barycentric coordinates in each layer are independent from each other. Therefore all maxima of the basis polynomials in definition 6 can be achieved independently, resulting in the highest possible function value for $\mathcal{B}_{\lambda}^{d}(\boldsymbol{\beta})$. □

4-3 B-net

The final element of the multiplex spline that should be discussed is the B-net. This is the collection of B-coefficients and their location in the multiplex. The next section, dealing with continuity, strongly relies on this section. The B-coefficients are treated as a set rather than

as a vector, with the notation $c_\lambda \in \boldsymbol{\mathcal{C}}$ for the multiplex spline coefficients and $c_\kappa \in \boldsymbol{C}$ for the simplex spline.

First the B-net is described as a combination of B-nets of the separate layers. The relation to the domain points and the maxima of the basis polynomials is covered. In subsection 4-3-2 the B-net is instead described as a subset of a higher dimensional simplex spline.

4-3-1 Bottom-up B-net

In the simplex spline the B-coefficients lie at so called domain points. Each B-coefficient corresponds to a basis polynomial, and thus to a single multi-index $\kappa = (\kappa_0, ..., \kappa_n)$. Given this multi-index, a domain point q is defined by Lai and Schumaker as [30]

$$q_\kappa := \frac{1}{e} \sum_{t=0}^{n} \kappa_t v_t \tag{4-11}$$

where v_t represents a vertex of the simplex, n is the dimension of the spline and e is the degree of the simplex spline.

From definition (4-8) of the permutations of $\lambda = (\lambda_1, ..., \lambda_\ell), \lambda_i = (\lambda_{i0}, ..., \lambda_{i\nu_i})$ we find that the domain points in the multiplex spline lie at locations described by the Cartesian product of all domain points in equation (4-11). That is, any combination of domain points from each layer will be a domain point, or

$$q_\lambda := (1 - \ell)w_0 + \sum_{i=1}^{\ell} \frac{1}{d_i} \sum_{j=0}^{\nu_i} \lambda_{ij} w_{ij} \tag{4-12}$$

with $w_0 := w_{i0}, \forall i \in \mathbb{Z}_+^\ell$ the origin point. Note that the equation for domain points resembles equation (4-2).

In subsection 4-2-3 it was found that the maximum of a basis polynomial lies at the place where all elements in the defining product (4-6) have a maximum. This combination is equivalent to the one defining the domain points. That is, the B-coefficients again lie at the maximum of the corresponding basis polynomial.

4-3-2 Top-down B-net

Like the basis polynomials, the B-coefficients directly relate to a multi-index. Therefore the reasoning of subsection 4-2-2 also implies that the B-coefficients of the multiplex spline form a subset of the B-coefficients of a higher dimensional simplex spline. To be more precise, the multiplex spline inherits only those coefficients corresponding to permutations of the multi-index κ for which $\kappa \in \boldsymbol{\lambda}$.

The location of the B-coefficients $\boldsymbol{\mathcal{C}}$ in the higher dimensional B-net $\boldsymbol{C}$ can be found by filling in the relevant permutations κ in equation (4-11), with $e = |d|$. To describe these locations we introduce the biased multiplex.

Definition 8 (Biased multiplex)**.** *Consider an $(n+\ell-1)$-simplex Δ with ℓ layer sets defined consisting of ν_i+1 vertices. The* biased ν-multiplex $\Gamma^{\boldsymbol{\xi}}$*, with $\boldsymbol{\xi}$ a vector of ℓ weights such that $\boldsymbol{\xi} \in \mathbb{B}^\ell$, is the convex hull of the points*

$$u_\phi = \sum_{i=1}^{\ell} \xi_i v_{i\phi_i} \tag{4-13}$$

on the simplex links of Δ, where ϕ is a vector that indicates what vertex is taken from each layer.

The name of the biased multiplex can be easily explained. The shape is equal to the multiplex defined in definition 5, but it is shifted towards one or more of the layer sets.

Using the biased multiplex, the B-net $\boldsymbol{\mathcal{C}}$ of the multiplex can be described as a subset of $\boldsymbol{C}$ as follows.

Proposition 5 (Multiplex B-net)**.** *Consider an $(n+\ell-1)$-simplex Δ with ℓ layer sets and an inscribed biased ν-multiplex $\Gamma^{\boldsymbol{\xi}}$ with $\boldsymbol{\xi} = (\frac{d_1}{|d|}, ..., \frac{d_\ell}{|d|})$. The B-net $\boldsymbol{\mathcal{C}}$ used in the multiplex spline is found at domain points in Δ that lie in $\Gamma^{\boldsymbol{\xi}}$.*

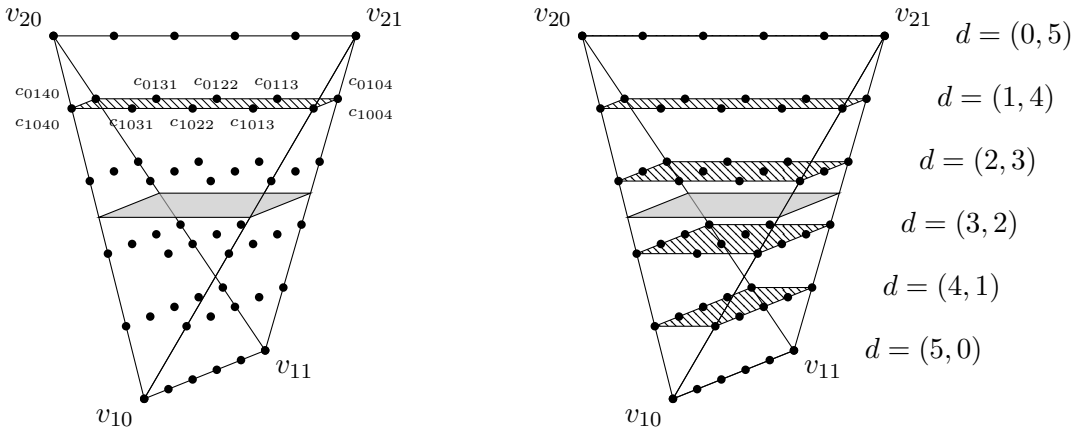
Proof. We replace κ in equation (4-11) with $\lambda = (\lambda_1, ..., \lambda_\ell)$ and the degree by the total degree $|d|$ to find

$$q = \frac{1}{|d|} \sum_{i=1}^{\ell} \sum_{j=0}^{\nu_i} \lambda_{ij} v_{ij}$$

where v_{ij} is a renaming of the vertices v_t, rather than a change of basis. By taking $\frac{1}{|d|}$ into the innermost sum we find sets of barycentric coordinates $\boldsymbol{\beta}_i = (\frac{\lambda_{i0}}{|d|}, ..., \frac{\lambda_{i\nu_i}}{|d|})$. Since $|\lambda_i| = d_i$ we find that $\boldsymbol{\beta}_i \in \frac{d_i}{|d|}\mathbb{B}^{\nu_i}$ in all layers and for any valid λ. This is equivalent to lying in the convex hull of extreme points for which $\boldsymbol{\beta}_i = (0, ...0, \frac{d_i}{|d|}, 0, ..., 0)$ in each layer, which immediately reduces to equation (4-13) with weights $\xi_i = \frac{d_i}{|d|}$. □

To illustrate the relation between the biased multiplex and the B-net, an example is provided for the (1,1)-multiplex.

Example 5 (B-net and the biased multiplex)**.** In Figure 4-5a the B-net of a 3-simplex $\{v_{10}, v_{11}, v_{20}, v_{21}\}$ is displayed as a set of black circles. The set of B-coefficients corresponding to the multi-index $\kappa = (\kappa_1, \kappa_2)$ with $|\kappa_1| = 1$ and $|\kappa_2| = 4$ is highlighted by a hatched plane. This hatched plane is the biased multiplex $\Gamma^{\boldsymbol{\xi}}$ with $\boldsymbol{\xi} = (\frac{1}{5}, \frac{4}{5})$. By making different subdivisions of the degree over the layers, the biased multiplices of Figure 4-5b are found. Defining a multiplex spline on the shaded multiplex amounts to projecting the B-coefficients on one of the biased multiplices onto the multiplex.



(a) $d = (1, 4)$ **(b)** $d = (d_1, d_2)$

Figure 4-5: The B-net of the 3-simplex $\{v_{10}, v_{11}, v_{20}, v_{21}\}$ as black circles. The inscribed (1,1)-multiplex is shaded transparent gray. The biased multiplex is shown for the indicated multiplex polynomial degrees d. The names of the B-coefficients are only provided for a small group of coefficients for clarity.

4-4 Continuity

In this section many of the results from the previous sections are combined to arrive at continuity conditions for a general multiplex spline. It will be discussed how these conditions can be taken from simplex spline theory.

To properly introduce the continuity conditions the first subsection aims to combine all top-down definitions of the previous sections. This results in the equivalent simplex spline, of which the multiplex spline is a lower-dimensional slice. The shape of the shared edge between two multiplices is studied in subsection 4-4-2. The knowledge of the shared edge and the equivalent simplex spline are combined in subsection 4-4-3 to define continuity conditions between equal type multiplices. Finally in subsection 4-4-4 continuity conditions between two different type, yet equal-dimensional multiplices are formulated.

4-4-1 The equivalent simplex spline

In the previous sections it was found that for every defining element of the multiplex spline, a higher-dimensional simplex equivalent exists. The multiplex is a cut of a higher-dimensional simplex, the basis polynomials form a scaled subset of a simplex polynomial and the B-net is a slice of the same higher-dimensional B-net. These concepts are now combined to define the equivalent simplex spline.

Definition 9 (Equivalent simplex spline)**.** *Let s_ν^d be a multiplex spline of degree d and dimension ν, defined on the ν-multiplex Γ, with a B-net $\boldsymbol{\mathcal{C}}$, basis polynomials $\boldsymbol{\mathcal{B}}_\nu^d$ and the corresponding set of multi-index permutations $\boldsymbol{\lambda}$. The* equivalent simplex spline *s_m^e of s_ν^d is the simplex spline with degree $e = |d|$, defined on the m-simplex Δ with $m = n + \ell - 1$, where $n = |\nu|$ and ℓ is the number of layers in Γ, with a B-net $\boldsymbol{C}$, basis polynomials $\boldsymbol{B}_m^e$ and multi-index permutations $\boldsymbol{\kappa}$ such that*

- Γ *is inscribed in* Δ*;*
- $c_\kappa = 0$ *for* $\kappa \in \boldsymbol{\kappa}$ *if* $\kappa \notin \boldsymbol{\lambda}$*;*
- $c_\kappa = \ell^e \frac{d!}{e!} c_\lambda$ *for* $\kappa \in \boldsymbol{\kappa}$ *if* $\kappa = \lambda \in \boldsymbol{\lambda}$*.*

By setting the B-coefficients zero if they do not lie in the multiplex B-net, all corresponding basis polynomials are eliminated. The other basis polynomials are scaled via the B-coefficients, such that function values are equal at the slice of Δ where Γ lies.

From the results of the previous sections it is clear that properties of the equivalent simplex spline can often be translated directly to the multiplex spline. The most notable example is continuity, which is the topic of the remainder of this section.

4-4-2 Shared edge between multiplices

Before discussing continuity conditions, it is important to define the edge shared by multiplices. In the case of the simplex spline, a shared edge between two n-simplices is an $(n-m)$-simplex (with $m \leq n$) [16]. This result is combined with the concept of the equivalent simplex spline in this subsection to arrive at the shared edge between multiplices.

In the remainder of this section it is assumed that the multiplices can be oriented such that they share a complete edge.

Theorem 2 (Shared edge)**.** *Consider the* ν*-multiplices* Γ *and* $\tilde{\Gamma}$*,* $|\nu| = n$*. A shared edge* $\bar{\Gamma}$ *between* Γ *and* $\tilde{\Gamma}$ *is a* $(\nu - \mu)$*-multiplex, with* $\mu \geq 0$ *a multi-index with as many entries as* ν*, as long as* $\nu - \mu \geq 0$*.*

Proof. We start with the case $|\mu| = 1$. Consider the $(n + \ell - 1)$-simplices Δ defined around Γ and $\tilde{\Delta}$ around $\tilde{\Gamma}$. Assume Δ and $\tilde{\Delta}$ share an $(n + \ell - 2)$-edge $\bar{\Delta}$.

If the out-of edge vertex v_{ooe} lies in the i^{th} layer set such that $\nu_i > 0$, then the same layer sets can be defined in $\bar{\Delta}$ as in Δ and $\tilde{\Delta}$. Reusing the simplex links that do not include v_{ooe} in definition 5 results in an inscribed $(\nu - \mu)$-multiplex $\bar{\Gamma}$ in $\bar{\Delta}$ with $\mu_i = 1$. Because the same simplex links are also used for the construction of both Γ and $\tilde{\Gamma}$, all three multiplices $\bar{\Gamma}$, Γ and $\tilde{\Gamma}$ share the extreme points defined by these simplex links. Therefore $\bar{\Gamma} \subset \Gamma$, $\bar{\Gamma} \subset \tilde{\Gamma}$ and thus $\bar{\Gamma} \subset (\Gamma \cap \tilde{\Gamma})$.

To show that $\bar{\Gamma} = \Gamma \cap \tilde{\Gamma}$ we observe that all points $\boldsymbol{x} \in \Gamma$, yet $\boldsymbol{x} \notin \bar{\Gamma}$, require v_{ooe} for their definition. Because Δ and $\tilde{\Delta}$ do not overlap, we find that $\boldsymbol{x} \notin \tilde{\Gamma}$.

For the more general case where $|\mu| > 1$ the above proof can still be used. In this case we iterate on the description of multiplices $\bar{\Gamma}$ of ever-decreasing dimension as an edge of Γ, until the appropriate dimension is achieved. □

From the above discussion we find that an $(n-1)$-edge of a ν-multiplex is a $(\nu-\mu)$-multiplex, with $|\mu| = 1$ and $\mu \geq 0$. That is, the two multiplices only differ in a single layer in terms of their layer dimensions. It is useful to give the layer in which they differ a name.

Definition 10 (Out-of-edge layer)**.** *Consider a ν-multiplex Γ, $|\nu| = n$, and a multiplex $\tilde{\Gamma}$ that share a $(\nu - \mu)$-multiplex $\bar{\Gamma}$, with $|\mu| = 1$ and $\mu \geq 0$. The* out-of-edge layer *of Γ is the layer in which $\bar{\Gamma}$ has a different dimension.*

As μ described in the above definition has only one non-zero value, we may also say that the out-of-edge layer is the i^{th} layer of Γ such that $\mu_i = 1$. Also, if all layers of Γ are defined such that they lie on the shared edge, only the out-of-edge layer will stick out. The vertex that is not on the shared edge is the out-of-edge vertex of Γ. Note that this implies that the out-of-edge vertex of a multiplex lies in its out-of-edge layer.

A remaining question is what happens when the out-of-edge vertex lies in a zero-dimensional layer set (that is, the layer set is eliminated). This corresponds to the condition $\nu - \mu \not\geq 0$.

Proposition 6 (No shared edge)**.** *Consider the simplices Δ and $\tilde{\Delta}$ sharing an edge $\bar{\Delta}$ and the inscribed multiplices Γ and $\tilde{\Gamma}$. If a layer set of Δ is entirely out-of-edge for $\bar{\Delta}$, Γ and $\tilde{\Gamma}$ do not share an edge.*

Proof. Assume that in the process of eliminating vertices from Δ we arrive at an intermediate simplex Δ_I of which one layer set is zero-dimensional. Now if the vertex v making up the zero-dimensional layer set is taken to be the next out-of-edge vertex, we arrive at $\bar{\Delta}_I$ as the shared edge between Δ_I and $\tilde{\Delta}_I$. Considering the inscribed multiplices Γ_I in Δ_I and $\bar{\Gamma}_I$ in $\bar{\Delta}_I$, we find using definition 5 and lemma 1 that Γ_I is a scaled, parallel copy of $\bar{\Gamma}_I$ translated in the direction of v. Since this direction is not spanned by $\bar{\Gamma}_I$ we have $\Gamma_I \cap \bar{\Gamma}_I = \emptyset$. Combining this with the proof of theorem 2 we find that Γ_I and $\tilde{\Gamma}_I$ do not share an edge. □

The importance of this result is that the shared edge between multiplices lies in the shared edge between the circumscribed simplices. This implies that continuity conditions can be taken from the equivalent simplex spline directly. This is the cornerstone of the remainder of this section.

4-4-3 Continuity conditions between equal multiplices

In this subsection continuity conditions are formulated for connections between two equal ν-multiplices Γ and $\tilde{\Gamma}$. Equal in this case means that they have the same dimensionality in all layers, or ν is the same in both multiplices.

The first way of arriving at continuity conditions is to set the derivatives of the polynomials on both multiplices equal on both sides of the shared edge. Using this approach, we expect to require at least continuity in n linearly independent directions for omnidirectional continuity. An example could be in-layer continuity.

Definition 11 (In-layer continuity)**.** *Consider two ν-multiplices with layers γ_1 up to γ_ℓ and $\tilde{\gamma}_1$ up to $\tilde{\gamma}_\ell$ sharing all layers but γ_i and $\tilde{\gamma}_i$.* In-layer continuity *is simplex continuity between γ_i and $\tilde{\gamma}_i$ imposed on all slices of the B-net parallel to γ_i.*

Although in-layer continuity obviously guarantees continuity in directions parallel to a layer, it is not immediate that polynomials are also continuous in other directions. For this we turn to the equivalent simplex spline.

In a simplex spline, continuity is guaranteed if the coefficients on the shared edge are equal in both simplices Δ and $\tilde{\Delta}$ and the derivatives are equal in the direction of the out-of-edge vertex [30]. In general we can choose any direction from a vertex on the shared edge to the out-of-edge vertex. Assuming the out-of-edge vertex is named first in both simplices, the k^{th} order continuity conditions simplify to

$$\tilde{c}_{(k,\kappa_1,\ldots,\kappa_m)} = \sum_{|\sigma|=k} c_{(0,\kappa_1,\ldots,\kappa_m)+\sigma} B^k_\sigma(\tilde{v}_{ooe}) \tag{4-14}$$

with σ a multi-index and $\tilde{v}_{ooe}$ the out-of-edge vertex of $\tilde{\Delta}$. Note that $\tilde{v}_{ooe}$ should be defined using barycentric coordinates in Δ.

In-layer continuity can be directly related to continuity in the equivalent simplex spline.

Theorem 3 (Continuity equivalence)**.** *Let s^d_ν be a (ν)-multiplex spline of degrees d and s^e_m its equivalent simplex spline as in definition 9 with dimension $m = n + \ell - 1, n = |\nu|$ and degree $e = |d|$. Continuity between two ν-multiplices Γ and $\tilde{\Gamma}$ that share a $(\nu - \mu)$-edge $\bar{\Gamma}$, where $|\mu| = 1$, is equivalent to continuity in s^e_m between two m-simplices Δ and $\tilde{\Delta}$ that share an $(m-1)$-edge $\bar{\Delta}$ such that $\Gamma \subseteq \Delta$, $\tilde{\Gamma} \subseteq \tilde{\Delta}$ and $\bar{\Gamma} \subseteq \bar{\Delta}$.*

Proof. The directional derivative of s^d_ν can be easily derived from the derivative of s^e_m, combined with equation (4-10). From the discussion of theorem 2.28 in [30] we have that the general directional derivative of s^e_m is a linear combination of derivatives in m independent directions. For s^d_ν only the directions that lie in the multiplex remain. That is, a total of $n = |\nu|$ directional derivatives is required.

In s^e_m the first $m-1$ directions are obtained by setting the B-coefficients that lie in $\bar{\Delta}$ equal. Because only a slice of the B-net is non-zero, many of these constraints are superfluous. By eliminating those constraints only $n-1$ constraint directions remain.

The final direction in both cases is the direction out of the shared edge, towards $\tilde{v}_{ooe}$. This results in equation (4-14) for both cases. □

The above theorem shows that we can use equation (4-14) for multiplex continuity. An interesting special case of these conditions occurs when the out-of-edge layers of Γ and $\tilde{\Gamma}$ are cospatial. With cospatial it is meant that the two ν_i-dimensional layers can be defined using the same basis for $\mathbb{R}^{\nu_i}$. Cospatial lines are collinear, cospatial triangles are coplanar, and so forth.

Proposition 7 (Continuity in cospatial layers)**.** *Let Δ and $\tilde{\Delta}$ be two m-simplices in the equivalent simplex spline s^e_m of a multiplex spline s^d_ν, sharing an $(m-1)$-edge. Consider the case where the out-of-edge vertex lies in the i^{th} layer set in both simplices. Then equation* (4-14) *simplifies to in-layer continuity in the i^{th} layer if those layers are cospatial.*

Proof. Using $n+1$ barycentric coordinates with respect to an n-simplex, any point can be described in $\mathbb{R}^n$ as long as the coordinates are allowed to be negative. This means that if two layers of different multiplices are cospatial, any point in the layer of the first multiplex can be defined using only barycentric coordinates in the same layer of the other multiplex.

Returning to the continuity conditions in equation (4-14) we observe that we need a barycentric description of $\tilde{v}_{ooe}$ in Γ. The above implies that this can be done using only barycentric coordinates in the i^{th} layer of Γ. The other barycentric coordinates are zero. If σ has non-zero entries out of the i^{th} layer set, the basis polynomial reduces to zero. Eliminating these zero terms results in

$$\tilde{c}_\lambda = \sum_{|\rho|=k} c_{(\lambda_1,\ldots,\lambda_{i-1},\lambda_i+\rho,\lambda_{i+1},\ldots,\lambda_\ell)} B^k_\rho(\tilde{v}_{ooe}) \tag{4-15}$$

which is a simplex spline continuity condition in the i^{th} layer. Note that we use simplex basis polynomials, which is equivalent to the multiplex basis polynomial with degree zero in all other layers, if we eliminate all inherently zero elements of $b(\tilde{v}_{ooe})$. Also, scaling of the B-coefficients due to the conversion from equivalent simplex spline to multiplex spline is canceled. □

To clarify the concepts discussed in this subsection, an example is presented. We use the (1,1)-multiplex to be able to make illustrations within three dimensions.

Example 6 (Continuity between (1,1)-multiplices)**.** The top down definition of continuity conditions is illustrated in Figure 4-6. In Figure 4-6a two 3-simplices Δ and $\tilde{\Delta}$ and their B-net for polynomial degree 4 are shown. The (1,1)-multiplices Γ and $\tilde{\Gamma}$ that can be inscribed in both these simplices by using the layer sets $\{v_{10}, v_{11}\}$ and $\{v_{20}, v_{21}\}$ are the same as those in Figure 4-3a.

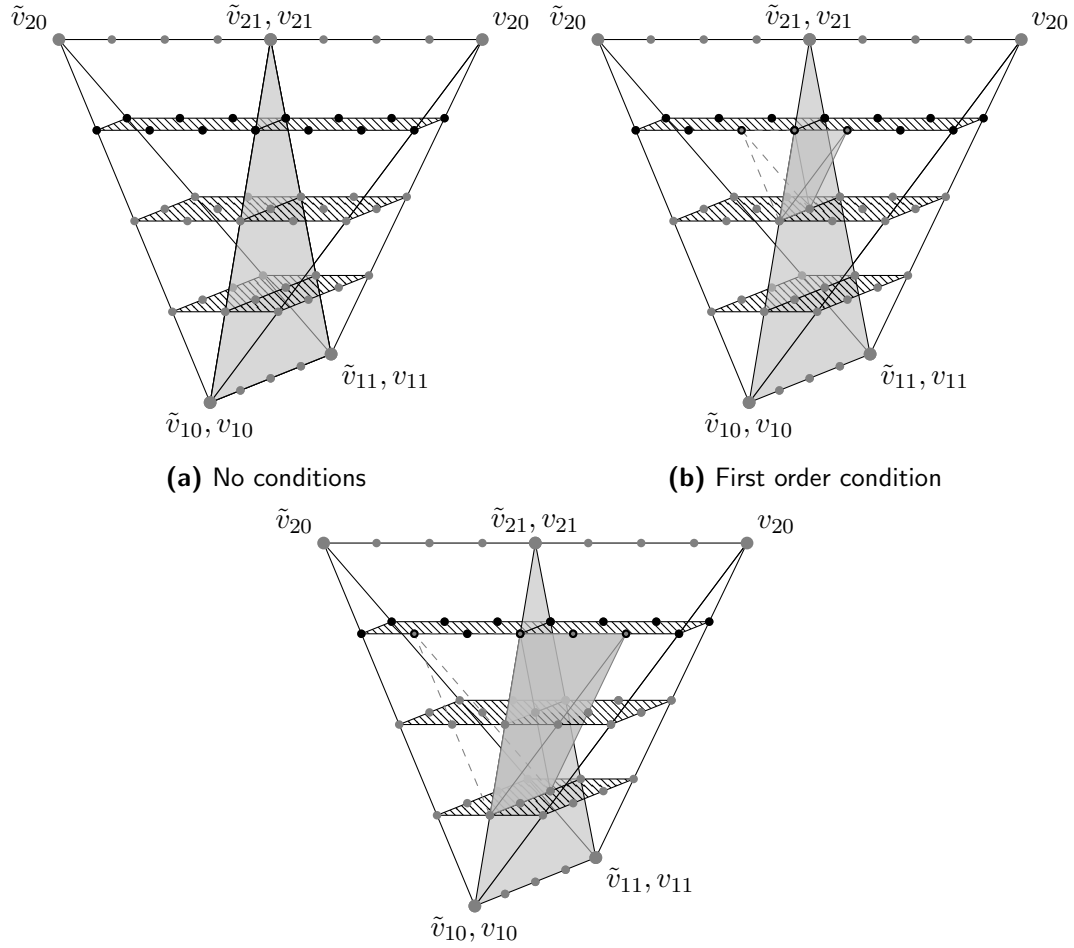
Let's assume that we are interested in a multiplex polynomial of degree (1,3). This corresponds to the top biased multiplex in Figure 4-6, $\Gamma^{(1/4,3/4)}$ in Δ and $\tilde{\Gamma}^{(1/4,3/4)}$ in $\tilde{\Delta}$. In Figure 4-6b a first order continuity condition between Δ and $\tilde{\Delta}$ is drawn. The coefficients that do not lie in the biased multiplex of interest however are set to zero, and do therefore not contribute. What remains is a collinear condition involving three coefficients. If another condition is chosen of which the coefficient in $\tilde{\Delta}$ does not lie on the biased multiplex $\tilde{\Gamma}^{(1/4,3/4)}$, a condition remains that sets the weighted sum of B-coefficients in Γ to zero. This is odd, since such a continuity condition is independent of the function values in $\tilde{\Gamma}$.

Only if the second layer sets $\{v_{20}, v_{21}\}$ and $\{\tilde{v}_{20}, \tilde{v}_{21}\}$ are collinear (as is the case in this example), this problem does not occur. From proposition 7 we find that such a condition would be zero in its totality and thus would not pose a constraint.

Figure 4-6c shows a second order continuity condition. Again only the top row of coefficients actually takes part in the condition for the inscribed multiplices, due to proposition 7.

4-4-4 Continuity conditions between different multiplices

Continuity conditions can only be defined between two domain elements if they share an $(n-1)$-dimensional edge. However, it is not necessary that both multiplices have an equal shape. To be more precise, theorem 2 can be expanded to state that for a ν-multiplex Γ and a $\tilde{\nu}$-multiplex $\tilde{\Gamma}$ to share an $(n-1)$-edge, we require $|\nu| = |\tilde{\nu}| = n$ and $\nu - \mu = \tilde{\nu} - \tilde{\mu}$. Here the multi-indices μ and $\tilde{\mu}$ are again defined by the conditions $\mu, \tilde{\mu} \geq 0$ and $|\mu|, |\tilde{\mu}| = 1$. In this subsection continuity conditions that arise in such situations are derived and described.



(a) No conditions

(b) First order condition

(c) Second order condition

Figure 4-6: Continuity between two 3-simplices Δ (right) and $\tilde{\Delta}$ with polynomial degree 4. The biased multiplices are hatched in both simplices. Continuity conditions in the simplex spline take the form of a single B-coefficient in one simplex (where the dashed lines intersect) and a miniature n-simplex in the other. In the (1,1)-multiplex spline with degree $d = (1, 3)$ only the black B-coefficients are nonzero, yielding collinear continuity conditions as described by proposition 7. The B-coefficients affected by the continuity condition are black with a gray center.

Instead of setting the derivative on both ends of the shared edge equal [31], we choose to take a more general approach through the circumscribed simplex. To illustrate the procedure, we start with an example for continuity between a rectangle and a triangle.

Example 7 (First order rectangle-triangle continuity)**.** We start by identifying both shapes as multiplices. The rectangle is a $(1,1)$-multiplex Γ, as it can be constructed by combining two one-dimensional simplices (line segments). The vertices of the layers are depicted by w_{10} up to w_{21} in Figure 4-7a. The triangle on the other hand is the 2-simplex. However, a zero-dimensional layer can be added without changing the shape, which makes the triangle a $(2,0)$-multiplex $\tilde{\Gamma}$. This implies that we can keep on adding zero-dimensional layers, but in general we never require $\ell > n$. The vertices of $\tilde{\Gamma}$ are indicated by $\tilde{w}_{ij}$. Note that $\tilde{w_{20}}$ is added for completeness, but its location is arbitrary.

Second, we verify whether Γ and $\tilde{\Gamma}$ share an edge. Indeed, if we choose $\mu = (0,1)$ and $\tilde{\mu} = (1,0)$, then $\nu - \mu = \tilde{\nu} - \tilde{\mu} = (1,0)$. That is, Γ and $\tilde{\Gamma}$ share a 1-simplex, a line segment. For the rectangle Γ this corresponds to the entire first layer, for the triangle this is only one edge of the first layer.

The circumscribed simplices are of dimension $n + \ell - 1$. For the rectangle this clearly results in a 3-simplex. By adding the zero-dimensional layer, the same holds for the triangle. The multiplices Γ and $\tilde{\Gamma}$ and their circumscribed simplices Δ and $\tilde{\Delta}$ are depicted in Figure 4-7a. The B-net for a bi-quadratic rectangle and a cubic triangle are shown in relation to the total degree simplex B-net in Figure 4-7b.

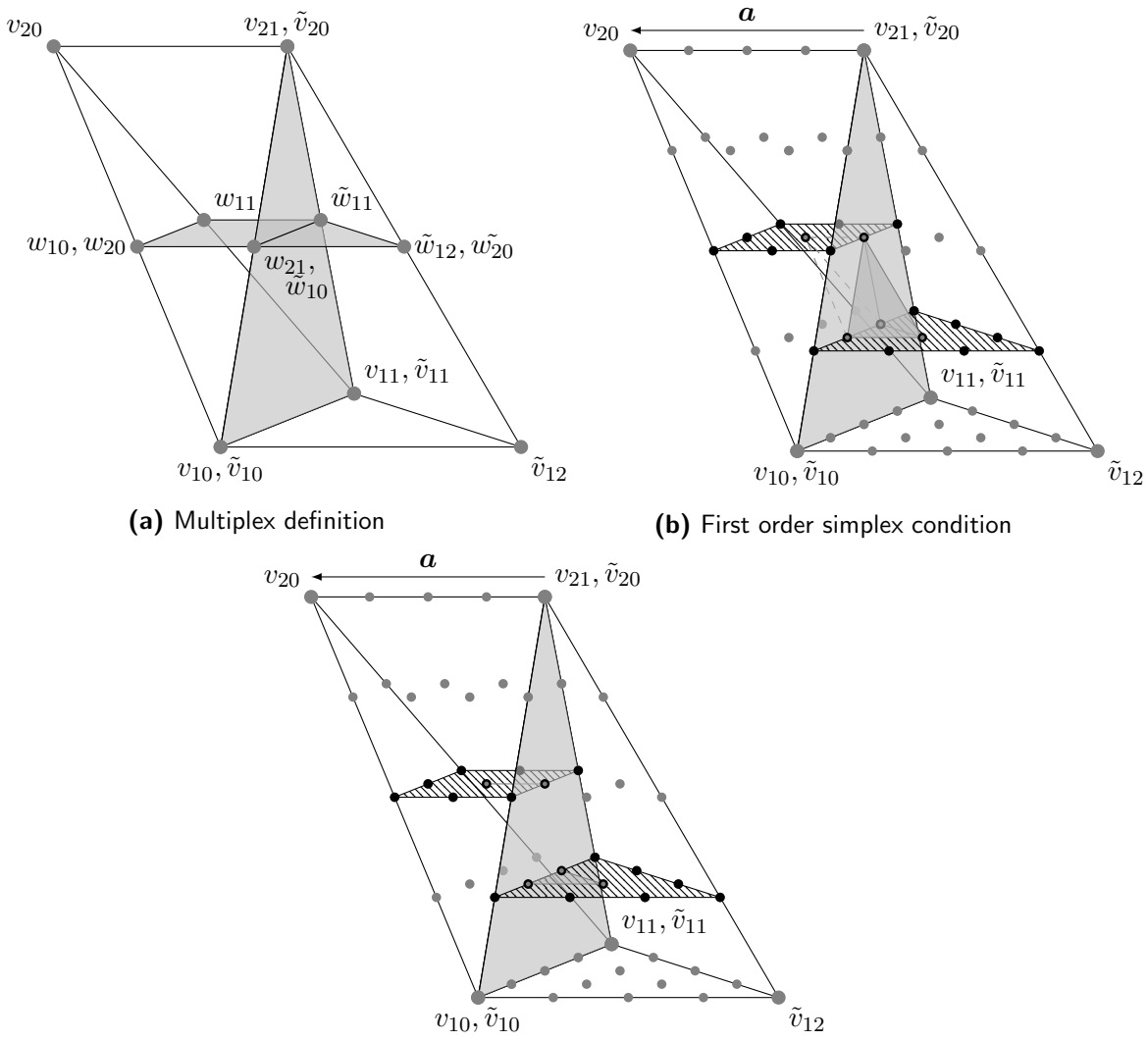
To have a k-continuous join between the two multiplices, we require that the k^{th} order derivatives of the polynomials on the shared edge are equal. Lai shows that in the simplex spline this is equivalent to setting the k^{th} de Casteljau iterations of B-coefficients at a distance k from the shared edge, equal in both simplices [30]. These iterations are performed using directional coordinates describing a vector not parallel to the shared edge.

The first order continuity conditions can be greatly simplified by choosing these directional coordinates $\boldsymbol{a}$ from v_{21} to v_{20}. We then obtain $\boldsymbol{a} = (a_{10}, a_{11}, a_{20}, a_{21}) = (0,0,1,-1)$ and $\tilde{\boldsymbol{a}} = (\tilde{b}_{10}, \tilde{b}_{11}, \tilde{b}_{12}, 0)$. Contrary to standard simplex continuity (shown in Figure 4-7b and derived by Lai [30]), the term a_{21} is not brought to the side of the triangle. Therefore only the terms of the de Casteljau iteration in $\tilde{\Gamma}$ in which λ_{20} is not changed, are non-zero. This effectively means that the B-coefficients that are affected by the continuity condition, lie in a plane parallel to $\tilde{\Gamma}$. The resulting conditions is shown in Figure 4-7c.

For the zeroth order continuity conditions to also hold, the cubic polynomial on the shared edge should be reduced to a quadratic one. On top of that, the zeroth order conditions on quadratic coefficients should be converted to hold for the cubic coefficients.

The process described in example 7 above can be generalized to any number of dimensions, number of layers, degrees and continuity orders. The resulting procedure is presented in the remainder of this subsection. First the geometric considerations are described. Second the continuity conditions in the circumscribed simplex are considered, as well as the way in which they simplify. Finally the reduction of all continuity conditions to a single degree polynomial is discussed briefly.

As implied by theorem 2, the shared edge $\bar{\Gamma}$ of Γ and $\tilde{\Gamma}$ is a multiplex with circumscribed simplex $\bar{\Delta}$. This in turn implies that we can always define circumscribed simplices Δ and $\tilde{\Delta}$



(a) Multiplex definition

(b) First order simplex condition

(c) First order multiplex condition

Figure 4-7: Continuity conditions between different type multiplices can be found using the top-down approach. In Figure 4-7a the geometric definition of multiplices and circumscribed simplices is presented. In Figure 4-7b an example first order continuity condition in the equivalent simplex spline is shown. The gray coefficients are set to zero, whereas the ones with a gray core are affected by the continuity condition. By choosing $\boldsymbol{a}$ as indicated, the conditions simplify to the ones in Figure 4-7c. Note that we use a higher degree in the triangular patch. In this case the rectangular patch is biquadratic and the triangular one is cubic.

such that they share the edge $\bar{\Delta}$. The remaining out-of-edge vertices should be defined such, that Γ and $\tilde{\Gamma}$ are cospatial. From lemma 1 it can be deduced that the out-of-edge layer sets need to be parallel.

Before continuing with the derivation of continuity conditions, we introduce some notation regarding the polynomial expressions. In general terms, all but two layers will be completely shared. We give those equal indices in both multiplices. The two remaining (out-of-edge) layers are aligned with the layers with which they share all but one vertex. Without loss of generality we can choose w_{10} and $\tilde{w}_{\ell\nu_\ell}$ as the out-of-edge vertices. This means that in our search for continuity conditions we generally require $(\kappa_{10}, ..., \kappa_{\ell\nu_\ell}) = (\tilde{\kappa}_{\ell\tilde{\nu}_\ell}, \tilde{\kappa}_{10}, ..., \tilde{\kappa}_{\ell,\tilde{\nu}_\ell-1})$. Note that a distinction is made here between κ and $\tilde{\kappa}$.

Continuity conditions of order k between simplices, as defined in equation (4-14), are derived by setting the derivatives equal on both sides of the shared edge. If we do this in the equivalent simplex splines defined in definition 9, we find for continuity order k

$$\begin{aligned}\frac{e!}{(e-k)!}\sum_{|\kappa|=e-k} c^{(k)}_{(0,\kappa_{11},...,\kappa_{\ell\nu_\ell})}(\boldsymbol{a})B^{e-k}_{(0,\kappa_{11},...,\kappa_{\ell\nu_\ell})} = \\ \frac{e!}{(e-k)!}\sum_{|\tilde{\kappa}|=e-k} \tilde{c}^{(k)}_{(\tilde{\kappa}_{10},...,\tilde{\kappa}_{\ell,\tilde{\nu}_\ell-1},0)}(\tilde{\boldsymbol{a}})B^{e-k}_{(\tilde{\kappa}_{10},...,\tilde{\kappa}_{\ell,\tilde{\nu}_\ell-1},0)}\end{aligned} \tag{4-16}$$

As the basis polynomials are equal on the shared edge, these conditions quickly simplify to equation (4-16) [30].

$$c^{(k)}_{(0,\kappa_{11},...,\kappa_{\ell\nu_\ell})}(\boldsymbol{a}) = \tilde{c}^{(k)}_{(\tilde{\kappa}_{10},...,\tilde{\kappa}_{\ell,\nu_\ell-1},0)}(\tilde{\boldsymbol{a}}), \quad |\kappa| = e-k \tag{4-17}$$

That is, the de Casteljau iterations must be equal on both sides of the shared edge.

The directional coordinates $\boldsymbol{a}$ and $\tilde{\boldsymbol{a}}$ describe a vector that is not parallel to the shared edge. Because the out-of-edge layer sets must be parallel (as stated before), it is possible to choose a vector that is parallel to both these layer sets. One obvious choice is the vector between the out-of-edge vertex and any other vertex of the lowest-dimensional of the two out-of-edge layer sets. This vector will definitely also lie in the other out-of-edge layer set. In other words, the coordinates $\boldsymbol{a}$ and $\tilde{\boldsymbol{a}}$ will have zero elements corresponding to vertices from all other layer sets. This leads to the significant simplification that the de Casteljau iterations of equation (4-17) reduce to iterations in the out-of-edge layer sets. Combining this with the scaling of the B-coefficients in definition 9 and by removing all zero terms from $\boldsymbol{a}$ and $\tilde{\boldsymbol{a}}$, we find the following continuity conditions.

$$d!\left(c^{(k)}_{(0,\lambda_{11},...,\lambda_{1\nu_1})}(\boldsymbol{a})\right)_{(\lambda_2,...,\lambda_\ell)} = \tilde{d}!\left(\tilde{c}_{(\tilde{\lambda}_1,...,\tilde{\lambda}_{\ell-1})}\right)^{(k)}_{(\tilde{\lambda}_{\ell 0},...,\tilde{\lambda}_{\ell,\nu_\ell-1},0)}(\tilde{\boldsymbol{a}}) \tag{4-18}$$

Note that all collected terms of λ and $\tilde{\lambda}$ are not affected by this de Casteljau iteration, and that the (summing) conditions from equation (4-16) still hold.

Equation (4-18) provides a very clear and insightful description of continuity conditions between different multiplices. After all, the conditions take the form of miniature copies of the

out-of-edge layers. But these sets of B-coefficients may not always align with each other on both ends of the shared edge. This can be explained as follows. Before performing the de Casteljau iteration in (4-18), we find both in the first and the last layer a difference in degree. For example, we may observe that as $(\lambda_{11}, ..., \lambda_{1\nu_1}) = \tilde{\lambda}_1$ and $\lambda_{10} = k$, we have $|\lambda_1| = |\tilde{\lambda}_1| + k$. In layer ℓ we find the similar condition $|\lambda_\ell| + k = |\tilde{\lambda}_\ell|$, which indicates a shift in degree of k from layer 1 to layer ℓ. This shift in degree is equal to the continuity order k and will thus be different for each order. To collect all conditions in the B-net of one single set of degrees d and $\tilde{d}$, we may either perform degree raising or lowering on the conditions. If we choose to lower the degree of the conditions (so that $d = \tilde{d}$), extra requirements must be satisfied on the degree to guarantee total approximation power [31]. If a degree raising of lower order conditions is performed, no such constraints are anticipated, as the number of B-coefficients is increased. The procedure is as follows. First the degree raising function of Lai is used to find expressions for the high degree coefficients [30].

$$c_\lambda = \frac{1}{d_i + 1} \sum_{|\rho|=1} \lambda_{ij} c_{(\lambda_1,...,\lambda_{i-1},\lambda_i-\rho,\lambda_{i+1},...,\lambda_\ell)}, \qquad \rho_j = 1$$

The resulting conditions are collected in a matrix, of which the left inverse is taken. Because of this inverting operation, the continuity conditions are scattered over a large range of coefficients and loose their clear structure.

On top of these conversions, extra conditions are required to ensure that the degree is sufficiently low at or near the shared edge. To be more precise, the degree must be constrained such that the effective degree of the polynomials described by $c_{k,\lambda_1 1,...,\lambda_{\ell\nu_\ell}}$ is $\tilde{d} + k$. This is done by converting the barycentric coordinates to Cartesian coordinates, assuming $\beta_{ij} = x_{ij}$ and $\beta_{10} = 1 - x_{11} - ... - x_{\ell\nu_\ell}$. After filling this into the basis polynomials, the high-degree elements are grouped according to degree distribution over the variables, and the sum per group is set to zero.

The described procedure has been successfully tested for two-dimensional tessellations with first order continuity conditions. As a matter of fact, the front page of this document presents a first order continuous two-dimensional mixed multiplex spline. On the rectangles bi-cubic polynomials are defined, on the triangles quartic ones.

The top-down approach for finding continuity conditions can also be used to connect two equal multiplices of which the out-of-edge layers are not cospatial. In that case the simplex spline continuity conditions actively require all coefficients, which prevents the proposed simplification. However, by using degree raising or lowering on the proper coefficients, continuity conditions can be obtained [31]. Because this situation has not been studied in detail, it is not discussed further here.

Chapter 5

System identification

In chapter 4 the mathematical basis was laid for the multivariate multiplex spline. The remaining question in this thesis is whether it can be applied for system identification purposes. To answer this question, a study was undertaken into the behavior of the multiplex spline on different test functions and data distributions. The results of this study were then used to formulate and verify a system identification algorithm.

In section 5-1 some characteristics of the multiplex spline are discussed as results of its definition. Examples include the directionality of polynomial degree and the linearity in the parameters. This culminates in a short discussion of possible techniques of finding optimal B-coefficients based on a scattered data set. Next in section 5-2 a set of test functions and data distributions is used to clarify the behavior of the multiplex spline. The efforts of defining multiplex splines results in an algorithm for system identification, presented in section 5-3. Finally in section 5-4 a quadrotor model is set up to generate a representative data set to fit a multiplex spline to. As stated in section 2-2 the quadrotor is used here because of the availability of reliable and well-understood analytic models. A multiplex and simplex spline are generated and compared in the same section.

5-1 Multiplex spline characteristics

The description of the multiplex spline in chapter 4 focused purely on the mathematical basis, whereas the remainder of this report describes the application. To bridge this gap, in this section some of the most important characteristics and behaviors of the multiplex spline are discussed. This discussion will provide essential explanations for effects that will be encountered in the remainder of this and the next chapter.

To structure the discussion, a subdivision in subsections is used. Subsection 5-1-1 is meant to provide some insights in the geometric effects of using multiplices instead of simplices. Next, in subsection 5-1-2 the effects on polynomial degree and continuity order are considered. Finally in subsection 5-1-3 the available optimization techniques are discussed.

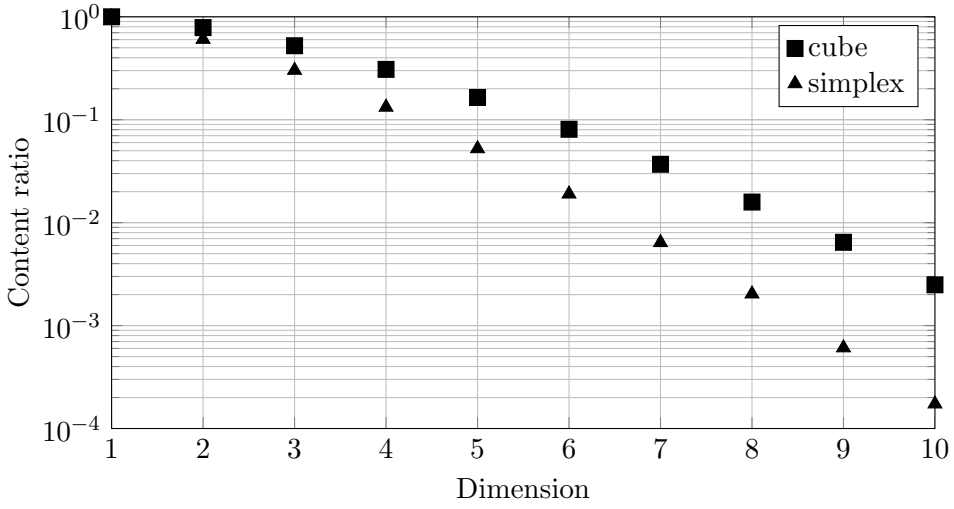


Figure 5-1: With increasing dimension it becomes increasingly hard to properly fill a polytope with data points. The advantage in this respect of an n-cube over an n-simplex is the concentration of content near the center. This can be deduced from the ratio of content of the polytopes and their respective inscribed circles. As content moves away from the center it becomes harder to properly fill all corners with data, resulting in reduced numerical performance.

5-1-1 Geometric characteristics

In this subsection the simplex and multiplex are compared based on their content distribution. This is an important measure for the ease with which a domain element can be uniformly filled with data. On top of that, two strategies for defining a tessellation of multiplices are presented.

In the literature study (subsection 3-1-2) the content distribution in simplices was mentioned. A potential benefit of the multiplex as a domain element is that it may have a better distribution of content. One way of establishing this mathematically is by comparing the ratio of the content of a polytope to the content of its inscribed sphere. This inscribed sphere, or insphere, is the n-sphere that touches all $(n-1)$-edges of a polytope. The method is inspired by Tarantola, who uses the following example to illustrate the emptiness of high dimensional space [41]. Consider an n-cube and its inscribed n-sphere. As the dimension n increases it becomes increasingly hard to hit the inscribed sphere when throwing a dart into the cube. More precisely, more and more content of the cube tends away from the core. This is easily found from the decreasing ratio between the content of the cube and its inscribed sphere. Here this example is extended by including the regular n-simplex and its inscribed sphere, for which Coxeter and Kendall were consulted [12, 28]. The content ratios of both polytopes are given here as Figure 5-1. It is clear from this figure that in the cube the content is concentrated around the center more than in the simplex. In practice this means that it is easier to fill a cube with data points evenly. Since local lack of data causes unwanted freedom for the polynomials, the cube seems to be the preferred geometric basis from this perspective. It is expected that the ratios for the other multiplices will lie between the ones of the cube and the simplex. In fact, a trend can be expected in which more and lower-dimensional layers provide a better content ratio for a given dimension than fewer, high-dimensional layers.

To form a tessellation, several multiplices need to be combined to span a domain. In its purest form, tessellating a domain amounts to placing multiplices across the domain in such a way that they always share complete edges and together they encompass the entire domain. These rules for tessellations have been discussed in more detail in subsection 3-1-1. This direct form of tessellation allows for the use of different types of multiplices in a single tessellation, but is hard to perform for high dimensional domains. At this point there is no known algorithm but visual inspection to construct a tessellation in this way.

A more practical approach to tessellating high dimensional domains is the combination of layer triangulations. This procedure consists of four steps. First the variables are subdivided over a set of low-dimensional layers. Using one- and two-dimensional layers is prefered, as it simplifies the second step, in which the layers are triangulated. The triangulations in the layers are combined in the third step to form a valid full-dimensional tessellation. This amounts to constructing a multiplex from every possible combination of simplices from each layer. Finally in the fourth step, multiplices are removed if they contain too little data points to estimate the coefficients.

This approach is especially useful when data sets are higher dimensional. To construct two 4-cubes that share a cubic edge may be a challenge if it has to be done with the naked eye. With this approach one simply constructs four one-dimensional layers, of which one contains two simplices and the others one. The combination can be done fully automatically, and the result is a valid tessellation. This is therefore the preferred method in the current research, and forms a major part of the system identification algorithm presented in section 5-3.

5-1-2 Spline degree and continuity order

Because the polynomials of a multiplex spline are the result of a tensor product of polynomials, we may choose different degrees and continuity orders in different layers. This implies that continuity has a more directional nature in the multiplex spline. On top of that, the total degree of these polynomials may quickly rise, introducing the risk of Runge's phenomenon. Both these effects are described in this subsection.

In studying the continuity order, it is useful to first consider a small example. Two rectangles that share a complete edge may be joined with arbitrary continuity order. From the top-down approach of defining continuity conditions described in subsection 4-4-3 we can conclude that continuity is omnidirectional across the shared edge, and equal to that chosen in the out of edge layer. This example can be generalized to state that the continuity order is dictated by the order in the out-of-edge layer for the shared edge of interest. On top of that, this order holds for any direction, as long as a point on the shared edge is considered. On an intersection of multiple shared edges the lowest continuity order dictates the overall order.

It was observed earlier, in subsection 4-2-2, that the basis polynomials can be seen as higher dimensional, total degree polynomials. This total degree increases quickly and may cause unwanted effects, such as Runge's phenomenon. This is especially important along the diagonals of a multiplex, at the extremes of the domain. Along a diagonal multiple barycentric coordinates from different layers are equal, resulting in pure total degree polynomials. For example, in a cube (or (1,1,1)-multiplex) with in each layer a cubic polynomial, the degree along the diagonal is 9. This high degree may lead to overfitting or even the occurrence of Runge's phenomenon. This problem is intensified by the fact that a diagonal runs between

vertices, implying that, especially in higher dimensional multiplices, very little data may be available at the ends of a diagonal. On the other hand, the height of the total degree also assures high approximation power, even when degrees are chosen low in all layers.

5-1-3 Scattered data fitting

The final step in generating spline models is finding an estimate for the B-coefficients. The structure of the optimization problem of finding this estimate leads to a set of estimators that can be used. Here the problem itself is discussed first, followed by a reference to a number of available estimators.

First of all it is useful to write the polynomial model in matrix form. The polynomials are formed by a sum of products of coefficients $\boldsymbol{\mathcal{C}}$ and basis polynomials $\boldsymbol{\mathcal{B}}$. To allow for matrix multiplication of the basis polynomials with the complete set of B-coefficients, the latter is structured as a column vector and the basis polynomials are collected in a matrix B, with a row per data point. If there are multiple domain elements, the row vectors of basis polynomials $\boldsymbol{\mathcal{B}}$ are padded with zeros to exclude influences from other domain elements.

$$\boldsymbol{\mathcal{B}}^d_\lambda(\boldsymbol{\beta}(\boldsymbol{x})) := \begin{cases} \mathcal{B}^d_\lambda(\boldsymbol{\beta}^{\Gamma_i}(\boldsymbol{x})) & \text{if } \boldsymbol{x} \in \Gamma_i \\ 0 & \text{otherwise} \end{cases} \tag{5-1}$$

The notation $\boldsymbol{\beta}^{\Gamma_i}(\boldsymbol{x})$ signifies the barycentric coordinates of a point $\boldsymbol{x}$ with respect to the multiplex Γ_i.

The smoothness conditions can be collected in a similar fashion in the smoothness matrix H, such that we require $\mathrm{H}\boldsymbol{\mathcal{C}} = \mathbf{0}$. The state values are collected in a matrix X such that each row represents a data point, and the measured output is collected in a column vector $\boldsymbol{y}$. Then the constrained least squares problem can be formulated.

$$\begin{aligned} &\min ||\boldsymbol{y} - \mathrm{B}(\boldsymbol{\beta}(\mathrm{X}))\boldsymbol{\mathcal{C}}|| \\ &\text{subject to: } \mathrm{H}\boldsymbol{\mathcal{C}} = \mathbf{0} \end{aligned} \tag{5-2}$$

This problem is exactly equal to that which is solved for simplex spline estimation (see equation 3-7 in subsection 3-1-1). Therefore the same estimators can be used.

To solve problem (5-2) a generalized least squares approach has been proposed [17, 16]. This technique uses Lagrange multipliers to guarantee constraints are complied with. For larger problems the computational load can be reduced by using a recursive least squares estimator [16]. In the current research this estimator was found to converge very fast in most cases.

Preliminary results of other research have shown the possibility of using distributed solvers for finding the B-coefficients of simplex splines. In such solvers the triangulation is split up in several parts for which the optimization problem is solved individually. In general these parts will be combined afterwards by applying recursive corrections based on continuity conditions between the parts. Because of the similarity between the simplex and multiplex spline, these methods can directly be applied to the latter spline as well.

5-2 Verification on test functions

Before starting the actual system identification phase of the DelFly, a verification process is undertaken. The goal of this process is twofold. On the one hand the code should be tested for mistakes. An example of such a test is fitting a lower degree polynomial with a spline. In theory the spline should be able to produce a perfect fit. Several of these tests were performed, which ultimately lead to the conclusion that the code for generating arbitrary degree splines with arbitrary continuity order in any number of dimensions was working properly. The only constraint that was found is that layers should preferably be less than 4-dimensional. This is due to an error in the Delaunay triangulation code of Matlab, which is unable to properly align triangulated 4-cubes.

On the other hand the performance of the multiplex spline should be tested in a controlled environment. It is especially interesting to compare this performance with existing methods, the most obvious being the simplex spline. For this purpose several different test functions have been developed. In subsection 5-2-1 a four dimensional function is defined with which different layer structures can be compared. In subsection 5-2-2 the aim is to investigate the behavior of the multiplex spline when data is not uniformly distributed. In all cases a ordinary least squares estimation technique was used to estimate the B-coefficients.

5-2-1 Functions with uniform data distribution

To compare the approximation power of the multiplex spline with that of the simplex spline, the four-dimensional test function (5-3) was constructed to generate uniformly, randomly distributed data.

$$y = x_3^2 \sin(1.4\pi x_4)\mathrm{F}(2x_1 - 1, 2x_2 - 1), \qquad \forall i : x_i \in [0, 1] \tag{5-3}$$

In the test function, F(x, y) is the Franke function. No noise was added to the data sets. The function was specifically designed to create directional complexity. That is, the non-linear character differs per variable or group of variables. This allows for investigating the effects of different layer structures on the overall performance of the multiplex spline. In all tests a data set of 5000 uniformly distributed points was used for fitting the spline and a separate set of 1000 points was used for verification.

A large variety of simplex and multiplex splines was used to fit the generated data set. They are compared in Figure 5-2 based on the number of B-coefficients, the number of degrees of freedom (DoF) and the root mean square error on the verification data set (RMSE). By plotting the DoF against the RMSE a measure is found for the approximation power on the test function. The plot of number of B-coefficients versus the RMSE provides a measure of efficiency in terms of memory required on a computer. The splines used for these plots are given in Table 5-1. Each spline was tested (fitted and verified) on 10 different randomly generated data sets, after which the average RMSE value was stored. In all cases the standard deviation of the RMSE value was in the range $[2, 3] \times 10^{-3}$.

The first observation based on this set of splines is that there are a lot of options in constructing a model. Apart from splitting up the variables in any desirable way, the degrees

and continuity orders can be chosen independently in the resulting layers. Note that as a result, the presented plots are far from complete. For example, the (3,1)-multiplex spline is not considered at all in the current discussion. On top of that, the distribution of the variables over the layers is kept constant for each layer structure. In total there are 15 ways of distributing the variables over the layers (1 for the 4-simplex, 4 for the (3,1)-multiplex, 3 for the (2,2)-multiplex, 6 for the (2,1,1)-multiplex and 1 for the 4-cube); only four of those are considered here. Finally there is of course the possibility to split the domain of any variable (only considering generic type-I triangulations) and the choice of degrees and continuity orders. The presented data for the simplex spline can however be considered rather complete. Because simplices are strongly intertwined, the DoF is hard to increase by simply adding simplices. Note that due to a bug in the Matlab code for Delaunay triangulations, a triangulation of more than 22 simplices may not be perfectly aligned, leaving some shared faces free of continuity constraints.

The layers were chosen based on the knowledge of the test function. Because x_1 and x_2 together define a Franke function, it is reasonable to keep them together. The quadratic function in x_3 on the other hand justifies isolating this variable from the others. Comparing the performance of the (2,1,1)- and the (2,2)-multiplex splines does give some indications that this effect indeed plays a role. The results are however not conclusive in this respect. On the other hand there is a clear improvement when the degree in x_3 is lowered from 3 to 2 (compare for example spline M1.1 and M1.2).

Some of the performance curves in Figure 5-2a have a clear parabolic nature. If the DoF is too low, not all nonlinear effects can be modeled. On the other hand, when the DoF is too high, overfitting occurs. This is especially visible in the 4-cube spline. The 4-cube splines that deviate most from the parabolic trend, M3.2, M3.7 and M3.10, all have cubic polynomials to model the quadratic effect of x_3. The other outlier, M3.5, probably has a too high degree in the first two layers. This result underlines the benefit of choosing the proper degree in each layer. The simplex spline also shows a parabolic shape, but it has higher RMSE values than the cube spline over the entire range. For the other multiplex splines the parabolic shape is harder to identify. Splines M1.2 up to M1.5 lie slightly above the optimal curve because of the cubic polynomial in the second layer, which explains part of the deviation from a standard approximation performance curve. Splines M1.6, M1.8, M1.9 and M1.12 use a quartic equation to model the sine in x_4, which also seems sub-optimal.

In the efficiency plot of Figure 5-2b another benefit of the multiplex spline becomes evident. It presents the user with a very large amount of small spline models with adequate performance. Compare for example spline S1 and S2 with M1.1 to M1.5 and M3.1 up to M3.5. The multiplex splines present many more options, most of which have significantly different performance, whereas the simplex spline of cubic nature is much more limited. Lowering the degree of the simplex spline may be an option, but it is questionable whether it would result in desirable RMSE values. Note that there are several multiplex splines with fewer than 500 B-coefficients that perform better than the smallest simplex splines.

The final observation is one of geometric nature. By comparing Figures 5-2a and 5-2b a correlation can be deduced between the number of B-coefficients and the DoF depending on the layer distribution. The simplex splines all have limited DoF, but at the same time use a large number of B-coefficients. For the cube spline, the opposite is true. As a general rule the DoF per B-coefficient increases when more low dimensional layers are used. That is, the

simplex and (2,2)-multiplex spline use many B-coefficients, but get few degrees of freedom in return. The (2,1,1)-multiplex shows a uniform spread over both plots and the cube spline requires few coefficients for large DoF. This effect can be illustrated by closely examining the continuity conditions. Imagine having a cube, a prism (or (2,1)-multiplex) and a tetrahedron, all 3 centimeters high. Now if they are all set in a centimeter of water, the tetrahedron will, of the three, have the largest portion of its volume underwater. The same holds for continuity conditions. If the three polyhedrons contain a cubic polynomial (that is, their sides are split in three equally long parts) and one edge is connected to another polyhedron of the same type, the first order continuity conditions will constrain all coefficients on, and one centimeter above the shared edge. This leaves the fewest free coefficients in the tetrahedron, and the most in the cube.

5-2-2 Non-uniform data distribution

There is one major elements lacking from the discussion in the previous subsection: the distribution of data over the domain. It was assumed that the data was spread uniformly, but this is in reality hardly ever the case. Therefore it is interesting to test the effects of different data distributions on the performance of simplex and multiplex splines. In this subsection such an investigation is performed in two separate parts. First the performance of cubic two-dimensional multiplex splines is derived for several shapes of data sets. Then a single isolated domain element is studied in which the data lies in the inscribed circle. This is done only for the simplex and the hypercube, to limit the scope.

Data set shapes

To test the performance in interpolation and extrapolation of the splines, a group of data sets is generated with different shapes. Data is distributed randomly over the shape in all cases. The function that is approximated simply consists of a sine in one direction and an exponential function in the other.

$$y = \sin(2\pi x_1) + e^{x_2} \qquad x_1, x_2 \in [0, 1] \tag{5-4}$$

Note that the sine is more challenging to approximate than the exponential. The function is only two-dimensional to simplify the definition of the data sets.

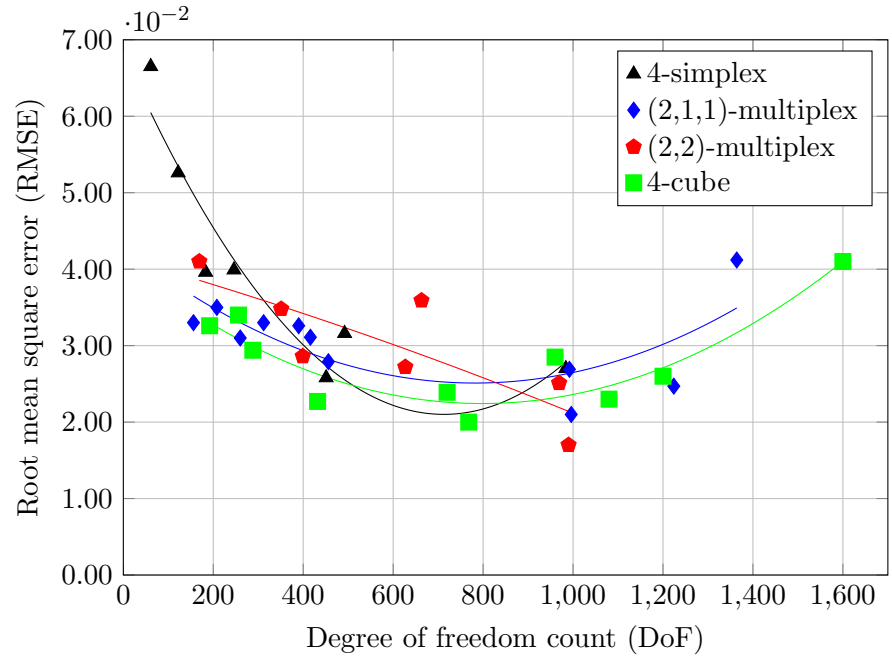
The function is approximated over the indicated domain. This is done by a multiplex spline consisting of a single (1,1)-multiplex and a simplex spline on a pair of simplices described by the vertices $\{[0,0],[1,0],[0,1]\}$ and $\{[0,1],[1,0],[1,1]\}$. If the continuity order in the simplex spline is kept at 0, both splines have equal amounts of degrees of freedom for equal polynomial degrees.

To test the effects of different data distributions, the following group of data set shapes is used.

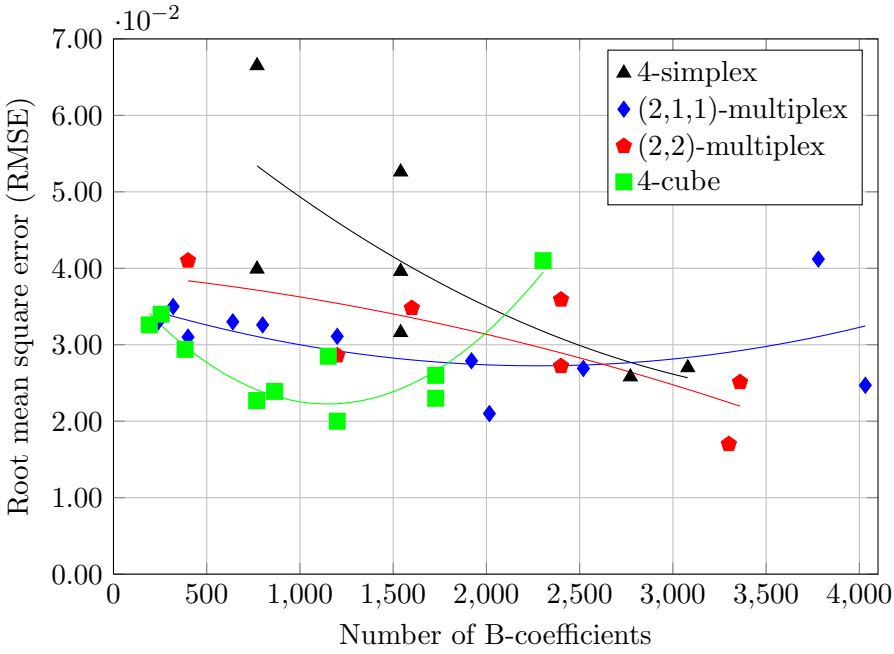
U: A uniform, square shape covering the whole domain, as a benchmark;

Table 5-1: Performance of several simplex and multiplex splines on the test function (5-3) with a uniform data distribution. The 'Tessellation' column indicates the amount of simplices in each layer. The product of this vector is the total amount of multiplices. The number of degrees of freedom (DoF) is the difference between the number of coefficients and the rank of the smoothness matrix. The root mean square error (RMSE) is calculated based on a separate 1000 point data set. Note that the RMSE-values are absolute and function values lie in the domain [-1,1.4].

ID	Dimension	Degree	Continuity	Tessellation	B-coeff.	DoF	RMSE
S1	4	3	1	22	770	61	0.067
S2	4	3	1	44	1540	122	0.053
S3	4	4	1	22	1540	183	0.040
S4	4	3	0	22	770	246	0.040
S5	4	5	1	22	2772	451	0.026
S6	4	3	0	44	1540	492	0.032
S7	4	3	0	88	3080	984	0.027
M1.1	(2,1,1)	(3,2,3)	(1,1,1)	(2,1,1)	240	156	0.033
M1.2	(2,1,1)	(3,3,3)	(1,1,1)	(2,1,1)	320	208	0.035
M1.3	(2,1,1)	(3,3,4)	(1,1,1)	(2,1,1)	400	260	0.031
M1.4	(2,1,1)	(3,3,3)	(1,1,1)	(2,1,2)	640	312	0.033
M1.5	(2,1,1)	(3,3,4)	(1,1,1)	(2,2,1)	800	390	0.033
M1.6	(2,1,1)	(3,2,4)	(1,1,1)	(2,2,2)	1200	416	0.031
M1.7	(2,1,1)	(3,2,3)	(1,1,1)	(4,2,2)	1920	456	0.028
M1.8	(2,1,1)	(4,2,4)	(1,1,1)	(2,2,2)	1800	672	0.029
M1.9	(2,1,1)	(5,2,4)	(1,1,1)	(2,2,2)	2520	992	0.027
M1.10	(2,1,1)	(5,2,3)	(1,1,1)	(8,1,1)	2016	996	0.021
M1.11	(2,1,1)	(5,2,3)	(1,1,1)	(4,2,2)	4032	1224	0.025
M1.12	(2,1,1)	(5,2,4)	(1,1,1)	(2,2,3)	3780	1364	0.041
M2.1	(2,2)	(3,3)	(1,1)	(2,2)	400	169	0.041
M2.2	(2,2)	(3,3)	(1,1)	(8,2)	1600	351	0.035
M2.3	(2,2)	(4,3)	(1,1)	(2,4)	1200	399	0.029
M2.4	(2,2)	(4,3)	(1,1)	(4,4)	2400	627	0.027
M2.5	(2,2)	(4,3)	(1,1)	(8,2)	2400	663	0.036
M2.6	(2,2)	(5,3)	(1,1)	(4,4)	3360	969	0.025
M2.7	(2,2)	(4,3)	(1,1)	(8,4)	3300	990	0.017
M3.1	(1,1,1,1)	(3,3,2,3)	(1,1,1,1)	(1,1,1,1)	192	192	0.033
M3.2	(1,1,1,1)	(3,3,3,3)	(1,1,1,1)	(1,1,1,1)	256	256	0.034
M3.3	(1,1,1,1)	(3,3,2,3)	(1,1,1,1)	(2,1,1,1)	384	288	0.029
M3.4	(1,1,1,1)	(3,3,2,3)	(1,1,1,1)	(2,2,1,1)	768	432	0.023
M3.5	(1,1,1,1)	(5,5,2,3)	(1,1,1,1)	(2,1,1,1)	864	720	0.024
M3.6	(1,1,1,1)	(4,4,2,3)	(1,1,1,1)	(2,2,1,1)	1200	768	0.020
M3.7	(1,1,1,1)	(5,5,3,3)	(1,1,1,1)	(2,1,1,1)	1152	960	0.029
M3.8	(1,1,1,1)	(5,5,2,3)	(1,1,1,1)	(2,1,1,2)	1728	1080	0.023
M3.9	(1,1,1,1)	(5,5,2,3)	(1,1,1,1)	(2,2,1,1)	1728	1200	0.026
M3.10	(1,1,1,1)	(5,5,3,3)	(1,1,1,1)	(2,2,1,1)	2304	1600	0.041



(a) Approximation power



(b) Memory efficiency

Figure 5-2: Plots showing the approximation power and efficiency for the splines presented in Table 5-1, as well as a quadratic trend line for each layer structure. The splines on 4-cubes clearly show the familiar performance curve with an optimum at spline M3.6. At lower DoF the approximation power is insufficient, at higher DoF overfitting occurs. Note that the RMSE-values are absolute and function values lie in the domain [-1,1.4].

D: A diamond, or a rectangle rotated over 45 degrees, positioned such that the vertices lie halfway on the sides of the domain;

C: A circle, touching the sides of the domain;

R: A ring with equal outer radius as the circle and inner radius equal to half the outer radius;

T: A lower triangular shape, filling the region spanned by vertices [0,0], [0,1] and [1,1];

NX: Not X, with X any of the above (except for U).

Note that shape T is oriented along another diagonal of the domain than the triangulation. This shape will thus fill the domain elements halfway, irrespective of the type of spline used. All data sets will consists of approximately 1000 data points to make sure there are no differences due to nonuniform distributions over the chosen shape.

The first observation that can be made in Table 5-2 is that the test function does not favor one of the splines. This can be deduced from the comparison of MU and SU. They show similar performance over the entire domain, which implies that differences between simplex and multiplex spline are purely caused by the definition of those splines.

Secondly, all splines but SU and MU show a significantly better performance inside the data set shape than outside. This can be read from Figure 5-3 by considering the angle from the x-axis at which the data points are located. The ratio between the RMSE-values is farthest from unity in the multiplex splines for all shapes except T and the inverse shapes NX. The inverse shapes show inconclusive results in this respect. The difference in performance ratio between the two types of splines is twofold. On the one hand the performance of the multiplex spline inside the shape is often better. This was observed before and is confirmed by the data in Figure 5-3. On the other hand the performance of the multiplex spline is worse than that of the simplex spline in the rest of the domain. The cause of these two effects is probably the same: the effective degree of the multiplex spline is higher. This increases approximation power, but also the risk of running into Runge's phenomenon. This latter effect is also clear from the maximum error (ME) in table 5-2.

Now it is interesting to focus on the difference between the normal shapes and their inverse counterparts. In the case of T and NT, the two data sets are equivalent, which shows in the RMSE values. For the other shapes however, there are significant differences visible in Figure 5-3. For both D and R the splines on the inverse shape perform significantly better outside the shape. This implies that it is essential to have data available near the edges. Inside the inverse shape the performance is generally worse, which implies that the approximation power is concentrated near the center of the domain. The cause for the large difference between D and C is uncertain. It may lie in the fact that the diamond cuts parallel to lines in the B-net, but this was not investigated further.

Finally it should be noted that NR consists of two parts. Therefore it is interesting to check the performance on these two parts when the data set R is used for estimating the B-coefficients. In that case we find for SR 0.0157 ± 0.0011 and 0.0360 ± 0.0064 inside and outside the ring respectively. This confirms the idea that spline models are only valid inside the convex hull of the data set [16]. In MR this effect is even stronger, as we find RMSE-values of 0.0100 ± 0.0006

and 0.0671 ± 0.0111 respectively. That is, the performance inside the convex hull is better, whereas outside the convex hull it is worse. It is very likely that this is again caused by the difference in total degree.

Isolated domain elements

From the discussions above and in subsection 5-2-1 two major differences between the multiplex and simplex spline are found to cause differences in performance. The first is the shape of the domain elements, and especially the distribution of content over these shapes. The second is the total degree, which is higher in the multiplex spline. To find out what effect is dominant under what circumstances, the following test is set up. A comparison is made between the n-simplex and the n-cube spline by performing an unconstrained optimization based on a data set defined inside the inscribed sphere of a single isolated domain element. For this purpose a test function is defined as a sine of the radius.

$$f(r) = \sin\left(1.4\pi \frac{r}{R}\right) \tag{5-5}$$

No noise was added. In the test function the R signifies the radius of the inscribed sphere and r the distance from the incenter. Because regular polytopes are used, the incenter coincides with the barycenter. In all cases a cubic spline is used to approximate the sine. The amount of data points for fitting and verification depends on the dimension n and is set to 5×10^n

The performance of the splines for the dimensions 2 up to 5 are given in Table 5-3. As the dimension increases the number of B-coefficients goes up, which results in higher approximation power in both splines. The performance inside the insphere is generally up to standards (considering the maximum function value of one), but outside there is no approximation at all.

The dominance of the two effects mentioned before can be derived from a comparison between the simplex and multiplex splines' performance outside the insphere. From the start the multiplex spline seems to perform better in this region of the domain. This implies that at first the difference in geometric basis is dominant in the extrapolation performance. This is rather surprising, because the difference in content ratio are not too grave at low dimensions, as depicted in Figure 5-1. It may however be that the distance from the barycenter to the vertices is more relevant in this respect. The difference in this distance may be a lot greater. In five dimensions the multiplex spline performance suddenly deteriorates and errors become very large. This is typical for the Runge phenomenon, which could have been expected for a polynomial degree of 15 along the diagonal. That is, we may expect that at high dimensions the negative effect of total degree will overrule the positive effect of the geometric basis of the multiplex spline. The breaking point will however depend on the degrees and dimensions chosen in the layers.

5-3 System identification algorithm

From section 5-2 a set of directives and constraints for multiplex spline construction can be deduced. In short, variables with different degrees of nonlinearity should be in separate

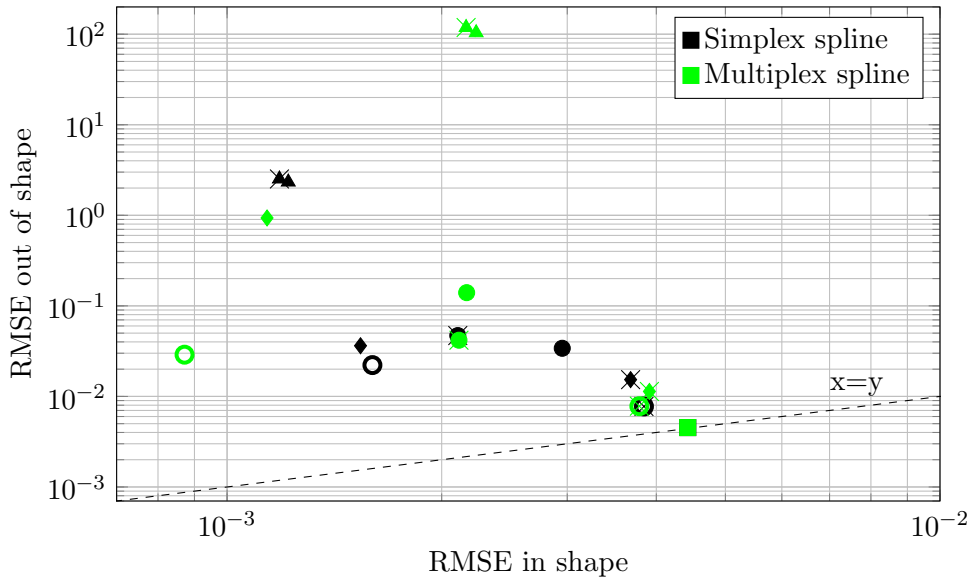


Figure 5-3: The performance of the cubic 2-simplex spline and bicubic (1,1)-multiplex spline on different shapes of data sets. Note that the shape of the marker corresponds to the shape of the data set. A marker with a cross ($\times$) signifies the inverse data set (N).

Table 5-2: The performance in terms of the root mean square error of several simplex and multiplex splines on non-uniformly distributed data. A single multiplex is considered and the continuity order in the simplex spline is always 0. Because the dimensions are equal for all splines, as well as the degree (cubic), they are not provided in the table. In the RMSE values a subdivision is made regarding data inside and outside the data shape used for fitting. ME stands for (absolute) maximum error. Each table entry consists of the mean $\pm$ the standard deviation measured over 10 tests with different random data sets.

ID	Data set	RMSE (in), 10^{-2}	RMSE (out), 10^{-2}	ME
SU	U	0.443±0.015	-	0.223±0.018
SD	D	0.154±0.017	3.63±0.748	0.607±0.155
SC	C	0.295±0.016	3.40±0.482	0.619±0.074
SR	R	0.160±0.011	2.22±0.177	0.699±0.068
ST	T	0.122±0.014	233±37.0	6.58±0.589
SND	ND	0.368±0.021	1.53±0.088	0.223±0.005
SNC	NC	0.211±0.019	4.70±0.504	0.436±0.021
SNR	NR	0.384±0.021	0.775±0.065	0.149±0.007
SNT	NT	0.118±0.023	252±44.8	6.46±0.382
MU	U	0.443±0.016	-	0.230±0.020
MD	D	0.114±0.017	93.5±13.8	4.42±0.483
MC	C	0.217±0.008	14.0±1.67	1.25±0.103
MR	R	0.087±0.006	2.89±0.267	0.865±0.079
MT	T	0.224±0.023	103±18.7$\times 10^2$	50.2±5.73
MND	ND	0.391±0.022	1.13±0.073	0.185±0.007
MNC	NC	0.211±0.020	4.18±0.321	0.385±0.009
MNR	NR	0.379±0.023	0.780±0.067	0.152±0.010
MNT	NT	0.216±0.025	118±18.3$\times 10^2$	51.2±3.13

Table 5-3: Performance characteristics of the single domain elements. All polynomials are cubic. The RMSE is split in a part inside and a part outside the inscribed sphere. Note that the RMSE inside the insphere is scaled, whereas the RMSE outside of the insphere is not. The RMSE-values are absolute and consist of a mean and standard deviation over 10 runs with different random data sets. Function values range from -1 to 1.

ID	Dimension	RMSE (in), 10^{-2}	RMSE (out)
S2	2	2.71±0.84	10.6±1.75
S3	3	1.04±0.15	27.9±2.32
S4	4	0.787±0.056	52.2±1.75
S5	5	0.729±0.042	84.7±2.72
M2	(1,1)	3.01±0.60	1.37±0.149
M3	(1,1,1)	1.10±0.06	3.06±0.23
M4	(1,1,1,1)	0.782±0.017	12.3±7.24
M5	(1,1,1,1,1)	0.688±0.012	595±178

layers and the total degree should not become too high. Also, a multiplex with many low-dimensional layers is easier to uniformly fill with data. With these rules in mind, a variable driven system identification algorithm can be constructed. In this algorithm a tessellation is defined by triangulations is separate layers. Another type of system identification algorithm can be found in the recommendations in chapter 8, because it has not been defined in detail yet.

The algorithm for constructing a multiplex spline is presented schematically in Figure 5-4. The individual steps, including some that are not specifically presented in the figure, are described in detail below.

Layer selection

This first step is conjoined with data projection in Figure 5-4, but treated separately here. The process of selecting the layers is not trivial when little knowledge is available about the system under consideration. For the DelFly, studied in chapter 6, this is definitely the case. When layers cannot be chosen based on expert knowledge, a proper data analysis may provide useful insights into variable dependencies. As a last resort the user can decide to generate a large set of splines and derive the desired dependencies based on that. Both these strategies were employed in the system identification of the DelFly.

During this step it is important to keep an eye on the total dimension and the number of layers. When the number of layers increases, so does the total degree. This may cause unwanted effects, as discussed in section 5-2. The total dimension of the tessellation has direct effects on the spread of the data. If the data becomes too scattered, it becomes harder to include data points in each multiplex.

Two-dimensional layers are generally considered favorable by the author, because they are very intuitive and can easily be constructed by hand and inspected with the naked eye.

Data projection

The step of data projection is an automated process, but essential as a link between choosing layers and triangulating them. One should always keep in mind that a lot of information is lost when high-dimensional data is projected onto lower-dimensional spaces. This is especially a problem when one-dimensional layers are used. The complete knowledge of the shape of the data set is lost in the process.

In this step it is useful to incorporate some methods proposed in subsection 3-3-2. One may include color coding to show the local data density, or the variance in the variables that are not plotted. In this work only the first method is used.

Triangulation

The third step in the process of this variable driven system identification is triangulating the layers. Any method of triangulating a domain can be applied. Optimization algorithms can be employed, as well as standard triangulations of polytopes encompassing the domain. If layers are one up to three-dimensional, triangulations can also be generated by hand.

The number of simplices in each layer should be kept low. In the combination process all possible combinations of simplices from the different layers will be used, resulting in as many multiplices as the product of the number of simplices in each layer. This number may rise quickly when several layers are used with dense triangulations.

Combination into tessellation

Just like data projection, the combination of triangulations is an automated step. Every combination of simplices from each layer will result in a multiplex. The multiplices are tested on data content and the empty ones are removed.

Spline definition and fitting

Finally a spline can be defined on the tessellation. As the degree and continuity order are chosen per layer, this step can also be performed before combining the triangulations. The same holds for the construction of the smoothness matrix. The fitting process is described in subsection 5-1-3. The B-coefficients are estimated using a part of the available data, the other part is used for model quality assessment.

5-4 Quadrotor modeling

To verify the system identification algorithm using the multiplex spline it is applied to a data set of quadrotor thrust generated using an analytic model. This platform is chosen because it is well-understood and analytic models are widely available.

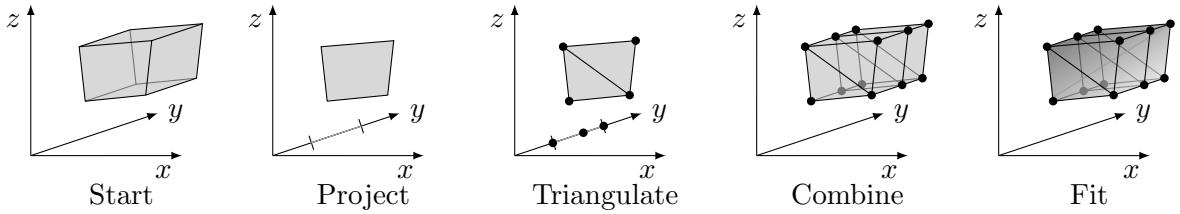


Figure 5-4: A schematic overview of the variable driven system identification algorithm. The data is indicated as a transparent box in the first figure. After projecting the data onto the layers, triangulations are made and combined. Finally a polynomial degree and continuity order are chosen and a spline model is fitted.

5-4-1 Analytic model

The analytic model used to generate a data set is mostly based on helicopter theory that has been applied to quadrotors before [26, 43]. The core of the model is the ideal thrust

$$T = \frac{\eta P}{v_\infty \sin\alpha + v_i} \tag{5-6}$$

P signifies the electric engine power input, which is converted to aerodynamic power using a single efficiency $\eta = 0.7$. For the electric power we use [26]

$$P = \frac{U - K_{emf}\omega}{R_a} U \tag{5-7}$$

In this model $K_{emf} = 0.005$ is a constant related to the back EMF, namely the product of the number of windings, the area of the coil and strength of the magnetic field. The value was estimated by assuming logical values for these metrics. The resistance of the armament R_a is assumed to be 0.1Ω.

The induced velocity v_i is given by the well-known equation

$$v_i = \frac{v_h^2}{\sqrt{(v_\infty \cos\alpha)^2 + (v_\infty \sin\alpha + v_i)^2}} \tag{5-8}$$

where v_h is the induced velocity in hover that is required to support one quarter of the assumed weight of the quadrotor. In the range of vertical velocity $-2v_h \leq v_z < 0$, where $v_z = v_\infty \sin\alpha$, it is assumed that the rotor is in the vortex ring state. Its induced velocity is then approximated by [32]

$$v_i = v_h \left(k_0 + k_1 \frac{v_z}{v_h} + k_2 \left(\frac{v_z}{v_h}\right)^2 + k_3 \left(\frac{v_z}{v_h}\right)^3 + k_4 \left(\frac{v_z}{v_h}\right)^4 \right) \tag{5-9}$$

with $k_0 = 1, k_1 = -1.125, k_2 = -1.372, k_3 = -1.718$, and $k_4 = -0.655$.

The data set is generated using a set of 5000 random inputs within the operating range of the motor. Pseudo-random noise bounded to 1% of the maximum thrust is added to the simulated output.

5-4-2 Spline model set-up

From equations (5-6) up to (5-9) it can be found that the state consists of v_∞, α, ω and U. The spline models will thus be four-dimensional. Also, there is a clear subdivision of the model into two parts. The first is the engine part in equation (5-7), which is a quadratic function of input voltage U and rotational rate ω. The second part in equations (5-8) and (5-9) consists of quartic equations describing rotor inflow.

The design of the simplex spline is rather limited. With the aim of keeping both models simple, it is wise to choose a minimum triangulation of the four-dimensional domain. This type-I triangulation already comprises 22 simplices. The degree that is required is dictated by the highest degree of the analytic model. In this case that is four, meaning the entire spline will be quartic. Even though most of the analytic model is second order smooth in all directions, a continuity order of 1 is chosen to more easily compare the results with the multiplex spline. The resulting simplex spline has 1540 B-coefficients, of which 183 are free. Clearly there is a large amount of constraints, which may limit the approximation power.

The design of the multiplex spline presents a lot more flexibility and choices. Because of the experience gathered in the previous subsection, it seems reasonable to collect variables of similar nonlinearity in layers. In this case this means that the engine model and the inflow model can be separated. By combining v_∞ and α in the first layer and ω and U in the second, the analytic model can be approximated perfectly. In the first layer a quartic polynomial is required, whereas in the second layer a quadratic polynomial will suffice. First order continuity is imposed in both layers. From early iterations it was found that extra approximation power was required in angle of attack, so the domain of that variable was split in two before generating a minimal type-I triangulation in both layers. The tessellation thus consists of 8 (2,2)-multiplices, together containing 1440 B-coefficients of which 297 are free.

5-4-3 Spline model comparison

The goal of this exercise was to construct a valid, efficient model based on a simple analytic model. The validity can as before be tested using standard statistical measures. The efficiency however is harder to quantify. The number of B-coefficients, the number of constraints and the general ease of use are relevant as well as the overall performance. Therefore the performance of the models is discussed first.

To measure the quality of the fit, the relative root mean square error (RRMS, RMS of the error divided by RMS of the data), the mean square error (MSE), and maximum error (ME) are used. The results are presented in table 5-4. Note that due to the nature of the model, it is possible to generate a verification data set that does not contain noise. The RRMS and MSE values are given for both data sets. In the first place these figures show that no significant overfitting to the noise in the modeling data takes place, since the errors are smaller for the crisp verification data set. Most important is that in every metric the multiplex spline model performs better than the simplex spline.

The differences between the two splines become clear when looking at the error metrics. The relative root mean square error shows that both splines have a small error relative to the system dynamics and noise. The multiplex spline however is almost three times as accurate in predicting system outcome as the simplex spline. The mean square error for the simplex

Table 5-4: The relative root mean square error (RRMS), mean square error (MSE, normalized with maximum thrust), and maximum error (ME, in percent of maximum thrust) for both the modeling data set (with noise) and the verification data set. The differences between the two data sets show that no significant overfitting to noise takes place. The overall performance of the multiplex spline model is better than that of the simplex spline model.

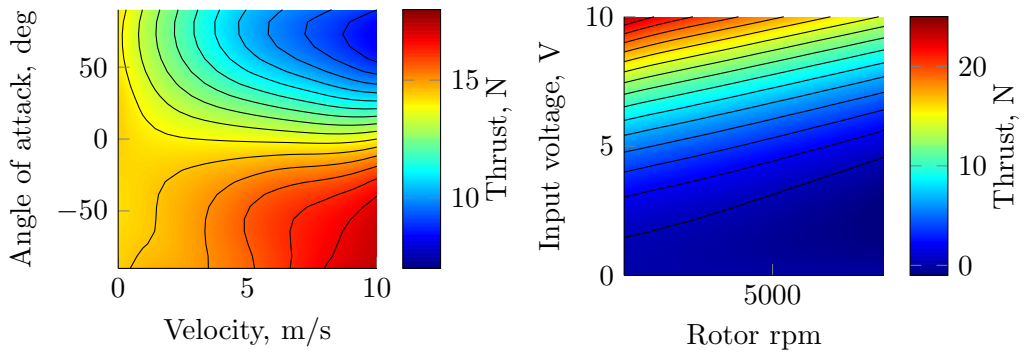
	Modeling			Verification		
	RRMS, %	MSE, 10^{-5}	ME, %	RRMS, %	MSE, 10^{-5}	ME, %
Simplex spline	2.397	5.012	4.494	1.608	2.153	3.487
Multiplex spline	1.934	3.269	1.998	0.672	0.376	1.879

spline is more than a factor 8 higher, indicating that this spline is less capable of modeling the complete dynamics in the way the multiplex spline does. The same can be observed from the difference in maximum error. One explanation for this may be that the continuity constraints in the simplex spline are not in line with the system dynamics. That is, they relate B-coefficients that lie at control points that differ in all variables, whereas the constraints in the multiplex spline lie in the chosen layers.

It is also interesting to point out the differences between the slices of the models displayed in Figure 5-5. Especially in the inflow layer it is clear that some errors are introduced in the simplex spline. The contours of the familiar inflow model are clear in Figure 5-5c, but are a lot less smooth in Figure 5-5a. Also in the engine dynamics slices, there are some non-quadratic effects visible in the upper left corner of Figure 5-5b. The multiplex spline does not suffer from such effects, as the polynomial in the engine dynamics layer is per definition quadratic.

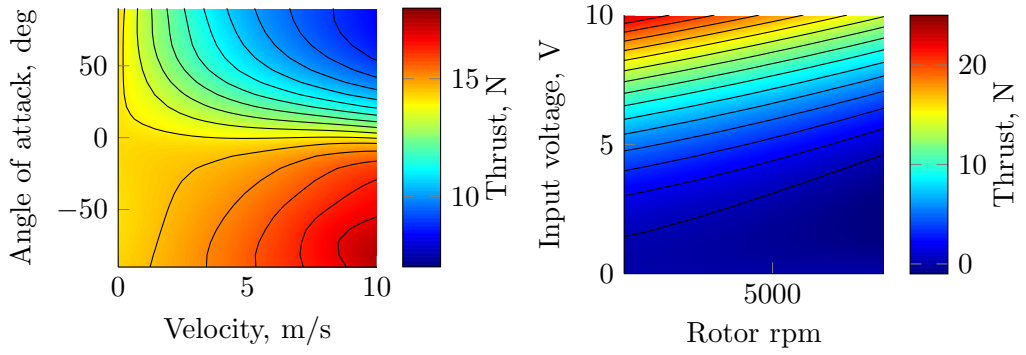
Apart from having a smaller approximation error, the multiplex spline also uses less B-coefficients, less continuity constraints and less domain elements. This saves both computation time and computer memory. Combining these metrics with the difference in approximation power, it is clear that the multiplex spline is more efficient.

The algorithm is also easy to automate. After all, because of the low dimensions of the layers, it is easy to construct a tessellation. This may even be done by hand in each layer, if the data is not uniformly distributed over a rectangular domain. This is a lot more difficult for the simplex spline, for which generally a standard triangulation of the hypercube is used.



(a) Simplex spline inflow model

(b) Simplex spline engine model



(c) Multiplex spline inflow model

(d) Multiplex spline engine model

Figure 5-5: Slices of the simplex and multiplex splines showing the inflow and engine dynamics separately. Note the deviations from the normal pattern of thrust depending on velocity and angle of attack in the simplex spline. Also in the engine model the simplex spline deviates slightly from the ideal quadratic nature.

Chapter 6

Validation on DelFly data

The goal of this chapter is twofold. First and foremost, the system identification algorithm of section 5-3 will be validated. This is done by applying it to model the Z- and X-force working on the DelFly over a range of states, especially during a maneuver. As splines were never used before to model this flapping wing MAV, this research can also be seen as a preliminary study into the feasibility of global modeling of the DelFly using splines. Therefore an extensive discussion of the available data sets is provided.

For the research described in this chapter an existing data set was used. This data set is analyzed in section 6-1-2. The focus is on determining whether the data set is fit for global system identification. In section 6-2-2 the spline models that were made based on the provided data are presented and compared to existing methods. This is done to validate the system identification algorithm of section 5-3.

6-1 DelFly flight data

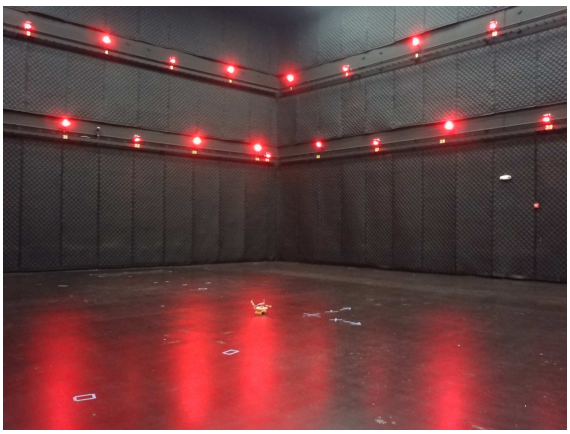
In this section the source and treatment of the DelFly data is discussed. The data set was taken from previous research projects. Therefore subsection 6-1-1 on data acquisition is kept short here. The interested reader is referred to the work of Caetano et al. for more details [8, 9, 10]. Subsection 6-1-2 on data analysis is more elaborate. The goal is to find the relevant states and the feasibility of modeling the DelFly dynamics using the data set.

6-1-1 Data acquisition

The data used in the validation phase of this research was obtained in the Micro Air Vehicles Integration and Application Institute (μAVIARI) of the US Air Force Research Laboratory. This institute has a test chamber in which a system of cameras can track a set of markers at a frequency of 200Hz and with very high precision [9]. This chamber is shown in Figure 6-1b.

The DelFly II itself is a flapping wing micro air vehicle with a span of 274 mm. It has a conventional tail section with a rudder and elevator. The DelFly is shown in Figure 6-1a

(a) DelFly with markers [9]



(b) μAVIARI test chamber [10]

Figure 6-1: The data set of the DelFly was obtained in a test chamber. The markers on the body and wings of the DelFly are tracked by the cameras in the chamber. From these position measurements, both the velocities and the attitude were derived.

with the tracking markers attached. The flight envelope consists of two major flight regimes. The first is a slow flight close to hover, in which the body of the DelFly is oriented close to vertical. The second is a fast forward flight in which the body is close to horizontal. In the current discussion only the first regime is considered. For favorable hover characteristics the center of gravity was put aft, at 83% of the chord.

Several different maneuvers are described in the work of Caetano et al. [9]. These include step, doublet and triplet inputs on the elevator. These maneuvers are especially useful in linear system identification. The data set used in this research especially contains nominal slow forward flight and step inputs on the elevator. This maneuver causes the DelFly to pitch up, stall, fall in an almost vertical orientation and then recover.

All states are derived from the positions of the markers attached to the DelFly body. The velocities and rates are calculated using a three point difference method.

A number of tests contained significant turns and other lateral effects during maneuvers. To be able to isolate longitudinal effects, these sets were removed entirely. The remaining data set contains a large amount of nominal forward flight. In order to prevent overfitting to this data, many of these data points were removed. This was done by isolating the maneuvers based on visual inspection of the flight path.

A set of measurement errors was found in which a Z-force was available but the acceleration in z-direction was zero. These data points were removed before further processing.

6-1-2 Data analysis

As pointed out in the previous subsection, the data set was originally intended for linear system identification. In this study however the data is investigated mainly based on states and inputs. The main goal in this analysis was to determine whether the set is suitable for spline approximation. That is, is the data well distributed over the state space? This investigation was limited to the body velocities u and w, pitch angle θ and rate q, and flap

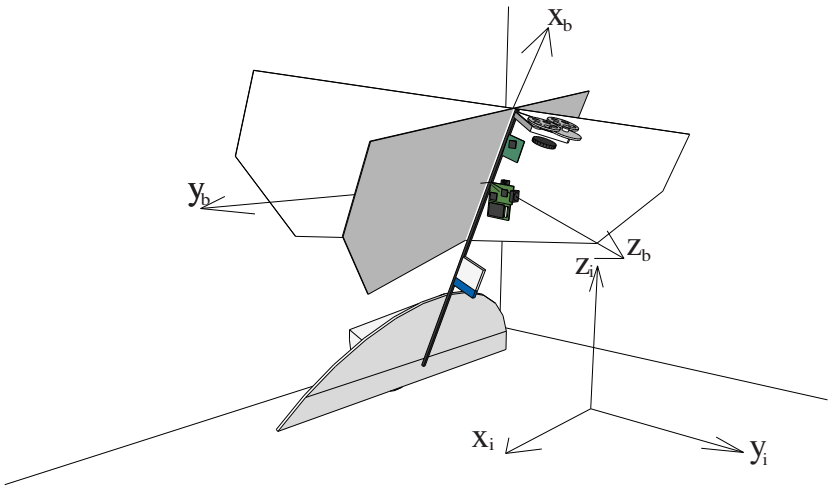


Figure 6-2: The body reference frame with respect to the inertial frame and the DelFly body [9]. Note that z_b points in the direction of flight in the low speed regime.

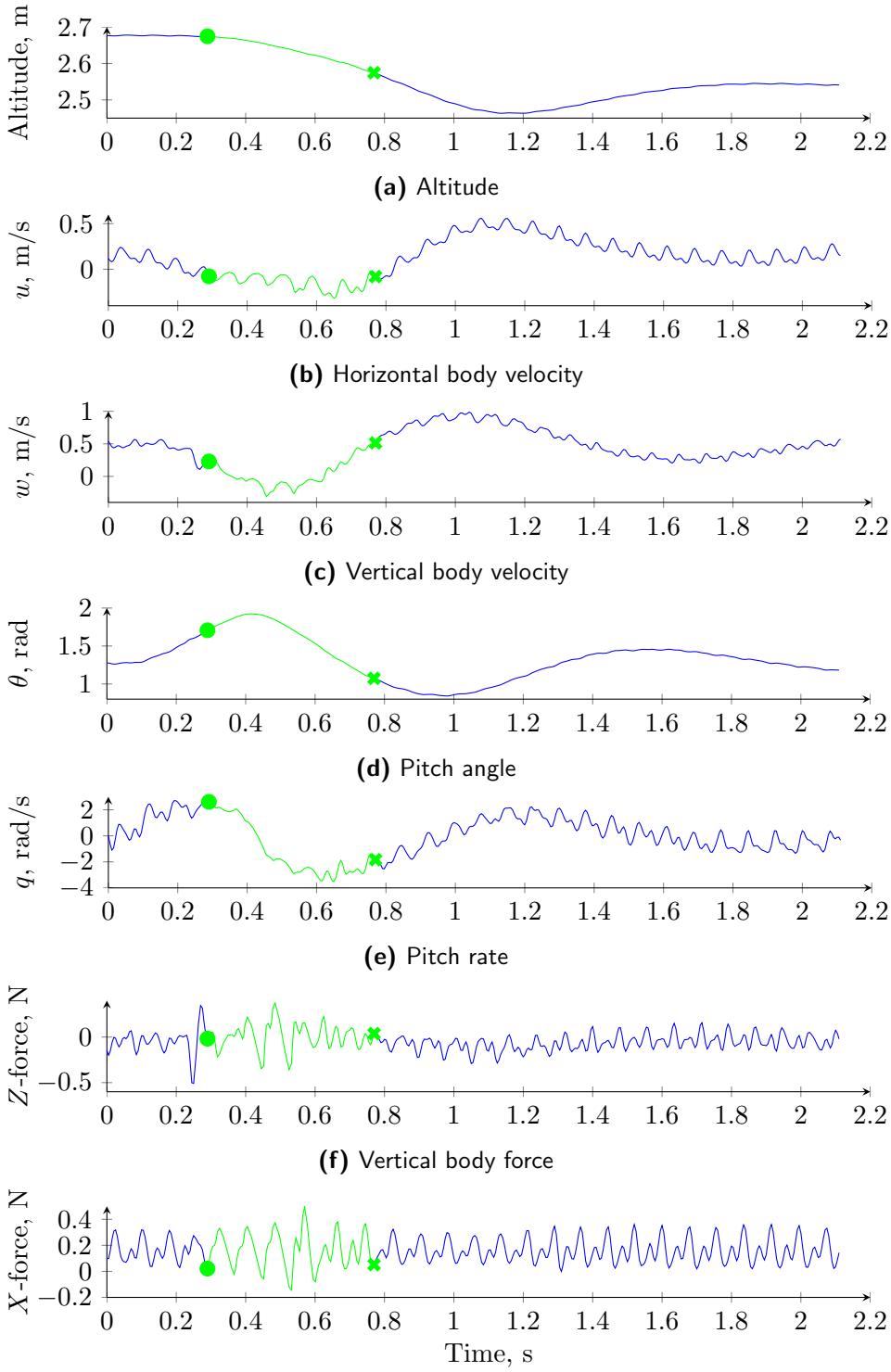
angle δ_f and rate $\dot{\delta}_f$. As outputs both the Z- and X-force were studied. Note that the velocities and forces are defined in the body frame as defined in Figure 6-2. In this subsection the set is plotted for multiple combinations of states to investigate the shape of the data set in detail. On the other hand, w is plotted against the output Z-force to investigate the presence of dynamic effects. The X-force is not treated in detail here, because of the great overlap with the discussion of Z-force. A representative group of states and outputs is plotted against time for a typical maneuver in Figure 6-3.

Shape of the data set

From an aerodynamic point of view it seems reasonable to construct three layers. In the first the velocities are combined, in the second the pitch angle and rate and in the third the flap related states. To investigate whether this yields desirable data distributions for making triangulations, and to find outliers, the data set is projected onto these candidate layers, as depicted in Figure 6-4.

The first layer considered here is the velocity layer. This is the combination of horizontal and vertical velocity in the body axes, u and w. The projection of the data set onto this layer is shown in Figure 6-4a. The data is strongly clustered around moderate velocity values. Because the slow forward flight of the DelFly is studied, it comes as no surprise that in general w is larger than u. The outliers in the range $u > 1$m/s stem from a single flight test, and are removed before generating spline models.

The second layer candidate is the pitch layer. This combines the pitch angle and rate, as plotted in Figure 6-4b. A subdivision in four regions is sketched in this plot. The regions relate to regions of other states. For example, the low pitch region P1 can be related to high velocities w. These relations are discussed in more detail in the next part on dynamic effects. Purely considering the pitch angle and rate, a few notes can be made regarding the shape of the data set. The core of the data set has a rectangular shape with a small gap between P2 and P3. The outliers can be clearly identified as single trajectories and are filtered out before



(a) Altitude

(b) Horizontal body velocity

(c) Vertical body velocity

(d) Pitch angle

(e) Pitch rate

(f) Vertical body force

(g) Horizontal body force

Figure 6-3: Time domain plots of the investigated states and output during a typical maneuver. Note that the start and end point of the (dramatized) path through Figure 6-5 are also shown.

spline fitting. Such trajectories are still visible in the left half of region P1. Overall the data seems to be properly spread over these two variables.

The last layer is that of the flapping dynamics, plotted in Figure 6-4c. A rough subdivision in different flap stages is also provided. From region F1 to F6 the stages can be described as wings open (F1), wings closing (F2 and F3), wings closed (F4) and wings opening (F5 and F6). The three remaining regions that are indicated can be directly coupled to parts of the maneuvers flown. To be more precise, when the DelFly is falling during the maneuver, the opening of the wings goes purely through regions F7 up to F9. During these phases the DelFly does not visit regions F5 and F6. It was found that this is caused by missing data. During this phase of the maneuvers the camera system is unable to track the markers continuously during the opening of the wings. As the wings are opening the system tracks the markers once every two samples. Because the flap rate is calculated using the sample frequency and not the actual time between samples, the rate is higher for these data points in Figure 6-4c. However, this effect is not purely numeric, as the same angles are measured every time. This implies that the wing oscillates with a frequency between six and eight times as high as the flap frequency during wing opening. Only when the wing foil is deflected in a certain direction, it is tracked.

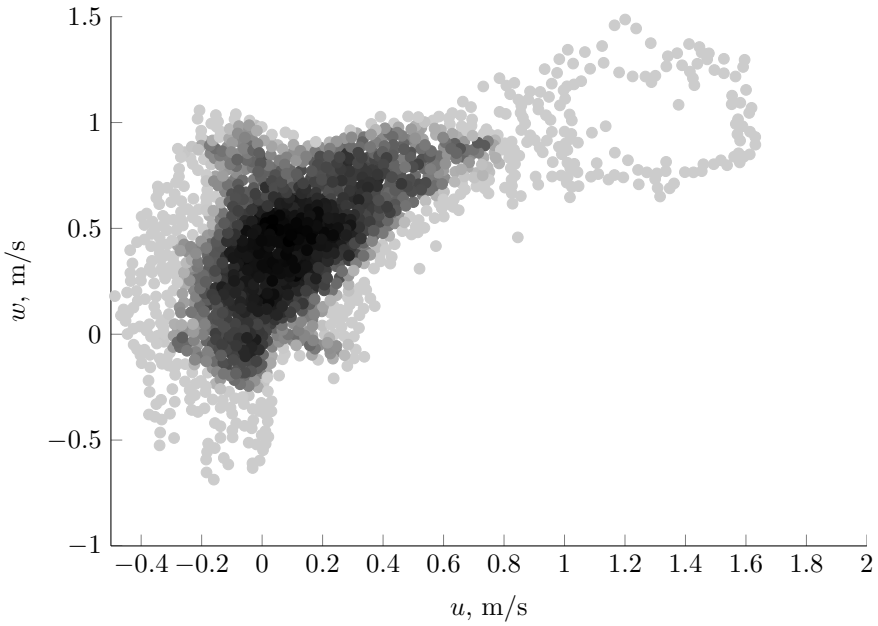
Because of the circular shape of the flap dynamics the layer can be reduced to a single variable. This is done by choosing a center at the mean of the data points (close to the point (0.6,0)) and describing the position of the data points purely by the phase angle ζ with respect to this origin. For the regions F1 up to F6 the difference between the inner and outer radius of the data ring can be considered insignificant. As pointed out above, the regions F7 up to F9 are separated from the regions F5 and F6 in due to numeric effects. These two arguments together validate neglecting the distance from the center of the circle.

Inclusion of dynamic effects

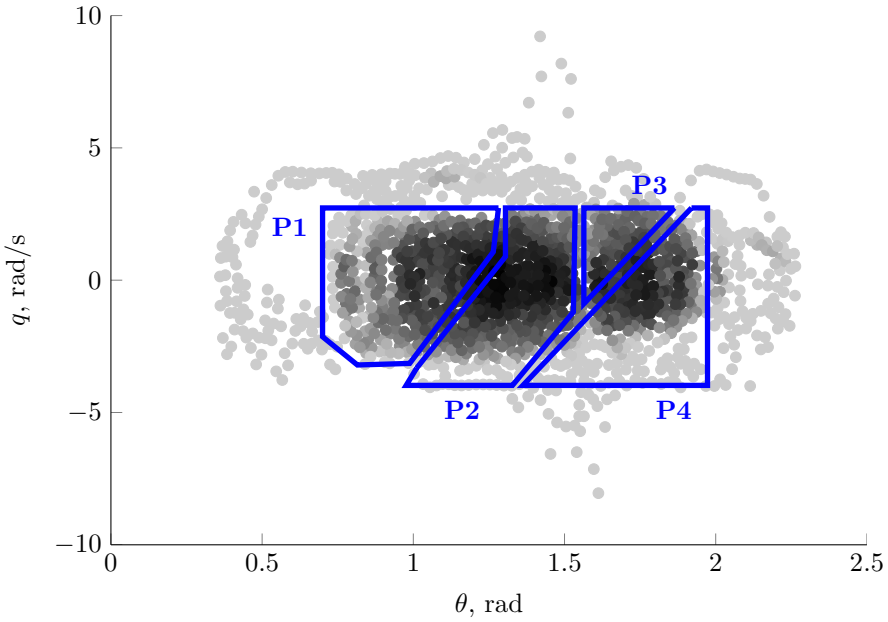
In fixed wing aircraft as well as rotorcraft, the aerodynamic forces generally depend heavily on the angle of attack. As we are studying the body axes components of the velocity, such relations can be expected to be found when instead considering the velocity w. Therefore in Figure 6-5 the projection of the data set onto the w, Z-plane is plotted. The indicated regions are discussed below. A description of a maneuver is related to the green path in the same figure.

Region Z1 in Figure 6-5 can be seen as the core of the data set. The data density is very high, leaving less than half of the data points for the other regions. This region can be subdivided in more parts, but those are strongly intertwined. Note that the vertical velocity has a nominal value, corresponding to slow forward flight. In general the data points with positive Z-force can be coupled to a high right wing angle (F1 in Figure 6-4c) and the negative force to a low wing angle (F4). The pitch angle is in the medium to high range, whereas pitch rates are relatively small (mainly P2). When velocity decreases within Z1, the pitch angle moves up into P3 and later even the upper half of P4. In terms of flapping the data points lie purely in the ring, regions F1 up to F6.

Region Z2 differs slightly from region Z1 and can be hard to identify. It is chosen based on the width of the set, which decreases for higher vertical velocities. These data points with higher vertical body velocity are clearly separated from Z1 in the pitch layer. There these

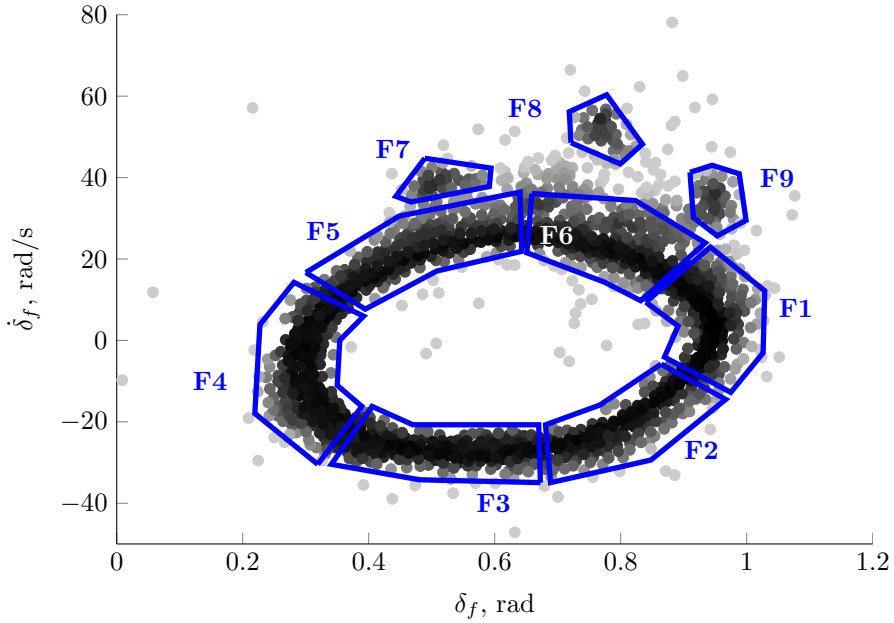


(a) Velocities



(b) Pitch angle and rate

Figure 6-4: Combined scatter and density plots of projections of the data set onto three different planes of the state space. In the plots several regions are sketched that can be directly related to other regions in other states. Note that many data points were removed from the dense parts of the data set after calculating the density, to facilitate plotting. (Figure continues on the next page.)



(c) Flap angle and rate

Figure 6-4: (Continued) Combined scatter and density plots of projections of the data set onto three different planes of the state space. In the plots several regions are sketched that can be directly related to other regions in other states. Note that many data points were removed from the dense parts of the data set after calculating the density, to facilitate plotting.

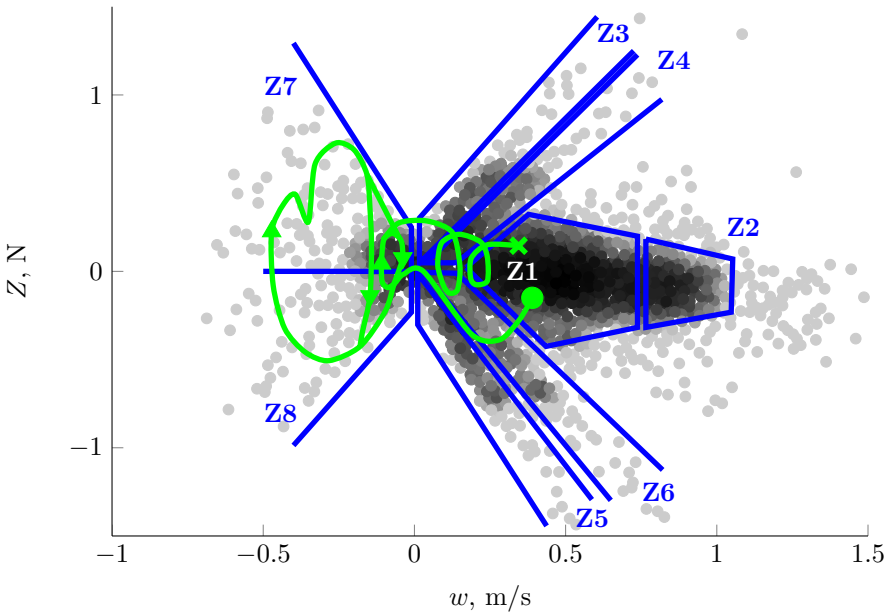


Figure 6-5: A combined scatter and density plot of the data set, projected on the w, Z-plane. Darker colors indicate higher density, regions of interest are closed off by blue lines and numbered. The green path indicates the consecutive states during a maneuver, starting from the circle and ending at the cross (as also indicated in Figure 6-3). The small and large cycle in Z7 and Z8 are indicated using vectors. Note that many data point were removed from the dense areas, especially area 6, to circumvent memory issues in plotting.

points lie purely in P1, with high pitch rates, yet small pitch angles. This region can therefore be related to the recovery phase that continues after the end of the path in Figure 6-5. This is also visible in the evolution of w after the end of the path, plotted in Figure 6-3c.

Apart from the core of the data set, there are six regions in which Z changes more or less linearly with w. All these regions can be coupled to flap stages and the start and end of maneuvers. Region Z3 for example corresponds to a high wing angle that is increasing towards the maximum wing angle (F6, F1 and F9). At the same time these data points correspond to low angles of attack. The flight path reveals that these points all stem from the start of maneuvers and the recovery phase. This is also visible in q, where slightly higher values are observed, lying in the upper halves of regions P2 and P3. Region Z4 deviates from Z3 in that the wing is at its maximum position (F1) and the angle of attack is slightly higher.

Regions Z5 and Z6 are in many respects the mirror image of Z3 and Z4. Not only the Z-force is of different sign, but the wing angle is also at its minimum (F4), and going up (F4 and F5) in region Z5. The pitch rate is slightly below the nominal level. On the other hand the angle of attack is again low and in the flight path the data points occur at the start and end of maneuvers.

Finally there are two regions of negative w and consequently negative angle of attack. Close inspection shows that they can be seen as continuations of the linear regions in positive w. In the flight path these data points cover the entire part in which the DelFly is falling after the initiation of the maneuver and before recovery. This is also visible in the high pitch angle combined with low pitch rate in region P4. In region Z7 the wing angle decreases from medium values in F2, through F3 to the lower bound in F4, then increases to the medium values in F7. Region Z8 deviates in the sense that the high wing angles are dominant. The process of increasing the wing angle to F1 goes through the regions F8 and F9. The other way around it is found that all data points in F7 lie on a straight line through the parts of Z7 and Z5 with moderate values for w. A similar observation can be made for the data points in F8 and F9. The points in the latter lie on a straight line through Z8 and Z3. The points in region F8 differ slightly by laying on a straight line through Z8 and the start of Z4.

In conclusion the maneuvers can be described as a sequence of regions, displayed as a green path in Figure 6-5. The plotted path is slightly more extreme than the actual paths encountered. Also, several cycles in Z7 and Z8 were combined for clarity. In nominal flight the DelFly is in region Z1, at the green circle. Due to the elevator input the DelFly moves towards region Z6, and starts falling as the wings are closed. The aircraft then tilts backwards and goes through one or two small flap cycles in the rightmost part of Z7 and Z8. Then when the wings open again, the DelFly instead goes through two large flap cycles in Z7 and Z8. Finally it recovers by going through another small cycle in the negative w domain and then continuing through several flap cycles into Z1. The recovery phase continues in this manner through Z2 and then back to Z1, as can be seen in the time evolution of w in Figure 6-3c.

Because the DelFly data set is used to validate the system identification process, it is important to include enough dynamic effects. Because the regions Z3 up to Z8 show the largest spread in Z-force and they are strongly coupled to the wing positions, a subflap model will be constructed. This model will thus combine both flight dynamics and flapping aerodynamics, in contrast to the models discussed in the literature review (subsection 3-3-1).

The recognizability of flight phases in the data sets is not just a positive thing. After all, a spline requires a uniform distribution of data over the state space. For the multiplex spline

this is even more true than for the simplex spline (see subsection 5-2-2). Regions such as Z7 and Z8 in Figure 6-5 that also correspond to the fixed flap angle regions F7 up to F9, are detached from the total data set in multiple dimensions. This puts upper bounds on the degrees of the spline that should prevent overfitting effects in the voids.

6-2 Multiplex spline models of the DelFly

Now that the data is analyzed, all required knowledge is available for constructing the spline models. Both longitudinal forces, the Z- and X-force, are modeled as a function of (parts of) the state (u, w, θ, q, ζ). This will be done along the lines of the system identification algorithm presented in section 5-3. The construction and structure of the models is covered in subsection 6-2-1. Some reference is made to the previous section when data analysis is considered. The performance of the resulting spline models is assessed in subsection 6-2-2.

6-2-1 Model structure

The steps of the system identification process were undertaken to arrive at the models presented in Tables 6-1 and 6-2 for the Z- and X-force respectively. Although the algorithm is linear, the models are a result of an extensive iterative process. In the end only eight multiplex splines for each force are discussed in this section. These are four models that use the complete state and the four best performing splines with less states. All models however include the flap phase ζ and at least two other states. The process that lead to the model structures is discussed below.

The first step in the system identification algorithm is to define layers. It was found in sections 5-2 and 5-4 that this choice of layers may have a significant effect on the model performance. It is especially important to separate independent states. Therefore the flap phase is isolated in a one-dimensional layer in all cases. In the splines MZ1 up to MZ4 and MX1 up to MX4, that include the entire state vector, the velocities (u, w) are never mixed with pitch states (θ, q). When smaller state vectors were considered it was found that models that do mix these states hardly ever perform as good as the other splines. Only MZ7 has such a mix in its first layer (w, q). Note that all layers are chosen one- or two-dimensional. This is done to prevent problems with the alignment of triangulated cubes due to errors in the Matlab Delaunay code.

When the data is projected onto the layers, a triangulation can be defined in each layer. A standard type-I triangulation of the rectangle is used in two-dimensional layers to facilitate the automation of the process. Note that the resulting triangulations satisfy the quality criteria mentioned in section 3-1 of the literature review. The iterations on this step lead to the conclusion that extra approximation power is required in the states w, θ and ζ. Therefore their domain is split in half before triangulating the resulting rectangles. Note that one-dimensional layers are not split in multiple parts if they contain u or q.

In parallel with the triangulation of layers, the degree and continuity order of the polynomials are defined. Due to the poor distribution of the data over the state space, overfitting to local data clusters is a major concern. Therefore the polynomials were chosen linear in all layers, except the layer containing the flap phase. This state was modeled using a quadratic

polynomial because visual inspection of the data implied a quadratic relation between ζ and the forces. All continuity orders were set to zero. Several tests were performed with higher degree polynomials, but their performance in terms of RMSE was a lot worse than that of the models in Tables 6-1 and 6-2.

To compare the performance of the multiplex spline models with existing methods, a linear and a quadratic simplex spline were made to model both forces using all states. The triangulation used in these models was a standard type-I triangulation of a single orthotope encompassing the entire domain. This was done to prevent the issue of alignment between triangulated orthotopes mentioned before. On top of that, the number of B-coefficients of the simplex spline was kept in the same range as the multiplex spline by not splitting the domain in multiple orthotopes before triangulating. A linear, a quadratic and a quintic polynomial model were also constructed for each force. In both cases the quintic polynomial provides the optimal performance of polynomial models on the studied data sets.

6-2-2 Model performance analysis

In this subsection the performance of the spline models of both Z- and X-force is assessed. All splines are fitted to a randomly selected set containing 80% of the data points. The remaining data points are used to calculate the performance measures discussed in this subsection. In selecting a verification data set it is important that the data points are well spread over the entire domain and contain enough dynamic effects. Because it is hard to do this for a system of which the dynamic effects are largely unknown, a bulk approach was taken. That is, each spline was generated 10 times using different random splits of the data set. The performance measures presented in Tables 6-1 and 6-2 are made up of the mean and standard deviation of the calculated metrics for all iterations.

The models of Z-force were investigated first, and are therefore treated in most detail here. The X-force models serve to show that the results can be repeated under similar conditions.

The performance assessment is three-fold. First of all the splines are subjected to a visual inspection. For five-dimensional models this means that slices of the domain are taken such that the behavior in two states at a time can be investigated. Secondly the statistical performance measures RMSE and RRMS are calculated. The first indicates the ability to estimate physical outcome, whereas with the second this ability is compared with the simplest model available: the mean of the data. Finally the behavior of B-coefficients is investigated to show the behavior of the splines near the edges of the domain.

Visual inspection

We start the validation process by visual inspection. The goal of this inspection is to check whether the splines are actually following any visible trend in the data. Three slices of the (2,2,1)-multiplex spline modeling the Z-force MZ1 are shown in Figure 6-6. The first two slices in Figures 6-6a and 6-6b correspond to the layers (u, w) and (θ, q). In the last figure a slice through the variables θ and ζ is plotted to show the distribution of data over the different states. The data points in Figure 6-6 lie within a small radius from the location of the slice.

In the plots of Figure 6-6 it is clear that the data is often concentrated along lines through the high-dimensional space. This means that it is possible that the linear spline elements

interpolate over large distances, ignoring local effects. In Figure 6-6c, two regions are visible at θ-values of 1.3 and 1.8 radians in which data is available. The linear spline in the direction of θ will mainly focus on interpolating across the gap in the data, which prevents it from modeling local effects.

On the other hand it is possible that closely packed data with little dynamic information is approximated using a large number of polynomials. For example, in Figure 6-6b two groups of closely packed data can again be identified. The larger region is a rather linear group of data points, whereas the small group does not directly present any evolution in this plane. In the interpolation effort the spline has the freedom to construct two linear planes to approximate this second small group, which results in overfitting effects. The spline predicts Z-values of 1N at the edges based on data that is interpolated at -0.2N. This is also visible in the velocity plane in Figure 6-6a.

Performance assessment using error metrics

To provide a global measure for the approximation performance of the splines, the RMSE and RRMS are calculated. The outcome of these calculations is presented in Table 6-1 for the Z-force models. The RMSE-values are absolute, and should thus be compared with the actual range of the output variables. In the data set used, the Z-force lies in the range [-1.4,1.5]N, but the nominal value lies between -0.2 and 0.2N. Compared to this nominal value, the RMSE-values of 0.14 indicate poor approximation performance.

Therefore it is deemed more informative to check the RRMS-value. This value is calculated by dividing the RMSE of the model by the standard deviation of the data itself. It thus compares the approximation power of the model with that of the mean. From the RRMS-values in Table 6-1 it is clear that the multiplex splines all perform better than the mean, and thus model some dynamic effects.

On the other hand the RRMS-values of more than 75% are probably not sufficient for most applications. Comparing the multiplex spline performance to that of the other, validated methods, it is clear that this poor performance is not due to the multiplex spline itself. It is most likely that the data set in its current form is not fit for global system identification using splines. This may either be because of the inclusion of few dynamic effects or the poor spread of data points over the state space. As only multiple realizations of the same maneuver were studied, both are viable causes of modeling difficulties.

A remaining question is whether the multiplex spline and the related system identification algorithm can be validated based on the achieved approximation performance. To answer this question the multiplex splines are compared to the simplex splines and the polynomial models for Z-force. As in section 5-2 the first observation is that the multiplex spline presents many different options for model structure. The ability to isolate states with a different requirement for polynomial degree seems to improve the RMSE, as can be derived from the comparison of for example MZ1 and the simplex splines. The performance is improved by making the model linear in one group of states, and quadratic in another (ζ in this case). Another improvement lies in the amount of B-coefficients and the resulting amount of degrees of freedom. As seen before in section 5-2 the simplex spline requires many more coefficients and is less efficient in converting them to degrees of freedom. The small DoF is most likely to be the cause of the poor performance of SZ1.

Table 6-1: The performance of the spline models for Z-force is measured using both the RMSE and the RRMS. For both metrics the mean and standard deviation over 10 runs are given. The difference between the runs was the distribution of data points over the fitting and verification sets. Note that RMSE values are absolute, and should be compared to the range $Z \in [-1.4, 1.5]N$ and nominal values between -0.2 and 0.2N.

ID	Layers	Degrees	Coefficients	DoF	RMSE	RRMS
MZ1	$(u,w),(\theta,q),\zeta$	(1,1,2)	702	140	0.1437 ± 0.0045	0.7790 ± 0.0095
MZ2	$(u,w),\theta,q,\zeta$	(1,1,1,2)	540	142	0.1413 ± 0.0040	0.7658 ± 0.0131
MZ3	$u,w,(\theta,q),\zeta$	(1,1,1,2)	576	144	0.1429 ± 0.0042	0.7746 ± 0.0113
MZ4	u,w,θ,q,ζ	(1,1,1,1,2)	384	144	0.1419 ± 0.0035	0.7694 ± 0.0139
MZ5	w,θ,q,ζ	(1,1,1,2)	192	72	0.1470 ± 0.0052	0.7964 ± 0.0131
MZ6	$w,(\theta,q),\zeta$	(1,1,2)	288	72	0.1483 ± 0.0033	0.7992 ± 0.0143
MZ7	$(w,q),\zeta$	(1,2)	72	24	0.1568 ± 0.0035	0.8392 ± 0.0088
MZ8	$(u,w),\zeta$	(1,2)	72	24	0.1571 ± 0.0039	0.8308 ± 0.0056
SZ1	(u,w,θ,q,ζ)	1	564	28	0.1666 ± 0.0058	0.8858 ± 0.0078
SZ2	(u,w,θ,q,ζ)	2	1554	192	0.1481 ± 0.0045	0.7901 ± 0.0173
PZ1	(u,w,θ,q,ζ)	1	6	6	0.1836 ± 0.0053	0.9792 ± 0.0028
PZ2	(u,w,θ,q,ζ)	2	21	21	0.1624 ± 0.0037	0.8566 ± 0.0067
PZ3	(u,w,θ,q,ζ)	5	252	252	0.1402 ± 0.0042	0.7491 ± 0.0200

Table 6-2: The performance of the spline models for X-force. Note that RMSE values are absolute, and should be compared to the range $X \in [-0.65, 0.90]N$ and nominal values between 0 and 0.4N.

ID	Layers	Degrees	Coefficients	DoF	RMSE	RRMS
MX1	$(u,w),(\theta,q),\zeta$	(1,1,2)	702	140	0.0673 ± 0.0014	0.6727 ± 0.0138
MX2	$(u,w),\theta,q,\zeta$	(1,1,1,2)	540	142	0.0663 ± 0.0023	0.6632 ± 0.0208
MX3	$u,w,(\theta,q),\zeta$	(1,1,1,2)	576	144	0.0645 ± 0.0023	0.6450 ± 0.0216
MX4	u,w,θ,q,ζ	(1,1,1,1,2)	384	144	0.0641 ± 0.0021	0.6415 ± 0.0207
MX5	u,w,ζ	(1,1,2)	48	24	0.0658 ± 0.0024	0.6627 ± 0.0233
MX6	$(u,w),\zeta$	(1,2)	72	24	0.0701 ± 0.0019	0.6903 ± 0.0172
MX7	w,θ,q,ζ	(1,1,1,2)	192	72	0.0719 ± 0.0024	0.7218 ± 0.0127
MX8	$w,(\theta,q),\zeta$	(1,1,2)	288	72	0.0733 ± 0.0024	0.7308 ± 0.0120
SX1	(u,w,θ,q,ζ)	1	564	28	0.0938 ± 0.0019	0.9374 ± 0.0063
SX2	(u,w,θ,q,ζ)	2	1554	192	0.0783 ± 0.0013	0.7984 ± 0.0096
PX1	(u,w,θ,q,ζ)	1	6	6	0.0968 ± 0.0016	0.9716 ± 0.0038
PX2	(u,w,θ,q,ζ)	2	21	21	0.0945 ± 0.0018	0.9343 ± 0.0073
PX3	(u,w,θ,q,ζ)	5	252	252	0.0705 ± 0.0035	0.6989 ± 0.0304

Also when compared to the polynomial models the multiplex spline performs well. The linear and quadratic polynomials PZ1 and PZ2 are unable to match the RMSE and RRMS-values of MZ1 up to MZ4. The quintic polynomial PZ3 was added to the list because it represents the optimal performance of the polynomial models. Note that it performs slightly, though hardly significantly better than MZ2 and MZ4. This is an interesting result that cannot be explained at this point. In general it was found that degrees higher than two only deteriorated performance for the spline models. It is however important to realize that in the multiplex splines MZ1 up to MZ4 the total degrees are four, five, five and six respectively. This may explain the similarity in the performance of the multiplex splines and the quintic polynomial model.

The performance of the models for X-force in Table 6-2 confirm these results. Note that overall the performance of the splines is better than those modeling Z-force, both in RMSE and RRMS. The difference is smallest in RRMS, as the X-force varies less than the Z-force. The spread of performance over the different types of models is also larger. The splines MX1 up to MX5 are clearly the best models. This time the quintic polynomial PX3 is unable to reach the same RRMS-values, and the simplex and multiplex splines are even further apart.

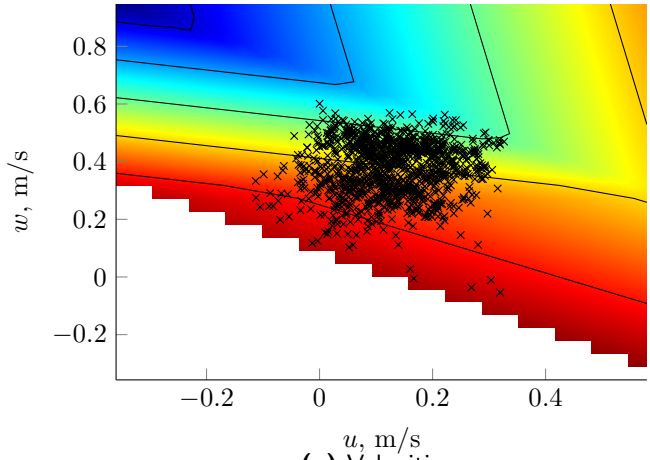
All in all, the error metrics show that the multiplex spline is able to match or improve the performance of existing modeling techniques on the given data set. Because the results can be achieved for both Z- and X-force, the spline and system identification algorithm are validated. It is however advised that the algorithm is tested on a more suitable data set, preferably with differences in complexity between states.

Finally a note should be made regarding the relevance of states for modeling the two longitudinal forces on the DelFly. All multiplex splines MZ5 up to MZ8 and MX5 up to MX8 use the vertical velocity. As these splines were the ones with the best performance, it may be concluded that w is essential in modeling both Z- and X-force. The pitch related states are especially important for modeling the Z-force, whereas in X-force the horizontal velocity u seems more relevant. Note for example that the smallest multiplex spline of all, MX5, is able to achieve similar performance as MX1 up to MX4 with only 24 degrees of freedom.

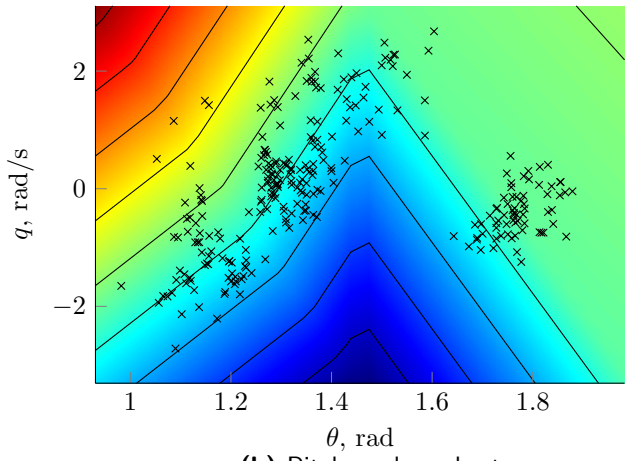
Behavior of B-coefficients

In Figure 6-7 a number of boxplots is presented that indicate the spread of B-coefficients of MZ1 and MZ4 over the state space domain. In each plot the coefficients in the wing flap layer are used. In the pitch layer the coefficients are fixed to one of the coefficients at the edge of the domain. That is, each boxplot shows the spread of the coefficients at one vertex of the domain of pitch and the indicated location in flap, over the velocity domain. This is done for all coefficients in the flap layer.

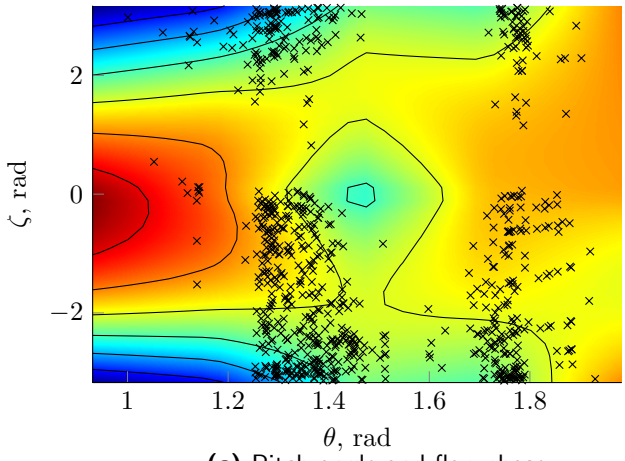
In these plots the excursions at domain edges can be clearly identified. The spread in coefficients of MZ1 is largest at the domain vertices, as expected. The coefficients of MZ4 are more extreme than those of MZ1, and especially show excursion in the center of the flap layer simplices. This may be caused by the fact that circular constraints were installed, meaning both ends of each simplex were bound to each other. At least it is clear that the geometric and polynomial effects described in section 5-2 also play a role in these very simple models. As the data set is mostly concentrated in the center of the multiplices, the multiplex with less, yet higher-dimensional layers will behave better outside the convex hull of the data points.



(a) Velocities

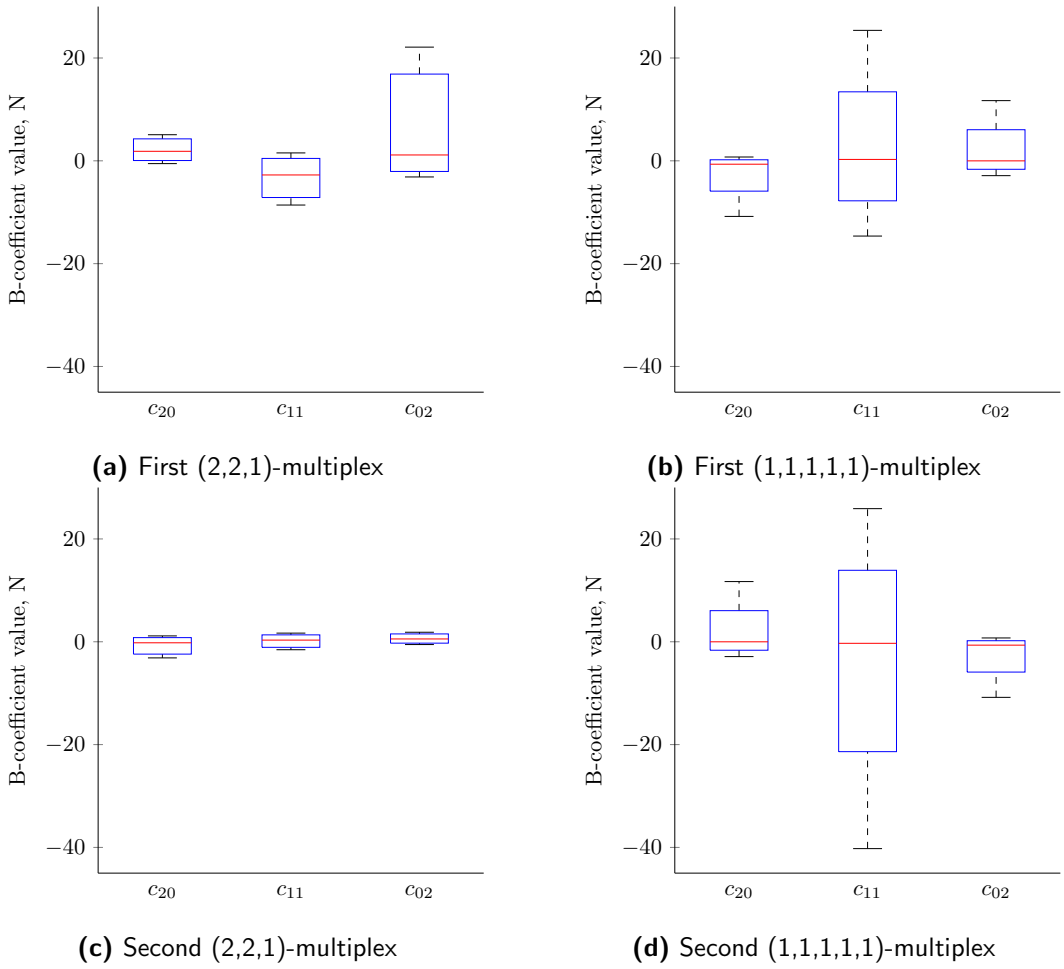


(b) Pitch angle and rate



(c) Pitch angle and flap phase

Figure 6-6: Several slices of the (2,2,1)-multiplex spline MZ1 used to approximate the DelFly Z-force. The cuts that were used were $u = -0.063$m/s, $w = 0.49$m/s, $\theta = 1.36$rad, $q = 0.16$rad/s and $\zeta = 0$rad. These values were chosen to give a representative view of the locally available data, and because they lie close to the core of the data set.



(a) First (2,2,1)-multiplex

(b) First (1,1,1,1,1)-multiplex

(c) Second (2,2,1)-multiplex

(d) Second (1,1,1,1,1)-multiplex

Figure 6-7: Boxplots of the B-coefficients of the (2,2,1)- and the (1,1,1,1,1)-multiplex splines MZ1 and MZ4. Each boxplot corresponds to a vertex on the edge of the pitch angle and rate domain, and to the indicated coefficient in the flap layer. The spread is thus purely in the velocity layer. The first multiplex corresponds to $\zeta \in [0, \pi]$.

Similar results were found when only the pitch layer is varied, although the effects turned out more extreme than in the plots of Figure 6-7. This may indicate that the spread of the data in that layer is even worse.

Chapter 7

Conclusions and recommendations

As the research described in this thesis is broad, it is useful to split the conclusions in parts according to their occurrence in the chapters of this report. First the conclusion regarding the definition of the multiplex spline are presented. They are all derived from chapter 4. In section 7-2 the conclusion regarding the resulting multiplex spline characteristics are given. The focus lies on the differences between the multiplex and simplex spline, as discussed extensively in chapter 5. Finally in section 7-3 the conclusions regarding the validation process of the system identification algorithm are collected. As the analysis of the flight test data of the DelFly was a major part of chapter 6, it is also discussed here. Conclusions are linked to the research questions in chapter 2 for clarity.

7-1 Multiplex spline definition

The development of the multiplex spline theory mostly answers research question 2 regarding the effects of nesting multivariate splines. The goal of this development is to allow for fitting scattered data. In the literature review question 1 concerning the requirements for such an application was answered. In this section this set of characteristics is coupled to the characteristics of the multiplex spline derived in chapter 4.

In section 4-1 the multiplex is defined. It is also shown in what way points in the multiplex can be uniquely described using barycentric coordinates. This is a vital prerequisite for the use of the spline in any application. One of the major results of the preliminary research is theorem 1, the equivalence of the multiplex to a subset of a higher-dimensional simplex. This implies that the barycentric coordinates in the multiplex can be derived directly from those in such a simplex. This was found to be a very useful property when deriving many of the multiplex spline properties.

For the basis polynomials it is shown in section 4-2 that they result from a tensor-product between multiple Bernstein polynomials of less variables. From this it immediately follows that the basis polynomials preserve their partition of unity and non-negativity properties.

These are important indicators that the set of polynomials forms a stable local basis. The polynomial space spanned by this basis is the typical tensor-product space.

Contrary to the rectangular Bézier patch, the multiplex spline has a fixed B-net. Although the coefficients can be thought of as having a spatial location, this location is not variable. The fact that the values of these coefficients are the only variables facilitates the use of efficient least squares approximation techniques.

In many applications of function approximators, a certain degree of smoothness is desirable. In section 4-4 it is shown that an arbitrary continuity order can be defined between multiplices of equal layer structure. It is even shown that such conditions can be defined between any two multiplices of equal dimension.

All the above culminates in the conclusion that the multiplex spline possesses all required characteristics for system identification applications. In fact, the methods of data fitting presented in the literature review for simplex splines can be used directly on this new spline, yielding a wide range of efficient solution methods.

7-2 Multiplex spline characteristics and behavior

The use of the multiplex spline for function approximation was discussed in chapter 5. This discussion started with an overview of some important characteristics of the multiplex spline, followed by a set of tests in which those characteristics were investigated. Results of both these parts are described here to (partially) answer research questions 2, 3 and 7.

The two most important characteristics that are relevant for function approximation tasks are the geometric basis and the directionality of polynomial degree. The multiplex differs from the simplex in its content distribution. In the first a larger part of the content is concentrated in the inscribed sphere. This effectively means that all angles of the multiplex are more obtuse that those of the simplex, which simplifies the process of filling it with data points.

On top of that the polynomial degree can be chosen independently in each layer. This causes a directionality in approximation power. On the other hand the total degree (defined as the sum of all degrees) of the multiplex spline can rise quickly. This is not directly visible in the layers, but along the diagonal we do find total-degree behavior. When the total degree is too high, big excursions are to be expected near the edges of the domain due to overfitting to data in other parts of the domain. This can be counteracted by applying differential constraints at the boundaries, as discussed in more detail in section 8-2.

The test functions were constructed so that single effects could be isolated. By making a high dimensional test function with different levels of non-linearity in different variables, the effect of choosing layer structures was investigated. This lead to the conclusion that choosing layers based on the knowledge of the system is beneficial for overall performance. It was also made clear that the multiplex spline presents the user with a lot of options, including some that are smaller, yet better performing than the simplex splines.

To test the concept of total degree along the diagonal, a simple two-dimensional function was used. The data set was given a certain shape that either included or excluded the vertices of the domain. The performance of the multiplex spline is better than that of the simplex spline in most cases, but only within the convex hull of the data set. Outside the convex hull the

performance is worse than that of the simplex spline, and a lot worse than inside the convex hull. Both results can be explained by the difference in total degree. Inside the convex hull of the data the spline is required to provide a good fit. A higher degree will lead to more freedom in doing so, resulting in better overall performance. Away from the data, near the edges, the freedom becomes a curse as it is exploited for a better fit to the data elsewhere in the domain. This leads to large excursions that do not bear any physical meaning.

The effect of the total degree and of the contents distribution were compared by fitting splines to data in the inscribed sphere of a single domain element. The performance of cubic splines on a simplex and a cube inside the inscribed sphere are similar. Outside this insphere however the multiplex spline performs better for low-dimensional splines. In that regime the benefit of the multiplex shape is greater than the drawback of high total degree. At some point however the polynomial degree simply becomes too high and overrules the geometric benefits. At what dimension this cross-over can be expected is likely to depend on the test function.

Finally the multiplex spline is tested against the simplex spline on a data set generated using an analytic model of a quadrotor. In this case, where the expected model outcome is known, it is especially clear what the benefits are of having a different degree in different layers. The familiar relation between velocity, angle of attack and thrust is clearly visible in both splines, but the simplex spline shows a serious distortion in the low and high angle of attack range. The same holds up to a lesser extent in the engine dynamics. This test also shows that the proposed system identification algorithm works properly.

All in all the multiplex spline's characteristics allow for better approximation power and more flexibility than the simplex spline. This is especially useful when the level of non-linearity in different variables is known to be different. The increased approximation power comes at the price of a high total degree. This is truly the dilemma for the user. He should keep layers low-dimensional to retain oversight and make easy-to-fill polytopes, yet use few layers to keep the total degree down. At the same time he may use more layers if he keeps the degree in the layers low.

7-3 Validation of system identification algorithm

The process of validation of the system identification using multiplex splines was partially dominated by analyzing the DelFly data set. The conclusions regarding the dynamic effects and variables found in this analysis partially fulfill research questions 8, 9 and 10. The larger part of this section focuses on the spline model performance. This refers to questions 6 and 7

Throughout the process of system identification it was found that the data set is poorly distributed over the state space. In almost all states that were studied, groups of data points can be identified that are also clustered in other states. This implies that these regions are not mixed well. Only in terms of velocities these regions were less clear. The data set can thus be seen as a checkerboard, in which data lies only on the black squares. This makes it hard to fit a spline on a global scale.

In the plots of slices of the splines an even more severe problem is observed. In the pitch the data is mostly concentrated along lines through the two-dimensional space, whereas in the velocity layer it is even concentrated around a single point. Because these locations vary depending on the value of the other variables, the tessellation cannot be easily made smaller

within the proposed system identification algorithm. Some solutions to these problems are proposed in section 8-3 of the recommendations.

The system identification algorithm was employed again to produce the models of the DelFly Z- and X-force. It was found that the steps of choosing and triangulating layers blend in well with the process of data analysis. The projections of data sets onto lower-dimensional planes was very helpful in choosing layers. This is especially the case when little is known about the dynamics of the system to be modeled. The step of projecting the data onto the layers forces the user to become aware of the shape of his data set. The complete algorithm was again easily automated to produce several splines with different layer structure, tessellation or polynomial properties.

On the other hand it was found to be hard to predict, purely based on the projections of the data, how dense the triangulations in the layers should be. In many slices of the resulting splines multiplices were found that were rather empty. This is partially caused by the data set, but also by the inability to vary the number of simplices in one layer over the range of another layer. This problem can be tackled by using more advanced tessellation techniques. An example of such a technique is provided in the section 8-2.

In terms of constructing a global model of the DelFly based on the current data set and system identification algorithm, it seems that the optimal performance has been reached. From the large amount of models that was generated, the presented models were the best in terms of RMSE, RRMS and the behavior of B-coefficients. Some tessellations for a (2,2,1)-multiplex spline were made by hand to test this hypothesis. This exercise did not result in any improvement in terms of any of the performance metrics.

Although the RMSE of multiplex spline models is close to the nominal Z-force value, the RRMS shows that the splines at least perform better than the mean. Also, the multiplex splines perform significantly better than the (validated) simplex splines and polynomial models of similar degree. Not only the errors are smaller, the number of coefficients is also drastically lower. This trend is confirmed by the models for X-force. The combination of these facts leads to the conclusion that the system identification algorithm using multiplex splines is validated. It is however advised that a more thorough research is set-up in which a more suitable data set is used, to make sure the encountered performance is not the upper bound.

Chapter 8

Future research

In this thesis the multiplex spline was introduced for the first time to the engineering domain. It is therefore not surprising that many recommendations can be given for future research. Instead of providing an overview of many of these research topics, only a few are selected and discussed in some detail.

Ideas for future research stem from all aspects of the research presented in this report. Therefore a subdivision is made in three categories. In each category two research topics are presented, of varying size. First in section 8-1 the recommendations regarding the further development of the mathematical theory behind the multiplex spline are given. They are rather fundamental in nature, meaning there is no direct application of them in current engineering problems. Those theoretical topics that may have a direct application are presented in section 8-2. They are called tools for performance enhancement, as it is expected that they can improve the function approximation performance in the near future. Finally in section 8-3 the possible applications of the multiplex spline are discussed. This includes a set of recommendations regarding DelFly research.

8-1 Fundamental topics

As pointed out in the introduction to this chapter, there are many research topics still open in this field. In this section the focus lies on the geometric properties of the multiplex and the circumscribed simplex. First the possibility of triangulating the multiplex is discussed. Then the concept of the circumscribed triangulation is introduced.

As triangulations are used in many engineering applications, the existence of triangulations of polytopes has been an active research topic for a long time. One important result is that a hypercube can always be triangulated. In most cases however it is not trivial to find out whether such an operation is possible for a certain polytope. It is therefore interesting to investigate whether the tight constraints on the multiplex definition allow for a guarantee of existence of a triangulation.

One starting point for such a research can already be given: each hypercube can be uniformly tessellated. That is, each hypercube can be subdivided in a finite number of equal multiplices. This can be illustrated as follows. A hypercube is a (1,...,1)-multiplex. If this polytope is to be tessellated, we can always choose to use the (2,0,1,...,1)-multiplex. That is, one two-dimensional side of the hypercube is triangulated in some way, and the remaining layers are copied from the hypercube itself.

This process is actually closely related to the triangulation of layers described in the system identification algorithm. In the case of the hypercube, each layer would be a cubic domain. By triangulating the layers and combining the triangulations, a valid uniform tessellation of the hypercube is constructed. The question of triangulating a multiplex can now be reduced to the problem of triangulating a well-defined part of the hypercube.

Another interesting topic arises when defining a higher dimensional triangulation around a tessellation. In section 4-1 it was discussed that any multiplex is a slice of a higher dimensional simplex. In section 4-4 this fact was used to derive continuity conditions in the multiplex spline from the equivalent simplex spline. It was implied that it is always possible to construct two neighboring simplices around neighboring multiplices, as they share a complete $(n-1)$-edge. In many cases of larger tessellations it was found that it is not always possible to construct a circumscribed triangulation.

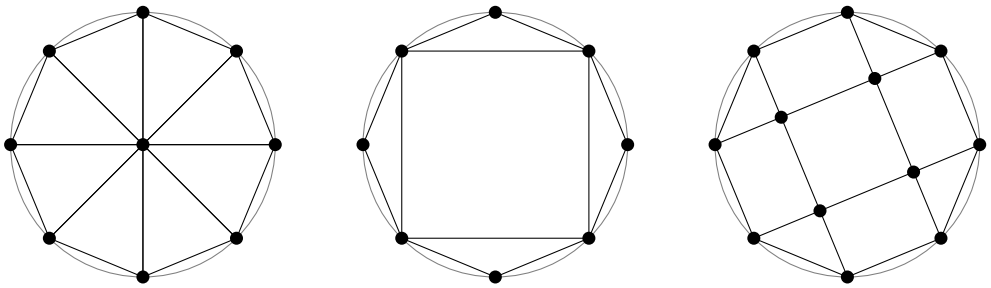
A circumscribed triangulation would be a valid triangulation that contains the same shared edges as the tessellation. It is expected that any tessellation constructed using the system identification algorithm described in section 5-3 does have an equivalent triangulation. That is because the layer sets can be defined by simply translating the triangulations of the layers into the higher-dimensional space in an appropriate way. Any combination from these layer sets would result in a valid circumscribed simplex, and it would be neighboring all the right simplices. As soon as the layer structure is abandoned, this property seems to vanish. One example of this is a rhombic tessellation of three elements, of which the shared edges form a capital Y. What the exact requirements for the existence of a circumscribed triangulation are, is however still uncertain.

8-2 Tools for performance enhancement

Apart from the purely fundamental research into the multiplex spline, some mathematical extras may be added to artificially enhance the approximation performance. Two of these tools are discussed here. The first is the option to introduce differential constraints, both along the borders of the domain and inside a multiplex. The second is a proposal to move away from the layered structure towards a nonuniform tessellation, or mixed grid partition.

Differential constraints have already been applied to simplex splines at the boundaries of the domain [18]. This prevents the B-coefficients at the edges of the domain elements from growing out of the logical physical bounds. This is a problem that was encountered often in the verification and validation processes described in this report. It is therefore proposed to apply these constraints to the multiplex spline as well.

Apart from fixating the domain edges, it is possible to apply similar constraints inside the multiplices. This idea was sparked by the fact that the total polynomial degree can be very high along the diagonal of the multiplex. If the diagonal can be constrained to a lower



(a) Triangulation **(b)** Tessellation 1 **(c)** Tessellation 2

Figure 8-1: Triangulation and tessellations that may be used to approximate the shape of a circular data set. The octagon can be filled using both simplices and multiplices, in a pure triangulation or a mixed tessellation.

polynomial degree, this may reduce the large excursion near the edges of the domain. One approach to this may be to assume the diagonal to be a one-dimensional polynomial. The basis polynomials of this polynomial can be found by setting certain variables equal (which is the definition of a diagonal). Then standard differential constraints can be applied to this polynomial.

It is however unclear what the effect of this may be. First of all, there is not really an isolated univariate polynomial on the diagonal, so the actual reduction of excursions is uncertain. There may not even be B-coefficients on the diagonal, if the degrees are chosen to be different in different layers. On top of that the approximation power is likely to be affected negatively. An in depth study would have to show how severe these effects on the approximation are, and what it means for the polynomial basis.

When knowledge of the system is completely lacking, a different approach to system identification may be taken. In these data driven techniques a tessellation is defined purely to encompass the data set. This also allows for using mixed tessellations (the use of different type multiplices), as the layers are no longer necessarily global.

One situation in which this may yield good results, is a two dimensional circular data set. Say we would be satisfied with an approximation by an octagon. For a triangulation of such a polytope we could choose to make it rotationally symmetric. With the multiplex spline this is not possible, but by using a mixed tessellations the same approximation of the circle can be attained. This is illustrated in Figure 8-1.

The front page of this report is also an example of a mixed tessellation. Both the tessellation and the data set were made by hand to illustrate the flexibility and power of such an approach. The starting point was the blue rectangle in the top left corner, in which the B-coefficients were constrained to zero. The same was done with the interior coefficients in the DelFly silhouette below the title block. The title block itself contains data points with function value 2, whereas the remainder of the domain is filled with function value 1. Finally the color map was, for artistic reasons, set to range from TU Delft blue to white and then to black. Note that around the DelFly the tessellation is a lot more dense to allow for attenuation of the zero value inside the wings. This attenuation is visible in blue and gray ripples around the DelFly and on the top of the page. In the construction of this front page it was found that the triangular patches are especially useful for attenuation of trends, as they have a higher

degree. Function values propagate further into the tessellation along the sides of a rectangular patch.

In the two-dimensional case these tessellations seem rather trivial. However, in more dimensions and on more complicated data set shapes, it becomes increasingly hard to properly align simplices in a triangulation. It is well possible that this is easier when mixed tessellations are used, especially along the lines of the tessellation in Figure 8-1c. This scheme can be extended to any number of dimensions by, in each change of multiplex, removing one one-dimensional layer and adding a dimension to the highest dimensional layer. For example, the gaps of a cube with boxes on each side (or (1,1,1)-multiplices) would be filled with prisms (or (2,1)-multiplices) and the remaining corners are filled with tetrahedra (3-simplices).

With tessellation 2 there are however still (1,1)-multiplices along the edges, which was shown to be undesirable in subsection 5-2-2. A more detailed research is required to find a general approach to mixed tessellations that truly combines the best of both worlds.

8-3 Possible applications

Function approximators have a wide range of applications. To limit the scope of this section, the two most relevant applications are presented. Because a large part of this thesis was dedicated to the DelFly, this system is discussed here as well. The structure of the multiplex spline however also presents a very interesting opportunity in the domain of non-linear control.

In section 6-2-2 it was already stated that the performance of the spline models is sufficient for validation of the system identification algorithm, but far from satisfactory in all other respects. In section 7-3 it was suggested that this is mainly caused by the shape and nature of the flight data used. The data is not spread uniformly over the state space, but rather concentrated in lower-dimensional regions in a checkerboard orientation.

One recommendation would thus be to improve this data set. This can and should be done in multiple ways. First of all the number of different maneuvers should be increased, preferably without adding extra states. That is, it is best to separate longitudinal and lateral maneuvers as much as possible. At the same time a wider range of nominal flight regimes is required to expand the core of the data set. At this point it seems that the range of velocities in the core of the data set is very limited, which is the result of using a single trim condition. The larger core can connect the data from different maneuvers and provide a solid basis of slow dynamics for the spline to fit.

To attain a more uniform spread of data over the state space, it should be kept in mind that ideally the data set contains points for all possible combinations of state values. To get close to this ideal it is advised to combine accelerating and decelerating flight with descending and climbing maneuvers. The combinations of such maneuvers quickly results in a good spread over velocities and attitude angles.

These extra data points may already be available up to some extent. More data sets were made in the μAVARI facility in different flight regimes. It may however be that these regimes are too far apart in the state space. For example, combining a low speed and a high speed data set simply introduces two separate clouds of data, between which the spline should bridge the gap. The different trim conditions are achieved by changing the center of gravity location.

This change should either be modeled analytically outside of the spline model, or it should be incorporated in the multiplex spline itself.

Apart from system identification, an interesting application lies in nonlinear control. In section 5-1 it was indicated that one of the strengths of the multiplex spline is the directional degree. This is especially useful in Nonlinear Dynamic Inversion (NDI). This control technique requires a model of the system that is affine in the input. A research effort by Tol et al. showed that to use simplex splines for NDI, some form of linearization or nonlinear optimization is required to find the control input when this requirement cannot be met [42]. Using the multiplex spline however, a division can be made between the states and the inputs by placing them in different layers. In such a model the state can be modeled in any desired nonlinear fashion, whereas at the same time the input layer is linear. Even when multiple simplices are used in the input layer, algorithms for NDI are expected to be greatly simplified.

Bibliography

[1] I. J. Anderson, M. G. Cox, and J. C. Mason. Tensor-product spline interpolation to data on or near a family of lines. *Numerical Algorithms*, 5:193–204, 1993.

[2] G. Awanou, M. J. Lai, and P. Wenston. The multivariate spline method for scattered data fitting and numerical solutions of partial differential equations. In G. Chen and M. J. Lai, editors, *Wavelets and Splines*, pages 24–74. Nasboro Press, 2006.

[3] M. Bern, D. Eppstein, and J. Gilbert. Provably good mesh generation. *Journal of Computer and System Sciences*, 48(3):384–409, 1994.

[4] S. Bouabdallah. *Design and control of quadrotors with application to autonomous flight*. PhD thesis, Lausanne EPFL, February 2007.

[5] S. Bouabdallah, P. Murrieri, and R. Siegwart. Design and control of an indoor micro quadrotor. In *Proceedings of IEEE/RSJ International Conference on Robotics and Automation*, volume 5, pages 4393–4398, May 2004.

[6] S. Bouabdallah, A. Noth, and R. Siegwart. PID vs LQ control techniques applied to an indoor micro quadrotor. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2451–2456, October 2004.

[7] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–112, 2010.

[8] J. V. Caetano, C. C. de Visser, G. C. H. E. de Croon, B. D. W. Remes, C. de Wagter, J. Verboom, and M. Mulder. Linear aerodynamic model identification of a flapping wing MAV based on flight test data. *International Journal of Micro Air Vehicles*, 5(4):273–286, 2013.

[9] J. V. Caetano, C. C. de Visser, B. D. W. Remes, C. de Wagter, and M. Mulder. Modeling a flapping wing MAV: flight path reconstruction of the Delfly II. In J. Scharl, editor, *Proceedings of the AIAA Modeling and Simulation Technologies Conference*, pages 1–12, 2013.

[10] J. V. Caetano, C. C. de Visser, B. D. W. Remes, C. de Wagter, E. van Kampen, and M. Mulder. Controlled flight maneuvers of a flapping wing micro air vehicle: a step towards the Delfly II identification. In R. Lind, editor, *Proceedings of the AIAA Atmospheric Flight Mechanics Conference*, pages 1–15, 2013.

[11] R. T. N. Chen. A survey of nonuniform inflow models for rotorcraft flight dynamics and control applications. Technical Report 102219, NASA, November 1989.

[12] H. S. M. Coxeter. *Regular polytopes*. Crowell-Collier Publishing Company, New York, second edition, 1963.

[13] C. de Boor. B-form basics. In G. Farin, editor, *Geometric Modeling: Algorithms and New Trends*, pages 131–148. Philadelphia, 1987.

[14] C. de Boor and R. Devore. Approximation by smooth multivariate splines. *Transactions of the American Mathematical Society*, 276(2):775–788, April 1983.

[15] G. C. H. E. de Croon, M. A. Groen, C. de Wagter, B. D. W. Remes, R. Ruijsink, and B. W. van Oudheusden. Design, aerodynamics and autonomy of the DelFly. *Bioinspiration and Biomimetics*, 7(2):1–27, June 2012.

[16] C. C. de Visser. *Global nonlinear model identification with multivariate splines*. PhD thesis, Delft University of Technology, July 2011.

[17] C. C. de Visser, Q. P. Chu, and J. A. Mulder. A new approach to linear regression with multivariate splines. *Automatica*, 45(12):2903–2909, December 2009.

[18] C. C. de Visser, Q. P. Chu, and J. A. Mulder. Differential constraints for bounded recursive idetification with multivariate splines. *Automatica*, 47:2059–2066, July 2011.

[19] C. C. de Visser, E. van Kampen, Q. P. Chu, and J. A. Mulder. Intersplines: a new approach to globally optimal multivariate splines using interval analysis. *Reliable Computing*, 17:153–191, December 2012.

[20] G. Farin. *Curves and surfaces for computer aided geometric design: a practical guide*. Academic Press, San Diego, fourth edition, 1997.

[21] S. Floater. Properties of tensor-product Bernstein polynomials. In J. M. Pena, editor, *Shape preserving representations in computer-aided geometric desing*, pages 149–162. Nova Science Publications, 1999.

[22] N. Govindarajan, C. C. de Visser, and K. Krishnakumar. A sparse collocation method for solving time-dependent HJB-equations using multivariate B-splines. *Automatica*, 2014. in press.

[23] N. Govindarajan, C. C. de Visser, E. van Kampen, K. Krishnakumar, J. Barlow, and V. Stepanyan. Optimal control framework for estimating autopilot safety margins. *Journal of Guidance, Control, and Dynamics*, 2014. in press.

[24] B. Grünbaum and G. C. Shepard. *Tilings and patterns*. W.H. Freeman and Company, New York, first edition, 1987.

[25] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, March 2003.

[26] G. M. Hoffmann, H. Huang, S. L. Waslander, and C. J. Tomlin. Quadrotor helicopter flight dynamics and control: theory and experiment. In *Proceedings of AIAA Guidance, Navigation and Control Conference and Exhibit*, August 2007.

[27] H. Huang, G. M. Hoffmann, S. L. Waslander, and C. J. Tomlin. Aerodynamics and control of autonomous quadrotor helicopters in aggressive maneuvering. In *Proceedings of IEEE International Conference on Robotics and Automation*, pages 3277–3282, May 2009.

[28] M. G. Kendall. *The geometry of n dimensions*. Number eight in Griffin's Statistical Monographs & Courses. Hafner publishing company, first edition, 1961.

[29] R. Kohavi and G. H. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2):273–324, December 1997.

[30] M. Lai and L. L. Schumaker. *Spline functions on triangulations*. Encyclopedia of mathematics and its applications. Cambridge University Press, 2007.

[31] M. J. Lai. *On Construction of Bivariate and Trivariate Vertex Splines on Arbitrary Mixed Grid Partitions*. PhD thesis, Hangzhou University, Hangzhou, China, August 1989.

[32] J. G. Leishman. *Principles of Helicopter Aerodynamics*. Cambridge University Press, New York, NY, 2000.

[33] M. Mirzargar and A. Entezari. Voronoi splines. *IEEE Transactions on Signal Processing*, 58(9):4572–4582, September 2010.

[34] M. Neamtu. Bivariate simplex B-splines: a new paradigm. In *Proceedings of IEEE Spring Conference on Computer Graphics*, pages 71–78, 2001.

[35] V. Pereyra and G. Scherer. Least squares scattered data fitting by truncated SVDs. *Applied Numerical Mathematics*, 40(1-2):73–86, January 2002.

[36] P. Pounds, R. Mahony, and P. Corke. Modelling and control of a large quad-rotor robot. *Control Engineering Practice*, 18(7):691–699, February 2010.

[37] J. Ruppert. A delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of Algorithms*, 18(3):548–585, 1995.

[38] J. R. Shewchuk. What is a good linear element? - Interpolation, conditioning, and quality measures. In *Proceedings of the 11th International Meshing Roundtable*, pages 115–126, September 2002.

[39] K. Strøm. Products of B-patches. *Numerical Algorithms*, 4:323–337, 1993.

[40] L. G. Sun, C. C. de Visser, and Q. P. Chu. A new multivariate tensor-product simplex B-spline with its application in real-time modular adaptive flight control. In *Proceedings of AIAA Guidance, Navigation and Control Conference*, January 2014.

[41] A. Tarantola. *Inverse problem theory and methods for model parameter estimation*. Society for Industrial and Applied Mathematics, 2005.

[42] H. J. Tol, C. C. de Visser, E. van Kampen, and Q. P. Chu. Nonlinear multivariate spline-based control allocation for high-performance aircraft. *Journal of Guidance, Control, and Dynamics*, 2014. in press.

[43] R. Zawiski and M. Blachuta. Dynamics and optimal control of quadrotor platform. In *Proceedings of AIAA Guidance, Navigation and Control Conference*, August 2012.

Part III

Draft Mathematical Paper

A New Approach to Continuity in Splines of Tensor-Product Bernstein Polynomials

Tim Visser, Cornelis C. de Visser[a]

[a]*Department of Aerospace Engineering, Delft University of Technology, Delft, The Netherlands*

Abstract

If two equal dimensional geometric bases of Bernstein polynomials share a complete edge, C^r continuity can be defined across that edge. In this paper a new way of deriving such continuity conditions for general tensor-product Bernstein polynomials is presented. By defining a circumscribed simplex, higher-dimensional simplex continuity conditions can be used in a simplified form. It is found that for C^r continuity with $r > 0$, degree changing operations are required when the geometric bases are unequal in form.

Keywords: Bernstein polynomials, Tensor-product polynomials, Simplex B-splines, C^r Continuity

1. Introduction

Let $\pi(\boldsymbol{x}) \in \Pi_n^d, \boldsymbol{x} \in \mathbb{R}^n$ be a tensor-product Bernstein polynomial defined as the product of linear combinations of multivariate Bernstein basis polynomials.

$$\pi(\boldsymbol{x}) = \prod_{i=1}^{\ell} \sum_{|\kappa_i|=d_i} c_{\kappa_i} B_{\kappa_i}^{d_i}(\boldsymbol{x}_i) \tag{1}$$

In this definition $\kappa_i = (\kappa_{i0}, ..., \kappa_{i\nu_i}) \in \mathbb{N}^{\nu_i}$ is a multi-index with $|\kappa_i|$ its 1-norm and $\nu_i \in \mathbb{N}$ the dimension of the polynomial. That is, $\boldsymbol{x}_i \in \mathbb{R}^{\nu_i}, \forall i \in \mathbb{Z}_+^{\ell}$ are subsets of the vector $\boldsymbol{x}$, here referred to as layers of the domain. To simplify the notation we introduce multi-indices for the dimension $\nu = (\nu_1, ..., \nu_\ell) \in \mathbb{N}^\ell$ such that $|\nu| = n$ and for the degree $d = (d_1, ..., d_\ell) \in \mathbb{N}^\ell$, with an entry for each layer.

After combining the coefficients c_{κ_i} in (1) to a total coefficient $c_\lambda = \prod_{\kappa_i \subset \lambda} c_{\kappa_i}$, the order of summation and multiplication can be inverted to obtain

$$\pi(\boldsymbol{x}) = \sum_{|\lambda|_\nu = d} c_\lambda \prod_{i=1}^{\ell} B_{\kappa_i}^{d_i}(\boldsymbol{x}_i) := \sum_{|\lambda|_\nu = d} c_\lambda \mathcal{B}_\lambda^d(\boldsymbol{x}) \tag{2}$$

where we introduce the total multi-index $\lambda = (\kappa_1, ..., \kappa_\ell)$ and the notation $|\lambda|_\nu = (|\kappa_1|, ..., |\kappa_\ell|)$. Note that the tensor-product basis polynomial $\mathcal{B}_\lambda^d$ is defined as the product of Bernstein basis polynomials $B_{\kappa_i}^{d_i}$.

The Bernstein basis polynomials are defined as in [1].

$$B_{\kappa_i}^{d_i}(\boldsymbol{x}) := \frac{d_i!}{\kappa_i!} \prod_{j=0}^{\nu_i} b_{ij}^{\kappa_{ij}} \tag{3}$$

Note that $\kappa_i! := \prod_{j=0}^{\nu_i} \kappa_{ij}!$. When all permutations of κ_i are considered, the Bernstein polynomials form a stable local basis for $\Pi_{\nu_i}^{d_i}$ [1]. The product in (2) implies a definition of the tensor-product basis polynomials.

$$\mathcal{B}_\lambda^d(\boldsymbol{x}) := \prod_{i=1}^{\ell} \frac{d_i!}{\kappa_i!} \prod_{j=0}^{\nu_i} b_{ij}^{\kappa_{ij}} := \frac{d!}{\lambda!} \boldsymbol{\beta}^\lambda \tag{4}$$

Many of the Bernstein polynomial's properties are preserved over the tensor-product [2], among which the partition of unity property and non-negativity.

The basis polynomials in (3) are defined using barycentric coordinates $\boldsymbol{b}_i = (\ b_{i0} \quad ... \quad b_{i\nu_i}\ ) \in \mathbb{R}^{\nu_i+1}$, with $|\boldsymbol{b}_i| = 1$. These coordinates are defined with respect to the vertices of a ν_i-simplex in the i^{th} layer. The total set of barycentric coordinates $\boldsymbol{\beta} \in \mathbb{R}^{n+\ell}$ is the collection of all barycentric descriptions in the layers.

The goal of this paper is to present a novel procedure to arrive at continuity conditions between two tensor-product polynomials. The procedure strongly relies on a discussion of the geometric implications of the tensor-product in (2). Therefore the problem is first discussed extensively in the next section. In section 3 the concept of a circumscribed simplex is discussed on which the method is based. The actual definition of continuity conditions is done in section 4, followed by an example for continuity between a rectangular and triangular B-patch in section 5.

2. Problem statement

The total set of barycentric coordinates $\boldsymbol{\beta} = (\ \boldsymbol{b}_1 \quad ... \quad \boldsymbol{b}_\ell\ )$ defines points in $\mathbb{R}^n$ by defining the projections onto the layers. This implies that the interior of the simplices in these layers define an n-dimensional polytope. As the shape of this polytope is defined by multiple simplices of different degrees, we propose to call it the ν-multiplex.

Definition 1 (Multiplex and layers)**.** *Consider a set of ℓ simplices Δ_i of dimensions $\nu = (\nu_1, ..., \nu_\ell)$ with vertices $\mathcal{W}_i$ defined in space $\mathcal{M}_i$, $i \in \mathbb{N}_+^\ell$ such that $\mathcal{M}_1 \times ... \times \mathcal{M}_\ell = \mathbb{R}^n$, with $n = |\nu|$. The* ν-multiplex Γ *is the convex hull of the vertices $\mathcal{W} = \mathcal{W}_1 \times ... \times \mathcal{W}_\ell$ of the* layers Δ_i*.*

Note that in order to define a point in a ν-multiplex Γ we require the complete set $\boldsymbol{\beta}$ of $n + l$ barycentric coordinates. Because the coordinates in each layer add up to 1, we find $|\boldsymbol{\beta}| = \ell$.

As an example, the (2,1)-multiplex is shown in figure 1. The vertices are indicated with w_{ij}, with $i \in \mathbb{N}_+^\ell$ and $j \in \mathbb{N}_0^{\nu_i}$ depending on i. Note that an origin vertex w_0 that is shared by all layers can be chosen freely.

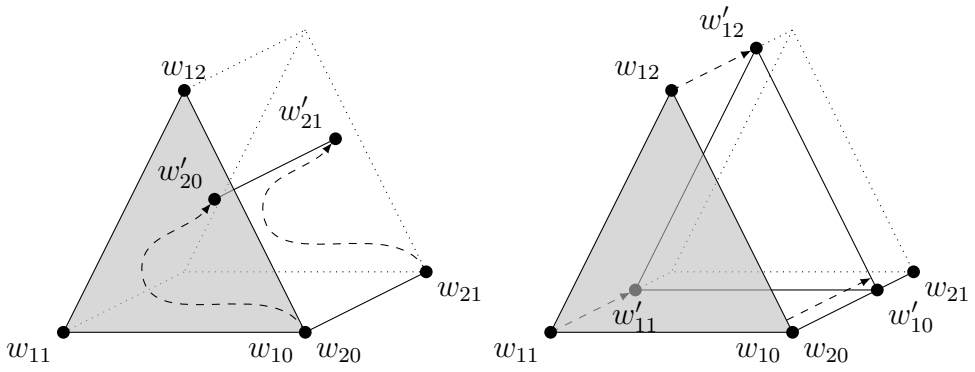
Now consider the ν-multiplex Γ and the $\tilde{\nu}$-multiplex $\tilde{\Gamma}$ with $|\nu| = |\tilde{\nu}| = n$.

Proposition 1 (Shared edge)**.** *A complete $(n-1)$-edge shared by Γ and $\tilde{\Gamma}$ is a μ-multiplex $\bar{\Gamma}$ where $|\mu| = n - 1$ and $\exists \eta, \tilde{\eta} \in \mathbb{N}^\ell : |\eta|, |\tilde{\eta}| = 1$ such that $\nu - \eta = \tilde{\nu} - \tilde{\eta} = \mu \geq 0$.*

Proof. The correction η signifies the elimination of a vertex from a layer Δ_i of Γ. As Δ_i is a ν_i-simplex, the same layer of $\bar{\Gamma}$ will be a $(\nu_i - 1)$-simplex. According to definition 1 $\bar{\Gamma}$ is then a μ-multiplex, where $\mu = (\nu_1, ..., \nu_{i-1}, \nu_i - 1, \nu_{i+1}, ..., \nu_\ell) = \nu - \eta$. The same discussion holds for $\tilde{\Gamma}$, with the elimination of a vertex from layer Δ_j using $\tilde{\eta}$. If in both cases the correction results in finding $\bar{\Gamma}$, then Γ and $\tilde{\Gamma}$ can share this edge. □

If $\mu \not\geq 0$ some layers have become negative-dimensional, which is considered impossible in the current discussion.

If the geometric bases on which two polynomials are defined, share a complete $(n-1)$-edge, continuity can be defined by setting the derivatives equal on both sides of the shared edge. Finding general continuity conditions for the case described by proposition 1, is the goal of this paper.



(a) (2,1)-multiplex (b) (1,2)-multiplex

Figure 1: A (2,1)-multiplex or prism can be constructed by either adding a 1-simplex (line segment) to a 2-simplex (triangle), or the other way around. The order does not change the polytope.

3. The circumscribed simplex

Because each layer is a simplex, we have $\sum_{i=0}^{\ell}(\nu_i + 1) = n + \ell$ barycentric coordinates. These coordinates therefore describe a convex region in the $(n + \ell - 1)$-simplex Θ, if they are scaled so that $\boldsymbol{b} = \frac{1}{\ell}\boldsymbol{\beta}$. In this new set of coordinates the subsets belonging to layers can still be identified. In this case they are set to refer to *layer sets* of vertices of Θ.

The extreme points of the region in Θ are found at combinations of extreme points in the layers. This can be formalized as follows.

Definition 2 (Simplex links)**.** *Consider an* $(n+\ell-1)$*-simplex* Θ *with vertices* $\mathcal{V}$ *subdivided in layer sets with vertices* $\mathcal{V}_i = \{v_{i0}, ..., v_{i\nu_i}\}, i \in \mathbb{Z}_+^\ell$*. A* simplex link *is an* ℓ*-simplex with vertices* $\tilde{\mathcal{V}}_\phi = \{v_{1\phi_1}, ..., v_{\ell\phi_\ell}\}$ *with* ϕ *a vector with* $\phi_i \in \mathbb{N}_0^{\nu_i}$*, such that* $\mathcal{V}_i \cap \tilde{\mathcal{V}}_\phi = \{v_{i\phi_i}\}, \forall i \in \mathbb{N}_+^\ell$*.*

In other words, a simplex link is an ℓ-simplex spanned by a combination of one vertex from each layer set. The extreme points of the region described by $\boldsymbol{\beta}$ are the barycenters of these simplex links.

It can be shown that the region spanned by the extreme points in any simplex is a multiplex. For now it is only relevant that a simplex Θ can be constructed such that the region is equal to a predefined multiplex Γ.

Theorem 1 (Circumscribed simplex)**.** *Consider a ν-multiplex Γ. There exist an infinite number of* circumscribed simplices Θ *such that Γ is the convex hull of the barycenters of simplex links spanned between predefined layer sets of Θ.*

Proof. A simplex Θ can be constructed by taking copies of the layers as layer sets and translating them into the higher dimensional space. For each layer Δ_i of Γ with vertices $\mathcal{W}_i = \{w_{i0}, ..., w_{i\nu_i}\}$ we define a layer set with vertices $\mathcal{V}_i = \{v_{i0}, ..., v_{i\nu_i}\}$, such that

$$v_{ij} = \begin{cases} (\ \ell(w_{ij} - w_0) + w_0 \quad e_i\ ) & \forall i \in \mathbb{N}_+^{\ell-1}, \forall j \in \mathbb{N}_0^{\nu_i} \\ (\ \ell(w_{ij} - w_0) + w_0 \quad -\mathbf{1}_{\ell-1}\ ) & i = \ell, \forall j \in \mathbb{N}_0^{\nu_\ell} \end{cases} \tag{5}$$

with $e_i \in \mathbb{R}^{\ell-1}$ the i^{th} unit vector and $\mathbf{1}_{\ell-1}$ a vector of $\ell - 1$ ones.

Now finding the barycenter of simplex links is equivalent to adding vertices of layers, which results in the extreme points of Γ. The appendices e_i and $-\mathbf{1}_{\ell-1}$ can be scaled to any number, implying an infinite amount of solutions. □

To derive continuity conditions between the multiplices from the simplices Θ and $\tilde{\Theta}$, we require that those simplices also share a complete $(n-1)$-edge. This is always possible by defining the layers of Γ and $\tilde{\Gamma}$ using the shared vertices, thus making sure that the circumscribed simplex $\bar{\Theta}$ of the shared edge $\bar{\Gamma}$ is shared by Γ and $\tilde{\Gamma}$.

The equivalence between the multiplex spline and the slice of the simplex spline goes beyond the geometric basis. The tensor-product basis polynomials in (4) form a scaled subset of polynomials like (3) of total degree $|d|$ and dimension $n + \ell - 1$. Correcting the multinomial coefficients and filling in the new barycentric coordinates, we find

$$\mathcal{B}_\lambda^d(\boldsymbol{\beta}) = \ell^{|d|} \frac{d!}{|d|!} B_\lambda^{|d|}(bsset) \tag{6}$$

In a similar way the B-net in the multiplex is a slice of the total degree simplex spline B-net. The control points in Θ at which the B-coefficients of the multiplex spline lie, are defined by λ as

$$q = \frac{1}{|d|} \sum_{i=1}^{\ell} \sum_{j=0}^{\nu_i} \lambda_{ij} v_{ij} \tag{7}$$

The constraints on the multi-index λ stemming from the degree in the layers, place the B-net on a slice of the simplex B-net parallel to the multiplex itself. The projection of these coefficients onto the multiplex is covered by the scaling of (6).

4. Continuity conditions

Because the multiplex spline is a slice of a higher dimensional simplex spline of total degree, the continuity conditions for the first can be taken from the latter. These simplex conditions are found by setting the derivatives equal on the shared edge. The conditions can be simplifed to a equivalence of de Casteljau iterations by observing the basis polynomials are equal on both ends [1].

$$c^{(k)}_{(0,\kappa_{11},\ldots,\kappa_{\ell\nu_\ell})}(\boldsymbol{a}) = \tilde{c}^{(k)}_{(\tilde{\kappa}_{10},\ldots,\tilde{\kappa}_{\ell,\nu_\ell-1},0)}(\tilde{\boldsymbol{a}}), \quad |\kappa| = |\tilde{\kappa}| = |d| - k \tag{8}$$

Note that it is assumed that the out-of-edge vertices are v_{10} and $\tilde{v}_{\ell\nu_i}$. The directional coordinate $\boldsymbol{a}$ belongs to a vector that is not parallel to the shared edge. Geometrically, the conditions in (8) form small copies of the simplex in the B-net.

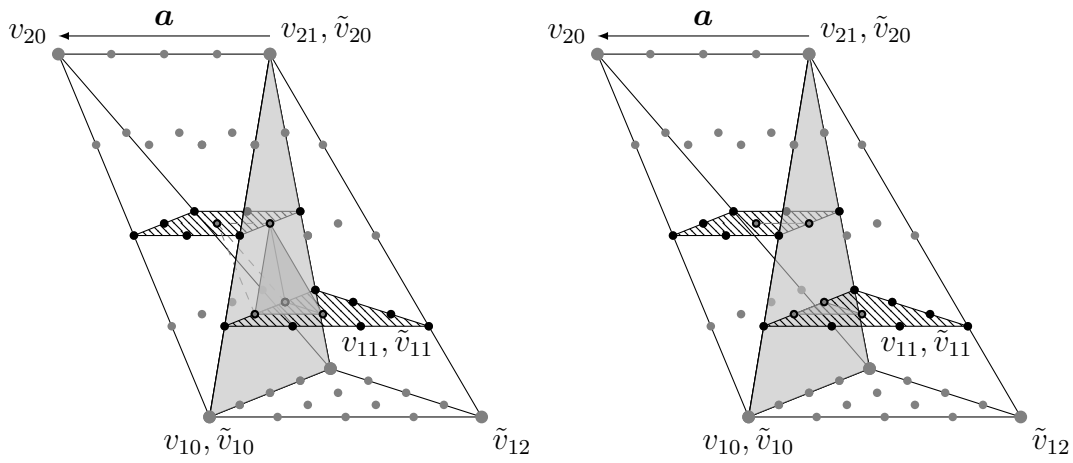
The choice of $\boldsymbol{a}$ presents an opportunity to greatly simplify the conditions. To have cospatial multiplices, the out-of-edge layers should also be cospatial. Therefore we can choose $\boldsymbol{a}$ such that it lies in both out-of-edge layers. For example, we may choose to use a vector from one shared vertex to the out-of-edge vertex of the multiplex with the lowest-dimensional out-of-edge layer.

In doing this, the elements of $\boldsymbol{a}$ are zero in all layers but the out-of-edge layer Δ_0. The same holds for $\tilde{\boldsymbol{a}}$ in the elements corresponding to $\tilde{\Delta}_\ell$. If we fill this into (8) and add the scaling of the derivatives to match the multiplex spline definition, we find

$$\begin{gathered} d!\left(c^{(k)}_{(0,\lambda_{11},\ldots,\lambda_{1\nu_1})}(\boldsymbol{a})\right)_{(\lambda_2,\ldots,\lambda_\ell)} = \tilde{d}!\left(\tilde{c}_{(\tilde{\lambda}_1,\ldots,\tilde{\lambda}_{\ell-1})}\right)^{(k)}_{(\tilde{\lambda}_{\ell 0},\ldots,\tilde{\lambda}_{\ell,\nu_\ell-1},0)}(\tilde{\boldsymbol{a}}), \\ |\lambda_1| = d_1 - k, |\tilde{\lambda}_\ell| = \tilde{d}_\ell - k \end{gathered} \tag{9}$$

These continuity conditions now form miniature copies of the out-of-edge layers in the B-nets.

Conditions in (9) refer to B-nets of different degree polynomials in Γ and $\tilde{\Gamma}$ if the out-of-edge layers are not equal-dimensional. The shift in degree



(a) Simplex continuity (b) Multiplex continuity

Figure 2: First order continuity conditions between a rectangle and a triangle derived from simplex continuity. The conditions take the form of the out-of-edge layer in both multiplices.

is equal to the continuity order, since $|\lambda_1| - |\tilde{\lambda}_1| = k$. The conditions from different orders can be combined onto a single B-net by changing the degree and applying degree changing constraints to the B-net.

One approach is to lower the degree of the conditions. This approach however does require a minimum degree to preserve approximation power [3]. The other option is to raise the degree in the highest-dimensional out-of-edge layer. In this way the approximation power is not in danger, as degrees of freedom are added. It does however require the inclusion of extra conditions to make sure that polynomials that are used in conditions are all of equal degree.

5. Example

As an example, the result for first order continuity between a triangular (2,0)-multiplex Γ and rectangular (1,1)-multiplex $\tilde{\Gamma}$ is given. The two multiplices and their circumscribed simplices are depicted in figure 2.

First of all we verify whether Γ and $\tilde{\Gamma}$ share an edge. Indeed, if we choose $\eta = (0, 1)$ and $\tilde{\eta} = (1, 0)$, then $\nu - \eta = \tilde{\nu} - \tilde{\eta} = (1, 0)$. That is, Γ and $\tilde{\Gamma}$ share a 1-simplex $\bar{\Gamma}$, a line segment. For the rectangle Γ this corresponds to the entire first layer, for the triangle this is only one edge of the first layer.

The circumscribed simplices are of dimension $n + \ell - 1 = 3$. The multiplices Γ and $\tilde{\Gamma}$ and their circumscribed simplices Θ and $\tilde{\Theta}$ are depicted in figure 2. The B-net for a bi-quadratic rectangle and a cubic triangle are also shown with respect to the B-net of the total degree simplex spline.

The first order continuity conditions can be greatly simplified by choosing the directional coordinates $\boldsymbol{a}$ from v_{21} to v_{20}. We then obtain $\boldsymbol{a} = (a_{10}, a_{11}, a_{20}, a_{21}) = (0, 0, 1, -1)$ and $\tilde{\boldsymbol{a}} = (\tilde{b}_{10}, \tilde{b}_{11}, \tilde{b}_{12}, 0)$. Contrary to standard simplex continuity (shown in figure 2a), the term a_{21} is not brought to the side of the triangle. Therefore only the coefficients that lie in the B-net of $\tilde{\Gamma}$ are included in the de Casteljau iteration, as can be seen in figure 2b.

For the zeroth order continuity conditions to also hold, the cubic polynomial on the shared edge should be reduced to a quadratic one. On top of that, the zeroth order conditions on quadratic coefficients should be converted to hold for the cubic coefficients.

6. References

[1] M. Lai, L. L. Schumaker, Spline functions on triangulations, Encyclopedia of mathematics and its applications, Cambridge University Press, 2007.

[2] S. Floater, Properties of tensor-product Bernstein polynomials, in: J. M. Pena (Ed.), Shape preserving representations in computer-aided geometric desing, Nova Science Publications, 1999, pp. 149–162.

[3] M. J. Lai, On construction of bivariate and trivariate vertex splines on arbitrary mixed grid partitions, Ph.D. thesis, Hangzhou University, Hangzhou, China (August 1989).