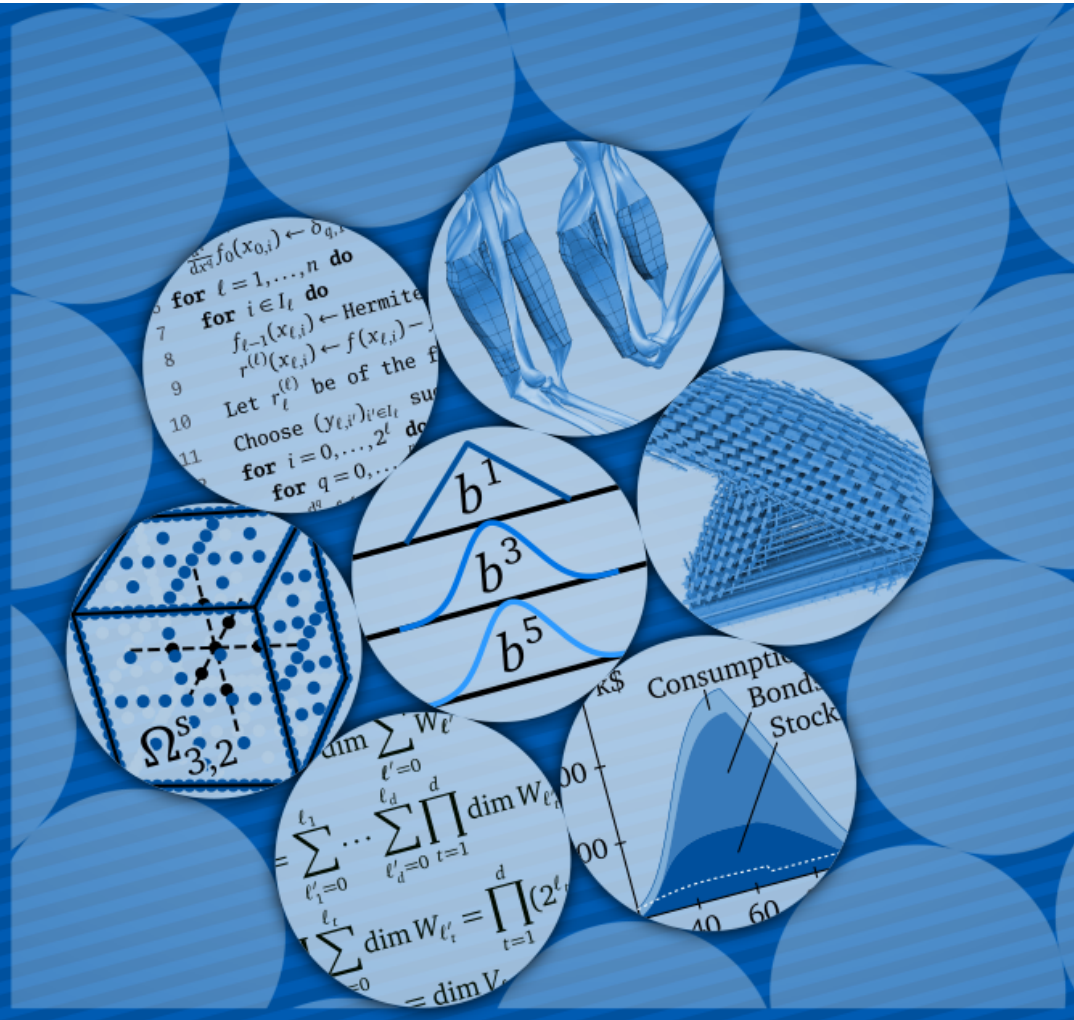
In simulation technology, computationally expensive objective functions are often replaced by cheap surrogates, which can be obtained by interpolation. Full grid interpolation methods suffer from the so-called *curse of dimensionality,* rendering them infeasible if the parameter domain of the function is higher-dimensional (four or more parameters). *Sparse grids* constitute a discretization method that drastically eases the curse, while the approximation quality deteriorates only insignificantly. However, conventional basis functions such as piecewise linear functions are not smooth (continuously differentiable). Hence, these basis functions are unsuitable for applications in which gradients are required. One example for such an application is gradient-based optimization, in which the availability of gradients greatly improves the speed of convergence and the accuracy of the results.

This thesis demonstrates that *hierarchical B-splines on sparse grids* are well-suited for obtaining smooth interpolants for higher dimensionalities. The thesis is organized in two main parts: In the first part, we derive new B-spline bases on sparse grids and study their implications on theory and algorithms. In the second part, we consider three real-world applications in optimization: topology optimization, biomechanical continuum-mechanics, and dynamic portfolio choice models in finance. The results reveal that the optimization problems of these applications can be solved accurately and efficiently with hierarchical B-splines on sparse grids.

Valentin

B-SPLINES FOR SPARSE GRIDS

Algorithms and Application to Higher-Dimensional Optimization



Julian Valentin

B-SPLINES FOR SPARSE GRIDS

Algorithms and Application to Higher-Dimensional Optimization

B-Splines for Sparse Grids: Algorithms and Application to Higher-Dimensional Optimization

Vom Stuttgarter Zentrum für Simulationswissenschaften (SC SimTech) und
der Fakultät für Informatik, Elektrotechnik und Informationstechnik
der Universität Stuttgart zur Erlangung der Würde eines Doktors
der Naturwissenschaften (Dr. rer. nat.) genehmigte Abhandlung

Vorgelegt von

Julian Valentin

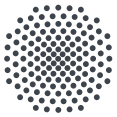
aus Stuttgart

Hauptberichter: Prof. Dr. Dirk Pflüger

Mitberichter: Prof. Dr. Stephen Roberts

.....

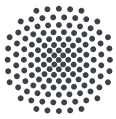
Tag der mündlichen Prüfung: .....



Universität Stuttgart

Institut für Parallele und Verteilte Systeme der Universität Stuttgart

2019



University of Stuttgart
Germany

Submitted to the University of Stuttgart

Involved institutions and departments:
Cluster of Excellence in Simulation Technology
Institute for Parallel and Distributed Systems
Chair of Simulation Software Engineering
Chair of Simulation of Large Systems

Julian Valentin
Simulation Software Engineering
Institute for Parallel and Distributed Systems
University of Stuttgart
Universitätsstr. 38
70569 Stuttgart
Germany

D 93 (dissertation)

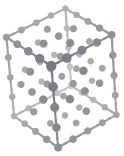
Compiled as version v13441 on December 3, 2018 at 9:34am.
Committed as 44f3bce3 (manuscript-v1) on December 3, 2018 at 9:27am.

Typeset using LaTeX and cover design by the author.

It seems that it is not enough
to have a good ***idea*** *or* ***insight.***
One needs, like Schoenberg,
the ***appreciation***
and ***courage***
to develop the idea systematically,
make its objects ***mathematically***
presentable *by giving them names,*
and give them much
exposure *in many papers.*

— Carl de Boor [Boor16]

Contents



Lists of Figures, Tables, Algorithms, and Theorems

List of Figures

List of Tables

List of Algorithms

List of Theorems

List of Symbols and Acronyms

Symbol	Meaning	Page with first occurrence
$\mathbf{0}$	$(0,\dots,0) \in \mathbb{N}_0^d$	27
$\mathbf{1}$	$(1,\dots,1) \in \mathbb{N}^d$	27
$\boldsymbol{A}$	Interpolation matrix with entries $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$	75
b^p	Cardinal B-spline of degree p	49
$c_{\boldsymbol{\ell},\boldsymbol{i}}$	Full grid interpolation coefficients	27
d	Dimensionality $d \in \mathbb{N}$	27
dim	Vector space dimension	30
e	Euler constant exp(1)	242
$\boldsymbol{e}_t$	t-th standard basis vector $\boldsymbol{e}_t := (0,\dots,0,1,0,\dots,0) \in \mathbb{R}^d$	111
$\mathbb{E}[X]$	Expectation of the random variable X	194
f	Objective function $f : [\mathbf{0},\mathbf{1}] \to \mathbb{R}$	21
$f\|_D$	Restriction $f\|_D : D \to \mathbb{R}$, $f\|_D(x) := f(x)$, onto a sub-domain D	36
$f_{\boldsymbol{\ell}}$	Full grid interpolant of f in $V_{\boldsymbol{\ell}}$	27
f^{s}	Sparse grid interpolant of f in V^{s} (on some grid Ω^{s})	22
$f^{\mathrm{s}}_{n,d}$	Regular sparse grid interpolant of f in $V^{\mathrm{s}}_{n,d}$	36
$\boldsymbol{g}$	Inequality constraint function $\boldsymbol{g} : [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{m_g}$ (constraint $\boldsymbol{g}(\boldsymbol{x}) \leq \mathbf{0}$)	21
$\boldsymbol{h}_{\boldsymbol{\ell}}$	Mesh size $\boldsymbol{h}_{\boldsymbol{\ell}} := \mathbf{2}^{-\boldsymbol{\ell}}$	26
$\boldsymbol{I}$	Identity matrix	125
$\boldsymbol{i}$	Hierarchical index $\boldsymbol{i} = \mathbf{0},\dots,\mathbf{2}^{\boldsymbol{\ell}}$	26
$I_{\boldsymbol{\ell}}$	Set of odd hierarchical indices of level $\boldsymbol{\ell}$	31
K	Finite set of hierarchical level-index pairs $(\boldsymbol{\ell},\boldsymbol{i})$ or continuous indices $\boldsymbol{k} \in \mathbb{N}_0^d$	39
K_{pole}	Pole $K_{\mathrm{pole}} \in K/\!\sim_t$ in some dimension t (K_{pole} is a subset of K)	78
$\boldsymbol{k}$	Continuously enumerated index $\boldsymbol{k} \in \mathbb{N}_0^d$ of the level-index pairs $(\boldsymbol{\ell},\boldsymbol{i})$	77
$\boldsymbol{k}_{-t}$	Vector of all entries $k_{t'}$ except the t-th	77
$\boldsymbol{k}_T$	Vector of all dimensions that are contained in $T \in \{1,\dots,d\}^j$	77
$\boldsymbol{k}_{-T}$	Vector of all dimensions that are not contained in $T \in \{1,\dots,d\}^j$	77

Symbol	Meaning	Page with first occurrence
$\boldsymbol{k}_{a:b}$	Vector of all dimensions that are contained in $a, a+1, \ldots, b$	77
$\boldsymbol{\ell}$	Hierarchical level $\boldsymbol{\ell} \in \mathbb{N}_0^d$	26
L	Finite subset $L \subseteq \mathbb{N}_0^d$ of levels	37
$\mathfrak{L}$	Linear operator $\mathfrak{L}\colon \mathbb{R}^N \to \mathbb{R}^N$ on grid point data	75
$L_{\ell,i}$	Lagrange polynomial of level ℓ, index i	66
n	Level $\in \mathbb{N}_0$ of full or sparse grid	30
N	Number of grid points for a finite set $\Omega^s \subseteq [\mathbf{0}, \mathbf{1}]$ of grid points	74
$\mathbb{N}$	Natural numbers without zero $(1, 2, 3, \ldots)$	27
$\mathbb{N}_0$	Natural numbers with zero $(\mathbb{N} \cup \{0\})$	26
$\mathcal{O}(f(x))$	Big-$\mathcal{O}$ Landau notation	30
$P^{\boldsymbol{p}}$	Space of all d-variate polynomials of coordinate degree $\leq \boldsymbol{p}$ on $[\mathbf{0}, \mathbf{1}]$	68
$\mathbb{R}$	Real numbers	21
$\mathbb{R}_{>0}$	Positive real numbers	99
$\mathbb{R}_{\geq 0}$	Non-negative real numbers	102
$S_\ell^{p,[0,1]}$	Spline space of degree p on the grid $\{x_{\ell,i} \mid i = 0, \ldots, 2^\ell\}$ of level ℓ	64
span	Linear span (set of all linear combinations)	27
$\operatorname{supp} f$	Support of a function (i.e., the closure of $\operatorname{supp}^\circ f$)	49
$\operatorname{supp}^\circ f$	Interior of the support of a continuous function (i.e., $\{\boldsymbol{x} \mid f(\boldsymbol{x}) \neq 0\}$)	72
t	Time	193
$\boldsymbol{u}$	Input of the linear operator $\mathfrak{L}$	75
$V_{\boldsymbol{\ell}}$	Nodal space of level $\boldsymbol{\ell}$	27
$V_{n,d}$	Multivariate nodal space $:= V_{n\cdot\mathbf{1}}$ of level n with dimensionality d	30
$V_{n,d}^{\mathrm{s}}$	Regular sparse grid space of level n with dimensionality d	36
V^{s}	Arbitrary sparse grid space (possibly spatially adaptive)	37
$V\vert_D$	Restriction $:= \{f\vert_D \mid f \in V\}$ onto a sub-domain D for a function space V	52
$W_{\boldsymbol{\ell}}$	Hierarchical subspace of level $\boldsymbol{\ell}$	31
$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$	Grid point $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} := \boldsymbol{i} \cdot \boldsymbol{h}_{\boldsymbol{\ell}}$	26
$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}^{\mathrm{cc}}$	Clenshaw–Curtis grid point	59
$\boldsymbol{x}^{\mathrm{opt}}$	Solution of an optimization problem of the form $\boldsymbol{x}^{\mathrm{opt}} = \operatorname{\mathbf{arg\,min}} f(\boldsymbol{x})$	120
$\boldsymbol{x}^{\mathrm{opt},*}$	Approximation for $\boldsymbol{x}^{\mathrm{opt}} = \operatorname{\mathbf{arg\,min}} f(\boldsymbol{x})$	129
$\boldsymbol{y}$	Output of the linear operator $\mathfrak{L}$	75
$\mathbb{Z}$	Integer numbers $(\ldots, -2, -1, 0, 1, 2, \ldots)$	49

Symbol	Meaning	Page with first occurrence
$\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$	Hierarchical surpluses	35
$\delta_{A,B}$	Kronecker delta $:= 1$ if $A = B$ and $:= 0$ otherwise (A, B arbitrary objects)	49
$\Theta(f(x))$	Big-Θ Landau notation	137
φ	Parent function $\varphi : [0,1] \to \mathbb{R}$ ($\varphi_{\ell,i}$ is scaled translate of φ)	96
$\varphi_{\boldsymbol{\ell},\boldsymbol{i}}$	Hierarchical basis function of level $\boldsymbol{\ell}$, index $\boldsymbol{i}$	26
$\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell},\boldsymbol{i}}$	Hierarchical standard B-spline basis function of level $\boldsymbol{\ell}$, index $\boldsymbol{i}$, degree $\boldsymbol{p}$	27
$\Omega(f(x))$	Big-Ω Landau notation	76
$\partial\Omega$	Topological boundary of a set $\Omega \subseteq \mathbb{R}^d$	36
$\Omega_{\boldsymbol{\ell}}$	Set of full grid points of level $\boldsymbol{\ell}$	29
Ω^{s}	Arbitrary sparse grid (possibly spatially adaptive)	37
$\mathring{\Omega}^{\mathrm{s}}$	Set $:= \Omega^s \cap \,]\mathbf{0}, \mathbf{1}[$ of interior grid points for a finite set $\Omega^{\mathrm{s}} \subseteq [\mathbf{0}, \mathbf{1}]$ of grid points	40
$\Omega^{\mathrm{s}}_{n,d}$	Set of regular sparse grid points of level n, dimensionality d	36
$\Omega^{\mathrm{s}(b)}_{n,d}$	Set of regular sparse grid points of level n, dim. d, boundary parameter b	41
$(\cdot)^1$	Superscript for "Piecewise linear"	27
$(\cdot)^{\mathrm{cc}}$	Superscript for "Clenshaw–Curtis"	59
$(\cdot)^{\mathrm{fs}}$	Superscript for "Fundamental spline"	99
$(\cdot)^{\mathrm{mod}}$	Superscript for "Modified"	46
$(\cdot)^{\mathrm{nak}}$	Superscript for "Not-a-knot"	65
$(\cdot)^{\mathrm{opt}}$	Superscript for "Optimal"	120
$(\cdot)^p$	Superscript for "B-splines of degree p"	51
$(\cdot)^{\mathrm{s}}$	Superscript for "Sparse grid"	36
$(\cdot)^{\mathrm{s}(b)}$	Superscript for "Sparse grid with boundary parameter b"	42
$(\cdot)^{\mathrm{T}}$	Transpose of a vector or matrix	125
$(\cdot)^{\mathrm{wfs}}$	Superscript for "Weakly fundamental spline"	113
$(\cdot)_+$	Non-negative part $\max(\cdot, 0)$	126
$[\boldsymbol{a}, \boldsymbol{b}]$	Closed hyper-rectangle $\{\boldsymbol{x} \in \mathbb{R}^d \mid \boldsymbol{a} \le \boldsymbol{x} \le \boldsymbol{b}\}$	24
$]\boldsymbol{a}, \boldsymbol{b}[$	Open hyper-rectangle $\{\boldsymbol{x} \in \mathbb{R}^d \mid \boldsymbol{a} < \boldsymbol{x} < \boldsymbol{b}\}$	40
$[\boldsymbol{a}, \boldsymbol{b}[$	Half-open hyper-rectangle $\{\boldsymbol{x} \in \mathbb{R}^d \mid \boldsymbol{a} \le \boldsymbol{x} < \boldsymbol{b}\}$	49
$\lfloor\cdot\rfloor$, $\lceil\cdot\rceil$	Floor/ceiling function (greatest/smallest integer $\le/\ge$ than $\cdot$)	65
$[\cdot]_\sim$	Equivalence class of $\cdot$ (set of all elements equivalent to $\cdot$) with respect to $\sim$	78
$\cdot/\sim$	Set of equivalence classes for an equivalence relation $\sim$ on a set $\cdot$	79
$\equiv$	Equality of functions everywhere on their domain (i.e., $\forall_x\, f_1(\boldsymbol{x}) = f_2(\boldsymbol{x})$)	30
$\boldsymbol{\nabla}_{\boldsymbol{x}} f$	Gradient of a function f with respect to $\boldsymbol{x}$ (transposed Jacobian)	22
$\boldsymbol{\nabla}^2_{\boldsymbol{x}} f$	Hessian of a function f with respect to $\boldsymbol{x}$	121
$\lVert\cdot\rVert_1$	ℓ_1 norm $\lVert\boldsymbol{x}\rVert_1 := \sum_{t=1}^d \lvert x_t\rvert$	27
$\lVert\cdot\rVert_{L^2}$	L^2 norm $\lVert f\rVert_{L^2} := \sqrt{\int_\Omega f(\boldsymbol{x})^2 \,\mathrm{d}\boldsymbol{x}}$ of a function $f : \Omega \to \mathbb{R}$	30
$\lVert\cdot\rVert_{L^\infty}$	L^∞ norm $\lVert f\rVert_{L^\infty} := \max_{\boldsymbol{x}\in\Omega} \lvert f(\boldsymbol{x})\rvert$ of a continuous function $f : \Omega \to \mathbb{R}$	185
$\oplus$	Internal direct sum of vector spaces	32
$\dot\cup$	Disjoint union of sets	31
BFS	Breadth-first search	74
CRRA	Constant relative risk aversion	193
DAG	Directed acyclic graph	92

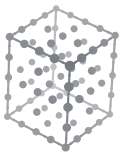
Symbol	Meaning	Page with first occurrence
FEM	Finite element method	48
HFT	Hierarchical fundamental transformation	95
LHS	Left-hand side	78
PDE	Partial differential equation	38
RHS	Right-hand side	97
SPD	Symmetric positive definite	158
TIFT	Translation-invariant fundamental transformation	97
UP	Unidirectional principle	74

Abstract/Kurzzusammenfassung

Abstract

In simulation technology, computationally expensive objective functions are often replaced by cheap surrogates, which can be obtained by interpolation. Full grid interpolation methods suffer from the so-called curse of dimensionality, rendering them infeasible if the parameter domain of the function is higher-dimensional (four or more parameters). Sparse grids constitute a discretization method that drastically eases the curse, while the approximation quality deteriorates only insignificantly. However, conventional basis functions such as piecewise linear functions are not smooth (continuously differentiable). Hence, these basis functions are unsuitable for applications in which gradients are required. One example for such an application is gradient-based optimization, in which the availability of gradients greatly improves the speed of convergence and the accuracy of the results.

This thesis demonstrates that hierarchical B-splines on sparse grids are well-suited for obtaining smooth interpolants for higher dimensionalities. The thesis is organized in two main parts: In the first part, we derive new B-spline bases on sparse grids and study their implications on theory and algorithms. In the second part, we consider three real-world applications in optimization: topology optimization, biomechanical continuum-mechanics, and dynamic portfolio choice models in finance. The results reveal that the optimization problems of these applications can be solved accurately and efficiently with hierarchical B-splines on sparse grids.



Kurzzusammenfassung

In der Simulationstechnik werden zeitaufwendige Zielfunktionen oft durch einfache Surrogate ersetzt, die durch Interpolation gewonnen werden können. Vollgitter-Interpolationsmethoden leiden unter dem sogenannten Fluch der Dimensionalität, der sie unbrauchbar macht, falls der Parameterbereich der Funktion höherdimensional ist (vier oder mehr Parameter). Dünne Gitter sind eine Diskretisierungsmethode, die den Fluch drastisch lindert und die Approximationsqualität nur leicht verschlechtert. Leider sind konventionelle Basisfunktionen wie die stückweise linearen Funktionen nicht glatt (stetig differenzierbar). Daher sind sie für Anwendungen ungeeignet, in denen Gradienten benötigt werden. Ein Beispiel für eine solche Anwendung ist gradientenbasierte Optimierung, in der die Verfügbarkeit von Gradienten die Konvergenzgeschwindigkeit und die Ergebnisgenauigkeit deutlich verbessert.

Diese Dissertation demonstriert, dass hierarchische B-Splines auf dünnen Gittern hervorragend geeignet sind, um glatte Interpolierende für höhere Dimensionalitäten zu erhalten. Die Dissertation ist in zwei Hauptteile gegliedert: Der erste Teil leitet neue B-Spline-Basen auf dünnen Gittern her und untersucht ihre Implikationen bezüglich Theorie und Algorithmen. Der zweite Teil behandelt drei Realwelt-Anwendungen aus der Optimierung: Topologieoptimierung, biomechanische Kontinuumsmechanik und Modelle der dynamischen Portfolio-Wahl in der Finanzmathematik. Die Ergebnisse zeigen, dass die Optimierungsprobleme dieser Anwendungen durch hierarchische B-Splines auf dünnen Gittern genau und effizient gelöst werden können.

Preface

Before I start, I want to thank my advisor Dirk Pflüger. His ideas and his valuable input have driven me in my time as PhD student. I also thank Stephen Roberts for his readiness to examine my thesis.

Similarly, I thank Peter Schober (Prof. Dr. Raimond Maurer, Goethe University Frankfurt), Daniel Hübner (Prof. Dr. Michael Stingl, FAU Erlangen-Nürnberg), Michael Sprenger (Prof. Oliver Röhrle, PhD, SimTech/University of Stuttgart), and Stefan Zimmer (University of Stuttgart) for the collaborations and for the enlightening discussions. I am also grateful for the exciting time with the whole group of SSE (Simulation Software Engineering) and SGS (Simulation of Large Systems), for which I want to thank all past and current PhD students and postdocs of Dirk Pflüger and Miriam Mehl. I thank Benjamin, Carolin, Gregor, Henriette, Malte, Michael, Peter, Ralf, and Theresa for throughly proofreading parts of drafts of this thesis.

Probably the most important role for the success of my PhD thesis has played my family. Without their moral support and distraction from the daily work, I doubt that this thesis would have been possible.

Likewise, I am very grateful for the financial support from the Juniorprofessurenprogramm of the Baden-Württemberg Stiftung. I thank the SimTech Cluster of Excellence for supporting my three-month research stay in Canberra, Australia.

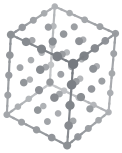
In addition, I want to thank the open-source community for making it possible to write this thesis in an aesthetically sophisticated manner. The list of software that was used to write this thesis includes LaTeX, LuaLaTeX, BibLaTeX, KOMA-Script, Ti*k*Z, Python, Matplotlib, and many more.

Now, I wish that you, dear reader, obtain as much insight as possible while reading the remaining 241 pages of this thesis.

Enjoy!

Stuttgart, December 2018

Julian Valentin



Introduction

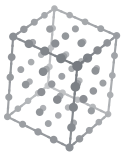
> *There is a fine line between wrong and visionary. Unfortunately, you have to be a visionary to see it...*
>
> — Sheldon Cooper (The Big Bang Theory)

Before simulation became available as a widespread tool, knowledge in science and engineering could only advance through theoretical or experimental considerations. Nowadays, processes can be simulated that would be too complicated or even impossible to be studied theoretically or experimentally, justifying that simulation is widely viewed as the "third pillar in knowledge acquisition" besides theory and experiments [Bun14].

However, simulations cannot be performed without constructing a suitable model beforehand (e.g., based on first principles). Such a model often depends on a number of uncertain or unknown parameters. Simulations can only represent the real-world circumstances well if the parameters are well-chosen. The problem of determining appropriate values for these parameters, given experimental data, is known as the *inverse problem*. Unfortunately, the solution process of such an inverse problem is non-trivial: Inverse problems are equivalent to optimization problems of the form

$$\min f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^d \text{ s.t. } \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \tag{1.1}$$

where $f(\boldsymbol{x})$ gives, for instance, a measure for the error between the simulation for the



model with the parameter $\boldsymbol{x}$ and experimental real-world data and $\boldsymbol{g}$ constrains the set of feasible parameters. Since simulations are often time-consuming, exhaustive search fails if the dimensionality d is already moderately large ($d > 4$): Full grid approaches sample each dimension of the domain independently and construct the Cartesian product of the univariate samples. The number of resulting full grid points grows exponentially in the dimensionality d, which is known as the *curse of dimensionality* [Bel61].

Inverse problems are a motivating example for optimization problems of the form (1.1), which we will consider in this thesis as our main application. The curse affects not only optimization, but also various tasks such as interpolation, quadrature, and regression, which play a vital role in numerics and computational science.

Sparse grid surrogates. The surrogate-based approach we pursue in this thesis is simple and yet powerful: Instead of directly optimizing the expensive objective function f, we replace it with a surrogate f^s that can be evaluated cheaply. We choose interpolation as our method to construct the surrogates, although other methods such as quasi-interpolation [Höl13] or regression [Pfl10] exist. Again, conventional full grid interpolation schemes are afflicted by the curse of dimensionality, which rules them out for our purposes.

This is where *sparse grids* come into play. In their simplest form, sparse grids give an *a priori* selection of full grid points and corresponding basis functions such that the exponential dependency of the grid size on the dimensionality is removed, while not deteriorating the L^2 interpolation error too much [Bun04]. However, sparse grids can also be employed spatially adaptively, where grid points are refined *a posteriori* according to suitable refinement criteria. This is of particular interest for the scope of this thesis, as spatial adaptivity enables us to increase the accuracy in regions of interest, simultaneously keeping the number of grid points at an acceptable level.

B-splines. Conventional basis functions for sparse grids (most common are piecewise linear functions) are not continuously differentiable. This poses problems for gradient-based optimization algorithms, which use the gradient $\nabla_{\boldsymbol{x}} f$ of the objective function f to update the search direction. Employing finite differences as a remedy is time-consuming and introduces new error sources.

Previous studies [Pfl10; Sic11; Vale14] suggest that *hierarchical B-splines* as sparse grid basis functions may significantly improve results. B-splines of degree p are $(p-1)$ times continuously differentiable piecewise polynomials of degree p that form a basis of the space of splines. As the derivatives of B-splines can be evaluated fast and explicitly, the convergence of gradient-based optimization techniques is greatly accelerated. Additionally, the higher order of B-splines increases the accuracy of surrogates obtained by interpolation when compared to piecewise linear bases.

Main goals. So far, there is no work that brings sparse grids and B-splines together, thoroughly examining the theoretical implications on algorithms and assessing the practical performance in real-world applications. This thesis addresses this very intersection of theory and practice of *B-splines for sparse grids*. The main goals of the thesis are

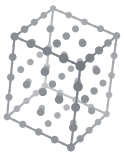
- to establish a consistent notational and theoretical framework for sparse grids with general basis functions,
- to construct new algorithmically efficient B-spline-based basis function types for sparse grids,
- to study the algorithmic properties of the new bases and to formulate suitable new algorithms, while proving their formal correctness, and
- to apply the new bases and algorithms to different real-world optimization scenarios.

These goals set the agenda for the outline of the rest of the thesis, which is described in the following.

Outline. We start in Chap. 2 by defining sparse grids for arbitrary tensor product basis functions. The advantage of introducing the notation independently of the type of basis functions is that different hierarchical B-spline bases can be substituted easily for the following chapters. We first define sparse grids with points on the boundary of the domain and then study options for the boundary treatment (as opposed to some literature [Bun04; Pfl10]).

In Chap. 3, we define the standard hierarchical B-spline basis for sparse grids. In addition, we construct various new hierarchical B-spline basis types, such as non-uniform B-splines (e.g., Clenshaw–Curtis B-splines) and modified B-splines. A mismatch of dimensions between the uniform spline space and the hierarchical B-spline space implies that, surprisingly, polynomials cannot be replicated by the standard hierarchical B-spline basis. Hence, we have to incorporate specific boundary conditions *(not-a-knot conditions)* into the hierarchical B-splines, which we explain in the second half of Chap. 3.

The new hierarchical B-spline bases call for novel algorithmic approaches to solve numerical tasks such as hierarchization (interpolation) on sparse grids. In Chap. 4, we show new algorithms for spatially adaptive sparse grids with the example problem of hierarchization based on existing algorithms, which work for B-splines only in specific special cases. In the course of Chap. 4, we construct several new hierarchical B-spline basis types to enable the applicability of the new algorithms. As mentioned above, we prove the formal correctness of every algorithm that we repeat from the literature or develop from scratch.



Chapter 5 shows how to apply B-splines on sparse grids to gradient-based optimization problems. We briefly discuss different optimization scenarios and how to solve them with various gradient-free and gradient-based optimization techniques. Numerical results are given for a number of test scenarios as well as for an example application from fuzzy arithmetic.

Three real-world applications follow in Chapters 6 to 8. In these chapters, the theoretical knowledge gained in the first half of the thesis is applied to the solution of the three real-world optimization problems:

First, in Chap. 6, we study topology optimization via a homogenized two-scale approach. For this application, the key ingredient is an interpolation scheme that preserves both the positive definiteness of the interpolated tensors and their explicit differentiability.

Second, in Chap. 7, we consider a biomechanical application in which the interpolated data values are the result of very expensive continuum-mechanical calculations. The optimization problems posed in this chapter ask for muscle activation levels such that a specific joint angle is attained, which is a recurring problem in medicine and robotics. B-spline surrogates on sparse grids decrease the necessary computing time significantly.

Third, in Chap. 8, we examine dynamic portfolio choice models. This financial application features some peculiarities that have to be considered when solving the corresponding optimization problems. For instance, it is necessary to evaluate interpolants outside their domain and calculate integrals due to random factors such as stock returns.

Chapter 9 concludes the thesis by summarizing its results and giving an overview of possible future work. In the appendix, one can find supplementary information such as technical proofs that are too verbose to be included in the main text.

Original contribution. This thesis is written to be largely self-contained. Therefore, it is necessary that some introductory definitions and results are repeated from the literature, which is properly attributed in the respective chapters. In addition, some new results have already been published. Whenever a publication is co-authored by collaborators, the original contribution of the author of this thesis is highlighted at the beginning of the respective chapters or sections.

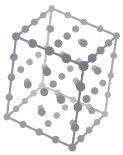
Notation. The notation of this thesis should be intuitive and suggestive. It is designed to be as natural as possible (i.e., not distracting) and as unambiguous as necessary. One example is that vectors are written in bold face, which leads to very similar formulas for the univariate and the multivariate cases. For instance, $[0,1]$ and $\sum_{\ell=0}^{n}$ become $[\mathbf{0},\mathbf{1}] = [0,1]^d$ and $\sum_{\boldsymbol{\ell}=\mathbf{0}}^{\boldsymbol{n}} = \sum_{\ell_1=0}^{n_1} \cdots \sum_{\ell_d=0}^{n_d}$, respectively. This and other necessary notation is introduced in the text when needed. If a symbol or an abbreviation is unclear, it is likely explained in the glossary at the beginning of the thesis.

Sparse Grids with Arbitrary Tensor Product Bases

> *We combine two sparse grid approximations and call it "deep sparse grids," because everything is deep today!*
>
> — In a talk at the 5th Workshop on Sparse Grids and Applications

Sparse grids are a versatile tool in numerics and scientific computing. As already mentioned in Chap. 1, their motivation is to ease the curse of dimensionality, which states that the number of full grid points grows exponentially in the dimensionality d of the underlying domain. Their general formulation and the possibility to employ sparse grids in a regular, dimensionally adaptive, or spatially adaptive fashion opens a broad field of theoretical and practical applications to sparse grids.

Sparse grids have been known for at least half a century, albeit not under this name. A paper by Smolyak [Smo63] is usually regarded as the first modern treatment of sparse grids in the form of the combination technique [Gar13]. Additionally, there are close connections to hyperbolic crosses [Tem82] and to Boolean interpolation operators [Delv82; Delv89]. The term *sparse grids* was coined by Zenger in 1991 [Zen91]. Some important subsequent work for hierarchical bases was done by Bungartz [Bun92; Bun98; Bun04]. Since then, sparse grids have been applied to various fields, for instance, data mining



[Gar01; Pan08; Pfl10], interpolation [Sic11], quadrature [Ger98], density estimation [Gri10; Peh14], PDEs [Bal94; Bun98; Nob16], and optimization [Fere05; Don09; Vale16]. Various software toolboxes for sparse grids have been developed [Kli05; Pfl10; Stoy18]. For a general introduction to sparse grids, see the tutorial by Garcke [Gar13] or the more extensive survey by Bungartz and Griebel [Bun04].

This chapter provides a consistent notational framework for the definition of sparse grids with general basis functions. The reason not to employ specific bases such as the common hat functions or B-splines of higher degrees is two-fold: First, we will define various new "flavors" of B-splines, which is easier if the basis is left open. Second, most of the statements and theorems that we will make in this thesis will hold for general basis functions (in some cases with additional assumptions) and not just for B-splines.

Besides the derivation of sparse grids with coarser boundary points in Sec. 2.4.1, this section is mostly a repetition of the definition of sparse grids with general basis functions. Our notation and presentation will roughly follow [Pfl10] and [Gar13]. A more detailed introduction to sparse grids can be found in [Bun04]. Original contributions of the thesis in this chapter are the formalization of the hierarchical splitting for arbitrary basis functions in Sections 2.1 and 2.2 and the definition of sparse grids with coarse boundary in Sec. 2.4.1.

2.1 Nodal Basis and Nodal Space

2.1.1 Univariate Case

IN THIS SECTION

Grid and basis functions. In this thesis, we consider univariate functions that are defined on the unit interval $[0,1]$. We discretize this domain by splitting it into 2^ℓ equally-sized segments, where $\ell \in \mathbb{N}_0$ is the *level*. The resulting $2^\ell + 1$ *grid points* $x_{\ell,i}$ are given by

$$x_{\ell,i} := i \cdot h_\ell, \quad i = 0, \ldots, 2^\ell, \tag{2.1}$$

where i is the *index* and $h_\ell := 2^{-\ell}$ is the *mesh size*.[1] Every grid point is associated with a *basis function*

$$\varphi_{\ell,i} : [0,1] \to \mathbb{R}. \tag{2.2}$$

[1] Note that from a strict formal perspective, this equation defined $x_{\ell,i}$ only for $i = 0, \ldots, 2^\ell$, but we will later need $x_{\ell,i}$ also for $i < 0$ or $i > 2^\ell$. The convention in this thesis is that all definitions are implicitly generalized whenever needed.

We assume $\varphi_{\ell,i}$ to be arbitrary, satisfying required assumptions when needed and stated. However, it helps for both the theory and the intuition to have a specific example of basis functions in mind. The so-called *hat functions* (linear B-splines) are the most common choice for $\varphi_{\ell,i}$:

$$\varphi^1_{\ell,i}(x) := \max(1 - |\tfrac{x}{h_\ell} - i|, 0). \tag{2.3}$$

Here and in the following, the superscript "1" stands for the degree of the linear B-spline and is not to be read as an exponent. We generalize this notation to B-splines $\varphi^p_{\ell,i}$ of arbitrary degrees p in Chap. 3.

Nodal space. The *nodal space* V_ℓ of level ℓ is defined as the linear span of all basis functions $\varphi_{\ell,i}$:

$$V_\ell := \operatorname{span}\{\varphi_{\ell,i} \mid i = 0, \ldots, 2^\ell\}. \tag{2.4}$$

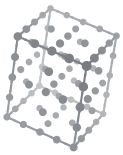
We assume that the functions $\varphi_{\ell,i}$ form a basis of V_ℓ, i.e., they are linearly independent. Consequently, every linear combination of these functions is unique. This ensures that for every objective function $f : [0,1] \to \mathbb{R}$, there is a unique function $f_\ell : [0,1] \to \mathbb{R}$ such that

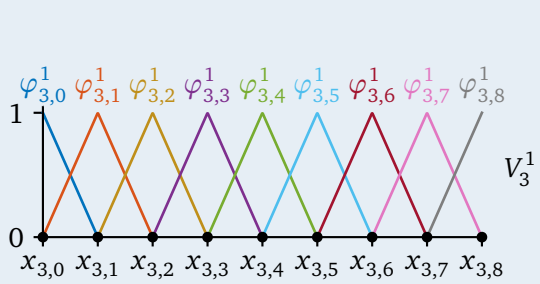
$$f_\ell = \sum_{i=0}^{2^\ell} c_{\ell,i}\varphi_{\ell,i}, \quad \forall_{i=0,\ldots,2^\ell}\ f_\ell(x_{\ell,i}) = f(x_{\ell,i}), \tag{2.5}$$

for some $c_{\ell,i} \in \mathbb{R}$. In this case, f_ℓ is called *interpolant* of f in V_ℓ. The nodal space V^1_ℓ is defined analogously to V_ℓ as the span of the hat functions $\varphi^1_{\ell,i}$. It is the space of all linear splines, that is, the space of all continuous functions on $[0,1]$ that are piecewise linear polynomials on $[x_{\ell,i}, x_{\ell,i+1}]$ for $i = 0, \ldots, 2^\ell - 1$ [Höl13]. The nodal hat function basis of level $\ell = 3$ and a linear combination are shown in Fig. 2.1.

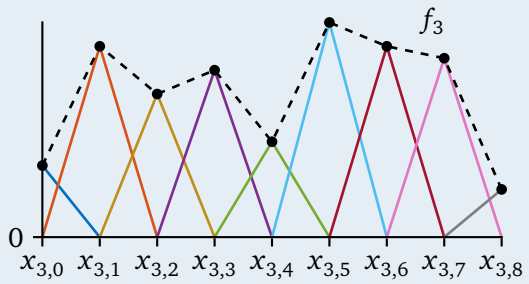
2.1.2 Multivariate Case

Cartesian and tensor products. For the multivariate case with $d \in \mathbb{N}$ dimensions, we employ a tensor product approach, for which we replace all indices, points, and functions with multi-indices, Cartesian products, and tensor products, respectively. Therefore, the domain is now $[\mathbf{0}, \mathbf{1}] := [0,1]^d$, which can be partitioned into $\prod_{t=1}^d 2^{\ell_t} = 2^{\|\boldsymbol{\ell}\|_1}$ equally-sized hyper-rectangles, where $\boldsymbol{\ell} = (\ell_1, \ldots, \ell_d) \in \mathbb{N}_0^d$ is the d-dimensional level and $\|\boldsymbol{\ell}\|_1 := \sum_{t=1}^d |\ell_t|$ is the level sum. The corners of the hyper-rectangles are given by





A Basis functions $\varphi^1_{\ell,i}$ $(i = 0,\dots,2^\ell)$ and grid points $x_{\ell,i}$ *(dots)*.



B Piecewise linear interpolant f_ℓ of some function data $f(x_{\ell,i})$ as a weighted sum of the nodal hat functions.

FIGURE 2.1 Univariate nodal hat functions of level $\ell = 3$.

the grid points

$$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} := \boldsymbol{i} \cdot \boldsymbol{h}_{\boldsymbol{\ell}}, \quad \boldsymbol{i} = \mathbf{0},\dots,\mathbf{2}^{\boldsymbol{\ell}}. \tag{2.6}$$

Relations and operations with vectors in bold face are to be read coordinate-wise in this thesis, unless stated otherwise. Bold-faced numbers like $\mathbf{0}$ are defined to be the vector $(0,\dots,0)$ in which every entry is equal to that number. This is to allow a somewhat intuitive and suggestive notation. For example, (2.6) is equivalent to the much longer formula

$$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} := (i_1 h_{\ell_1}, \dots, i_d h_{\ell_d}), \quad i_t = 0,\dots,2^{\ell_t}, \quad t = 1,\dots,d, \tag{2.7}$$

with the d-dimensional mesh size $\boldsymbol{h}_{\boldsymbol{\ell}} := \mathbf{2}^{-\boldsymbol{\ell}} = (h_{\ell_1},\dots,h_{\ell_d})$. Again, every grid point is associated with a basis function that is defined as the tensor product of the univariate functions:[2]

$$\varphi_{\boldsymbol{\ell},\boldsymbol{i}} \colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}, \quad \varphi_{\boldsymbol{\ell},\boldsymbol{i}}(\boldsymbol{x}) := \prod_{t=1}^{d} \varphi_{\ell_t,i_t}(x_t). \tag{2.8}$$

Fig. 2.2 shows an example of a bivariate nodal hat function $\varphi^1_{\boldsymbol{\ell},\boldsymbol{i}}$.

[2]Note that, although Eq. (2.8) does not cover it, one could employ basis functions of different types in each dimension, for example B-splines of different degrees. All remaining considerations in this thesis regarding tensor product basis functions are independent of whether we use the same function type or different types in each dimension.

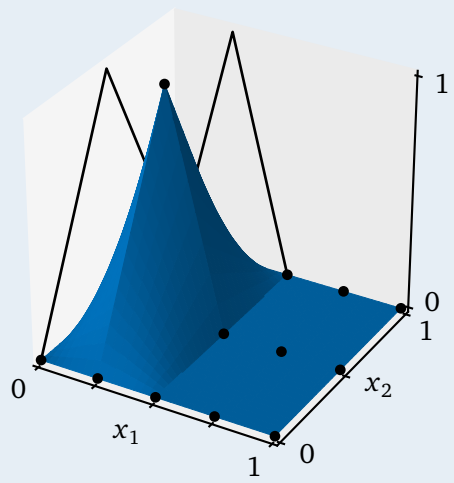


FIGURE 2.2
Bivariate nodal hat function of level $\boldsymbol{\ell} = (2,1)$ and index $i = (1,1)$ as the tensor product of two univariate nodal hat functions.

Multivariate nodal space. The multivariate nodal space V_ℓ is defined analogously to the univariate case:

$$V_\ell := \operatorname{span}\{\varphi_{\ell,i} \mid \boldsymbol{i} = \mathbf{0}, \ldots, \mathbf{2}^\ell\}. \tag{2.9}$$

In the case of hat functions $\varphi^1_{\ell,i}$, the nodal space V^1_ℓ is the d-linear spline space [Höl13], i.e., the space of all continuous functions on $[\mathbf{0},\mathbf{1}]$ that are piecewise d-linear polynomials on all hyper-rectangles

$$[\boldsymbol{x}_{\ell,\boldsymbol{i}}, \boldsymbol{x}_{\ell,\boldsymbol{i}+\mathbf{1}}] := [x_{\ell_1,i_1}, x_{\ell_1,i_1+1}] \times \cdots \times [x_{\ell_d,i_d}, x_{\ell_d,i_d+1}], \quad \boldsymbol{i} = \mathbf{0}, \ldots, \mathbf{2}^\ell - \mathbf{1}. \tag{2.10}$$

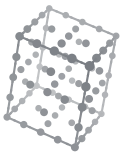
Analogously to (2.5), we can interpolate objective functions $f : [\mathbf{0},\mathbf{1}] \to \mathbb{R}$ in the nodal space V_ℓ with $f_\ell : [\mathbf{0},\mathbf{1}] \to \mathbb{R}$ satisfying

$$f_\ell = \sum_{\boldsymbol{i}=\mathbf{0}}^{\mathbf{2}^\ell} c_{\ell,\boldsymbol{i}} \varphi_{\ell,\boldsymbol{i}}, \quad \forall_{\boldsymbol{i}=\mathbf{0},\ldots,\mathbf{2}^\ell} \; f_\ell(\boldsymbol{x}_{\ell,\boldsymbol{i}}) = f(\boldsymbol{x}_{\ell,\boldsymbol{i}}), \tag{2.11}$$

where $c_{\ell,\boldsymbol{i}} \in \mathbb{R}$ and the sum is over all $\boldsymbol{i} = \mathbf{0}, \ldots, \mathbf{2}^\ell$ (i.e., $i_t = 0, \ldots, 2^{\ell_t}$, $t = 1, \ldots, d$). To ensure that the coefficients $c_{\ell,\boldsymbol{i}}$ exist for every objective function f and are uniquely determined by the values at the grid points

$$\Omega_\ell := \{\boldsymbol{x}_{\ell,\boldsymbol{i}} \mid \boldsymbol{i} = \mathbf{0}, \ldots, \mathbf{2}^\ell\}, \tag{2.12}$$

we prove the following statement:



LEMMA 2.1 (linear independence of tensor products)
The functions $\varphi_{\ell,\boldsymbol{i}}$ ($\boldsymbol{i} = \mathbf{0},\dots,\mathbf{2}^{\ell}$) form a basis of V_ℓ, if the univariate functions φ_{ℓ_t,i_t} ($i_t = 0,\dots,2^{\ell_t}$) form a basis of the univariate nodal space V_{ℓ_t} for $t = 1,\dots,d$.

PROOF Assume that $c_{\ell,\boldsymbol{i}} \in \mathbb{R}$ are chosen in (2.11) such that $f_\ell \equiv 0$. Then for all $\boldsymbol{i}' = \mathbf{0},\dots,\mathbf{2}^{\ell}$, we can evaluate (2.11) at $\boldsymbol{x}_{\ell,\boldsymbol{i}'}$ to obtain

$$\sum_{i_1=0}^{2^{\ell_1}}\left(\sum_{i_2=0}^{2^{\ell_2}}\cdots\left(\sum_{i_d=0}^{2^{\ell_d}} c_{\ell,\boldsymbol{i}}\varphi_{\ell_d,i_d}(x_{\ell_d,i'_d})\right)\cdots\varphi_{\ell_2,i_2}(x_{\ell_2,i'_2})\right)\varphi_{\ell_1,i_1}(x_{\ell_1,i'_1}) = 0. \tag{2.13}$$

We apply the univariate linear independence (x_1 direction) to infer that the sum over i_2 must vanish for all $i_1 = 0,\dots,2^{\ell_1}$. Repeating this argument for all dimensions, we have $c_{\ell,\boldsymbol{i}} = 0$ for all $\boldsymbol{i} = \mathbf{0},\dots,\mathbf{2}^{\ell}$, implying the linear independence of the functions $\varphi_{\ell,\boldsymbol{i}}$. ■

A common choice for the level $\boldsymbol{\ell}$ is $n\cdot\mathbf{1}$ for some $n \in \mathbb{N}_0$. In this case, we replace "$\boldsymbol{\ell}$" in the subscripts with "n,d" (for example, $V_{n,d} := V_{n\cdot\mathbf{1}}$). For the hat function basis $\varphi^1_{\ell,\boldsymbol{i}}$, it can be shown that the L^2 interpolation error of the interpolant $f_{n,d} \in V_{n,d}$ is given by

$$\|f - f_{n,d}\|_{L^2} = \mathcal{O}(h_n^2), \tag{2.14}$$

i.e., the order of the interpolation error is quadratic in the mesh size [Höl13; Bun04].

2.2 Hierarchical Basis and Hierarchical Subspace

IN THIS SECTION

The dimension of the nodal space V_ℓ is given by

$$\dim V_\ell = |\Omega_\ell| = \prod_{t=1}^{d}(2^{\ell_t}+1). \tag{2.15}$$

If we choose the same level $n \in \mathbb{N}_0$ in all dimensions, then the dimension of $V_{n,d}$ and the number of grid points grow at least as fast as $2^{nd} = (h_n^{-1})^d$. This exponential dependency between $\dim V_{n,d}$ and d is known as the *curse of dimensionality* [Bel61]. The curse makes interpolation on V_ℓ computationally infeasible for dimensionalities $d > 4$, as we would have to calculate and store $\dim(V_\ell)$-many coefficients $c_{\ell,\boldsymbol{i}}$.

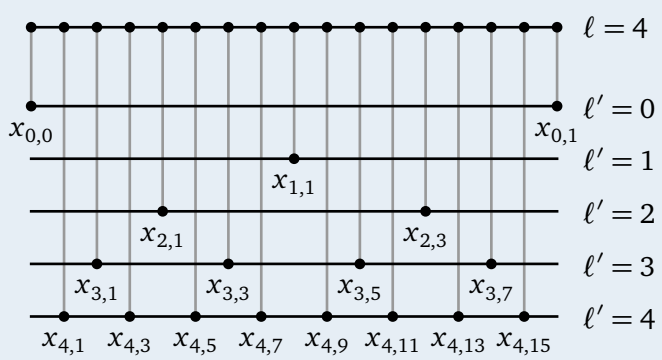


FIGURE 2.3
The set of grid points Ω_ℓ of level $\ell = 4$ *(top)* decomposes into hierarchical grids of level $\ell' \le \ell$, whose grid points $x_{\ell',i'}$ have odd indices $i' \in I_{\ell'}$ ($x_{0,0}$ being the only exception).

2.2.1 Hierarchical Splitting in the Univariate Case

Hierarchical subspaces. In order to reduce the computational effort, we first split V_ℓ into smaller subspaces and then identify subspaces that we can omit at the cost of a slightly larger error. In the univariate case, the key observation is that a grid point of a level ℓ can be written as a grid point of a higher level ℓ':

$$x_{\ell,i} = x_{\ell',i'}, \quad \ell' \ge \ell, \quad i' = 2^{\ell'-\ell} i. \tag{2.16}$$

Conversely, this implies that every grid point $x_{\ell,i}$ of level $\ell \ge 1$ and index $i \ge 1$ can be uniquely written as a grid point of a coarser level ℓ' (or $\ell' = \ell$) and an odd index i':

$$x_{\ell,i} = x_{\ell',i'}, \quad \ell' = \ell - \big[\log_2(\mathrm{xor}(i, i-1)+1) - 1\big], \quad i' = 2^{\ell'-\ell} i, \tag{2.17}$$

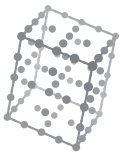
where xor is the bitwise "exclusive or" function. The term in square brackets is the exponent of the highest power of two that divides i. The two boundary points zero and one are obtained by inserting an additional level $\ell' = 0$ with indices $i' \in \{0, 1\}$. As shown in Fig. 2.3, this implies that Ω_ℓ decomposes into

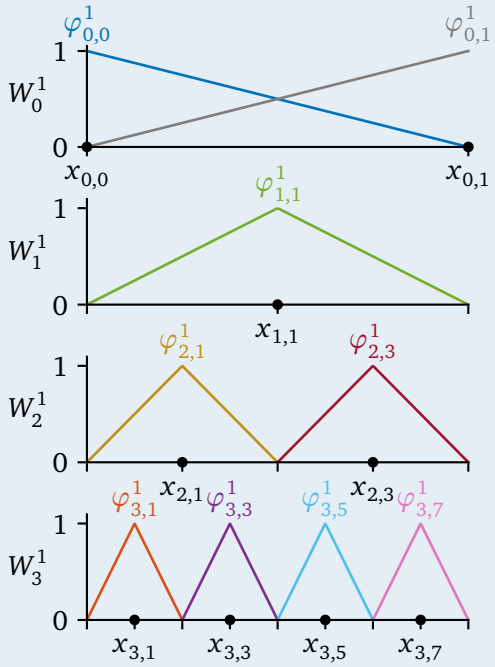
$$\Omega_\ell = \dot{\bigcup_{\ell'=0}^{\ell}} \{x_{\ell',i'} \mid i' \in I_{\ell'}\}, \quad I_{\ell'} := \begin{cases} \{i' = 0, \ldots, 2^{\ell'} \mid i' \text{ odd}\}, & \ell' > 0, \\ \{0, 1\}, & \ell' = 0, \end{cases} \tag{2.18}$$

where $\dot{\cup}$ indicates the disjoint union. We call the spaces spanned by the basis functions that correspond to the index sets $I_{\ell'}$ *hierarchical subspaces* $W_{\ell'}$:

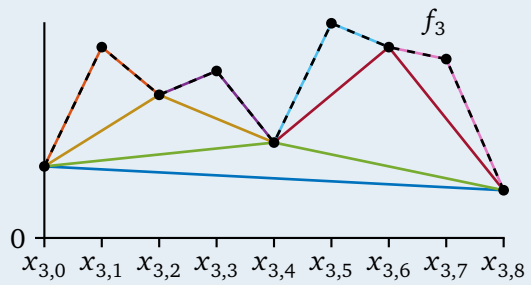
$$W_{\ell'} := \mathrm{span}\{\varphi_{\ell',i'} \mid i' \in I_{\ell'}\}. \tag{2.19}$$

The corresponding basis functions $\varphi_{\ell',i'}$, $\ell' = 0, \ldots, \ell$, $i' \in I_{\ell'}$, are called *hierarchical basis functions*. The hierarchical hat function basis is shown in Fig. 2.4.





A Basis functions $\varphi^1_{\ell',i'}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$) and grid points $x_{\ell',i'}$ *(dots)*. The domain is the unit interval $[0,1]$.



B Piecewise linear interpolant f_ℓ of some function data $f(\boldsymbol{x}_{\ell,i})$ as a linear combination of hierarchical hat functions *(stacked)*. The two boundary functions are combined to a single function *(blue)* for simplicity.

FIGURE 2.4 Univariate hierarchical hat functions up to level $\ell = 3$.

Hierarchical splitting. For the hat function basis $\varphi^1_{\ell,i}$ and other basis types, we can prove that the corresponding nodal space decomposes into the direct sum of all hierarchical subspaces of coarser levels or the same level, i.e.,

$$V_\ell \overset{?}{=} \bigoplus_{\ell'=0}^{\ell} W_{\ell'}, \tag{2.20}$$

We call this relation *hierarchical splitting*. Here, the direct sum $\oplus$ is the vector space sum that additionally indicates that the dimension of the sum $\sum_{\ell'=0}^{\ell} W_{\ell'}$ is the sum of the dimensions of the summands $W_{\ell'}$ (analogously to $|\Omega_\ell| = \sum_{\ell'=0}^{\ell} |\{x_{\ell',i'} \mid i' \in I_{\ell'}\}|$, where Ω_ℓ is the disjoint union of the sets $\{x_{\ell',i'} \mid i' \in I_{\ell'}\}$). In general, (2.20) may not be true, depending on the type of basis functions. The following lemma provides a characterization that can be used to prove (2.20) for hat functions.

LEMMA 2.2 (univariate hierarchical splitting characterization)
Equation (2.20) *is equivalent to the satisfaction of both of the following conditions:*

- *The hierarchical subspaces* $W_{\ell'}$ $(\ell' \le \ell)$ *are subspaces of* V_ℓ.
- *The hierarchical functions* $\varphi_{\ell',i'}$ $(\ell' \le \ell,\ i' \in I_{\ell'})$ *are linearly independent.*

PROOF The first condition is equivalent to $\sum_{\ell'=0}^{\ell} W_{\ell'} \subseteq V_\ell$. The second condition is equivalent to $\dim \sum_{\ell'=0}^{\ell} W_{\ell'} = \sum_{\ell'=0}^{\ell} \dim W_{\ell'}$, i.e., to the directness of the sum. Therefore, the logical conjunction of both is equivalent to $\bigoplus_{\ell'=0}^{\ell} W_{\ell'} \subseteq V_\ell$. If the sum is direct, the dimension of the sum is equal to $2+\sum_{\ell'=1}^{\ell} 2^{\ell'-1} = 2^\ell + 1$ (due to $\dim W_{\ell'} = |I_{\ell'}| = 2^{\ell'-1}$ for $\ell' > 0$ and $\dim W_{\ell'} = 2$ for $\ell' = 0$), which is also the dimension of V_ℓ. The only subspace of V_ℓ that has the same dimension as V_ℓ is V_ℓ itself, so we infer $\bigoplus_{\ell'=0}^{\ell} W_{\ell'} = V_\ell$. ■

COROLLARY 2.3 (univariate hierarchical splitting for hat functions)
The hierarchical splitting (2.20) *holds for the hat function basis.*

PROOF The first condition of Lemma 2.2 is satisfied as piecewise linear splines of level ℓ' are also piecewise linear splines of higher levels $\ell \ge \ell'$. We can prove the linear independence for the second condition by induction over ℓ: If a linear combination of $\varphi^1_{\ell',i'}$ $(\ell' \le \ell,\ i' \in I_{\ell'})$ vanishes everywhere, then the coefficients of level ℓ must be zero, as otherwise the basis functions $\varphi^1_{\ell,i'}$ $(i' \in I_\ell)$ would introduce kinks at $x_{\ell,i'}$, which the zero function does not have. This means that we have a zero linear combination of $\varphi^1_{\ell',i'}$ for $\ell' \le \ell - 1$, $i' \in I_{\ell'}$, and by the induction hypothesis, the other coefficients also vanish. ■

2.2.2 Hierarchical Splitting in the Multivariate Case

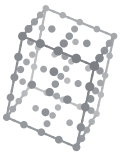
Multivariate hierarchical subspaces are defined analogously to the univariate case:

$$W_{\boldsymbol{\ell}} := \operatorname{span}\{\varphi_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}, \quad I_{\boldsymbol{\ell}} := I_{\ell_1} \times \cdots \times I_{\ell_d}, \quad \boldsymbol{\ell} \in \mathbb{N}_0^d. \tag{2.21}$$

The univariate hierarchical splitting (2.20) can now be generalized to

$$V_{\boldsymbol{\ell}} \overset{?}{=} \bigoplus_{\boldsymbol{\ell}'=\boldsymbol{0}}^{\boldsymbol{\ell}} W_{\boldsymbol{\ell}'}, \tag{2.22}$$

Again, this relation does not hold in general. We use a multivariate counterpart of Lemma 2.2 (univariate hierarchical splitting characterization) to prove that (2.22) holds if the corresponding univariate relation (2.20) holds for all dimensions:



LEMMA 2.4 (multivariate hierarchical splitting characterization)
Equation (2.22) *is equivalent to the satisfaction of both of the following conditions:*

- *The hierarchical subspaces* $W_{\boldsymbol{\ell}'}$ $(\boldsymbol{\ell}' \leq \boldsymbol{\ell})$ *are subspaces of* $V_{\boldsymbol{\ell}}$.
- *The basis functions* $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ $(\boldsymbol{\ell}' \leq \boldsymbol{\ell},\ \boldsymbol{i}' \in I_{\boldsymbol{\ell}'})$ *are linearly independent.*

PROOF If the sum is direct, then its dimension is given by

$$\dim \sum_{\boldsymbol{\ell}'=\mathbf{0}}^{\boldsymbol{\ell}} W_{\boldsymbol{\ell}'} = \sum_{\ell'_1=0}^{\ell_1} \cdots \sum_{\ell'_d=0}^{\ell_d} \prod_{t=1}^{d} \dim W_{\ell'_t} = \prod_{t=1}^{d} \sum_{\ell'_t=0}^{\ell_t} \dim W_{\ell'_t} = \prod_{t=1}^{d} (2^{\ell_t} + 1) = \dim V_{\boldsymbol{\ell}} \tag{2.23}$$

using (2.15). The rest is analogous to the proof of Lemma 2.2. ■

PROPOSITION 2.5 (from univariate to multivariate splitting)
If univariate splitting (2.20) *holds for every dimension, then the multivariate splitting* (2.22) *holds as well.*

PROOF We check the two conditions of Lemma 2.4 given the two univariate conditions of Lemma 2.2:

1. The hierarchical basis functions $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ of $W_{\boldsymbol{\ell}'}$ $(\boldsymbol{\ell}' \leq \boldsymbol{\ell},\ \boldsymbol{i}' \in I_{\boldsymbol{\ell}'})$ are tensor products of functions $\varphi_{\ell'_t,i'_t}$. According to the first condition of Lemma 2.2, each $\varphi_{\ell'_t,i'_t}$ can be written as a linear combination of the nodal basis φ_{ℓ_t,i_t} $(i_t = 0,\ldots,2^{\ell_t})$. We can expand the tensor product to a linear combination of tensor products of the univariate nodal basis functions. Therefore, $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ is a linear combination of multivariate nodal functions, i.e., $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'} \in V_{\boldsymbol{\ell}}$. As this is true for all $\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}$, we obtain $W_{\boldsymbol{\ell}'} \subseteq V_{\boldsymbol{\ell}}$.
2. The linear independence of the hierarchical functions $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ $(\boldsymbol{\ell}' \leq \boldsymbol{\ell},\ \boldsymbol{i}' \in I_{\boldsymbol{\ell}'})$ can be shown completely analogously to the proof of Lemma 2.1 (linear independence of tensor products).

According to Lemma 2.4, the multivariate splitting (2.22) holds. ■

A direct consequence of Prop. 2.5 is that the hierarchical splitting holds for the hierarchical hat function basis.

COROLLARY 2.6 (multivariate hierarchical splitting for hat functions)
The multivariate hierarchical splitting (2.22) *holds for the hat function basis.*

PROOF Follows directly by applying Cor. 2.3 (univariate hierarchical splitting for hat functions) to Prop. 2.5. ∎

2.3 Sparse Grids

The idea of sparse grids is to use the hierarchical splitting (2.22) to keep only the most important hierarchical subspaces, omitting the remaining ones. There are three main "flavors" of sparse grids: regular, dimensionally adaptive, and spatially adaptive.

IN THIS SECTION

2.3.1 Regular Sparse Grids

Hierarchical contributions. To assess the importance of a subspace, we consider again the interpolant $f_\ell \in V_\ell$ of a function $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$. According to the splitting (2.22), the interpolant can be written as

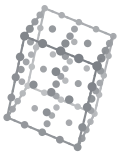
$$f_\ell = \sum_{\ell'=\mathbf{0}}^{\ell} \sum_{i' \in I_{\ell'}} \alpha_{\ell',i'} \varphi_{\ell',i'}, \quad \forall_{i=\mathbf{0},\dots,2^\ell} \; f_\ell(\boldsymbol{x}_{\ell,i}) = f(\boldsymbol{x}_{\ell,i}). \tag{2.24}$$

The coefficients $\alpha_{\ell',i'}$ with respect to the hierarchical basis $\varphi_{\ell',i'}$ are the *hierarchical surpluses*. When using the hat function basis $\varphi^1_{\ell,i}$, one can prove the following representation for the corresponding surpluses [Bun04; Gar13]:

$$\alpha_{\ell',i'} = (-1)^d 2^{-\|\ell'+\mathbf{1}\|_1} \int_{\mathbf{0}}^{\mathbf{1}} \varphi^1_{\ell',i'}(\boldsymbol{x}) \frac{\partial^{2d}}{\partial x_1^2 \cdots \partial x_d^2} f(\boldsymbol{x}) \, \mathrm{d}\boldsymbol{x}, \tag{2.25}$$

if $\boldsymbol{\ell} \geq \mathbf{1}$ and f is twice continuously differentiable in every dimension simultaneously, i.e., $\frac{\partial^{2d}}{\partial x_1^2 \cdots \partial x_d^2} f$ exists and is continuous.[3,4] Consequently, the contribution of the summand of

[3] Again, the notation implies that the integration domain is the unit hyper-cube $[\mathbf{0}, \mathbf{1}] = [0, 1]^d$.
[4] The statement is even valid for functions in the Sobolev space $H^2_{\text{mix}}([\mathbf{0}, \mathbf{1}])$ with dominating mixed derivative, as its proof mainly relies on integration by parts [Bun04; Gar13].



level $\boldsymbol{\ell}$ can be estimated by

$$\left\| \sum_{\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^1_{\boldsymbol{\ell}',\boldsymbol{i}'} \right\|_{L^2} \leq 3^{-d} \cdot 2^{-2\|\boldsymbol{\ell}\|_1} \cdot \left\| \frac{\partial^{2d}}{\partial x_1^2 \cdots \partial x_d^2} f \right\|_{L^2} \tag{2.26}$$

for the hat function surpluses $\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}$ [Bun04; Gar13].

Definition of regular sparse grids. Equation (2.26) motivates to omit those summands from the sum (2.24) whose level sum $\|\boldsymbol{\ell}\|_1$ exceeds a certain value $n \in \mathbb{N}_0$, as their contribution can be neglected compared to the summands with coarser level sums. More formally, the selection of the relevant subspaces can be formulated as a continuous knapsack problem [Bun04]. The resulting function space and grid point set

$$V^s_{n,d} := \bigoplus_{\|\boldsymbol{\ell}\|_1 \leq n} W_{\boldsymbol{\ell}}, \qquad \Omega^s_{n,d} := \dot{\bigcup_{\|\boldsymbol{\ell}\|_1 \leq n}} \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\} \tag{2.27}$$

are called *regular sparse grid space* and *regular sparse grid* of level n, respectively. The functions $f^s_{n,d}$ contained in $V^s_{n,d}$ have the form

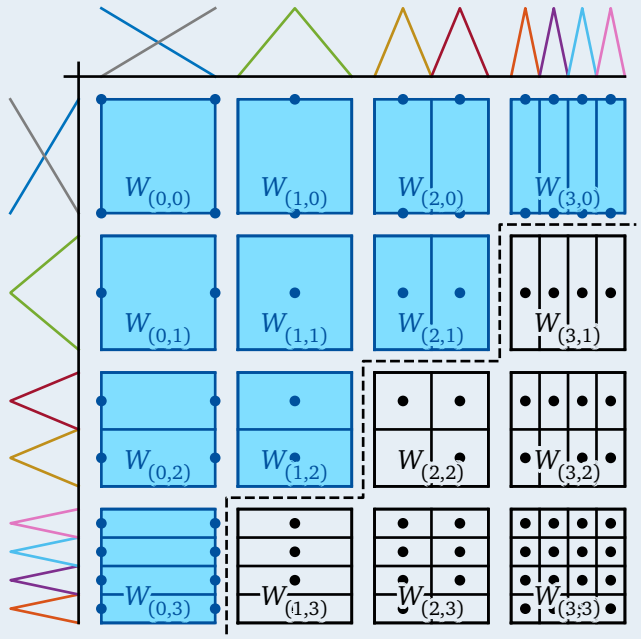
$$f^s_{n,d} = \sum_{\|\boldsymbol{\ell}\|_1 \leq n} \sum_{\boldsymbol{i} \in I_{\boldsymbol{\ell}}} \alpha_{\boldsymbol{\ell},\boldsymbol{i}} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}. \tag{2.28}$$

To better distinguish the different grids, we call the nodal spaces and grids *full grids*. We generalize the definition to arbitrary bases $\varphi_{\boldsymbol{\ell},\boldsymbol{i}}$, although sparse grids have been motivated using the hat function basis $\varphi^1_{\boldsymbol{\ell},\boldsymbol{i}}$ (the estimate (2.26) does not hold anymore in the general case). Figure 2.5 shows the construction of a regular sparse grid in two dimensions.

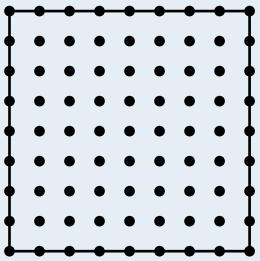
Grid size and interpolation error. One can prove that for homogeneous boundary conditions $f|_{\partial[\mathbf{0},\mathbf{1}]} \equiv 0$, the number of required inner grid points ($\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in \Omega^s_{n,d}$ where $\boldsymbol{\ell} \geq \mathbf{1}$) grows like $\mathcal{O}(h_n^{-1}(\log_2 h_n^{-1})^{d-1})$ [Bun04; Gar13], which is much less than the corresponding number $\mathcal{O}((h_n^{-1})^d)$ in the full grid case (see (2.15)). The L^2 error of the sparse grid interpolant $f^s_{n,d} \in V^s_{n,d}$ using hat functions (still assuming homogeneous boundary conditions) decays like

$$\|f - f^s_{n,d}\|_{L^2} = \mathcal{O}(h_n^2(\log_2 h_n^{-1})^{d-1}), \tag{2.29}$$

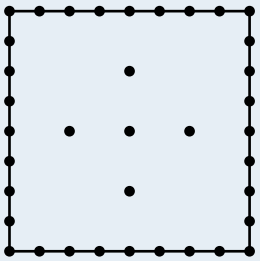
which is only slightly worse than the full grid error by the factor of $(\log_2 h_n^{-1})^{d-1}$ [Bun04; Gar13].



A Hierarchical splitting and subspace selection. The rectangles indicate the support of the bivariate hat basis functions.



B Full grid obtained by adding all subspaces of level $\boldsymbol{\ell} \leq n \cdot \mathbf{1}$.



C Regular sparse grid obtained by adding all subspaces whose level $\boldsymbol{\ell}$ satisfies $\|\boldsymbol{\ell}\|_1 \leq n$ *(blue)*.

FIGURE 2.5 Regular sparse grid of level $n = 3$ in two dimensions.

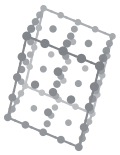
2.3.2 Dimensionally Adaptive Sparse Grids

The idea of dimensional adaptivity is to spend more grid points along specific dimensions depending on the objective function. Different criteria for the choice of dimensions exist, for example the maximal absolute value of the linear hierarchical surpluses. To incorporate dimensional adaptivity into sparse grids, one has to generalize the symmetric choice of subspaces in the definition of regular sparse grids to allow asymmetric preferences. Generally, function spaces V^s and grid sets Ω^s of *dimensionally adaptive sparse grids* have the form

$$V^s = \bigoplus_{\boldsymbol{\ell} \in L} W_{\boldsymbol{\ell}}, \qquad \Omega^s = \dot{\bigcup_{\boldsymbol{\ell} \in L}} \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}, \tag{2.30}$$

where L is a *downward closed* set, i.e., a finite subset $L \subseteq \mathbb{N}_0^d$ for which $\forall_{\boldsymbol{\ell} \in L} \forall_{\boldsymbol{\ell}' \leq \boldsymbol{\ell}} \boldsymbol{\ell}' \in L$. Regular sparse grids are a special case by setting $L = \{\boldsymbol{\ell} \in \mathbb{N}_0^d \mid \|\boldsymbol{\ell}\|_1 \leq n\}$.

Combination technique. The key advantage of dimensionally adaptive sparse grids over spatially adaptive approaches is the so-called *combination technique*. For regular



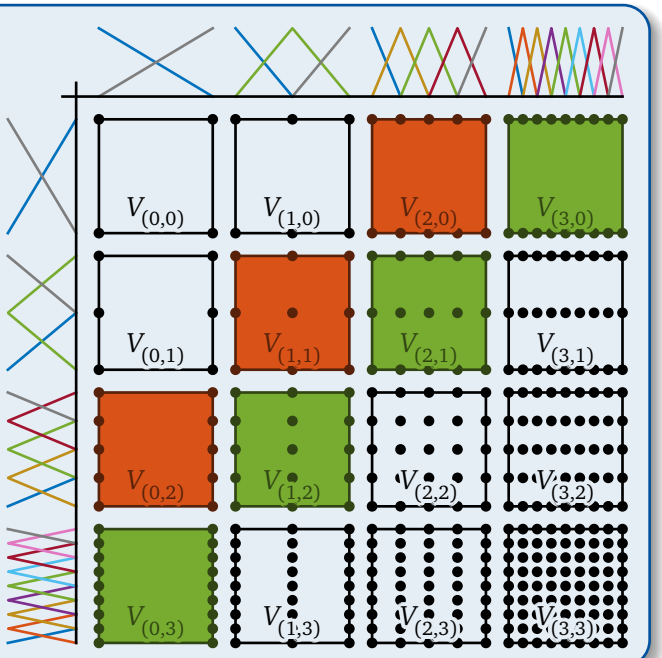


FIGURE 2.6
The combination technique combines nodal subspaces in a weighted sum to form a regular sparse grid space of level $n = 3$ in two dimensions. The red subspaces ($q = 1$ in (2.31)) are subtracted from the sum of the green subspaces ($q = 0$).

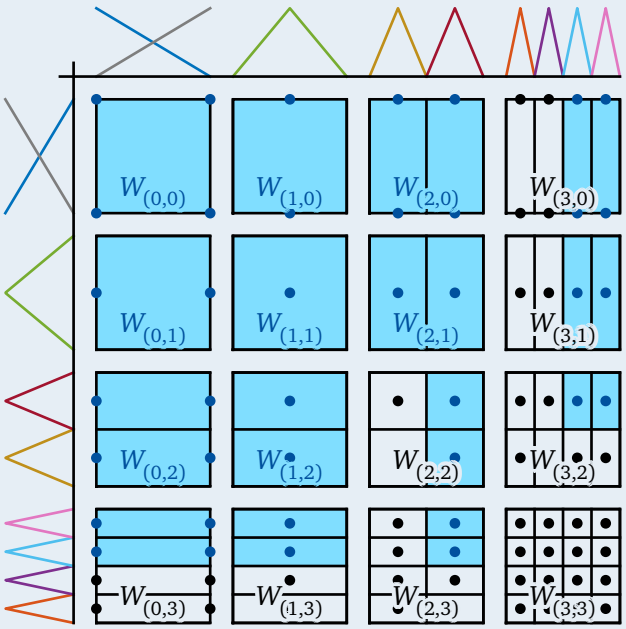
sparse grids, one can show that the sparse grid interpolant $f^s_{n,d}$ can be written as

$$f^s_{n,d} = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\boldsymbol{\ell}\|_1 = n-q} \sum_{\boldsymbol{i}=\mathbf{0}}^{2^\ell} c_{\boldsymbol{\ell},\boldsymbol{i}} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}, \tag{2.31}$$

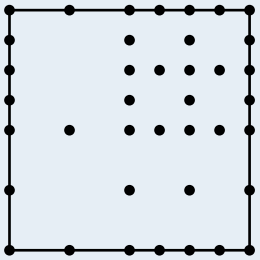
where the $c_{\boldsymbol{\ell},\boldsymbol{i}} \in \mathbb{R}$ ($\boldsymbol{i} = \mathbf{0}, \ldots, \mathbf{2}^\ell$) are the interpolation coefficients on the full grid $\Omega_{\boldsymbol{\ell}}$ of level $\boldsymbol{\ell}$, i.e., $\forall_{\boldsymbol{i}'=\mathbf{0},\ldots,\mathbf{2}^\ell} \sum_{\boldsymbol{i}=\mathbf{0}}^{2^\ell} c_{\boldsymbol{\ell},\boldsymbol{i}} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}'}) = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}'})$ [Smo63; Zen91]. For general dimensionally adaptive sparse grids, a similar formula exists [Nob16]. The combination formula (2.31) splits the sparse grid interpolant into a weighted sum of full grid interpolants (see Fig. 2.6). In applications, each grid can be processed in parallel, drastically speeding up computations like the solution of partial differential equations (PDEs) [Hee18]. In addition, existing code working on nodal bases does not have to be rewritten in terms of implementing hierarchical functions, which means that the combination technique allows sparse grids to be employed in existing software in a minimally invasive way.

2.3.3 Spatially Adaptive Sparse Grids

Dimensional adaptivity does not suffice to resolve local features of the objective function. Especially in some applications, it is crucial for the interpolant to be highly accurate in specific regions of the domain. For instance in optimization, it is not necessary to have a small global interpolation error. Instead, high accuracy near the optima is important.



A Hierarchical splitting and grid point selection. The rectangles indicate again the support of the bivariate hat basis functions.



B Resulting spatially adaptive sparse grid.

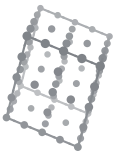
FIGURE 2.7 Spatially adaptive sparse grid in two dimensions. More grid points were generated in the top right corner, which can help to resolve fine oscillations of the objective function.

This can be achieved by *spatially adaptive sparse grids,* on which this thesis focuses. Generally, their function spaces V^s and grid sets Ω^s have the form

$$V^s = \operatorname{span}\{\varphi_{\ell,i} \mid (\ell, i) \in K\}, \qquad \Omega^s = \{x_{\ell,i} \mid (\ell, i) \in K\}, \tag{2.32}$$

where K is a finite set of level-index pairs (ℓ, i) with $\ell \in \mathbb{N}_0^d$ and $i \in I_\ell$. An example for a spatially adaptive sparse grid is shown in Fig. 2.7.

Algorithms for sparse grids often make specific assumptions about K. If they are not met, then the algorithms do not produce the correct results. For example when working with hat functions $\varphi^1_{\ell,i}$, the grid should contain the hierarchical ancestors of every grid point. Otherwise, the so-called unidirectional principle [Bal94], which is used for instance to efficiently calculate hierarchical surpluses, does not hold in general. However, as we will see in Chap. 4, the unidirectional principle cannot be applied to B-splines of general degree, even if the hierarchical ancestors exist. Hence, for most of our considerations, we will not restrict the choice of K.



2.4 Boundary Treatment

IN THIS SECTION

One issue of regular sparse grids $\Omega^s_{n,d}$ is that the number of grid points still grows very fast with the level n and the dimensionality d [Pfl10]. This is mainly because the finest mesh size h_n on the boundary of the domain $[\mathbf{0},\mathbf{1}]$ is finer than the finest mesh size h_{n-d+1} that can be found in the interior. If we define $\mathring{\Omega}^s_{n,d}$ as the set of interior grid points in $\Omega^s_{n,d}$,[5] i.e.,

$$\mathring{\Omega}^s_{n,d} := \Omega^s_{n,d} \cap]\mathbf{0},\mathbf{1}[= \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in \Omega^s_{n,d} \mid \boldsymbol{\ell} \geq \mathbf{1}\}, \tag{2.33}$$

then the following relation about the number of grid points of $\Omega^s_{n,d}$ can be shown:

LEMMA 2.7 (number of regular sparse grid points)

$$|\Omega^s_{n,d}| = \sum_{q=0}^{d} 2^q \binom{d}{q} |\mathring{\Omega}^s_{n,d-q}| \tag{2.34}$$

PROOF See [Bun04]. ∎

Here, we define zero-dimensional grids to contain exactly one grid point such that $|\mathring{\Omega}^s_{n,0}| = 1$. The number of interior grid points can be calculated as follows:

LEMMA 2.8 (number of interior regular sparse grid points)

$$|\mathring{\Omega}^s_{n,d}| = \sum_{q=0}^{n-d} 2^q \binom{d-1+q}{d-1} \tag{2.35}$$

PROOF See [Bun04]. ∎

Intuitively, Lemma 2.7 splits the sparse grid $\Omega^s_{n,d}$ into lower-dimensional sparse grids $\mathring{\Omega}^s_{n,d-q}$ of the same level, but without boundary points. The factor $2^q\binom{d}{q}$ is the number of $(d-q)$-dimensional faces of the d-dimensional unit hyper-cube. In the three-dimensional example of Fig. 2.8, the unit cube $[0,1]^3$ decomposes into

- $2^0\binom{3}{0} = 1$ interior cube $]0,1[^3$,
- $2^1\binom{3}{1} = 6$ sides (two-dimensional faces) like $]0,1[^2 \times \{0\}$,

[5] Note that in the literature (e.g., [Pfl10]), the regular sparse grid space of level n without boundary points is often defined via $\|\boldsymbol{\ell}\|_1 \leq n+d-1$ to ensure that the finest mesh size is given by h_n. In our notation, this corresponds to $\mathring{\Omega}^s_{n+d-1,d}$.

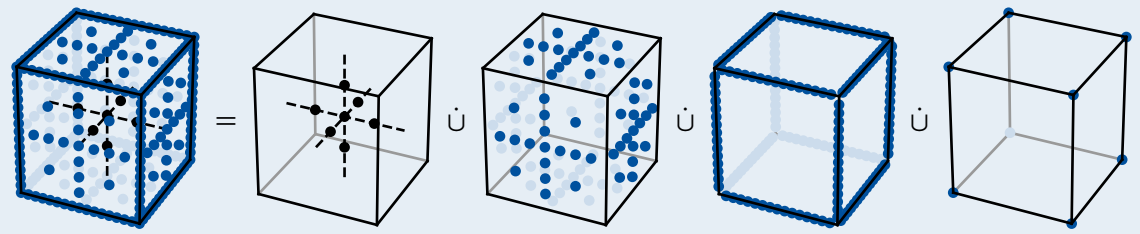


FIGURE 2.8 Decomposition of the three-dimensional sparse grid $\Omega^{\mathrm{s}}_{n,d}$ ($n = 4$, $d = 3$) into lower-dimensional sparse sub-grids. The main axes (axis-parallel lines through $0.5 \cdot \mathbf{1}$, *dashed*) serve as a visual aid.

- $2^2\binom{3}{2} = 12$ edges (one-dimensional faces) like $]0,1[\times \{(0,0)\}$, and
- $2^3\binom{3}{3} = 8$ corners (zero-dimensional faces) like $(0,0,0)$.

On each of these $(d-q)$-dimensional faces, the sparse grid $\Omega^{\mathrm{s}}_{n,d}$ contains the interior of a sparse grid of level n and dimensionality $d-q$, the size of which grows like $\mathcal{O}(2^n n^{d-q-1})$. As the number of boundary faces increases exponentially with the dimensionality d, the size of $\Omega^{\mathrm{s}}_{n,d}$ quickly exhausts the available computational memory. To deal with this issue, there are mainly two solutions, which are described below.

2.4.1 Sparse Grids with Coarser Boundaries

Inserting boundary points at higher levels. The first solution is to insert the boundary level functions and grid points at a higher level than at level zero. A popular choice is the insertion at level one, which corresponds to

$$\Omega^{\mathrm{s}(1)}_{n,d} := \dot{\bigcup_{\boldsymbol{\ell} \in L^{\mathrm{s}(1)}_{n,d}}} \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}, \quad L^{\mathrm{s}(1)}_{n,d} := \{\boldsymbol{\ell} \in \mathbb{N}_0^d \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n\}, \tag{2.36}$$

where **max** is to be read coordinate-wise as usual. This choice is equivalent to treating zero-level components as level one in the subspace selection. This ensures that the finest mesh sizes in the interior of $[\mathbf{0},\mathbf{1}]$ and on its boundary coincide to be h_{n-d+1}, which reduces the number of grid points on the boundary significantly.

Another solution that can be found in the literature of sparse grids with hat functions [Baa15] is to start with the "constant one" function on level zero with corresponding grid point 0.5, then employ the two boundary functions and points on level one, and finally proceed as usual for the higher levels ≥ 2. Apart from a constant shift of the resulting sparse grid levels, this is equivalent to inserting the boundary functions and points at level two. This solution leads to even less grid points than the previous approach, as now

the mesh size is finer in the interior of the domain than on the boundary. However, for very high dimensionalities this might still lead to computationally infeasible sparse grids.

Regular sparse grids with coarse boundary. We generalize these two solutions to the definition of a sparse grid $\Omega_{n,d}^{s(b)}$ that is equivalent to inserting the boundary functions and points at an arbitrary level $b \in \mathbb{N}$:

DEFINITION 2.9 (regular sparse grid with coarse boundary)
The regular sparse grid of level $n \in \mathbb{N}$, dimensionality $d \leq n$, and boundary parameter $b \in \mathbb{N}$ is defined as

$$\Omega_{n,d}^{s(b)} := \dot{\bigcup_{\boldsymbol{\ell} \in L_{n,d}^{s(b)}}} \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}, \tag{2.37a}$$

$$\begin{aligned} L_{n,d}^{s(b)} := &\{\boldsymbol{\ell} \in \mathbb{N}^d \mid \|\boldsymbol{\ell}\|_1 \leq n\} \\ &\dot{\cup} \left(\{\boldsymbol{\ell} \in \mathbb{N}_0^d \setminus \mathbb{N}^d \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n - b + 1\} \cup \{\mathbf{0}\}\right). \end{aligned} \tag{2.37b}$$

For convenience, we define $\Omega_{n,d}^{s(0)} := \Omega_{n,d}^{s}$.

The definition is motivated by partitioning the levels $\boldsymbol{\ell} \in \mathbb{N}_0^d$ into interior levels ($\boldsymbol{\ell} \in \mathbb{N}^d$) and boundary levels ($\boldsymbol{\ell} \in \mathbb{N}_0^d \setminus \mathbb{N}^d$). By including the levels of the interior grid $\mathring{\Omega}_{n,d}^{s}$, the mesh size in the interior is the same as before (h_{n-d+1}). Like in (2.36), we treat boundary levels as level one, but we subtract $b-1$ from the upper bound to ensure the correct mesh size $h_{n-d-b+2}$ on the boundary. We append $\mathbf{0}$ to the level set to ensure that at least the 2^d corner points are included in the resulting sparse grid. Note that this definition is consistent with (2.36) as $L_{n,d}^{s(b)} = \{\boldsymbol{\ell} \in \mathbb{N}_0^d \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n\}$ for $b = 1$. Examples of $\Omega_{n,d}^{s(b)}$ are shown in Fig. 2.9. The flip book animation in the bottom right corner of the odd-numbered pages of this thesis visualizes $\Omega_{n,d}^{s(b)}$ for $n = 4$, $d = 3$, and $b = 1$.

The number of grid points of $\Omega_{n,d}^{s(b)}$ can be calculated as follows:

PROPOSITION 2.10 (number of regular sparse grid points with coarse boundary)

$$|\Omega_{n,d}^{s(b)}| = |\mathring{\Omega}_{n,d}^{s}| + \sum_{q=1}^{d} 2^q \binom{d}{q} |\mathring{\Omega}_{n-q-b+1,d-q}^{s}|, \quad b \in \mathbb{N} \tag{2.38}$$

PROOF See Appendix A.1.1. ∎

As can be seen in Tab. 2.1 for three dimensions and in Tab. 2.2 for ten dimensions, the number of grid points decreases drastically for increasing values of b, especially when compared with $\Omega_{n,d}^{s} = \Omega_{n,d}^{s(0)}$.

	$\lvert\mathring{\Omega}^{s}_{n,d}\rvert$	$\lvert\Omega^{s(b)}_{n,d}\rvert/\lvert\mathring{\Omega}^{s}_{n,d}\rvert$					
		$b=0$	$b=1$	$b=2$	$b=3$	$b=4$	$b=5$
$n=3$	1	123.0	27.0	9.00	9.00	9.00	9.00
$n=4$	7	42.4	11.6	4.71	2.14	2.14	2.14
$n=5$	31	22.7	7.3	3.39	1.84	1.26	1.26
$n=6$	111	14.9	5.3	2.75	1.67	1.23	1.07
$n=7$	351	10.9	4.3	2.37	1.55	1.21	1.07
$n=8$	1023	8.5	3.6	2.13	1.47	1.19	1.07
$n=9$	2815	7.0	3.2	1.96	1.41	1.17	1.07
$n=10$	7423	6.0	2.9	1.83	1.36	1.16	1.06

TABLE 2.1 For $d = 3$: Grid size of the interior grid $\mathring{\Omega}^{s}_{n,d}$ *(second column)* and ratios $\lvert\Omega^{s(b)}_{n,d}\rvert/\lvert\mathring{\Omega}^{s}_{n,d}\rvert$ *(beginning with third column)* of the sizes of the grid $\Omega^{s(b)}_{n,d}$ with boundary points to the size of the interior grid of the same level. The table starts with the first level $n = 3$ for which the interior grid $\mathring{\Omega}^{s}_{n,d}$ is not empty.

	$\lvert\mathring{\Omega}^{s}_{n,d}\rvert$	$\lvert\Omega^{s(b)}_{n,d}\rvert/\lvert\mathring{\Omega}^{s}_{n,d}\rvert$					
		$b=0$	$b=1$	$b=2$	$b=3$	$b=4$	$b=5$
$n=10$	1	$3.3\cdot10^{8}$	59 049	1025	1025.0	1025.0	1025.0
$n=11$	21	$4.3\cdot10^{7}$	21 558	2813	49.8	49.8	49.8
$n=12$	241	$1.0\cdot10^{7}$	10 046	1879	246.0	5.2	5.2
$n=13$	2001	$3.4\cdot10^{6}$	5407	1211	227.2	30.5	1.5
$n=14$	13 441	$1.3\cdot10^{6}$	3213	806	181.1	34.7	5.4
$n=15$	77 505	$6.2\cdot10^{5}$	2054	558	140.6	32.2	6.8
$n=16$	397 825	$3.1\cdot10^{5}$	1390	401	109.5	28.2	7.1
$n=17$	1 862 145	$1.7\cdot10^{5}$	984	298	86.5	24.2	6.8

TABLE 2.2 For $d = 10$: Grid size of the interior grid $\mathring{\Omega}^{s}_{n,d}$ *(second column)* and ratios $\lvert\Omega^{s(b)}_{n,d}\rvert/\lvert\mathring{\Omega}^{s}_{n,d}\rvert$ *(beginning with third column)* of the sizes of the grid $\Omega^{s(b)}_{n,d}$ with boundary points to the size of the interior grid of the same level. The table starts with the first level $n = 10$ for which the interior grid $\mathring{\Omega}^{s}_{n,d}$ is not empty.

```
1  function L_{n,d}^{s(b)} = computeSGCoarseBoundary(n, d, b)
2    L^(1) ← {0, 1, ..., n−d+1}                          ⇝ one-dimensional grid
3    for t = 2, ..., d do
4      L^(t) ← ∅                                          ⇝ t-dimensional grid
5      for ℓ ∈ L^(t−1) do
6        if ‖max(ℓ, 1)‖_1 ≤ n−d+t−b or ℓ = 0 then
7          L^(t) ← L^(t) ∪ {(ℓ, 0)}                     ⇝ add corners (with (ℓ,0) := (ℓ_1, ..., ℓ_{t−1}, 0))
8        if ℓ ∈ ℕ^{t−1} then
9          ℓ* ← n−d+t−‖ℓ‖_1                             ⇝ add interior points
10       else
11         ℓ* ← n−d+t−b+1−‖max(ℓ, 1)‖_1                 ⇝ add boundary points
12       L^(t) ← L^(t) ∪ {(ℓ, ℓ_t) | ℓ_t = 1, ..., ℓ*}   ⇝ with (ℓ, ℓ_t) := (ℓ_1, ..., ℓ_{t−1}, ℓ_t)
13   L_{n,d}^{s(b)} ← L^(d)
```

ALGORITHM 2.1 Generation of the sparse grid $\Omega_{n,d}^{s(b)}$ with coarse boundary. Inputs are the level $n \in \mathbb{N}$, the dimensionality $d \leq n$, and the boundary parameter $b \in \mathbb{N}$. Output is the level set $L_{n,d}^{s(b)}$ that corresponds to $\Omega_{n,d}^{s(b)}$.

Algorithm 2.1 shows how to generate the necessary set of hierarchical levels. Its correctness can be formally proven with the following invariant:

PROPOSITION 2.11 (invariant of SG generation with coarse boundary)
After iteration t of Alg. 2.1 ($t = 1, \ldots, d$), it holds

$$L^{(t)} = \{\boldsymbol{\ell} \in \mathbb{N}^t \mid \|\boldsymbol{\ell}\|_1 \leq n-d+t\} \,\dot\cup\, \left(\{\boldsymbol{\ell} \in \mathbb{N}_0^t \setminus \mathbb{N}^t \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n-d+t-b+1\} \cup \{\mathbf{0}\}\right). \tag{2.39}$$

PROOF See Appendix A.1.2. ∎

COROLLARY 2.12 *Algorithm 2.1 is correct.*

PROOF Follows immediately from Prop. 2.11 by setting $t = d$, as then (2.39) becomes (2.37b) from Def. 2.9 (regular sparse grid with coarse boundary). ∎

Hierarchization and other algorithms. An important implication of the regular sparse grids $\Omega_{n,d}^{s(b)}$ as defined in Def. 2.9 is that, in general, the unidirectional principle cannot be directly applied anymore. For example, this is relevant when calculating hierarchical surpluses for the hat function basis. As we mostly deal with B-splines, for which the unidirectional principle cannot be applied even on regular sparse grids, this issue is not important in the scope of this thesis.

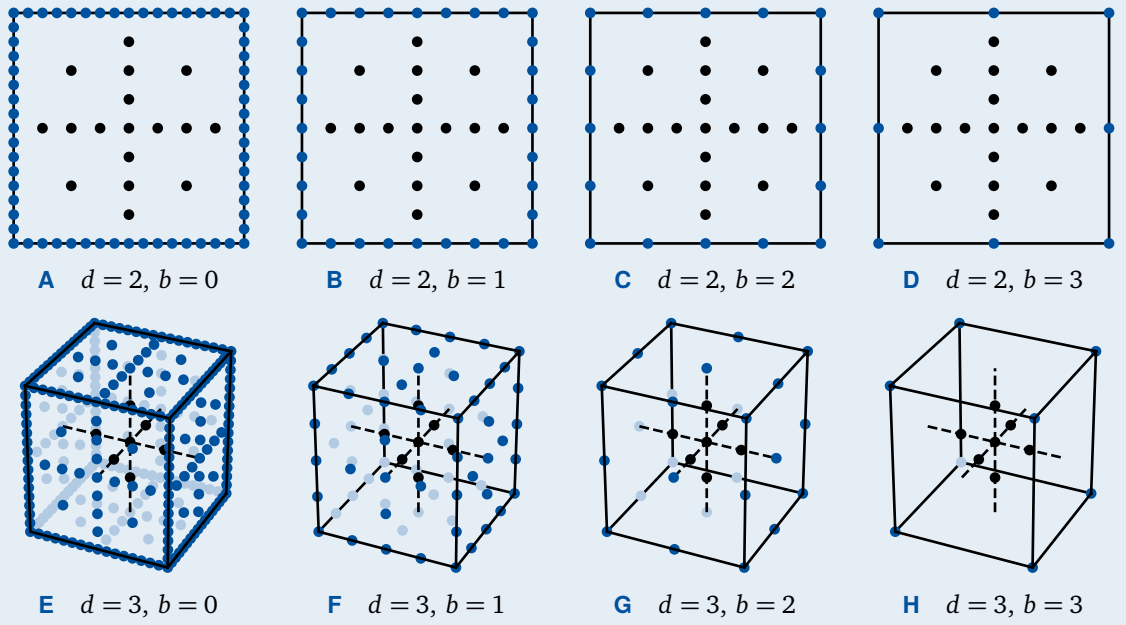


FIGURE 2.9 Sparse grids $\Omega_{n,d}^{s(b)}$ of level $n = 3$ in two and three dimensions for different values of the boundary parameter b. For constant d and n, the points in the interior of $[\mathbf{0}, \mathbf{1}]$ (black) are the same, while the points on the boundary of $[\mathbf{0}, \mathbf{1}]$ *(blue)* become coarser for increasing values of b. The main axes (axis-parallel lines through $0.5 \cdot \mathbf{1}$, *dashed*) serve as a visual aid.

However, it is possible to calculate the hierarchical surpluses of hat functions on $\Omega_{n,d}^{s(b)}$ in a three-step algorithm. First, we compute the surpluses of the boundary grid $\Omega_{n,d}^{s(b)} \setminus \mathring{\Omega}_{n,d}^{s}$. Second, we subtract the values of the resulting "boundary interpolant" at the inner grid points $\mathring{\Omega}_{n,d}^{s}$. Third, we calculate the surpluses of the inner grid points as usual with the unidirectional principle. As the corresponding "inner interpolant" vanishes on the boundary, this does not influence the interpolated values in the first step.

2.4.2 Sparse Grids Without Boundary Points and Modified Bases

Omitting boundary points. The second solution to reduce the number of grid points on the boundary is to omit the boundary points and the basis functions altogether. For the hat function basis $\varphi_{\ell,i}^{1}$, this is a feasible option if the objective function $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$ satisfies homogeneous boundary conditions $f|_{\partial[\mathbf{0},\mathbf{1}]} \equiv 0$, as $\varphi_{\ell,i}^{1}$ vanishes on the boundary if and only if $\boldsymbol{\ell} \geq \mathbf{1}$, i.e., if the basis function corresponds to an inner grid point. Consequently, the surpluses corresponding to boundary points vanish for a grid with boundary points and homogeneous boundary conditions, implying that these points can be removed from the grid.

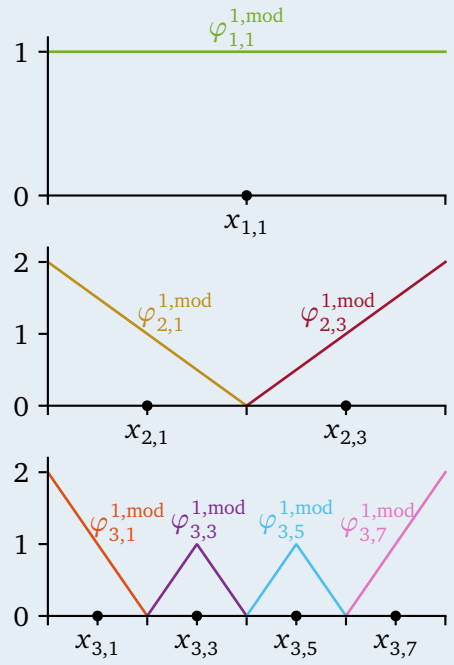


FIGURE 2.10
Modified hierarchical hat functions $\varphi^{1,\mathrm{mod}}_{\ell',i'}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$) up to level $\ell = 3$.

Modified linear basis. Of course, this approach is not viable for functions with non-zero boundary values or general hierarchical bases, making it necessary to change the basis. For hat functions, Pflüger modified the leftmost and rightmost univariate basis function of each level (with indices $i = 1$ and $i = 2^\ell - 1$ respectively) such that the modified functions extrapolate the inner values linearly towards the boundary [Pfl10]. The basis function on level one is replaced by the "constant one" function. All other basis functions remain unchanged. The resulting *modified hat functions* $\varphi^{1,\mathrm{mod}}_{\ell,i} : [0,1] \to \mathbb{R}$ are shown in Fig. 2.10 and defined as follows:

$$\varphi^{1,\mathrm{mod}}_{\ell,i}(x) := \begin{cases} 1, & \ell = 1, \quad i = 1, \\ \max(2 - \frac{x}{h_\ell}, 0), & \ell \geq 2, \quad i = 1, \\ \varphi^1_{\ell,i}(x), & \ell \geq 2, \quad i \in I_\ell \setminus \{1, 2^\ell - 1\}, \\ \varphi^{1,\mathrm{mod}}_{\ell,1}(1-x), & \ell \geq 2, \quad i = 2^\ell - 1. \end{cases} \tag{2.40}$$

The modified linear basis provides "reasonable" boundary values without the need to insert basis functions and grid points on the boundary. For other bases such as B-splines, similar modifications are possible, which we will discuss when we introduce the corresponding unmodified functions (see Chapters 3 and 4).

3

Hierarchical B-Splines

> *B-splines are not enough!*
>
> — In a talk at the 2017 SIAM Conference on Computational Science and Engineering

Piecewise linear "hat functions" $\varphi^1_{\ell,i}$, which served in the previous chapter as the motivation to define sparse grids for arbitrary tensor product basis functions, are not continuously differentiable. This has two implications. First, the approximation order of hat functions is lower than the order of other basis function types such as higher-degree splines [Sic11] or the piecewise polynomial basis by Bungartz [Bun98]. Second, we cannot compute globally continuous gradients of the interpolant of a smooth objective function if we use non-smooth basis functions[1]. However, the availability of continuous gradients is crucial in gradient-based optimization, which is highly important in the scope of simulation technology (see Chap. 1) and which we target in this thesis. In this chapter, we define the hierarchical and higher-order B-spline basis as a generalization of the well-known hat functions to obtain both higher-order approximations and continuous derivatives.

The first to study B-splines was Schoenberg in 1946 [Schoenb46], but he claimed that they had already been known to Laplace [Boor76]. It was also Schoenberg who

[1] ... which include the hat function basis as well as the piecewise polynomials by Bungartz.

coined the term “B-splines,” which is short for “basis splines” [Schoenb67]. De Boor pioneered B-splines, developed basic algorithms, and proved fundamental theoretical results [Boor72]. Research and industry recognized the possibilities of B-splines when the finite element method (FEM) emerged in the 1960s. The FEM remains one of the driving forces behind the research of B-splines [Höl03] as the recent rise of isogeometric analysis (IGA) shows [Cot09; Höl12]. Researchers have also applied B-splines to geometric modeling with non-uniform rational B-splines (NURBS) [Coh01; Höl13], financial mathematics [Pfl10], molecular and atomic physics [Bac01; McC04], and numerous other scientific and industrial areas [Vale12]; [Mar17]. Theoretical and practical aspects of B-splines on sparse grids have also been studied before [Pan08; Pfl10; Sic11; Vale16].

In this chapter, we define hierarchical B-splines on sparse grids. The chapter is divided into two sections: First, we define hierarchical B-splines for both uniform and non-uniform knot sequences in Sec. 3.1. Second, we learn in Sec. 3.2 that the boundary behavior of the standard uniform B-spline basis is problematic. Incorporating not-a-knot boundary conditions into the B-spline basis mitigates the problems caused by the boundary behavior.

Section 3.1.1 is a repetition of the definition of nodal B-splines [Höl03; Höl13] and hierarchical B-splines [Pfl10; Vale14]. Original contributions of the thesis in this chapter are the proof of the linear independence of hierarchical B-splines in Sec. 3.1.2 (improved version of [Vale14], published in [Vale16]), the modified hierarchical Clenshaw-Curtis B-splines in Sec. 3.1.4, and the hierarchical not-a-knot B-spline basis in Sec. 3.2.

3.1 Uniform and Non-Uniform Hierarchical B-Splines

IN THIS SECTION

In this section, we mainly follow the presentation of [Pfl10; Vale14; Vale16] to define hierarchical B-splines starting from the well-known nodal B-spline basis [Höl03; Höl13; Qua16]. Note that thanks to the groundwork laid in Chap. 2, especially Lemma 2.1 (linear independence of tensor products) and Prop. 2.5 (from univariate to multivariate splitting), it suffices to study the univariate case of all bases that will be defined in the rest of this thesis. The multivariate case is treated canonically by tensor products.

3.1.1 Uniform Hierarchical B-Splines

Cardinal B-splines. The *cardinal B-spline* $b^p : \mathbb{R} \to \mathbb{R}$ of *degree* $p \in \mathbb{N}_0$ is defined by

$$b^p(x) := \begin{cases} \int_0^1 b^{p-1}(x-y)\,\mathrm{d}y, & p \geq 1, \\ \chi_{[0,1[}(x), & p = 0, \end{cases} \tag{3.1}$$

where $\chi_{[0,1[}$ is the characteristic function of the half-open unit interval $[0,1[$ (see [Höl13]). The cardinal B-spline b^p has the following properties [Höl03], which are shown in Fig. 3.1:

1. *Compact support:* The support of b^p is given by $\operatorname{supp} b^p = [0, p+1]$.
2. *Bounds and symmetry:* The cardinal B-spline b^p is non-negative and bounded from above by one. It is symmetric with respect to $x = \frac{p+1}{2}$, i.e., $b^p(x) = b^p(p+1-x)$.
3. *Recursion:* The cardinal B-spline b^p $(p \geq 1)$ satisfies the following recurrence relation (which can be used as an alternative definition):

$$b^p(x) = \frac{x}{p} b^{p-1}(x) + \frac{p+1-x}{p} b^{p-1}(x-1). \tag{3.2}$$

4. *Spline:* On every *knot interval* $[k, k+1[$ for $k = 0, \ldots, p$, b^p is a polynomial of degree p, i.e., b^p is a spline of degree p (piecewise polynomial).
5. *Derivative:* At the *knots* $k = 0, \ldots, p+1$, b^p is $(p-1)$ times continuously differentiable (if $p \geq 1$). The derivative can be computed by differentiating (3.1):

$$\frac{\mathrm{d}}{\mathrm{d}x} b^p(x) = b^{p-1}(x) - b^{p-1}(x-1), \quad x \in \mathbb{R}. \tag{3.3}$$

6. *Integral:* The B-spline b^p has unit integral, i.e., $\int_{\mathbb{R}} b^p(x)\,\mathrm{d}x = 1$.
7. *Convolution:* The integral in the definition of b^p is the convolution $b^{p-1} * b^0$ of the B-spline b^{p-1} of degree $p-1$ with the B-spline b^0 of degree zero.
8. *Generalization:* As a special case, b^1 is a hat function, interpolating the data $\{(k, \delta_{k,1}) \mid k \in \mathbb{Z}\}$. For $p \to \infty$, the normalized cardinal B-splines converge pointwise to the standard Gaussian function $b^\infty(x) := (2\pi)^{-1/2} \exp(-x^2/2)$ [Uns92]:[2]

$$\lim_{p\to\infty} c^p b^p(c^p x + \tfrac{p+1}{2}) = b^\infty(x), \quad c^p := \sqrt{\frac{p+1}{12}}, \quad x \in \mathbb{R}. \tag{3.4}$$

[2] This can also be seen as a consequence of the central limit theorem applied to uniformly distributed random variables. The pointwise convergence of the probability density functions can be proven from the convergence in distribution using a converse to Scheffé's theorem [Boos85].

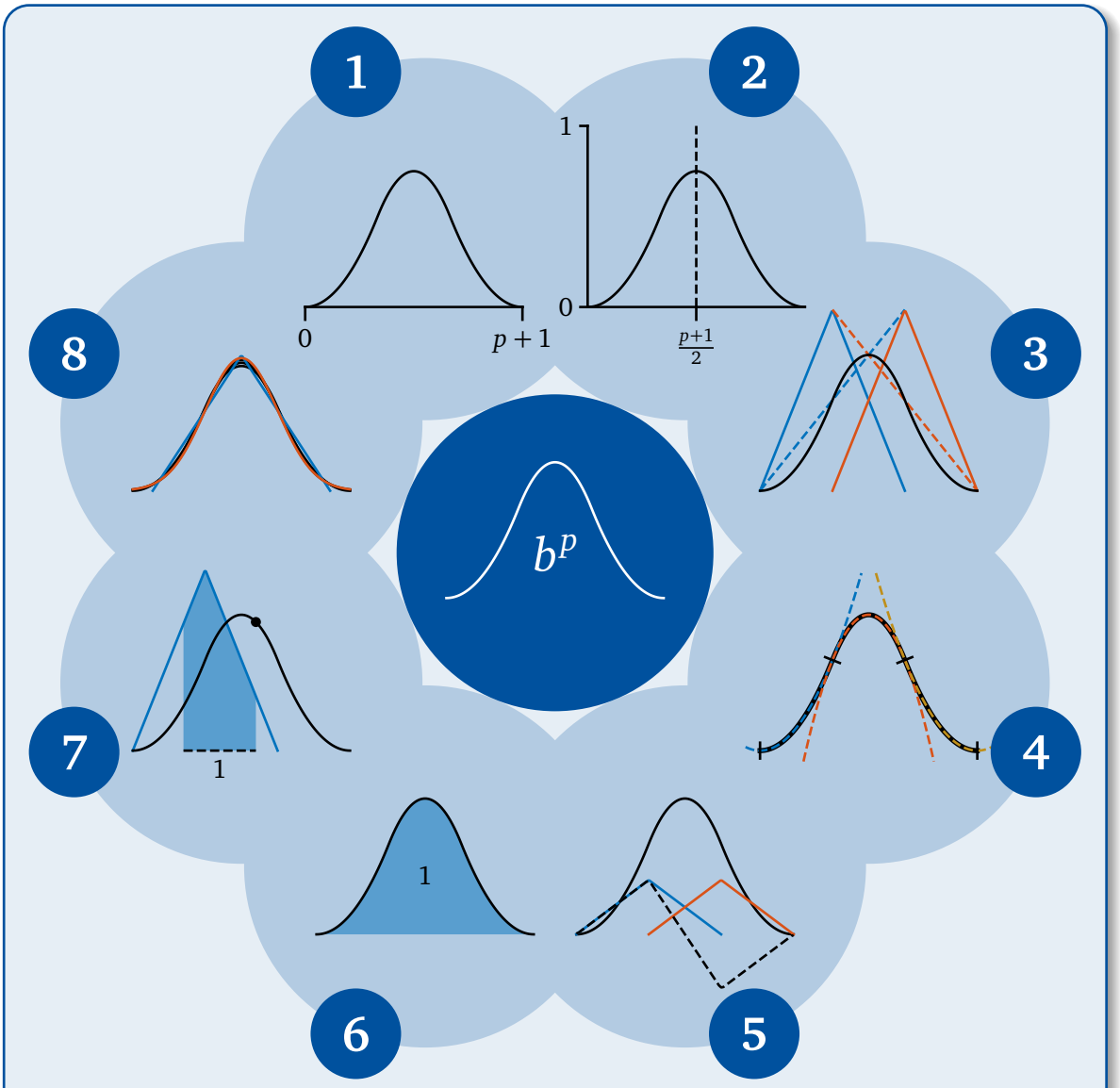


FIGURE 3.1 Eight properties of cardinal B-splines using the quadratic case $p = 2$ as an example.

1. b^p is compactly supported on $[0, p+1]$.
2. b^p is symmetric and $0 \leq b^p \leq 1$.
3. b^p is a weighted combination of b^{p-1} *(blue)* and $b^{p-1}(\cdot - 1)$ *(red)*.
4. b^p is a piecewise polynomial of degree p.
5. $\frac{\mathrm{d}}{\mathrm{d}x} b^p$ *(dashed)* is the difference of b^{p-1} *(blue)* and $b^{p-1}(\cdot - 1)$ *(red)*.
6. b^p has unit integral.
7. b^p is the convolution of b^{p-1} *(blue)* and b^0.
8. Hat function *(blue)* and Gaussian function *(red)* are special cases of b^p.

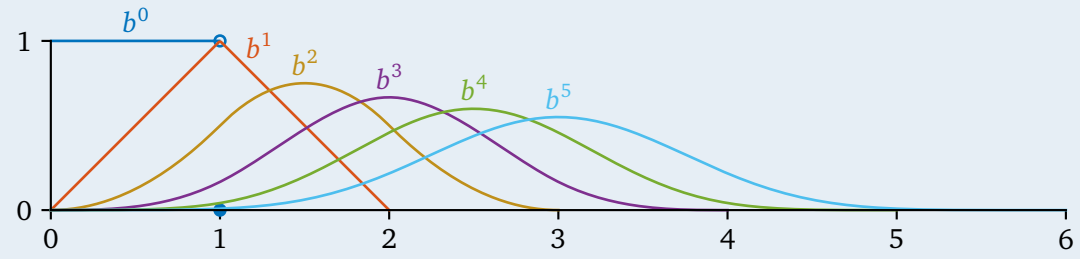


FIGURE 3.2 Cardinal B-splines b^p up to quintic degree $p = 5$.

The cardinal B-splines of the first degrees are shown in Fig. 3.2. Due to the convolution property, cardinal B-splines of degree $p \geq 2$ are "smoothed versions" of the hat function. This is shown in the flip book animation in the bottom left corner of the even-numbered pages of this thesis.

Definition of uniform hierarchical B-splines. As for the hat functions in Chap. 2, we can use the cardinal B-spline b^p as a "parent function" to define the uniform hierarchical B-spline $\varphi^p_{\ell,i} : [0,1] \to \mathbb{R}$ of level $\ell \in \mathbb{N}_0$ and index $i \in I_\ell$ via an affine parameter transformation [Pfl10]:

$$\varphi^p_{\ell,i}(x) := b^p\left(\tfrac{x}{h_\ell} + \tfrac{p+1}{2} - i\right). \tag{3.5}$$

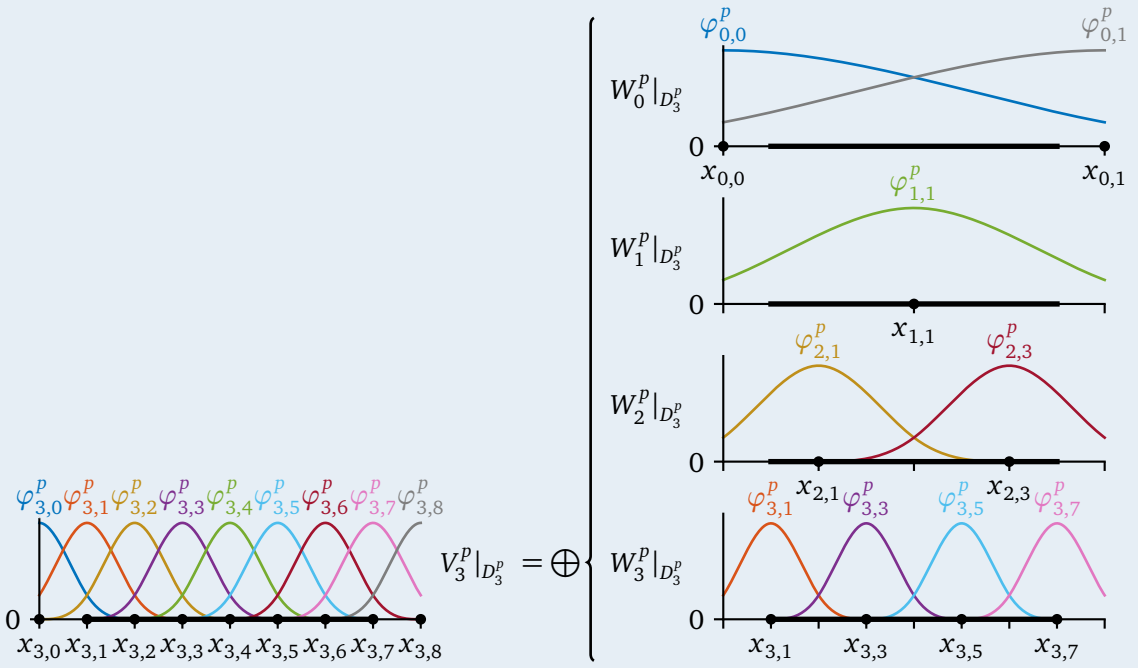
The support of $\varphi^p_{\ell,i}$ is given by $\operatorname{supp} \varphi^p_{\ell,i} = [0,1] \cap [x_{\ell,i-(p+1)/2}, x_{\ell,i+(p+1)/2}]$. The hat function basis $\varphi^1_{\ell,i}$ defined in (2.3) is a special case of (3.5) for $p = 1$, which allows us to use the same notation $\varphi^p_{\ell,i}$ for both. Note that due to the *translation invariance* of $\varphi^p_{\ell,i}$ (i.e., the basis functions are the same up to scaling and translation), it suffices to precompute and implement the polynomial pieces of b^p to enable evaluations of all hierarchical B-splines $\varphi^p_{\ell,i}$ ($\ell \in \mathbb{N}_0$, $i \in I_\ell$).

Odd and even degrees. In this thesis, we will only allow odd degrees $p = 1, 3, 5, \ldots$ for hierarchical B-splines. Many theoretical considerations fail for even degrees. The basic reason is that for odd degrees, the knots of $\varphi^p_{\ell,i}$ coincide with the grid points [Vale14]

$$x_{\ell,i-(p+1)/2}, \quad \ldots, \quad x_{\ell,i}, \quad \ldots, \quad x_{\ell,i+(p+1)/2}. \tag{3.6}$$

For even degrees p, the knots of $\varphi^p_{\ell,i}$ lie exactly in the middle between two subsequent grid points:

$$x_{\ell,i-p/2} - \frac{h_\ell}{2}, \quad \ldots, \quad x_{\ell,i} - \frac{h_\ell}{2}, \quad x_{\ell,i} + \frac{h_\ell}{2}, \quad \ldots, \quad x_{\ell,i+p/2} + \frac{h_\ell}{2}. \tag{3.7}$$



A Nodal B-splines $\varphi^p_{\ell,i}$ $(i \in I_\ell)$ and grid points $x_{\ell,i}$ *(dots)*.

B Hierarchical B-splines $\varphi^p_{\ell',i'}$ $(\ell' \le \ell, i' \in I_{\ell'})$ and grid points $x_{\ell',i'}$ *(dots)*.

FIGURE 3.3 Univariate nodal and hierarchical cubic B-splines ($p = 3$) up to level $\ell = 3$. When restricting all functions to D^p_ℓ *(thick black line)*, the nodal space $V^p_\ell|_{D^p_\ell}$ decomposes into the direct sum of the hierarchical subspaces $W^p_{\ell'}|_{D^p_\ell}$ $(\ell' \le \ell)$.

This fact has many adverse implications on the hierarchical basis. The most crucial implication is that for even degrees p, the hierarchical splitting (2.20) does not hold. Furthermore, we would not be able to define non-uniform hierarchical B-splines as simple as for odd degrees and fundamental splines would not be defined at all (as we will see in Sec. 4.4.3). Additionally, odd degrees include the hat function case ($p = 1$) and the most commonly applied cubic degree ($p = 3$). Therefore, it is reasonable to restrict ourselves to odd degrees for the rest of the thesis.

3.1.2 Non-Uniform B-Splines and Proof of the Hierarchical Splitting

Non-uniform B-splines and spline space. With the hierarchical B-splines $\varphi^p_{\ell,i}$, we can define the nodal spaces V^p_ℓ and hierarchical subspaces W^p_ℓ as in Chap. 2. However, in order for the hierarchical splitting (2.20) to be correct, we have to prove that the conditions of Lemma 2.2 (univariate hierarchical splitting characterization) are satisfied. To investigate how the nodal space V^p_ℓ looks like, we introduce the notion of non-uniform B-splines.

DEFINITION 3.1 (non-uniform B-splines)
Let $m, p \in \mathbb{N}_0$ and $\xi = (\xi_0, \dots, \xi_{m+p})$ be an increasing sequence of real numbers *(knot sequence)*. For $k = 0, \dots, m-1$, the *(non-uniform) B-spline* $b^p_{k,\xi}$ of degree p with knots ξ and index k is defined by the Cox–de Boor recurrence [Cox72; Boor72; Höl13]

$$b^p_{k,\xi}(x) := \begin{cases} \dfrac{x - \xi_k}{\xi_{k+p} - \xi_k} b^{p-1}_{k,\xi}(x) + \dfrac{\xi_{k+p+1} - x}{\xi_{k+p+1} - \xi_{k+1}} b^{p-1}_{k+1,\xi}(x), & p \geq 1, \\ \chi_{[\xi_k, \xi_{k+1}[}(x), & p = 0. \end{cases} \tag{3.8}$$

Note that when choosing $\xi = (0, 1, \dots, p+1)$ and $k = 0$, we obtain the cardinal B-spline b^p. Definition 3.1 can be used to characterize the nodal space V^p_ℓ:

PROPOSITION 3.2 (spline space)
Let $\xi = (\xi_0, \dots, \xi_{m+p})$ be a knot sequence. Then, the B-splines $b^p_{k,\xi}$ $(k = 0, \dots, m-1)$ form a basis of the spline space

$$S^p_\xi := \operatorname{span}\{b^p_{k,\xi} \mid k = 0, \dots, m-1\}. \tag{3.9}$$

S^p_ξ contains exactly those functions that are continuous on $D^p_\xi := [\xi_p, \xi_m]$, polynomials of degree $\leq p$ on every knot interval $[\xi_k, \xi_{k+1}[$ in D^p_ξ $(k = p, \dots, m-1)$ and at least $(p-1)$ times continuously differentiable at every knot ξ_k in the interior of D^p_ξ $(k = p+1, \dots, m-1)$.

PROOF See [Höl13]. ∎

This proposition gives the reason for the letter "B" in "B-splines," which stands for "basis" (of the space of splines) [Schoenb67]. One example of a knot sequence and the corresponding B-splines is given in Fig. 3.4. The key observation is that B-splines of a knot sequence ξ do not form a basis of the spline space on the union $[\xi_0, \xi_{m+p}]$ of the B-spline supports. Instead, they form a basis of the spline space on a proper sub-interval D^p_ξ. Intuitively, for every point in D^p_ξ that is not a knot, exactly $p+1$ B-splines must be *relevant* (i.e., non-zero) to uniquely span the spline space, as on every knot interval, the spline is a polynomial of degree $\leq p$ and therefore, there must be $p+1$ degrees of freedom. Outside of D^p_ξ, there are too few relevant B-splines to span the spline space. This fact, which is shown in Fig. 3.5, forces us to restrict the nodal space and the hierarchical subspaces to D^p_ξ:

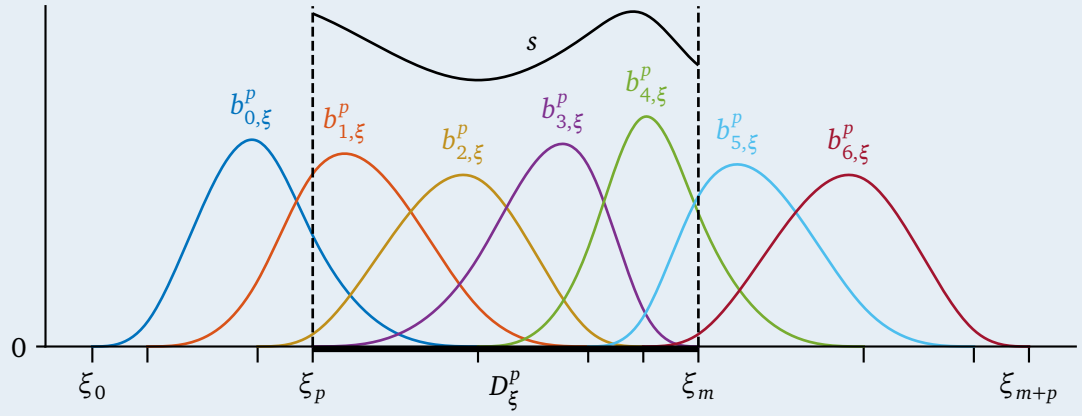


FIGURE 3.4 Knot sequence $\boldsymbol{\xi} = (\xi_0, \ldots, \xi_{m+p})$ with the corresponding $m = 7$ non-uniform cubic B-splines $b^p_{k,\xi}$ $(k = 0, \ldots, m-1,\ p = 3)$. On D^p_ξ *(thick line, delimited by dashed lines),* which starts with the last knot interval of the first B-spline $b^p_{0,\xi}$ and ends with the first knot interval of the last B-spline $b^p_{m-1,\xi}$, the B-splines span the spline space S^p_ξ. Elements of this space are splines $s\colon D^p_\xi \to \mathbb{R}$ *(black line),* which are linear combinations $s = \sum_{k=0}^{m-1} c_k b^p_{k,\xi}$ of the B-splines.

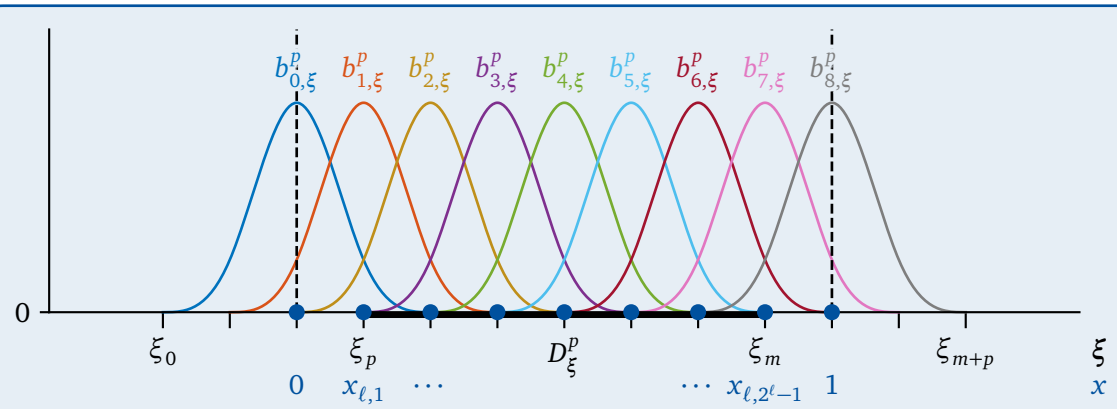


FIGURE 3.5 Uniform knot sequence ξ^p_ℓ *(ticks on horizontal axis)* and corresponding nodal cubic B-splines $(p = 3)$ of level $\ell = 3$. In the domain $[0, 1]$ *(delimited by dashed lines),* the grid points Ω_ℓ *(blue dots)* coincide with the B-spline knots. The spline interpolation domain $D^p_\ell := D^p_{\xi^p_\ell}$ *(thick line)* is only a proper subset of $[0, 1]$.

COROLLARY 3.3 (nodal B-spline space)
The restricted nodal B-splines $\varphi^p_{\ell,i}|_{D^p_\ell}$ $(i = 0, \dots, 2^\ell)$ *of level* $\ell \in \mathbb{N}_0$ *are a basis of the spline space* $S^p_{\xi^p_\ell}$, *where*

$$\xi^p_{\ell,k} := (k - \tfrac{p+1}{2})h_\ell, \quad k = 0, \dots, m+p, \quad m := 2^\ell + 1, \tag{3.10a}$$

$$D^p_\ell := [\tfrac{p-1}{2}h_\ell,\ 1 - \tfrac{p-1}{2}h_\ell], \tag{3.10b}$$

and consequently

$$V^p_\ell|_{D^p_\ell} = S^p_\ell := S^p_{\xi^p_\ell}. \tag{3.11}$$

PROOF We have $\varphi^p_{\ell,i} = b^p_{i,\xi^p_\ell}$ for $i = 0, \dots, m-1$, as the B-splines on both sides have the same knots. The assertions now follow from Prop. 3.2 (spline space). ∎

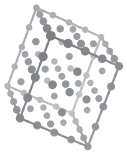
Note that D^p_ℓ might contain only a single point or even be empty, if p is too large or ℓ is too small. However, the corresponding B-splines $\varphi^p_{\ell,i}$ are still linearly independent on $[0,1]$ (see [Höl13]). Similarly, the corollary also implies that the hierarchical functions $\varphi^p_{\ell,i}$ of level ℓ $(i \in I_\ell)$ are linearly independent on $[0,1]$.

Hierarchical splitting for uniform B-splines. We can use Prop. 3.2 and Cor. 3.3 to prove the hierarchical splitting for the uniform B-spline basis [Vale16].

LEMMA 3.4 (hierarchical B-splines in nodal space)
The restricted hierarchical subspaces $W^p_{\ell'}|_{D^p_\ell}$ $(\ell' \le \ell)$ *are subspaces of the restricted nodal space* $V^p_\ell|_{D^p_\ell}$.

PROOF Every function $\varphi^p_{\ell',i'}$ $(i' \in I_{\ell'})$ is continuous on D^p_ℓ, a polynomial of degree $\le p$ on every knot interval of ξ^p_ℓ (due to p odd), and at the knots themselves at least $(p-1)$ times continuously differentiable. Proposition 3.2 implies $\varphi^p_{\ell',i'} \in S^p_\ell$ and from Cor. 3.3, it follows $\varphi^p_{\ell',i'} \in V^p_\ell|_{D^p_\ell}$. As the functions $\varphi^p_{\ell',i'}$ $(i' \in I_{\ell'})$ span $W^p_{\ell'}|_{D^p_\ell}$, we can conclude $W^p_{\ell'}|_{D^p_\ell} \subseteq V^p_\ell|_{D^p_\ell}$. ∎

It is crucial to note that this lemma does not hold for even p, as the knots of the B-splines of level $\ell - 1$ are not contained in the knots of level ℓ. This implies that in general, $W^p_{\ell-1}|_{D^p_\ell}$ is not contained in $V^p_\ell|_{D^p_\ell}$ for even degrees p. Therefore, the hierarchical splitting equation (2.20) does not hold.



PROPOSITION 3.5 (hierarchical B-splines are linearly independent)
The hierarchical B-splines $\varphi^p_{\ell',i'}$ $(\ell' \leq \ell,\ i' \in I_{\ell'})$ *are linearly independent.*

PROOF See Appendix A.2.1. ∎

Although we have to restrict all functions and spaces to D^p_ℓ, Lemma 2.2 (univariate hierarchical splitting characterization) is still applicable to prove that the hierarchical splitting equation (2.20) is correct for hierarchical B-splines:

COROLLARY 3.6 (hierarchical splitting for uniform B-splines)
The hierarchical splitting (2.20) *holds for the hierarchical B-spline basis if restricting all functions to* D^p_ℓ*:*

$$S^p_\ell = V^p_\ell|_{D^p_\ell} = \bigoplus_{\ell'=0}^{\ell} W^p_{\ell'}|_{D^p_\ell}. \tag{3.12}$$

PROOF Analogously to the proof of Lemma 2.2 (univariate hierarchical splitting characterization) and apply Cor. 3.3 (nodal B-spline space). ∎

This corollary is also visualized in Fig. 3.3. We can now proceed to define multivariate nodal spaces V^p_ℓ, multivariate hierarchical subspaces W^p_ℓ, and sparse grid spaces $V^{s,p}_{n,d}$ as in Chap. 2. Note that it is possible to choose different degrees p_t for different dimensions $t = 1,\dots,d$, since the hierarchical splitting (2.22) does not require the bases in each dimension to be the same. Consequently, we can define degree-dimension-adaptive (so-called *hp-adaptive*) sparse grids $V^{s,\boldsymbol{p}}_{n,d}$ for arbitrary odd degree vectors $\boldsymbol{p}$.

In the course of this thesis, we will derive multiple variations of the standard hierarchical B-spline basis. We will not repeat formal proofs of the hierarchical splitting equation (2.20) (i.e., verifying the two conditions of Lemma 2.2) for each of theses bases for the sake of brevity. The idea of the proof of Prop. 3.5, which is inductively exploiting the smoothness conditions given by B-splines of previous levels, can be transferred to similar B-spline bases.

3.1.3 Modification

Marsden's identity. Similar to the piecewise linear case in Sec. 2.4.2, Pflüger defined modified hierarchical B-splines to obtain reasonable values on the boundary without having to place grid points there [Pfl10]. The main motivation is to define basis functions

$\varphi_{\ell,i}^{p,\mathrm{mod}}$ that satisfy natural boundary conditions, i.e.,

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi_{\ell,i}^{p,\mathrm{mod}}(x) = 0, \quad x \in \partial[0,1] = \{0,1\}. \tag{3.13}$$

Originally, this requirement stems from financial problems [Pfl10]. For the left boundary, (3.13) can be satisfied by modifying the left-most function $\varphi_{\ell,1}^{p}$ such that $\varphi_{\ell,1}^{p,\mathrm{mod}}(x) = 2 - \frac{x}{h_\ell}$ is a linear polynomial when x is "near" the boundary. As in [Pfl10], we append $\varphi_{\ell,1}^{p}$ with B-splines $\varphi_{\ell,i}^{p}$ with index $i \leq 0$ and use *Marsden's identity* to compute the corresponding coefficients. The identity enables us to explicitly compute the B-spline coefficients of an arbitrary polynomial of degree $\leq p$ by giving an explicit formula for the B-spline coefficients of shifted monomials $(\cdot - y)^p$. Interestingly, the coefficients are polynomials themselves (in y):

LEMMA 3.7 (Marsden's identity)
Let $p \in \mathbb{N}_0$ and $\boldsymbol{\xi} = (\xi_0, \ldots, \xi_{m+p})$ be a knot sequence. Then, for all $x \in D_{\boldsymbol{\xi}}^{p}$ and $y \in \mathbb{R}$,

$$(x-y)^p = \sum_{k=0}^{m-1} \left[(\xi_{k+1} - y) \cdots (\xi_{k+p} - y)\right] b_{k,\boldsymbol{\xi}}^{p}(x). \tag{3.14}$$

PROOF See [Höl13]. ∎

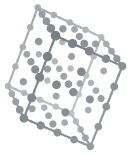
Modified hierarchical B-splines. By extending the sum in Marsden's identity to $i \in \mathbb{Z}$ and comparing the coefficients of y^p and y^{p-1} of both sides, we obtain representations for the first two monomials:

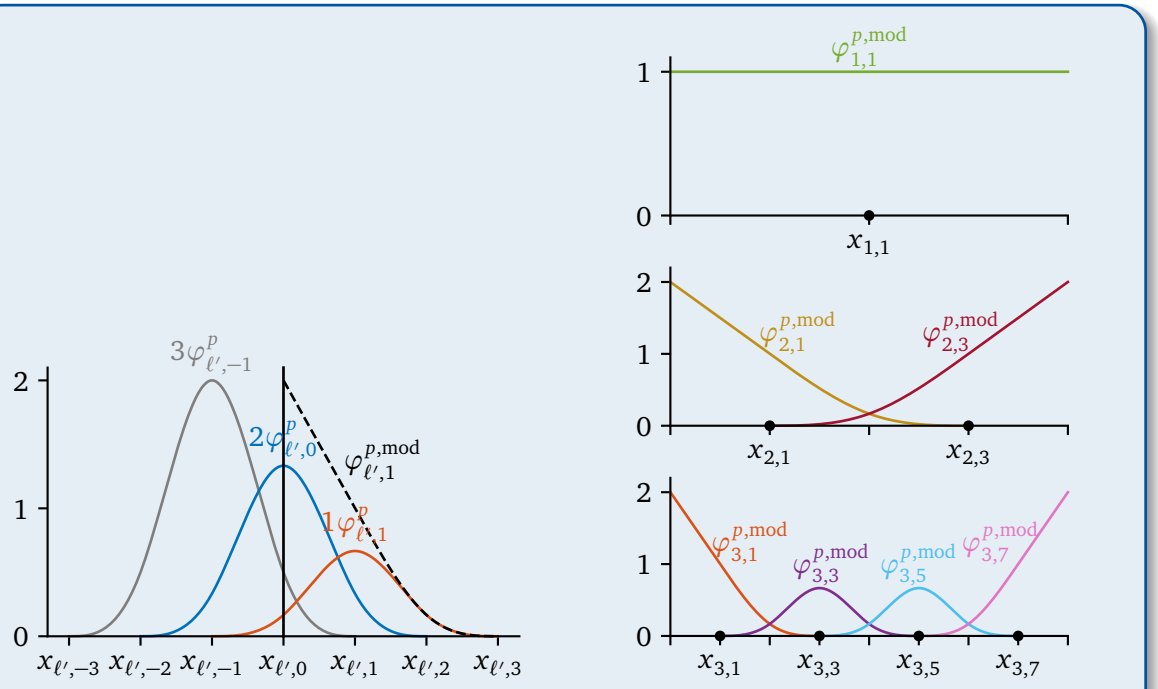
$$1 = \sum_{i\in\mathbb{Z}} \varphi_{\ell,i}^{p}(x), \quad x = \sum_{i\in\mathbb{Z}} x_{\ell,i}\varphi_{\ell,i}^{p}(x), \quad x \in \mathbb{R}. \tag{3.15}$$

This can be used to derive a definition for $\varphi_{\ell,i}^{p,\mathrm{mod}}$:

$$2 - \frac{x}{h_\ell} = 2\sum_{i\in\mathbb{Z}} \varphi_{\ell,i}^{p}(x) - \frac{1}{h_\ell}\sum_{i\in\mathbb{Z}} x_{\ell,i}\varphi_{\ell,i}^{p}(x) = \sum_{i\in\mathbb{Z}} (2-i)\varphi_{\ell,i}^{p}(x), \quad x \in \mathbb{R}. \tag{3.16}$$

Note that only the B-splines with $i \geq 1 - \frac{p+1}{2}$ are relevant for the unit interval as all other B-splines vanish in $[0,1]$. Pflüger omits summands with $i > 1$ as he only wants to modify $\varphi_{\ell,1}^{p}$ left of its grid point $x_{\ell,1}$ [Pfl10]. The right-most function $\varphi_{\ell,2^\ell-1}^{p,\mathrm{mod}}$ can be derived analogously by mirroring $\varphi_{\ell,1}^{p,\mathrm{mod}}$ at $x = \frac{1}{2}$. For $\ell = 1$, again the "constant one"





A Construction of the modified hierarchical cubic B-spline $\varphi^{p,\mathrm{mod}}_{\ell',1}$ (*dashed*, $\ell' \geq 2$) as a linear combination of neighboring B-splines $\varphi^p_{\ell',i'}$.

B Modified hierarchical cubic B-splines $\varphi^{p,\mathrm{mod}}_{\ell',i'}$ ($\ell' = 1,\dots,\ell$, $i' \in I_{\ell'}$) and grid points $x_{\ell',i'}$ *(dots)*.

FIGURE 3.6 Construction of modified hierarchical cubic B-splines ($p = 3$) and the resulting basis up to level $\ell = 3$.

function is taken for the definition of modified hierarchical B-splines (see Fig. 3.6):

$$\varphi^{p,\mathrm{mod}}_{\ell,i}(x) := \begin{cases} 1, & \ell = 1, \quad i = 1, \\ \displaystyle\sum_{i'=1-(p+1)/2}^{1} (2-i')\varphi^p_{\ell,i'}(x), & \ell \geq 2, \quad i = 1, \\ \varphi^p_{\ell,i}(x), & \ell \geq 2, \quad i \in I_\ell \setminus \{1, 2^\ell - 1\}, \\ \varphi^{p,\mathrm{mod}}_{\ell,1}(1-x), & \ell \geq 2, \quad i = 2^\ell - 1. \end{cases} \tag{3.17}$$

By Prop. 3.2 (spline space), this definition implies that $\varphi^p_{\ell,1}(x) = 2 - \frac{x}{h_\ell}$ ($\ell \geq 2$) is only valid for $x \in [0, \frac{5-p}{2} h_\ell]$, i.e., the second derivative at $x = 0$ vanishes only for $p \leq 5$. For higher degrees, it is non-zero, albeit very small in its absolute value. To enforce natural boundary conditions for higher than quintic degrees, the upper bound of i' in the sum in (3.17) must be extended to $\frac{p+1}{2}$. In addition, more than just the left-most and the right-most inner hierarchical B-spline must be modified for $p \geq 5$, as the size of the supports of $\varphi^p_{\ell,i}$ increases for growing p.

3.1.4 Non-Uniform Hierarchical B-Splines

Sparse grid spaces and their corresponding grid point sets, as we have defined them in Chap. 2, are completely independent of the actual location of the grid points $x_{\ell,i}$. Therefore, it is possible to use different distributions for the grid points than the standard equidistant choice of $x_{\ell,i} = i \cdot h_\ell$, if the basis functions are altered accordingly [Vale14]. The so-called Chebyshev points $x^{\text{cc}}_{\ell,i}$ and the resulting Clenshaw–Curtis grids and B-splines will serve as an example.

Chebyshev points and Clenshaw–Curtis grids. The *Chebyshev points* $x^{\text{cc}}_{\ell,i}$ of level $\ell \in \mathbb{N}_0$ are defined as

$$x^{\text{cc}}_{\ell,i} := \frac{1-\cos(\pi x_{\ell,i})}{2}, \quad i = 0,\ldots,2^\ell, \tag{3.18}$$

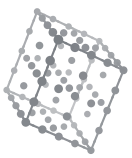
see [Xu16] for example. Chebyshev points are the locations of the extrema[3] of the Chebyshev polynomials T_{2^ℓ}, defined as $T_{2^\ell} : [0,1] \to \mathbb{R}$, $T_{2^\ell}(x) := \cos(2^\ell \arccos(2x-1))$. They are geometrically obtained by dividing a semicircle into 2^ℓ equally-sized segments and subsequently orthogonally projecting the segment endpoints onto the diameter (see Fig. 3.7). Analytically, they are determined by minimizing the interpolation error term in polynomial interpolation. The most practical use of Chebyshev points is in polynomial interpolation to avoid Runge's phenomenon and in numerical integration (quadrature), resulting in the so-called Clenshaw–Curtis quadrature rules.

In some settings, it may be beneficial to use full or sparse grids consisting of Chebyshev points, which are then called *Clenshaw–Curtis grids,* instead of uniform grids. Besides the already mentioned advantages for polynomials and quadrature, Clenshaw–Curtis grids can help to reduce interpolation errors in a neighborhood of the boundary of the domain due to the increased grid point density near the boundary (at the cost of a lower grid point density in the interior). If we want to use Chebyshev points as grid points for sparse grids, we have to employ an appropriate basis to ensure that interpolation is still possible.

Hierarchical Clenshaw–Curtis B-splines. The *hierarchical Clenshaw–Curtis B-spline* $\varphi^{p,\text{cc}}_{\ell,i}$ of level $\ell \in \mathbb{N}_0$ and index $i \in I_\ell$ is defined as [Vale14]

$$\varphi^{p,\text{cc}}_{\ell,i} := b^p_{i,\xi^{p,\text{cc}}_\ell}, \tag{3.19}$$

[3] The literature sometimes uses the name "Chebyshev points" for the roots of T_{2^ℓ}, which are closely connected with the extrema. One way to distinguish them is to call the extrema "Chebyshev–Lobatto points" and the roots "Chebyshev–Gauss points" [Xu16].



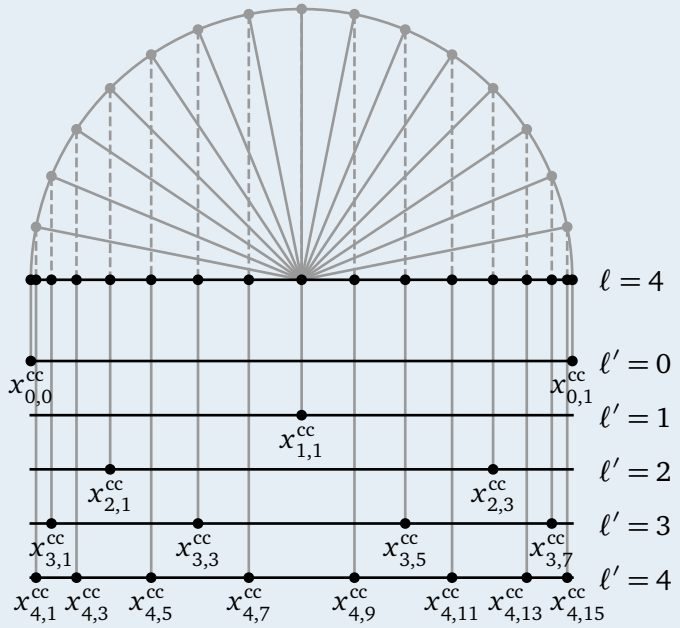


FIGURE 3.7
The set of Chebyshev points Ω^{cc}_{ℓ} of level $\ell = 4$ *(top)* decomposes into hierarchical grids of level $\ell' \le \ell$ (compare with Fig. 2.3). The Chebyshev points are constructed as the orthogonal projection of the endpoints of 2^{ℓ} equally-sized segments of a semicircle onto its diameter *(gray, top)*.

where

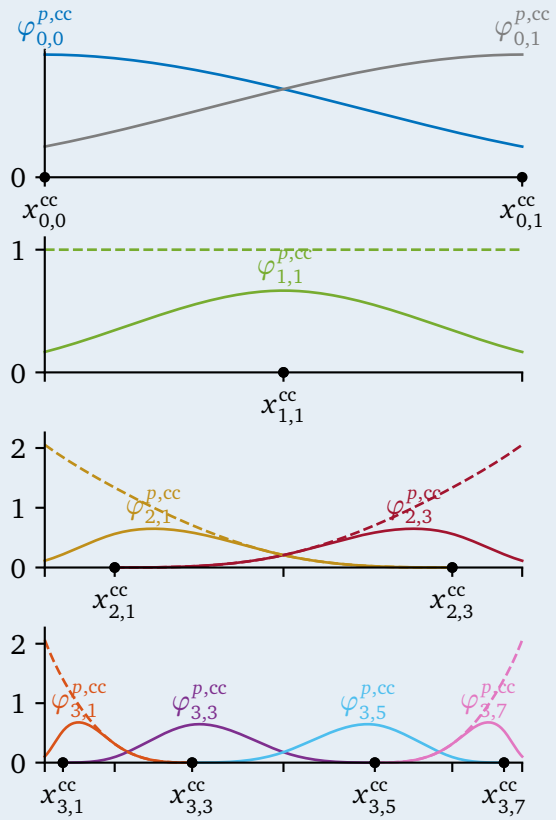
$$\xi^{p,cc}_{\ell,k} := \begin{cases} (k - \frac{p+1}{2}) \cdot x^{cc}_{\ell,1}, & k = 0, \dots, \frac{p-1}{2}, \\ x^{cc}_{\ell,k-(p+1)/2}, & k = \frac{p+1}{2}, \dots, 2^{\ell} + \frac{p+1}{2}, \\ 1 + (k - 2^{\ell} - \frac{p+1}{2}) \cdot x^{cc}_{\ell,1}, & k = 2^{\ell} + \frac{p+3}{2}, \dots, 2^{\ell} + p + 1, \end{cases} \tag{3.20a}$$

$$k = 0, \dots, m + p, \quad m := 2^{\ell} + 1. \tag{3.20b}$$

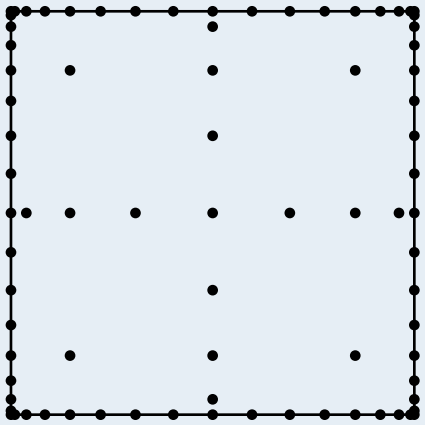
For the construction of the knot sequence $\xi^{p,cc}_{\ell}$, the Chebyshev points $x^{cc}_{\ell,i}$ are equidistantly extended onto the real line $\mathbb{R}$. As for the standard hierarchical B-spline basis, it is now straightforward to define nodal spaces $V^{p,cc}_{\ell}$ and hierarchical subspaces $W^{p,cc}_{\ell}$ as well as sparse grid spaces $V^{s,p,cc}_{n,d}$ and grid point sets $\Omega^{s,cc}_{n,d}$ using tensor products of Clenshaw–Curtis B-splines and Cartesian products of Chebyshev point sets. The one-dimensional cubic Clenshaw–Curtis basis and a sparse Clenshaw–Curtis grid are shown in Fig. 3.8.

Note that Clenshaw–Curtis B-splines $\varphi^{p,cc}_{\ell,i}$ are not translation-invariant, in contrast to uniform B-splines $\varphi^{p}_{\ell,i}$. As a result, both implementation effort and computation time for evaluation are significantly higher for Clenshaw–Curtis B-splines than for uniform B-splines, as the Clenshaw–Curtis B-splines cannot be precomputed.

Modification and generalizations. We define *modified hierarchical Clenshaw–Curtis B-splines* $\varphi^{p,cc,mod}_{\ell,i}$ using the same method as in Eq. (3.17). Here, the second derivative of $\varphi^{p,cc,mod}_{\ell,1}$ does not vanish at $x = 0$ even for degrees $p \le 5$, as the formula (3.16) derived



A Hierarchical cubic Clenshaw–Curtis B-splines $\varphi^{p,cc}_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$, $p = 3$) and modified Clenshaw–Curtis B-splines $\varphi^{p,cc,mod}_{\ell',i'}$ *(dashed)*.

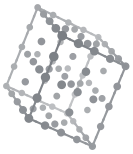


B Sparse Clenshaw–Curtis grid $\Omega^{s,cc}_{n,d}$ of level $n = 4$ in $d = 2$ dimensions.

FIGURE 3.8 Clenshaw–Curtis B-splines and sparse grids.

from Lemma 3.7 (Marsden's identity) assumes uniform knots. However, since most of the B-spline knots in the summation formula lie outside $[0,1]$ and are thus uniform according to Eq. (3.20), the absolute deviation of the second derivative from zero is small. Again, to enforce natural boundary conditions, we can recompute the coefficients of the components $\varphi^{p,cc}_{\ell,i'}$ dynamically with Marsden's identity using the correct Chebyshev knots in Eq. (3.14).

Note that our framework permits arbitrary grid point distributions $x^*_{\ell,i}$, as long as two requirements are met: First, their number should grow exponentially (i.e., there are $2^\ell + 1$ points $x^*_{\ell,i}$ in each level $\ell \in \mathbb{N}_0$), and second, they should be nested (i.e., Eq. (2.16) holds). Appropriate non-uniform B-spline bases can be defined analogously to Clenshaw–Curtis B-splines.



3.2 Boundary Behavior of Hierarchical B-Splines

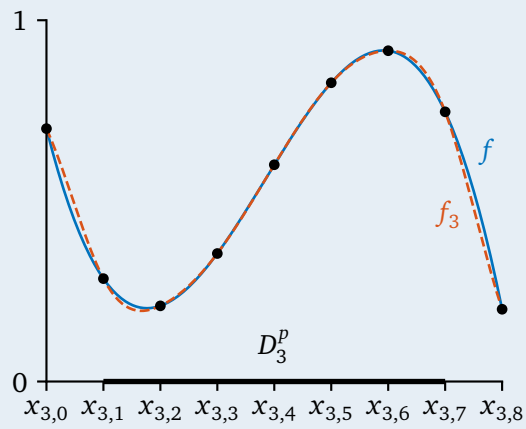
As we have seen in the last section (see Cor. 3.6), the hierarchical splitting equation (2.20) only holds when restricting the function spaces to $D_\ell^p = [\frac{p-1}{2}h_\ell,\ 1-\frac{p-1}{2}h_\ell]$, which is a proper subset of the domain $[0,1]$ if $p>1$. As we will see, the implications of this fact on the approximation quality of the hierarchical B-spline basis are severe. In this section, we study the underlying reasons of the restriction and we introduce a new hierarchical B-spline basis that does not suffer from this issue.

3.2.1 Approximation Quality of Uniform Hierarchical B-Splines

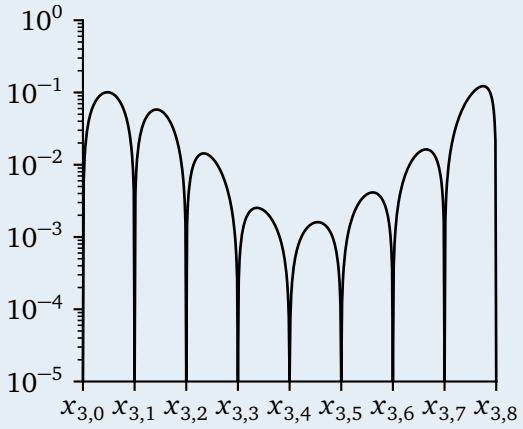
Interpolation of polynomials. Splines are a piecewise generalization of polynomials. Approximation spaces spanned by splines of degree p should at least contain all polynomials of degree $\leq p$. Unfortunately, this statement is not true for uniform B-splines $\varphi_{\ell,i}^p$ as we have defined them in the last section. A counterexample is given in Fig. 3.9, in which a cubic polynomial f is interpolated with hierarchical cubic B-splines. We can clearly see deviations of the interpolant from the polynomial near the boundary, where the pointwise relative error exceeds 10 %. The oscillations are even visible in the interior of the spline interpolation domain D_ℓ^p. Obviously, this phenomenon impairs the approximation quality for other non-polynomial functions as well.

This issue can be explained as follows: According to Cor. 3.6 (hierarchical splitting for uniform B-splines), we have $S_\ell^p = \bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p|_{D_\ell^p}$ with $p=3$. Since cubic polynomials are also cubic splines, it follows $f \in S_\ell^p$ and hence $f \in \bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p|_{D_\ell^p}$. This means that there is a linear combination of hierarchical B-splines $\varphi_{\ell',i'}^p$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$) that replicates f on the whole domain D_ℓ^p (not be confused with f_ℓ in Fig. 3.9, which does not replicate f exactly on D_ℓ^p). However, in general, this interpolant is not equal f outside of D_ℓ^p (i.e., in $[0,1]\setminus D_\ell^p$), as Prop. 3.2 (spline space) only holds for D_ℓ^p. In particular, the interpolant evaluated at $x \in \{0,1\}$ is not equal to $f(x)$. If we now force the additional interpolation conditions in $x_{\ell,0}=0$ and $x_{\ell,2^\ell}=1$, the resulting interpolant f_ℓ cannot be the same as the previous interpolant, which is why f and f_ℓ differ inside D_ℓ^p.

Schoenberg–Whitney conditions. Formally, the unique existence of an interpolating spline is described by the *Schoenberg–Whitney conditions:*



A Objective function f *(blue)*, interpolant f_ℓ *(red, dashed)*, grid points *(dots)*, and spline interpolation domain D_ℓ^p *(thick line)*.



B Pointwise relative error $|(f - f_\ell)/f|$ on a logarithmic scale.

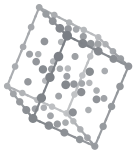
FIGURE 3.9 Hierarchical cubic B-splines $\varphi_{\ell',i'}^p$ ($\ell' \le \ell$, $i' \in I_{\ell'}$, $p = 3$) fail to replicate a cubic polynomial f (here: $f(x) := -10.2x^3 + 14.7x^2 - 5x + 0.7$) when interpolating on the grid of level $\ell = 3$.

PROPOSITION 3.8 (Schoenberg–Whitney conditions)
Let $\xi = (\xi_0, \dots, \xi_{m+p})$ be a knot sequence and $t_0, \dots, t_{m-1}$ a sequence of interpolation points with $t_0 < \cdots < t_{m-1}$ and $\xi_p \le t_0 < t_{m-1} \le \xi_m$. Then, there exists a unique interpolating spline $s = \sum_{k=0}^{m-1} c_k b_{k,\xi}^p$ for arbitrary data if and only if

$$\xi_k < t_k < \xi_{k+p+1}, \quad k = 0, \dots, m-1. \tag{3.21}$$

PROOF See [Höl13]. ■

The Schoenberg–Whitney conditions require that the interpolation points are contained in D_ℓ^p, which is not the case for $p = 3$ (see Fig. 3.9), as D_ℓ^p does not contain the points $x = 0$ and $x = 1$. For general degree p, the first $\frac{p-1}{2}$ and the last $\frac{p-1}{2}$ grid points of level ℓ are missing from D_ℓ^p, thus violating the Schoenberg–Whitney conditions. One possible remedy would be to move these interpolation points inside D_ℓ^p without changing the corresponding basis functions (i.e., the knots stay the same) [Höl13]. For instance in the cubic case, we could move $x = 0$ to $x = 1.5h_\ell$ and $x = 1$ to $x = 1 - 1.5h_\ell$. However, with this approach, we would not be able to interpolate boundary values. In addition, the condition of the interpolation problem will most likely worsen if we place interpolation points near the ends of the supports of the corresponding basis functions.



Mismatch of dimensions. To find a solution for this issue, let $S_\ell^{p,[0,1]}$ denote the space of all splines of degree p on the grid of level ℓ, i.e., the space S_ξ^p with

$$\xi_k := (k-p)h_\ell, \quad k = 0, \dots, m+p, \quad m := 2^\ell + p. \tag{3.22}$$

We have $D_\xi^p = [0,1]$ for this choice of ξ. Hence, the grid points $x_{\ell,i}$ $(i = 0, \dots, 2^\ell)$ satisfy the Schoenberg–Whitney conditions for the uniform B-spline basis. Clearly, the sum $\bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p$ is a subspace of $S_\ell^{p,[0,1]}$, but it cannot be equal to $S_\ell^{p,[0,1]}$ due to

$$\dim \bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p = 2^\ell + 1 < 2^\ell + p = m = \dim S_\ell^{p,[0,1]}, \quad p > 1, \tag{3.23}$$

by Prop. 3.2 (spline space). There are too few nodal (and hierarchical) basis functions to span the whole spline space $S_\ell^{p,[0,1]}$.

Restriction to spline subspaces. The key idea is now to impose additional $p-1$ boundary conditions on the basis functions to restrict $S_\ell^{p,[0,1]}$ to a reasonable subspace with the correct dimension $(2^\ell + p) - (p-1) = 2^\ell + 1$. "Reasonable" means that besides this dimension constraint, two requirements should be met: First, the Schoenberg-Whitney conditions should be satisfied for the new subspace and the grid of level ℓ. Second, the new subspace should contain all polynomials of degree $\leq p$, eliminating the issue discussed in Fig. 3.9.

3.2.2 Hierarchical Not-A-Knot B-Splines

Not-a-knot conditions. A suitable subspace can be obtained by incorporating the so-called *not-a-knot boundary conditions* into $S_\ell^{p,[0,1]}$. For the cubic case $p = 3$ (for which we need two conditions), the not-a-knot conditions demand that for all splines s in the subspace, $\frac{\mathrm{d}^3}{\mathrm{d}x^3}s$ is continuous at the first and at the last interior knot $x_{\ell,1} = h_\ell$ and $x_{\ell,2^\ell-1} = 1-h_\ell$ [Höl13]. This means that $x_{\ell,1}$ and $x_{\ell,2^\ell-1}$ are effectively removed from the knot sequence, as this is equivalent to requiring that s is a cubic polynomial on $[0, x_{\ell,2}]$ and $[x_{\ell,2^\ell-2}, 1]$ (hence "not-a-knot"). For general degree p (for which we need $(p-1)$ conditions), we require that the p-th derivative $\frac{\mathrm{d}^p}{\mathrm{d}x^p}s$ is continuous at the first $\frac{p-1}{2}$ and the last $\frac{p-1}{2}$ inner grid points

$$x_{\ell,i}, \quad i \in \{1, \dots, \tfrac{p-1}{2}\} \cup \{2^\ell - \tfrac{p-1}{2}, \dots, 2^\ell - 1\}. \tag{3.24}$$

This is equivalent to removing these knots from the knot sequence ξ, or, alternatively, to requiring that s is a polynomial of degree $\leq p$ on $[0, x_{\ell,(p+1)/2}]$ and on $[x_{\ell,2^\ell-(p+1)/2}, 1]$.

The knot sequence $\xi_\ell^{p,\mathrm{nak}}$ with not-a-knot boundary conditions is defined as follows:

$$\xi_\ell^{p,\mathrm{nak}} := (\xi_{\ell,0}^{p,\mathrm{nak}}, \dots, \xi_{\ell,m+p}^{p,\mathrm{nak}}), \quad m := 2^\ell + 1, \tag{3.25a}$$

$$\xi_{\ell,k}^{p,\mathrm{nak}} := \begin{cases} x_{\ell,k-p}, & k = 0, \dots, p, \\ x_{\ell,k-(p+1)/2}, & k = p+1, \dots, 2^\ell, \\ x_{\ell,k-1}, & k = 2^\ell + 1, \dots, 2^\ell + p + 1. \end{cases} \tag{3.25b}$$

This knot sequence $\xi_\ell^{p,\mathrm{nak}}$ can be obtained by removing the knots given in (3.24) from the knot sequence (3.22) for the full grid of level ℓ. We show $\xi_\ell^{p,\mathrm{nak}}$ and the corresponding B-spline functions in Fig. 3.10. The resulting spline space

$$S_\ell^{p,\mathrm{nak}} := S^p_{\xi_\ell^{p,\mathrm{nak}}} \tag{3.26}$$

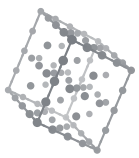
is a subspace of $S_\ell^{p,[0,1]}$ with the desired dimensionality:

$$\dim \bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p = 2^\ell + 1 = \dim S_\ell^{p,\mathrm{nak}}. \tag{3.27}$$

The space $S_\ell^{p,\mathrm{nak}}$ satisfies our two requirements: First, the spline interpolation domain $D^p_{\xi_\ell^{p,\mathrm{nak}}} = [\xi_{\ell,p}^{p,\mathrm{nak}}, \xi_{\ell,m}^{p,\mathrm{nak}}]$ of $S_\ell^{p,\mathrm{nak}}$ equals the whole unit interval $[0,1]$. This means that the Schoenberg–Whitney conditions are satisfied for $S_\ell^{p,\mathrm{nak}}$, since all interpolation points (grid points) are contained in $D^p_{\xi_\ell^{p,\mathrm{nak}}} = [0,1]$. Second, $S_\ell^{p,\mathrm{nak}}$ still contains all polynomials of degree $\leq p$, as we have only removed knots compared to $S_\ell^{p,[0,1]}$.

However, $\bigoplus_{\ell'=0}^{\ell} W_{\ell'}^p$ is not a subspace of $S_\ell^{p,\mathrm{nak}}$ anymore, since the hierarchical basis functions $\varphi_{\ell',i'}^p$ are not not-a-knot splines (due to the additional knots that we removed from $S_\ell^{p,\mathrm{nak}}$). For this reason, we have to incorporate the not-a-knot boundary conditions into the hierarchical basis.

Before defining the new hierarchical basis functions, we make two additional observations. First, $\xi_\ell^{p,\mathrm{nak}}$ coincides with the uniform knot sequence ξ_ℓ^p of Cor. 3.3 (nodal B-spline space) for the piecewise linear case of $p = 1$. This is intuitively clear: For this case, we do not need to remove any knots as the hierarchical splitting already holds for the full domain by Cor. 2.6. Second, the removal of the knots in (3.24) is only possible if $p + 1 \leq 2^\ell$, which is equivalent to $\ell \geq \lceil \log_2(p+1) \rceil$. For coarser levels, there are not enough interior knots that could be removed. Without any special treatment, we would not be able to obtain enough basis functions to span the spline space.



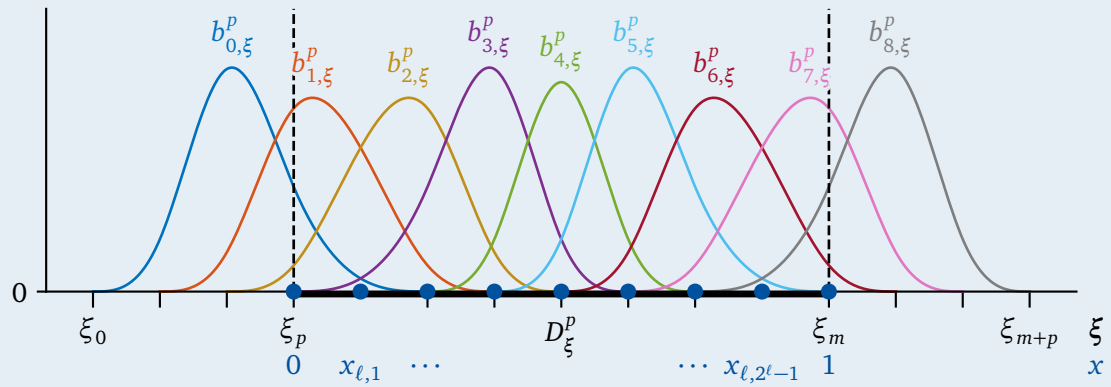


FIGURE 3.10 Not-a-knot knot sequence $\xi_\ell^{p,\mathrm{nak}}$ *(ticks on horizontal axis)* and nodal cubic not-a-knot B-splines ($p = 3$) of level $\ell = 3$. In the domain $[0, 1]$ *(delimited by dashed lines)*, the first $\frac{p-1}{2}$ and the last $\frac{p-1}{2}$ interior grid points of the set of grid points Ω_ℓ *(blue dots)* have been removed from the set of knots. The spline interpolation domain $D^p_{\xi_\ell^{p,\mathrm{nak}}}$ *(thick line)* equals the whole domain $[0, 1]$.

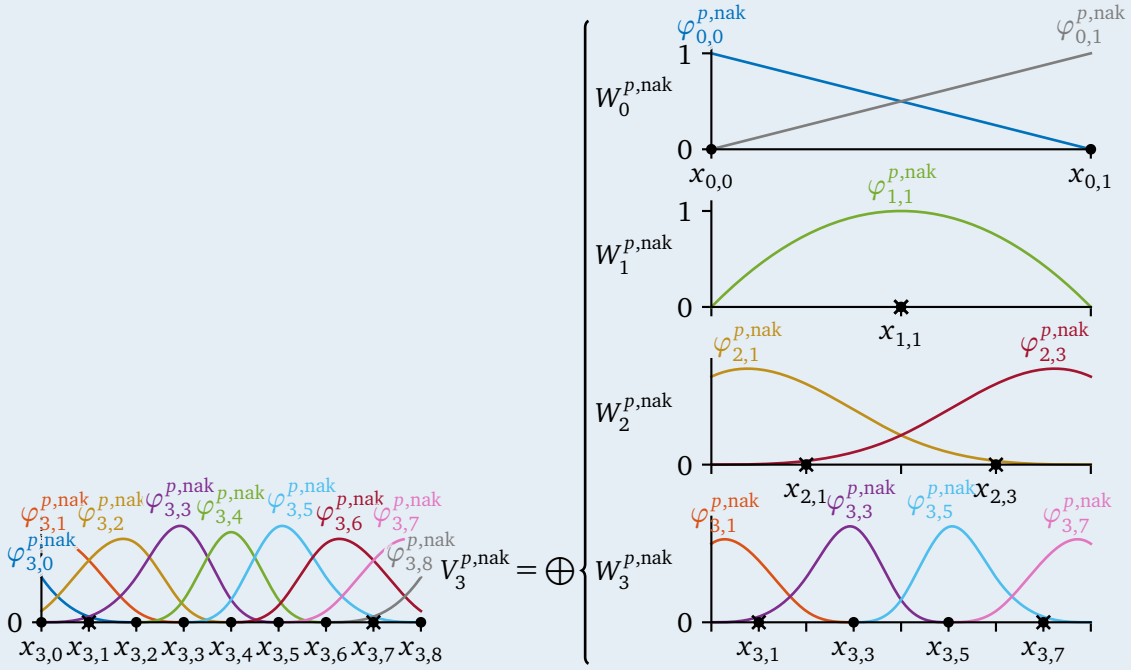
Definition of hierarchical not-a-knot B-splines. For the definition of *hierarchical not-a-knot B-splines* $\varphi_{\ell,i}^{p,\mathrm{nak}}$ based on Def. 3.1 (non-uniform B-splines), we use global Lagrange polynomials for the coarser levels:

$$\varphi_{\ell,i}^{p,\mathrm{nak}} := \begin{cases} L_{\ell,i}, & \ell < \lceil \log_2(p+1) \rceil, \\ b^p_{i,\xi_\ell^{p,\mathrm{nak}}}, & \ell \geq \lceil \log_2(p+1) \rceil, \end{cases} \qquad \ell \in \mathbb{N}_0, \quad i \in I_\ell, \tag{3.28a}$$

$$L_{\ell,i} : [0,1] \to \mathbb{R}, \quad L_{\ell,i}(x) := \prod_{\substack{i'=0,\dots,2^\ell \\ i' \neq i}} \frac{x - x_{\ell,i'}}{x_{\ell,i} - x_{\ell,i'}}. \tag{3.28b}$$

The hierarchical not-a-knot B-spline basis is shown for the cubic case $p = 3$ in Fig. 3.11. The function $L_{\ell,i}$ is the i-th *Lagrange polynomial* of level ℓ, that is, the unique polynomial of degree $\leq 2^\ell$ that interpolates the data $\{(x_{\ell,i'}, \delta_{i,i'}) \mid i' = 0, \dots, 2^\ell\}$. Since its degree $\deg L_{\ell,i}$ is bounded by 2^ℓ, we have $\deg L_{\ell,i} < p + 1$, as the Lagrange polynomials are employed only when $\ell < \lceil \log_2(p+1) \rceil$. Due to 2^ℓ even (when $\ell \geq 1$) and p odd, we can conclude from $\deg L_{\ell,i} \leq 2^\ell \leq p$ that actually $\deg L_{\ell,i} \leq 2^\ell < p$ (for $p > 1$; for $p = 1$, the case $\ell = 0$ is the exception).

The motivation for using Lagrange polynomials for coarse levels is that they form a basis of the polynomial space and that they can be implemented and calculated quickly. However, the specific choice of basis functions for the levels $\ell < \lceil \log_2(p+1) \rceil$ is arbitrary, as long as these functions are linearly independent (of each other and of the "true" not-a-knot B-splines $\varphi_{\ell,i}^{p,\mathrm{nak}}$, $\ell \geq \lceil \log_2(p+1) \rceil$) and contained in the space $S_\ell^{p,\mathrm{nak}}$.



A Nodal not-a-knot B-splines $\varphi^{p,\mathrm{nak}}_{\ell,i}$ $(i \in I_\ell)$ and grid points $x_{\ell,i}$ *(dots)*.

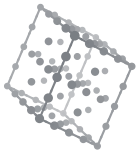
B Hierarchical not-a-knot B-splines $\varphi^{p,\mathrm{nak}}_{\ell',i'}$ $(\ell' \le \ell,\ i' \in I_{\ell'})$ and grid points $x_{\ell',i'}$ *(dots)*.

FIGURE 3.11 Univariate nodal and hierarchical cubic not-a-knot B-splines ($p = 3$) up to level $\ell = 3$. The nodal space $V^{p,\mathrm{nak}}_\ell$, which coincides with the not-a-knot spline space $S^{p,\mathrm{nak}}_\ell$, decomposes into the direct sum of the hierarchical subspaces $W^{p,\mathrm{nak}}_{\ell'}$ $(\ell' \le \ell)$. The knots of each level ℓ' are given by removing the first $\frac{p-1}{2}$ and last $\frac{p-1}{2}$ inner points *(crosses)* from the set of grid points $x_{\ell',i'}$ $(i' = 0,\dots,2^{\ell'})$.

Implementation. Note that in each level $\ell \ge \lceil \log_2(p+1) \rceil$, only the first $\frac{p+1}{2}$ (indices $i = 1,3,\dots,p$) and the last $\frac{p+1}{2}$ (indices $i = 2^\ell - p, 2^\ell - p + 2, \dots, 2^\ell - 1$) hierarchical basis functions $\varphi^{p,\mathrm{nak}}_{\ell,i}$ differ from $\varphi^{p}_{\ell,i}$, i.e., we have

$$\varphi^{p,\mathrm{nak}}_{\ell,i} = \varphi^{p}_{\ell,i}, \quad i = p+2,\ p+4,\ \dots,\ 2^\ell - p - 2. \tag{3.29}$$

This means that we can reuse uniform B-spline code for the inner functions. Due to $\varphi^{p,\mathrm{nak}}_{\ell,i}(x) = \varphi^{p,\mathrm{nak}}_{\ell,2^\ell - i}(1-x)$ (because of the symmetry of $\xi^{p,\mathrm{nak}}_\ell$), we only have to reimplement $\frac{p+1}{2}$ not-a-knot B-splines per level ℓ. As $\varphi^{p,\mathrm{nak}}_{\ell,i}$ and $\varphi^{p,\mathrm{nak}}_{\ell+1,i}$ use the same knots up to an affine transformation for ℓ large enough ($\ell \ge 3$ suffices for $p = 3$), only a number of special cases for coarse levels ℓ must be implemented. In other words, the not-a-knot approach is "minimally invasive" with respect to an implementation that already uses uniform B-splines.



Hierarchical splitting. The main benefit of the hierarchical not-a-knot B-spline basis is the validity of the hierarchical splitting. As usual, we define $V_\ell^{p,\mathrm{nak}}$ and $W_\ell^{p,\mathrm{nak}}$ as the nodal and the hierarchical not-a-knot subspace of level ℓ, respectively.

PROPOSITION 3.9 (univariate hierarchical splitting for not-a-knot B-splines) *The hierarchical splitting* (2.20) *holds for the hierarchical not-a-knot B-spline basis:*

$$S_\ell^{p,\mathrm{nak}} = V_\ell^{p,\mathrm{nak}} = \bigoplus_{\ell'=0}^{\ell} W_{\ell'}^{p,\mathrm{nak}}, \tag{3.30}$$

where for $\ell < \lceil \log_2(p+1) \rceil$, $S_\ell^{p,\mathrm{nak}}$ is defined as the space P^{2^ℓ} of polynomials of degree $\leq 2^\ell$ on $[0,1]$. (For $\ell \geq \lceil \log_2(p+1) \rceil$, $S_\ell^{p,\mathrm{nak}}$ is the not-a-knot spline space.)

PROOF For $\ell < \lceil \log_2(p+1) \rceil$, all three spaces coincide with P^{2^ℓ} and nothing is to prove.

For $\ell \geq \lceil \log_2(p+1) \rceil$, we check the two conditions of Lemma 2.2 (univariate hierarchical splitting characterization). First, the hierarchical subspace $W_{\ell'}^{p,\mathrm{nak}}$ ($\ell' \leq \ell$) is a subspace of $S_\ell^{p,\mathrm{nak}} = V_\ell^{p,\mathrm{nak}}$. This is a conclusion of Prop. 3.2 (spline space), as every function $\varphi_{\ell',i'}^{p,\mathrm{nak}}$ ($i' \in I_{\ell'}$) is continuous on $[0,1]$, a polynomial on every knot interval of $\xi_\ell^{p,\mathrm{nak}}$, and at the knots themselves at least $(p-1)$ times continuously differentiable.

Second, the hierarchical functions $\varphi_{\ell',i'}^{p,\mathrm{nak}}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$) are linearly independent. This can be shown similarly to the proof of Prop. 3.5 (hierarchical B-splines are linearly independent). The linear independence of the Lagrange polynomials can be checked by inserting grid points into a zero linear combination. The linear combination collapses and only one term remains, the coefficient corresponding to the grid point. Hence, all coefficients must vanish. ∎

COROLLARY 3.10 (multivariate hierarchical splitting for not-a-knot B-splines) *It holds*

$$S_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}} = V_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}} = \bigoplus_{\boldsymbol{\ell}'=\mathbf{0}}^{\boldsymbol{\ell}} W_{\boldsymbol{\ell}'}^{\boldsymbol{p},\mathrm{nak}}, \tag{3.31}$$

where $S_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}}$ is the tensor product space of $S_{\ell_t}^{p_t,\mathrm{nak}}$ ($t = 1,\dots,d$) as defined in Prop. 3.9.

PROOF Follows directly from Prop. 2.5 (from univariate to multivariate splitting). ∎

Sparse grids with not-a-knot B-splines. Regular sparse grid spaces using the new hierarchical not-a-knot basis are defined analogously to the uniform case, i.e.,

$$V_{n,d}^{\mathrm{s},\boldsymbol{p},\mathrm{nak}} := \bigoplus_{\|\boldsymbol{\ell}\|_1 \le n} W_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}}. \tag{3.32}$$

If the level n is large enough, then $V_{n,d}^{\mathrm{s},\boldsymbol{p},\mathrm{nak}}$ contains the space $P^{\boldsymbol{p}}$ of all d-variate polynomials of coordinate degree $\le \boldsymbol{p}$ on $[\mathbf{0},\mathbf{1}]$ (i.e., functions $f:[0,1]\to\mathbb{R}$ of the form $f(\boldsymbol{x}) := \sum_{\boldsymbol{q}=\mathbf{0}}^{\boldsymbol{p}} c_{\boldsymbol{q}} \prod_{t=1}^{d} x_t^{q_t}$ with $c_{\boldsymbol{q}} \in \mathbb{R}$). This means that in contrast to the uniform B-spline basis, hierarchical not-a-knot B-splines on sparse grids are able to replicate global polynomials on $[\mathbf{0},\mathbf{1}]$:

COROLLARY 3.11 *If $n \ge \|\lceil \log_2(\boldsymbol{p}+\mathbf{1})\rceil\|_1$, then $P^{\boldsymbol{p}} \subseteq V_{n,d}^{\mathrm{s},\boldsymbol{p},\mathrm{nak}}$.*

PROOF Let $\boldsymbol{\ell} := \lceil \log_2(\boldsymbol{p}+\mathbf{1})\rceil$ and $n \ge \|\boldsymbol{\ell}\|_1$. By Cor. 3.10, we have $\bigoplus_{\boldsymbol{\ell}'=\mathbf{0}}^{\boldsymbol{\ell}} W_{\boldsymbol{\ell}'}^{\boldsymbol{p},\mathrm{nak}} = S_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}}$. In addition, all $\boldsymbol{\ell}' \in \mathbb{N}_0^d$ with $\boldsymbol{\ell}' \le \boldsymbol{\ell}$ satisfy $\|\boldsymbol{\ell}'\|_1 \le n$ and thus, $\bigoplus_{\boldsymbol{\ell}'=\mathbf{0}}^{\boldsymbol{\ell}} W_{\boldsymbol{\ell}'}^{\boldsymbol{p},\mathrm{nak}} \subseteq V_{n,d}^{\mathrm{s},\boldsymbol{p},\mathrm{nak}}$ by (3.32). We conclude $P^{\boldsymbol{p}} \subseteq S_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{nak}} \subseteq V_{n,d}^{\mathrm{s},\boldsymbol{p},\mathrm{nak}}$, which is the asserted claim. ■

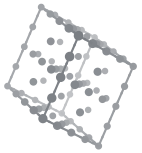
3.2.3 Modified and Non-Uniform Hierarchical Not-A-Knot B-Splines

Modified hierarchical not-a-knot B-splines. As for uniform and Clenshaw–Curtis B-splines (Sec. 3.1), it is possible to define a modified version of the hierarchical not-a-knot B-spline basis to obtain "reasonable" boundary values without boundary points. However, we cannot use Lemma 3.7 (Marsden's identity) similarly to (3.16): Due to the removal of knots, there is only a single not-a-knot B-spline $\varphi_{\ell,0}^{p,\mathrm{nak}}$ left of $\varphi_{\ell,1}^{p,\mathrm{nak}}$. B-splines $\varphi_{\ell,i}^{p,\mathrm{nak}}$ with index $i<0$ would vanish on $[0,1]$.

While we are therefore not able to construct modified functions whose second derivative vanishes in a neighborhood of $x=0$, we can use $\varphi_{\ell,0}^{p,\mathrm{nak}}$ to let the second derivative vanish in $x=0$ itself:

$$\varphi_{\ell,i}^{p,\mathrm{nak,mod}}(x) := \begin{cases} 1, & \ell=1, \quad i=1, \\ \varphi_{\ell,1}^{p,\mathrm{nak}}(x) - \dfrac{\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi_{\ell,1}^{p,\mathrm{nak}}(0)}{\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi_{\ell,0}^{p,\mathrm{nak}}(0)}\varphi_{\ell,0}^{p,\mathrm{nak}}(x), & \ell\ge 2, \quad i=1, \\ \varphi_{\ell,i}^{p,\mathrm{nak}}(x), & \ell\ge 2, \quad i\in I_\ell \setminus \{1, 2^\ell-1\}, \\ \varphi_{\ell,1}^{p,\mathrm{nak,mod}}(1-x), & \ell\ge 2, \quad i=2^\ell-1. \end{cases} \tag{3.33}$$

The resulting modified hierarchical not-a-knot B-spline basis $\varphi_{\ell,i}^{p,\mathrm{nak,mod}}$ is shown with dashed lines in Fig. 3.12A. As before, we have to implement $\varphi_{\ell,1}^{p,\mathrm{nak,mod}}$ only for a single



level ℓ, as modified functions of higher levels are the same up to an affine parameter transformation. Note again that for $p \geq 5$, we would have to modify additional interior B-splines as the interior of their support then extends to the boundary of $[0,1]$. We refrain from doing so to keep the definition (3.33) simple.

Non-uniform hierarchical not-a-knot B-splines. The not-a-knot construction is completely independent from the distribution of the grid points at hand. Consequently, we can define hierarchical not-a-knot B-splines for non-uniform distributions. For instance, to define not-a-knot B-splines for Chebyshev points (see Sec. 3.1.4), we first specify the knot sequence as

$$\xi_\ell^{p,\mathrm{cc,nak}} := (\xi_{\ell,0}^{p,\mathrm{cc,nak}}, \dots, \xi_{\ell,m+p}^{p,\mathrm{cc,nak}}), \quad m := 2^\ell + 1, \tag{3.34a}$$

$$\xi_{\ell,k}^{p,\mathrm{cc,nak}} := \begin{cases} x_{\ell,k-p}^{\mathrm{cc}}, & k = 0, \dots, p, \\ x_{\ell,k-(p+1)/2}^{\mathrm{cc}}, & k = p+1, \dots, 2^\ell, \\ x_{\ell,k-1}^{\mathrm{cc}}, & k = 2^\ell + 1, \dots, 2^\ell + p + 1, \end{cases} \tag{3.34b}$$

and then define hierarchical not-a-knot Clenshaw–Curtis B-splines as

$$\varphi_{\ell,i}^{p,\mathrm{cc,nak}} := \begin{cases} L_{\ell,i}^{\mathrm{cc}}, & \ell < \lceil \log_2(p+1) \rceil, \\ b_{i,\xi_\ell^{p,\mathrm{cc,nak}}}^p, & \ell \geq \lceil \log_2(p+1) \rceil, \end{cases} \quad \ell \in \mathbb{N}_0, \quad i \in I_\ell, \tag{3.35a}$$

$$L_{\ell,i}^{\mathrm{cc}} : [0,1] \to \mathbb{R}, \quad L_{\ell,i}^{\mathrm{cc}}(x) := \prod_{\substack{i'=0,\dots,2^\ell \\ i' \neq i}} \frac{x - x_{\ell,i'}^{\mathrm{cc}}}{x_{\ell,i}^{\mathrm{cc}} - x_{\ell,i'}^{\mathrm{cc}}}. \tag{3.35b}$$

This definition can even be combined with the modification of hierarchical not-a-knot B-splines as discussed above. We can use exactly the same approach as in (3.33), if we replace the not-a-knot basis functions with their non-uniform not-a-knot counterparts (not-a-knot Clenshaw–Curtis B-splines in the above example). The hierarchical not-a-knot Clenshaw–Curtis B-spline basis of cubic degree and the corresponding modified functions are shown in Fig. 3.12B.

3.2.4 Other Approaches to Incorporate Boundary Conditions

Not-a-knot boundary conditions are not the only approach to obtain a subspace of $S_\ell^{p,[0,1]}$ with the right dimension $2^\ell - 1$. Another possibility, which we want to discuss briefly, are *natural boundary conditions*. In the cubic case, for which they are usually formulated [Höl13], these boundary conditions require that the second derivatives $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi_{\ell,i}$ of the

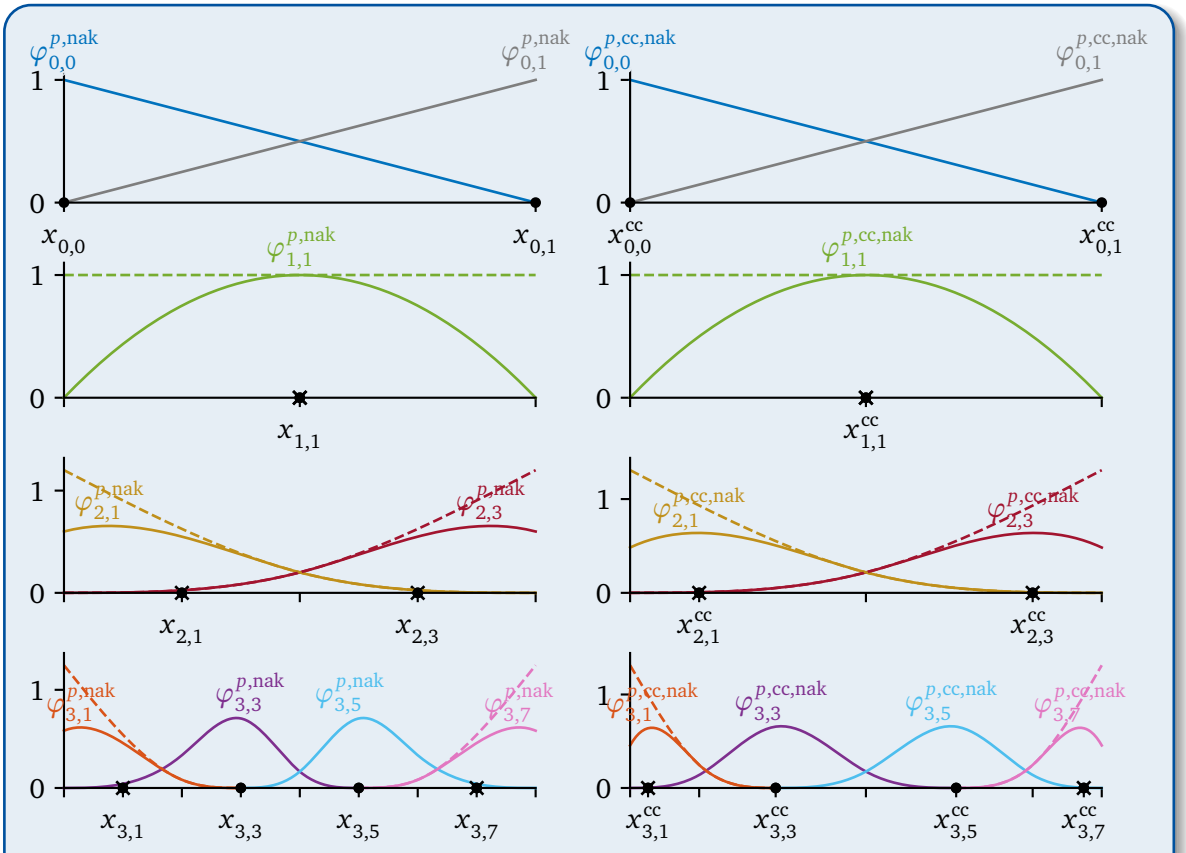


FIGURE 3.12 Comparison of uniform *(left)* and Clenshaw–Curtis *(right)* hierarchical cubic not-a-knot B-splines $\varphi^{p,\mathrm{nak}}_{\ell',i'}$ and $\varphi^{p,\mathrm{cc,nak}}_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$, $p = 3$) up to level $\ell = 3$ together with the respective modified versions $\varphi^{p,\mathrm{nak,mod}}_{\ell',i'}$ and $\varphi^{p,\mathrm{cc,nak,mod}}_{\ell',i'}$ *(dashed)*. The knots of each level ℓ' are given by removing the first $\frac{p-1}{2}$ and last $\frac{p-1}{2}$ inner points *(crosses)* from the set of grid points $x_{\ell',i'}$ or $x^{\mathrm{cc}}_{\ell',i'}$ ($i' = 0, \dots, 2^{\ell'}$), respectively.

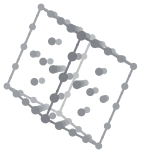
basis functions vanish at the boundary $x \in \{0,1\}$. To obtain the necessary number of $p-1$ constraints also for higher degrees p, we require that all derivatives $\frac{\mathrm{d}^q}{\mathrm{d}x^q}\varphi_{\ell,i}$ of order $q = 2, \dots, \frac{p+1}{2}$ vanish at $x \in \{0,1\}$.

Consequently, we can define hierarchical natural B-splines as

$$\varphi^{p,\mathrm{nat}}_{\ell,i}(x) := \begin{cases} L_{0,i}(x), & \ell = 0, \\ \varphi^{p}_{\ell,i} + \sum_{j \in J^{p,\mathrm{nat}}_i} c_{\ell,i,j}\varphi^{p}_{\ell,j}, & \ell \ge 1, \end{cases} \qquad \ell \in \mathbb{N}_0, \quad i \in I_\ell, \tag{3.36a}$$

$$J^{p,\mathrm{nat}}_i := \{i - \tfrac{p-1}{2}, \dots, i-1\} \cup \{i+1, \dots, i + \tfrac{p-1}{2}\}, \tag{3.36b}$$

where the coefficients $c_{\ell,i,j} \in \mathbb{R}$ are chosen such that the natural boundary conditions are



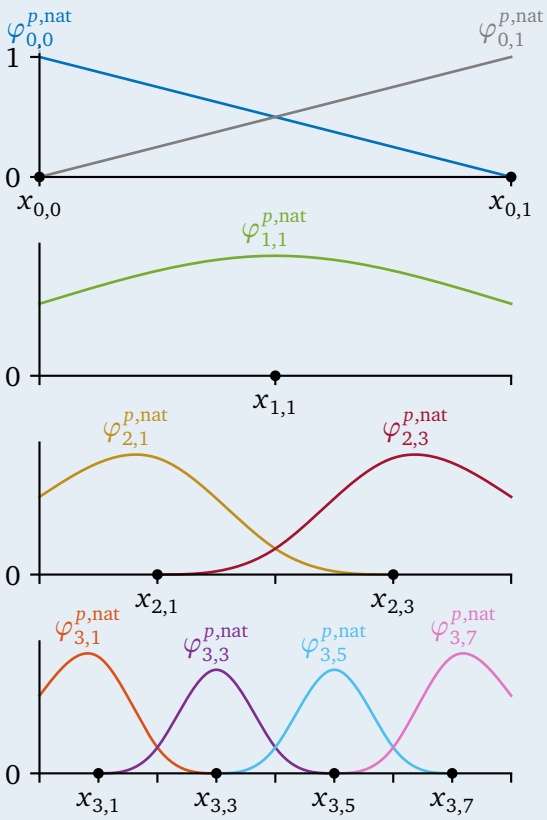


FIGURE 3.13
Hierarchical cubic natural B-splines $\varphi^{p,\mathrm{nat}}_{\ell',i'}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$, $p = 3$) and grid points $x_{\ell',i'}$ *(dots)* up to level $\ell = 3$.

satisfied:

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q}\varphi^{p,\mathrm{nat}}_{\ell,i}(x) = 0, \quad \ell \geq 1, \quad i \in I_\ell, \quad q = 2, \ldots, \tfrac{p+1}{2}, \quad x \in \{0,1\}. \tag{3.37}$$

The first half of the coefficients $c_{\ell,i,j} \in \mathbb{R}$ ($j < i$) vanishes if the interior of the support of $\varphi^p_{\ell,i}$ does not contain $x = 0$ (i.e., $i \geq \frac{p+1}{2}$). The second half of the coefficients ($j > i$) vanishes analogously if $1 \notin \operatorname{supp} \varphi^p_{\ell,i} \iff i \leq 2^\ell - \frac{p+1}{2}$. This means that only the first $\lfloor \frac{p+1}{4} \rfloor$ and the last $\lfloor \frac{p+1}{4} \rfloor$ hierarchical functions have to be altered in each level.

Figure 3.13 shows the hierarchical natural spline basis. The main disadvantage of natural boundary conditions is that we are not able to replicate arbitrary polynomials exactly on $[0,1]$ with this approach. Only polynomials that satisfy natural boundary conditions themselves (linear polynomials for example) can be replicated exactly. For this reason, we do not further consider this basis in the rest of the thesis.

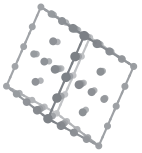
Algorithms for B-Splines on Sparse Grids

> *Who are you? How did you get in my house?*
>
> — Donald E. Knuth about one-based array indices in algorithms (according to xkcd[1])

Little is known about the algorithmic challenges that hierarchical bases of B-spline type (or even of general type) pose on sparse grids. In general, we are not able to directly apply the sparse grid algorithms that were designed for hat functions $\varphi^1_{\ell,i}$. Hence, we have to generalize these algorithms to higher-order B-splines $\varphi^p_{\ell,i}$ or even to arbitrary tensor product basis functions. The main problem is the larger support of higher-order B-splines when compared to degree $p = 1$. The larger support introduces new dependencies between values of grid points that cannot be resolved with conventional algorithms.

This chapter gives an overview of six algorithms for B-splines on sparse grids. Two of these algorithms are already known [Gri92; Bal94], while the remaining four are new. Correctness results are given for every algorithm. We use hierarchization as the exemplary problem for our algorithms, but the ideas of the algorithms can be generalized to any linear operator. Furthermore, most algorithms are not tailored to B-splines $\varphi^p_{\ell,i}$,

[1] `https://xkcd.com/163/`



but applicable to general tensor product basis functions $\varphi_{\ell,i}$. Whether an algorithmic approach is feasible for the sparse grid at hand or not depends on the grid's type: full grid, dimensionally adaptive sparse grid, or spatially adaptive sparse grid. The more assumptions the grid satisfies, the faster and easier the corresponding algorithms will be.

Section 4.1 explains hierarchization as our example problem and defines the notation used in this chapter. The remaining four sections treat the three different types of grids: First, Sec. 4.2 deals with full grids to formalize and repeat the well-known unidirectional principle (UP). Second, Sec. 4.3 focuses on algorithms for dimensionally adaptive sparse grids. Third, Sec. 4.4 and Sec. 4.5 treat arbitrary (spatially adaptive) sparse grids, which is the most interesting case for us. Section 4.4 employs breadth-first search (BFS) for hierarchization, while Sec. 4.5 uses the UP.

This original chapter is the main theoretical contribution of this thesis. Although the unidirectional principle in Sec. 4.2 and the combination technique in Sec. 4.3 are well-known, the presentation with formal proofs of correctness is new. Parts of the chapter have already been published in scientific papers, namely Sec. 4.4 [Vale18a]. The weakly fundamental splines (Sec. 4.5.4) and the Hermite hierarchization method (Sec. 4.5.5) are based on an idea by Dr. Stefan Zimmer (University of Stuttgart, Germany).

4.1 The Hierarchization Problem

Let $\Omega^s \subseteq [0,1]^d$ be a general (sparse) grid that may be spatially adaptive, i.e., of the form $\Omega^s = \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid (\boldsymbol{\ell},\boldsymbol{i}) \in K\}$, where K is a set of level-index pairs $(\boldsymbol{\ell},\boldsymbol{i})$ with $\boldsymbol{\ell} \in \mathbb{N}_0^d$ and $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$ such that $N := |\Omega^s| = |K| < \infty$ (see Sec. 2.3.3). The *hierarchization problem* is finding *hierarchical surpluses* $(\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'})_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \in \mathbb{R}^N$ such that

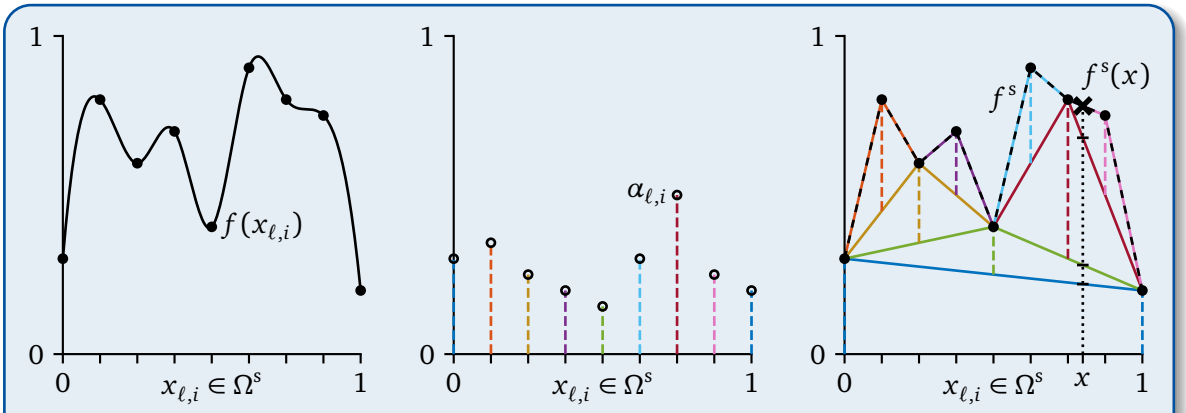
$$\sum_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \quad \text{for all} \quad (\boldsymbol{\ell},\boldsymbol{i}) \in K, \tag{4.1}$$

where $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ are arbitrary tensor product basis functions and $(f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}))_{(\boldsymbol{\ell},\boldsymbol{i})\in K} \in \mathbb{R}^N$ is a set of function values $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ at the grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$. This then defines the interpolant f^s as

$$f^s\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}, \quad f^s := \sum_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}, \tag{4.2}$$

which interpolates f at the grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ of Ω^s. Figure 4.1 shows the process of hierarchizing given function values and evaluating the resulting interpolant.

We explicitly allow $\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ to be nodal basis functions, in which case $\boldsymbol{\ell}'$ is constant and Ω^s is a full grid. Strictly speaking, the problem is then an *interpolation problem* and the



A The objective function f is sampled at the grid points $x_{\ell,i} \in \Omega^s$ to obtain function values $f(x_{\ell,i})$, which form the input vector $\boldsymbol{u}$ for the linear operator $\mathfrak{L} = \boldsymbol{A}^{-1}$.

B The linear operator $\mathfrak{L}$ is applied to $\boldsymbol{u}$ to obtain the output vector $\boldsymbol{y}$, which contains the hierarchical surpluses $\alpha_{\ell,i}$ ($(\ell,i) \in K$).

C The interpolant f^s *(black dashed line)* is evaluated at $x \in [0,1]$ by adding contributions *(black dotted lines)* of weighted basis functions $\alpha_{\ell,i}\varphi_{\ell,i}$ *(colored)*, obtaining $f^s(x)$ *(cross)*.

FIGURE 4.1 Hierarchization of function values $f(x_{\ell,i})$ *(left)* to obtain hierarchical surpluses $\alpha_{\ell,i}$ *(center)* and evaluation of the resulting interpolant f^s *(right)*, using a univariate grid and the piecewise linear basis as an example.

$\alpha_{\ell',i'}$ are *interpolation coefficients*. However, we still apply the terms "hierarchization" and "hierarchical surpluses" in this case to keep the terminology consistent.

Hierarchization as a linear operator. The example of hierarchization can be generalized to arbitrary linear operators

$$\mathfrak{L}\colon \mathbb{R}^N \to \mathbb{R}^N, \quad \boldsymbol{u} \mapsto \boldsymbol{y} = \mathfrak{L}[\boldsymbol{u}], \tag{4.3}$$

where $\mathfrak{L}$ depends on the grid Ω^s at hand. Input $\boldsymbol{u}$ and output $\boldsymbol{y}$ are scalar-valued data

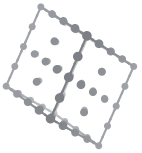
$$\boldsymbol{u} = (u_{\ell,i})_{(\ell,i)\in K} \in \mathbb{R}^N, \quad \boldsymbol{y} = (y_{\ell,i})_{(\ell,i)\in K} \in \mathbb{R}^N, \tag{4.4}$$

which give one scalar per grid point $\boldsymbol{x}_{\ell,i} \in \Omega^s$. For the case of hierarchization, $\mathfrak{L}$ is the inverse of the *interpolation matrix* $\boldsymbol{A} \in \mathbb{R}^{N\times N}$:

$$\mathfrak{L} = \boldsymbol{A}^{-1}, \quad \boldsymbol{A} = (\varphi_{\ell',i'}(\boldsymbol{x}_{\ell,i}))_{(\ell,i),(\ell',i')\in K}, \quad \boldsymbol{u} = (f(\boldsymbol{x}_{\ell,i}))_{(\ell,i)\in K}, \quad \boldsymbol{y} = (\alpha_{\ell',i'})_{(\ell',i')\in K}. \tag{4.5a}$$

This means that we can determine the $\alpha_{\ell',i'}$ by solving the $N \times N$ system of linear equations

$$\boldsymbol{y} = \mathfrak{L}[\boldsymbol{u}] \quad \Longleftrightarrow \quad \boldsymbol{A} \cdot (\alpha_{\ell',i'})_{(\ell',i')\in K} = (f(\boldsymbol{x}_{\ell,i}))_{(\ell,i)\in K}. \tag{4.5b}$$



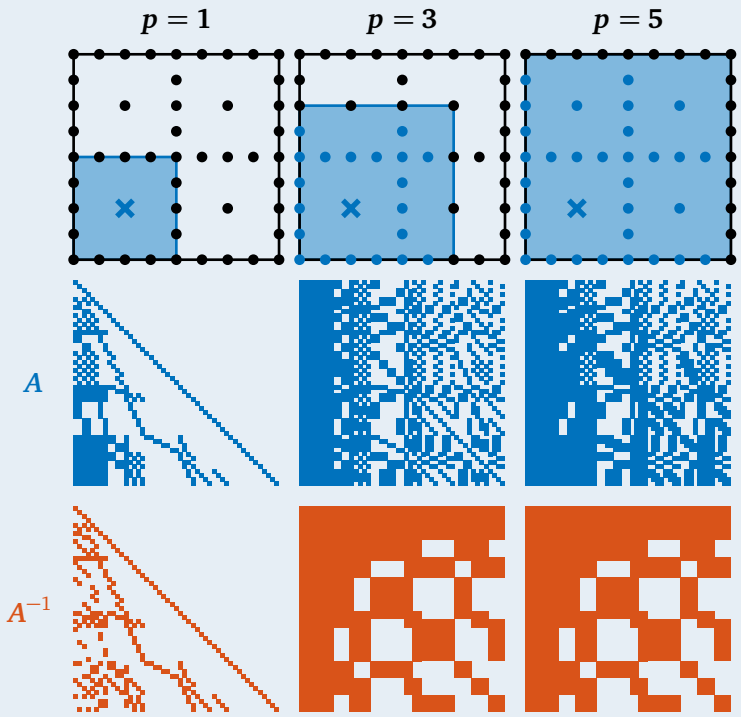


FIGURE 4.2
Density pattern of the hierarchization matrix $\boldsymbol{A}$ *(middle row, blue)* and of its inverse $\boldsymbol{A}^{-1}$ *(bottom row, red)* for the regular sparse grid $\Omega^{s(1)}_{n,d}$ with $n = 4$ and $d = 2$ *(top row)* and uniform hierarchical B-splines $\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell},\boldsymbol{i}}$ for degrees $\boldsymbol{p} \in \{\boldsymbol{1}, \boldsymbol{3}, \boldsymbol{5}\}$. The blue areas in the top row show the extent of the support of one specific basis function $\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ with $\boldsymbol{\ell}' = (2,2)$ and $\boldsymbol{i}' = (1,1)$ (*cross:* corresponding grid point $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}$). The blue points are the grid points at which $\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ is non-zero.

Complexity of B-spline hierarchization. As noted in [Vale18a], hierarchization on sparse grids with hierarchical B-splines $\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell},\boldsymbol{i}}$ of degree $\boldsymbol{p}$ as basis functions $\varphi_{\boldsymbol{\ell},\boldsymbol{i}}$ is a tedious task. The corresponding linear system (4.5) is in general non-symmetric (i.e., $\varphi^{\boldsymbol{p}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \neq \varphi^{\boldsymbol{p}}_{\boldsymbol{\ell},\boldsymbol{i}}(\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'})$) and densely populated. This is because the matrix entry in the $(\boldsymbol{\ell},\boldsymbol{i})$-th row and $(\boldsymbol{\ell}',\boldsymbol{i}')$-th column vanishes if and only if

$$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \notin \mathring{\operatorname{supp}}\, \varphi^{\boldsymbol{p}}_{\boldsymbol{\ell}',\boldsymbol{i}'} \iff \exists_{t=1,\ldots,d}\ x_{\ell_t,i_t} \notin \left]x_{\ell'_t,i'_t} - \tfrac{p_t+1}{2} h_{\ell'_t},\, x_{\ell'_t,i'_t} + \tfrac{p_t+1}{2} h_{\ell'_t}\right[, \tag{4.6}$$

where $\mathring{\operatorname{supp}}$ is the interior of the support [Vale18a]. For coarse levels $\boldsymbol{\ell}'$, the mesh size $h_{\ell'_t}$ is large in every dimension t, which implies that $\mathring{\operatorname{supp}}\, \varphi^{\boldsymbol{p}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ contains most of the grid points. In contrast to the hat function case ($\boldsymbol{p} = \boldsymbol{1}$), the value of $\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}$ depends not only on $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ and the data of the $3^d - 1$ neighboring grid points on the boundary of $\operatorname{supp} \varphi^{\boldsymbol{1}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$, but potentially on the data of the whole grid. This is shown in Fig. 4.2: There are at most $3^d = 9$ non-zero entries in each row of $\boldsymbol{A}^{-1}$ for $\boldsymbol{p} = \boldsymbol{1}$ and $d = 2$. As soon as the B-spline degree is increased, both $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$ become significantly denser.

This prohibits the use of the UP, which we will discuss in the next section, on sparse grids with hierarchical B-splines. Consequently, we have to solve the linear system (4.5), which is significantly more time-consuming, as it takes between $\Omega(N^2 d)$ and $\mathcal{O}(N^2(N+d))$ time via Gaussian elimination.[2] In addition, if we use an explicit solver for the linear

[2] $\Omega(N^2 d)$ for calculating $\boldsymbol{A}$ and $\mathcal{O}(N^3)$ for solving the system.

system, we additionally have to store an $N \times N$ matrix in memory. However, a grid of size $N = 116\,000$ already exceeds the memory of a 128 GiB supercomputer node, if we explicitly store the full matrix in double precision. In comparison, for the hat function basis, the UP only requires $\mathcal{O}(Nd)$ time and $\mathcal{O}(N)$ memory.

Notation. We do not need the hierarchical level-index information $(\boldsymbol{\ell}, \boldsymbol{i})$ in Ω^s, K, $\boldsymbol{u}$, and $\boldsymbol{y}$ for most of the considerations in this chapter. In these cases, we assume that in each dimension t, the level-index pairs (ℓ_t, i_t) ($\ell_t \in \mathbb{N}_0$, $i_t \in I_{\ell_t}$) are continuously enumerated by a single index $k_t = k_t(\ell_t, i_t) \in \mathbb{N}_0$. We identify $(\boldsymbol{\ell}, \boldsymbol{i})$ with a single index $\boldsymbol{k}$, whose t-th component is given by $k_t(\ell_t, i_t)$. Hence, we regard K as a subset $K := \{\boldsymbol{k} \mid \boldsymbol{x}_{\boldsymbol{k}} \in \Omega^s\}$ of $\mathbb{N}_0^d$. We will switch between the notations whenever appropriate. All statements that are formulated in the $\boldsymbol{k}$ notation are valid for both the nodal and the hierarchical basis.

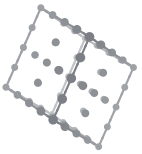
In the following, k_t denotes the t-th component of a d-vector $\boldsymbol{k}$ as usual. With $\boldsymbol{k}_{-t}$, we denote the $(d-1)$-vector that is obtained from $\boldsymbol{k}$ by omitting the t-th component, i.e., $\boldsymbol{k}_{-t} := (k_1, \ldots, k_{t-1}, k_{t+1}, \ldots, k_d)$. For a j-tuple $T = (t_1, \ldots, t_j) \in \{1, \ldots, d\}^j$, we define $\boldsymbol{k}_T$ to be the j-vector $(k_{t_1}, \ldots, k_{t_j})$ that only contains the entries of the dimensions listed in T. Accordingly, $\boldsymbol{k}_{-T}$ is defined as the $(d-j)$-vector that contains the entries of the remaining dimensions (sorted by the dimension t). We define $\boldsymbol{k}_{a:b} := (k_a, k_{a+1}, \ldots, k_b)$ as an indexing shortcut ($a \le b$).

4.2 Hierarchization on Full Grids (Unidirectional Principle)

If Ω^s is a full grid $\Omega_{\boldsymbol{\ell}}$ (see Sec. 2.1), the well-known UP can be used to apply $\mathfrak{L}$ to input data $\boldsymbol{u}$. As shown in Fig. 4.3, the idea of the UP is to apply the corresponding one-dimensional operators on the one-dimensional subgrids (the *poles*) of Ω^s, which is repeated for all dimensions. In this section, we first formulate the UP for general linear operators $\mathfrak{L}$ and then prove its correctness for the case $\mathfrak{L} = \boldsymbol{A}^{-1}$ of hierarchization. The correctness for arbitrary tensor product operators will follow from the considerations in Sec. 4.5.

Unidirectional principle and its correctness. We state the UP in Alg. 4.1. The algorithm is given a permutation $(t_1, \ldots, t_d)$ of $(1, \ldots, d)$ that specifies the order of dimensions in which the UP should be applied. We denote with $\mathfrak{L}^{(t_j), K_{\text{pole}}}$ the one-dimensional version of $\mathfrak{L}$ applied in dimension t_j ($j = 1, \ldots, d$) on the pole K_{pole}. Formally, a pole is an equivalence class of the *pole equivalence relation* $\sim_{t_j}$ on K:

$$\boldsymbol{k}' \sim_{t_j} \boldsymbol{k}'' \iff \boldsymbol{k}'_{-t_j} = \boldsymbol{k}''_{-t_j}, \quad \boldsymbol{k}', \boldsymbol{k}'' \in K. \tag{4.7}$$



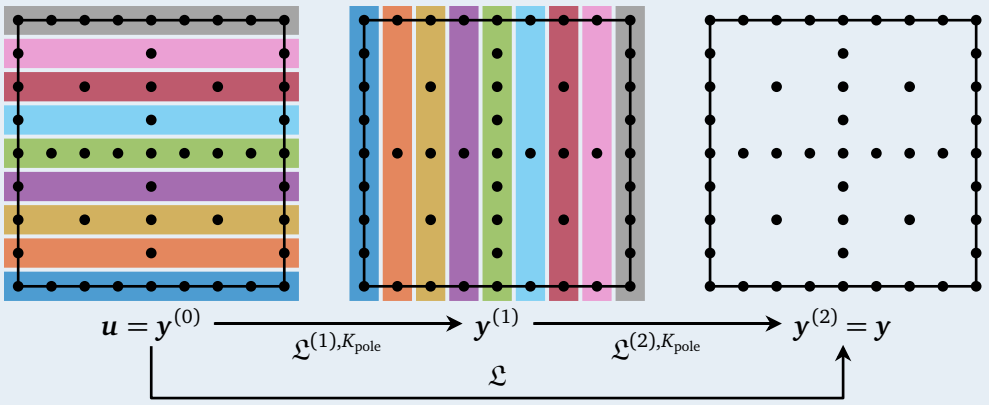


FIGURE 4.3 Application of a linear operator $\mathfrak{L}$ on two-dimensional sparse grid data with the unidirectional principle. *Left:* The univariate operator $\mathfrak{L}^{(1),K_{\text{pole}}}$ is applied on the input data $\boldsymbol{u}$ along poles K_{pole} of the first dimension x_1. *Center:* The univariate operator $\mathfrak{L}^{(2),K_{\text{pole}}}$ is applied on the resulting intermediate data $\boldsymbol{y}^{(1)}$ along poles K_{pole} of the second dimension x_2. *Right:* Final values $\boldsymbol{y} = \mathfrak{L}[\boldsymbol{u}]$ after the application on both dimensions. All grid points of the same color are part of the same pole K_{pole} (equivalence classes of $\sim_t$ in Alg. 4.1).

We prove the correctness of the UP with the following invariant:

PROPOSITION 4.1 (invariant of unidirectional principle for hierarchization)
Let $\mathfrak{L}$ be the hierarchization operator on a full grid, i.e., $\mathfrak{L} = \boldsymbol{A}^{-1}$, $\boldsymbol{u} = (f(\boldsymbol{x}_{\boldsymbol{k}}))_{\boldsymbol{k}\in K}$, $\boldsymbol{y} = (\alpha_{\boldsymbol{k}})_{\boldsymbol{k}\in K}$, $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ is the univariate interpolation operator $(\boldsymbol{A}^{(t_j)})^{-1}$, and $K = \{\boldsymbol{0},\dots,\boldsymbol{2}^{\boldsymbol{\ell}}\}$ corresponds to a full grid $\Omega_{\boldsymbol{\ell}}$ of level $\boldsymbol{\ell}$. After iteration j of Alg. 4.1 ($j = 1,\dots,d$), it holds for $T := (t_1,\dots,t_j)$

$$\sum_{\boldsymbol{k}_T=\boldsymbol{0}}^{\boldsymbol{2}^{\boldsymbol{\ell}_T}} y^{(j)}_{(\boldsymbol{k}_T,\boldsymbol{k}'_{-T})} \varphi_{\boldsymbol{k}_T}(\boldsymbol{x}_{\boldsymbol{k}'_T}) = f(\boldsymbol{x}_{\boldsymbol{k}'}), \quad \boldsymbol{k}' = \boldsymbol{0},\dots,\boldsymbol{2}^{\boldsymbol{\ell}}, \tag{4.8}$$

where $(\boldsymbol{k}_T,\boldsymbol{k}'_{-T})$ is shorthand for $\boldsymbol{k}''$ with $k''_t := k_t$ if $t \in T$ and $k''_t := k'_t$ if $t \notin T$.

PROOF We prove the assertion by induction over $j = 1,\dots,d$. We set $T' := (t_1,\dots,t_{j-1})$, $T := (t_1,\dots,t_{j-1},t_j)$, and we exploit the tensor product structure of the basis to write the left-hand side (LHS) of the assertion for j and arbitrary $\boldsymbol{k}' = \boldsymbol{0},\dots,\boldsymbol{2}^{\boldsymbol{\ell}}$ as

$$\sum_{\boldsymbol{k}_T=\boldsymbol{0}}^{\boldsymbol{2}^{\boldsymbol{\ell}_T}} y^{(j)}_{(\boldsymbol{k}_T,\boldsymbol{k}'_{-T})} \varphi_{\boldsymbol{k}_T}(\boldsymbol{x}_{\boldsymbol{k}'_T}) = \sum_{\boldsymbol{k}_{T'}=\boldsymbol{0}}^{\boldsymbol{2}^{\boldsymbol{\ell}_{T'}}} \varphi_{\boldsymbol{k}_{T'}}(\boldsymbol{x}_{\boldsymbol{k}'_{T'}}) \sum_{k_{t_j}=0}^{2^{\ell_{t_j}}} y^{(j)}_{(\boldsymbol{k}_T,\boldsymbol{k}'_{-T})} \varphi_{k_{t_j}}(x_{k'_{t_j}}). \tag{4.9}$$

If we choose the equivalence class $K_{\text{pole}} := [(\boldsymbol{k}_T,\boldsymbol{k}'_{-T})]_{\sim_{t_j}}$ ($\boldsymbol{k}_T$ arbitrary), then the inner

```
1 function y = unidirectionalPrinciple(u, K, (t_1,...,t_d))
2   y^(0) ← u
3   for j = 1,...,d do
4     for K_pole ∈ K/∼_{t_j} do
5       (y_k^(j))_{k∈K_pole} ← 𝔏^{(t_j),K_pole}[(y_k^(j−1))_{k∈K_pole}]   ⇝ apply univariate operator on pole
6   y ← y^(d)
```

ALGORITHM 4.1 Application of a tensor product operator $\mathfrak{L}$ with the unidirectional principle. Inputs are the vector $\boldsymbol{u} = (u_{\boldsymbol{k}})_{\boldsymbol{k}\in K}$ of input data, the set K of grid indices, and the permutation $(t_1,\dots,t_d)$ specifying the order in which the one-dimensional operators $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ should be applied. The output is the vector $\boldsymbol{y} = (y_{\boldsymbol{k}})_{\boldsymbol{k}\in K}$ of output data.

sum over k_{t_j} equals

$$\sum_{\boldsymbol{k}''\in K_{\text{pole}}} y^{(j)}_{\boldsymbol{k}''}\varphi_{k''_{t_j}}(x_{k'_{t_j}}) = \left((\mathfrak{L}^{(t_j),K_{\text{pole}}})^{-1}\left[(y^{(j)}_{\boldsymbol{k}''})_{\boldsymbol{k}''\in K_{\text{pole}}}\right]\right)_{k'_{t_j}} = y^{(j-1)}_{(\boldsymbol{k}_{T'},\boldsymbol{k}'_{-T'})} \tag{4.10}$$

by line 5 of Alg. 4.1. We can conclude that the LHS Eq. (4.9) equals

$$\sum_{\boldsymbol{k}_{T'}=\boldsymbol{0}}^{\boldsymbol{2}^{\boldsymbol{\ell}_{T'}}} y^{(j-1)}_{(\boldsymbol{k}_{T'},\boldsymbol{k}'_{-T'})}\varphi_{\boldsymbol{k}_{T'}}(\boldsymbol{x}_{\boldsymbol{k}'_{T'}}), \tag{4.11}$$

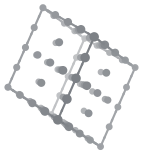
which, by the induction hypothesis, equals $f(\boldsymbol{x}_{\boldsymbol{k}'})$ as desired (if $j > 1$). The same reasoning for (4.10) can be used to establish the base case for $j = 1$. ■

COROLLARY 4.2 *Algorithm 4.1 is correct for hierarchization on full grids.*

PROOF We apply Prop. 4.1 for $j = d$ to obtain $\sum_{\boldsymbol{k}=\boldsymbol{0}}^{\boldsymbol{2}^{\boldsymbol{\ell}}} y^{(j)}_{\boldsymbol{k}}\varphi_{\boldsymbol{k}}(\boldsymbol{x}_{\boldsymbol{k}'}) = f(\boldsymbol{x}_{\boldsymbol{k}'})$ for all $\boldsymbol{k}' = \boldsymbol{0},\dots,\boldsymbol{2}^{\boldsymbol{\ell}}$, i.e., the $y^{(j)}_{\boldsymbol{k}}$ are the correct interpolation coefficients according to (4.1). ■

Complexity. We compare the complexity of the UP for hierarchization with the direct solution of the system (4.5) of linear equations. If we assume that d is constant and that $\mathfrak{L}$ and $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ apply Gaussian elimination to solve the multivariate and univariate systems, respectively, then directly solving (4.5) takes $\mathcal{O}(N^2(N+d))$ time and $\mathcal{O}(N^2)$ memory. In contrast, the UP only requires $\mathcal{O}(N\sum_t N_t^2)$ time[3] and $\mathcal{O}(\max\{N_1^2,\dots,N_d^2,N\})$ memory, where N_t is the grid size $|\{k_t \mid \boldsymbol{k}\in K\}|$ in dimension $t = 1,\dots,d$. The dependency from the univariate grid sizes N_t instead of N makes the UP significantly less computationally

[3] There are N/N_t poles in the t-th iteration of Alg. 4.1. Each pole requires the solution of an $N_t \times N_t$ linear system, which takes $\mathcal{O}(N_t^3)$ time.



expensive. As already mentioned, the UP is even more efficient in the piecewise linear case, where the univariate interpolation operators can be applied in-place. Hence, it only needs $\mathcal{O}(Nd)$ time and $\mathcal{O}(N)$ memory in this case.

4.3 Hierarchization on Dimensionally Adaptive Sparse Grids

IN THIS SECTION

Dimensionally adaptive sparse grids, which are sums of different hierarchical subspaces as described in Sec. 2.3.2, have the advantage over general spatially adaptive sparse grids that algorithms can be formulated and applied more easily. In this section, we describe two methods: first, the well-known combination technique, which was already mentioned in Sec. 2.3.2, and second, a new algorithm based on residual interpolation.

4.3.1 The Combination Technique and Its Combinatorial Proof

The combination technique was one of the first methods that were developed by Griebel et al. in [Gri92] (for two and three dimensions) after the term "sparse grids" was coined in 1991 [Zen91]. However, the combination technique predates the development of sparse grids by decades, as it was already mentioned by Smolyak in 1963 [Smo63; Heg07]. Delvos developed and proved the standard combination formula in the framework of Boolean interpolation operators in 1982 [Delv82; Delv89].

Formal description and outline of a combinatorial proof. In the following, we give a formal description of the sparse grid combination technique and we outline a new combinatorial proof of its correctness. While we discuss a high-level explanation of the proofs in this section, the proofs themselves can be found in Appendix A.3.1, since most of them are rather technical. For simplicity, we formulate the combination technique and its proof for regular sparse grids (see Sec. 2.3.1). However, the main ideas of the chain of proofs are also applicable to dimensionally adaptive sparse grids (see Sec. 2.3.2).

THEOREM 4.3 (sparse grid combination technique)
Let $K := \{(\boldsymbol{\ell}, \boldsymbol{i}) \mid \|\boldsymbol{\ell}\|_1 \leq n,\ \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}$ *correspond to the regular sparse grid* $\Omega^{\mathrm{s}}_{n,d}$ *and let* $(f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}))_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ *be given function values on* $\Omega^{\mathrm{s}}_{n,d}$. *If we define*

- *the combined sparse grid interpolant* $f_{n,d}^{\mathrm{s,ct}}$ *via* (2.31)*, i.e.,*

$$f_{n,d}^{\mathrm{s,ct}} = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\boldsymbol{\ell}'\|_1 = n-q} f_{\boldsymbol{\ell}'}, \tag{4.12}$$

  where $f_{\boldsymbol{\ell}'} \in V_{\boldsymbol{\ell}'}$ *is the full grid interpolant of* f *with level* $\boldsymbol{\ell}'$*, and*

- *the hierarchical sparse grid interpolant* $f_{n,d}^{\mathrm{s}}$ *via* (4.1) *and* (4.2)

and if we assume that the hierarchical splitting equation (2.22) *holds, then the combined and the hierarchical sparse grid interpolants coincide:*

$$f_{n,d}^{\mathrm{s,ct}} = f_{n,d}^{\mathrm{s}}. \tag{4.13}$$

PROOF (SKETCH) Let $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in \Omega_{n,d}^{\mathrm{s}}$ be an arbitrary point of the regular sparse grid. First, we split the inner sum of $f_{n,d}^{\mathrm{s,ct}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ into levels $\boldsymbol{\ell}'$ whose full grid sets $\Omega_{\boldsymbol{\ell}'}$ contain $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ and levels whose full grid sets do not contain $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$:

$$f_{n,d}^{\mathrm{s,ct}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \left(\sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}}} f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) + \sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\boldsymbol{\ell}'} \not\ni \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}}} f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \right). \tag{4.14}$$

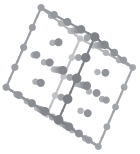
The summands $f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ of the first inner sum each equal $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ due to the full grid interpolation property (2.11). Therefore, the first inner sum is equal to the product of $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ with the number of summands:

$$\begin{aligned} f_{n,d}^{\mathrm{s,ct}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = {} & f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \cdot \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \left| \{\boldsymbol{\ell}' \mid \|\boldsymbol{\ell}'\|_1 = n-q,\ \Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}\} \right| \\ & + \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\boldsymbol{\ell}'} \not\ni \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}}} f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \end{aligned} \tag{4.15}$$

After this sketch of proof, we will prove that the first of the two summands in Eq. (4.15) equals one (see Prop. 4.4) and that the second of the two summands equals zero (see Prop. 4.8). Consequently, we infer

$$f_{n,d}^{\mathrm{s,ct}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \tag{4.16}$$

i.e., $f_{n,d}^{\mathrm{s,ct}}$ interpolates f at $\Omega_{n,d}^{\mathrm{s}}$. Note that $f_{n,d}^{\mathrm{s,ct}}$ is contained in $V_{n,d}^{\mathrm{s}}$, if the hierarchical



splitting equation (2.22) holds, as

$$f_{\boldsymbol{\ell}'} \in V_{\boldsymbol{\ell}'} = \bigoplus_{\boldsymbol{\ell}''=\mathbf{0}}^{\boldsymbol{\ell}'} W_{\boldsymbol{\ell}''} \subseteq V^{\mathrm{s}}_{n,d}, \quad \|\boldsymbol{\ell}'\|_1 \leq n, \tag{4.17}$$

due to $\|\boldsymbol{\ell}''\|_1 \leq \|\boldsymbol{\ell}'\|_1 \leq n$ for $\boldsymbol{\ell}'' \leq \boldsymbol{\ell}'$, i.e., $W_{\boldsymbol{\ell}''} \subseteq V^{\mathrm{s}}_{n,d}$ for $\boldsymbol{\ell}'' = \mathbf{0}, \ldots, \boldsymbol{\ell}'$.[4] As both $f^{\mathrm{s,ct}}_{n,d}$ and $f^{\mathrm{s}}_{n,d}$ are contained in $V^{\mathrm{s}}_{n,d}$ and interpolate f on $\Omega^{\mathrm{s}}_{n,d}$, they coincide due to the uniqueness of sparse grid interpolation (linear independence of the hierarchical basis). ■

Inclusion-exclusion principle. It remains to prove that the first sum in (4.15) is indeed one and that the second sum vanishes. The first statement is a direct consequence of the *inclusion-exclusion principle* [Heg07]. In its simplest form, the idea of the principle is that the cardinality of the union of two finite subsets A and B of some set is given by

$$|A \cup B| = |A| + |B| - |A \cap B|, \tag{4.18}$$

i.e., we first count (include) the elements in A and then in B, but as we have counted the elements of $A \cap B$ twice, we have to subtract (exclude) its cardinality again.

The setting is similar for the combination technique. If we add all grids in Fig. 2.6 on the green diagonal, then every point whose index is not odd will be counted multiple times. By subtracting the number of occurrences of the points on the red diagonal, the result of the "weighted counting" is exactly one for every point. The following proposition, whose proof is of purely combinatorial nature, generalizes this argument to higher dimensions:

PROPOSITION 4.4 (inclusion-exclusion principle)
For every $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in \Omega^{\mathrm{s}}_{n,d}$, we have

$$\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \left|\{\boldsymbol{\ell}' \mid \|\boldsymbol{\ell}'\|_1 = n-q,\ \Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}\}\right| = 1. \tag{4.19}$$

PROOF See Appendix A.3.1. ■

Canceling out function values. The second statement about the vanishing second sum in (4.15) is much harder to prove. It says that at every grid point $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$, the contributions $f_{\boldsymbol{\ell}'}$ of levels $\boldsymbol{\ell}'$ that do not contain that point cancel out, which may seem quite surprising. The key observation is as follows: The values of $f_{\boldsymbol{\ell}'}, f_{\boldsymbol{\ell}''}$ for two levels $\boldsymbol{\ell}', \boldsymbol{\ell}''$ are the same at $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$, if all non-equal entries ℓ'_t, ℓ''_t of the levels are each greater or equal to ℓ_t.

[4] This argumentation can be straightforwardly adapted for general dimensionally adaptive sparse grids with downward closed level sets as mentioned in Sec. 2.3.2.

For a higher-level explanation, note that the statement $\ell'_t \geq \ell_t$ is equivalent to $\Omega_{\ell'_t} \ni x_{\ell_t,i_t}$. Both $f_{\ell'}, f_{\ell''}$ interpolate at $\boldsymbol{x}_{\ell,i}$ when projected onto the t-th dimension, so their contribution to $f_{\ell'}(\boldsymbol{x}_{\ell,i})$ and $f_{\ell''}(\boldsymbol{x}_{\ell,i})$ must be the same. Although there may be dimensions t for which $\Omega_{\ell'_t} \not\ni x_{\ell_t,i_t}$, these dimensions do not matter if $\ell'_t = \ell''_t$, as the univariate restrictions of $f_{\ell'}, f_{\ell''}$ interpolate the same data and they are evaluated at the same point x_{ℓ_t,i_t}.

One can formalize these considerations by defining an equivalence relation on the set of levels such that the values of $f_{\ell'}$ at $\boldsymbol{x}_{\ell,i}$ are constant on the equivalence classes.

DEFINITION 4.5 (equivalence relation for the proof of the combination technique) Let $\boldsymbol{x}_{\ell,i} \in \Omega^{\mathrm{s}}_{n,d}$ be fixed and

$$L := \{\boldsymbol{\ell}' \mid \exists_{q=0,\ldots,d-1} \|\boldsymbol{\ell}'\|_1 = n-q,\ \Omega_{\boldsymbol{\ell}'} \not\ni \boldsymbol{x}_{\ell,i}\} \tag{4.20}$$

be the set of levels that do not contain $\boldsymbol{x}_{\ell,i}$. We define a relation $\sim$ on L as follows: For $\boldsymbol{\ell}', \boldsymbol{\ell}'' \in L$, we set $\boldsymbol{\ell}' \sim \boldsymbol{\ell}''$ if and only if

$$\forall_{t \notin T_{\ell',\ell''}} \ \min\{\ell'_t, \ell''_t\} \geq \ell_t, \quad T_{\ell',\ell''} := \{t \mid \ell'_t = \ell''_t < \ell_t\}. \tag{4.21}$$

LEMMA 4.6 *Let $\boldsymbol{\ell}', \boldsymbol{\ell}'' \in L$ with $\boldsymbol{\ell}' \sim \boldsymbol{\ell}''$. Then, $f_{\ell'}(\boldsymbol{x}_{\ell,i}) = f_{\ell''}(\boldsymbol{x}_{\ell,i})$.*

PROOF See Appendix A.3.1. ∎

By exploiting the tensor product structure of the basis functions, the proof shows an even stronger version, which is shown in Fig. 4.4: The components $f_{\ell'}$ and $f_{\ell''}$ are equal on the m-dimensional affine subspace through $\boldsymbol{x}_{\ell,i}$ parallel to the m coordinates in $T_{\ell',\ell''}$ (where $m := |T_{\ell',\ell''}|$). The lemma allows to group summands in the second sum of (4.15) by function values. Hence, it remains to count the number of levels in each equivalence class of $\sim$. Therefore, we need a characterization of the equivalence classes:

LEMMA 4.7 (characterization of equivalence classes)
Let $L_0 \in L/\sim$ be an equivalence class of $\sim$. If we define

$$T_{L_0} := \{t \mid \exists_{\ell^*_t < \ell_t} \forall_{\boldsymbol{\ell}' \in L_0} \ell'_t = \ell^*_t\} \tag{4.22}$$

as the set of dimensions t in which all levels in L_0 have the same entry $\ell^*_t < \ell_t$, then

$$L_0 = \{\boldsymbol{\ell}' \in L \mid \forall_{t \in T_{L_0}} \ell'_t = \ell^*_t,\ \forall_{t \notin T_{L_0}} \ell'_t \geq \ell_t\}. \tag{4.23}$$

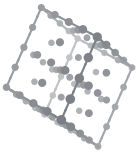
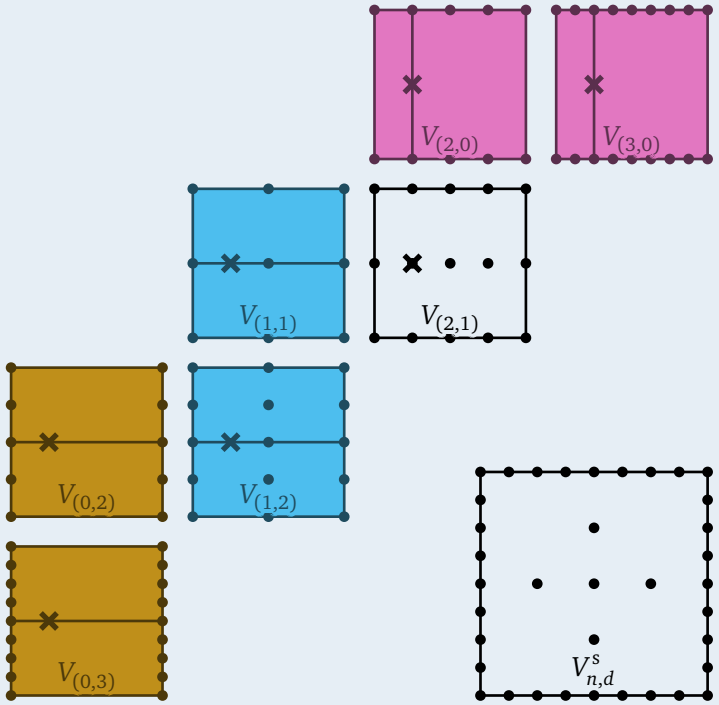


FIGURE 4.4
Nodal subspaces V_ℓ contributing to the combination technique solution for the two-dimensional regular sparse grid $V^s_{n,d}$ of level $n = 3$ *(bottom right)*. After picking a point $\boldsymbol{x}_{\ell,i} \in \Omega^s_{n,d}$ *(cross,* here $\boldsymbol{\ell} = (2,1)$, $\boldsymbol{i} = (1,1)$*)*, the set L of levels whose grids do not contain $\boldsymbol{x}_{\ell,i}$ *(colored subspaces)* decompose into three disjoint equivalence classes *(colors)* given by the relation $\sim$. In every equivalence class $L_0 \in L/\sim$, the interpolants $f_{\ell'}$ ($\boldsymbol{\ell}' \in L_0$) equal on an affine subspace *(dark lines),* which contains $\boldsymbol{x}_{\ell,i}$. Due to the combination coefficients, the contribution to the combined solution vanishes per equivalence class.



PROOF See Appendix A.3.1. ∎

The lemma states that every equivalence class L_0 is exactly the set of the levels whose entries are equal and smaller than ℓ_t in some dimensions (which are contained in T_{L_0}) and whose entries are greater or equal than ℓ_t in all other dimensions. While this statement may seem intuitively correct, the proof is rather technical. Finally, we are now able to show that the second sum in (4.15) vanishes:

PROPOSITION 4.8 (function value cancellation)
For every $\boldsymbol{x}_{\ell,i} \in \Omega^s_{n,d}$, we have

$$\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\ell'} \not\ni \boldsymbol{x}_{\ell,i}}} f_{\ell'}(\boldsymbol{x}_{\ell,i}) = 0. \tag{4.24}$$

PROOF See Appendix A.3.1. ∎

The proof essentially first counts the number of possible levels in an equivalence class and then applies known combinatorial identities to prove that the sum must vanish. This proves Thm. 4.3 (sparse grid combination technique).

1 **function** $y = \text{combinationTechnique}(u, n, d)$
2 **for** $q = 0, \ldots, d-1$ **do**
3 **for** $\ell' \in \mathbb{N}_0^d$ with $\|\ell'\|_1 = n - q$ **do**
4 Let $(\alpha_{\ell,i}^{(\ell')})_{\ell=0,\ldots,\ell', i\in I_\ell}$ be such that $\sum_{\ell=0}^{\ell'} \sum_{i\in I_\ell} \alpha_{\ell,i}^{(\ell')} \varphi_{\ell,i} \equiv f_{\ell'}$
5 $\alpha_{\ell,i}^{(\ell')} \leftarrow 0$ for all $(\ell, i) \in K$ with $\neg(\ell \le \ell')$ ⇝ *extend surpluses*
6 $y_{\ell,i} = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\ell'\|_1 = n-q} \alpha_{\ell,i}^{(\ell')}$ for all $(\ell, i) \in K$ ⇝ *combine surpluses*

ALGORITHM 4.2 Application of the hierarchization operator $\mathfrak{L} = A^{-1}$ with the combination technique. For simplicity, the algorithm is described for regular sparse grids, but it can be generalized to arbitrary dimensionally adaptive sparse grids. Inputs are the vector $u = (u_{\ell,i})_{(\ell,i)\in K}$ of input data (function values $f(x_{\ell,i})$ at the grid points), the level n, and the dimensionality d of the regular sparse grid, where K is the set of all feasible level-index pairs (ℓ, i), i.e., $\|\ell\|_1 \le n$, $i \in I_\ell$. The output is the vector $y = (y_{\ell,i})_{(\ell,i)\in K}$ of output data (hierarchical surpluses $\alpha_{\ell,i}$).

4.3.2 Hierarchization with the Combination Technique

It is straightforward to hierarchize function values $f(x_{\ell,i})$ on dimensionally adaptive sparse grids with the combination technique. The resulting hierarchization algorithm is given as Alg. 4.2. In line 4, the hierarchical surpluses corresponding to the full grid interpolant $f_{\ell'} \in V_{\ell'}$ have to be computed (see (2.11)). As shown in Sec. 4.2, we can easily calculate these surpluses with the unidirectional principle in Alg. 4.1. The surpluses are then combined with the same combination formula as in Thm. 4.3 (sparse grid combination technique). Note that it is imperative to employ the hierarchical basis functions $\varphi_{\ell,i}$ with $\ell = \mathbf{0}, \ldots, \ell'$ and $i \in I_\ell$ and not the nodal basis, i.e., $\varphi_{\ell',i'}$ with $i' = \mathbf{0}, \ldots, \mathbf{2}^{\ell'}$.

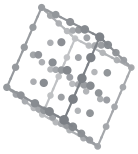
Correctness. Of course, the proof of the correctness of Alg. 4.2 relies on the correctness of the combination technique (see Thm. 4.3). If determining the combination coefficients correctly [Nob16], the algorithm can even be applied to all dimensionally adaptive sparse grids. The proof of the following proposition can be generalized accordingly.

PROPOSITION 4.9 (correctness of combination technique)
Algorithm 4.2 is correct for hierarchization on regular sparse grids.

PROOF According to line 4 of Alg. 4.2, the full grid interpolants $f_{\ell'}$ can be written as

$$f_{\ell'} = \sum_{\|\ell\|_1 \le n} \sum_{i\in I_\ell} \alpha_{\ell,i}^{(\ell')} \varphi_{\ell,i} \tag{4.25}$$

where the surpluses have been extended from $\ell = \mathbf{0}, \ldots, \ell'$ to all ℓ with $\|\ell\|_1 \le n$ by zero



in line 5. Theorem 4.3 now allows to write the hierarchical interpolant $f_{n,d}^s$ in terms of the full grid components:

$$f_{n,d}^s = f_{n,d}^{s,\mathrm{ct}} = \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\boldsymbol{\ell}'\|_1 = n-q} f_{\boldsymbol{\ell}'} \tag{4.26a}$$

$$= \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\boldsymbol{\ell}'\|_1 = n-q} \sum_{\|\boldsymbol{\ell}\|_1 \le n} \sum_{\boldsymbol{i} \in I_{\boldsymbol{\ell}}} \alpha_{\boldsymbol{\ell},\boldsymbol{i}}^{(\boldsymbol{\ell}')} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}. \tag{4.26b}$$

$$= \sum_{\|\boldsymbol{\ell}\|_1 \le n} \sum_{\boldsymbol{i} \in I_{\boldsymbol{\ell}}} \underbrace{\left(\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \sum_{\|\boldsymbol{\ell}'\|_1 = n-q} \alpha_{\boldsymbol{\ell},\boldsymbol{i}}^{(\boldsymbol{\ell}')} \right)}_{= y_{\boldsymbol{\ell},\boldsymbol{i}}} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}, \tag{4.26c}$$

where $y_{\boldsymbol{\ell},\boldsymbol{i}}$ is the $(\boldsymbol{\ell},\boldsymbol{i})$-th entry of the output vector of Alg. 4.2. Note that the hierarchical interpolant $f_{n,d}^s$ can be written as $f_{n,d}^s = \sum_{\|\boldsymbol{\ell}\|_1 \le n} \sum_{\boldsymbol{i} \in I_{\boldsymbol{\ell}}} \alpha_{\boldsymbol{\ell},\boldsymbol{i}} \varphi_{\boldsymbol{\ell},\boldsymbol{i}}$ (see (2.28)), where the surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ are unique due to the linear independence of the hierarchical basis. As (4.26c) equals $f_{n,d}^s$ and has the same form, the coefficients $y_{\boldsymbol{\ell},\boldsymbol{i}}$ must coincide with the surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$. ■

4.3.3 Hierarchization with Residual Interpolation

Another method to hierarchize function values on dimensionally adaptive sparse grids is the *method of residual interpolation*. The advantage over the combination technique is that it only needs to operate on so-called *active nodal spaces*. In contrast, the combination technique needs to perform computations on additional non-active nodal subspaces (for the regular sparse grid case: summands with $q \ge 1$ in (2.31)).

Active nodal spaces. Algorithm 4.3 describes the procedure of the method of residual interpolation, given the function values $\boldsymbol{u}$ corresponding to the grid points and the levels L contained in the sparse grid (see (2.30)). The list $\boldsymbol{\ell}^{(1)}, \ldots, \boldsymbol{\ell}^{(m)}$ of active nodal spaces in line 3 is determined by the condition

$$\bigcup_{j=1}^{m} \{\boldsymbol{\ell} \in \mathbb{N}_0^d \mid \boldsymbol{\ell} \le \boldsymbol{\ell}^{(j)}\} = L, \quad \forall_{j_1 \ne j_2} \ \neg(\boldsymbol{\ell}^{(j_1)} \le \boldsymbol{\ell}^{(j_2)}). \tag{4.27}$$

This means that the corresponding sparse grid Ω^s is the (non-disjoint) union of the full grid sets $\Omega_{\boldsymbol{\ell}^{(j)}}$ $(j = 1, \ldots, m)$ and no full grid set is contained in another, i.e., no full grid set can be omitted without removing points from the union Ω^s.

```
function y = residualInterpolation(u, L)
  r^(0)(x_{ℓ,i}) ← f(x_{ℓ,i}) for all (ℓ,i) ∈ K
  Compute list ℓ^(1),...,ℓ^(m) of active nodal spaces from L (see (4.27))
  Sort ℓ^(1),...,ℓ^(m) by decreasing level sum
  for j = 1,...,m do
    Let r^(j-1)_{ℓ^(j)} ∈ V_{ℓ^(j)} be the interpolant of r^(j-1) on Ω_{ℓ^(j)}
    Let (α^(j)_{ℓ,i})_{(ℓ,i)∈K} be such that Σ_{ℓ=0}^{ℓ^(j)} Σ_{i∈I_ℓ} α^(j)_{ℓ,i} φ_{ℓ,i} ≡ r^(j-1)_{ℓ^(j)}   ⇝ interpolation
    r^(j)(x_{ℓ,i}) ← r^(j-1)(x_{ℓ,i}) − r^(j-1)_{ℓ^(j)}(x_{ℓ,i}) for all (ℓ,i) ∈ K   ⇝ new residuals
  y ← Σ_{j=1}^{m} α^(j) (where α^(j)_{ℓ,i} = 0, (ℓ,i) ∈ K, if ¬(ℓ ≤ ℓ^(j)))   ⇝ combine surpluses
```

ALGORITHM 4.3 Application of the hierarchization operator $\mathfrak{L} = A^{-1}$ with residual interpolation for dimensionally adaptive sparse grids. Inputs are the vector $\boldsymbol{u} = (u_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of input data (function values $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ at the grid points) and the set L of levels that are part of the sparse grid (see (2.30)), where K is the set of all feasible level-index pairs $(\boldsymbol{\ell},\boldsymbol{i})$, i.e., $\boldsymbol{\ell} \in L$, $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$. The output is the vector $\boldsymbol{y} = (y_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of output data (hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$).

Correctness. The principle of Algorithm 4.3 is maintaining a vector $(r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}))_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of residuals and interpolating the residual data subsequently on the active nodal spaces. Again, note that it is necessary to compute the coefficients $\alpha^{(j)}_{\boldsymbol{\ell},\boldsymbol{i}}$ in the hierarchical basis, despite interpolating on the full grid $\Omega_{\boldsymbol{\ell}^{(j)}}$. In Appendix A, we prove that the algorithm satisfies the following invariant, which can be used to show its correctness:

PROPOSITION 4.10 (invariant of residual interpolation)
For $j = 1,\dots,m$, it holds

$$\text{(4.28a)} \quad r^{(j-1)}_{\boldsymbol{\ell}^{(j)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0, \qquad \boldsymbol{\ell} \le \boldsymbol{\ell}^{(j')},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}},\quad j' = 1,\dots,j-1,$$

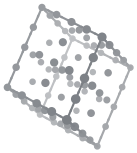
$$\text{(4.28b)} \quad r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0, \qquad \boldsymbol{\ell} \le \boldsymbol{\ell}^{(j')},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}},\quad j' = 1,\dots,j,$$

$$\text{(4.28c)} \quad r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - f^{\mathrm{s},(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad \boldsymbol{\ell} \in L,\quad \boldsymbol{i} \in I_{\boldsymbol{\ell}},$$

$$\text{(4.29)} \quad \text{where} \quad f^{\mathrm{s},(j)} := \sum_{\boldsymbol{\ell}'\in L} \sum_{\boldsymbol{i}'\in I_{\boldsymbol{\ell}'}} \left(\sum_{j'=1}^{j} \alpha^{(j')}_{\boldsymbol{\ell}',\boldsymbol{i}'} \right) \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}.$$

PROOF See Appendix A.3.2. ∎

COROLLARY 4.11 (correctness of residual interpolation)
Algorithm 4.3 is correct for hierarchization on dimensionally adaptive sparse grids.



PROOF Let $\boldsymbol{\ell} \in L$ and $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$. By construction of the active nodal spaces, there exists some $j' \in \{1, \dots, m\}$ such that $\boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j')}$. By Prop. 4.10, we obtain for $j = m$

$$\sum_{\boldsymbol{\ell}' \in L} \sum_{\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}} \underbrace{\left(\sum_{j''=1}^{m} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}^{(j'')} \right)}_{=y_{\boldsymbol{\ell}',\boldsymbol{i}'}} \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \overset{(4.29)}{=} f^{\mathrm{s},(m)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \overset{(4.28c)}{=} f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - r^{(m)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \tag{4.30a}$$

$$\overset{(4.28b)}{=} f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{4.30b}$$

As the hierarchical interpolant f^{s} (see (4.2)) has the same form $\sum_{\boldsymbol{\ell}' \in L} \sum_{\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}$ as the LHS of (4.30) with unique surpluses $\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}$ such that the function values are interpolated (see (4.1)), the coefficients $y_{\boldsymbol{\ell}',\boldsymbol{i}'}$ (output of Alg. 4.3) coincide with the surpluses $\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}$. ■

Proposition 4.10 shows that $r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ is the residual of the interpolant $f^{\mathrm{s},(j)}$ of iteration j to the objective function f at the grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ (Eq. (4.28c)). After interpolating $r^{(j-1)}$ on the grid $\Omega_{\boldsymbol{\ell}^{(j)}}$ to obtain the function $r_{\boldsymbol{\ell}^{(j)}}^{(j-1)}$ and subtracting the resulting values from the old residual values, the new residual values $r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ vanish not only on the grid $\{(\boldsymbol{\ell}^{(j)}, \boldsymbol{i}) \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}^{(j)}}\}$, but also on all previous grids $\{(\boldsymbol{\ell}^{(j')}, \boldsymbol{i}) \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}^{(j')}}\}$, $j' \leq j$ (Eq. (4.28b)). The proof of Prop. 4.10 shows this by exploiting the auxiliary statement of Eq. (4.28a) and the tensor product structure of the hierarchical basis.

An example for the application of Alg. 4.3 on a two-dimensional sparse grid can be seen in Fig. 4.5. Note that $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}^{(j)} \neq 0$ can only be true if $\boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j)}$. Therefore, if $(\boldsymbol{\ell}, \boldsymbol{i})$ is not contained in one of the grids that are processed in one of the remaining iterations $j+1, \dots, m$, then $y_{\boldsymbol{\ell},\boldsymbol{i}}^{(j)}$ is already equal to the correct surplus $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$, where $y_{\boldsymbol{\ell},\boldsymbol{i}}^{(j)} := \sum_{j'=1}^{j} \alpha_{\boldsymbol{\ell},\boldsymbol{i}}^{(j')}$ denotes the intermediate result obtained after j iterations.

4.4 Hierarchization on Spatially Adaptive Sparse Grids with Breadth-First Search

IN THIS SECTION

Unfortunately, we cannot apply the algorithms presented in the last sections to spatially adaptive sparse grids with hierarchical B-splines. The reason is that the algorithms relied on the final interpolant f^{s} being a linear combination of full grid solutions $f_{\boldsymbol{\ell}}$, which is only possible for dimensionally adaptive sparse grids. Consequently, the problem of hierarchization becomes significantly harder if we operate on spatially adaptive sparse grids. An exception is the case of piecewise linear basis functions ($p = 1$), where we are

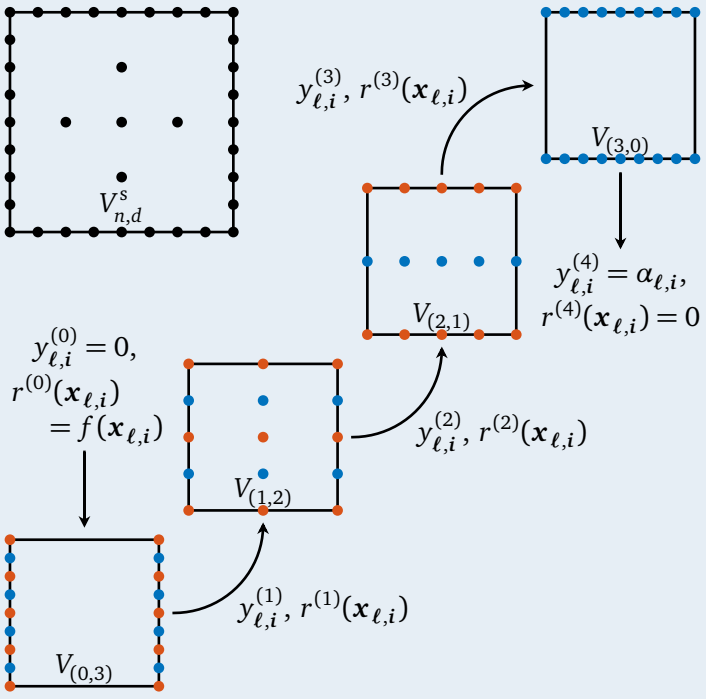


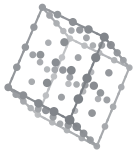
FIGURE 4.5
Hierarchization of function value data on the two-dimensional regular sparse grid $V^s_{n,d}$ of level $n = 3$ *(top left)* using the method of residual interpolation. In this figure, we use $y^{(j)}_{\ell,i} := \sum_{j'=1}^{j} \alpha^{(j')}_{\ell,i}$ as an abbreviation. The order of the nodal spaces (here: bottom left to top right) does not matter. The data $y^{(j)}_{\ell,i}$ corresponding to blue grid points will not be modified in the remaining iterations and, therefore, already equals the correct surpluses $\alpha_{\ell,i}$. The data corresponding to red grid points will be modified as the grid points appear in one of the remaining nodal grids.

still able to apply the UP, as we will show in Sec. 4.5. In this section, we study one approach to hierarchize on spatially adaptive sparse grids, namely transforming the hierarchical basis to so-called fundamental bases to enable a BFS algorithm for hierarchization.

The approach in this section has already been published [Vale18a]. Again, note that while B-splines are our target application, the considerations in this chapter are fully independent of the choice of basis functions $\varphi_{\ell,i}$, as long as they have tensor product structure. Although we do not state it explicitly, it is possible to employ different types of basis functions φ_{ℓ_t,i_t} in different dimensions, e.g., B-splines of different degrees p_t to enable p-adaptivity.

4.4.1 Hierarchization with Breadth-First Search on Fundamental Bases

Fundamental property. As already discussed in Sec. 4.1, the main cause of the difficulty of the hierarchization with B-splines $\varphi^p_{\ell,i}$ is their overlapping support (which they need for their approximation order). Thus, high-level B-splines $\varphi^p_{\ell',i'}$ do not vanish at all coarse-level grid points $\boldsymbol{x}_{\ell,i}$, $\boldsymbol{\ell} < \boldsymbol{\ell}'$. In the univariate case, the idea is to transform the B-spline



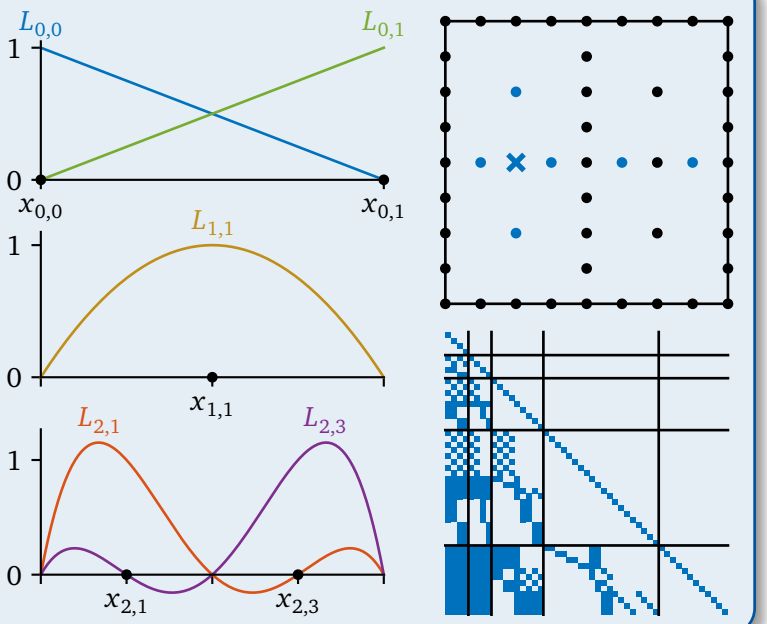


FIGURE 4.6
Fundamental property with Lagrange polynomials.
Left: Univariate Lagrange polynomials up to level $\ell = 2$.
Top right: Regular sparse grid $\Omega^{s(1)}_{n,d}$ ($n = 4$, $d = 2$). The fundamental basis function $\varphi^{f}_{\ell',i'}$ corresponding to the marked grid point *(cross)* does not vanish at the blue points $\boldsymbol{x}_{\ell,i}$ (which satisfy (4.32)).
Bottom right: Corresponding density pattern of $\boldsymbol{A}$ when sorting rows and columns by increasing level sum $\|\boldsymbol{\ell}\|_1 = 0,\dots,n$ *(black bars)*.

basis to obtain new basis functions $\varphi^{f}_{\ell',i'}\colon [0,1] \to \mathbb{R}$ ($\ell' \in \mathbb{N}_0$, $i' \in I_{\ell'}$) that satisfy

$$\varphi^{f}_{\ell',i'}(x_{\ell,i}) = 0, \qquad \ell < \ell', \quad i \in I_\ell, \tag{4.31a}$$

$$\varphi^{f}_{\ell',i'}(x_{\ell',i}) = \delta_{i,i'}, \qquad\qquad i \in I_{\ell'}. \tag{4.31b}$$

We call (4.31) *fundamental property* and functions $\varphi^{f}_{\ell',i'}$ that fulfill this property *fundamental basis functions*. The first Eq. (4.31a) ensures that basis functions of level ℓ' vanish at grid points of coarser levels $\ell < \ell'$. The second Eq. (4.31b) requires the basis functions $\varphi^{f}_{\ell',i'}$ to additionally vanish at all grid points of the same level ℓ' with different index $i \neq i'$. An example for fundamental basis functions are the piecewise linear B-splines $\varphi^{1}_{\ell',i'}$ or the Lagrange polynomials $L_{\ell',i'}$ (see Fig. 4.6, left). The statement that $\varphi^{f}_{\ell',i'}(x_{\ell',i'})$ should equal one is not an additional restriction, if the value $\varphi^{f}_{\ell',i'}(x_{\ell',i'})$ is non-zero, since we can just replace $\varphi^{f}_{\ell',i'}$ with $\varphi^{f}_{\ell',i'}/\varphi^{f}_{\ell',i'}(x_{\ell',i'})$ to obtain $\varphi^{f}_{\ell',i'}(x_{\ell',i'}) = 1$.

Multivariate case. For the multivariate case of $d \in \mathbb{N}$ dimensions, we define as usual tensor product versions $\varphi^{f}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ of the univariate fundamental bases $\varphi^{f}_{\ell'_t,i'_t}$ ($t = 1,\dots,d$). Equation (4.31) then implies

$$\varphi^{f}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \neq 0 \implies \forall_{t=1,\dots,d}\ \big[(\ell'_t < \ell_t) \vee ((\ell'_t, i'_t) = (\ell_t, i_t))\big], \quad (\boldsymbol{\ell},\boldsymbol{i}),(\boldsymbol{\ell}',\boldsymbol{i}') \in K. \tag{4.32}$$

This means that every basis function $\varphi^{f}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ can only be non-zero at the grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ that, in every dimension t, have a strictly higher level ℓ_t or the same level-index pair (ℓ_t, i_t) as

the basis function. We show an example for this relation in Fig. 4.6 (top right).

Triangular interpolation matrix. The main motivation for enforcing the fundamental property is the fact that it results in the hierarchization matrix $\boldsymbol{A}$ being triangular, if the rows and columns are arranged in the order of monotonously increasing level sum: We assume that $k = k(\boldsymbol{\ell}, \boldsymbol{i}) \in \{1, \dots, N\}$ is a single continuously enumerated index of the level-index pairs $(\boldsymbol{\ell}, \boldsymbol{i}) \in K$ (where $N = |K|$) such that

$$k(\boldsymbol{\ell}, \boldsymbol{i}) \leq k(\boldsymbol{\ell}', \boldsymbol{i}') \implies \|\boldsymbol{\ell}\|_1 \leq \|\boldsymbol{\ell}'\|_1, \quad (\boldsymbol{\ell}, \boldsymbol{i}), (\boldsymbol{\ell}', \boldsymbol{i}') \in K, \tag{4.33}$$

i.e., we sort the row indices $k = k(\boldsymbol{\ell}, \boldsymbol{i})$ and the column indices $k' = k(\boldsymbol{\ell}', \boldsymbol{i}')$ of $\boldsymbol{A}$ by level sum $\|\cdot\|_1$. Consequently, $\boldsymbol{A} = (A_{k,k'})_{k=1,\dots,N,\, k'=1,\dots,N}$ is in lower block-triangular form:

$$A_{k,k'} = \varphi^{\mathrm{f}}_{k'}(\boldsymbol{x}_k) = 0, \qquad \|\boldsymbol{\ell}\|_1 < \|\boldsymbol{\ell}'\|_1, \tag{4.34a}$$

as $\|\boldsymbol{\ell}\|_1 < \|\boldsymbol{\ell}'\|_1 \implies \exists_t\, \ell_t < \ell'_t$ and using (4.31a). Additionally, the diagonal blocks are unit matrices due to

$$A_{k,k'} = \varphi^{\mathrm{f}}_{k'}(\boldsymbol{x}_k) \overset{(*)}{=} \delta_{(\boldsymbol{\ell},\boldsymbol{i}),(\boldsymbol{\ell}',\boldsymbol{i}')}, \quad \|\boldsymbol{\ell}\|_1 = \|\boldsymbol{\ell}'\|_1, \tag{4.34b}$$

since $\|\boldsymbol{\ell}\|_1 = \|\boldsymbol{\ell}'\|_1$ implies that either $\left[\exists_t\, \ell_t < \ell'_t\right]$ or $\boldsymbol{\ell} = \boldsymbol{\ell}'$.[5] In the former case, both sides of $(*)$ vanish (according to (4.31a)), and in the latter case, both sides equal $\delta_{\boldsymbol{i},\boldsymbol{i}'}$ (according to (4.31b)). Hence, $\boldsymbol{A}$ is a lower-triangular matrix. This is visualized for a two-dimensional example in Fig. 4.6 (bottom right).

Forward substitution. The triangular structure of $\boldsymbol{A}$ implies that we can determine the surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ via forward substitution:

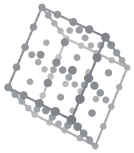
LEMMA 4.12 (forward substitution)
The hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$, which are determined by (4.5) *with respect to $\varphi^{\mathrm{f}}_{\boldsymbol{\ell},\boldsymbol{i}}$, satisfy*

$$\alpha_{\boldsymbol{\ell},\boldsymbol{i}} = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\substack{(\boldsymbol{\ell}',\boldsymbol{i}') \in K \\ \|\boldsymbol{\ell}'\|_1 < \|\boldsymbol{\ell}\|_1}} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad (\boldsymbol{\ell}, \boldsymbol{i}) \in K. \tag{4.35}$$

PROOF The linear system (4.5) is given by

$$f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = \sum_{(\boldsymbol{\ell}',\boldsymbol{i}') \in K} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad (\boldsymbol{\ell}, \boldsymbol{i}) \in K. \tag{4.36a}$$

[5] Note that as specified in the list of symbols at the beginning of this thesis, the Kronecker delta $\delta_{X,Y}$ is defined for arbitrary objects X and Y that can be compared with "=."



```
1  function y = breadthFirstSearch(u, K)
2    y ← u
3    K_p ← {0} × {0,1}^d                                  ⇝ set of processed points
4    Q ← FIFO queue initialized with contents of K_p      ⇝ points to be processed
5    while Q ≠ ∅ do
6      (ℓ', i') ← Q.pop()                                 ⇝ obtain next point
7      for {(ℓ,i) ∈ K \ {(ℓ',i')} | ∀_{t=1,...,d} (ℓ'_t < ℓ_t) ∨ ((ℓ'_t, i'_t) = (ℓ_t, i_t))} do
8        y_{ℓ,i} ← y_{ℓ,i} − y_{ℓ',i'} φ^f_{ℓ',i'}(x_{ℓ,i})   ⇝ update surpluses according to Lemma 4.12
9      for {(ℓ,i) ∈ K \ K_p | (ℓ,i) direct child of (ℓ',i')} do
10       Q.push((ℓ,i))                                    ⇝ add children to queue
11       K_p ← K_p ∪ {(ℓ,i)}                              ⇝ mark as processed
```

ALGORITHM 4.4 Hierarchization with breadth-first search on spatially adaptive sparse grids with fundamental bases. Inputs are the vector $\boldsymbol{u} = (u_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of input data (function values $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ at the grid points) and the set K of level-index pairs of the sparse grid (see (2.32)). The output is the vector $\boldsymbol{y} = (y_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of output data (hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$).

According to (4.34), all summands with $\|\boldsymbol{\ell}'\|_1 > \|\boldsymbol{\ell}\|_1$ vanish and from the summands with $\|\boldsymbol{\ell}'\|_1 = \|\boldsymbol{\ell}\|_1$, only the $(\boldsymbol{\ell},\boldsymbol{i})$-th summand remains with $\varphi^{\mathrm{f}}_{\boldsymbol{\ell},\boldsymbol{i}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 1$:

$$\cdots = \alpha_{\boldsymbol{\ell},\boldsymbol{i}} + \sum_{\substack{(\boldsymbol{\ell}',\boldsymbol{i}')\in K \\ \|\boldsymbol{\ell}'\|_1 < \|\boldsymbol{\ell}\|_1}} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{4.36b}$$

■

Breadth-first search. Exploiting this lemma, we formulate a hierarchization algorithm (see Alg. 4.4) that applies forward substitution by BFS in the directed acyclic graph (DAG) of the spatially adaptive sparse grid Ω^{s}. The nodes of the DAG are the level-index pairs $(\boldsymbol{\ell},\boldsymbol{i}) \in K$. An edge connects $(\boldsymbol{\ell},\boldsymbol{i})$ to $(\boldsymbol{\ell}',\boldsymbol{i}')$, if $(\boldsymbol{\ell},\boldsymbol{i})$ is a direct ancestor of $(\boldsymbol{\ell}',\boldsymbol{i}')$, i.e., if

$$\exists_{t=1,\ldots,d}\ \boldsymbol{\ell}'_{-t} = \boldsymbol{\ell}_{-t},\ \boldsymbol{i}'_{-t} = \boldsymbol{i}_{-t},\ \ell'_t = \ell_t + 1,\ i'_t \in \begin{cases} \{1\}, & \ell_t = 0, \\ \{2i_t - 1, 2i_t + 1\}, & \ell_t > 0. \end{cases} \tag{4.37}$$

An example for the resulting DAG for a regular sparse grid in two dimensions is shown in Fig. 4.7. We make two assumptions for Alg. 4.4. First, K should contain at least all 2^d corners of the domain $[\boldsymbol{0},\boldsymbol{1}]$:

$$K \supseteq \{\boldsymbol{0}\} \times \{0,1\}^d = \{(\boldsymbol{0},\boldsymbol{i}) \mid \boldsymbol{i} \in \{0,1\}^d\}. \tag{4.38a}$$

Second, all grid points should be reachable from the corners:

$$\forall_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \exists_{m\in\mathbb{N}_0} \exists_{(\boldsymbol{\ell}^{(0)},\boldsymbol{i}^{(0)}),\ldots,(\boldsymbol{\ell}^{(m)},\boldsymbol{i}^{(m)})\in K} \big[(\boldsymbol{\ell}^{(0)},\boldsymbol{i}^{(0)}) \to \cdots \to (\boldsymbol{\ell}^{(m)},\boldsymbol{i}^{(m)}), \\ \boldsymbol{\ell}^{(0)} = \boldsymbol{0},\ (\boldsymbol{\ell}^{(m)},\boldsymbol{i}^{(m)}) = (\boldsymbol{\ell}',\boldsymbol{i}')\big], \tag{4.38b}$$

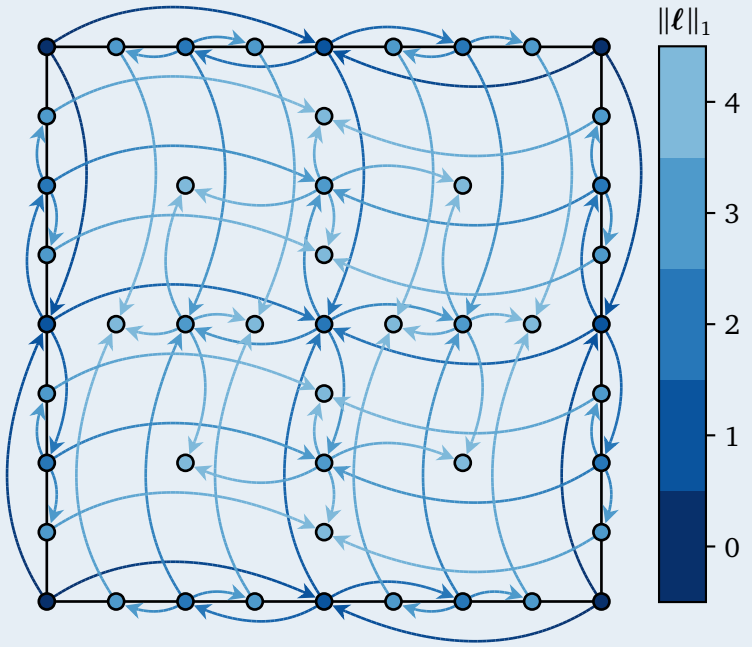


FIGURE 4.7
Ancestor relationships *(arrows)* in a regular sparse grid $\Omega_{n,d}^{s(1)}$ *(points)* of level $n = 4$ and dimensionality $d = 2$. The color indicates the level sum $\|\boldsymbol{\ell}\|_1$. Breadth-first search, as implemented in Alg. 4.4, visits all grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ with level sum $\|\boldsymbol{\ell}\|_1 = 0$ first, then those with $\|\boldsymbol{\ell}\|_1 = 1$, and so on.

where "→" is the direct ancestor relation (4.37). One can also use a different initial set than the corners of $[\mathbf{0}, \mathbf{1}]$, e.g., when working with sparse grids without boundary points. In general, there are three requirements on the initial set: First, all grid points should be reachable from this set. Second, the grid points in the set are sorted by increasing level sum (if the set contains grid points with different level sums). Third, the surpluses corresponding to the initial grid points need to be pre-calculated correctly before the **`while`** loop in Alg. 4.4.

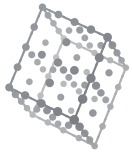
Correctness. The correctness of Alg. 4.4 can be shown with the following invariant:

PROPOSITION 4.13 (invariant of breadth-first-search hierarchization)
Under the assumption (4.38)*, it holds after* **pop***ping all grid points with level sum* $< q$ *from the queue* Q *in Alg. 4.4:*

$$y_{\boldsymbol{\ell},\boldsymbol{i}} = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\substack{(\boldsymbol{\ell}',\boldsymbol{i}')\in K \\ \|\boldsymbol{\ell}'\|_1 < q}} y_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad (\boldsymbol{\ell},\boldsymbol{i}) \in K, \ \|\boldsymbol{\ell}\|_1 = q. \tag{4.39}$$

PROOF See Appendix A.3.3. ■

COROLLARY 4.14 *Algorithm 4.4 is correct.*



PROOF As noted in the proof of Prop. 4.13, the result $y_{\ell,i}$ after popping all grid points with level sum q as stated in Prop. 4.13 is also the final result of the algorithm:

$$y_{\ell,i} = f(\boldsymbol{x}_{\ell,i}) - \sum_{\substack{(\ell',i')\in K \\ \|\ell'\|_1 < \|\ell\|_1}} y_{\ell',i'} \varphi^{\mathrm{f}}_{\ell',i'}(\boldsymbol{x}_{\ell,i}), \quad (\ell,i) \in K. \tag{4.40}$$

By Lemma 4.12, the correct hierarchical surpluses $\alpha_{\ell,i}$ satisfy the same relation. Inductively, $y_{\ell,i}$ and $\alpha_{\ell,i}$ must coincide. ∎

Complexity. The BFS algorithm in Alg. 4.4 is not as efficient as the UP: It still needs to perform $\mathcal{O}(N^2 d)$ many univariate basis evaluations (compared to $\mathcal{O}(Nd)$ for the UP). However, it only needs linear space $\mathcal{O}(N)$ similar to the UP. This is a significant advantage over directly solving the system (4.5) of linear equations, which typically needs quadratic space $\mathcal{O}(N^2)$.

4.4.2 Constructing Fundamental Bases

Unfortunately, the hierarchical B-splines $\varphi^{p}_{\ell',i'}$ do not satisfy the fundamental property (4.31). We now focus on the construction of univariate fundamental bases $\varphi^{\mathrm{f}}_{\ell',i'}$ starting from an arbitrary hierarchical basis $\varphi_{\ell',i'}$. To this end, we study two transformations $\varphi_{\ell',i'} \mapsto \varphi^{\mathrm{f}}_{\ell',i'}$. As usual, the multivariate case is treated with the tensor product approach.

Hierarchical fundamental transformation (HFT). The canonical way to find a fundamental basis $\varphi^{\mathrm{f}}_{\ell',i'}$ is to use a linear combination $\varphi^{\mathrm{hft}}_{\ell',i'}$ of coarser basis functions $\varphi_{\ell'',i''}$ as an ansatz and require that the fundamental property (4.31) is fulfilled:

$$\varphi^{\mathrm{hft}}_{\ell',i'} := \sum_{\ell''=0}^{\ell'} \sum_{i''\in I_{\ell''}} c^{\ell',i'}_{\ell'',i''} \varphi_{\ell'',i''} \quad \text{such that} \quad \forall_{\ell=0,\dots,\ell'} \forall_{i\in I_\ell} \ \varphi^{\mathrm{hft}}_{\ell',i'}(x_{\ell,i}) = \delta_{(\ell,i),(\ell',i')}. \tag{4.41}$$

This means that $\varphi^{\mathrm{hft}}_{\ell',i'}$ ($\ell' \in \mathbb{N}_0$, $i' = 0,\dots,2^{\ell'}$) interpolates the data $\{(x_{\ell',i}, \delta_{i,i'}) \mid i = 0,\dots,2^{\ell'}\}$. The coefficients $c^{\ell',i'}_{\ell'',i''} \in \mathbb{R}$ are, in general, different for each basis function $\varphi^{\mathrm{hft}}_{\ell',i'}$. This complicates precomputation and storage of the $2^{\ell'}+1$ coefficients, as they have to be determined by solving a system of linear equations. In addition, the transformation $\varphi_{\ell',i'} \mapsto \varphi^{\mathrm{hft}}_{\ell',i'}$ does not preserve the locality of the support of the basis functions. Consequently, $\varphi^{\mathrm{hft}}_{\ell',i'}$ may be globally supported, which means that we have to evaluate up to $2^{\ell'}+1$ basis functions $\varphi_{\ell'',i''}$ when evaluating $\varphi^{\mathrm{hft}}_{\ell',i'}$ at a single point $x \in [0,1]$. The global support of the resulting transformed basis for uniform hierarchical B-splines (which are locally supported) can be seen in Fig. 4.8.

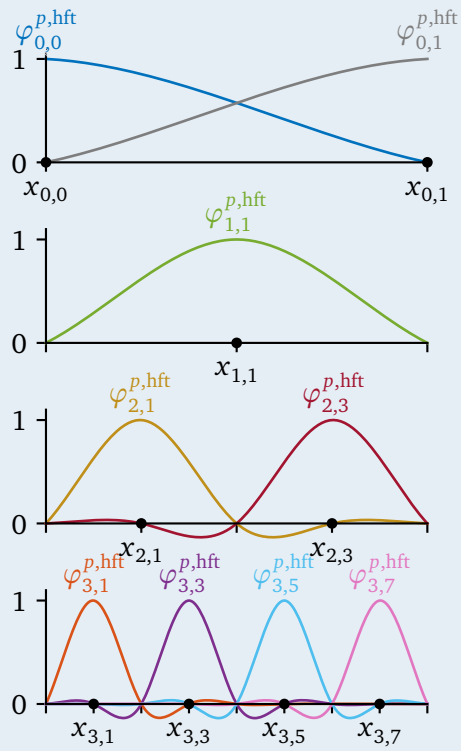


FIGURE 4.8
Resulting basis functions $\varphi^{p,\mathrm{hft}}_{\ell',i'}$ $(\ell' \le \ell,\ i' \in I_{\ell'})$ after applying the hierarchical fundamental transformation to hierarchical cubic B-splines ($p = 3$) and grid points $x_{\ell',i'}$ *(dots)* up to level $\ell = 3$.

We call the transformation $\varphi_{\ell',i'} \mapsto \varphi^{\mathrm{hft}}_{\ell',i'}$ *hierarchical fundamental transformation (HFT)*. The following proposition shows that this is only a change of basis, as the spanned sparse grid space remains unchanged. While the proposition is formulated for regular sparse grids, a similar statement can be proven for the dimensionally adaptive case.

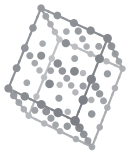
PROPOSITION 4.15 (spanned sparse grid space for the HFT)
If $K := \{(\boldsymbol{\ell}, \boldsymbol{i}) \mid \|\boldsymbol{\ell}'\|_1 \le n,\ \boldsymbol{i}' \in I_{\boldsymbol{\ell}'}\}$ *is the set of level-index pairs for the regular sparse grid of level* n *and dimensionality* d*, then*

$$V^{\mathrm{s}}_{n,d} = V^{\mathrm{s,hft}}_{n,d} := \operatorname{span}\{\varphi^{\mathrm{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'} \mid (\boldsymbol{\ell}', \boldsymbol{i}') \in K\}. \tag{4.42}$$

PROOF We have $V^{\mathrm{s}}_{n,d} \supseteq V^{\mathrm{s,hft}}_{n,d}$ as $\varphi^{\mathrm{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in V^{\mathrm{s}}_{n,d}$ for all $(\boldsymbol{\ell}', \boldsymbol{i}') \in K$:

$$\varphi^{\mathrm{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'} = \prod_{t=1}^{d} \sum_{\ell''_t=0}^{\ell'_t} \sum_{i''_t \in I_{\ell''_t}} c^{\ell'_t,i'_t}_{\ell''_t,i''_t} \varphi_{\ell''_t,i''_t} = \sum_{\boldsymbol{\ell}''=\mathbf{0}}^{\boldsymbol{\ell}'} \sum_{\boldsymbol{i}'' \in I_{\boldsymbol{\ell}''}} c^{\boldsymbol{\ell}',\boldsymbol{i}'}_{\boldsymbol{\ell}'',\boldsymbol{i}''} \varphi_{\boldsymbol{\ell}'',\boldsymbol{i}''} \in V^{\mathrm{s}}_{n,d}, \quad c^{\boldsymbol{\ell}',\boldsymbol{i}'}_{\boldsymbol{\ell}'',\boldsymbol{i}''} := \prod_{t=1}^{d} c^{\ell'_t,i'_t}_{\ell''_t,i''_t}. \tag{4.43}$$

To prove that $V^{\mathrm{s}}_{n,d} \subseteq V^{\mathrm{s,hft}}_{n,d}$, we show that the dimension of $V^{\mathrm{s,hft}}_{n,d}$ matches $\dim V^{\mathrm{s}}_{n,d} = |K|$. It suffices to show that the functions $\varphi^{\mathrm{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ $((\boldsymbol{\ell}', \boldsymbol{i}') \in K)$, are linearly independent. Let



$\alpha_{\ell',i'} \in \mathbb{R}$ be with $\sum_{(\ell',i')\in K} \alpha_{\ell',i'} \varphi^{\text{hft}}_{\ell',i'} \equiv 0$. By evaluating at $\boldsymbol{x}_{\ell,i}$ ($(\boldsymbol{\ell},\boldsymbol{i}) \in K$), we obtain

$$\sum_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\text{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0, \quad (\boldsymbol{\ell},\boldsymbol{i}) \in K. \tag{4.44}$$

This is a lower triangular system according to (4.34), which implies $\alpha_{\boldsymbol{\ell}',\boldsymbol{i}'} = 0$ for all $(\boldsymbol{\ell},\boldsymbol{i}) \in K$. Hence, the functions $\varphi^{\text{hft}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ ($(\boldsymbol{\ell}',\boldsymbol{i}') \in K$) are linearly independent. ∎

Translation-invariant fundamental transformation (TIFT). Another disadvantage of the HFT is that it does not preserve the so-called *translation invariance* of the original basis. A basis $\varphi_{\ell,i}$ ($\ell \in \mathbb{N}_0$, $i = 0,\dots,2^\ell$) is translation-invariant, if there is a *parent function* $\varphi : \mathbb{R} \to \mathbb{R}$ such that

$$\varphi_{\ell,i}(x) = \varphi(\tfrac{x}{h_\ell} - i), \quad \ell \in \mathbb{N}_0,\ i = 0,\dots,2^\ell,\ x \in [0,1]. \tag{4.45}$$

The fact that the HFT does not preserve translation invariance means that for each basis function $\varphi^{\text{hft}}_{\ell',i'}$, we have to calculate its individual $2^{\ell'}+1$ coefficients $c^{\ell',i'}_{\ell'',i''}$.

To solve this problem, we use a similar ansatz as for the HFT, but we replace the hierarchical basis functions $\varphi_{\ell'',i''}$ ($\ell'' = 0,\dots,\ell'$, $i'' \in I_{\ell''}$) with nodal basis functions $\varphi_{\ell',i''}$ and allow general integer indices $i'' \in \mathbb{Z}$:

$$\varphi^{\text{tift}}_{\ell',i'} := \sum_{i''\in\mathbb{Z}} c^{\ell',i'}_{i''} \varphi_{\ell',i''} \quad \text{such that} \quad \forall_{i\in I_{\ell'}}\ \varphi^{\text{tift}}_{\ell',i'}(x_{\ell',i}) = \delta_{i,i'}, \tag{4.46}$$

where $\ell' \in \mathbb{N}_0$, $i' = 0,\dots,2^{\ell'}$, and $c^{\ell',i'}_{i''} \in \mathbb{R}$. We have to make three assumptions for (4.46) to make sense:

- The functions $\varphi_{\ell',i''}$ have to be defined for integer indices $i'' \in \mathbb{Z}$, i.e., the functions $\varphi_{\ell',i''} : [0,1] \to \mathbb{R}$ must also exist for $i'' < 0$ or $i'' > 2^{\ell'}$. This is the case for translation-invariant bases $\varphi_{\ell',i''}$ (such as B-splines $\varphi^p_{\ell',i''}$), as they can be generalized to $i'' \in \mathbb{Z}$ via Eq. (4.45).
- The set

  $$J_{\ell'} := \left\{ i'' \in \mathbb{Z} \;\middle|\; \varphi_{\ell',i''}|_{[0,1]} \not\equiv 0 \right\}, \quad \ell' \in \mathbb{N}_0, \tag{4.47}$$

  of relevant indices should be finite, so that in each point $x \in [0,1]$ only a finite number of basis functions $\varphi_{\ell',i''}$ of level ℓ' is non-zero. This means that the series in (4.46) collapses to a finite sum over $i'' \in J_{\ell'}$. The condition is met for compactly supported and translation-invariant basis functions such as B-splines $\varphi^p_{\ell',i''}$. For $d \in \mathbb{N}$ dimensions and $\boldsymbol{\ell}' \in \mathbb{N}_0^d$, we define $J_{\boldsymbol{\ell}'} := J_{\ell'_1} \times \cdots \times J_{\ell'_d}$.

- The coefficients $c_{i''}^{\ell',i'}$, such that (4.46) holds, exist and are uniquely determined.

Let $\varphi_{\ell',i''}$ be translation-invariant and let $\ell' \in \mathbb{N}_0$ and $i' = 0, \dots, 2^{\ell'}$ be arbitrary. Then we have

$$\varphi_{\ell',i'}^{\text{tift}}(x) = \sum_{i'' \in \mathbb{Z}} c_{i''}^{\ell',i'} \varphi_{\ell',i''}(x) \overset{(4.45)}{=} \sum_{i'' \in \mathbb{Z}} c_{i''}^{\ell',i'} \varphi(\tfrac{x}{h_{\ell'}} - i'') = \sum_{i'' \in \mathbb{Z}} c_{i'+i''}^{\ell',i'} \varphi((\tfrac{x}{h_{\ell'}} - i') - i''), \tag{4.48}$$

where the function defined by $\sum_{i'' \in \mathbb{Z}} c_{i'+i''}^{\ell',i'} \varphi(\cdot - i'')$ satisfies

$$\sum_{i'' \in \mathbb{Z}} c_{i'+i''}^{\ell',i'} \varphi(i - i'') = \sum_{i'' \in \mathbb{Z}} c_{i'+i''}^{\ell',i'} \varphi((\tfrac{x_{\ell',i+i'}}{h_{\ell'}} - i') - i'') \overset{(4.48)}{=} \varphi_{\ell',i'}^{\text{tift}}(x_{\ell',i+i'}) \overset{(4.46)}{=} \delta_{i',i+i'} \tag{4.49a}$$

$$= \delta_{i,0}, \quad i \in \mathbb{Z}. \tag{4.49b}$$

Due to the translation invariance and the third assumption in the list above, there is only a single set of coefficients of $\varphi(\cdot - i'')$ such that (4.49) holds. This means that $c_{i'+i''}^{\ell',i'}$ does not depend on ℓ' or i', if $\varphi_{\ell',i'}$ is translation-invariant. Consequently, if we set $c_{i''} := c_{i'+i''}^{\ell',i'}$ for some arbitrary $\ell' \in \mathbb{N}_0$ and $i' = 0, \dots, 2^{\ell'}$, then $\varphi^{\text{tift}} \colon \mathbb{R} \to \mathbb{R}$ defined by

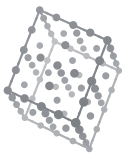
$$\varphi^{\text{tift}}(x) := \sum_{i'' \in \mathbb{Z}} c_{i''} \varphi(x - i''), \quad x \in [0,1], \tag{4.50}$$

is a parent function of $\varphi_{\ell',i'}^{\text{tift}}$ that satisfies

$$\forall_{i \in \mathbb{Z}} \ \varphi^{\text{tift}}(i) = \delta_{i,0}. \tag{4.51}$$

The fact that φ^{tift} is the parent function of $\varphi_{\ell',i'}^{\text{tift}}$ easily follows from (4.48) as the right-hand side (RHS) is exactly $\varphi^{\text{tift}}(\frac{x}{h_{\ell'}} - i')$, as required by (4.45). This shows that the transformation $\varphi_{\ell',i'} \mapsto \varphi_{\ell',i'}^{\text{tift}}$ preserves translation-invariance. Therefore, we call the transformation $\varphi_{\ell',i'} \mapsto \varphi_{\ell',i'}^{\text{tift}}$ *translation-invariant fundamental transformation (TIFT)*.

In contrast to the HFT, the TIFT is only a change of basis if we consider the extended nodal spaces that also include basis functions with indices outside of $\{0, \dots, 2^{\ell'}\}$. This is the statement of the following proposition (generalized to the d-variate case). Note that although the proposition involves basis functions $\varphi_{\ell',i'}$ and $\varphi_{\ell',i'}^{\text{tift}}$ outside the domain $[0,1]$ (in the sense that $\boldsymbol{x}_{\ell',i'} \notin [0,1]$), we still restrict all functions to $[0,1]$. We cannot formulate an equivalent version of Prop. 4.15 (spanned sparse grid space for the HFT), as it might be that, in one dimension, $\varphi_{\ell',i''}$ ($i'' < 0$ or $i'' > 2^{\ell'}$) is not contained in the nodal space $V_{\ell'}$.



PROPOSITION 4.16 (spanned nodal space for the TIFT)
We have

$$\text{(4.52)} \quad \operatorname{span}\{\varphi_{\ell',\boldsymbol{i}'} \mid \boldsymbol{i}' \in J_{\ell'}\} =: V_{\ell'}^{\text{ext}} = V_{\ell'}^{\text{tift,ext}} := \operatorname{span}\{\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}} \mid \boldsymbol{i}' \in J_{\ell'}\}, \quad \boldsymbol{\ell}' \in \mathbb{N}_0^d.$$

PROOF We have $V_{\ell'}^{\text{ext}} \supseteq V_{\ell'}^{\text{tift,ext}}$ as $\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}} \in V_{\ell'}^{\text{ext}}$ for all $\boldsymbol{i}' \in J_{\ell'}$:

$$\text{(4.53)} \quad \varphi_{\ell',\boldsymbol{i}'}^{\text{tift}} = \prod_{t=1}^{d} \sum_{i_t'' \in J_{\ell_t'}} c_{i_t''}^{\ell_t',i_t'} \varphi_{\ell_t',i_t''} = \sum_{\boldsymbol{i}'' \in J_{\ell'}} c_{\boldsymbol{i}''}^{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi_{\boldsymbol{\ell}',\boldsymbol{i}''} \in V_{\ell'}^{\text{ext}}, \quad c_{\boldsymbol{i}''}^{\boldsymbol{\ell}',\boldsymbol{i}'} := \prod_{t=1}^{d} c_{i_t''}^{\ell_t',i_t'}.$$

To prove that $V_{\ell'}^{\text{ext}} \subseteq V_{\ell'}^{\text{tift,ext}}$, we show that the dimensions of the two spaces match. As before, it suffices to show that the functions $\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}}$ $(\boldsymbol{i}' \in J_{\ell'})$ are linearly independent. Let $\alpha_{\ell',\boldsymbol{i}'} \in \mathbb{R}$ be with $\sum_{\boldsymbol{i}' \in J_{\ell'}} \alpha_{\ell',\boldsymbol{i}'} \varphi_{\ell',\boldsymbol{i}'}^{\text{tift}} \equiv 0$. By evaluating at $\boldsymbol{x}_{\ell',\boldsymbol{i}}$ $(\boldsymbol{i} \in J_{\ell'})$, we obtain

$$\text{(4.54)} \quad 0 = \sum_{\boldsymbol{i}' \in J_{\ell'}} \alpha_{\ell',\boldsymbol{i}'} \underbrace{\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}}(\boldsymbol{x}_{\ell',\boldsymbol{i}})}_{=\delta_{\boldsymbol{i},\boldsymbol{i}'}} = \alpha_{\ell',\boldsymbol{i}}, \quad \boldsymbol{i} \in J_{\ell'},$$

i.e., all coefficients $\alpha_{\ell',\boldsymbol{i}'}$ must vanish.[6] Hence, the functions $\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}}$ $(\boldsymbol{i}' \in J_{\ell'})$ are linearly independent. ∎

4.4.3 Hierarchical Fundamental Splines

Definition. We now apply the translation-invariant fundamental transformation to hierarchical B-splines $\varphi_{\ell,i}^p$ of degree p. The parent function $\varphi^p : \mathbb{R} \to \mathbb{R}$ of B-splines and the set J_ℓ^p of relevant indices for level $\ell \in \mathbb{N}_0$ are given by

$$\text{(4.55)} \quad \varphi^p(x) := b^p(x + \tfrac{p+1}{2}), \qquad J_\ell^p := \{-\tfrac{p-1}{2}, -\tfrac{p-1}{2}+1, \ldots, 2^\ell + \tfrac{p-1}{2}\},$$

respectively. According to (4.51), the coefficients $c_{k,p} \in \mathbb{R}$ of the transformed parent function $\varphi^{p,\text{tift}}$ in Eq. (4.50) are determined by a bi-infinite-dimensional system of linear

[6] Note that we have to allow evaluations outside the domain $[0,1]$ for this step. However, this is feasible for proving the linear independence of $\varphi_{\ell',\boldsymbol{i}'}^{\text{tift}}$, since we can just restrict the functions after showing that the extended nodal spaces equal.

equations:

$$(4.56)\qquad \begin{pmatrix} \ddots & \ddots & \ddots & & \\ \ddots & b^p(\frac{p+1}{2}) & b^p(\frac{p+1}{2}-1) & b^p(\frac{p+1}{2}-2) & \\ \ddots & b^p(\frac{p+1}{2}+1) & b^p(\frac{p+1}{2}) & b^p(\frac{p+1}{2}-1) & \ddots \\ & b^p(\frac{p+1}{2}+2) & b^p(\frac{p+1}{2}+1) & b^p(\frac{p+1}{2}) & \ddots \\ & & \ddots & \ddots & \ddots \end{pmatrix} \cdot \begin{pmatrix} \vdots \\ c_{-1,p} \\ c_{0,p} \\ c_{1,p} \\ \vdots \end{pmatrix} = \begin{pmatrix} \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}.$$

As in each row only p entries are non-zero, the system matrix is a symmetric banded Toeplitz matrix[7]. One can show that the linear system (4.56) is uniquely solvable:

THEOREM 4.17 (unique existence of fundamental spline coefficients)
The system (4.56) *has a unique solution* $(c_{k,p})_{k\in\mathbb{Z}}$ *and the corresponding parent function* $\varphi^{p,\mathrm{fs}}\colon \mathbb{R}\to\mathbb{R}$ *defined by* $\varphi^{p,\mathrm{fs}}(x) := \sum_{k\in\mathbb{Z}} c_{k,p}\varphi^p(x-k)$ *satisfies*

$$(4.57)\qquad \exists_{\beta_p,\gamma_p\in\mathbb{R}_{>0}} \forall_{x\in\mathbb{R}}\ |\varphi^{p,\mathrm{fs}}(x)| \le \beta_p\cdot(\gamma_p)^{-|x|}.$$

PROOF See Theorems 1 and 2 in [Schoenb72]. ∎

The function $\varphi^{p,\mathrm{fs}}$ from Thm. 4.17 is well-known as the *fundamental spline* of degree p [Schoenb72; Schoenb73]. Applications of fundamental splines are interpolation and the definition of spline wavelets [Chu92]. The fundamental splines $\varphi^{p,\mathrm{fs}}$ of low degrees p and their bounding functions $\beta_p\cdot(\gamma_p)^{-|x|}$ are plotted in Fig. 4.9.

Definition of hierarchical fundamental splines. The fundamental spline $\varphi^{p,\mathrm{fs}}$ defines hierarchical fundamental spline functions $\varphi^{p,\mathrm{fs}}_{\ell,i}\colon[0,1]\to\mathbb{R}$ via Eq. (4.45), i.e.,

$$(4.58)\qquad \varphi^{p,\mathrm{fs}}_{\ell,i}(x) := \varphi^{p,\mathrm{fs}}(\tfrac{x}{h_\ell}-i),\quad \ell\in\mathbb{N}_0,\ i=0,\dots,2^\ell,\ x\in[0,1].$$

The hierarchical cubic fundamental spline basis is depicted in Fig. 4.10. As usual, we define d-variate hierarchical fundamental splines as tensor products of their univariate counterparts. According to Prop. 3.2 (spline space) and Prop. 4.16 (spanned nodal space for the TIFT), the common extended nodal space $V_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{ext}} = V_{\boldsymbol{\ell}}^{\boldsymbol{p},\mathrm{fs},\mathrm{ext}}$ is equal to the spline space $S_{\boldsymbol{\ell}}^{\boldsymbol{p},[0,1]}$ defined by the Cartesian product of knot sequences of the form (3.22), i.e., the space of all splines of degree $\boldsymbol{p}$ on the full grid of level $\boldsymbol{\ell}$.

The B-spline coefficients $(c_{k,p})_{k\in\mathbb{Z}}$ of the fundamental spline $\varphi^{p,\mathrm{fs}}$ decay with the same

[7] The entries $a_{k,j}$ of a Toeplitz matrix $\boldsymbol{A}$ solely depend on $k-j$, i.e., $a_{k,j}=c_{k-j}$ for some vector $\boldsymbol{c}$.

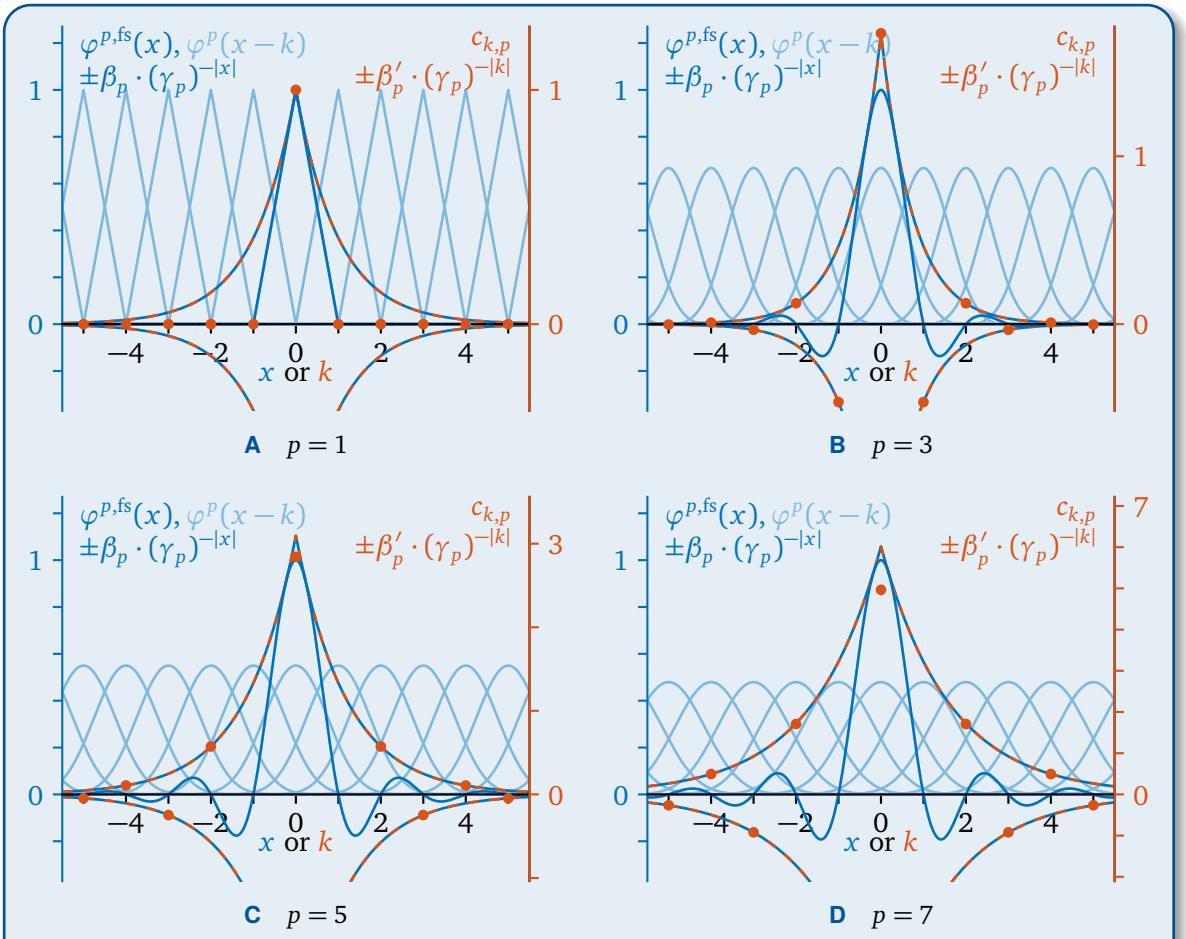


FIGURE 4.9 The fundamental spline $\varphi^{p,\mathrm{fs}}$ *(blue)* is a linear combination of cardinal B-splines $b^p(\cdot - k)$, $k \in \mathbb{Z}$ *(light blue)*, with coefficients $c_{k,p}$ *(red points)*. The absolute values of the fundamental spline $\varphi^{p,\mathrm{fs}}$ and its coefficients are bounded by a multiple of $(\gamma_p)^{-|k|}$ *(blue-red-dashed line)*. The axis for $c_{k,p}$ (on the right side) is scaled such that both bounding functions are on top of each other.

rate as the fundamental spline itself due to the stability of the B-spline basis [Höl13], i.e.,

$$|c_{k,p}| \le \beta'_p \cdot (\gamma_p)^{-|k|}, \quad k \in \mathbb{Z}, \tag{4.59}$$

for some $\beta'_p > 0$ independent of k, which is also shown in Fig. 4.9. For $p > 1$, there is a surprising relationship between the optimal (i.e., largest) decay rate γ_p and the polynomial $\sum_{k=1}^{p} b^p(k)x^{k-1}$, whose coefficients are the values of the cardinal B-spline b^p at its inner knots: The decay rate is given by the absolute value of the largest root smaller than -1 of said polynomial (see [Chu92; Schoenb73]).

Due to (4.59), we may solve the system (4.56) of linear equations approximatively, if we symmetrically truncate the linear system to $2n_p - 1$ rows and columns and set $c_{k,p} := 0$

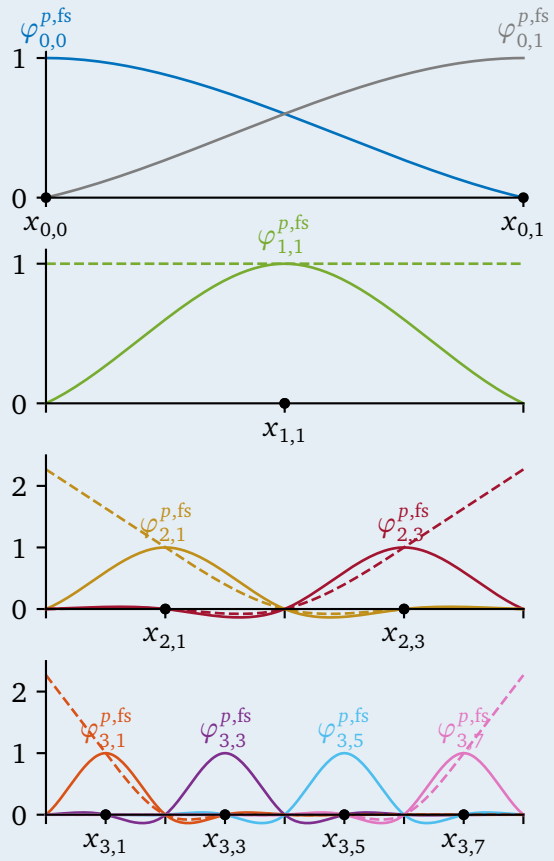


FIGURE 4.10
Hierarchical cubic fundamental splines $\varphi^{p,\mathrm{fs}}_{\ell',i'}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$, $p = 3$), their modified versions $\varphi^{p,\mathrm{fs,mod}}_{\ell',i'}$ *(dashed)*, and grid points $x_{\ell',i'}$ *(dots)* up to level $\ell = 3$.

for all $|k| \geq n_p$, where $n_p \in \mathbb{N}$ is a truncation index. Note that we only have to perform $p+1$ cardinal B-spline evaluations to evaluate $\varphi^{p,\mathrm{fs}}$ once. In Tab. 4.1, we list the decay rates γ_p, the factors β_p and β'_p, and the truncation indices n_p for different p.

4.4.4 Modified Hierarchical Fundamental Splines

Similar to the B-spline bases introduced in Chap. 3, it is possible to define a modified version of the hierarchical fundamental spline basis to obtain reasonable boundary values when working with sparse grids without boundary points. The definition of the modified fundamental spline $\varphi^{p,\mathrm{fs,mod}}_{\ell,i} : [0,1] \to \mathbb{R}$ of level $\ell \in \mathbb{N}$, index $i \in I_\ell$, and degree p is defined as follows:

$$\varphi^{p,\mathrm{fs,mod}}_{\ell,i}(x) := \begin{cases} 1, & \ell = 1, \quad i = 1, \\ \varphi^{p,\mathrm{fs,mod}}\left(\frac{x}{h_\ell}\right), & \ell \geq 2, \quad i = 1, \\ \varphi^{p,\mathrm{fs}}_{\ell,i}(x), & \ell \geq 2, \quad i \in I_\ell \setminus \{1, 2^\ell - 1\}, \\ \varphi^{p,\mathrm{fs,mod}}_{\ell,1}(1-x), & \ell \geq 2, \quad i = 2^\ell - 1, \end{cases} \tag{4.60}$$

p	1	3	5	7	9	11	13	15
γ_p	2.718	3.732	2.322	1.868	1.645	1.512	1.425	1.363
β_p	1	1.241	1.104	1.058	1.037	1.026	1.019	1.014
β'_p	1	1.732	3.095	6.016	12.27	25.82	55.56	121.6
n_p	1	18	29	40	52	64	77	90

TABLE 4.1 Optimal decay rates γ_p and corresponding factors β_p and β'_p for the bound functions of the fundamental spline $\varphi^{p,\mathrm{fs}}$ and its coefficients $c_{k,p}$, i.e., $\forall_{x\in\mathbb{R}}\ |\varphi^{p,\mathrm{fs}}(x)| \le \beta_p(\gamma_p)^{-|x|}$ and $\forall_{k\in\mathbb{Z}}\ |c_{k,p}| \le \beta'_p(\gamma_p)^{-|k|}$ (approximated values). The truncation indices n_p are the smallest numbers such that $\forall_{|k|\ge n_p}\ |c_{k,p}| < 10^{-10}$.

where $\varphi^{p,\mathrm{fs,mod}}$ is a linear combination

$$\varphi^{p,\mathrm{fs,mod}}\colon \mathbb{R}_{\ge 0} \to \mathbb{R}, \quad \varphi^{p,\mathrm{fs,mod}}(x) := \sum_{k=1-(p+1)/2}^{\infty} c_{k,p}^{\mathrm{mod}} \varphi^p(x-k), \tag{4.61}$$

whose coefficients $c_{k,p}^{\mathrm{mod}} \in \mathbb{R}$ are chosen such that

$$\varphi^{p,\mathrm{fs,mod}}(i) = \delta_{i,1}, \quad i \in \mathbb{N}, \tag{4.62a}$$

$$\tfrac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi^{p,\mathrm{fs,mod}}(1) = 0, \tag{4.62b}$$

$$\tfrac{\mathrm{d}^q}{\mathrm{d}x^q}\varphi^{p,\mathrm{fs,mod}}(0) = 0, \qquad q = 2,3,\ldots,\tfrac{p+1}{2}, \tag{4.62c}$$

if $p > 1$. For $p = 1$, we define $\varphi^{p,\mathrm{fs,mod}} := \varphi_{2,1}^{p,\mathrm{mod}}(\frac{\cdot}{4})$. Since the modification coefficients $c_{k,p}^{\mathrm{mod}}$ experience the same decay as the coefficients $c_{k,p}$ of the fundamental spline, we can also approximate $c_{k,p}^{\mathrm{mod}}$ by solving a truncated system of linear equations. The resulting function $\varphi^{p,\mathrm{fs,mod}}$ is shown in Fig. 4.11. The corresponding hierarchical basis $\varphi_{\ell,i}^{p,\mathrm{fs,mod}}$ is included in Fig. 4.10 (dashed lines).

The conditions stated in (4.62) are motivated by the case $p = 3$ of cubic fundamental splines. The first relevant cardinal B-spline is the one with index $k = 1 - \frac{p+1}{2}$ ($k = -1$ in the cubic case), as the B-splines with indices $\le -\frac{p+1}{2}$ vanish on $\mathbb{R}_{\ge 0}$. The modified function $\varphi^{p,\mathrm{fs,mod}}$ should satisfy the fundamental property (4.46) at all positive integer points $k \in \mathbb{N}$. In contrast to the standard fundamental spline $\varphi^{p,\mathrm{fs}}$, we do not enforce the fundamental property at $k = 0$, as our aim is to obtain non-zero boundary values. This leaves us exactly two degrees of freedom in the cubic case, namely $k = -1$ and $k = 0$. We use these to let $\varphi^{p,\mathrm{fs,mod}}$ extrapolate linearly on $[0,1]$, as we did for uniform hierarchical B-splines (see Sec. 3.1.3). Therefore, in the cubic case, we set the second

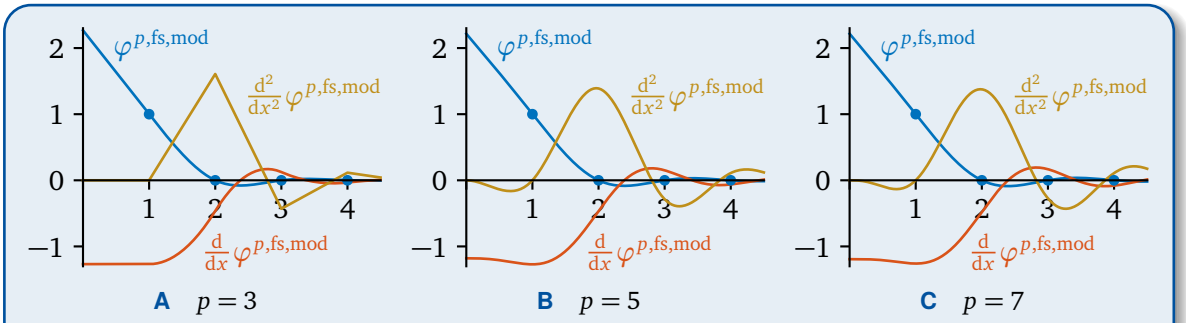


FIGURE 4.11 Modified fundamental spline $\varphi^{p,\mathrm{fs,mod}}$ *(blue)* together with its first *(red)* and second *(brown)* derivatives and the function value interpolation conditions from Eq. (4.62) *(blue dots)*. For $p = 3$, the second derivative vanishes on $[0,1]$. For higher degrees $p > 3$, the second derivative is close to zero on this interval, vanishing at $x = 0$.

derivative $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi^{p,\mathrm{fs,mod}}$ to zero at $x = 0$ and at $x = 1$. This suffices since $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi^{p,\mathrm{fs,mod}}$ is piecewise linear for $p = 3$. For higher degrees $p > 3$, we use the additional degrees of freedom (in total $\frac{p+1}{2}$) to increase the multiplicity of the root of $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi^{p,\mathrm{fs,mod}}$ at $x = 0$. This ensures that $\varphi^{p,\mathrm{fs,mod}}$ is "as linear as possible" near $x = 0$. Note that we cannot maintain $\frac{\mathrm{d}^2}{\mathrm{d}x^2}\varphi^{p,\mathrm{fs,mod}} \equiv 0$ on $[0,1]$ for higher degrees $p > 3$, since this would require $p-1$ conditions and we only have $\frac{p+1}{2}$ degrees of freedom left, after taking the fundamental conditions into account.

4.4.5 Fundamental Not-A-Knot Splines

The hierarchical fundamental spline basis suffers from the same problem as the uniform hierarchical B-spline basis. As explained in Sec. 3.2.1, there is a mismatch of dimensions of the nodal B-spline space V_ℓ^p of level ℓ when compared with the spline space $S_\ell^{p,[0,1]}$ on the grid $\{x_{\ell,i} \mid i = 0,\dots,2^\ell\}$ of level ℓ. This issue also affects the fundamental spline basis.

Definition of fundamental not-a-knot splines. It is possible to combine the idea of fundamental splines with the not-a-knot approach from Sec. 3.2. We define hierarchical fundamental not-a-knot splines $\varphi_{\ell',i'}^{p,\mathrm{fs,nak}} : [0,1] \to \mathbb{R}$ as linear combinations of nodal not-a-knot B-splines of the same level, where the coefficients are chosen such that the fundamental property (4.31) is satisfied:

$$\varphi_{\ell',i'}^{p,\mathrm{fs,nak}} := \sum_{i''=0}^{2^{\ell'}} c_{i'',p}^{\ell',i',\mathrm{fs}} \varphi_{\ell',i''}^{p,\mathrm{nak}} \quad \text{such that} \quad \forall_{i=0,\dots,2^{\ell'}} \ \varphi_{\ell',i'}^{p,\mathrm{fs,nak}}(x_{\ell',i}) = \delta_{i,i'}, \tag{4.63}$$

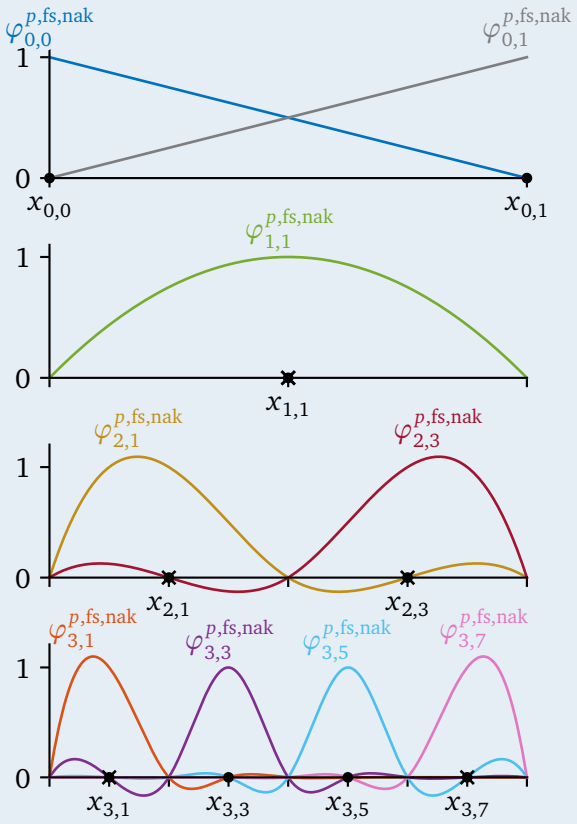


FIGURE 4.12
Hierarchical cubic fundamental not-a-knot splines $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$, $p = 3$), grid points $x_{\ell',i'}$ *(dots)*, and removed knots *(crosses)* up to level $\ell = 3$.

where $\ell' \in \mathbb{N}_0$, $i' = 0, \ldots, 2^{\ell'}$, and $c^{\ell',i',\mathrm{fs}}_{i'',p} \in \mathbb{R}$. This approach is similar to the HFT in Sec. 4.4.2, see Eq. (4.41). We show the hierarchical fundamental not-a-knot spline basis of cubic degree in Fig. 4.12.

The fundamental not-a-knot splines $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ of level $\ell' < \lceil \log_2(p+1) \rceil$ equal the Lagrange polynomials $L_{\ell',i'}$ ($i' = 0, \ldots, 2^{\ell'}$), This is because the i'-th summand $\varphi^{p,\mathrm{nak}}_{\ell',i'}$ of (4.63) equals $L_{\ell',i'}$ and as $L_{\ell',i'}$ already fulfills the fundamental interpolation conditions given in (4.63) (see Eq. (3.28)), we obtain $c^{\ell',i',\mathrm{fs}}_{i'',p} = \delta_{i',i''}$, i.e., $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'} = L_{\ell',i'}$.

Implementation. We make two remarks with respect to the efficient implementation of hierarchical fundamental not-a-knot splines. First, Eq. (4.63) requires the solution of a system of linear equations with dimension $2^{\ell'} + 1$, which grows exponentially in the level ℓ'. However, as the coefficients decay roughly in the same order as the fundamental spline coefficients $c_{k,p}$ in Eq. (4.59), we can solve a truncated system of linear equations instead.

Second, the fundamental not-a-knot spline basis $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ is not translation-invariant anymore. This means that theoretically, we have to compute the $c^{\ell',i',\mathrm{fs}}_{i'',p}$ individually for each basis function $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$. Nevertheless, when truncating the linear system for a fixed

level ℓ', almost all the inner basis functions $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ will be identical to hierarchical fundamental splines $\varphi^{p,\mathrm{fs}}_{\ell',i'}$, if the distance of the region with the removed knots to the grid point $x_{\ell',i'}$ is large enough (if the removed knots are outside of the truncated support of $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$). For different levels ℓ', the fundamental not-a-knot splines $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ are the same up to scaling (if the level ℓ' is high enough).

Consequently, an efficient implementation only has to implement $\varphi^{p,\mathrm{fs,nak}}_{\ell',i'}$ for some special cases for coarse levels. The other basis functions can then be derived via an affine parameter transformation.

4.5 Hierarchization on Spatially Adaptive Sparse Grids with the Unidirectional Principle

In this final section of the chapter, we further decrease the computational complexity for the application of the linear operator $\mathfrak{L}$ on spatially adaptive sparse grids from quadratic to linear time with two algorithms based on the UP.

IN THIS SECTION

4.5.1 Iteratively Applying the Unidirectional Principle with Iterative Refinement

The first algorithm can be applied if two requirements are met:

- The inverse $\mathfrak{L}^{-1}$ is known and can be efficiently applied.
- There is an operator $\mathfrak{L}'$ that is "sufficiently close" to $\mathfrak{L}$ and can be efficiently applied.

For hierarchization with B-splines on sparse grids, we choose $\mathfrak{L}$ to be the hierarchization operator given in Eq. (4.5) and $\mathfrak{L}'$ to be the UP directly applied on the sparse grid. Both of the assumptions are then satisfied, as $\mathfrak{L}^{-1}$ is known (interpolation matrix $\boldsymbol{A}$ of basis function evaluations) and $\mathfrak{L}^{-1}$ and $\mathfrak{L}'$ can be applied fast. The UP $\mathfrak{L}'$ generally produces wrong results for hierarchical B-splines due to missing coupling points. However, especially for low B-spline degrees, $\mathfrak{L}'$ does not deviate too much from the true operator $\mathfrak{L}$. Below, we will specify a sufficient criterion for the "closeness."

```
1 function y = iterativeRefinement(u, y^(0))
2   r^(0) ← u − L^{-1} y^(0)                    ⇝ initial residual
3   for m = 0, 1, 2, ... do
4     y^(m+1) ← y^(m) + L' r^(m)                ⇝ update solution
5     r^(m+1) ← r^(m) − L^{-1} L' r^(m)         ⇝ update residual
6   y ← last computed y^(m)
```

ALGORITHM 4.5 Application of a tensor product operator $\mathfrak{L}$ on spatially adaptive sparse grids with iterative refinement, where $\mathfrak{L}'$ is an approximation of $\mathfrak{L}$. Inputs are the vector $\boldsymbol{u} = (u_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of input data (function values $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ at the grid points) and an initial solution $\boldsymbol{y}^{(0)}$. The output is the vector $\boldsymbol{y} = (y_{\boldsymbol{\ell},\boldsymbol{i}})_{(\boldsymbol{\ell},\boldsymbol{i})\in K}$ of output data (hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$).

Iterative refinement. Under the two assumptions above, we can apply the procedure given in Alg. 4.5. The algorithm is equivalent to the well-known method of *iterative refinement,* which has been developed to stabilize the numerical solution of a linear system influenced by rounding errors [Hig02]. The operator $\mathfrak{L}'$ acts like a preconditioner, which is why it is required to be close to $\mathfrak{L}$. Note that the algorithm is similar to the repeated application of the method of residual interpolation (see Sec. 4.3.3) on the whole sparse grid.

The loop in Alg. 4.5 has to be terminated after some iterations. The following simple lemma allows to use a stopping criterion based on the size of the residual $\boldsymbol{r}^{(m)}$ to the true solution, which we denote with $\boldsymbol{y}^* := \mathfrak{L}\boldsymbol{u}$.

LEMMA 4.18 *In Alg. 4.5, we have* $\boldsymbol{y}^{(m)} \to \boldsymbol{y}^* \iff \boldsymbol{r}^{(m)} \to \boldsymbol{0}$ *for* $m \to \infty$.

PROOF It suffices to prove $\mathfrak{L}\boldsymbol{r}^{(m)} = \boldsymbol{y}^* - \boldsymbol{y}^{(m)}$ for $m \in \mathbb{N}$ by induction. For $m = 0$, we have $\mathfrak{L}\boldsymbol{r}^{(0)} = \mathfrak{L}\boldsymbol{u} - \mathfrak{L}\mathfrak{L}^{-1}\boldsymbol{y}^{(0)} = \boldsymbol{y}^* - \boldsymbol{y}^{(0)}$. For $m \to m+1$, it holds $\mathfrak{L}\boldsymbol{r}^{(m+1)} = \mathfrak{L}\boldsymbol{r}^{(m)} - \mathfrak{L}\mathfrak{L}^{-1}\mathfrak{L}'\boldsymbol{r}^{(m)} = (\boldsymbol{y}^* - \boldsymbol{y}^{(m)}) - \mathfrak{L}'\boldsymbol{r}^{(m)} = \boldsymbol{y}^* - \boldsymbol{y}^{(m+1)}$. ∎

Next, we give a sufficient condition for the convergence of Alg. 4.5 to the true solution.

PROPOSITION 4.19 (sufficient condition for the convergence of Alg. 4.5)
If we have $\limsup_{m\to\infty} \sqrt[m]{\|(\mathrm{id} - \mathfrak{L}^{-1}\mathfrak{L}')^m\|} < 1$ *with an arbitrary operator matrix norm* $\|\cdot\|$ *and the identity operator* id*, then* $\boldsymbol{y}^{(m)} \to \boldsymbol{y}^*$ *for* $m \to \infty$ *in Alg. 4.5 for every initial solution* $\boldsymbol{y}^{(0)}$.

PROOF A short proof by induction shows that

$$\boldsymbol{y}^{(m)} = \boldsymbol{y}^{(0)} + \mathfrak{L}' \sum_{m'=0}^{m-1} (\mathrm{id} - \mathfrak{L}^{-1}\mathfrak{L}')^{m'} \boldsymbol{r}^{(0)}, \tag{4.64}$$

where $(\mathrm{id}-\mathfrak{L}^{-1}\mathfrak{L}')^{m'}\boldsymbol{r}^{(0)} = \boldsymbol{r}^{(m')}$. For $m \to \infty$ and with the assumption on $\|(\mathrm{id}-\mathfrak{L}^{-1}\mathfrak{L}')^m\|$, the sum converges to the Neumann series $\sum_{m'=0}^{\infty}(\mathrm{id}-\mathfrak{L}^{-1}\mathfrak{L}')^{m'} = (\mathrm{id}-(\mathrm{id}-\mathfrak{L}^{-1}\mathfrak{L}'))^{-1} = (\mathfrak{L}')^{-1}\mathfrak{L}$ (see, e.g., [Wer11]). In this case, we infer that the limit of $\boldsymbol{y}^{(m)}$ is given by

$$\boldsymbol{y}^{(0)} + \mathfrak{L}'(\mathfrak{L}')^{-1}\mathfrak{L}\boldsymbol{r}^{(0)} = \boldsymbol{y}^{(0)} + \mathfrak{L}\boldsymbol{u} - \mathfrak{L}\mathfrak{L}^{-1}\boldsymbol{y}^{(0)} = \mathfrak{L}\boldsymbol{u} = \boldsymbol{y}^*, \tag{4.65}$$

as claimed. ■

The sufficient condition given in Prop. 4.19 is quite strong, as it can be shown that $\limsup_{m\to\infty} \sqrt[m]{\|(\mathrm{id}-\mathfrak{L}^{-1}\mathfrak{L}')^m\|} \leq 1$ is necessary for convergence. Unfortunately, in the case of hierarchization with B-splines, numerical experiments show that this condition is only met for low dimensionalities d and low B-spline degrees p. Algorithm 4.5 generally diverges for higher dimensionalities or higher degrees.

4.5.2 Duality of the Unidirectional Principle

To motivate the second algorithm that we present in this section, we study why we cannot directly apply the UP (as introduced in Alg. 4.1) on spatially adaptive sparse grids. As before, we denote with K the level-index set of the spatially adaptive sparse grid (see Sec. 4.1).

The UP, as stated in Alg. 4.1 for full grids, subsequently applies one-dimensional operators $\mathfrak{L}^{(t_j),K_{\text{pole}}} : \mathbb{R}^{|K_{\text{pole}}|} \to \mathbb{R}^{|K_{\text{pole}}|}$ on the poles K_{pole} of the sparse grid at hand, iterating over a permutation $t_1, \ldots, t_d$ of the dimensions $1, \ldots, d$. We recall the pole equivalence relation $\sim_{t_j}$ from Eq. (4.7): Two points $\boldsymbol{k}', \boldsymbol{k}'' \in K$ are $\sim_{t_j}$-equivalent, if $\boldsymbol{k}'$ is contained in the pole through $\boldsymbol{k}''$ with respect to the t_j-th dimension, i.e.,

$$\boldsymbol{k}' \sim_{t_j} \boldsymbol{k}'' \iff \boldsymbol{k}'_{-t_j} = \boldsymbol{k}''_{-t_j}, \quad \boldsymbol{k}', \boldsymbol{k}'' \in K. \tag{4.66}$$

Operators for the unidirectional principle. The combined application of all one-dimensional operators $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ ($K_{\text{pole}} \in K/\sim_{t_j}$) of the j-th iteration of Alg. 4.1 is equivalent to a single application of the following operator $\mathfrak{L}^{(t_j)} : \mathbb{R}^{|K|} \to \mathbb{R}^{|K|}$:

$$(\mathfrak{L}^{(t_j)})_{\boldsymbol{k}'',\boldsymbol{k}'} := \begin{cases} (\mathfrak{L}^{(t_j),K_{\text{pole}}})_{k''_{t_j},k'_{t_j}}, & \exists_{K_{\text{pole}}\in K/\sim_{t_j}} \boldsymbol{k}', \boldsymbol{k}'' \in K_{\text{pole}}, \\ 0, & \boldsymbol{k}' \not\sim_{t_j} \boldsymbol{k}'', \end{cases} \tag{4.67}$$

where $(\mathfrak{L}^{(t_j)})_{\boldsymbol{k}'',\boldsymbol{k}'}$ denotes the entry of row $\boldsymbol{k}''$ and column $\boldsymbol{k}'$ of the matrix corresponding to $\mathfrak{L}^{(t_j)}$ (similar for $(\mathfrak{L}^{(t_j),K_{\text{pole}}})_{k''_{t_j},k'_{t_j}}$). The reason for this equivalence is that the poles K_{pole} are pairwise disjoint equivalence classes. Consequently, every point $\boldsymbol{k}$ is only acted upon

by a single one-dimensional operator $\mathfrak{L}^{(t_j),K_{\text{pole}}}$, namely the one with $K_{\text{pole}} = [\boldsymbol{k}]_{\sim_{t_j}}$. This leads to the block-diagonal structure of $\mathfrak{L}^{(t_j)}$ given in (4.67), if the rows of the matrix of $\mathfrak{L}^{(t_j)}$ are grouped by poles K_{pole} and the columns are arranged accordingly.

Correctness and duality of the unidirectional principle. For the remaining considerations, we assume that the operators $\mathfrak{L}$ and $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ are invertible. In this case, $\mathfrak{L}^{(t_j)}$ is also invertible and $(\mathfrak{L}^{(t_j)})^{-1}$ is given by the block-diagonal matrix composed of the inverses of the blocks $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ of $\mathfrak{L}^{(t_j)}$. This is satisfied by dehierarchization operators $\boldsymbol{A}$ due to the linear independence of the hierarchical basis functions.

We are now able to describe the whole UP of Alg. 4.1 as the operator $\mathfrak{L}^{(t_1,\ldots,t_d)}\colon \mathbb{R}^{|K|} \to \mathbb{R}^{|K|}$ given by

$$\mathfrak{L}^{(t_1,\ldots,t_d)} := \mathfrak{L}^{(t_d)} \ldots \mathfrak{L}^{(t_1)}. \tag{4.68}$$

The right-most operator is $\mathfrak{L}^{(t_1)}$, since it is applied first. We say that the UP is *correct* for $\mathfrak{L}$ and $(t_1,\ldots,t_d)$, if

$$\mathfrak{L}^{(t_1,\ldots,t_d)} \stackrel{?}{=} \mathfrak{L}. \tag{4.69}$$

This relation is not satisfied in general, especially for B-spline hierarchization with the operator $\mathfrak{L} = \boldsymbol{A}^{-1}$. However, for operators like these, whose inverse $\mathfrak{L}^{-1} = \boldsymbol{A}$ can be described and applied much easier, we can make use of the so-called *duality of the UP:*

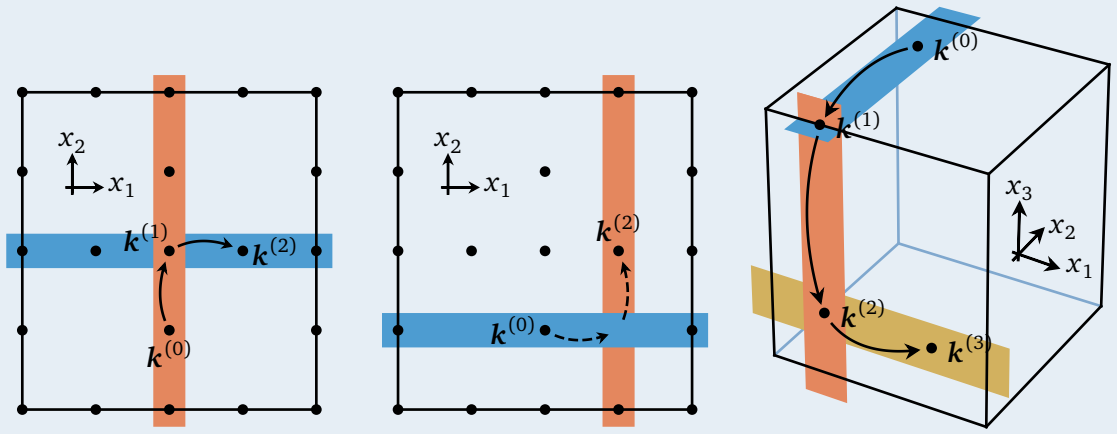
LEMMA 4.20 (duality of the unidirectional principle)
Let the operators $\mathfrak{L}$ and $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ be invertible for all poles K_{pole} in K. Then the UP is correct for $\mathfrak{L}$ and $(t_1,\ldots,t_d)$ if and only if the UP is correct for $\mathfrak{L}^{-1}$ and $(t_d,\ldots,t_1)$.

PROOF The correctness of the UP for $\mathfrak{L}$ and $(t_1,\ldots,t_d)$ is by definition equivalent to

$$\mathfrak{L}^{(t_d)} \ldots \mathfrak{L}^{(t_1)} = \mathfrak{L}. \tag{4.70}$$

By inverting both sides, we obtain the definition of the correctness of the UP for $\mathfrak{L}^{-1}$ and $(t_d,\ldots,t_1)$. ■

This duality means that in order to establish the correctness of $\mathfrak{L}$ for some arbitrary permutation $(t_1,\ldots,t_d)$ of $1,\ldots,d$, it suffices to establish the UP's correctness for the inverse operator $\mathfrak{L}^{-1}$ and the reverse permutation $(t_d,\ldots,t_1)$. This is especially of interest for our main application, the hierarchization operator $\mathfrak{L} = \boldsymbol{A}^{-1}$ for B-splines.



A $d = 2, (t_1, t_2) = (2, 1)$ **B** $d = 2, (t_1, t_2) = (1, 2)$ **C** $d = 3, (t_1, t_2, t_3) = (2, 3, 1)$

FIGURE 4.13 Examples for chains in two and three dimensions. *Left:* A chain from $\boldsymbol{k}^{(0)}$ to $\boldsymbol{k}^{(2)}$ with respect to $(t_1, t_2) = (2, 1)$ in a two-dimensional sparse grid. *Center:* With respect to the reverse permutation $(t_1, t_2) = (1, 2)$ of the dimensions, there is no chain from $\boldsymbol{k}^{(0)}$ to $\boldsymbol{k}^{(2)}$, because the corresponding chain point $\boldsymbol{k}^{(1)}$ is missing in the grid. *Right:* A chain in three dimensions.

4.5.3 Chains and Equivalent Correctness Conditions

We first define the notion of a chain between two grid points $\boldsymbol{k}'$ and $\boldsymbol{k}''$.

DEFINITION 4.21 (chain)
Let $\boldsymbol{k}', \boldsymbol{k}'' \in K$ and $(t_1, \ldots, t_j)$ be a permutation of j of the dimensions $1, \ldots, d$. We define the *chain* from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \ldots, t_j)$ as the sequence $(\boldsymbol{k}^{(0)}, \ldots, \boldsymbol{k}^{(j)})$, where

$$\boldsymbol{k}^{(j')}_{T_{j'}} := \boldsymbol{k}''_{T_{j'}}, \quad \boldsymbol{k}^{(j')}_{-T_{j'}} := \boldsymbol{k}'_{-T_{j'}}, \quad T_{j'} := (t_1, \ldots, t_{j'}), \quad j' = 0, \ldots, j, \tag{4.71}$$

if $\boldsymbol{k}^{(j)} = \boldsymbol{k}''$ and $\boldsymbol{k}^{(j')} \in K$ for all $j' = 0, \ldots, j$.

This definition is equivalent to $\boldsymbol{k}^{(j'-1)} \sim_{t_{j'}} \boldsymbol{k}^{(j')}$ for $j' = 1, \ldots, j$. Figure 4.13 shows examples of chains in two and three dimensions. As it is shown in Fig. 4.13B, the order $(t_1, \ldots, t_j)$ of the dimensions is important for whether the grid contains the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$. The grid must contain all intermediate points, otherwise it is not a chain.

We now show two lemmas. First, we prove that $(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$ is sufficient for the existence of a chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$:

LEMMA 4.22 (sufficient condition for chain existence)
If $(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$ *for some* $j = 0,\ldots,d$, *then the grid* K *contains the chain from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_j)$.

PROOF See Appendix A.3.4. ■

Second, we show that the equality of $(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}^{(j)},\boldsymbol{k}'}$ and the product of the one-dimensional operators is necessary for the existence of a chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$:

LEMMA 4.23 (necessary condition for chain existence)
If the grid K *contains the chain* $(\boldsymbol{k}^{(0)},\ldots,\boldsymbol{k}^{(j)})$ *from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_j)$ *for some* $j = 0,\ldots,d$, *then*

$$(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}^{(j)},\boldsymbol{k}'} = (\mathfrak{L}^{(t_1),[\boldsymbol{k}^{(1)}]_{\sim t_1}})_{k''_{t_1},k'_{t_1}} \cdots (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim t_j}})_{k''_{t_j},k'_{t_j}}. \tag{4.72}$$

PROOF See Appendix A.3.4. ■

These two lemmas can be used to prove the following characterization of the correctness of the UP. Here, we need an additional assumption on the structure of the operator $\mathfrak{L}$, which we call *tensor product structure:*

PROPOSITION 4.24 (characterization of the correctness of the UP)
Let $\mathfrak{L}$ *have tensor product structure: For all* $\boldsymbol{k}',\boldsymbol{k}'' \in K$ *with the chain* $(\boldsymbol{k}^{(0)},\ldots,\boldsymbol{k}^{(d)})$ *from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_d)$, *we assume that*

$$(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} = \prod_{j=1}^{d} (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim t_j}})_{k''_{t_j},k'_{t_j}}. \tag{4.73}$$

Then the UP is correct for $\mathfrak{L}$ *and* $(t_1,\ldots,t_d)$ *if and only if the grid* K *contains the chain from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_d)$ *for all* $\boldsymbol{k}',\boldsymbol{k}'' \in K$ *for which* $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$.

PROOF See Appendix A.3.4. ■

When applied to the hierarchization operator, the combination of Prop. 4.24 with Lemma 4.20 (duality of the unidirectional principle) can be summarized in the following corollary:

COROLLARY 4.25 (equivalent statements for correctness of UP for hierarchization)
The following statements are equivalent:

- *The UP is correct for* $\boldsymbol{A}^{-1}$ *and* $(t_1,\ldots,t_d)$.

- *The UP is correct for* $\boldsymbol{A}$ *and* $(t_d, \ldots, t_1)$.
- *The grid* K *contains the chain from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_d, \ldots, t_1)$ *for all* $\boldsymbol{k}', \boldsymbol{k}'' \in K$ *for which* $\varphi_{\boldsymbol{k}'}(\boldsymbol{x}_{\boldsymbol{k}''}) \neq 0$.

PROOF The corollary is a direct consequence of Lemma 4.20 and Prop. 4.24, applied to the dehierarchization operator $\mathfrak{L} = \boldsymbol{A}$.

The assumption of Lemma 4.20 is satisfied: The operators $\mathfrak{L}^{(t_j),K_{\text{pole}}}$ are invertible for all poles K_{pole} in K due to the uniqueness of univariate interpolants (linear independence of the basis functions). Similarly, $\mathfrak{L}$ is invertible due to the uniqueness of multivariate interpolants. In addition, the assumption of Prop. 4.24 is satisfied, since

$$(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} = (\boldsymbol{A})_{\boldsymbol{k}'',\boldsymbol{k}'} = \prod_{j=1}^{d} \varphi_{k'_{t_j}}(x_{k''_{t_j}}) = \prod_{j=1}^{d} (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim t_j}})_{k''_{t_j},k'_{t_j}} \tag{4.74}$$

due to the tensor product basis functions. ■

Inserting chain points. This means that we can establish the correctness of the UP for the hierarchization operator $\mathfrak{L} = \boldsymbol{A}^{-1}$, if we insert all missing chain points that are specified by Prop. 4.24 into the grid.

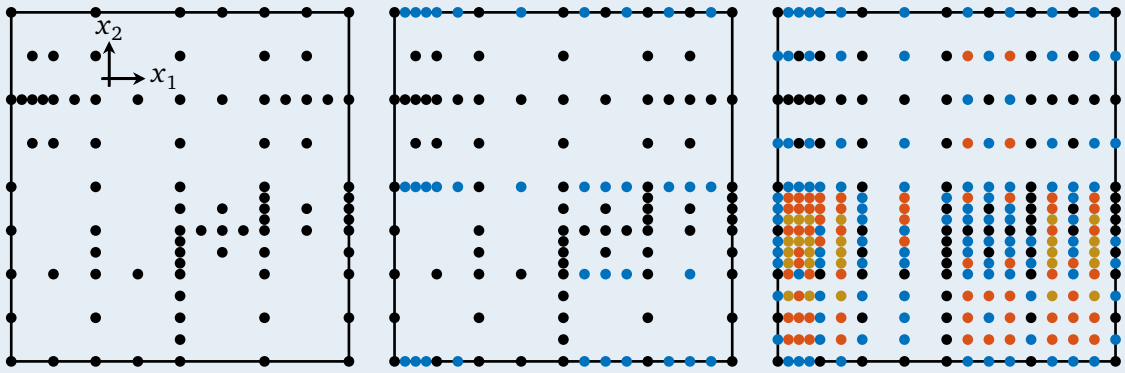
We take the case $p = 1$ of piecewise linear standard B-splines $\varphi^1_{\ell,i}$ as an example. We assume that we iteratively generated a spatially adaptive sparse grid such that all grid points are reachable from the corners of $[\boldsymbol{0}, \boldsymbol{1}]$ in the sense of Eq. (4.38b). If we want to ensure the correctness of the UP for all possible permutations $(t_1, \ldots, t_d)$ of the dimensions $(1, \ldots, d)$, then the existence of the necessary chains in Cor. 4.25 is equivalent to the requirement that the grid should contain the hierarchical ancestors of every grid point in every direction:

$$\text{(4.75a)} \quad \forall_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \forall_{\{t=1,\ldots,d \mid \ell'_t > 1\}} (\boldsymbol{\ell}, \boldsymbol{i}) \in K, \quad \boldsymbol{\ell} := \boldsymbol{\ell}' - \boldsymbol{e}_t, \quad i_t := 2\lfloor \tfrac{i'_t}{4} \rfloor + 1, \quad \boldsymbol{i}_{-t} = \boldsymbol{i}'_{-t},$$

$$\text{(4.75b)} \quad \forall_{(\boldsymbol{\ell}',\boldsymbol{i}')\in K} \forall_{\{t=1,\ldots,d \mid \ell'_t = 1\}} (\boldsymbol{\ell}, \boldsymbol{i}) \in K, \quad \boldsymbol{\ell} := \boldsymbol{\ell}' - \boldsymbol{e}_t, \quad i_t := 0, \qquad\quad \boldsymbol{i}_{-t} = \boldsymbol{i}'_{-t},$$

where $\boldsymbol{e}_t$ is the t-th standard basis vector. This is a standard assumption on spatially adaptive sparse grids with piecewise linear basis functions [Pfl10]. However, we only have to satisfy the conditions of Cor. 4.25 for a single permutation $(t_1, \ldots, t_d)$ of the dimensions in order to hierarchize with the UP. Figure 4.14 shows the necessary ancestor chain points (colored points in Fig. 4.14B) for an example of a two-dimensional spatially adaptive sparse grid (Fig. 4.14A).

Unfortunately, we have to insert these points recursively, e.g., the inserted points may generate new chains, for which other missing points have to be inserted and so on



A Original grid ($N = 85$) **B** Chain points for $p = 1$ ($N = 121$) **C** Chain points for $p = 3$ ($N = 289 = 17 \times 17$)

FIGURE 4.14 Necessary chain points for the correctness of the unidirectional principle with respect to $(t_1, t_2) = (1, 2)$ for hierarchical B-splines $\varphi^p_{\ell,i}$ on a two-dimensional spatially adaptive sparse grid. The colors indicate the recursion depth in which the chain points have been inserted. Black points are contained in the original grid ("zero-order points"). Blue points are part of chains between original grid points ("first-order chain points"). Red points are second-order chain points, i.e., they are part of chains from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ where $\boldsymbol{k}'$ and $\boldsymbol{k}''$ are original grid points or first-order chain points and at least one of them is a first-order chain point. Analogously, brown points are third-order chain points. N is the number of points in the final grid.

("higher-order chain points" in Fig. 4.14). Therefore, the number of points to be inserted may be large. The worst case is that the final grid is a full grid, i.e., the Cartesian product of the union of the poles in the different dimensions:

$$\left(\bigcup_{\boldsymbol{k}\in K} [\boldsymbol{k}]_{\sim_1}\right) \times \cdots \times \left(\bigcup_{\boldsymbol{k}\in K} [\boldsymbol{k}]_{\sim_d}\right), \tag{4.76}$$

i.e., we fully lose the advantage of sparse grids, whose purpose is to ease the curse of dimensionality. For the standard hierarchical B-spline basis $\varphi^p_{\ell,i}$, this worst case often occurs as there are many non-zero entries in the corresponding interpolation matrices $\boldsymbol{A}$ (see Sec. 4.1 and Fig. 4.14C).

4.5.4 Hierarchical Weakly Fundamental Splines

Motivation. In order to reduce the number of chain points to be inserted, we have to use other spline bases such that the resulting interpolation matrices $\boldsymbol{A}$ have more zero entries. The hierarchical fundamental splines as introduced in Sec. 4.4.3 are one possi-

bility. However, they are globally supported, which implies a number of disadvantages concerning the algorithms and the implementations. The most significant disadvantage is that although we can use BFS for the univariate hierarchization operators, the time complexity for the univariate hierarchization is still quadratic. We search for a locally supported spline basis for which the univariate hierarchization can be done in linear time.

To meet these goals, we have to relax the fundamental property to a weaker version, which results in the so-called *weakly fundamental property.* A univariate hierarchical basis $\varphi^{\mathrm{wf}}_{\ell',i'}\colon [0,1]\to\mathbb{R}$ is called *weakly fundamental,* if

$$\varphi^{\mathrm{wf}}_{\ell',i'}(x_{\ell,i})=0,\quad \ell<\ell',\ i\in I_\ell. \tag{4.77}$$

This is exactly the first condition (4.31a) of the fundamental property (4.31). We drop the requirement that the basis functions should vanish at the other grid points of the same level. The relation (4.32) from the fundamental case becomes

$$\varphi^{\mathrm{wf}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})\neq 0 \implies \boldsymbol{\ell}'\leq\boldsymbol{\ell}, \tag{4.78}$$

i.e., every basis function $\varphi^{\mathrm{wf}}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ can only be non-zero at grid points $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ with higher or equal level $\boldsymbol{\ell}$.

Definition of hierarchical weakly fundamental splines. We construct the *weakly fundamental spline parent function* $\varphi^{p,\mathrm{wfs}}\colon\mathbb{R}\to\mathbb{R}$ by forming a linear combination of as few neighboring uniform B-splines as possible such that $\varphi^{p,\mathrm{wfs}}$ satisfies the weakly fundamental property (4.77):

$$\varphi^{p,\mathrm{wfs}}(x) := \sum_{k=-(p-1)/2}^{(p-1)/2} c_{k,p}\varphi^p(x-k) \quad \text{such that} \tag{4.79a}$$

$$c_{0,p}=1,\quad \varphi^{p,\mathrm{wfs}}(k')=0,\ k'=-p+2,\,-p+4,\,\ldots,\,p-2. \tag{4.79b}$$

Hierarchical weakly fundamental splines $\varphi^{p,\mathrm{wfs}}_{\ell,i}\colon[0,1]\to\mathbb{R}$ are now defined canonically via an affine parameter transformation:

$$\varphi^{p,\mathrm{wfs}}_{\ell,i}(x) := \varphi^{p,\mathrm{wfs}}(\tfrac{x}{h_\ell}-i),\quad \ell\geq 1. \tag{4.80}$$

For $\ell=0$, we define $\varphi^{p,\mathrm{wfs}}_{\ell,i}$ to be the linear Lagrange polynomial of level zero:[8]

$$\varphi^{p,\mathrm{wfs}}_{0,i} := L_{0,i},\quad i=0,1. \tag{4.81}$$

[8]This will simplify the description of the Hermite hierarchization algorithm in Sec. 4.5.5.

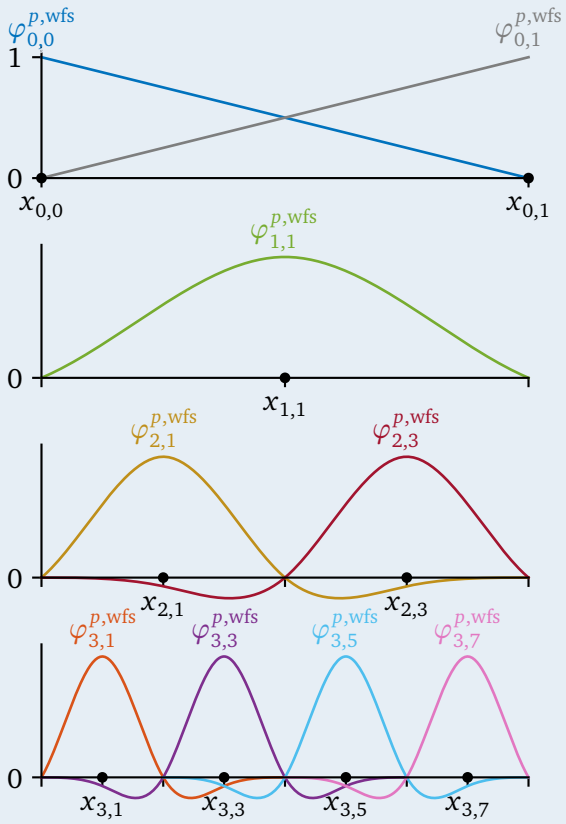


FIGURE 4.15
Hierarchical cubic weakly fundamental splines $\varphi^{p,\mathrm{wfs}}_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$, $p = 3$) and grid points $x_{\ell',i'}$ *(dots)* up to level $\ell = 3$.

The hierarchical weakly fundamental spline basis is shown in Fig. 4.15. Note that these basis functions are translation-invariant by construction (starting with level $\ell \ge 1$). As the weakly fundamental parent spline $\varphi^{p,\mathrm{wfs}}$ vanishes at all odd integers and as the support of $\varphi^{p,\mathrm{wfs}}_{\ell,i}$ is local ($\operatorname{supp} \varphi^{p,\mathrm{wfs}}_{\ell,i} = [x_{\ell,i-p}, x_{\ell,i+p}] \cap [0,1]$), this implies that the weakly fundamental property (4.77) is fulfilled.

Chain points for weakly fundamental splines. The first advantage of the weakly fundamental spline basis $\varphi^{p,\mathrm{wfs}}_{\ell,i}$ over standard uniform B-splines $\varphi^{p}_{\ell,i}$ is that the condition $\varphi_{\boldsymbol{k}'}(\boldsymbol{x}_{\boldsymbol{k}''}) \neq 0$ in Cor. 4.25 is satisfied much more rarely. Consequently, fewer chain grid points have to be inserted to ensure the correctness of the UP for hierarchization. Figure 4.16 shows the inserted points for the same grid as in Fig. 4.14.

In the special case of regular sparse grids $\Omega^{\mathrm{s}}_{n,d}$, we do not have to insert any grid points for the correctness of the UP. We can verify this statement with Cor. 4.25 (equivalent statements for correctness of UP for hierarchization): Let $(\boldsymbol{\ell}', \boldsymbol{i}')$ and $(\boldsymbol{\ell}'', \boldsymbol{i}'')$ with $\|\boldsymbol{\ell}'\|_1, \|\boldsymbol{\ell}''\|_1 \le n$ and $\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}$, $\boldsymbol{i}'' \in I_{\boldsymbol{\ell}''}$, such that $\varphi^{p,\mathrm{wfs}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell}'',\boldsymbol{i}''}) \neq 0$. Furthermore, let $(\boldsymbol{\ell}^{(0)}, \boldsymbol{i}^{(0)}), \ldots, (\boldsymbol{\ell}^{(d)}, \boldsymbol{i}^{(d)})$ be the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $t_1, \ldots, t_d$. Note that $\boldsymbol{\ell}^{(j)} \le \mathbf{max}\{\boldsymbol{\ell}', \boldsymbol{\ell}''\}$ due to the definition of chain points (Def. 4.21). Therefore, we have

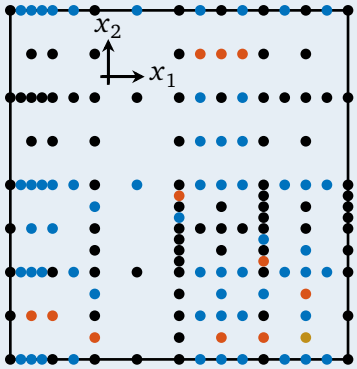


FIGURE 4.16
Necessary chain points for the correctness of the unidirectional principle with respect to $(t_1, t_2) = (1, 2)$ for hierarchical cubic weakly fundamental splines $\varphi_{\ell,i}^{p,\mathrm{wfs}}$ $(p = 3)$ on the same two-dimensional spatially adaptive sparse grid as in Fig. 4.14A. The colors indicate the recursion depth in which the chain points have been inserted (see caption of Fig. 4.14). The number of points in the final grid is $N = 157$.

for $j = 0, \ldots, d$ by (4.78):

$$\boldsymbol{\ell}' \le \boldsymbol{\ell}'' \implies \boldsymbol{\ell}^{(j)} \le \mathbf{max}\{\boldsymbol{\ell}', \boldsymbol{\ell}''\} \le \boldsymbol{\ell}'' \implies \|\boldsymbol{\ell}^{(j)}\|_1 \le \|\boldsymbol{\ell}''\|_1 \le n. \tag{4.82}$$

Hence, $\Omega_{n,d}^{\mathrm{s}}$ contains the grid points corresponding to $(\boldsymbol{\ell}^{(j)}, \boldsymbol{i}^{(j)})$ for all $j = 0, \ldots, d$. Consequently, the conditions of Cor. 4.25 are satisfied without inserting any additional chain points. This statement is even valid for arbitrary dimensionally adaptive sparse grids.

4.5.5 Hermite Hierarchization

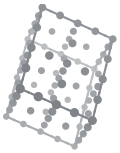
Hermite interpolation. The second advantage of the weakly fundamental spline basis is that due to the reduced coupling, the univariate hierarchization operators can be applied easier than for standard uniform B-splines. This results in the formulation of the so-called *Hermite hierarchization* algorithm. We first recall higher-order Hermite interpolation:

LEMMA 4.26 (higher-order Hermite interpolation)
Let $p \in \mathbb{N}$ be odd and $a, b \in \mathbb{R}$ with $a < b$. Furthermore, let $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f(a) \in \mathbb{R}$ and $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f(b) \in \mathbb{R}$ be given data for $q = 0, \ldots, \frac{p-1}{2}$. Then there is a unique polynomial $s \in P^p$ such that

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q} f(a) = \frac{\mathrm{d}^q}{\mathrm{d}x^q} s(a), \quad \frac{\mathrm{d}^q}{\mathrm{d}x^q} f(b) = \frac{\mathrm{d}^q}{\mathrm{d}x^q} s(b), \quad q = 0, \ldots, \frac{p-1}{2}. \tag{4.83}$$

PROOF See [Fre07]. ■

Hermite hierarchization algorithm. The interpolating polynomial s and its derivatives can be efficiently evaluated using Hermite basis functions (generalized Lagrange polynomials [Fre07]). With Hermite interpolation, we formulate Alg. 4.6 for the hierarchization with hierarchical weakly fundamental splines. While we formulate Alg. 4.6 only for regular univariate grids and weakly fundamental splines, a slightly reformulated version of the



1 **function** $y = \texttt{hermiteHierarchization1D}(u, n)$
2 **for** $i = 0, 1$ **do** ⇝ *set values for level* 0
3 $y_{0,i} \leftarrow f(x_{0,i})$
4 $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_0(x_{0,i}) \leftarrow \delta_{q,0} \cdot f(x_{0,i}) + \delta_{q,1} \cdot (f(x_{0,1}) - f(x_{0,0}))$ for all $q = 0, \ldots, \frac{p-1}{2}$
5 **for** $\ell = 1, \ldots, n$ **do**
6 **for** $i \in I_\ell$ **do**
7 $f_{\ell-1}(x_{\ell,i}) \leftarrow$ Hermite interpolation of $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i\pm1})$ $(q = 0, \ldots, \frac{p-1}{2})$
8 $r^{(\ell)}(x_{\ell,i}) \leftarrow f(x_{\ell,i}) - f_{\ell-1}(x_{\ell,i})$ ⇝ *residual to be interpolated*
9 Let $r_\ell^{(\ell)}$ be of the form $\sum_{i' \in I_\ell} y_{\ell,i'} \varphi_{\ell,i'}^{p,\mathrm{wfs}}$ ⇝ *contribution of level* ℓ
10 Choose $(y_{\ell,i'})_{i' \in I_\ell}$ such that $r_\ell^{(\ell)}(x_{\ell,i}) = r^{(\ell)}(x_{\ell,i})$ for all $i \in I_\ell$
11 **for** $i = 0, \ldots, 2^\ell$ **do** ⇝ *for all points (current level and ancestors)*
12 **for** $q = 0, \ldots, \frac{p-1}{2}$ **do**
13 $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_\ell(x_{\ell,i}) \leftarrow \frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i}) + \frac{\mathrm{d}^q}{\mathrm{d}x^q} r_\ell^{(\ell)}(x_{\ell,i})$ ⇝ *update values*

ALGORITHM 4.6 Hermite hierarchization on one-dimensional regular grids. Inputs are the vector $u = (u_{\ell,i})_{(\ell,i) \in K}$ of input data (function values $f(x_{\ell,i})$ at the grid points) and the level n of the regular grid, where $K = \{(\ell, i) \mid \ell = 0, \ldots, n,\ i \in I_\ell\}$. The output is the vector $y = (y_{\ell,i})_{(\ell,i) \in K}$ of output data (hierarchical surpluses $\alpha_{\ell,i}$).

algorithm also correctly operates on spatially adaptive univariate grids (with the assumption that the grids contain the parents of their grid points) and other weakly fundamental bases that are piecewise polynomials of degree $\leq p$.

The idea of Alg. 4.6, which is also illustrated in Fig. 4.17, is to hierarchize the function value data level by level, which is only possible because of the weakly fundamental property (4.77). For each level ℓ, we calculate surpluses $\alpha_{\ell,i} = y_{\ell,i}$, while keeping track of the values and derivatives $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_\ell(x_{\ell,i})$ of the "current" interpolant f_ℓ (up to level ℓ). Hermite interpolation is used to determine the "delta" to the interpolant of the next level. Note that in line 13, we have to evaluate the derivatives of $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i})$ of the Hermite interpolant determined in line 7. This is not an issue since in an implementation one would typically simultaneously evaluate the Hermite interpolant and its derivatives.

For hierarchical weakly fundamental splines, the complexity of the ℓ-th iteration of Alg. 4.6 is linear in the number of grid points of level ℓ, i.e., $\mathcal{O}(2^\ell)$. The reason for this is the bandedness (with bandwidth $\mathcal{O}(p)$) of the system of linear equations corresponding to the interpolation problem of lines 9 and 10, which means that the interpolation problem can be solved in linear time and memory. In total, the complexity of Alg. 4.6 is given by $\mathcal{O}(\sum_{\ell=0}^n 2^\ell) = \mathcal{O}(2^n)$, i.e., the time and memory required by Alg. 4.6 is only linear in the number of grid points.

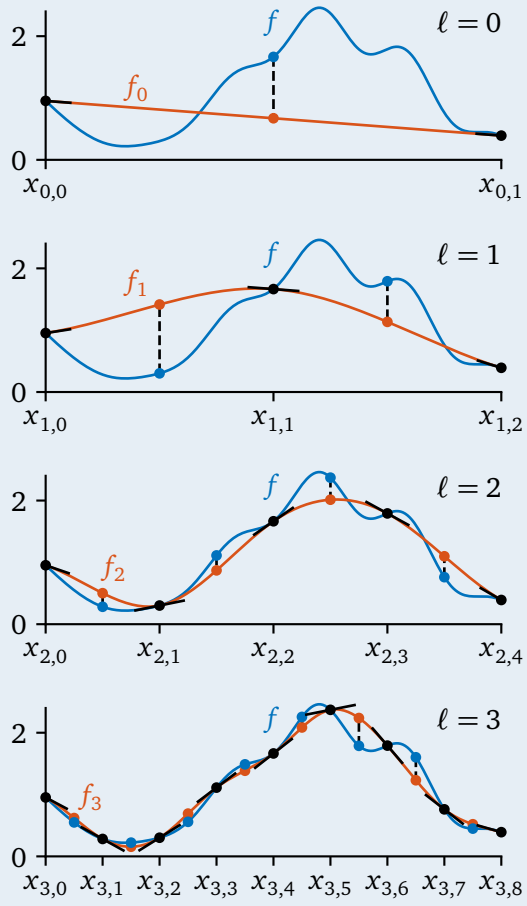


FIGURE 4.17
Hermite hierarchization on a regular grid in one dimension with cubic weakly fundamental splines $\varphi_{\ell,i}^{p,\mathrm{wfs}}$ $(p=3)$. The interpolants f_ℓ *(red)* of the objective function *(blue)* are computed level by level. For each level ℓ, the values $f_\ell(x_{\ell,i})$ and the derivatives $\frac{\mathrm{d}}{\mathrm{d}x} f_\ell(x_{\ell,i})$ of the current interpolant f_ℓ at the grid points $x_{\ell,i}$ $(i=0,\ldots,2^\ell)$ are saved *(black dots and bars)*. The values and derivatives are used for the Hermite interpolation of the residual $f - f_\ell$. The interpolated residual is then added to the current interpolant such that the sum vanishes in the grid points of the next level $\ell+1$ *(black dashed lines between red and blue dots)*. Due to the weakly fundamental property, the previously interpolated values of f remain unchanged.

Correctness. We prove the correctness of Hermite hierarchization with the following invariant.

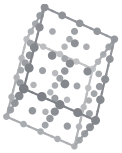
PROPOSITION 4.27 (invariant of Hermite hierarchization)
In Alg. 4.6, it holds for $\ell = 0,\ldots,n$ *and* $i = 0,\ldots,2^\ell$

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_\ell(x_{\ell,i}) = \sum_{\ell'=0}^{\ell} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi_{\ell',i'}^{p,\mathrm{wfs}}(x_{\ell,i}), \quad q = 0,\ldots,\frac{p-1}{2}. \tag{4.84}$$

PROOF See Appendix A.3.5. ∎

COROLLARY 4.28 *Algorithm 4.6 is correct.*

PROOF See Appendix A.3.5. ∎



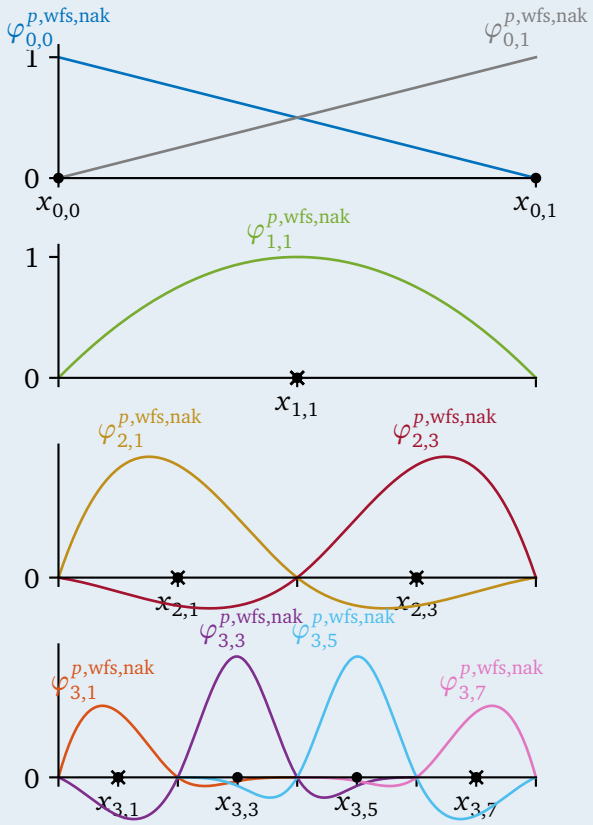


FIGURE 4.18
Hierarchical cubic weakly fundamental not-a-knot splines $\varphi^{p,\mathrm{wfs,nak}}_{\ell',i'}$ ($\ell' \leq \ell$, $i' \in I_{\ell'}$, $p = 3$), grid points $x_{\ell',i'}$ *(dots)*, and removed knots *(crosses)* up to level $\ell = 3$.

4.5.6 Hierarchical Weakly Fundamental Not-A-Knot Splines

Finally, as for fundamental splines, it is possible to combine the weakly fundamental basis with the not-a-knot idea from Sec. 3.2 to construct hierarchical weakly fundamental not-a-knot spline functions $\varphi^{p,\mathrm{wfs,nak}}_{\ell',i'}$. The approach is similar to the fundamental not-a-knot splines in Sec. 4.4.5 (see Eq. (4.63)): Instead of combining uniform B-splines as in (4.79a), we combine not-a-knot B-splines such that the weakly fundamental property is satisfied.

However, the exact construction is somewhat complicated, as one has to carefully consider which conditions to enforce with which basis functions. There are some special cases, if the index of the basis function $\varphi^{p,\mathrm{wfs,nak}}_{\ell',i'}$ is near the boundary (near $i' = 0$ or near $i' = 2^{\ell'}$). Nevertheless, there are only finitely many special cases; for higher levels ℓ', one can just scale the basis functions of coarser levels. In the scope of this thesis, it suffices to show the resulting basis functions for the cubic case ($p = 3$) in Fig. 4.18, instead of rigorously stating the technical formulas.

5

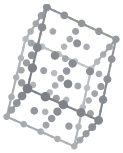
Gradient-Based Optimization with B-Splines on Sparse Grids

> *Premature optimization is the root of all evil.*
>
> — Donald E. Knuth [Knu74]

In this chapter, we apply the hierarchical B-spline bases derived in Chapters 3 and 4 to optimization, which is a major task in simulation technology, for instance in inverse problems (see Chap. 1). We pursue a three-step surrogate-based optimization approach: First, we sample the objective function at specific sparse grid points to retrieve objective function values. Second, by interpolating these values with hierarchical bases, we obtain a surrogate for the objective function. Third and finally, we discard the original objective function and apply already existing optimization methods to the surrogate.

One of the key advantages of hierarchical B-splines over common hierarchical bases for sparse grids is their continuous differentiability. The derivatives of B-spline surrogates on sparse grids are not only continuous, but also explicitly known, and they can be evaluated fast. This gives the opportunity to employ gradient-based optimization methods, which usually converge significantly faster than gradient-free alternatives.

The outline of this chapter is as follows: We start in Sec. 5.1 with a compact overview of textbook optimization algorithms, which comprises gradient-free and gradient-based



optimization algorithms for unconstrained problems as well as algorithms for constrained problems. In Sec. 5.2, we present the main method that conflates the various optimization algorithms and the generation of spatially adaptive sparse grids with the Novak–Ritter criterion to create a single method for the optimization of sparse grid surrogates. Section 5.3 continues with a small array of test problems for unconstrained and constrained optimization. In Sec. 5.4, we apply the presented method to the test problems, studying the influence of the different hierarchical B-splines on optimality gaps and conducting numerical experiments. Finally, in Sec. 5.5, we examine the fuzzy extension principle as an example application of the optimization of hierarchical B-spline surrogates on sparse grids.

Parts of this chapter have already been published in previous work [Vale14], especially the overview of optimization algorithms (Sec. 5.1) and the methodology of the optimization of sparse grid surrogates (Sec. 5.2). However, as the previous work included other basis functions as well, this chapter represents the first comprehensive study that focuses on the application of hierarchical B-splines to optimization.

5.1 Overview of Optimization Algorithms

IN THIS SECTION

Problem setting. Generally, *unconstrained optimization problems* have the form

$$\boldsymbol{x}^{\text{opt}} = \operatorname{\mathbf{arg\,min}} f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^d, \tag{5.1}$$

where $f : \mathbb{R}^d \to \mathbb{R}$ is the *objective function*. *Constrained optimization problems* are given by

$$\boldsymbol{x}^{\text{opt}} = \operatorname{\mathbf{arg\,min}} f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^d \text{ s.t. } \boldsymbol{g}(\boldsymbol{x}) \leq \mathbf{0}, \tag{5.2}$$

where $\boldsymbol{g} : \mathbb{R}^d \to \mathbb{R}^{m_g}$ ($m_g \in \mathbb{N}$) is the *(inequality) constraint function*. This formulation also contains optimization problems with equality constraints $\boldsymbol{h}(\boldsymbol{x}) = \mathbf{0}$ by setting $\boldsymbol{g}(\boldsymbol{x}) := (\boldsymbol{h}(\boldsymbol{x}), -\boldsymbol{h}(\boldsymbol{x}))$. Equality constraints can also be solved by incorporating them into the unconstrained solver (e.g., see [Boy04] for an equality-constrained Newton method).

As sparse grid surrogates $f = f^s$ are only defined on the unit hyper-cube, the choice of $\boldsymbol{x}$ has to be restricted to $[\mathbf{0}, \mathbf{1}]$. In the case of (5.1), this results in a *box-constrained optimization problem*. A simple method for applying unconstrained optimization algorithms to box-constrained problems is extending f^s to $\mathbb{R}^d$ by $f^s(\boldsymbol{x}) := +\infty$ for all $\boldsymbol{x} \in \mathbb{R}^d \setminus [\mathbf{0}, \mathbf{1}]$. However, more sophisticated approaches are also available [Mor87].

Black-box optimization methods. Problems of the form (5.1) or (5.2) are *black-box optimization problems,* where we cannot gain any insight into the structure or algebraic properties of f. Black-box optimization methods perform a series of evaluations $f(\boldsymbol{x}_k)$, choosing the next evaluation point $\boldsymbol{x}_{k+1}$ based on the previous function values $f(\boldsymbol{x}_0), \ldots, f(\boldsymbol{x}_k)$. Gradient-based methods differ from gradient-free approaches in such a way that they also take values of the gradient $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k)$, of the Hessian $\nabla^2_{\boldsymbol{x}} f(\boldsymbol{x}_k)$, or of even higher-order derivatives into account.

A vast range of optimization methods exists in literature. Some methods are better suited for specific optimization problems than others. However, according to the "no-free-lunch theorem" and under some assumptions [Wol97], all methods perform equally well (or equally badly) in the mean of all possible optimization problems.

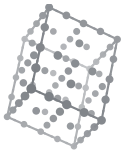
Local and global optima. Most optimization methods depend on an initial point $\boldsymbol{x}_0$ and only find local optima, where (5.1) or (5.2) only holds for $\boldsymbol{x}$ in a neighborhood of $\boldsymbol{x}^{\text{opt}}$. One can globalize local methods to increase the probability of finding a global optimum with a Monte Carlo multi-start approach: The local method is repeated with different pseudo-random initial points and the best local optimum is chosen as the result.

In the following, we give a brief survey of a small selection of optimization methods (see Tab. 5.1, Fig. 5.1, and Fig. 5.2), highlighting the key ingredients for each method.

5.1.1 Gradient-Free Unconstrained Optimization Methods

Nelder–Mead. The *Nelder–Mead method* [Nel65; Gao12; Vale14] maintains a list of $d+1$ vertices of a d-dimensional simplex, sorted by ascending function value. In each iteration, the method performs one of the operations *reflection, expansion, outer contraction, inner contraction,* and *shrinking* on the vertices. Typically, convergence can be detected by checking the size of the simplex, as the simplex tends to contract around local minima. However, there are counterexamples where the method converges to a non-critical point for an only bivariate objective function that is strictly convex and twice continuously differentiable [McK98].

Differential evolution. The method of *differential evolution* [Stor97; Zie09; Vale14] is an evolutionary meta-heuristic algorithm. Being similar to genetic algorithms, the method maintains a *population* of m points that is iteratively updated according to pseudo-random *mutations,* which are weighted sums of the points of the previous generation. The mutated vector is *crossed over* with the original vector entry by entry. The resulting *offsprings* are only accepted if they lead to an improvement in terms of objective function value.



Method	Type	C	D	S	I
Nelder–Mead	Simplex heuristic	✗	0	✗	✓
Differential evolution	Evolutionary	✗	0	✓	✓
CMA-ES	Evolutionary	✗	0	✓	✓
Simulated annealing	Temperature heuristic	✗	0	✓	✗
PSO	Swarm heuristic	✗	0	✓	✗
GP-LCB	Bayesian	✗	0	✓	✗
Gradient descent	Descent	✗	1	✗	✓
NLCG	Descent	✗	1	✗	✓
Newton	Newton	✗	2	✗	✓
BFGS	Quasi-Newton	✗	1	✗	✓
Rprop	Heuristic	✗	1	✗	✓
Levenberg–Marquardt	Least sq., trust-region	✗	1	✗	✓
Log-barrier	Interior-point	✓	0+	–	✓
Squared penalty	Penalty	✓	0+	–	✓
Augmented Lagrangian	Penalty	✓	0+	–	✓
SQP	Quadratic subproblems	✓	2	–	✗

TABLE 5.1 Selection of optimization methods. The columns show if constrained problems are supported (C), the order of required derivatives (D), if the algorithm is stochastic (S), and if the algorithm has been implemented in SG^{++} (I).

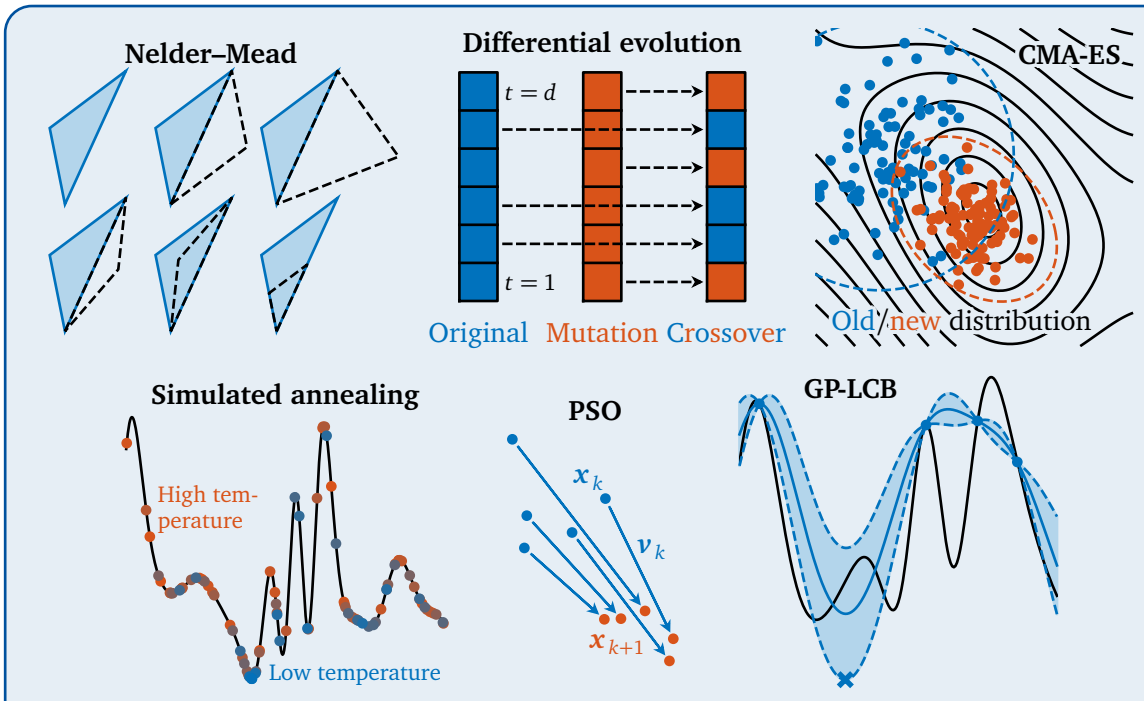


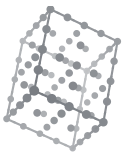
FIGURE 5.1 Sketch of the ideas of gradient-free optimization methods.

CMA-ES. *CMA-ES (covariance matrix adaption, evolution strategy)* [Hanse03] is an evolutionary algorithm that addresses the issue that simple evolution strategies do not prefer a search direction due to the lack of gradients [Tou15]. The name of the algorithm stems from the fact that it keeps track of the *covariance matrix* of the Gaussian search distribution. After m points have been sampled from the current distribution, the mean of the distribution for the next iteration is calculated as the weighted mean of the k best samples and the covariance matrix is adapted accordingly. An advantage of the method is that if the population is large enough, local minima are smoothed out [Tou15].

Simulated annealing. *Simulated annealing* [Laa87; Pre07; Kir14] imitates the cooling of a solid by randomly drawing samples from a proposal distribution and calculating an *acceptance probability* that depends on the function value improvement as well as on a *temperature* T. This temperature is slowly decreased in the course of the algorithm. Simulated annealing is closely connected to the Metropolis–Hastings algorithm for drawing random samples of arbitrary probability distributions. If run long enough, simulated annealing is guaranteed to find the global optimum [Tou15].

Particle swarm optimization (PSO). The method of *particle swarm optimization (PSO)* [Ken95; Zie09; Kir14] can be seen as another evolutionary algorithm that stems from swarm intelligence. For each *particle* of the population, not only the *position* $\boldsymbol{x}_k$ is stored, but also the current *velocity* $\boldsymbol{v}_k$, the best known position in a neighborhood of $\boldsymbol{x}_k$ (which may be the whole swarm), and the best known position of the k-th particle. The next velocity $\boldsymbol{v}_{k+1}$ is computed as a pseudo-randomly weighted sum of $\boldsymbol{v}_k$, the vector from $\boldsymbol{x}_k$ to the best neighborhood position, and the vector from $\boldsymbol{x}_k$ to the best own position.

GP-LCB. *GP-LCB (Gaussian process, lower confidence bound)* [Sri10; Tou15] is an example for a *Bayesian optimization* strategy. The objective function is treated as a stochastic process. A *prior distribution* is updated according to the previous function evaluations to calculate the *posterior distribution*. The posterior distribution is used to form the *acquisition function,* which in turn determines the point at which the objective function is evaluated next. The GP-LCB method is obtained by choosing *Gaussian processes* for the family of stochastic processes and *lower confidence bounds* (which are the sum of the mean and a multiple of the standard deviation) for the acquisition function.



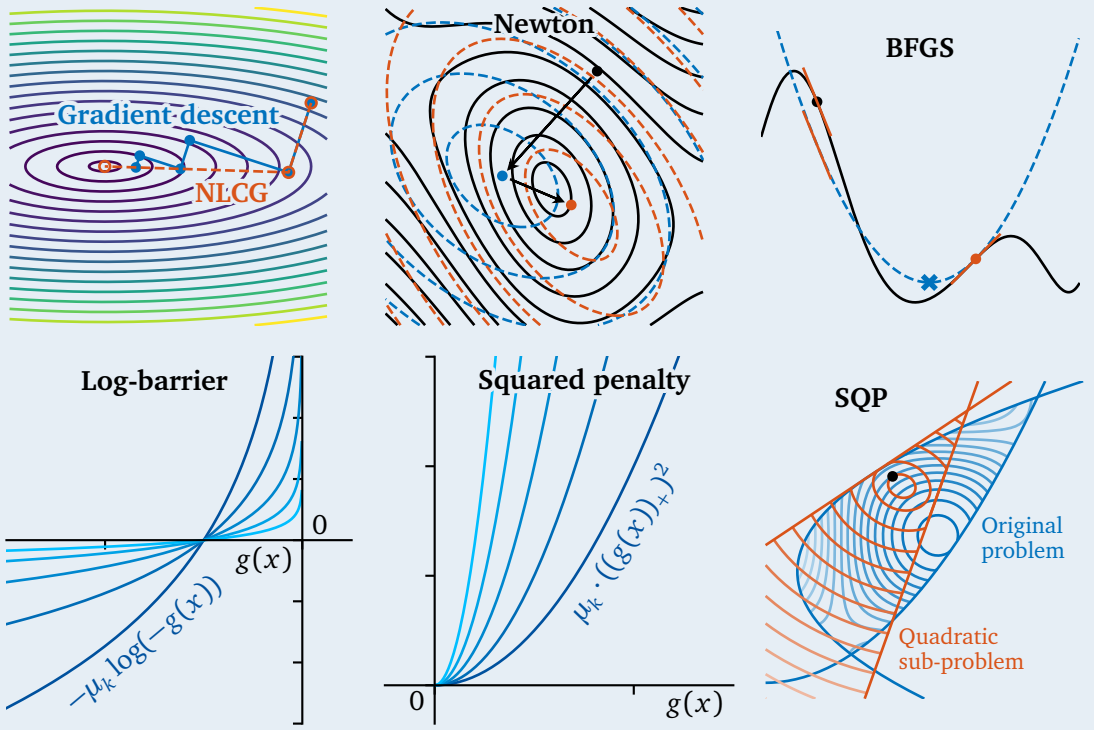


FIGURE 5.2 Sketch of the ideas of gradient-based or constrained optimization methods.

5.1.2 Gradient-Based Unconstrained Optimization Methods

Most gradient-based optimization algorithms determine in each iteration k a unit *search direction* $\boldsymbol{d}_k \in \mathbb{R}^d$ ($\|\boldsymbol{d}_k\|_2 = 1$) to update the current iterate $\boldsymbol{x}_k$:

$$\boldsymbol{x}_k \to \boldsymbol{x}_{k+1} := \boldsymbol{x}_k + \delta_k \boldsymbol{d}_k, \qquad \delta_k := \underset{\delta \in \mathbb{R}_{>0}}{\arg\min}\, f(\boldsymbol{x}_k + \delta \boldsymbol{d}_k), \tag{5.3}$$

where $\delta_k \in \mathbb{R}_{>0}$ is the *step size*. The algorithms essentially differ in the choice of the search direction $\boldsymbol{d}_k$, which should be oriented like the negative gradient ($\langle \boldsymbol{d}_k, \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k) \rangle_2 < 0$). The step size δ_k can then be determined independently of the algorithm via *line search,* for instance, the *Armijo line search algorithm* [Noc99; Ulb12; Vale14], which uses a heuristic acceptance criterion to find δ_k with a good enough improvement.

Gradient descent. *Gradient descent* [Ulb12; Vale14; Tou15] chooses $\boldsymbol{d}_k \propto -\nabla_{\boldsymbol{x}} f(\boldsymbol{x}_k)$ (i.e., normalized). The method suffers from slow convergence, if the Hessian $\nabla^2_{\boldsymbol{x}} f$ is ill-conditioned: One can show that for strictly convex quadratic functions, the error $f(\boldsymbol{x}_k) - f(\boldsymbol{x}^{\text{opt}})$ can decrease in each iteration only by the factor of $(\frac{\lambda^{\max} - \lambda^{\min}}{\lambda^{\max} + \lambda^{\min}})^2$, where $\lambda^{\min}$ and $\lambda^{\max}$ are the minimum and maximum eigenvalue of $\nabla^2_{\boldsymbol{x}} f$, respectively [Ulb12]. If the condition number $\frac{\lambda^{\max}}{\lambda^{\min}}$ of $\nabla^2_{\boldsymbol{x}} f$ is large, then this factor will be very close to one.

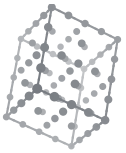
NLCG. A possible remedy for this issue is the method of *non-linear conjugate gradients (NLCG)* [Noc99; Vale14; Tou15]. It is equivalent to the CG method for solving symmetric positive definite linear systems $\boldsymbol{A}\boldsymbol{x} = \boldsymbol{b}$, if we optimize the strictly convex quadratic function $f(\boldsymbol{x}) := \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}^{\mathrm{T}}\boldsymbol{x}$ [Rei13; Vale14], i.e., it finds the optimum after only d steps for strictly convex quadratic functions. The NLCG method quickly converges even for non-convex objective functions, as due to the Taylor theorem, three times continuously differentiable functions with positive definite Hessian are "similar" to a strictly convex quadratic function in a neighborhood of $\boldsymbol{x}^{\mathrm{opt}}$ [Vale14].

Newton. The *Newton method* [Ulb12; Vale14; Tou15] replaces the objective function with the second-order Taylor approximation given by $f(\boldsymbol{x}_k+\boldsymbol{d}_k) \approx f(\boldsymbol{x}_k)+(\boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k))^{\mathrm{T}}\boldsymbol{d}_k + \frac{1}{2}(\boldsymbol{d}_k)^{\mathrm{T}}(\boldsymbol{\nabla}^2_{\boldsymbol{x}} f(\boldsymbol{x}_k))\boldsymbol{d}_k$ and determines the search direction such that $\boldsymbol{x}_k+\boldsymbol{d}_k$ is the minimum of the approximation, i.e., $\boldsymbol{d}_k \propto -(\boldsymbol{\nabla}^2_{\boldsymbol{x}} f(\boldsymbol{x}_k))^{-1} \boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k)$. Despite converging for strictly convex quadratic functions in a single step, the Hessian must not be ill-conditioned for the Newton method as well, as we have to solve a linear system with the matrix $\boldsymbol{\nabla}^2_{\boldsymbol{x}} f(\boldsymbol{x}_k)$. Hence, often a *regularization/damping term* $\lambda\boldsymbol{I}$ for some $\lambda > 0$ is added to the Hessian.

BFGS. The Newton method has the disadvantage that it needs to evaluate the Hessian $\boldsymbol{\nabla}^2_{\boldsymbol{x}} f$, which may be unavailable or too expensive. *Quasi-Newton methods* such as the method of *BFGS (Broyden, Fletcher, Goldfarb, Shanno)* [Noc99; Ulb12; Tou15] approximate the Hessian by a solution of the secant equation $\boldsymbol{\nabla}^2_{\boldsymbol{x}} f(\boldsymbol{x}_k)(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) \approx \boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k) - \boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_{k-1})$. As the solution is not unique for $d > 1$, Quasi-Newton methods differ in which solution to choose. The BFGS method performs a simple rank-one update.

Rprop. *Rprop (resilient propagation)* [Rie93; Tou15] considers the gradient entries $(\boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k))_t$ of each dimension $t = 1, \ldots, d$ separately and updates the entries $x_{k,t}$ of $\boldsymbol{x}_k$ according to the sign of the respective gradient entry, while adapting the step size dimension-wise. Although the algorithm is independent of the exact direction of $\boldsymbol{\nabla}_{\boldsymbol{x}} f(\boldsymbol{x}_k)$, it was found to often work robustly in machine learning scenarios [Tou15].

Levenberg–Marquardt. The *Levenberg–Marquardt method* [Noc99; Fre07; Tou15] can only solve *non-linear least-squares problems,* i.e., the objective function must be of the form $f(\boldsymbol{x}) = \|\boldsymbol{\phi}(\boldsymbol{x})\|_2^2 = \sum_{i=1}^{m_\phi} |\phi_i(\boldsymbol{x})|^2$ for some function $\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^{m_\phi}$. It is an improvement over the *Gauss–Newton method* (which is in turn a slight modification of the Newton method) and can be obtained by replacing the line search in the Gauss–Newton method with a *trust-region approach.*



5.1.3 Constrained Optimization Methods

Methods for constrained optimization usually solve a series of unconstrained *auxiliary problems* with an arbitrary unconstrained optimization method. The auxiliary function to be minimized is the sum of the objective function and *penalty terms,* which penalize if the current point $\boldsymbol{x}_k$ is near the boundary of the feasible domain or even outside. The penalty terms slowly increase to enforce the feasibility of the final result. Constrained optimization methods can roughly be divided into *interior-point or barrier methods,* where $\boldsymbol{x}_k$ always stays in the feasible domain, and *penalty methods,* where intermediate solutions $\boldsymbol{x}_k$ may be infeasible, in which case the penalty term is applied.

At least for the interior-point methods, a feasible initial solution $\boldsymbol{x}_0$ is required. We can find an initial solution by solving another auxiliary problem [Tou15], for instance

$$\min_{(\boldsymbol{x},s)\in\mathbb{R}^{d+1}} s \quad \text{s.t.} \quad s \geq 0,\ \boldsymbol{g}(\boldsymbol{x}) \leq s \cdot \mathbf{1}_{m_g}, \tag{5.4}$$

where $\mathbf{1}_{m_g} \in \mathbb{R}^{m_g}$ is the all-one vector. An initial solution for this problem can be explicitly given (for example, $\boldsymbol{x}_0 = \mathbf{0}$ and $s_0 = \max(\max(\boldsymbol{g}(\boldsymbol{x}_0)), 0)$).

Log-barrier. The *log-barrier method* [Boy04; Rei13; Tou15] is an interior-point method that adds a logarithmic *barrier function term* to the objective function near the boundary. The method solves $\min[f(\boldsymbol{x}) - \mu_k \sum_{i=1}^{m_g} \log(-g_i(\boldsymbol{x}))]$ for some decreasing $\mu_k \in \mathbb{R}_{>0}$.

Squared penalty. The *squared penalty method* [Pol71; Ulb12; Tou15] replaces the constrained problem with the penalized problem $\min[f(\boldsymbol{x}) + \mu_k \|(\boldsymbol{g}(\boldsymbol{x}))_+\|_2^2]$, where $\mu_k \in \mathbb{R}_{>0}$ is a penalty parameter and $(\cdot)_+ := \mathbf{max}(\cdot, \mathbf{0})$ denotes the non-negative part. With increasing μ_k, the constraint violation of the solution of the penalized problem decreases, although it may happen that it never vanishes.

Augmented Lagrangian. The method of the *augmented Lagrangian* [Rei13; Tou15] considers the auxiliary problem

$$\min_{\boldsymbol{x}\in\mathbb{R}^d} \left[f(\boldsymbol{x}) + \mu_k \sum_{i=1}^{m_g} [\lambda_{k,i} > 0]((g_i(\boldsymbol{x}))_+)^2 + \boldsymbol{\lambda}_k^{\mathrm{T}} \boldsymbol{g}(\boldsymbol{x}) \right], \tag{5.5}$$

where $[\lambda_{k,i} > 0] \in \{0, 1\}$ is defined as one if and only if $\lambda_{k,i} > 0$, and $\boldsymbol{\lambda}_k \in \mathbb{R}_{\geq 0}^{m_g}$ is an estimate of the *Lagrangian multipliers.* They are updated according to the penalty of the previous iteration, generating a "virtual gradient" that drastically decreases the necessary magnitude of the penalty parameter μ_k to achieve feasibility of the solution [Tou15].

Sequential quadratic programming (SQP). *Sequential quadratic programming (SQP) methods* [Ulb12; Rei13; Tou15] are one of the most powerful method classes for constrained optimization. They are motivated by the *Karush–Kuhn–Tucker (KKT) conditions,* which are necessary to hold in any optimal point (similarly to critical points in unconstrained optimization). The Newton method can be employed to solve the KKT conditions when written as a non-linear system of equations. The linear system of the resulting *Newton–Lagrange method* is equivalent to the KKT conditions of a *quadratic programming (QP) problem,* for which objective and constraint functions have the form $f(\boldsymbol{x}) = \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{Q}\boldsymbol{x} + \boldsymbol{d}^{\mathrm{T}}\boldsymbol{x}$ and $\boldsymbol{g}(\boldsymbol{x}) = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$, respectively.

5.2 Optimization of Surrogates on Sparse Grids

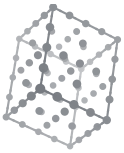
The methods presented in the last section can be combined to a "meta-method" for surrogate optimization. The surrogates are constructed as interpolants on spatially adaptive sparse grids, which we explain in the following.

IN THIS SECTION

5.2.1 Novak–Ritter Adaptivity Criterion

The classic surplus-based refinement strategy for spatially adaptive sparse grids is not tailored to optimization, as this refinement strategy aims to minimize the overall L^2 error. However, in optimization, it is reasonable to generate more points in regions where we suspect the global minimum to be to increase the interpolant's accuracy in these regions. Hence, we employ an adaptivity criterion proposed by Novak and Ritter [Nov96] for hyperbolic cross points. The Novak–Ritter criterion has also been applied to sparse grids [Fere05; Vale14; Vale16].

***m*-th order children.** As usual, the criterion works iteratively: Starting with an initial regular sparse grid of a very coarse level, the criterion selects a specific point $\boldsymbol{x}_{\ell,i}$ in each iteration and inserts all its children into the grid. This process is repeated until a given number $N_{\max}$ of grid points is reached, since we evaluate f at every grid point once, and we assume that function evaluations dominate the overall complexity. The difference to common refinement criteria is that a point may be selected multiple times, in which case



higher-order children are inserted. The m-th order children $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ of a grid point $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ satisfy

$$\boldsymbol{\ell}'_{-t} = \boldsymbol{\ell}_{-t},\ \boldsymbol{i}'_{-t} = \boldsymbol{i}_{-t},\ \ell'_t = \ell_t + m,\ i'_t \in \begin{cases} \{1\}, & (\ell_t = 0) \wedge (i_t = 0), \\ \{2^m - 1\}, & (\ell_t = 0) \wedge (i_t = 1), \\ \{2^m i_t - 1,\, 2^m i_t + 1\}, & \ell_t > 0, \end{cases} \tag{5.6}$$

where $m \in \mathbb{N}$ and $t \in \{1, \ldots, d\}$ (cf. Eq. (4.37) for $m = 1$). The order is chosen individually for each child point to be inserted as the lowest number m such that $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ does not yet exist in the grid.

Criterion. The Novak–Ritter refinement criterion [Nov96] refines the grid point $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ that minimizes the product[1]

$$(r_{\boldsymbol{\ell},\boldsymbol{i}} + 1)^{\gamma} \cdot (\|\boldsymbol{\ell}\|_1 + d_{\boldsymbol{\ell},\boldsymbol{i}} + 1)^{1-\gamma}. \tag{5.7}$$

Here, $r_{\boldsymbol{\ell},\boldsymbol{i}} := |\{(\boldsymbol{\ell}', \boldsymbol{i}') \in K \mid f(\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}) \leq f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})\}| \in \{1, \ldots, |K|\}$ is the *rank* of $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ (where K is the current set of level-index pairs of the grid), i.e., the place of the function value at $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ in the ascending order of the function values at all points of the current grid. We denote the *degree* $d_{\boldsymbol{\ell},\boldsymbol{i}} \in \mathbb{N}_0$ of $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ as the number of previous refinements of this point. Finally, $\gamma \in [0, 1]$ is the *adaptivity parameter*. We have to choose a suitable compromise between exploration ($\gamma = 0$) and exploitation ($\gamma = 1$). The best choice of course depends on the objective function f at hand, but for our purposes, we choose a priori a value of $\gamma = 0.15$. However, it may be an option to adapt the value of γ automatically during the grid generation phase.

5.2.2 Global Optimization of Sparse Grid Surrogates

Global, local, and globalized optimization methods. In Sec. 5.1, we presented various optimization methods for the unconstrained case, divided into global gradient-free methods such as differential evolution and local gradient-based methods, for example, gradient descent. A subset of these methods has been implemented in SG^{++} [Pfl10], see Tab. 5.1. The gradient-based methods need an initial point, and they may get stuck in local minima. Hence, we additionally implemented globalized versions of the gradient-based methods via a multi-start Monte Carlo approach with $m := \min(10d, 100)$ uniformly distributed pseudo-random initial points.[2] This means there are three types of methods:

[1] Compared to [Nov96], we added one in the base of each factor to avoid ambiguities for 0^0. In addition, we swapped γ with $1 - \gamma$ to make γ more consistent with its name as adaptivity.

[2] We split the number of permitted function evaluations evenly among the m parallel calls.

T1. Global gradient-free methods listed as implemented in Tab. 5.1

T2. Local gradient-based methods listed as implemented in Tab. 5.1[3]

T3. Globalized versions of the methods of type T2

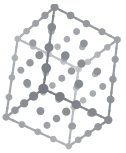
Unconstrained optimization of sparse grid surrogates. Given the objective function $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$, the maximal number $N_{\max} \in \mathbb{N}$ of evaluations of f, and the adaptivity parameter $\gamma \in [0, 1]$, we determine an approximation $\boldsymbol{x}^{\mathrm{opt},*} \in [\mathbf{0}, \mathbf{1}]$ of the global minimum $\boldsymbol{x}^{\mathrm{opt}}$ of f as follows:

1. Generate a spatially adaptive sparse grid Ω^{s} with the Novak–Ritter refinement criterion for f, $N_{\max}$, and γ.
2. Determine the sparse grid interpolant f^{s} of f by solving the linear system (4.1).
3. Optimize the interpolant: First, find the best grid point $\boldsymbol{x}^{(0)} := \mathbf{arg\,min}_{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in \Omega^{\mathrm{s}}} f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$. Second, apply the local methods of type T2 to the interpolant f^{s} with $\boldsymbol{x}^{(0)}$ as initial point. Let $\boldsymbol{x}^{(1)}$ be the resulting point with minimal objective function value. Third, we apply the global and globalized methods of types T1 and T3 to the interpolant f^{s}. Again, let $\boldsymbol{x}^{(2)}$ be the point with minimal f value. Finally, determine the point of $\{\boldsymbol{x}^{(0)}, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}\}$ with minimal f value and return it as $\boldsymbol{x}^{\mathrm{opt},*}$.

Note that the third step requires a fixed number of additional evaluations of the objective function, which can be neglected compared to $N_{\max}$. By default, we use the cubic modified hierarchical not-a-knot B-spline basis $\varphi_{\boldsymbol{\ell},\boldsymbol{i}}^{p,\mathrm{nak,mod}}$ ($p = 3$) for the construction of the sparse grid surrogate. However, we could apply any of the hierarchical (B-)spline bases presented in Chapters 3 and 4.

Comparison methods. We use two comparison methods. First, we apply the gradient-free methods (type T1) to the sparse grid interpolant using modified piecewise linear hierarchical basis functions (i.e., $p = 1$) on the same sparse grid as the cubic B-splines. We cannot employ gradient-based optimization as the objective function should be continuously differentiable and discontinuous derivatives are usually numerically problematic for gradient-based optimization methods (see, e.g., [Hüb14]). Second, we apply the gradient-free methods (type T1) directly to the objective function. We cannot use the gradient-based methods here as the gradient of the objective function is assumed to be unknown. For both of the comparison methods, we make sure that the objective function is evaluated at most $N_{\max}$ times by splitting the $N_{\max}$ evaluations evenly among all employed optimization methods.

[3] Excluding Levenberg–Marquardt, which is only applicable to least-squares problems.



Constrained optimization. For optimization problems with constraints, we proceed exactly as for unconstrained optimization, except that for optimizing the interpolant, we use the constrained optimization algorithms implemented in SG^{++} as listed in Tab. 5.1. We only replace the objective function f with a sparse grid surrogate f^s, and we assume that the constraint function $\boldsymbol{g}$ can be evaluated fast. However, it would also be possible to replace $\boldsymbol{g}$ with a sparse grid surrogate. In this case, it cannot be guaranteed that the resulting optimal point $\boldsymbol{x}^{\text{opt},*}$ is feasible, i.e., we could have $\neg(\boldsymbol{g}(\boldsymbol{x}^{\text{opt},*}) \leq \mathbf{0})$.

5.3 Test Problems

It is impossible to assess the capability of optimization methods for every possible optimization problem. The most widespread approach in literature is the selection of a subset of specific problems with different characteristics *(test problems)* and the application of the methods to only these problems, in the hope that the methods perform similarly in actual application settings.

Trivial test functions. When testing methods that involve sparse grid interpolation, one has to consider that the function to be interpolated does not satisfy a specific *trivial property*. A test function $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$ is trivial if f is a sum of tensor products of which at least one factor is a linear polynomial, i.e., if f is of the form

$$f(\boldsymbol{x}) \equiv \sum_{q=1}^{m} \prod_{t=1}^{d} f_{q,t}(x_t), \quad m \in \mathbb{N}_0, \;\; f_{q,t} : [0,1] \to \mathbb{R}, \;\; \forall_{q=1,\ldots,m} \exists_{t \in \{1,\ldots,d\}} \, f_{q,t} \in P^1, \tag{5.8}$$

where P^1 is the space of univariate polynomials up to linear degree. This is already fulfilled if the summands of $f(\boldsymbol{x})$ do not depend on all coordinates x_t of $\boldsymbol{x}$. One can show that for hat functions on sparse grids, the hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ for trivial functions vanish if $\boldsymbol{\ell} \geq \mathbf{1}$. This means that trivial functions can be well-approximated by hat functions on sparse grids just with boundary points, without placing any points in the interior. As this would distort our results, we avoid trivial test functions in the following, which include popular functions such as the Branin01, Rosenbrock, and Schwefel26 functions.

Selection of test problems. In the following, we select six unconstrained test problems and two constrained test problems, which are listed in Tab. 5.2 and plotted in Figures 5.3 and 5.4. The definitions of the problems are given in Appendix B. For the unconstrained case and the standard hierarchical B-spline basis, a more exhaustive list of test functions has been studied previously [Vale14]. Gavana [Gav13] and Runarsson/Yao [Run00] provide a good overview of unconstrained and constrained test problems, respectively.

Name	Abbr.	d	m_g	C	CD	MM	Reference
Branin02	Bra02	2	0	✓	✓	✓	[Mun98]
GoldsteinPrice	GoP	2	0	✓	✓	✓	[Gol71]
Schwefel06	Sch06	2	0	✓	✗	✗	[Schw77]
Ackley	Ack	d	0	✓	✓	✓	[Ack87]
Alpine02	Alp02	d	0	✓	✓	✓	[Cle99]
Schwefel22	Sch22	d	0	✓	✗	✗	[Schw77]
G08	G08	2	2	✓	✓	✓	[Schoena93]
G04Squared	G04Sq	5	6	✓	✓	✗	[Col68]

TABLE 5.2 Unconstrained *(top)* and constrained *(bottom)* test problems. The columns state the full and abbreviated names, the dimensionality d of the objective function f, the number m_g of constraints, whether f is continuous in the domain $[\mathbf{0}, \mathbf{1}]$ (C), whether f is continuously differentiable in the domain $[\mathbf{0}, \mathbf{1}]$ (CD), whether f is multi-modal (MM, i.e., whether there are multiple local minima), and a reference to the original literature that defines the problem.

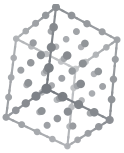
For each test problem, we state unscaled versions of objective functions $\bar{f} : [\boldsymbol{a}, \boldsymbol{b}] \to \mathbb{R}$, $\bar{\boldsymbol{x}} \mapsto \bar{f}(\bar{\boldsymbol{x}})$ (and the unscaled constraint function $\bar{\boldsymbol{g}}$, if present). The actual objective function $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$ can be obtained by $f(\boldsymbol{x}) := \bar{f}(\bar{\boldsymbol{x}})$ with the affine parameter transformation $x_t = \frac{\bar{x}_t - a_t}{b_t - a_t}$, $t = 1, \ldots, d$ (similarly for the constraint function).

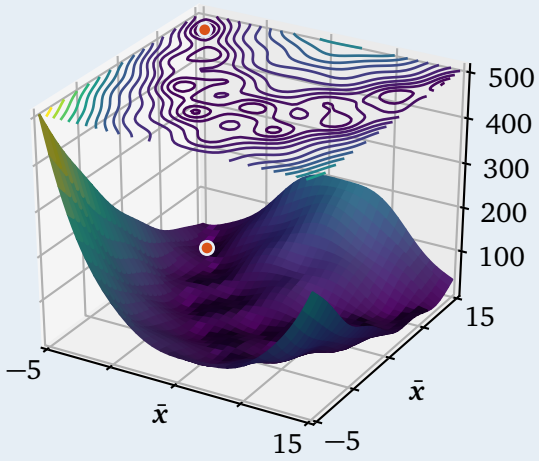
The parameter domains of some test problems have been slightly translated compared to the literature to avoid that the minima are located exactly at or close to the center of the domain. In these cases, sparse grids would be in advantage as they tend to place more points near the center of the domain (especially for high dimensionalities).

5.4 Numerical Results

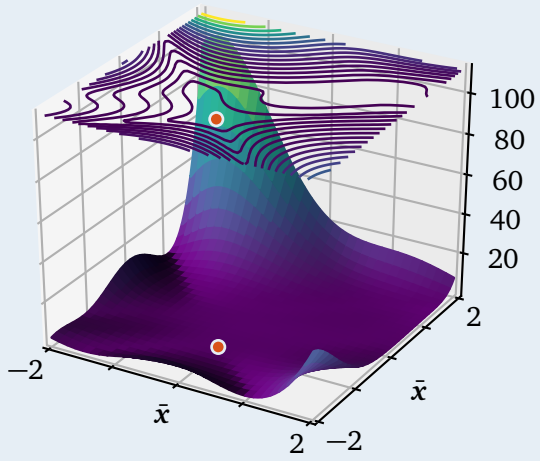
IN THIS SECTION

The following numerical experiments can be roughly divided into two parts. First, we study interpolation errors for the test functions to assess the effects of the hierarchical B-spline bases introduced in Chapters 2 and 3 on interpolation. Second, we consider the optimality gaps $f(\boldsymbol{x}^{\text{opt},*}) - f(\boldsymbol{x}^{\text{opt}})$ of the calculated approximations $\boldsymbol{x}^{\text{opt},*}$ of the point $\boldsymbol{x}^{\text{opt}}$ at which the objective function f is minimal.

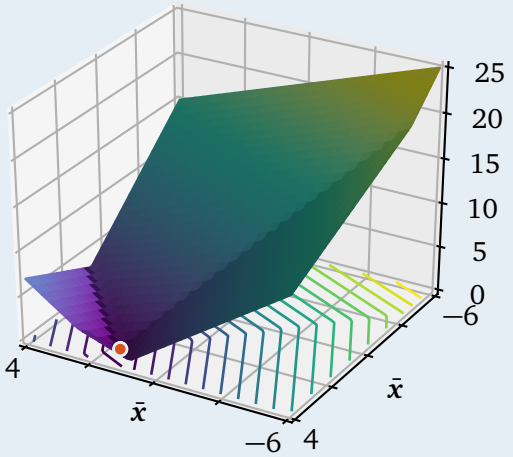




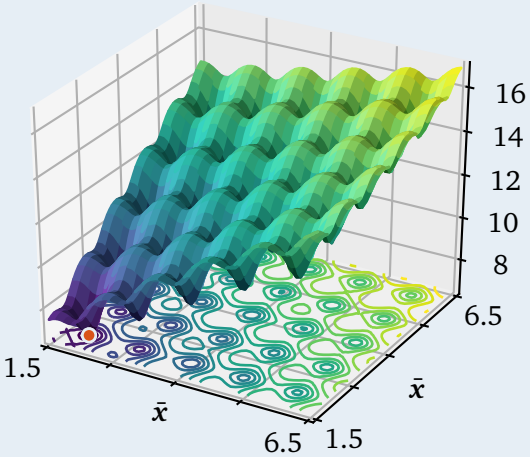
A Bra02



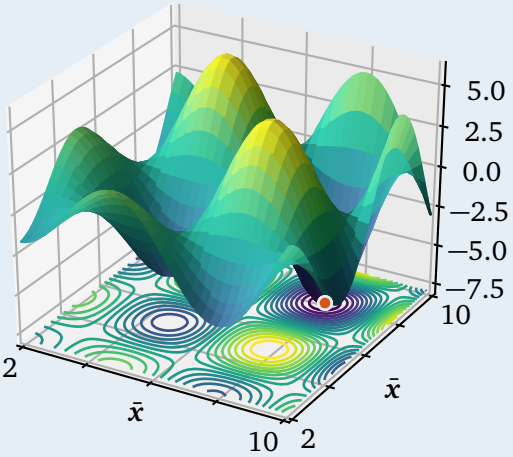
B GoP



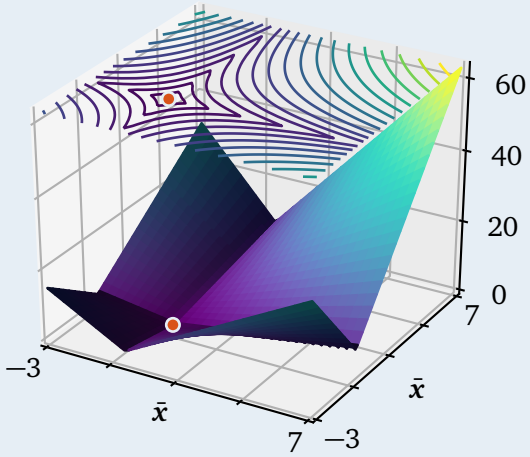
C Sch06



D Ack for $d = 2$

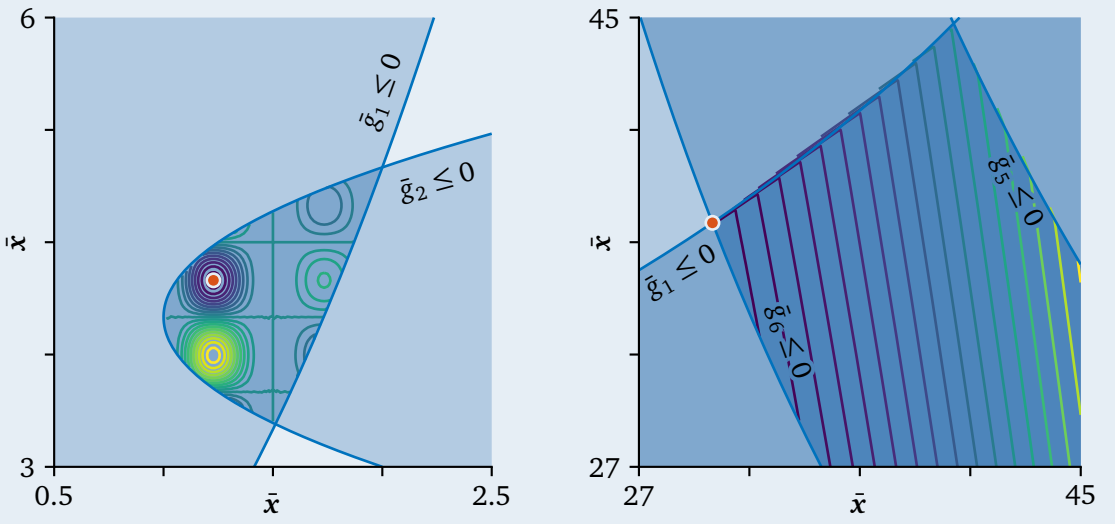


E Alp02 for $d = 2$



F Sch22 for $d = 2$

FIGURE 5.3 Bivariate test functions $\bar{f}$ in unconstrained optimization. The red dot indicates the location of the global minimum.



A G08

B G04Sq (bivariate projection over $\bar{x}_3$ and $\bar{x}_5$ onto $\bar{x}_t = \bar{x}_t^{\text{opt}}$ for $t = 1, 2, 4$)

FIGURE 5.4 Test problems in constrained optimization. The blue areas denote the inequality constraints and the red dot indicates the location of the global minimum.

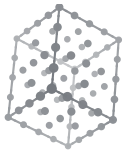
The results have been computed with the sparse grid toolbox SG^{++} [Pfl10],[4] which has been extended in the scope of this thesis. The new code has been written in such a way that it is scalable and efficient, while still being maintainable and portable [Pfl16].

5.4.1 Interpolation Error and Decay of Surpluses

Interpolation error for different test functions. Figure 5.5 shows the relative L^2 interpolation error $\frac{\|f - f^s\|_{L^2}}{\|f\|_{L^2}}$ of sparse grid interpolants f^s to different objective functions f (approximated via Monte Carlo quadrature using 10^4 uniformly pseudo-random samples). The interpolation is performed on regular sparse grids of increasing levels using hierarchical not-a-knot B-splines $\varphi_{\ell,i}^{p,\text{nak}}$ of degree $p = 1, 3, 5$. As a visual aid, the plots include gray lines that indicate different orders of convergence.

It is already known that—if the objective function is sufficiently smooth—the L^2 error of spline interpolants of degree p on d-dimensional regular sparse grids of level n asymptotically behaves like $\mathcal{O}(h_n^{p+1}(\log_2 h_n^{-1})^{d-1}) = \mathcal{O}(2^{-(p+1)n} n^{d-1})$ for $n \to \infty$ [Sic11]. We can numerically verify this fact easily with Fig. 5.5, in which we obtain the asserted orders of convergence for the bivariate functions that are continuously differentiable.

[4] `http://sgpp.sparsegrids.org/`



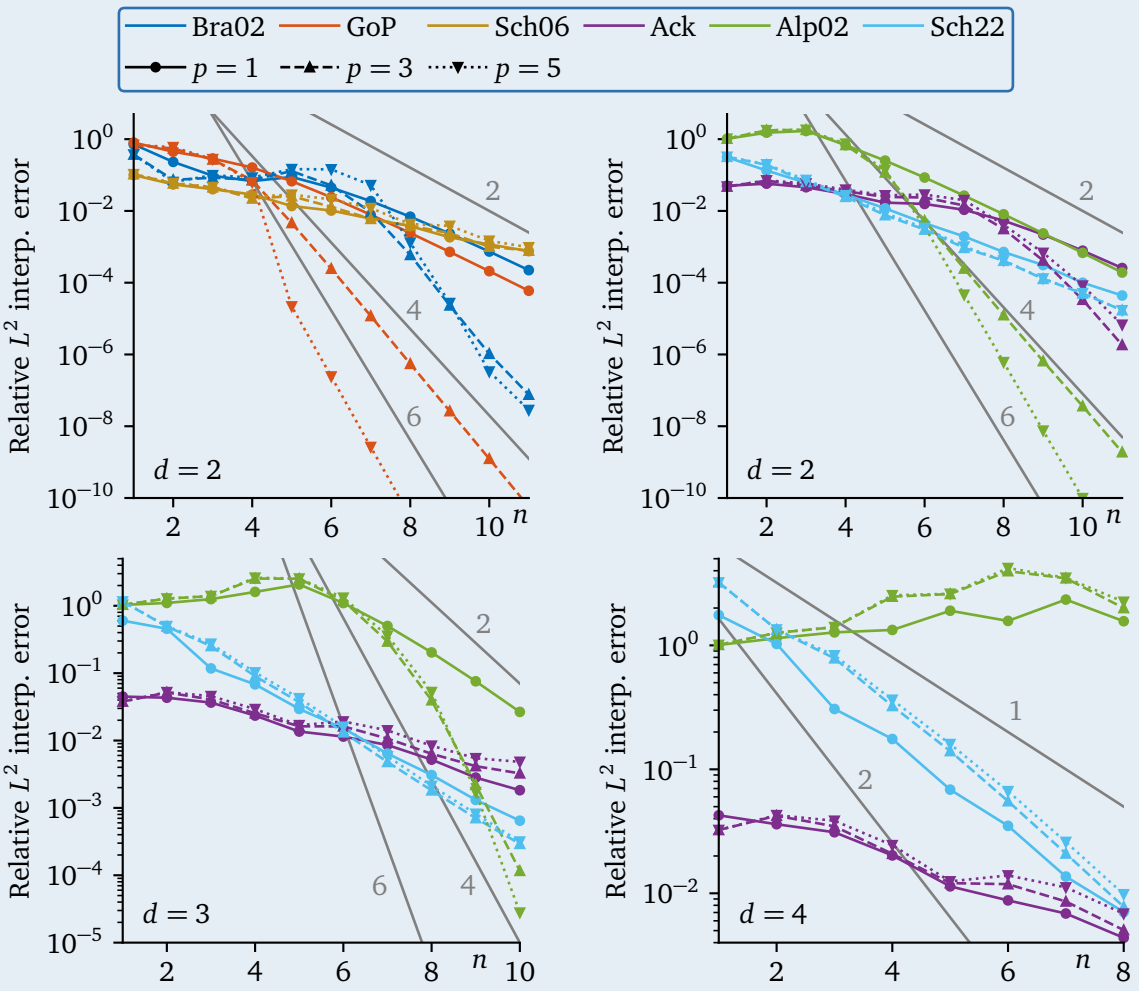


FIGURE 5.5 Relative L^2 interpolation error $\|f - f^s\|_{L^2} / \|f\|_{L^2}$ for different test functions f *(colors)* using hierarchical not-a-knot B-splines $\varphi_{\ell,i}^{p,\text{nak}}$ of different degrees p *(line styles/markers)* on regular sparse grids $\Omega_{n,d}^s$ of different levels n.

For the functions Sch06 and Sch22, which have a non-differentiable kink, only linear convergence can be achieved regardless of the B-spline degree.

The region where the asymptotic behavior dominates largely depends on the objective function at hand. Functions like Bra02 and Ack with many small oscillations require more interpolation points than "smoother" functions like GoP and Alp02. This is also the case for all functions in higher dimensionalities, as more interpolation points are necessary to sufficiently explore the domain (curse of dimensionality). In Fig. 5.5, this can already be seen for $d \geq 3$. This is not a consequence of employing higher-order B-splines for the hierarchical basis. However, it seems that higher-order B-splines lead to a slight increase of the interpolation error in the pre-asymptotic range.

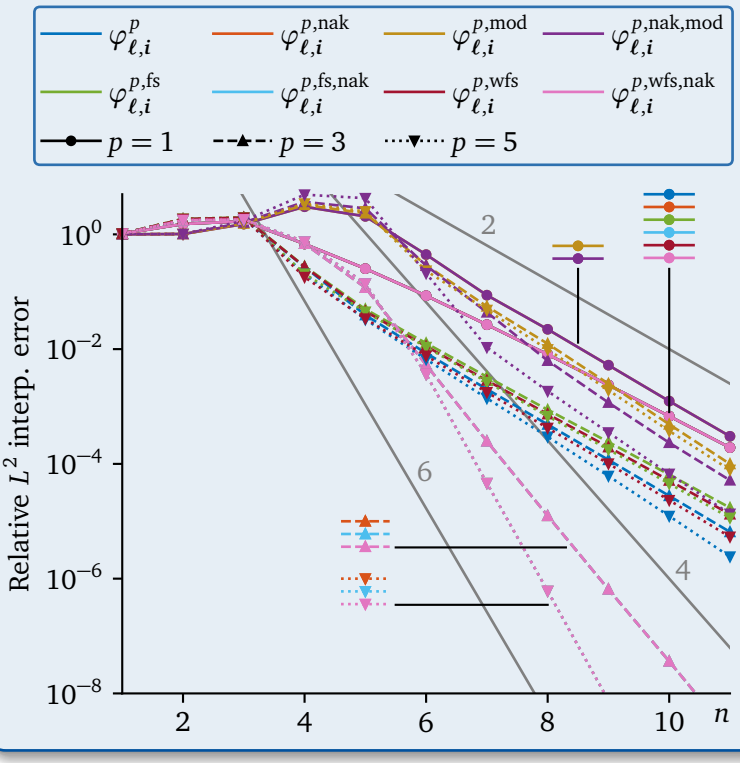


FIGURE 5.6
Relative L^2 interpolation error $\|f - f^s\|_{L^2}/\|f\|_{L^2}$ for the bivariate Alp02 function ($d = 2$) using different hierarchical basis functions $\varphi_{\ell,i}$ *(colors)* of different degrees p *(line styles/markers)* and regular sparse grids $\Omega^s_{n,d}$ of different levels n.
The basis functions shown here involve standard *(no superscript)*, not-a-knot (nak), modified (mod), fundamental (fs), and weakly fundamental (wfs) splines as well as the combinations introduced in Chapters 2 and 3.

Interpolation error for different basis functions. In Fig. 5.6, we fix the objective function and study the influence of the choice of hierarchical basis functions on the interpolation error. Shown are eight types of hierarchical B-spline bases as introduced in Chapters 2 and 3 for the degrees $p = 1, 3, 5$. Note that some lines exactly overlap, which is indicated in the figure.

For $p = 1$, the non-modified bases and the modified bases coincide. For higher degrees, the modified bases show worse results than the corresponding non-modified versions for the same level n. However, modified bases need significantly less grid points (no boundary points), which means that a direct comparison based on the sparse grid level n is somewhat skewed. In addition, we see that the not-a-knot bases coincide exactly for $p > 1$, as they span the same space for regular and dimensionally adaptive sparse grids. Only with the not-a-knot boundary conditions, we obtain the true theoretical order of convergence, which is $p + 1$ for degree p. Otherwise, only quadratic convergence can be achieved regardless of p, albeit with a smaller constant (offset).

Pointwise interpolation error. The importance of not-a-knot boundary conditions is also evident from plots of the pointwise interpolation error as in Fig. 5.7. The interpolation error grows for the standard hierarchical B-spline basis $\varphi^{p}_{\ell,i}$ as we move towards the boundary of the domain $[\mathbf{0}, \mathbf{1}]$, before dropping to zero or near-zero values at or near boundary grid points. With not-a-knot B-splines $\varphi^{p,\text{nak}}_{\ell,i}$, the interpolation error is uniformly

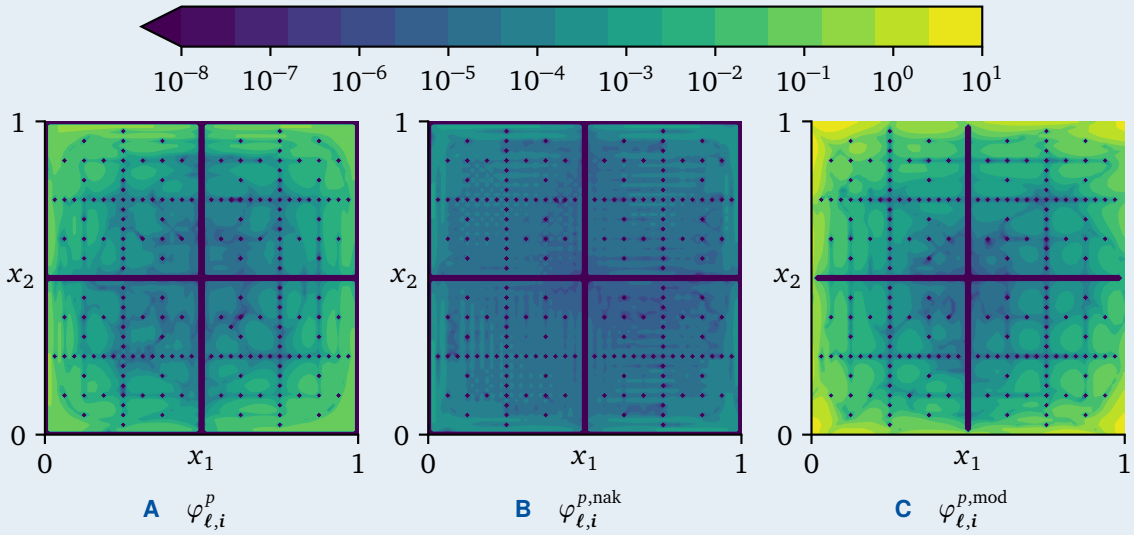


FIGURE 5.7 Pointwise interpolation error $|f(\boldsymbol{x}) - f^{\mathrm{s}}(\boldsymbol{x})|$ on a logarithmic scale for the bivariate GoP function ($d = 2$) using different hierarchical basis functions $\varphi_{\boldsymbol{\ell},\boldsymbol{i}}$ *(left, center, right)* of degree $p = 3$ on the regular sparse grid $\Omega^{\mathrm{s}}_{n,d}$ of level $n = 7$.

low. For comparison, modified B-splines $\varphi^{p,\mathrm{mod}}_{\boldsymbol{\ell},\boldsymbol{i}}$ incur even worse issues near the boundary, since the corresponding sparse grids do not contain boundary points.

Decay of surpluses. In the piecewise linear case ($p = 1$), the hierarchical surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ can be represented as the L^2 inner product of the corresponding hat function $\varphi^{1}_{\boldsymbol{\ell},\boldsymbol{i}}$ with the second mixed derivative $\frac{\partial^{2d}}{\partial x_1^2 \cdots \partial x_d^2} f$ of the objective function f, if $\boldsymbol{\ell} \geq \mathbf{1}$ and if this derivative exists and is continuous (see Eq. (2.25)). Consequently, one can prove that $|\alpha_{\boldsymbol{\ell},\boldsymbol{i}}| \leq 2^{-d} 2^{-2\|\boldsymbol{\ell}\|_1} \left\| \frac{\partial^{2d}}{\partial x_1^2 \cdots \partial x_d^2} f \right\|_{L^\infty}$ [Bun04], i.e., the absolute values of the hierarchical surpluses decay in quadratic order with the level sum $\|\boldsymbol{\ell}\|_1$. This relation can be used to estimate the convergent range of the corresponding interpolation error (Figures 5.5 and 5.6). A generalization of this estimate to higher B-spline degrees $p > 1$ is not straightforward, as the surpluses $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ then also depend on function values $f(\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'})$ at grid points of higher levels $\boldsymbol{\ell}' \geq \boldsymbol{\ell}$.

The decay of surpluses can be seen in Fig. 5.8, which shows the mean absolute value of surpluses corresponding to grid points grouped by their level sum $\|\boldsymbol{\ell}\|_1$. Due to the dependency of coarse-level surpluses on high-level grid points for $p > 1$, we have to fix the level n of the regular sparse grid for this analysis. Figure 5.8 suggests that the absolute value of the surpluses decays with order $p + 1$ for B-spline degree p, although no theoretical evidence is known to support this claim. Higher B-spline degrees seem to imply that $|\alpha_{\boldsymbol{\ell},\boldsymbol{i}}|$ generally increases, if $\|\boldsymbol{\ell}\|_1$ is in the pre-asymptotic range.

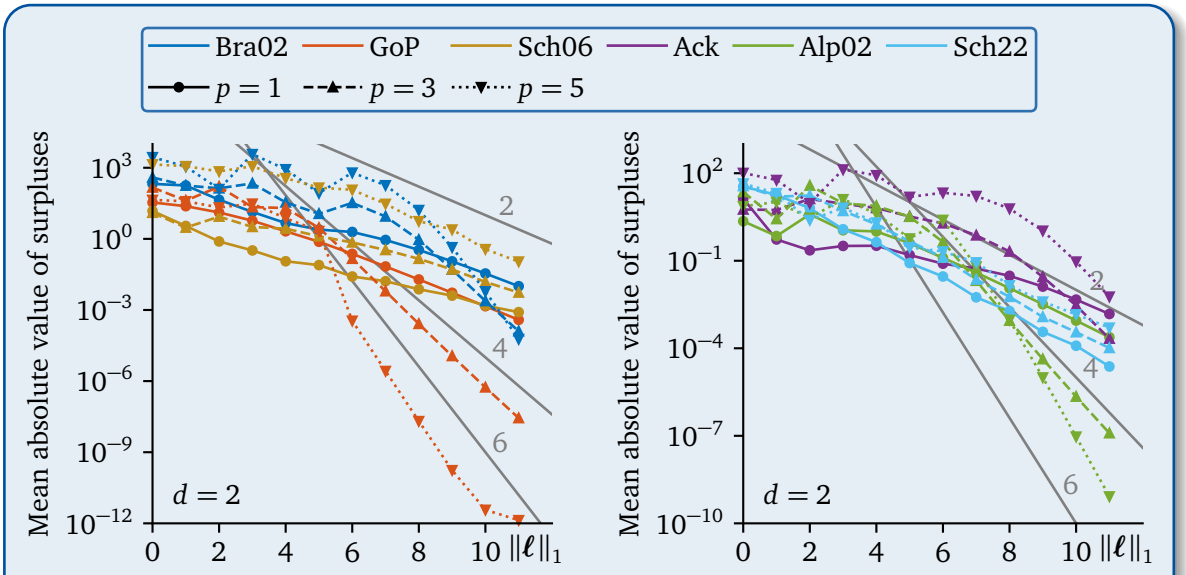


FIGURE 5.8 Mean absolute value of surpluses by level sum $\|\ell\|_1$ for different test functions f *(colors)* using hierarchical not-a-knot B-splines $\varphi_{\ell,i}^{p,\mathrm{nak}}$ of different degrees p *(line styles/markers)* on the regular sparse grid $\Omega_{n,d}^{s}$ of level $n = 11$.

5.4.2 Complexity of Hierarchization

In Chap. 4, we introduced a number of new hierarchical spline bases with the aim to reduce the complexity of algorithms with the key example of hierarchization. In the following, we study the suitability of the new bases to achieve this goal [Vale18a].

Complexity of fundamental splines. Figure 5.9 compares the hierarchization complexity of modified hierarchical B-splines $\varphi_{\ell,i}^{p,\mathrm{mod}}$ with the new modified hierarchical fundamental splines $\varphi_{\ell,i}^{p,\mathrm{fs,mod}}$ as measured on a laptop with Intel Core i5-4300U. For the modified hierarchical B-spline basis, we solve a linear system of size $N \times N$, for which Gaussian elimination takes $\Theta(N^3)$ time and $\Theta(N^2)$ memory for $N \to \infty$ (where N is the number of sparse grid points and d is assumed to be constant). More sophisticated methods to solve linear systems are not able to significantly reduce this complexity without any further assumptions on the system matrix $\boldsymbol{A}$ (e.g., symmetry, positive definiteness, or bandedness). As N grows, the space needed to store an $N \times N$ matrix quickly exceeds the available memory.

For the modified hierarchical fundamental splines, we can use the breadth-first search (BFS) algorithm presented in Sec. 4.4. BFS works in quadratic time $\mathcal{O}(N^2)$, but more importantly, it works in linear space $\mathcal{O}(N)$. Both can be seen very well in Fig. 5.9: The computation time drops from cubic to quadratic complexity for fundamental splines and the consumed memory is reduced from quadratic to linear complexity.

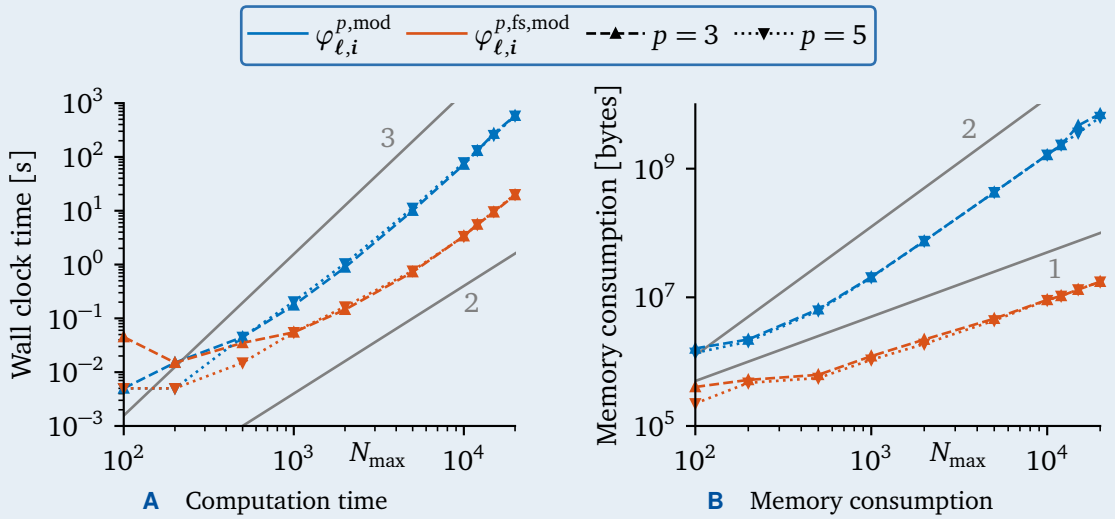


FIGURE 5.9 Computation time and memory consumption of hierarchization with modified hierarchical B-splines $\varphi_{\ell,i}^{p,\mathrm{mod}}$ *(blue)* and modified hierarchical fundamental splines $\varphi_{\ell,i}^{p,\mathrm{fs,mod}}$ *(red)* of degrees $p = 3$ and $p = 5$ on the spatially adaptive sparse grids generated by the criterion of Novak–Ritter for the optimization of Ack with $d = 4$ and $N_{\max}$ grid points. Adapted from [Vale18a].

Complexity of weakly fundamental splines. It is not straightforward to include weakly fundamental splines in Fig. 5.9 as we have to insert missing chain points to apply the unidirectional principle (see Sec. 4.5). As this increases the number of necessary evaluations of f, a comparison of computation times with standard B-splines would be skewed. Instead, we study in Fig. 5.10 the number of grid points that have to be inserted to ensure the correctness of the unidirectional principle. As we have seen in Sec. 4.5.3 (cf. Fig. 4.14), inserting all chains needed for standard hierarchical B-splines often results in a full grid, which suffers from the curse of dimensionality. This can be seen in Fig. 5.10 for hierarchical not-a-knot B-splines (dashed lines). By inserting all full grid points, the number of grid points increases by several orders of magnitude: If the initial grid has $N_{\max} = 10\,000$ points, then the grid size increases roughly by the factor 10^2 for $d = 2$, 10^4 for $d = 3$, and 10^6 for $d = 4$, resulting in computationally infeasible grids with 10^6, 10^8, and 10^{10} points, respectively. If we instead only insert the missing chain points needed for the hierarchical weakly fundamental not-a-knot basis (solid lines, cf. Fig. 4.16), then the number of grid points increases only slightly. For grids that have a low adaptivity (which correspond to low adaptivity parameters γ in the Novak–Ritter criterion, see Sec. 5.2.1), the grid size only increases by the factor of two. For highly-adaptive grids (corresponding to large γ), the number of necessary chain grid points increases significantly.

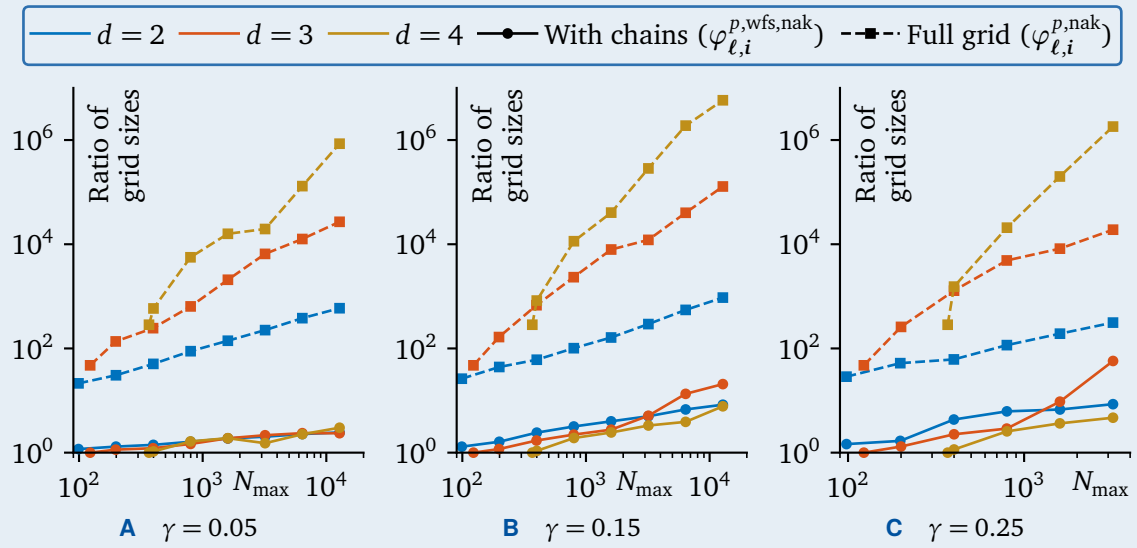


FIGURE 5.10 Total number of grid points after inserting all missing chains for cubic weakly fundamental not-a-knot splines $\varphi_{\ell,i}^{p,\text{wfs,nak}}$ ($p = 3$, *solid lines*), and after inserting all missing full grid points *(dashed)*. Shown are the ratios of the resulting grid sizes to the initial grid sizes before inserting points. The initial grids are the spatially adaptive sparse grids generated by the criterion of Novak–Ritter for the optimization of Ack with different dimensionalities *(colors)*, different adaptivity parameters γ *(left, center, right)*, and $N_{\max}$ grid points.

5.4.3 Optimality Gap

Optimality gaps and displacements. With the method described in Sec. 5.2, we find approximations $\boldsymbol{x}^{\text{opt},*}$ of the global minimum $\boldsymbol{x}^{\text{opt}}$ of some objective function f using optimization of a B-spline surrogate f^{s} of f on sparse grids. Obviously, the more accurate the sparse grid surrogate is, the better the approximation $\boldsymbol{x}^{\text{opt},*}$ will be. In the following plots, we show the optimality gaps $f(\boldsymbol{x}^{\text{opt},*}) - f(\boldsymbol{x}^{\text{opt}})$ in terms of function values.[5] The results are sensitive to even small displacements of the objective function, i.e., the results may change for the function $\boldsymbol{x} \mapsto f(\boldsymbol{x} - \boldsymbol{a})$ instead of $\boldsymbol{x} \mapsto f(\boldsymbol{x})$ for $\boldsymbol{x} \in [\boldsymbol{0}, \boldsymbol{1}]$ and some small $\boldsymbol{a} \in \mathbb{R}^d$.[6] Therefore, the optimization for each of the data points for Figures 5.11 and 5.12 was repeated five times with replacements $\boldsymbol{a}$ whose entries a_t were independent and identically distributed Gaussian pseudo-random numbers with zero mean and a standard deviation of 0.01. The optimality gaps shown in the figures of this section were computed as the mean of the five runs to increase confidence in the results.

[5] In order to calculate the optimality gap, it is crucial to determine $f(\boldsymbol{x}^{\text{opt}})$ as exact as possible. Otherwise, the optimality gap might either not converge to zero or it might even become negative.

[6] By using the formulas in Appendix B, all test functions f in Sec. 5.3 can be extended such that they can be evaluated at $\boldsymbol{x} - \boldsymbol{a}$ for all $\boldsymbol{x} \in [\boldsymbol{0}, \boldsymbol{1}]$, if $\boldsymbol{a} \in \mathbb{R}^d$ is small enough. Note that we set a_t to zero if a non-zero displacement in the t-th component would change the location of the global minimum.

Unconstrained optimization. Figure 5.11 shows the optimality gaps for different test functions f over the number $N_{\max}$ of allowed evaluations of f. For the continuously differentiable functions Bra02, GoP, Ack, and Alp02 in $d = 2$ dimensions (top row), the optimization of the corresponding cubic B-spline surrogates (solid lines) performs significantly better than using piecewise linear basis functions (dashed lines). The reason is two-fold: First, by using higher-order basis functions, the surrogates are more accurate in general as seen in the discussion of the interpolation error in Sec. 5.4.1. Second, the availability of surrogate gradients accelerates the convergence of the employed optimization methods. For some test functions, B-splines give better results than even the direct optimization of the objective function (dotted lines).

For the test functions Sch06 and Sch22 with discontinuous derivatives, the advantage of higher-order B-splines is not as evident (Sch22) or does not even exist (Sch06). However, in low dimensions, i.e., $d \leq 4$, B-splines achieve a slight advantage compared to the piecewise linear basis for the Sch22 function. In higher dimensionalities, i.e., $d \geq 6$ (bottom row), convergence visibly slows down for all methods shown in Fig. 5.11, although for some objective functions, B-splines are still able to perform better than the comparison methods (most notably for the Ack function).

Constrained optimization. Figure 5.12 shows the result for the two constrained optimization problems. The objective function value $f(\boldsymbol{x}^{\text{opt},*})$ at the approximated optimum $\boldsymbol{x}^{\text{opt},*}$ should not only be as small as possible, but $\boldsymbol{x}^{\text{opt},*}$ should also be feasible, i.e., $\boldsymbol{g}(\boldsymbol{x}^{\text{opt},*}) \leq \boldsymbol{0}$. Hence, we also plot the maximal violation $\|(\boldsymbol{g}(\boldsymbol{x}^{\text{opt},*}))_+\|_\infty$ of the constraints in the respective optimal points $\boldsymbol{x}^{\text{opt},*}$.

For the bivariate G08 problem, the hierarchical B-splines surrogates perform better than the direct gradient-free optimization of the problem for $N_{\max} \leq 3500$ objective function evaluations and better than the piecewise linear surrogate for $N_{\max} \geq 300$ objective function evaluations. All calculated points are feasible.[7]

The range of the objective function of the five-dimensional G04Sq problem is larger than the range of G08. This results in generally higher optimality gaps $f(\boldsymbol{x}^{\text{opt},*}) - f(\boldsymbol{x}^{\text{opt}})$ as we do not normalize with respect to the range. B-splines achieve good approximations $\boldsymbol{x}^{\text{opt},*}$ of $\boldsymbol{x}^{\text{opt}}$ already for $N_{\max} = 1000$ with an optimality gap of around one. Both comparison methods show optimality gaps that are seven orders of magnitude higher.

Additionally, the corresponding values of constraint violation are between 10^{-10} and 10^{-6}, i.e., the constraints are numerically met. In contrast, the optimizers struggle more for the comparison methods (optimization of the linear surrogate and of the objective function) to meet the constraints, as the values of the constraint violation partly exceed

[7] For plotting reasons, Fig. 5.12 shows $\max(\|(\boldsymbol{g}(\boldsymbol{x}^{\text{opt},*}))_+\|_\infty, 10^{-16})$ instead of the true constraint violation.

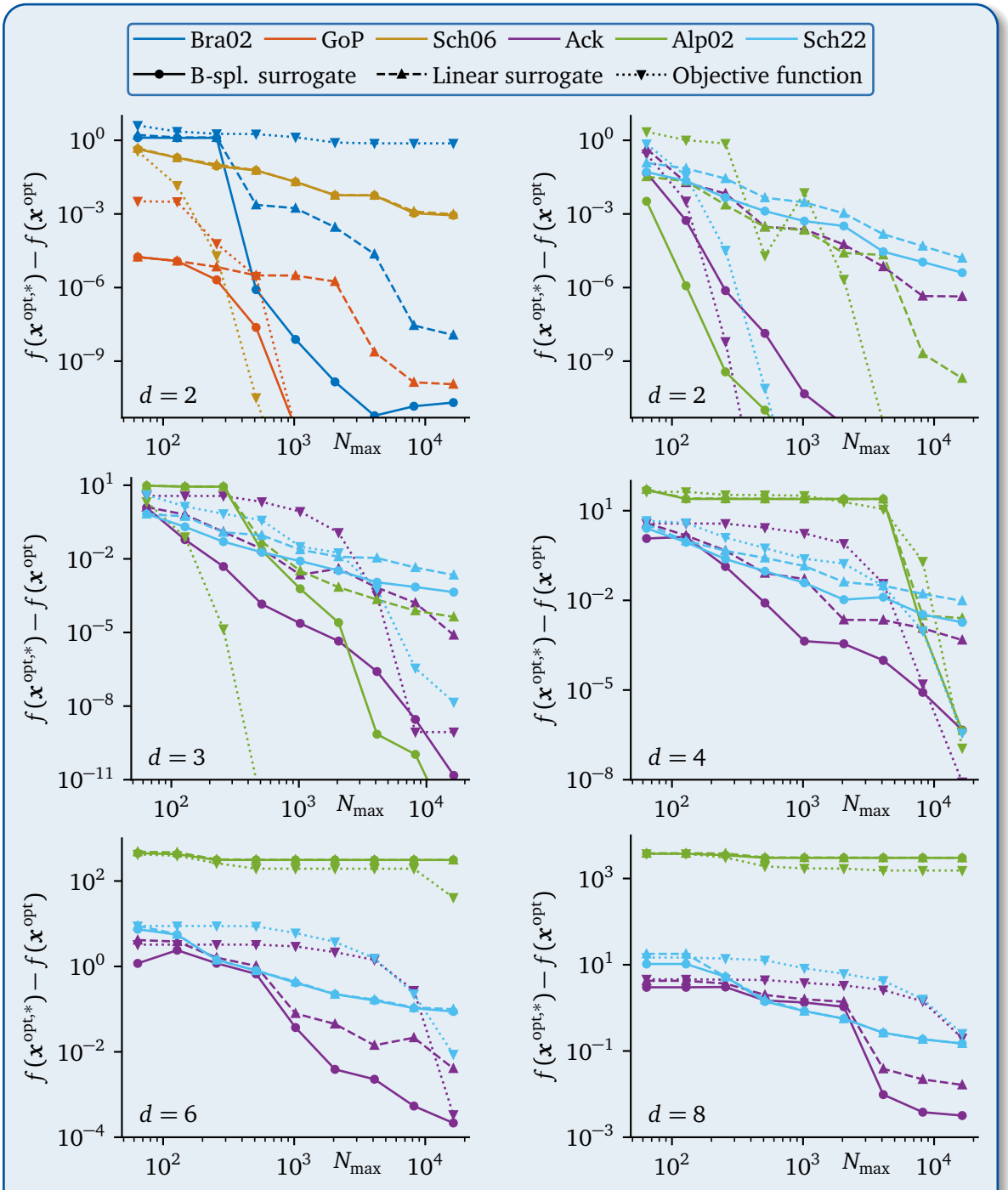


FIGURE 5.11 Optimality gaps $f(\boldsymbol{x}^{\text{opt},*}) - f(\boldsymbol{x}^{\text{opt}})$ between the function value at the approximated optimum $\boldsymbol{x}^{\text{opt},*}$ and the minimal function value at the actual optimum $\boldsymbol{x}^{\text{opt}}$ over the number N_{max} of objective function evaluations for different unconstrained objective functions f *(colors)*. Shown are the optimization results of the B-spline surrogate *(solid lines)*, the optimization results of the piecewise linear surrogate *(dashed)*, and the optimization results of the actual objective function *(dotted)* as described in Sec. 5.2.

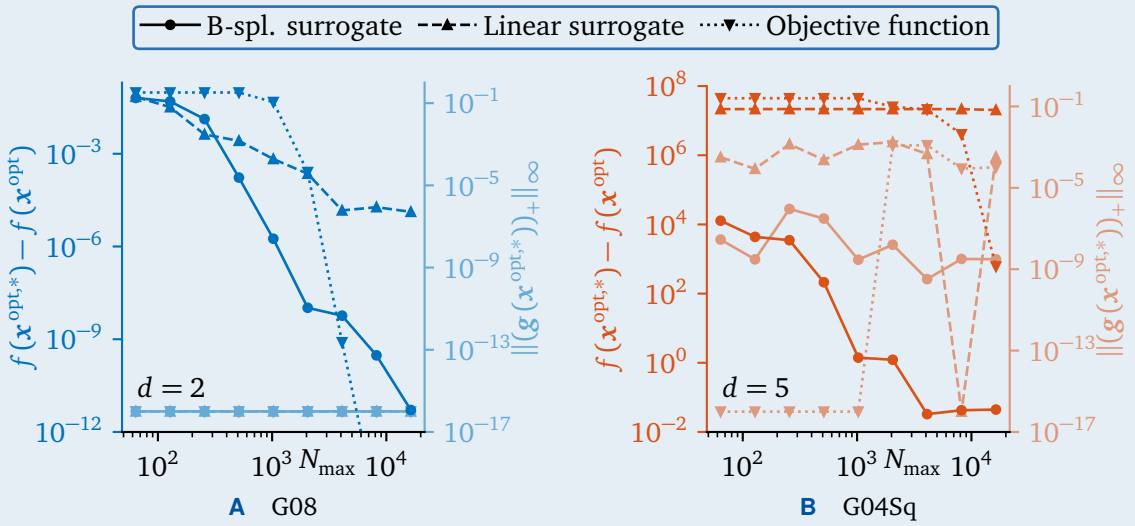


FIGURE 5.12 Optimality gaps $f(\boldsymbol{x}^{\mathrm{opt},*}) - f(\boldsymbol{x}^{\mathrm{opt}})$ between the function value at the approximated optimum $\boldsymbol{x}^{\mathrm{opt},*}$ and the minimal function value at the actual optimum $\boldsymbol{x}^{\mathrm{opt}}$ over the number $N_{\max}$ of objective function evaluations for different constrained optimization problems *(dark, left vertical axes)*. Additionally to the lines of Fig. 5.11, the constraint violation $\|(\boldsymbol{g}(\boldsymbol{x}^{\mathrm{opt},*}))_+\|_\infty$ at the approximated optimum $\boldsymbol{x}^{\mathrm{opt},*}$ is plotted *(light, right vertical axes)*.

10^{-3}. The availability of gradients seems to allow the constrained optimization methods to better enforce the feasibility of the resulting points $\boldsymbol{x}^{\mathrm{opt},*}$.

Note that while the results look already promising for the B-spline surrogate method, these results could still be improved upon. The Novak–Ritter criterion used to generate the spatially adaptive sparse grids does not take the constraints into account. Consequently, many sparse grid points are created outside the feasible domain. By modifying the criterion to prefer points that are in a neighborhood of the feasible domain, the quality of the interpolant close to potential optima should increase.

5.5 Example Application: Fuzzy Extension Principle

To conclude this chapter, we consider the fuzzy extension principle as an example application of optimization of B-spline sparse grid surrogates.

IN THIS SECTION

Aleatoric and epistemic uncertainties. Classical uncertainty quantification (UQ) distinguishes between aleatoric and epistemic uncertainties [Wal16]. Aleatoric uncertainties result from the variability of inputs or model

components and from the "intrinsic randomness" of quantities. They are best described by probability theory, giving exact probabilities. Epistemic uncertainties arise from subjectivity, simplifying modeling assumptions, and incomplete knowledge. These uncertainties are better captured by fuzzy theory, which is more imprecise than the "exact" stochastic assumptions of probabilities [Wal16].

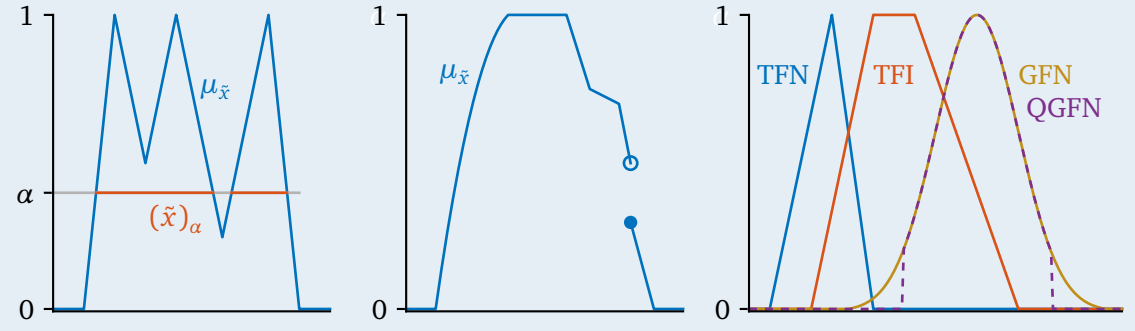
Uncertainty quantification with fuzzy uncertainties. In uncertainty quantification, the key question is as follows: Given a model and uncertain input parameters for the model, how uncertain is the model output? While there are many approaches available for probabilistic uncertainties, it is not straightforward to solve this task for fuzzy uncertainties. Fortunately, Zadeh proposed in 1975 the *fuzzy extension principle* [Zad75], which addresses this very question.

Sparse grids and B-splines for fuzzy uncertainties. As we explain in this section, the fuzzy extension principle requires the solution of numerous optimization problems that involve the original objective function f. This predestines the replacement of f with sparse grid surrogates, as explained in the beginning of the chapter. Previous work by Klimke [Kli06] already studied this approach for piecewise linear functions on uniform sparse grids and for global polynomials on sparse Clenshaw–Curtis grids. We assess the suitability of interpolation with higher-order hierarchical B-splines on sparse grids for the fuzzy extension principle. It should be mentioned that there is also work directly incorporating (non-hierarchical) B-splines into the framework of fuzzy theory for modeling uncertain surfaces [Ani00; Zak14].

5.5.1 Fuzzy Sets and Fuzzy Intervals

In the following, we repeat very briefly the necessary definitions of basic fuzzy theory. Examples for the definitions are shown in Fig. 5.13. A more in-depth introduction can be found in [Hanss05; Kli06; Wal16].

Fuzzy sets. Let $X \subseteq \mathbb{R}$ be a closed interval on the real line and $\mu_{\tilde{x}} : X \to [0,1]$ be a function. We call the graph $\tilde{x} := \{(x, \mu_{\tilde{x}}(x) \mid x \in X\}$ of $\mu_{\tilde{x}}$ a *fuzzy set* with *membership function* $\mu_{\tilde{x}}$. Fuzzy sets generalize ordinary subsets of X, which can be obtained by requiring $\mu_{\tilde{x}}(X) \subseteq \{0,1\}$. In this case, the fuzzy set is called *crisp* and $\tilde{x}$ can be identified with the ordinary set $\{x \in X \mid \mu_{\tilde{x}}(x) = 1\}$. A fuzzy set $\tilde{x}$ is *normalized* if $\max_{x \in X} \mu_{\tilde{x}}(x) = 1$. A *convex* fuzzy set $\tilde{x}$ satisfies $\min(\mu_{\tilde{x}}(a), \mu_{\tilde{x}}(c)) \leq \mu_{\tilde{x}}(b)$ for all $a, b, c \in X$ with $a \leq b \leq c$.



A Non-convex fuzzy set *(blue)* and α-cut *(red)*.

B Fuzzy interval.

C Common types of fuzzy numbers and intervals (see text).

FIGURE 5.13 Examples of membership functions of fuzzy sets and α-cuts.

Fuzzy intervals and α-cuts. A convex and normalized fuzzy set $\tilde{x}$ with piecewise continuous membership function $\mu_{\tilde{x}}$ is called *fuzzy interval*. If $\{x \in X \mid \mu_{\tilde{x}}(x) = 1\} = \{a\}$ for some $a \in X$, then the fuzzy interval $\tilde{x}$ is called *fuzzy number*.

For $\alpha \in [0,1]$, the α-cut of $\tilde{x}$ is defined as $(\tilde{x})_\alpha := \{x \in X \mid \mu_{\tilde{x}}(x) \geq \alpha\}$ for $\alpha > 0$ and $(\tilde{x})_0 := \operatorname{supp} \mu_{\tilde{x}}$ for $\alpha = 0$. The α-cuts of fuzzy intervals $\tilde{x}$ are always nested closed intervals, i.e., $(\tilde{x})_\alpha = [a,b]$ for some $a \leq b$ and $(\tilde{x})_{\alpha_1} \supseteq (\tilde{x})_{\alpha_2}$ for $\alpha_1 \leq \alpha_2$.

Common types of fuzzy numbers and intervals. There are various types of fuzzy numbers and intervals [Kli06]. Most common are *triangular fuzzy numbers* (TFNs, i.e., linear B-splines), *trapezoidal fuzzy intervals* (TFIs, where a plateau of height one is inserted at the peak, i.e., sums of two neighboring linear B-splines), and *Gaussian fuzzy numbers* (GFNs) with membership function $\mu_{\tilde{x}}(x) = \exp(-\frac{(x-\mu)^2}{(2\sigma)^2})$. As the support of Gaussian fuzzy numbers is unbounded, *quasi-Gaussian fuzzy numbers* (QGFNs) truncate the support to a fixed multiple of the standard deviation σ [Kli06]. However, it would be more natural to directly employ B-splines of degree $p > 1$ (normalized adequately), since they generalize triangular fuzzy numbers and their limit with respect to p is a Gaussian fuzzy number.

5.5.2 Fuzzy Extension Principle

Let $f : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$ be an objective function, whose values $y = f(\boldsymbol{x})$ represent the results of the simulation of a model with input parameters $(x_1, \ldots, x_d) = \boldsymbol{x}$. If the input parameters are uncertain and given as fuzzy sets $\tilde{x}_1, \ldots, \tilde{x}_d$, what is the resulting uncertain outcome "$\tilde{y} := f(\tilde{x}_1, \ldots, \tilde{x}_d)$"? Note that there is no definite answer to this question, as "$f(\tilde{x}_1, \ldots, \tilde{x}_d)$" is not well-defined. The fuzzy extension principle, suggested by Zadeh [Zad75], provides one possible definition.

Alternative fuzzy extension principle. We use an alternative formulation of the fuzzy extension principle, which is stated in [Kli06]. The original formulation is computationally more complex, as it requires the solution of equality-constrained optimization problems and one needs to know the range of f, which might not be given. The two formulations are equivalent, if $\tilde{x}_1, \ldots, \tilde{x}_d$ are (compactly supported) fuzzy intervals and f is continuous [Buc90], which we assume in the following.

The alternative fuzzy extension principle defines "$\tilde{y} = f(\tilde{x}_1, \ldots, \tilde{x}_d)$" as the fuzzy set $\tilde{y}$ with

$$\mu_{\tilde{y}}(y) := \sup\{\alpha \in [0,1] \mid y \in (\tilde{y})_\alpha\}, \quad y \in \mathbb{R}, \tag{5.9a}$$

$$(\tilde{y})_\alpha := \left[\min_{\boldsymbol{x} \in \Omega_\alpha} f(\boldsymbol{x}),\ \max_{\boldsymbol{x} \in \Omega_\alpha} f(\boldsymbol{x})\right], \qquad \alpha \in [0,1], \tag{5.9b}$$

$$\Omega_\alpha := (\tilde{x}_1)_\alpha \times \cdots \times (\tilde{x}_d)_\alpha, \qquad \alpha \in [0,1]. \tag{5.9c}$$

This definition is visualized in Fig. 5.14. The first equation defines $\tilde{y}$ via its α-cuts, which are given in the second equation as the closed interval between the minimal and the maximal value of f on some hyper-rectangular domain Ω_α. The third equation specifies this domain Ω_α as the Cartesian product of the univariate α-cuts. Hence, we only have to solve box-constrained optimization problems, as opposed to the general equality-constrained problems in the original formulation of the fuzzy extension principle.

Implementation. The implementation of the alternative fuzzy extension principle is straightforward and shown in Alg. 5.1 [Kli06]. The range $[0,1]$ of α is discretized into $m+1$ uniformly spaced values α_j (where $m \in \mathbb{N}$). For each of these values α_j, we compute the corresponding α_j-cut of $\tilde{y}$ by solving the two box-constrained optimization problems of Eq. (5.9). The fuzzy output interval $\tilde{y}$ can then be approximated by interpolating the interval bounds of the α_j-cuts of $\tilde{y}$.

5.5.3 Using B-Splines on Sparse Grids to Propagate Fuzzy Uncertainties

Following Klimke's approach [Kli06], we replace the objective function f in Alg. 5.1 with a sparse grid surrogate f^s. The solution of the optimization problems $\min_{\boldsymbol{x} \in \Omega_{\alpha_j}} f^s(\boldsymbol{x})$ and $\max_{\boldsymbol{x} \in \Omega_{\alpha_j}} f^s(\boldsymbol{x})$ with respect to the surrogate f^s instead of the true objective function f takes significantly less time, if evaluations of the objective function are expensive.

However, Klimke used piecewise linear functions as the hierarchical basis on uniform sparse grids and global polynomials on sparse Clenshaw–Curtis grids. The drawbacks of each of the bases are evident: First, piecewise linear surrogates are not continuously

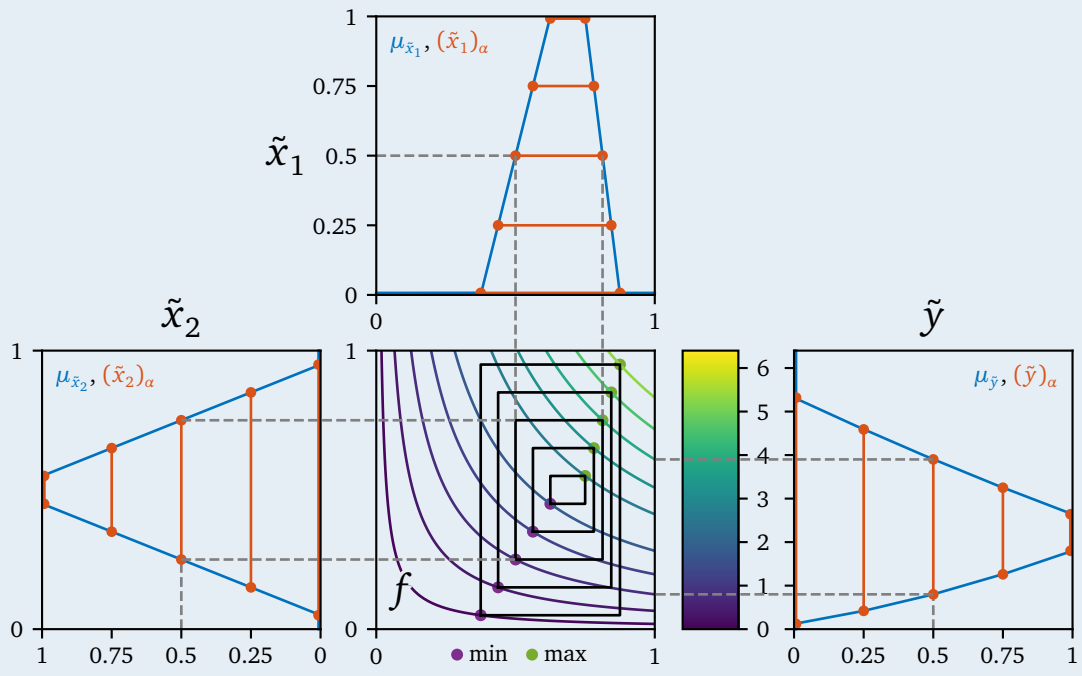


FIGURE 5.14 Example of the application of the alternative fuzzy extension principle to the bivariate objective function $f(\boldsymbol{x}) = 6.4x_1x_2$ *(bottom)* and triangular fuzzy input intervals $\tilde{x}_1$ and $\tilde{x}_2$ *(top and left)* to obtain the fuzzy output interval $\tilde{y}$ *(right)*. Adapted from [Kli06].

```
1 function ỹ = alternativeFuzzyExtensionPrinciple(m, x̃_1, ..., x̃_d)
2   for j = 0,...,m do
3     α_j ← j/m
4     for t = 1,...,d do Compute (x̃_t)_{α_j} = [a_{j,t}, b_{j,t}]
5     Ω_{α_j} ← (x̃_1)_{α_j} × ··· × (x̃_d)_{α_j} = [a_j, b_j]
6     Solve min_{x∈Ω_{α_j}} f(x) and max_{x∈Ω_{α_j}} f(x)
7     (ỹ)_{α_j} = [c_j, d_j] ← [min_{x∈Ω_{α_j}} f(x), max_{x∈Ω_{α_j}} f(x)]
8   D ← {(c_j, α_j) | j = 0, 1, ..., m} ∪ {(d_j, α_j) | j = m, m−1, ..., 0}
9   μ_ỹ ← Piecewise linear interpolant of D        ⇝ extend to X by zero
```

ALGORITHM 5.1 Alternative fuzzy extension principle. Inputs are the number of α segments to use as discretization and the d fuzzy intervals $\tilde{x}_1, \ldots, \tilde{x}_d$ (we have to be able to determine α-cuts of these fuzzy input intervals). The output is an approximation to the output $\tilde{y}$ of the alternative fuzzy extension principle (given by an approximation of its membership function $\mu_{\tilde{y}}$).

differentiable and can thus not be optimized well with gradient-based optimization methods. Second, global polynomials are only suitable for Clenshaw–Curtis grids (Chebyshev-distributed points) due to Runge's phenomenon, unnecessarily restricting the choice of grid points. Hierarchical B-splines of degree p are $(p-1)$ times continuously differentiable and defined for arbitrary point distributions, eliminating both drawbacks simultaneously.

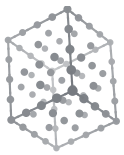
Methodology. Given a sparse grid Ω^s, which may be regular or spatially adaptive, we compute three solutions of the alternative fuzzy extension principle as follows:

- First, we replace f in Alg. 5.1 with the sparse grid interpolant $f^{s,p}$ on Ω^s using modified hierarchical not-a-knot B-splines $\varphi_{\ell,i}^{p,\text{nak,mod}}$ of cubic degree ($p = 3$). For solving the optimization problems over $f^{s,p}$ in Alg. 5.1, we use the globalized version of the method of gradient descent as described in Sec. 5.2.2 using 100 initial points. The resulting fuzzy output interval is denoted by $\tilde{y}^{s,p}$.
- Second, we replace f in Alg. 5.1 with the sparse grid interpolant $f^{s,1}$ on Ω^s using modified piecewise linear basis functions. For solving the optimization problems over $f^{s,1}$ in Alg. 5.1, we use a multi-start version of the Nelder–Mead method as described in Sec. 5.2.2 using 100 initial simplices[8]. The resulting fuzzy output interval corresponds to Klimke's method and is denoted by $\tilde{y}^{s,1}$.
- Third, for comparison, we solve Alg. 5.1 for the actual objective function f. For solving the optimization problems over f, we use a multi-start version of the Nelder–Mead method as described in Sec. 5.2.2 using 1000 initial simplices and 2 000 000 allowed evaluations of f. The resulting fuzzy output interval is denoted by $\tilde{y}^{\text{ref}}$ *(reference solution)*.

In the following, we fix the number of α segments in Alg. 5.1 as $m = 100$. As fuzzy input intervals $\tilde{x}_t$, $t = 1, \ldots, d$, we use the trapezoidal fuzzy interval with 0-cut $[0.125, 0.625]$ and 1-cut $[0.25, 0.375]$ if t is odd and the quasi-Gaussian fuzzy number with mean 0.5, standard deviation 0.125, and 0-cut $[0.125, 0.875]$ if t is even.

Convergence of fuzzy intervals on regular sparse grids. As an example, Fig. 5.15 shows the convergence of the fuzzy output intervals $\tilde{y}^{s,p}$ and $\tilde{y}^{s,1}$ obtained by the interpolation of the bivariate Alp02 function on regular sparse grids $\Omega^s = \Omega^s_{n,d}$ to the reference solution $\tilde{y}^{\text{ref}}$. Already for $n = 4$, the B-spline approximation is better than the piecewise linear approximation. For $n = 5$, no difference is visible anymore between $\tilde{y}^{s,p}$ and $\tilde{y}^{\text{ref}}$, while $\tilde{y}^{s,1}$ still clearly deviates from $\tilde{y}^{\text{ref}}$.

[8] The Nelder–Mead method does not require an initial point, but an initial simplex. The method is a hybrid between global and local optimization. If the initial simplex is chosen badly, Nelder–Mead may get stuck in local minima. Hence, we restart the algorithm for different initial simplices.



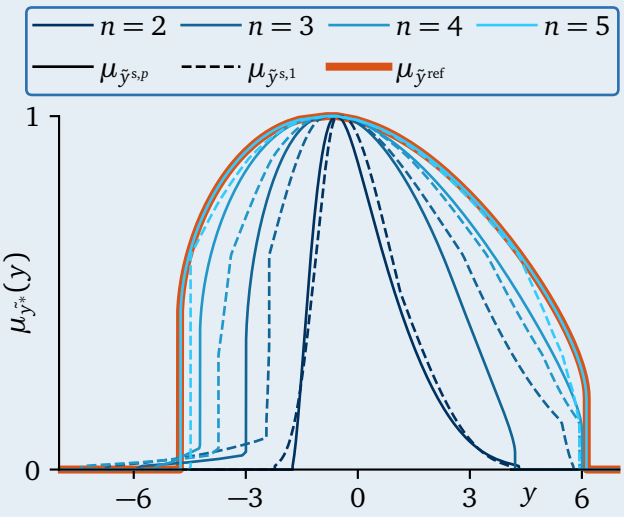


FIGURE 5.15
Convergence of the membership functions of the fuzzy output intervals $\tilde{y}^{s,p}$ (*solid lines,* modified hierarchical cubic not-a-knot B-splines) and $\tilde{y}^{s,1}$ (*dashed,* modified hierarchical hat functions) to the reference solution $\tilde{y}^{ref}$ *(red)* for the bivariate Alp02 function using regular sparse grids of level $n = 2, \ldots, 5$.

In Fig. 5.16, we study the convergence of the relative L^2 errors

$$e^{s,*} := \frac{\|\mu_{\tilde{y}^{ref}} - \mu_{\tilde{y}^{s,*}}\|_{L^2}}{\|\mu_{\tilde{y}^{ref}}\|_{L^2}}, \quad * \in \{1, p\}. \tag{5.10}$$

of the membership functions ("fuzzy errors"). The Alp02 ($d = 2$) errors that correspond to Fig. 5.15 are shown in green in the left-most plot of Fig. 5.16. For the Bra02, GoP, and Alp02 functions, B-spline surrogates achieve dramatic improvements over the hat function surrogates in the bivariate case. For the bivariate Ack function, B-splines yield an error that is still an order of magnitude smaller than the error of hat functions. Just little or even no improvement can be seen for the functions Sch06 and Sch22 with discontinuous derivatives or higher dimensionalities $d \geq 4$.

Fuzzy Novak–Ritter method. We want to employ spatial adaptivity to improve the results of regular sparse grids. To this end, we modify the Novak–Ritter criterion to create a grid generation method that is tailored for the fuzzy extension principle, resulting in Alg. 5.2. Its main idea is to generate more points near the optima of the fuzzy extension principle (Alg. 5.1) than in other regions of $[\mathbf{0}, \mathbf{1}]$. Therefore, we apply the Novak–Ritter criterion twice to every α level $\alpha_j = \frac{j}{m}$ $(j = 0, \ldots, m)$, once for the minimum and once for the maximum. For all α_j, the points to be refined are collected in a set. If a point is selected multiple times for different α_j, it is refined only once. In addition, we enlarge the search domain Ω_{α_j} by 10 %, since the minimum or the maximum might be near the boundary Ω_{α_j} and since the points to be inserted might not be close to the points to be refined. We ensure that the size of Ω_{α_j} is at least 0.05 in every coordinate direction. The remaining experiments use $\gamma = 0.1$ as adaptivity.

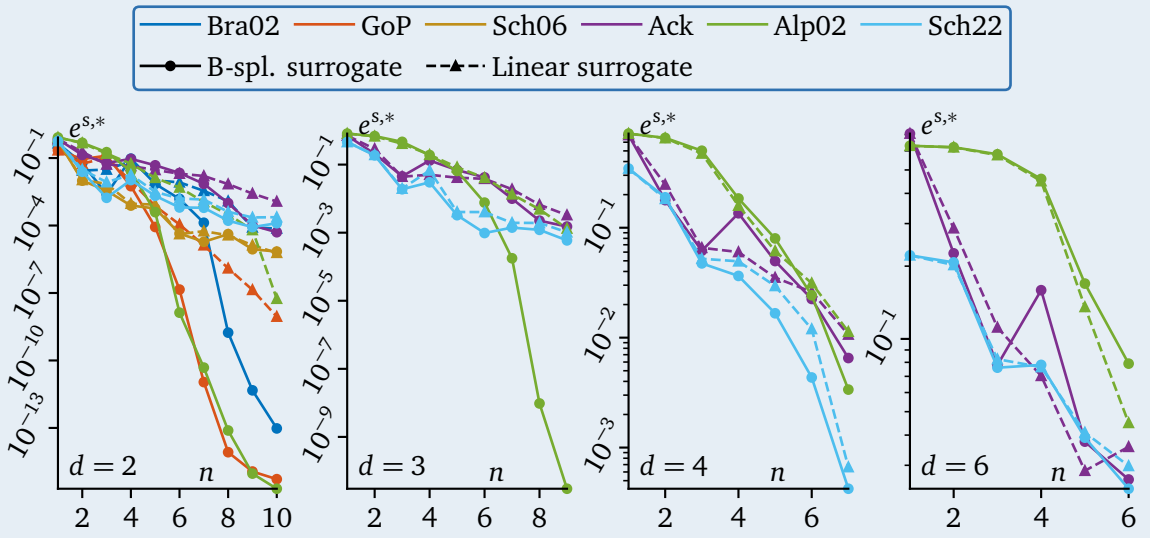
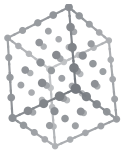


FIGURE 5.16 Fuzzy errors $e^{s,*} := \|\mu_{\tilde{y}^{\text{ref}}} - \mu_{\tilde{y}^{s,*}}\|_{L^2} / \|\mu_{\tilde{y}^{\text{ref}}}\|_{L^2}$ for regular sparse grids $\Omega^s_{n,d}$ and different objective functions f *(colors)* over the level n of the sparse grid.

```
1  function K = fuzzyNovakRitterMethod(f, γ, m, K, x̃_1, ..., x̃_d)
2    for (ℓ,i) ∈ K do d_{ℓ,i} ← 0                    ⇝ degrees (number of refinements)
3    while |K| < N_max do
4      R ← ∅
5      for j = 0,...,m do
6        α_j ← j/m
7        for t = 1,...,d do
8          [a_{j,t}, b_{j,t}] ← (x̃_t)_{α_j}          ⇝ determine α_j-cut
9          if b_{j,t} − a_{j,t} < 0.05 then          ⇝ ensure minimal size of 0.05
10           (a_{j,t}, b_{j,t}) ← ((a_{j,t}+b_{j,t})/2 − 0.025, (a_{j,t}+b_{j,t})/2 + 0.025)
11         (a_{j,t}, b_{j,t}) ← (a_{j,t} − 0.05(b_{j,t}−a_{j,t}), b_{j,t} + 0.05(b_{j,t}−a_{j,t}))   ⇝ enlarge by 10 %
12       K_j ← {(ℓ,i) ∈ K | x_{ℓ,i} ∈ [a_j, b_j] ∩ [0,1]}   ⇝ set of feasible points ([a_j, b_j] = Ω_{α_j})
13       for (ℓ,i) ∈ K_j do r_{ℓ,i} ← |{(ℓ',i') ∈ K_j | f(x_{ℓ',i'}) ≤ f(x_{ℓ,i})}|   ⇝ ranks
14       (ℓ*,i*) ← arg min_{(ℓ,i)∈K_j} [(r_{ℓ,i}+1)^γ (‖ℓ‖_1 + d_{ℓ,i} + 1)^{1−γ}]   ⇝ for minimum
15       (ℓ**,i**) ← arg min_{(ℓ,i)∈K_j} [(|K_j| − r_{ℓ,i} + 2)^γ (‖ℓ‖_1 + d_{ℓ,i} + 1)^{1−γ}]   ⇝ for maximum
16       R ← R ∪ {(ℓ*,i*), (ℓ**,i**)}
17     Refine all points in K that are in R
18     for (ℓ,i) ∈ R do d_{ℓ,i} ← d_{ℓ,i} + 1
```

ALGORITHM 5.2 Fuzzy Novak–Ritter method to generate spatially adaptive sparse grids for the fuzzy extension principle. Inputs are the objective function f, the adaptivity parameter $\gamma \in [0,1]$, the number of α segments, the initial sparse grid K as a set of level-index pairs, and the the d fuzzy intervals $\tilde{x}_1, \ldots, \tilde{x}_d$. The output is the spatially adaptive sparse grid K.



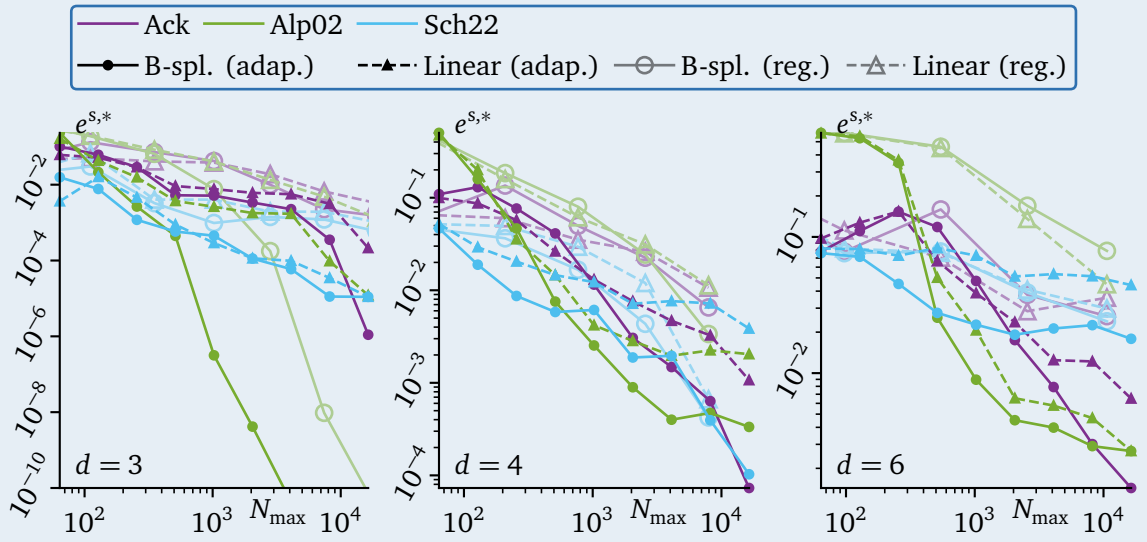


FIGURE 5.17 Fuzzy errors $e^{s,*} := \|\mu_{\tilde{y}^{\text{ref}}} - \mu_{\tilde{y}^{s,*}}\|_{L^2} / \|\mu_{\tilde{y}^{\text{ref}}}\|_{L^2}$ for spatially adaptive sparse grids Ω^s *(solid markers)* and different objective functions f *(colors)* over the number $N_{\max}$ of objective function evaluations. For comparison, the results of Fig. 5.16 for regular sparse grids are repeated *(hollow markers)*.

Convergence of fuzzy intervals on spatially adaptive sparse grids. As we can see in Fig. 5.17, the spatially adaptive sparse grids generated by the fuzzy Novak–Ritter method improve results significantly for both cubic B-spline and piecewise linear surrogates. However, the performance of the B-spline surrogates benefits more from the spatial adaptivity. Even for higher-dimensional settings such as $d = 6$, the spatial adaptivity helps to decrease the errors by one order of magnitude. For instance, for the Ack function in six variables, we can achieve an error of 2.6 % with a budget of 10 000 objective function evaluations (grid points) on regular sparse grids. With the same budget and with spatial adaptivity, the error drops below 0.25 %. Conversely, to achieve the same error as in the regular case (2.6 %), only 1600 evaluations are needed for spatially adaptive grids.

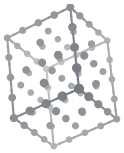
Application 1: Topology Optimization

> *Money. A social life. A shave.*
> *A Ph.D. student needs not such things.*
>
> — Mike Slackenerny (PHD Comics[1])

Now, we want to investigate the first real-world application, which is the field of topology optimization. The classical and widely-used method in engineering is shape optimization, where the shape of a component (parametrized by $\boldsymbol{x} \in \mathbb{R}^d$) has to be determined such that some objective function value $f(\boldsymbol{x})$ is optimal, i.e., minimal or maximal. For example, a bridge over a valley can be built in the shape of a parabolic arc. The task of shape optimization is then to choose the coefficients of the parabola such that the bridge's stability is maximized, possibly with the constraint that the volume occupied by the bridge does not exceed a certain value (to save construction costs) or that the size of the resulting passage meets some size requirements (e.g., at least 20 m wide and 6 m tall).

However, the framework of shape optimization unnecessarily prescribes the topology of the shapes in the search space [All16]. In the bridge example, it may well be that a

[1] `http://phdcomics.com/comics/archive.php?comicid=40`



viaduct-type bridge with three arcs instead of one is more stable while occupying less volume. We are not able to find such a bridge with shape optimization in the previous example, as we have restricted the search space to single-arc bridges. This issue is resolved by the more sophisticated framework of topology optimization, where the topology[2] is not given by the user, but chosen by the optimization algorithm (in a hopefully optimal way), rendering topology optimization a key area of simulation technology.

Recently, B-splines have been used for shape optimization [Mar16] and topology optimization [Qia13; Zha17]. Sparse grids have been employed for topology optimization [Hüb14] as well. In this chapter, we want to combine these two numerical tools, which have been used in isolation until now, to perform topology optimization using B-splines on sparse grids. The two most common approaches for topology optimization are the *level-set method* and the *homogenization method* [All16]. The level-set method describes the boundary of the object as the zero level set $\psi^{-1}(0)$ of a function $\psi : \tilde{\Omega} \to \mathbb{R}$ *(level-set function)* and uses a partial differential equation (PDE) to iteratively transport this function and, consequently, the object's boundary [All04]. However, we want to focus on the second method: the method of homogenization.

This chapter is structured as follows: Section 6.1 explains the homogenization method. In Sec. 6.2, we discuss the details of applying B-splines on sparse grids to this method. We set up different micro-cell models and scenarios in Sec. 6.3, before reviewing numerical results in Sec. 6.4. The results in this chapter have been obtained in collaboration with Prof. Dr. Michael Stingl and Daniel Hübner (both FAU Erlangen-Nürnberg, Germany). The author of this thesis contributed the parts related to interpolation and sparse grids, while the collaborators at FAU studied the engineering and application parts of the joint project (for example, they provided optimization scenarios and assessed the quality of the results).

6.1 Homogenization and the Two-Scale Approach

We roughly follow the presentation given in [Hüb14; Vale14; Vale16]. The necessary notation is summarized in Tab. 6.1.

IN THIS SECTION

[2] Two objects are considered "topologically different" if their numbers of "holes" differ. This stems from the fact that in the field of mathematical topology, the *genus* (i.e., the number of holes) of a topological space is a *topological invariant,* i.e., the genus is invariant under homeomorphism. If the genera of two topological spaces differ, then they cannot be homeomorphic and are thus considered topologically different.

$\tilde{\Omega}$	Object domain	$\boldsymbol{F}$	External force	$\tilde{\varrho}$	Global density fcn.
$\tilde{d}$	#dimensions of $\tilde{\Omega}$	$\boldsymbol{u}$	Displacement fcn.	ϱ	Micro-cell density fcn.
d	#micro-cell param.	$J(\tilde{\varrho})$	Compliance fcn.	ϱ^*	Density bound
$\boldsymbol{x}^{(q)}$	Micro-cell param.	$\mathrm{vol}(\tilde{\Omega})$	Total volume	$\boldsymbol{E}$	Elasticity tensor
		$\mathrm{vol}_{\tilde{\varrho}}(\tilde{\Omega})$	Volume w.r.t. $\tilde{\varrho}$	$\boldsymbol{R}$	Cholesky factor

TABLE 6.1 Glossary of the notation for topology optimization.

6.1.1 Homogenization

Density function. Let $\tilde{\Omega} \subseteq \mathbb{R}^{\tilde{d}}$ be the object domain.[3] Usually, we assume $\tilde{d} = 2$ or $\tilde{d} = 3$, although the method can be generalized to arbitrary dimensionalities $\tilde{d} \in \mathbb{N}$. Shapes and topologies are described by *density functions* $\tilde{\varrho} : \tilde{\Omega} \to [0,1]$. The function values $\tilde{\varrho}(\tilde{\boldsymbol{x}}) \in [0,1]$ tell if $\tilde{\boldsymbol{x}}$ is contained in the object (value of one) or not (value of zero). The *homogenization* approach also allows values between zero and one, giving the physical density of the material in $\tilde{\boldsymbol{x}}$.

Optimization of compliance values. Furthermore, for every density function $\tilde{\varrho}$, let $J(\tilde{\varrho})$ be an objective function value. In our setting, which is shown in Fig. 6.1, we exert a force $\boldsymbol{F}$ on the object, measure the resulting deformation, and compute the *compliance* (i.e., the inverse of the stiffness) as the objective function value $J(\tilde{\varrho})$:

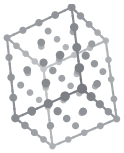
$$J(\tilde{\varrho}) = \int_{\tilde{\Omega}} \boldsymbol{F}^{\mathrm{T}} \boldsymbol{u}_{\tilde{\varrho}}(\tilde{\boldsymbol{x}}) \, \mathrm{d}\tilde{\boldsymbol{x}}, \tag{6.1}$$

where the *displacement function* $\boldsymbol{u}_{\tilde{\varrho}} : \tilde{\Omega} \to \mathbb{R}^{\tilde{d}}$ depends on the density [Hüb14]. We want to find the density function with the minimal compliance value:

$$\min_{\tilde{\varrho}} J(\tilde{\varrho}). \tag{6.2}$$

If we do not impose additional conditions, then there are often uninteresting trivial solutions. For example, choosing $\tilde{\varrho} :\equiv 1$ (i.e., filling the entire domain $\tilde{\Omega}$ with material) usually leads to the topology with the highest stiffness and, thus, the smallest displacement

[3] We use tildes to denote variables and quantities that correspond to the object domain $\tilde{\Omega}$ (e.g., $\tilde{\boldsymbol{x}}$ is a point in $\tilde{\Omega}$). In contrast, variables without a tilde will correspond to the sparse grid domain $[\boldsymbol{0}, \boldsymbol{1}] = [0,1]^d$ (e.g., $\boldsymbol{x}_{\ell,i} \in [\boldsymbol{0}, \boldsymbol{1}]$ will be a sparse grid point).



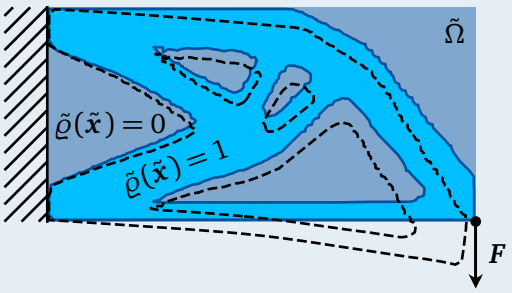


FIGURE 6.1
Example scenario for topology optimization. An object *(light blue)* is fixed on the left side of the object domain $\tilde{\Omega}$ *(darker blue)* and deformed by a force $\boldsymbol{F}$, resulting in a displaced object *(dashed)*. The density function $\tilde{\varrho}(\tilde{\boldsymbol{x}})$ is one inside the object and zero outside.

and compliance value. Therefore, we introduce the following volume constraint:

$$\frac{\mathrm{vol}_{\tilde{\varrho}}(\tilde{\Omega})}{\mathrm{vol}(\tilde{\Omega})} \leq \varrho^*, \quad \mathrm{vol}_{\tilde{\varrho}}(\tilde{\Omega}) := \int_{\tilde{\Omega}} \tilde{\varrho}(\tilde{\boldsymbol{x}})\,\mathrm{d}\tilde{\boldsymbol{x}}, \quad \mathrm{vol}(\tilde{\Omega}) := \mathrm{vol}_1(\tilde{\Omega}), \tag{6.3}$$

where $\mathrm{vol}(\tilde{\Omega}) = \int_{\tilde{\Omega}} 1\,\mathrm{d}\tilde{\boldsymbol{x}}$ is the volume of the object domain and $\varrho^* \in [0,1]$ is an upper bound on the volume fraction.

6.1.2 Two-Scale Approach

Discretization and two-scale approach. Of course, we cannot solve the problem (6.2) numerically, as there are infinitely many density functions $\tilde{\varrho}$. For simplicity, we assume that $\tilde{\Omega}$ is some hyper-rectangle $[\tilde{\boldsymbol{a}}, \tilde{\boldsymbol{b}}] = [\tilde{a}_1, \tilde{b}_1] \times \cdots \times [\tilde{a}_{\tilde{d}}, \tilde{b}_{\tilde{d}}]$; if it is not, we replace $\tilde{\Omega}$ with its bounding box. The object domain $\tilde{\Omega}$ can then be split into $M_1 \times \cdots \times M_{\tilde{d}}$ equally-sized and axis-aligned sub-hyper-rectangles, which we call *macro-cells* (where $M_1, \ldots, M_{\tilde{d}} \in \mathbb{N}$).

In the *two-scale approach,* we assume the material of the macro-cells to be repetitions of infinitesimally small periodic structures (i.e., identical for each macro-cell), called *micro-cells*. These micro-cells have a specific shape, which is parametrized by d *micro-cell parameters* $x_1, \ldots, x_d$, normalized to values in the unit interval $[0,1]$. For instance, in two dimensions, this shape may be an axis-aligned cross with thicknesses x_1 and x_2, as shown in Fig. 6.2. The choice of a suitable *micro-cell model* (parametrization of the micro-cells) depends on the optimization scenario and has to be done a priori.

Elasticity tensors. Note that while the shape of all micro-cells in one macro-cell is identical, the micro-cell parameters corresponding to different macro-cells differ in general. This enables varying densities in different regions of $\tilde{\Omega}$. We denote the micro-cell parameters corresponding to the q-th macro-cell with $\boldsymbol{x}^{(q)} = (x_q^{(1)}, \ldots, x_q^{(d)}) \in [\boldsymbol{0}, \boldsymbol{1}] = [0,1]^d$, where $q = 1, \ldots, M$ and $M := M_1 \cdots M_{\tilde{d}}$ is the number of macro-cells. With linear elastic-

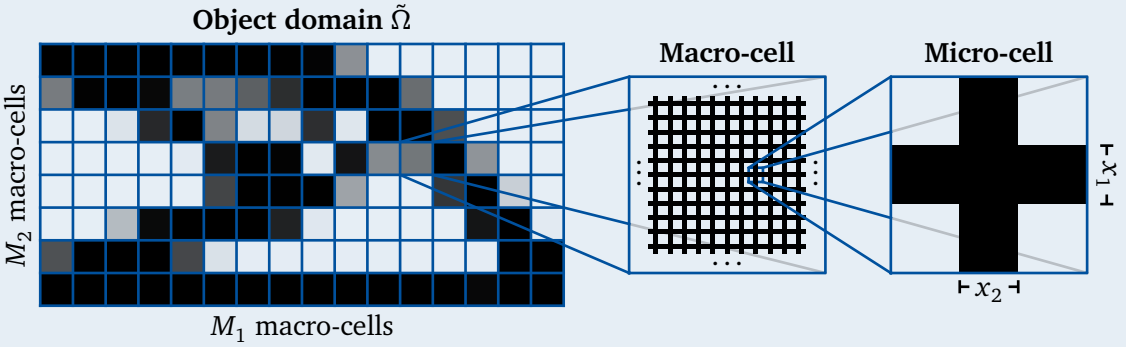


FIGURE 6.2 Two-scale approach to discretize the homogenized topology optimization problem in two dimensions ($\tilde{d} = 2$). *Left:* The object domain $\tilde{\Omega}$ is subdivided into $M_1 \times M_2$ macro-cells, each with its own density *(gray squares)*. *Center:* Every macro-cell is the repetition of infinitesimally small periodic micro-cells. *Right:* The shape of the structure in every micro-cell is described by a micro-cell model with d parameters $x_1, \dots, x_d$. Here, the micro-cell model is a cross with two parameters that represent the thickness of each crossbar.

ity, one can compute so-called *elasticity tensors* $\boldsymbol{E}^{(q)}$, which encode information about the material properties of the different macro-cells. The elasticity tensors can be written as symmetric matrices in $\mathbb{R}^{3\times3}$ (for $\tilde{d} = 2$) or in $\mathbb{R}^{6\times6}$ (for $\tilde{d} = 3$).[4] To simplify the following considerations, we assume that $\tilde{d} = 3$, i.e., $\boldsymbol{E}^{(q)} \in \mathbb{R}^{6\times6}$. The elasticity tensors are usually computed as the solution of a finite element method (FEM) problem *(micro-problem)*. Once all $\boldsymbol{E}^{(q)}$ are known, we can compute the compliance value by solving another FEM problem *(macro-problem),* see [All04] and [Hüb14].

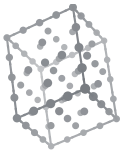
Discretized optimization problem. The new optimization problem emerging from the two-scale discretization process has the form

$$\text{(6.4a)} \qquad \min J(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)}), \quad \boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)} \in [\boldsymbol{0}, \boldsymbol{1}] \quad \text{s.t.} \quad \bar{\varrho}(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)}) \le \varrho^*,$$

$$\text{(6.4b)} \qquad \bar{\varrho}(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)}) := \frac{1}{M} \sum_{q=1}^{M} \varrho^{(q)}(\boldsymbol{x}^{(q)}).$$

Here, $\varrho^{(q)}(\boldsymbol{x}^{(q)}) \in [0, 1]$ is the density of the q-th macro-cell with micro-cell parameter $\boldsymbol{x}^{(q)}$ (i.e., the fraction of material volume of one micro-cell with respect to its total volume) and $\bar{\varrho}(\boldsymbol{x}^{(1)}, \dots, \boldsymbol{x}^{(M)}) \in [0, 1]$ is the resulting total mean density. This discretized optimization problem can now be implemented and solved numerically.

[4] In general, the elasticity tensor is a fourth-order tensor in $\mathbb{R}^{\tilde{d}\times\tilde{d}\times\tilde{d}\times\tilde{d}}$. One can reduce the size of the tensor by exploiting various symmetries [Hüb14] to obtain 6 or 21 stiffness coefficients in two or three dimensions, respectively. These coefficients can then be expressed as a symmetric matrix.



6.2 Approximating Elasticity Tensors

IN THIS SECTION

Optimization process. During the process of solving Eq. (6.4), optimization algorithms typically evaluate the objective function $J(\boldsymbol{x}^{(1)},\dots,\boldsymbol{x}^{(M)})$ iteratively at different *micro-cell parameter combinations* $(\boldsymbol{x}^{(1)},\dots,\boldsymbol{x}^{(M)}) \in (\mathbb{R}^d)^M$. Every evaluation of J corresponds to one solution of a macro-problem. However, to solve the macro-problem, the elasticity tensors $\boldsymbol{E}^{(q)}$ of all M macro-cells need to be known. Hence, in every optimization iteration, it is necessary to solve one macro-problem and M micro-cell problems, all with the FEM. This naive approach has two major drawbacks, which we explain in the following.

6.2.1 Drawbacks of the Naive Approach

Drawback 1: Computation time. First, this approach is computationally infeasible even for simple micro-cell models and optimization scenarios. The computation of a single elasticity tensor usually takes seconds to minutes. All M micro-cell problems per optimization iteration can be solved in parallel without any communication. However, M is typically in the range of thousands and there are thousands or tens of thousands optimization iterations (the optimization problem is $(d \cdot M)$-dimensional!). This implies that the overall computation may still take several days or even weeks to complete.

Drawback 2: Approximation of gradients. Second, most optimization algorithms require gradients of the objective function and of the constraints, i.e.,

$$\frac{\partial}{\partial x_t}\boldsymbol{E}^{(q)}(\boldsymbol{x}^{(q)}), \quad \frac{\partial}{\partial x_t}\varrho^{(q)}(\boldsymbol{x}^{(q)}), \qquad q = 1,\dots,M, \quad t = 1,\dots,d. \tag{6.5}$$

However, in general, both gradients are unavailable and have to be approximated by finite differences. This introduces new error sources and increases the number of elasticity tensors to be evaluated, further slowing down the solution process. Additionally, the number of optimization iterations necessary to achieve convergence might increase if there are discontinuities in the objective function or its gradient. Such discontinuities can already be caused by the inexact solution of the FEM. If we need Hessians or other higher-order derivatives, then the issues even worsen.

6.2.2 B-Splines on Sparse Grids for Topology Optimization

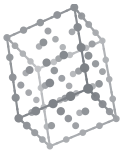
Elasticity tensor function. As a remedy, we replace the costly elasticity tensors with cheap surrogates. If we assume that all macro-cells use the same micro-cell model, the elasticity tensor $\boldsymbol{E}^{(q)}$ of the q-th macro-cell with the parameter $\boldsymbol{x}^{(q)} \in [\mathbf{0}, \mathbf{1}]$ can be written as the value $\boldsymbol{E}(\boldsymbol{x}^{(q)})$ of some function $\boldsymbol{E}\colon [\mathbf{0}, \mathbf{1}] \to \mathbb{R}^{6\times 6}$ (assuming that $\tilde{d} = 3$) at the point $\boldsymbol{x}^{(q)}$. In the following, $\boldsymbol{E}\colon [\mathbf{0}, \mathbf{1}] \to \mathbb{R}^m$ gives $m \in \mathbb{N}$ values from which the symmetric elasticity tensor can be uniquely reconstructed, i.e., $m = 6$ for $\tilde{d} = 2$ and $m = 21$ for $\tilde{d} = 3$. The vector-valued/matrix-valued versions of $\boldsymbol{E}$ will be used interchangeably.

Elasticity tensor surrogate. The idea is to use B-splines on sparse grids to approximate the elasticity tensor function $\boldsymbol{E}$. In contrast to the theoretical framework that we established in Chapters 2 to 4, the function to be interpolated is not scalar-valued, but vector-valued. This means that we have to construct m sparse grid interpolants E_j^{s} for the m components E_j of $\boldsymbol{E}$ $(j = 1, \ldots, m)$. Note that one could generate different spatially adaptive sparse grids for the different components E_j^{s}. However, it is not possible to evaluate only specific entries of $\boldsymbol{E}$ without also evaluating all other entries, which means that we would waste computational resources by selecting only a subset of the calculated entries. Therefore, we use the same grid for all components.

Additionally, we approximate the density $\varrho^{(q)}$ of the q-th macro-cell with a surrogate ϱ^{s} using B-splines on the same sparse grid as for E_j^{s} for reasons of implementation, resulting in $m + 1$ sparse grid interpolants in total. From a theoretical perspective, this is not necessary, since the density can be explicitly calculated with simple formulas for most micro-cell models, independently of evaluations of the elasticity tensor.

Advantages. Our approach has multiple obvious advantages:

- The sparse grid interpolant $\boldsymbol{E}^{\mathrm{s}}$ has to be generated only once in an *offline step* before the optimization algorithm starts. During the optimization *(online phase),* only inexpensive evaluations of $\boldsymbol{E}^{\mathrm{s}}$ are performed, saving much computation time.
- Sparse grids ease the curse of dimensionality, which prohibits conventional full grid interpolation methods if $d > 4$.
- With spatially adaptive sparse grids and a suitable refinement criterion, we can spend more grid points in regions of interest of $\boldsymbol{E}$, e.g., regions with large oscillations.
- By using B-splines as basis functions, the interpolant $\boldsymbol{E}^{\mathrm{s}}$ will be more accurate than with piecewise linear basis functions. In addition, we can calculate its derivatives $\frac{\partial}{\partial x_t}\boldsymbol{E}^{\mathrm{s}}(\boldsymbol{x}^{(q)})$ fast and explicitly, accelerating the speed of convergence of the optimizer.



6.2.3 Cholesky Factor Interpolation

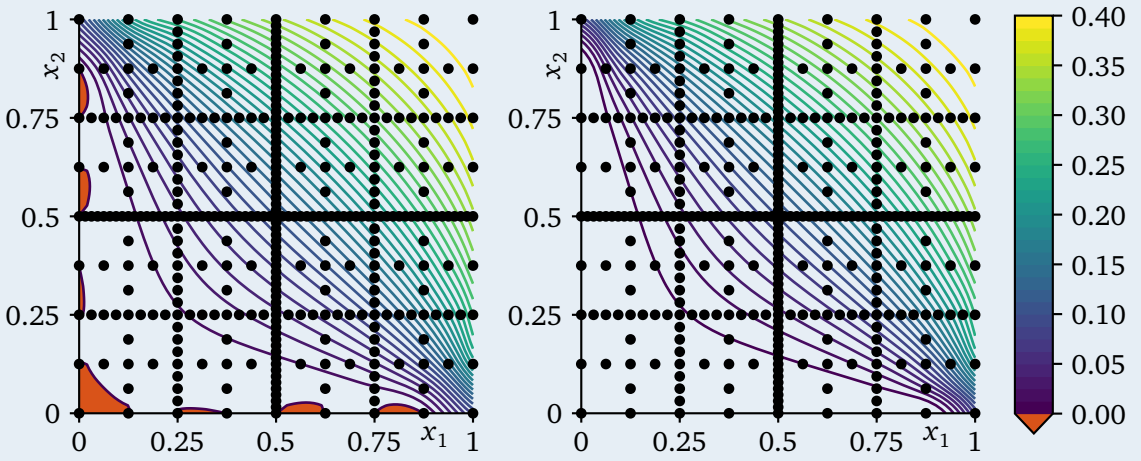
Positive definiteness of elasticity tensors. Unfortunately, just replacing elasticity tensors with B-spline surrogates often does not lead to correct results in practice. Experiments show that for only for some sparse grids, the optimization algorithm converges to an optimal point [Vale16]. The optimization algorithm crashes for most spatially adaptive grids, not being able to find any meaningful optimum. The root of the problem proves to be that the interpolated elasticity tensors $\boldsymbol{E}^{\mathrm{s}}(\boldsymbol{x})$ are not positive definite for specific micro-cell parameters $\boldsymbol{x} \in [\mathbf{0}, \mathbf{1}]$. However, indefinite or even negative definite tensors $\boldsymbol{E}^{\mathrm{s}}$ would mimic unphysical behavior.[5] Hence, it is imperative for the optimization process that the interpolated elasticity tensors are symmetric positive definite (SPD).

Positive definiteness of sparse grid interpolants. Interpolation on sparse grids per se does not preserve positive definiteness. A counterexample is shown in Fig. 6.3A, which displays the minimal eigenvalue of the elasticity tensor surrogate resulting from interpolation on a regular sparse grid. As the positivity of the diagonal is a necessary condition for positive definiteness, small oscillations of the interpolant of some entries already make the whole elasticity tensor non-positive-definite.

These oscillations are more likely to occur near the boundary of the domain $[\mathbf{0}, \mathbf{1}]$, such that there are larger regions where the interpolated tensor is not positive definite anymore. The reason is two-fold: First, sparse grids without boundary points are notoriously biased towards the center of the domain, as they place only few points near the boundary [Pfl10]. This leads to a loss of interpolation accuracy near the boundary when compared to the center of $[\mathbf{0}, \mathbf{1}]$. Second, both the minimal eigenvalue of $\boldsymbol{E}(\boldsymbol{x})$ and the norm of its gradient with respect to $\boldsymbol{x}$ are small near $x_1 = 0$ or $x_2 = 0$. Consequently, the "surface" of the minimal eigenvalue function is rather flat in these regions and almost vanishes, facilitating the existence of negative eigenvalues of surrogate functions $\boldsymbol{E}^{\mathrm{s}}(\boldsymbol{x})$.

Note that for most micro-cell models, the optimization algorithm often evaluates the objective function $J(\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(M)})$ at micro-cell parameter combinations for which many of the points $\boldsymbol{x}^{(j)}$ are near the boundary of $[\mathbf{0}, \mathbf{1}]$. This is because many of the macro-cells will either be empty or fully filled with material, which usually corresponds to micro-cell parameters near zero or one, respectively. Thus, $\boldsymbol{E}^{\mathrm{s}}$ is frequently evaluated in the regions of indefiniteness, which further worsens the issue.

[5] In the scalar case, this is analogous to Hooke's law for linear springs, where the force $F = kx$ needed to displace the end of a spring (fixed at the other end) by x is proportional to x. The proportionality constant k (which corresponds to the elasticity tensor) has to be positive.



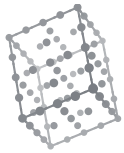
A Minimal eigenvalue of $\boldsymbol{E}^{s}(\boldsymbol{x})$ **B** Minimal eigenvalue of $\boldsymbol{E}^{\text{chol},s}(\boldsymbol{x})$

FIGURE 6.3 Minimal eigenvalue *(colored contour lines)* of elasticity tensor surrogates for the 2D cross model ($\tilde{d} = 2$, $d = 2$) and cubic hierarchical B-splines $\varphi^{p}_{\ell,\boldsymbol{i}}$ ($p = 3$) on the regular sparse grid $\Omega^{s(b)}_{n,d}$ *(dots)* with $n = 6$ and $b = 4$. *Left:* The minimal eigenvalue of $\boldsymbol{E}^{s}(\boldsymbol{x})$ becomes negative in some regions *(red areas)* of the domain $[\boldsymbol{0}, \boldsymbol{1}]$, indicating that $\boldsymbol{E}^{s}(\boldsymbol{x})$ is not positive definite. *Right:* The minimal eigenvalue of $\boldsymbol{E}^{\text{chol},s}(\boldsymbol{x})$ is non-negative in the whole domain $[\boldsymbol{0}, \boldsymbol{1}]$.

Positivity-preserving methods. Even in one dimension, it cannot be guaranteed that the interpolant of positive data remains positive, which is a key problem in the estimation of probability densities [Pfl10; Gri10; Fra16]. Just clamping the interpolated values via $\max(\cdot, 0)$ does not help: In our application, the tensor may still be indefinite; additionally, the calculated gradients of the interpolants do not match the actual gradients anymore. In density estimation, clamping a density-like function changes its integral, making it necessary to recalculate its normalization constant [Fra17].

One possible workaround is to apply a continuous injective transformation $T : \mathbb{R}_{>0} \to \mathbb{R}$ on the positive values (e.g., ln), then interpolate the resulting values, and finally apply the inverse transformation $T^{-1} : \mathbb{R} \to \mathbb{R}_{>0}$ on the interpolated values (e.g., exp).[6] For the piecewise linear hierarchical basis, another approach has been developed recently [Fra17], maintaining the positivity by inserting additional sparse grid points. In the context of spline approximation, positivity-preserving approximation schemes based on so-called quasi-interpolation are known [Höl13]. For our application, for which we need to preserve positive definiteness, it is conceivable that one could apply these positivity-preserving methods in the eigenspace, interpolating the positive eigenvalues.

[6] Formally, the inverse function $T^{-1} : T(\mathbb{R}_{>0}) \to \mathbb{R}_{>0}$ is only defined on the image $T(\mathbb{R}_{>0})$ of T, which might not be the whole real line. However, we assume that T^{-1} can be "reasonably" extended to $\mathbb{R}$ (e.g., $T := \sqrt{\cdot}$ and $T^{-1} = (\cdot)^2$).



Interpolation of Cholesky factors. Instead, we pursue a different, more canonical approach based on Cholesky factorization:

PROPOSITION 6.1 (Cholesky factorization)
For every SPD matrix $\boldsymbol{E} \in \mathbb{R}^{6\times 6}$, there is a unique upper triangular matrix $\boldsymbol{R} \in \mathbb{R}^{6\times 6}$ with positive diagonal entries such that

$$\boldsymbol{E} = \boldsymbol{R}^{\mathrm{T}}\boldsymbol{R}. \tag{6.6}$$

PROOF See [Ben24] or [Fre07]. ∎

In one dimension, the Cholesky factorization is equivalent to the application of a transformation T as above by choosing $T := \sqrt{\cdot}$ and $T^{-1} = (\cdot)^2$. Our approach is as follows:

1. Define $\boldsymbol{R}\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{6\times 6}$ as the Cholesky factor of $\boldsymbol{E}\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{6\times 6}$, i.e., $\boldsymbol{E}(\boldsymbol{x}) = \boldsymbol{R}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{R}(\boldsymbol{x})$ for all $\boldsymbol{x} \in [\mathbf{0},\mathbf{1}]$.
2. During the grid generation (offline phase), evaluate $\boldsymbol{E}(\boldsymbol{x}_{\ell,i})$ at the grid points $\boldsymbol{x}_{\ell,i}$, compute the Cholesky factors $\boldsymbol{R}(\boldsymbol{x}_{\ell,i})$ of $\boldsymbol{E}(\boldsymbol{x}_{\ell,i})$, and interpolate them instead of the elasticity tensors to obtain an interpolant $\boldsymbol{R}^{\mathrm{s}}\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{6\times 6}$.
3. During the optimization (online phase), every time the value $\boldsymbol{E}(\boldsymbol{x})$ of an elasticity tensor is needed, the interpolant $\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})$ is evaluated and we return

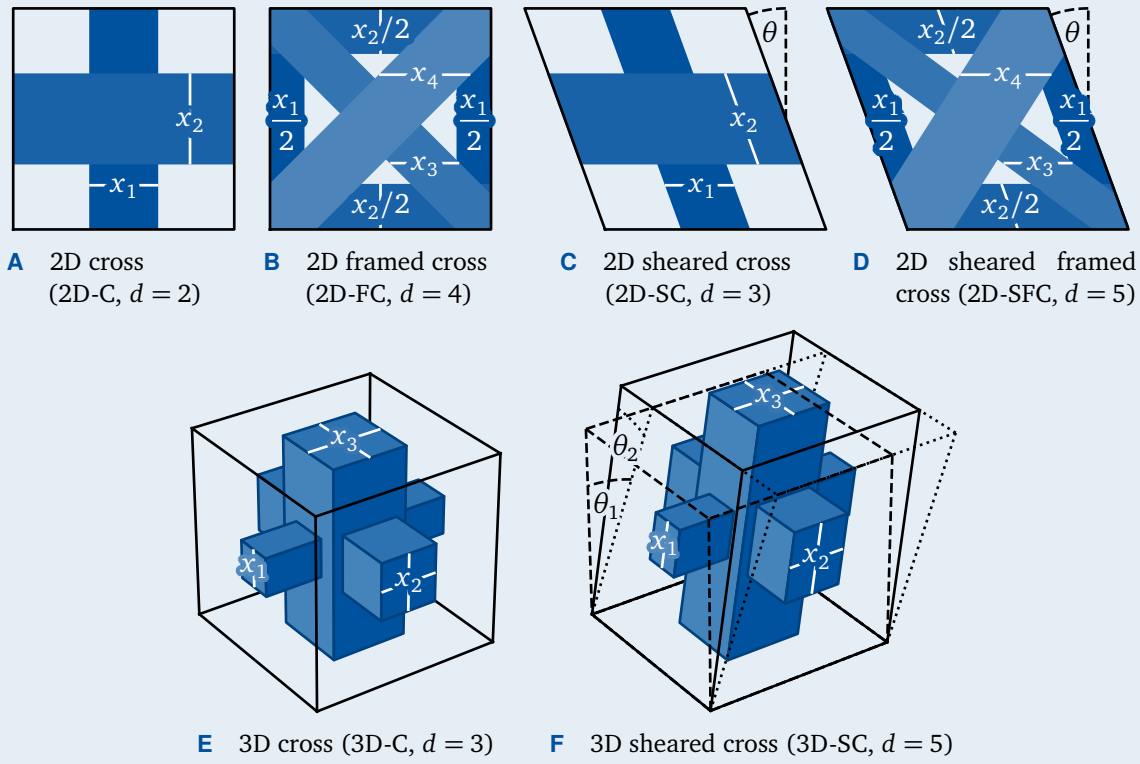
$$\boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x}) := \boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})^{\mathrm{T}}\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x}). \tag{6.7}$$

Advantages of Cholesky factor interpolation. As shown in Fig. 6.3B, the resulting elasticity tensor surrogate $\boldsymbol{E}^{\mathrm{chol,s}}$ is positive semidefinite on the whole domain and positive definite almost everywhere: The surrogate $\boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x})$ is singular if and only if $\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})$ is singular, which is in general only the case on a negligible null set in $[\mathbf{0},\mathbf{1}]$.

Another advantage of this approach is that not only the positive definiteness, but also the explicit differentiability of the surrogate $\boldsymbol{E}^{\mathrm{chol,s}}$ is preserved. The gradient can be computed easily and fast with the product rule:

$$\frac{\partial}{\partial x_t}\boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x}) = \boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})^{\mathrm{T}} \cdot \frac{\partial}{\partial x_t}\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x}) + \frac{\partial}{\partial x_t}\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})^{\mathrm{T}} \cdot \boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x}), \quad t = 1,\dots,d, \tag{6.8}$$

where both the sparse grid interpolant $\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})$ and its derivative $\frac{\partial}{\partial x_t}\boldsymbol{R}^{\mathrm{s}}(\boldsymbol{x})$ are known. As discussed above, this is key to the applicability of gradient-based optimization.



A 2D cross (2D-C, $d = 2$)
B 2D framed cross (2D-FC, $d = 4$)
C 2D sheared cross (2D-SC, $d = 3$)
D 2D sheared framed cross (2D-SFC, $d = 5$)
E 3D cross (3D-C, $d = 3$)
F 3D sheared cross (3D-SC, $d = 5$)

FIGURE 6.4 Types of micro-cell models in two dimensions *(top row)* and three dimensions *(bottom row)*.

6.3 Micro-Cell Models and Optimization Scenarios

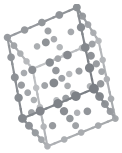
In the following, we present the different micro-cell models and optimization scenarios for which we perform numerical experiments in the next section.

IN THIS SECTION

6.3.1 Micro-Cell Models

We use the various micro-cell models that are depicted in Fig. 6.4. The models differ in the spatial dimensionality $\tilde{d}$ and the number d of micro-cell parameters $\boldsymbol{x} \in [\mathbf{0}, \mathbf{1}] = [0, 1]^d$. Note that the presented models are only some examples. One can easily design complicated micro-cell models with larger numbers of parameters.

Orthogonal (non-sheared) models in two dimensions. The basic component of the four two-dimensional models is a square with a cross (Fig. 6.4A) of two axis-aligned orthogonal bars, whose widths are determined by two micro-cell parameters x_1 and x_2. The micro-cell parameters are ratios of the bar widths to the edge lengths of the micro-cell



(although the actual micro-cells are infinitesimally small). This results in the *cross model*. For the *framed cross model* (Fig. 6.4B), we add a diagonal cross with orthogonal bars of widths x_3 and x_4 (horizontally measured). To simplify the boundary treatment, we shift the contents of the framed cross micro-cell by 50 % of the micro-cell's edge lengths in both directions, such that previous corners of the micro-cell correspond to the new center.

Sheared models in two dimensions. Both of these models can be extended by shearing. The idea is to increase the stability of the resulting macro-structure with respect to forces that act at angles other than 0° and 90° (cross model) or 0°, 90°, and 45° (framed cross model). If we just rotated the crosses in the micro-cells, then the micro-structure would not be periodic. Instead, we shear the whole micro-cell in the horizontal direction, where the shearing angle θ is an additional micro-cell parameter, which gives us another degree of freedom.[7] This results in the *sheared cross model* (Fig. 6.4C) and *sheared framed cross model* (Fig. 6.4D) with three and five micro-parameters each.

Models in three dimensions. The two-dimensional cross model can be transferred to three spatial dimensions by just adding another bar in the new dimension. Each of the three bars has square cross-section with given edge lengths x_1, x_2, or x_3, respectively, resulting in the *3D cross model* with three micro-cell parameters (Fig. 6.4E). By shearing in the two horizontal directions, we obtain two new degrees of freedom θ_1 and θ_2 (shearing angles). The emerging *3D sheared cross model* has five micro-cell parameters (Fig. 6.4F).

6.3.2 Test Scenarios

To benchmark the performance of the new method, we take a subset of the scenarios given in [Vald17], which reviews more than 100 papers on topology optimization to determine the most common test scenarios in the field. The geometry and the boundary conditions of the four scenarios (two for each 2D and 3D) are given in Fig. 6.5 (dimensions in meters). In contrast to [Vald17], we only use single-point loads (i.e., not loads applied to line segments, areas, or volumes) for implementational reasons. The upper bound on the density (see Sec. 6.1.1) is $\varrho^* = 50\,\%$ for the 2D scenarios and $\varrho^* = 10\,\%$ for the 3D scenarios. As in [Sig01] and for reasons of simplicity, we apply a force $\boldsymbol{F}$ with unit value (i.e., $\|\boldsymbol{F}\|_2 = 1\,\mathrm{N}$), and we use a hypothetical material with a Young's modulus (stiffness) of 1 Pa and a Poisson ratio (transversal expansion to axial compression) of 0.3.

[7] To be more precise, the angle θ corresponds to an additional micro-cell parameter x_3 (sheared cross) or x_5 (sheared framed cross) that is determined by normalization from $[-0.35\pi, 0.35\pi]$, i.e., $\theta/(0.7\pi)+1/2$.

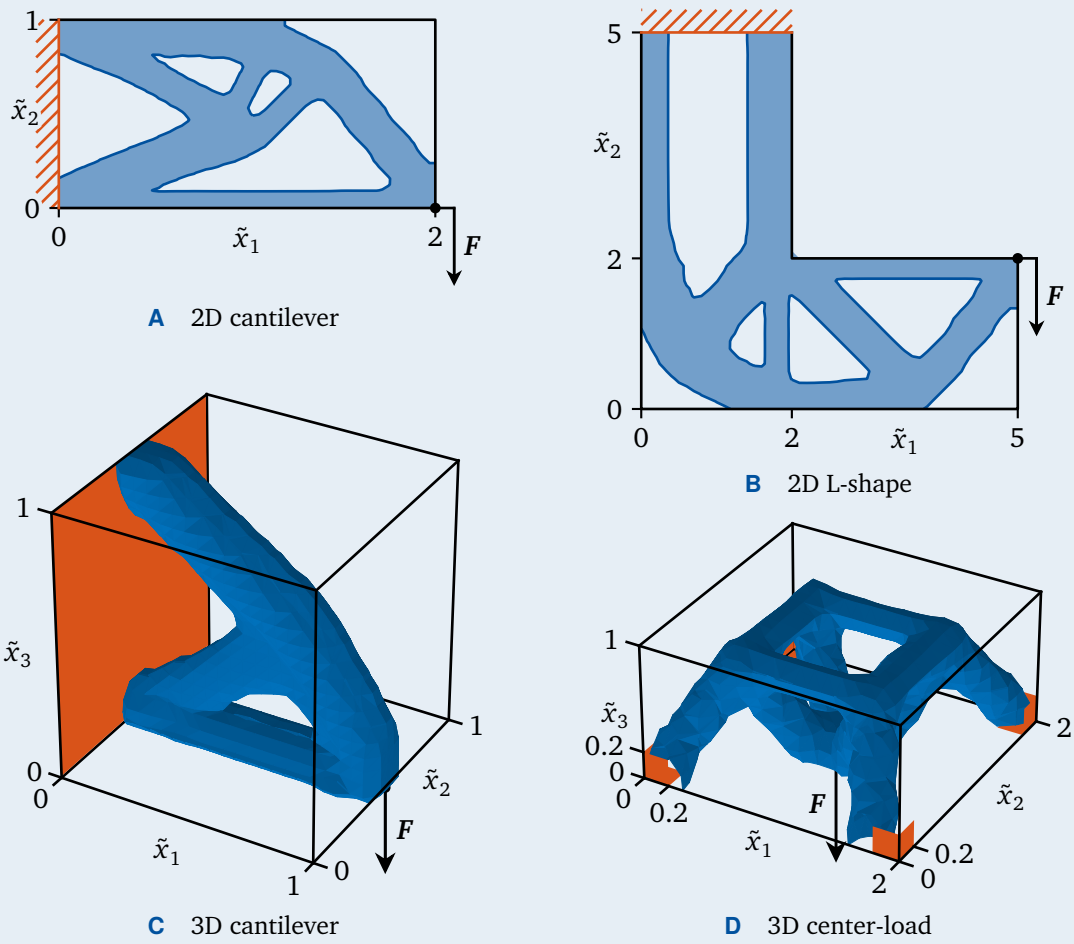


FIGURE 6.5 Test scenarios in topology optimization in two and three spatial dimensions. Shown are the domains $\tilde{\Omega}$, load points, locations of homogeneous Dirichlet boundary conditions *(red)*, and exemplary optimal structures *(blue)*.

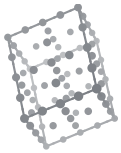
6.4 Implementation and Numerical Results

In this final section of the chapter, we study optimal results of the test scenarios and analyze interpolation errors and optimization results for topology optimization with B-spline surrogates on sparse grids.

IN THIS SECTION

6.4.1 Implementation

In the following, for simplicity, we combine the two functions to be interpolated, i.e., the Cholesky factor $\boldsymbol{R}\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{6\times 6}$ and the micro-cell density $\varrho\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}$, to one single objective function $\boldsymbol{f}\colon [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{m+1}$, from which both functions can be recovered.



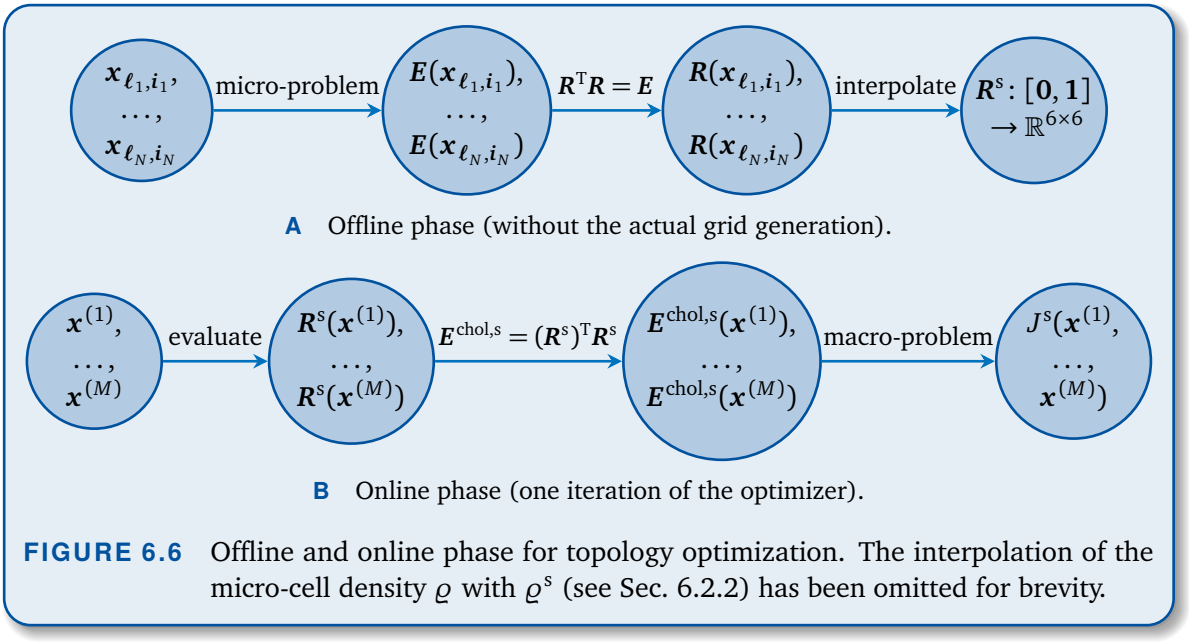


FIGURE 6.6 Offline and online phase for topology optimization. The interpolation of the micro-cell density ϱ with ϱ^s (see Sec. 6.2.2) has been omitted for brevity.

Overview of offline and online phase. Our method is divided into an offline phase and an online phase, both of which are sketched in Fig. 6.6. The offline phase consists of generating the spatially adaptive sparse grid $\Omega^s = \{x_{\ell_k,i_k} \mid k = 1,\ldots,N\}$, solving the corresponding micro-problems, computing the Cholesky factors, and hierarchizing the Cholesky factor entries and micro-cell densities to obtain the sparse grid interpolant f^s. Each optimization iteration of the online phase consists of evaluating the interpolant f^s for each micro-cell parameter $x^{(j)}$ $(j = 1,\ldots,M)$, reconstructing the elasticity tensor $E^{\text{chol,s}}$ from the Cholesky factors R^s,[8] and solving the macro-problem to retrieve the approximated compliance value $J^s(x^{(1)},\ldots,x^{(M)})$. The superscript in J^s indicates that we do not use the exact elasticity tensors E to compute the compliance value, but rather the reconstructed and interpolated tensors $E^{\text{chol,s}}$.

Generation of spatially adaptive sparse grids. We use the classical surplus-based refinement criterion (see, e.g., [Pfl10]) as shown in Alg. 6.1 to generate the spatially adaptive sparse grids. The difference to common surrogate settings is that the objective function $f : [0,1] \to \mathbb{R}^{m+1}$ is vector-valued. As the entries of R cannot be evaluated individually, the adaptivity criterion has to consider all entries at once to avoid performing unnecessary evaluations. We use the surpluses in the piecewise linear hierarchical basis, as their absolute values correlate with the second mixed derivative of the objective function due to Eq. (2.25). The surpluses are combined using the formula $\beta_k := c^T |\alpha_{\ell_k,i_k}|$ (with entry-wise absolute value) and the points with largest β_k are refined.

[8] In addition, the partial derivatives $\partial E^{\text{chol,s}} / \partial x_t$ $(t = 1,\ldots,d)$ are evaluated using Eq. (6.8). This is necessary to employ gradient-based optimization.

```
1 function Ω^s = offlinePhase(f, n, b, c, ℓ_max, κ, N_refine)
2   Ω^s ← Ω_{n,d}^{s(b)}                                  ⇝ initial regular sparse grid
3   while true do
4     N ← |Ω^s|                                           ⇝ number of grid points
5     Let (α_{ℓ_{k'},i_{k'}})_{k'=1,...,N} satisfy ∀_{k=1,...,N} Σ_{k'=1}^{N} α_{ℓ_{k'},i_{k'}} φ^1_{ℓ_{k'},i_{k'}}(x_{ℓ_k,i_k}) = f(x_{ℓ_k,i_k})
6     for k = 1,...,N do β_k ← c^T |α_{ℓ_k,i_k}|          ⇝ combine surpluses to a scalar value
7     K* ← {k = 1,...,N | ∃_{x_{ℓ,i} ∉ Ω^s} x_{ℓ_k,i_k} → x_{ℓ,i}, ||ℓ_k||_∞ < ℓ_max, |β_k| > κ}
8     if K* = ∅ then break                                ⇝ stop when there are no refinable grid points left
9     Refine ≤ N_refine of the points {x_{ℓ_k,i_k} ∈ Ω^s | k ∈ K*} with largest β_k
```

ALGORITHM 6.1 Generation of spatially adaptive sparse grids for topology optimization. Inputs are the objective function $\boldsymbol{f} : [\boldsymbol{0}, \boldsymbol{1}] \to \mathbb{R}^{m+1}$ (combination of the Cholesky factor of the elasticity tensor and the micro-cell density), the level $n \geq d$ and boundary parameter $b \in \mathbb{N}$ of the initial regular sparse grid, the vector $\boldsymbol{c} \in \mathbb{R}^{m+1}$ of coefficients with which the absolute values of the entries of the surpluses are combined, the maximal level $\ell_{\max} \in \mathbb{N}$, the refinement threshold $\kappa \in \mathbb{R}_{>0}$, and the number $N_{\text{refine}} \in \mathbb{N}$ of points to refine in each iteration. Output is the spatially adaptive sparse grid Ω^{s}.

Parameter bounds. In the micro-cell models presented in Sec. 6.3.1, extreme micro-cell parameters near zero or one may cause problems with the resulting elasticity tensors. For instance, many elasticity tensor entries corresponding to the 2D cross model are discontinuous near the lines $x_1 = 1$ or $x_2 = 1$ [Hüb14; Vale14]. This is due to the fact that the micro-cell is completely filled with material on these lines, independent of the other micro-cell parameter. Similar issues occur for the other models and the shearing angles. Hence, we have to restrict the range of the feasible micro-cell parameters, i.e., the sparse grid points $\boldsymbol{x} = \boldsymbol{x}_{\boldsymbol{\ell}_k, \boldsymbol{i}_k}$ are still defined on the unit hyper-cube $[\boldsymbol{0}, \boldsymbol{1}]$, but the actual micro-cell parameters $\bar{\boldsymbol{x}}$ are retrieved by an affine transformation $\bar{\boldsymbol{x}} := \boldsymbol{a} + (\boldsymbol{b} - \boldsymbol{a})\boldsymbol{x}$. For the models in Sec. 6.3.1, we restrict the bar widths to $[0.01, 0.99]$ and the shearing angles to $[-0.35\pi, 0.35\pi]$.

Software, algorithms, and domain discretization. The micro-problems and macro-problems were solved with the FEM software package CFS++ [Kal10].[9] The micro-problems were discretized by dividing the micro-cells into $128 \times 128 = 16\,384$ elements (models in two dimensions) or $16 \times 16 \times 16 = 4096$ elements (models in three dimensions). The macro-domains $\tilde{\Omega}$ were discretized using 32 macro-cells per meter in the 2D cantilever scenario (i.e., $64 \times 32 = 2048$ cells) and 10 macro-cells per meter in the other scenarios (i.e., 1600 cells for the 2D L-shape, 8000 cells for the 3D cantilever, and 4000 cells for the 3D center-load). The generation of the sparse grids (offline phase) was done via a

[9] http://www.lse.uni-erlangen.de/cfs/

MATLAB code, while the evaluation of the interpolants (online phase) was performed by the sparse grid toolbox SG^{++} [Pfl10].[10] For the solution of the emerging optimization problems, a sequential quadratic programming method was employed (see Sec. 5.1.3).

6.4.2 Error Sources

There are multiple sources that contribute to the numerical error of our method:

E1. Discretization of the micro-problem (i.e., the elasticity tensors $\boldsymbol{E}$ are inaccurate)

E2. Sparse grid interpolation (i.e., $\boldsymbol{E}^{\mathrm{s}} \neq \boldsymbol{E}$)

E3. Reconstruction of elasticity tensors with Cholesky factors (i.e., $\boldsymbol{E}^{\mathrm{chol,s}} \neq \boldsymbol{E}^{\mathrm{s}}$)

E4. Discretization of the macro-problem (i.e., the compliance J is inaccurate)

E5. Optimization (i.e., the minimum found by the optimizer is inaccurate or not global)

E6. Floating-point rounding errors (i.e., arithmetical operations are inaccurate)

E6-type errors are always present and will not be analyzed in this chapter. Errors of type E1 and E4 are intrinsic to the homogenization approach and will not be discussed here either. The optimization error E5 has already been discussed in Sec. 5.4.3 for explicit test functions. Therefore, in the remainder of this chapter, we will focus on the analysis of the errors of types E2 and E3, since the interpolation of Cholesky factors is the major new contribution to this application.

6.4.3 Interpolation Error

Spectral interpolation error measure. For the interpolation error E2 and the Cholesky factorization error E3, we cannot simply take the absolute value of the difference of the objective function $\boldsymbol{f} : [\boldsymbol{0}, \boldsymbol{1}] \to \mathbb{R}^{m+1}$ and its surrogate $\boldsymbol{f}^{\mathrm{s}}$, since both are vector-valued. As the micro-cell density ϱ is not affected by the Cholesky factorization, we consider only the elasticity tensor $\boldsymbol{E} : [\boldsymbol{0}, \boldsymbol{1}] \to \mathbb{R}^{6\times 6}$ and its surrogate $\boldsymbol{E}^{\mathrm{chol,s}} : [\boldsymbol{0}, \boldsymbol{1}] \to \mathbb{R}^{6\times 6}$ obtained by Cholesky factorization. To retrieve a scalar error measure, we use the spectral norm

$$\|\boldsymbol{E}(\boldsymbol{x}) - \boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x})\|_2, \quad \boldsymbol{x} \in [\boldsymbol{0}, \boldsymbol{1}], \tag{6.9}$$

i.e., the largest absolute eigenvalue of $\boldsymbol{E}(\boldsymbol{x}) - \boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x})$. However, the choice of the norm is arbitrary, as all matrix norms on $\mathbb{R}^{6\times 6}$ are equivalent to each other.

[10] `http://sgpp.sparsegrids.org/`

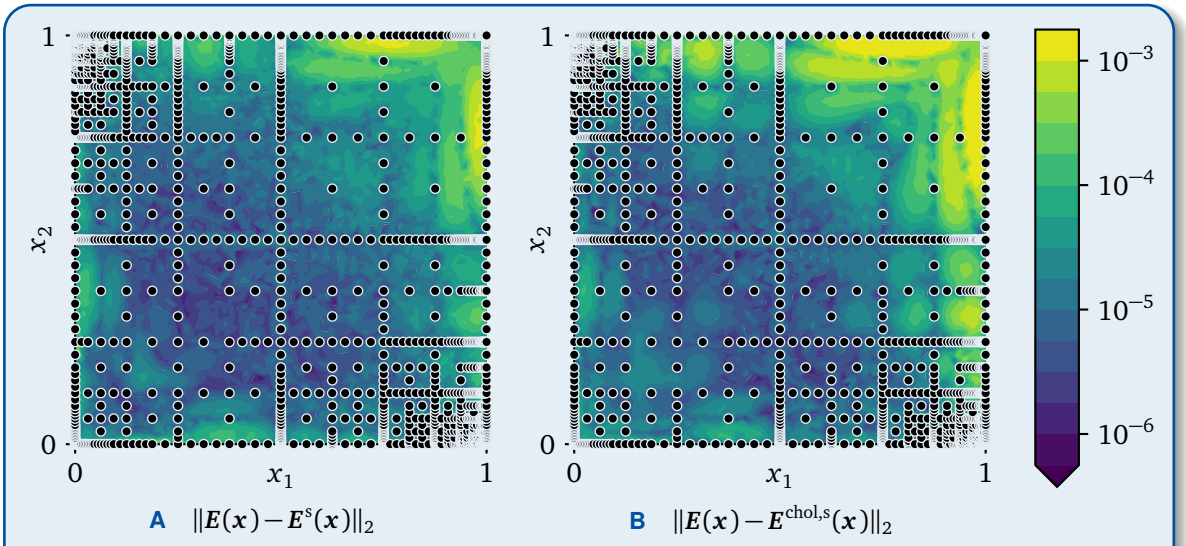


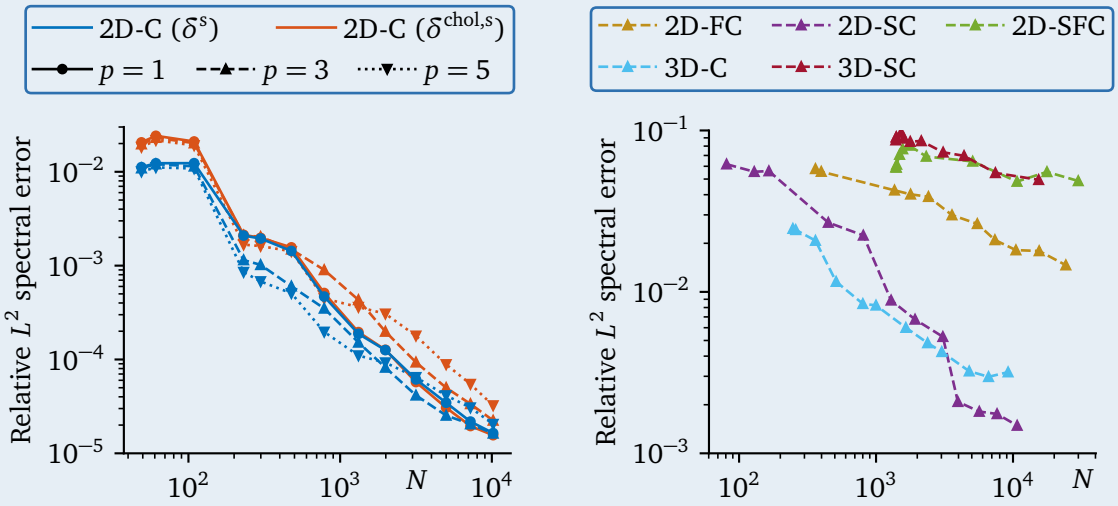
FIGURE 6.7 Pointwise spectral interpolation error for the 2D cross model and cubic B-splines on $N = 1320$ spatially adaptive sparse grid points *(dots)* for the direct elasticity tensor interpolation *(left)* and the Cholesky factor interpolation *(right)*.

Pointwise spectral interpolation error. Figure 6.7 shows the pointwise spectral interpolation error for the 2D cross model and the corresponding spatially adaptive sparse grid generated with the refinement algorithm as explained in Sec. 6.4.1. The above-mentioned discontinuity of elasticity tensor entries near $x_1 = 1$ or $x_2 = 1$ is most severe near the corners $\boldsymbol{x} \in \{(0,1),(1,0)\}$ (cf. Fig. 6.3), as some entries vanish if one of the micro-cell bars has zero width. Hence, most points are placed near these singularity corners.

The left plot (Fig. 6.7A) shows the spectral interpolation error $\|\boldsymbol{E}(\boldsymbol{x})-\boldsymbol{E}^{\mathrm{s}}(\boldsymbol{x})\|_2$ of the direct elasticity tensor interpolant without Cholesky factorization (i.e., error E2). The maximum error is $1.2 \cdot 10^{-3}$, which is attained near the critical lines $x_1 = 1$ or $x_2 = 1$. Note that the mean error over the whole domain $[\mathbf{0},\mathbf{1}]$ is only $4.5 \cdot 10^{-5}$. In the right plot (Fig. 6.7B), the picture changes slightly when looking at the spectral interpolation error $\|\boldsymbol{E}(\boldsymbol{x})-\boldsymbol{E}^{\mathrm{chol,s}}(\boldsymbol{x})\|_2$ of the elasticity tensor resulting from Cholesky factorization (i.e., errors E2 and E3 combined). The maximum error becomes $3.4 \cdot 10^{-3}$, while the mean error increases to $1.1 \cdot 10^{-4}$. We conclude that the Cholesky factorization leads to an increase of interpolation errors by only less than half an order of magnitude.

Convergence of spectral interpolation error. Figure 6.8A shows the convergence of the relative L^2 spectral interpolation errors

$$\varepsilon^{\mathrm{s}} := \frac{\left\| \|\boldsymbol{E}(\cdot)-\boldsymbol{E}^{\mathrm{s}}(\cdot)\|_2 \right\|_{L^2}}{\left\| \|\boldsymbol{E}(\cdot)\|_2 \right\|_{L^2}}, \qquad \varepsilon^{\mathrm{chol,s}} := \frac{\left\| \|\boldsymbol{E}(\cdot)-\boldsymbol{E}^{\mathrm{chol,s}}(\cdot)\|_2 \right\|_{L^2}}{\left\| \|\boldsymbol{E}(\cdot)\|_2 \right\|_{L^2}} \tag{6.10}$$



A ε^s *(blue)* and $\varepsilon^{chol,s}$ *(red)* for the 2D cross model and different degrees.

B $\varepsilon^{chol,s}$ for the other models and $p = 3$.

FIGURE 6.8 Convergence of relative L^2 spectral interpolation errors over the increasing number N of spatially adaptive grid points (i.e., decreasing threshold κ) for the 2D cross model without or with Cholesky factor interpolation and different degrees p *(left)* and for the other models and cubic degree *(right)*.

for the 2D cross model, i.e., the relative L^2 error of the functions depicted in Fig. 6.7. Relative errors of 1 ‰ are already obtained for $N = 200$ grid points. Unfortunately, even for higher B-spline degrees $p > 1$, the order of convergence is only quadratic due to the singularities of the elasticity tensor. This slow convergence does not improve for the other micro-cell models as shown in Figure 6.8B. In fact, the convergence decelerates even more as the number of micro-cell parameters increases. For the 2D sheared cross and 3D cross models with three parameters, the spatially adaptive sparse grid with $N \approx 10\,000$ grid points is able to achieve a relative error of around 3 ‰. However, for the 2D sheared framed cross and 3D sheared cross models with five parameters, only errors of about 5 % are reached for the same grid size.

6.4.4 Optimal Compliance Values and Structures

Optimal compliance values for different micro-cell models. In the following, we use for each micro-cell model a specific spatially adaptive sparse grid with around 10 000 points. The exact grid sizes and other details about the employed sparse grids can be found in Tab. C.2 (located in Appendix C). For hierarchical cubic B-splines ($p = 3$), Tab. 6.2 lists the compliance values $J(\boldsymbol{x}^{opt,*,(1)}, \ldots, \boldsymbol{x}^{opt,*,(M)})$ for each of the four scenarios

Scenario	2D-C	2D-FC	2D-SC	2D-SFC	3D-C	3D-SC
2D cantilever	74.974	70.816	**67.809**	68.602	—	—
2D L-shape	183.68	177.51	**169.60**	174.55	—	—
3D cantilever	—	—	—	—	247.60	**162.59**
3D center-load	—	—	—	—	169.27	**46.171**

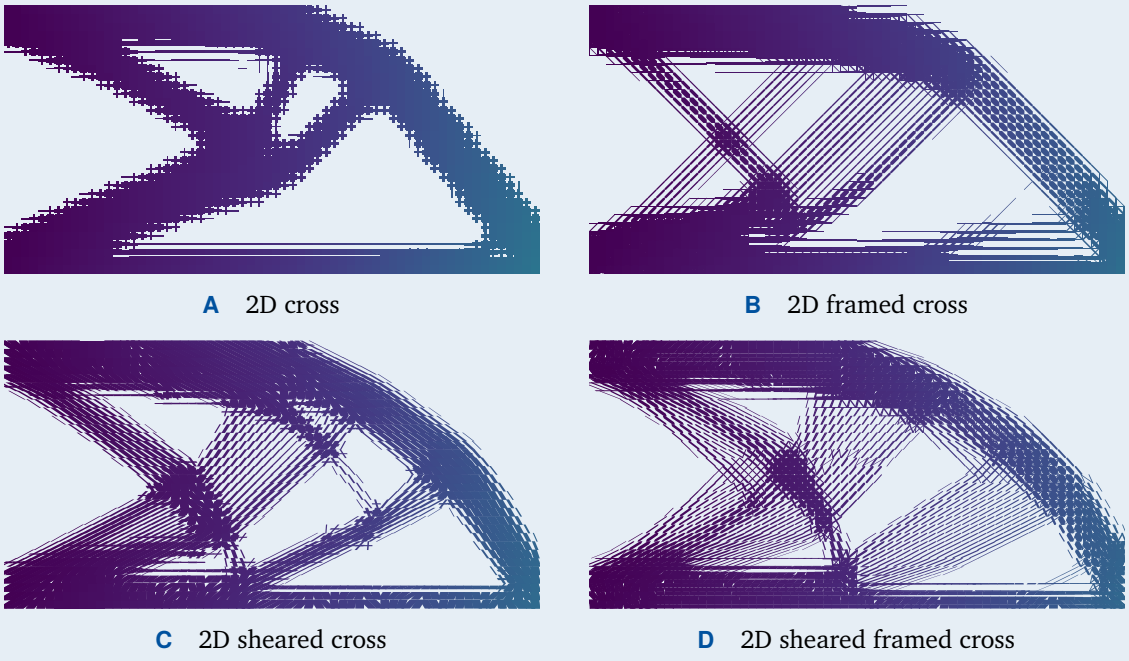
TABLE 6.2 Optimal compliance values for the different scenarios and micro-cell models using cubic B-splines (spatially adaptive grids with around 10 000 points). The entries highlighted in **bold face** indicate the best choice of micro-cell models for a given scenario. More details can be found in Tab. C.1.

and the corresponding possible micro-cell models, where $(\boldsymbol{x}^{\mathrm{opt},*,(1)}, \ldots, \boldsymbol{x}^{\mathrm{opt},*,(M)}) \in (\mathbb{R}^d)^M$ is the micro-cell parameter combination that is returned by the optimizer.[11] It is obvious that more complicated micro-cell models lead to lower (better) compliance values, as they are a generalization of the simple models. For instance, the 2D cross is a special case of the 2D framed cross, the 2D sheared cross, and the 2D sheared framed cross. By choosing the respectively best model for each scenario, we are able to decrease the compliance value (and, hence, increase the stability of the resulting structure) by 9.6 % in the 2D cantilever scenario, by 7.7 % in the 2D L-shape scenario, by 34 % in the 3D cantilever scenario, and by 73 % in the 3D center-load scenario. In general, this motivates the usage of more complicated micro-cell models, which cannot be computationally handled with conventional full grid interpolation methods. Consequently, sparse grids or similar methods have to be used.

Corresponding optimal structures. The corresponding optimal structures are shown in Fig. 6.9 for the 2D cantilever scenario and, for reasons of space, in Appendix C in Figures C.1 and C.2 for the other three scenarios. Of course, the periodic micro-cell structures cannot be plotted directly, as the micro-cells are infinitesimally small. Therefore, the figures show for each macro-cell only one single large micro-cell.

Two effects can be seen in the plots of the optimal structures: First, the simpler models are not able to direct the emerging forces at arbitrary angles. For example, the 2D framed cross model strongly prefers angles of 45°, which results in structures that are not as stable as they could be. The 2D sheared cross and 2D sheared framed cross models are considerably more flexible, allowing internal forces to act at almost arbitrary angles. Second, the sheared micro-cell models use the available material volume more

[11] Note that this true compliance value differs from the approximated value $J^s(\boldsymbol{x}^{\mathrm{opt},*,(1)}, \ldots, \boldsymbol{x}^{\mathrm{opt},*,(M)})$, which the optimizer reports as the optimal objective function value.



A 2D cross

B 2D framed cross

C 2D sheared cross

D 2D sheared framed cross

FIGURE 6.9 Topologically optimal structures in the 2D cantilever scenario for different micro-cell models using cubic B-splines (spatially adaptive grids with around 10 000 points). The colors indicate the length of the displacement, where dark regions correspond to weak displacements and bright regions to strong displacements. The color map is the same as in Fig. C.1. Only bars with widths ≥ 0.1 are shown. More details can be found in Tab. C.1.

efficiently than the cross model. This is most striking in the 3D case (see Fig. C.2), where it seems that the sheared cross structures use more volume than the simple cross structures, although the structures spend exactly the same amount of material volume. The reason is that for the cross model, both bars have to be used in order to connect the macro-cell to its neighbors. For the sheared cross model, a shearing of the vertical bar suffices and we save volume by not using the horizontal bar. Both of these effects explain the significantly lower compliance values for the sheared micro-cell models.

Comparison to the direct solution. B-splines on sparse grids lead to a drastic reduction in computation time. Solving the 2D cantilever scenario with the best-placed sheared cross model would take 453 days with exact elasticity tensor evaluations (i.e., without surrogates), assuming the same number of iterations as for the surrogate tensor case and sequential computation of the elasticity tensors. This estimate does not account for approximating the missing derivatives of the elasticity tensor. If we incorporate this and use 100 parallel processes, we still need weeks for the solution. In contrast, the computation time using our sparse grid surrogates is a matter of minutes or hours at

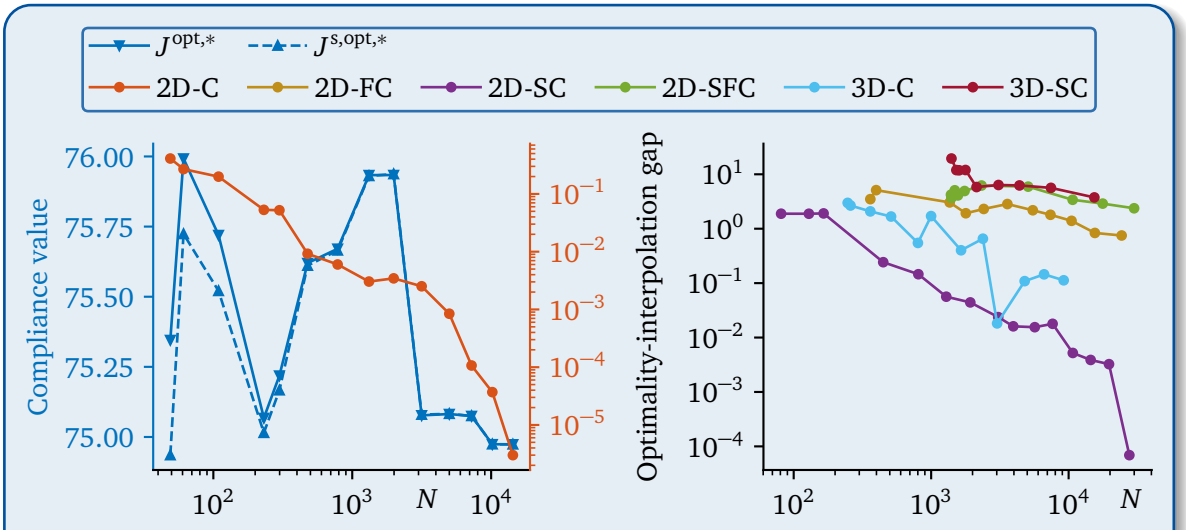


FIGURE 6.10 Convergence of the optimality-interpolation gap $|J^{\text{opt},*} - J^{\text{s,opt},*}|$ for the 2D/3D cantilever scenario and and different micro-cell models using cubic B-splines ($p = 3$). The left plot additionally shows $J^{\text{opt},*} := J(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)})$ and $J^{\text{s,opt},*} := J^{\text{s}}(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)})$ for the 2D cross model.

most, resulting in speedups of around 200. This is excluding the time for the offline phase, which is in the range of hours, but which has to be spent only once, as the resulting grid can be reused for different scenarios.

Optimality-interpolation gaps. Ideally, we would measure the true optimality gap

$$J(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)}) - J(\boldsymbol{x}^{\text{opt},(1)}, \dots, \boldsymbol{x}^{\text{opt},(M)}), \tag{6.11}$$

cf. error E5. Unfortunately, the true optimum $(\boldsymbol{x}^{\text{opt},(1)}, \dots, \boldsymbol{x}^{\text{opt},(M)})$ could not be computed: Apart from the time issue mentioned above, oscillations in the elasticity tensor evaluation and errors stemming from types E1 and E6 reliably led to optimizer crashes as it ran into discontinuities, which are smoothed out when using B-spline surrogates. However, as in Fig. 6.10, we can at least calculate the *optimality-interpolation gap*

$$|J(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)}) - J^{\text{s}}(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)})| \tag{6.12}$$

between the actual compliance value and the approximated, reported compliance value. This gap does not constitute any kind of bound on the true optimality gap; however, the idea is that as the interpolation error converges to zero, the optimality-interpolation gap should converge to zero, too.

Scenario	2D/3D cross $p=1$	2D/3D cross $p=3$	2D/3D cross $p=5$	2D/3D sheared cross $p=1$	2D/3D sheared cross $p=3$	2D/3D sheared cross $p=5$
2D cantilever	*82.365*	**74.974**	76.070	*68.889*	**67.809**	68.018
2D L-shape	*193.83*	**183.68**	183.70	*169.85*	169.60	**169.60**
3D cantilever	*249.75*	247.60	**247.33**	***148.72***	162.59	152.34
3D center-load	**162.68**	169.27	163.94	***45.713***	46.171	47.074

TABLE 6.3 Optimal compliance values for the different scenarios and B-spline degrees using the 2D/3D cross micro-cell model *(left)* and the 2D/3D sheared cross micro-cell model *(right)*. The spatially adaptive sparse grids are the same as in Tab. 6.2. The entries highlighted in **bold face** indicate the best choice of B-spline degree for a given scenario and micro-cell model. Optimization runs of entries marked as *italic* terminated prior to success due to numerical difficulties.

Figure 6.10 (left) shows that for the 2D cross model, the optimizer reports compliance values that are smaller than in reality ($J^{\mathrm{s,opt},*}$ vs. $J^{\mathrm{opt},*}$). However, the difference steadily converges to zero. This is similar for the other micro-cell model as shown in the right part of Fig. 6.10, although the convergence is much slower due to the higher number d of micro-cell parameters.

Optimal compliance values for different B-spline degrees. Finally, to study the effect of the B-spline degree on the optimization performance, Tab. 6.3 lists the compliance values for the degrees $p = 1, 3, 5$ and the 2D/3D cross and sheared cross micro-cell models. In the two-dimensional scenarios, higher-order B-splines decrease the compliance value by up to 9 %. In the three-dimensional scenarios, higher-order B-splines may perform worse than the piecewise linear functions ($p = 1$). (However, as indicated in Tab. 6.3, all optimization runs with piecewise linear functions terminated prematurely due to numerical difficulties with the discontinuous derivatives.) It may be suspected that if we used micro-cell models with less prominent discontinuities (i.e., "smoother" elasticity tensors), the advantage of higher-order B-splines would be more visible. All in all, the application of topology optimization underlines that good interpolation (and thus a good quality of the surrogate) is key to good optimization results.

Application 2: Musculoskeletal Models

> *Beware of bugs in the above code;*
> *I have only proved it correct, not tried it.*
>
> — Donald E. Knuth [Knu77]

Existing musculoskeletal models of muscle-tendon complexes, e.g., of the human upper limb, can mainly be divided into two different types. *Lumped-parameter musculoskeletal models,* for example Hill-type models based on multi-body simulations [Röh16; Vale18b], constitute the most common type. These models assume that the components of the musculoskeletal system are rigid. The mechanics is reduced to point masses associated with their moment of inertia; thus, these models can be described by few parameters.

Continuum-mechanical musculoskeletal models form the other type. Their advantage is that they are more detailed and, hence, more realistic. However, their increased complexity leads to higher computational costs. As an example, we consider an inverse problem (see Chap. 1) that involves a continuum-mechanical simulation of such a musculoskeletal model, where we search values of model parameters such that a specific movement is attained. Each iteration of the solution process for such an inverse problem may take hours or even days, depending on the model at hand.

F_T	Triceps force	r_T	Triceps lever arm	β_T	Triceps activation
F_B	Biceps force	r_B	Biceps lever arm	β_B	Biceps activation
F_L	Load force	r_L	Load lever arm	M	Moment
θ	Elbow angle	θ^*	Target elbow angle	θ_{F_L}	Equil. elbow angle
t	Time	$(\cdot)^s$	Sparse grid solution	$(\cdot)^{\text{ref}}$	Reference solution

TABLE 7.1 Glossary of the notation for musculoskeletal models.

Surrogate methods based on sparse grids help to decrease the complexity in two ways: First, the evaluation of surrogates is obviously drastically cheaper than the solution of the inverse problem. Second, the particular choice of sparse grids decreases the number of necessary samples to construct the surrogates. As for the previous application, the choice of B-splines as hierarchical basis functions enables the evaluation of continuously differentiable surrogate gradients. For the example of inverse problems, this means that gradient-based optimization methods may be employed, which significantly accelerates convergence compared to gradient-free methods.

This chapter is split into three sections. First, in Sec. 7.1, we introduce a continuum-mechanical model of the human upper limb. Second, in Sec. 7.2, we list the types of inverse problems of interest. Finally, in Sec. 7.3, we present numerical results regarding the solution of these inverse problems.

The results of this chapter are based on a collaboration with Prof. Oliver Röhrle, PhD, and Dr. Michael Sprenger (both SimTech/University of Stuttgart, Germany).[1] The collaborators contributed the biomechanical model with its theory, its geometry, and its implementation, while the author of this thesis contributed the sparse grid/B-spline methodology and computed the numerical results. Note that the results have already been published in a paper [Vale18b], which we will follow closely in this chapter.

7.1 Continuum-Mechanical Model of the Upper Limb

In the following, we first discuss the state of the art in biomechanical modeling. Then, we address details of the model of the human upper limb. For convenience, the most relevant symbols are listed in Tab. 7.1.

IN THIS SECTION

[1] Michael Sprenger left the University of Stuttgart in 2015.

7.1.1 Continuum-Mechanical Musculoskeletal Models

Limitations of classical models and benefits of continuum-mechanical models. Due to the simplicity of classical lumped-parameter models, their degree of realism is limited. Without any modifications, lumped-parameter models are not able to represent detailed heterogeneous material characteristics or non-trivial muscle force paths [Röh16].

The exploitation of continuum-mechanical constitutive laws for musculoskeletal models is a more recent development [Röh16]. The resulting models are capable to model spatial quantities such as complex muscle fiber field architectures, local activation principles, complex muscle geometries, or contact mechanics [Röh16; Vale18b]. Most of the existing work only treats single skeletal muscles in isolation [Lem05; Shara11; Hei14]. The model used in this thesis (which is the same model as in [Spr15; Röh16; Vale18b]) aims at studying the interplay of multiple muscles and bones.

Overdetermined antagonistic systems. Musculoskeletal systems are typically overdetermined [Röh16]. This means that the number of muscles that act on a specific joint is usually larger than the number of the joint's degrees of freedom. For instance, in a simple model of the human upper limb, there are two antagonistic muscles (i.e., muscles that work against each other), namely triceps and biceps, but only one joint angle at the elbow. Mathematically speaking, a single muscle suffices to attain a large range of elbow angles that are possible with an antagonistic muscle pair. However, the usage of two muscles enables faster movements and allows for abrupt changes of direction.

The overdetermination of most musculoskeletal models implies that additional conditions have to be imposed in order to obtain unique solutions. There exist various types of such conditions, for instance, minimal control effort, minimal control change, and minimal kinematic energy [Vale18b]. The idea behind these conditions is that the human body tries to minimize the energy effort that is associated with all types of motion.

Forward and inverse dynamics. Musculoskeletal simulations are usually based on either forward dynamics or inverse dynamics [Vale18b]. *Forward-dynamic approaches* use activation parameters for the muscles as the input and simulate the corresponding motion as the output. This requires that we know the muscle forces (depending on the activation levels) beforehand. For example, one can prescribe activation levels of facial muscles to achieve specific facial expressions [Wu13]. In contrast, *inverse-dynamic approaches* use experimental motion data as the input to estimate the muscle forces as the output [Röh16]. With inverse-dynamic simulations, one can investigate the wrapping of muscles around the knee joint [Fern05] or visualize the motion of skin [Lee09], for instance.

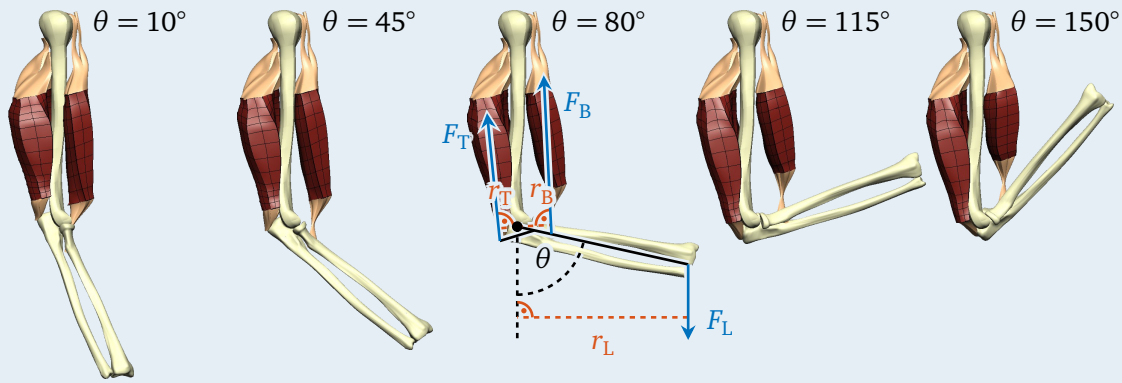


FIGURE 7.1 Human upper limb model geometry shown as a raising arm movement for the elbow angles $\theta = 10°$, 45°, 80°, 115°, and 150° *(from left to right)*. For $\theta = 80°$, the contributing forces *(blue)* and lever arms *(red)* are shown. Taken and adapted from [Spr15; Vale18b].

7.1.2 Details of the Human Upper Limb Model

As shown in Fig. 7.1, our model of the human upper limb consists of the three bones humerus, ulna, and radius, the elbow joint with one degree of freedom, and the antagonistic muscle pair of triceps brachii and biceps brachii [Vale18b]. The bones are rigid bodies and the muscle-tendon complex is simulated with a continuum-mechanical approach. This implies that the muscles deform when they contract.

Overall stress components. The continuum-mechanical part of the model is based on the theory of finite elasticity. When muscles deform, forces act on each infinitesimally small element of the muscles, which is known as *stress*. Usually, especially in linear elasticity, stress is measured with the *Cauchy stress tensor* (also called the *true stress*) [Sön13]. For non-linear stress-strain relations, one may use other measures such as the *second Piola-Kirchhoff stress*. The second Piola-Kirchhoff stress has the additional advantage that it is defined along the material directions, in contrast to the Cauchy stress tensor, which measures the stress in coordinate directions [Sön13].

In [Spr15; Röh16; Vale18b], the strain energy function is defined such that the resulting overall second Piola-Kirchhoff stress $\boldsymbol{S}_{\text{MTC}}$ of the muscle-tendon complex satisfies

$$\boldsymbol{S}_{\text{MTC}} = \boldsymbol{S}_{\text{iso}} + \boldsymbol{S}_{\text{aniso}} - p\boldsymbol{C}^{-1}, \tag{7.1}$$

where $\boldsymbol{S}_{\text{iso}}$ and $\boldsymbol{S}_{\text{aniso}}$ are the *isotropic and anisotropic parts* of the stress, respectively, p is the *hydrostatic pressure* to ensure incompressibility, and $\boldsymbol{C}$ is the *right Cauchy–Green*

deformation tensor. The anisotropic part $\boldsymbol{S}_{\text{aniso}}$ is defined as

$$\boldsymbol{S}_{\text{aniso}} := (\boldsymbol{S}_{\text{passive}} + \beta\gamma_{\text{M}}\boldsymbol{S}_{\text{active}})(1-\gamma_{\text{ST}}), \tag{7.2}$$

cf. [Vale18b]. Here, $\boldsymbol{S}_{\text{passive}}$ and $\boldsymbol{S}_{\text{active}}$ are the *passive and active contributions* due to the skeletal muscle fibers, $\beta \in [0,1]$ is the *activation parameter* of the respective muscle-tendon complex, and $\gamma_{\text{M}}, \gamma_{\text{ST}}$ are two *material parameters* with which we can differentiate between the different types of soft tissues of the muscle-tendon complex: fat, tendon, and muscle [Vale18b]. Isotropic fat tissue can be obtained by setting $\gamma_{\text{ST}} := 1$, passive anisotropic tendon tissue by setting $\gamma_{\text{ST}} := 0$ and $\gamma_{\text{M}} := 0$, and skeletal muscle tissue by setting $\gamma_{\text{ST}} := 0$ and $\gamma_{\text{M}} := 1$. A mixture of these pure materials is achieved by linear interpolation when setting γ_{M} and γ_{ST} to values between zero and one [Vale18b]. More details about the theory part of the model are given in [Spr15; Röh16; Vale18b].

7.2 Momentum Equilibrium and Elbow Angle Optimization

In this section, we give an overview of the methodology of our approach. We continue following the presentation of [Vale18b].

IN THIS SECTION

7.2.1 From Muscle Forces to Equilibrium Angles

Model inputs and outputs. In the following, we regard simulations of the human upper limb model described in Sec. 7.1 as a black box, which receives as its input the elbow angle $\theta \in [10°, 150°]$ and the activation parameters $(\beta_{\text{T}}, \beta_{\text{B}}) \in [\mathbf{0}, \mathbf{1}] = [0,1]^2$ of triceps and biceps.[2] The outputs of the black box simulation are the forces $F_{\text{T}}(\theta, \beta_{\text{T}})$ and $F_{\text{B}}(\theta, \beta_{\text{B}})$ that triceps and biceps exert. These forces depend on the elbow angle as well as on the respective activation parameter. Gravitational forces due to the masses of bones or muscles are neglected in this context. However, we allow the specification of an external load F_{L}, which is applied to the end of the forearm. This load may be the weight force of some object that the arm is supposed to keep in position.

Moments and lever arms. Each force exerts a *moment* (or *torque*) on the elbow joint. The moments are the products of the forces F_X with the respective lever arms r_X ($X \in$

[2] Here and in the following, the subscripts T, B, and L stand for triceps, biceps, and load, respectively.

$\{\mathrm{T}, \mathrm{B}, \mathrm{L}\}$). The lever arms are approximated as in [Röh16; Vale18b] by using the tendon-displacement method of [An84]:

$$r_{\mathrm{T}}(\theta) := (-0.0009399\{\theta\}^2 + 0.1126\{\theta\} + 22.21)\,\mathrm{mm}, \tag{7.3a}$$

$$r_{\mathrm{B}}(\theta) := (-0.001482\{\theta\}^2 + 0.1776\{\theta\} + 35.02)\,\mathrm{mm}, \tag{7.3b}$$

$$r_{\mathrm{L}}(\theta) := \sin(\theta) \cdot 282.5\,\mathrm{mm}, \tag{7.3c}$$

where $\{\theta\}$ denotes the dimensionless value of θ in degrees. The lever arms are non-negative and the forces are signed, i.e., positive forces pull the forearm downwards and negative forces pull it upwards. In general, $F_{\mathrm{T}}, F_{\mathrm{L}} \geq 0\,\mathrm{N}$ and $F_{\mathrm{B}} \leq 0\,\mathrm{N}$.

Total moment and equilibrium elbow angle. The *total moment* of the system is given by the function

$$M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}} : [10°, 150°] \to \mathbb{R}, \tag{7.4a}$$

$$M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}(\theta) := F_{\mathrm{T}}(\theta, \beta_{\mathrm{T}}) r_{\mathrm{T}}(\theta) + F_{\mathrm{B}}(\theta, \beta_{\mathrm{B}}) r_{\mathrm{B}}(\theta) + F_{\mathrm{L}} r_{\mathrm{L}}(\theta), \tag{7.4b}$$

cf. [Vale18b]. The system is in *equilibrium* if the total moment vanishes, i.e., $M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}(\theta) = 0\,\mathrm{Nm}$. We call the corresponding angle θ the *equilibrium elbow angle* for the load F_{L} and the activation parameters $\beta_{\mathrm{T}}, \beta_{\mathrm{B}}$. To find this angle for a given load F_{L} and activation parameters β_{T} and β_{B}, we first note that $M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}$ may have zero, exactly one, or multiple zeros in $[10°, 150°]$. Hence, the inverse function evaluated at $0\,\mathrm{Nm}$ is partially defined depending on the load and the activation parameters:

$$\theta_{F_{\mathrm{L}}} : D_{\theta_{F_{\mathrm{L}}}} \to [10°, 150°], \quad D_{\theta_{F_{\mathrm{L}}}} \subseteq [\mathbf{0}, \mathbf{1}], \quad \theta_{F_{\mathrm{L}}}(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) := (M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}})^{-1}(0\,\mathrm{Nm}), \tag{7.5}$$

which is well-defined whenever $M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}$ has a unique root. We approximate $\theta_{F_{\mathrm{L}}}(\beta_{\mathrm{T}}, \beta_{\mathrm{B}})$ with the Newton method [Röh16; Vale18b]:

$$\theta^{(j+1)} := \theta^{(j)} - \frac{M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}(\theta^{(j)})}{\frac{\partial}{\partial\theta} M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}(\theta^{(j)})}, \quad j \in \mathbb{N}, \tag{7.6}$$

with an initial value $\theta^{(0)} \in [10°, 150°]$ and the stopping criterion of $|M_{F_{\mathrm{L}},\beta_{\mathrm{T}},\beta_{\mathrm{B}}}(\theta^{(j)})| < 10^{-9}\,\mathrm{Nm}$. We repeat the Newton method for the initial values $\theta^{(0)} = 80°, 40°, 120°$ and use the first converged result (i.e., we check if $\theta^{(0)} = 80°$ converges; if not, we proceed with $\theta^{(0)} = 40°$, and so on). If all three initial values do not converge, we conclude that $(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) \notin D_{\theta_{F_{\mathrm{L}}}}$.

7.2.2 Optimization Problems

General problem. The general problem in our setting is as follows: For a given external load F_{L} and a target elbow angle θ^*, find activation parameters $(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) \in [\mathbf{0}, \mathbf{1}]$ such that the target elbow angle is attained in the equilibrium, i.e., $\theta_{F_{\mathrm{L}}}(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) = \theta^*$. Example applications of such a scenario are medicine and robotics, when a specific movement should be carried out.

List of optimization problems. As discussed in Sec. 7.1.1, musculoskeletal systems with an antagonistic muscle pair such as our human upper limb model are usually overdetermined. This means that there are multiple solutions to this general problem. As a remedy, one may solve one of the following two optimization problems [Vale18b]:

O1. For a given external load F_{L} and a target angle $\theta^* \in [10°, 150°]$, find the activation parameters $(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) \in [\mathbf{0}, \mathbf{1}]$ such that $\beta_{\mathrm{T}} + \beta_{\mathrm{B}}$ is minimized under the constraint $\theta_{F_{\mathrm{L}}}(\beta_{\mathrm{T}}, \beta_{\mathrm{B}}) = \theta^*$.

O2. For a given external load $F_{\mathrm{L}}(t_2)$ for a time $t_2 > t_1$, a target angle $\theta^*(t_2) \in [10°, 150°]$, and initial activation parameters $(\beta_{\mathrm{T}}(t_1), \beta_{\mathrm{B}}(t_1)) \in [\mathbf{0}, \mathbf{1}]$, find new activation parameters $(\beta_{\mathrm{T}}(t_2), \beta_{\mathrm{B}}(t_2)) \in [\mathbf{0}, \mathbf{1}]$ such that $(\beta_{\mathrm{T}}(t_2) - \beta_{\mathrm{T}}(t_1))^2 + (\beta_{\mathrm{B}}(t_2) - \beta_{\mathrm{B}}(t_1))^2$ is minimized under the constraint $\theta_{F_{\mathrm{L}}(t_2)}(\beta_{\mathrm{T}}(t_2), \beta_{\mathrm{B}}(t_2)) = \theta^*(t_2)$.

The motivation of both problems is that the human body tries to achieve a given movement with minimal energy effort.

Motivation of problem O1. For the first problem O1, this effort is quantified by $\beta_{\mathrm{T}} + \beta_{\mathrm{B}}$, i.e., the energy effort for each muscle is assumed to be proportional to its activation parameter.

Motivation of problem O2. The second problem O2 is motivated as follows: Before time $t = t_1$, the musculoskeletal system is in equilibrium for the external load $F_{\mathrm{L}}(t_1)$, activation parameters $\beta_{\mathrm{T}}(t_1), \beta_{\mathrm{B}}(t_1)$, and elbow angle $\theta^*(t_1) := \theta_{F_{\mathrm{L}}(t_1)}(\beta_{\mathrm{T}}(t_1), \beta_{\mathrm{B}}(t_1))$, i.e., $M_{F_{\mathrm{L}}(t_1), \beta_{\mathrm{T}}(t_1), \beta_{\mathrm{B}}(t_1)}(\theta^*(t_1)) = 0\,\mathrm{Nm}$. Directly after $t = t_1$, the external force and/or the target angle is suddenly changed to $F_{\mathrm{L}}(t_2)$ and $\theta^*(t_2)$, respectively. Consequently, triceps and biceps adapt their activation parameters such that the musculoskeletal system returns to equilibrium at some time $t = t_2 > t_1$. Hence, we have to determine the new activation parameters $\beta_{\mathrm{T}}(t_2), \beta_{\mathrm{B}}(t_2)$ such that $M_{F_{\mathrm{L}}(t_2), \beta_{\mathrm{T}}(t_2), \beta_{\mathrm{B}}(t_2)}(\theta^*(t_2)) = 0\,\mathrm{Nm}$. Again, these parameters $\beta_{\mathrm{T}}(t_2)$ and $\beta_{\mathrm{B}}(t_2)$ are not uniquely determined. Therefore, we want to find the pair of activation parameters that is closest (in terms of the Euclidean norm) to the initial activation parameters $\beta_{\mathrm{T}}(t_1), \beta_{\mathrm{B}}(t_1)$.

Optimization method. Problems O1 and O2 are both constrained optimization problems. For their solution, we employ the augmented Lagrangian method as described in Sec. 5.1.3 using an adaptive gradient descent algorithm for the gradient-based optimization of the penalized objective function (see Sec. 5.1.2).

7.2.3 B-Spline Surrogates on Sparse Grids

Complexity. To solve optimization problems O1 and O2, the optimization method needs to evaluate the objective and constraint functions multiple times during the algorithm. This requires the evaluation of θ_{F_L}, which in turn has to be approximated with the Newton method. As we see in Eq. (7.6), each iteration of the Newton method needs not only the values of the muscle forces F_T and F_B, but also their partial derivatives with respect to θ. These partial derivatives have to be approximated with finite differences.

Unfortunately, simulations of continuum-mechanical models are computationally expensive. One evaluation of the muscle force pair $F_\mathrm{T}, F_\mathrm{B}$ requires the solution of a solid mechanics model with a complex constitutive law, pre-stretch, and contact between bone and muscles [Vale18b]. On average, a single evaluation of F_T and F_B takes about half an hour on current desktop computers. If we assume that we need four Newton iterations on average, then a single iteration of the optimization algorithm to solve problems O1 and O2 will take four hours to complete (assuming one evaluation of objective and constraint functions per optimizer iteration and two evaluations of the muscle force pair per Newton iteration to approximate the missing derivative). Consequently, the whole optimization process takes two weeks to complete, if the optimizer converges after 100 iterations.

Sparse grid surrogates. A popular way to reduce complexity is to employ surrogates. In this case, the idea is to replace the muscle force functions $F_\mathrm{T}, F_\mathrm{B}$ with surrogates $F_\mathrm{T}^\mathrm{s}, F_\mathrm{B}^\mathrm{s}$ [Vale18b], e.g., by interpolation. We then automatically obtain a surrogate

$$M^\mathrm{s}_{F_\mathrm{L},\beta_\mathrm{T},\beta_\mathrm{B}} : [10°, 150°] \to \mathbb{R}, \tag{7.7a}$$

$$M^\mathrm{s}_{F_\mathrm{L},\beta_\mathrm{T},\beta_\mathrm{B}}(\theta) := F^\mathrm{s}_\mathrm{T}(\theta,\beta_\mathrm{T}) r_\mathrm{T}(\theta) + F^\mathrm{s}_\mathrm{B}(\theta,\beta_\mathrm{B}) r_\mathrm{B}(\theta) + F_\mathrm{L} r_\mathrm{L}(\theta), \tag{7.7b}$$

for the total moment (cf. Eq. (7.4)) and, consequently, a surrogate

$$\theta^\mathrm{s}_{F_\mathrm{L}} : D^\mathrm{s}_{\theta_{F_\mathrm{L}}} \to [10°, 150°], \quad D^\mathrm{s}_{\theta_{F_\mathrm{L}}} \subseteq [\mathbf{0}, \mathbf{1}], \quad \theta^\mathrm{s}_{F_\mathrm{L}}(\beta_\mathrm{T}, \beta_\mathrm{B}) := (M^\mathrm{s}_{F_\mathrm{L},\beta_\mathrm{T},\beta_\mathrm{B}})^{-1}(0\,\mathrm{Nm}), \tag{7.8}$$

for the equilibrium elbow angle function (cf. Eq. (7.5)). Since the surrogates are much cheaper to evaluate, the computation time is decreased by up to seven orders of magnitude, as experiments show.

The approach in [Vale18b] and in this thesis is to determine surrogates $F_X^{\mathrm{s}} : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$ ($X \in \{\mathrm{T}, \mathrm{B}\}$) by sparse grid interpolation. Compared to surrogate construction techniques based on full grids, sparse grids help to reduce the number of samples that are necessary to build "reasonably" accurate surrogates, especially if the number of dimensions is moderately large ($d \geq 4$, *curse of dimensionality*).

The present model only has $d = 2$ dimensions (β_{T} and β_{B}), since the model contains only two muscles. However, as we will see, already for this low-dimensional problem, sparse grids outperform conventional full grid interpolation. The results have to be seen as a proof of concept. One will be able to handle higher dimensionalities (i.e., models with a larger number of muscles) similarly with little or even no adjustments at all. The low dimensionality of the model in this thesis enables us to compute and compare against reference solutions, which would not be possible in a higher-dimensional setting.

Benefiting from B-splines. As in [Vale18b], we use higher-order hierarchical B-splines as basis functions for the sparse grid surrogates. This has three advantages when compared with conventional sparse grid bases such as piecewise linear functions: First, the partial derivative $\frac{\partial}{\partial \theta} M^{\mathrm{s}}$ needed for the Newton method in Eq. (7.6) is continuous and explicitly known. There is no need to approximate the derivative with finite differences, reducing both error and computation time. Second, we can use gradient-based optimization methods for the solution of the optimization problems O1 and O2, which involve the equilibrium elbow angle function $\theta_{F_{\mathrm{L}}}^{\mathrm{s}} : [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$. With the implicit function theorem [Kud95], we obtain for the derivative of $\theta_{F_{\mathrm{L}}}^{\mathrm{s}}$

$$\boldsymbol{\nabla}_{\beta_{\mathrm{T}},\beta_{\mathrm{B}}} \theta_{F_{\mathrm{L}}}^{\mathrm{s}} = -(\boldsymbol{\nabla}_{\beta_{\mathrm{T}},\beta_{\mathrm{B}}} M^{\mathrm{s}}) \cdot (\boldsymbol{\nabla}_{\theta} M^{\mathrm{s}})^{-1} = -\frac{\boldsymbol{\nabla}_{\beta_{\mathrm{T}},\beta_{\mathrm{B}}} M^{\mathrm{s}}}{\frac{\partial}{\partial \theta} M^{\mathrm{s}}}, \tag{7.9}$$

where $\boldsymbol{\nabla}_{\beta_{\mathrm{T}},\beta_{\mathrm{B}}}$ is the transposed Jacobian with respect to β_{T} and β_{B}.[3] For B-splines, both the transposed Jacobian $\boldsymbol{\nabla}_{\beta_{\mathrm{T}},\beta_{\mathrm{B}}} M^{\mathrm{s}}$ and the partial derivative $\frac{\partial}{\partial \theta} M^{\mathrm{s}}$ are continuous, explicitly known, and can be evaluated fast. Third and finally, the usage of higher-order B-splines as basis functions increases the order of convergence of interpolation errors as shown for test functions in Sec. 5.4.1. Thus, fewer interpolation points are necessary to construct a surrogate with the same error as for piecewise linear functions.

[3] For example, the first column is the gradient with respect to β_{T} and the second column is the gradient with respect to β_{B}.

7.3 Implementation and Numerical Results

7.3.1 Implementation

IN THIS SECTION

Parameters, implementation, and geometry. Details about implementational aspects of the model can be found in [Spr15; Röh16; Vale18b], for instance, values for the material parameters. The constitutive law has been implemented in the CMISS software package (an interactive computer program for Continuum Mechanics, Image analysis, Signal processing and System identification[4]). The emerging PDEs are discretized using quadratic finite element basis functions and the resulting linearized system is solved with CMISS. The geometry of the human upper limb model is based on the Visible Human Male's dataset [Spi96]. Again, we refer to [Spr15; Röh16] for details about the geometry.

7.3.2 Reference and Sparse Grid Solution

Reference solution. Since the model is only two-dimensional, we can compute a reference solution on a full grid. To this end, we evaluate the exerted muscle forces F_{T} and F_{B} on the full grid

$$\{10°, 11°, \dots, 150°\} \times \{0, 0.1, \dots, 1\} \ni (\theta, \beta_X), \quad X \in \{\mathrm{T}, \mathrm{B}\}. \tag{7.10}$$

The resulting 1551 grid points are interpolated with bicubic full grid splines[5] to obtain *reference solutions* $F_{\mathrm{T}}^{\mathrm{ref}}, F_{\mathrm{B}}^{\mathrm{ref}} \colon [10°, 150°] \times [0, 1] \to \mathbb{R}$, which are shown in Fig. 7.2. Due to the high resolution of the full grid, we may assume that the reference solutions are accurate enough to ensure $F_{\mathrm{T}}^{\mathrm{ref}} \approx F_{\mathrm{T}}$ and $F_{\mathrm{B}}^{\mathrm{ref}} \approx F_{\mathrm{B}}$. We refer to the resulting equilibrium elbow angle with $\theta_{F_{\mathrm{L}}}^{\mathrm{ref}}$. It is displayed in Fig. 7.3 for the loads of $F_{\mathrm{L}} = 22\,\mathrm{N}$, $-60\,\mathrm{N}$, and $180\,\mathrm{N}$.

Sparse grid solution. Additionally, we evaluate F_{T} and F_{B} at the $N = 49$ grid points

$$\{(\theta^{(k,\mathrm{unif})}, \beta_X^{(k,\mathrm{unif})}) \mid k = 1, \dots, N\} \subseteq [10°, 150°] \times [0, 1], \quad X \in \{\mathrm{T}, \mathrm{B}\}, \tag{7.11}$$

of the uniform regular sparse grid $\mathring{\Omega}_{n,d}^{\mathrm{s}}$ of level $n = 5$ in $d = 2$ dimensions without boundary points (to reduce the number of samples) and at the sparse Clenshaw–Curtis

[4] `https://www.cmiss.org/`
[5] Computed with the Geometric Tools Engine [Schn03], see `https://www.geometrictools.com/`.

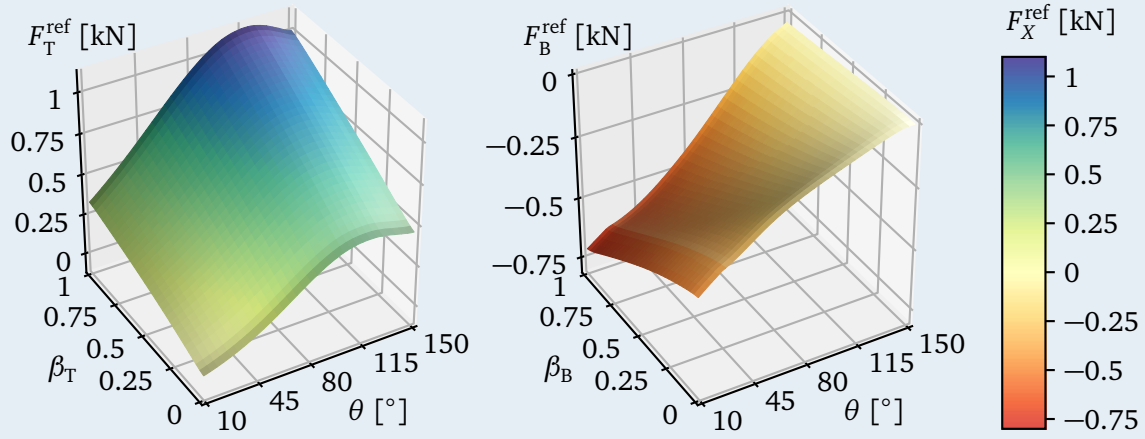


FIGURE 7.2 Reference triceps and biceps forces F_X^{ref} ($X \in \{\text{T}, \text{B}\}$).

grid

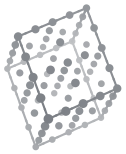
$$\{(\theta^{(k,\text{cc})}, \beta_X^{(k,\text{cc})}) \mid k = 1, \ldots, N\} \subseteq [10°, 150°] \times [0, 1], \quad X \in \{\text{T}, \text{B}\}, \tag{7.12}$$

of the same size and level.[6] These values are interpolated using three different hierarchical B-spline bases of degree $p = 1$, 3, and 5: modified hierarchical uniform B-splines $\varphi_{\ell,i}^{p,\text{mod}}$ (see Sec. 3.1.3), modified hierarchical Clenshaw–Curtis B-splines $\varphi_{\ell,i}^{p,\text{cc,mod}}$ (see Sec. 3.1.4), and modified hierarchical uniform not-a-knot B-splines $\varphi_{\ell,i}^{p,\text{nak,mod}}$ (see Sec. 3.2.3). The implementation was done using the sparse grid toolbox SG^{++} [Pfl10].[7] The corresponding interpolants and resulting quantities are denoted with the superscripts "s,p", "s,p,cc", or "s,p,nak", respectively. A superscript of "s" without any further specification means one of the three hierarchical B-spline bases in general. Note that the equilibrium elbow angle is *not* interpolated (neither in the full grid nor in the sparse grid case), but rather obtained by inserting the interpolated muscle forces into (7.7) and (7.8).

7.3.3 Errors of Muscle Forces and Equilibrium Angle

Quality of reference interpolants. Before we turn to the sparse grid interpolants, we assess the quality of the reference interpolants on the full grid. For this purpose, we evaluate the full grid interpolants $F_\text{T}^\text{s}, F_\text{B}^\text{s}$ at the sparse grid points $(\theta^{(k)}, \beta_X^{(k)})$ (which are not a subset of the full grid points!) and compare the resulting values with the known exact values $F_\text{T}(\theta^{(k)}, \beta_\text{T}^{(k)})$ and $F_\text{B}(\theta^{(k)}, \beta_\text{B}^{(k)})$ of the muscle forces F_T, F_B. We also incorporate

[6] The domain $[10°, 150°] \times [0, 1]$ is assumed to be implicitly normalized to the unit square $[\mathbf{0}, \mathbf{1}]$.
[7] `http://sgpp.sparsegrids.org/`



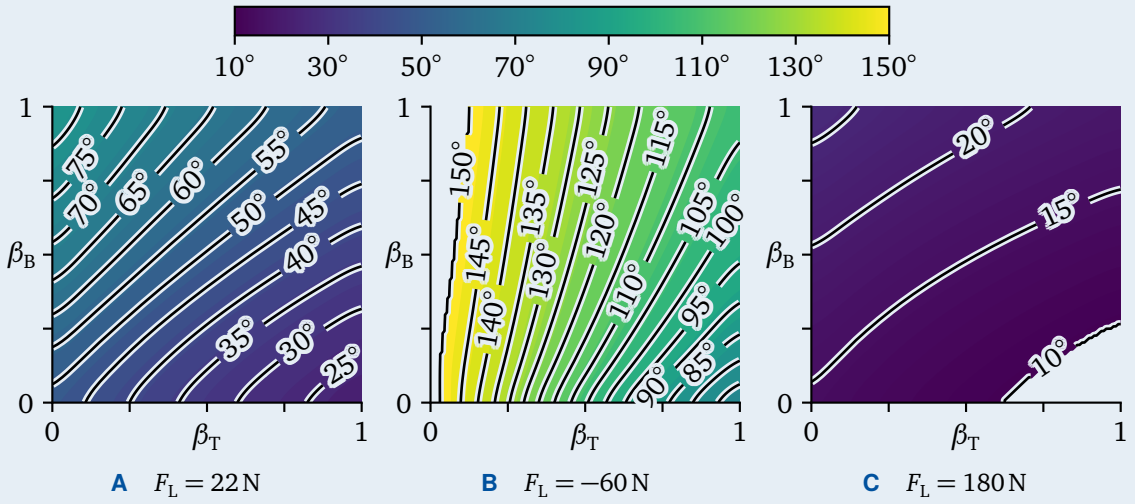


FIGURE 7.3 Reference equilibrium elbow angle $\theta^{\text{ref}}_{F_L}$ for different loads F_L. The empty areas correspond to activation pairs at which $\theta^{\text{ref}}_{F_L}$ is not well-defined (see Eq. (7.5)).

the known values at the sparse Clenshaw–Curtis grid points. In particular, let G be the union of $\{(\theta^{(k,\text{unif})}, \beta_X^{(k,\text{unif})}) \mid k = 1,\dots,N\}$ and $\{(\theta^{(k,\text{cc})}, \beta_X^{(k,\text{cc})}) \mid k = 1,\dots,N\}$. We then approximate the relative L^2 interpolation error of the reference interpolants by

$$\frac{\|F_X - F_X^{\text{ref}}\|_{L^2}}{\|F_X\|_{L^2}} \approx \frac{|G|^{-1/2}\|(F_X(\theta,\beta_X) - F_X^{\text{ref}}(\theta,\beta_X))_{(\theta,\beta_X)\in G}\|_2}{|G|^{-1/2}\|(F_X(\theta,\beta_X))_{(\theta,\beta_X)\in G}\|_2}, \quad X \in \{\text{T},\text{B}\}, \tag{7.13}$$

where $\|\cdot\|_2$ is the Euclidean norm.[8] After inserting the known values $F_X(\theta,\beta_X)$ and $F_X^{\text{ref}}(\theta,\beta_X)$ $((\theta,\beta_X)\in G)$ on the right-hand side, we obtain

$$\frac{\|F_\text{T} - F_\text{T}^{\text{ref}}\|_{L^2}}{\|F_\text{T}\|_{L^2}} \approx 2.19\,‰, \qquad \frac{\|F_\text{B} - F_\text{B}^{\text{ref}}\|_{L^2}}{\|F_\text{B}\|_{L^2}} \approx 2.06\,‰. \tag{7.14}$$

These errors are very small, which justifies our assumption of $F_\text{T}^{\text{ref}} \approx F_\text{T}$ and $F_\text{B}^{\text{ref}} \approx F_\text{B}$.

Error of sparse grid muscle forces. Table 7.2a contains the relative L^2 interpolation errors $\|F_X^{\text{ref}} - F_X^{\text{s}}\|_{L^2}/\|F_X^{\text{ref}}\|_{L^2}$ $(X \in \{\text{T},\text{B}\})$ of the sparse grid interpolants for all hierarchical bases and degrees $p = 1,3,5$. All reported errors are relatively small due to the smoothness of the original functions (cf. F_X^{ref} in Fig. 7.2). All in all, the modified Clenshaw–Curtis B-splines perform best, achieving relative L^2 errors of below 3.6 ‰ in the cubic case. Surprisingly, the not-a-knot B-splines are the worst choice in our comparison. Their corre-

[8] We have $|G| = 2N - 1$, since sparse grids of uniform and Clenshaw–Curtis type only share the center point $(\theta,\beta_X) = (80°, 0.5)$, if there are no boundary points.

A $\|F_X^{\mathrm{ref}} - F_X^{\mathrm{s}}\|_{L^2} / \|F_X^{\mathrm{ref}}\|_{L^2}$ [‰] given as triceps/biceps pairs ($X \in \{\mathrm{T}, \mathrm{B}\}$).

p	1	3	5
$\varphi_{\ell,i}^{p,\mathrm{mod}}$	3.60, 7.12	3.05, 7.00	**2.98**, 7.90
$\varphi_{\ell,i}^{p,\mathrm{cc,mod}}$	**3.28**, 4.35	3.31, **3.56**	3.35, 3.64
$\varphi_{\ell,i}^{p,\mathrm{nak,mod}}$	3.60, 7.12	3.09, 10.0	7.13, 24.6

B $\|\theta_{F_{\mathrm{L}}}^{\mathrm{ref}} - \theta_{F_{\mathrm{L}}}^{\mathrm{s}}\|_{L^2} / \|\theta_{F_{\mathrm{L}}}^{\mathrm{ref}}\|_{L^2}$ [‰] for $F_{\mathrm{L}} = 22\,\mathrm{N}$.

p	1	3	5
$\varphi_{\ell,i}^{p,\mathrm{mod}}$	4.15	3.74	3.72
$\varphi_{\ell,i}^{p,\mathrm{cc,mod}}$	3.42	**2.83**	2.86
$\varphi_{\ell,i}^{p,\mathrm{nak,mod}}$	4.15	4.06	8.28

TABLE 7.2 Relative L^2 errors of triceps/biceps force *(left)* and equilibrium elbow angle *(right)* for different hierarchical bases $\varphi_{\ell,i}$ and B-spline degrees p. Highlighted entries are the best among those with the same hierarchical basis or the same degree (similar to Nash equilibria).

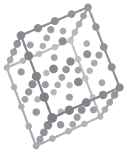
sponding errors exceed 1 % for the triceps and $p > 1$. The possible reasons are two-fold: First, there might be slight noise in the given muscle force data, which is visible in Fig. 7.2, as there seems to be a kink in $F_{\mathrm{B}}^{\mathrm{ref}}$ at $\theta \approx 25°$. Second, the employed regular sparse grids might be too coarse as the higher convergence order of not-a-knot B-splines only pays off in the asymptotic range (see Sec. 5.4.1). The same observations hold for the degree p, for which $p = 3$ seems to be the best choice, as the errors increase again for $p = 5$.

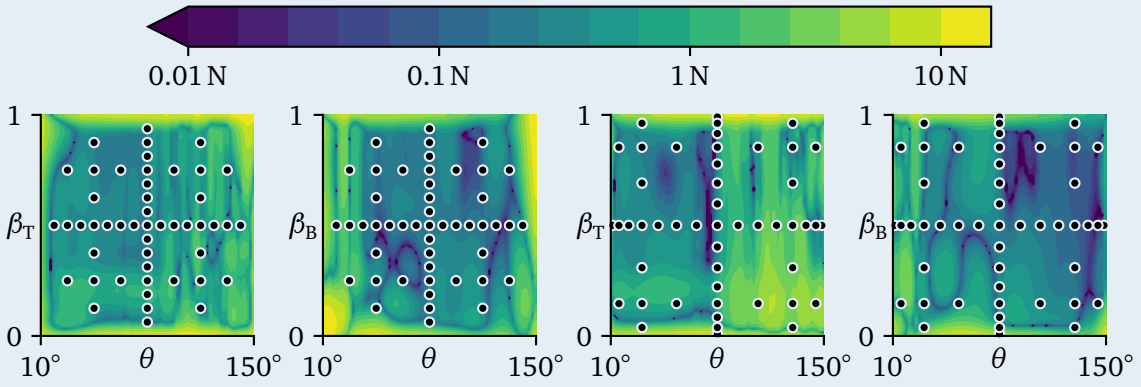
Figure 7.4 shows the pointwise absolute error $|F_X^{\mathrm{ref}}(\theta, \beta_X) - F_X^{\mathrm{s}}(\theta, \beta_X)|$ for the modified B-splines $\varphi_{\ell,i}^{p,\mathrm{mod}}$ and $\varphi_{\ell,i}^{p,\mathrm{cc,mod}}$ on uniform and Clenshaw–Curtis grids in the cubic case $p = 3$. Note that in contrast to usual interpolation settings, the absolute errors $|F_X^{\mathrm{ref}} - F_X^{\mathrm{s}}|$ shown in Fig. 7.4 do not vanish at the sparse grid points $(\theta^{(k)}, \beta_X^{(k)})$ ($X \in \{\mathrm{T}, \mathrm{B}\}$, $k = 1, \ldots, N$), since F_X^{s} does not interpolate F_X^{ref} at these points.[9] As it is typical for (modified) sparse grid interpolants, the error is the largest near the boundary of the domain. However, the Clenshaw–Curtis points help to decrease the error due to the higher density of grid points near the boundary. In the Clenshaw–Curtis case, the maximal errors are

$$\|F_{\mathrm{T}}^{\mathrm{ref}} - F_{\mathrm{T}}^{\mathrm{s},p,\mathrm{cc}}\|_{L^\infty} \approx 10.6\,\mathrm{N}, \qquad \|F_{\mathrm{B}}^{\mathrm{ref}} - F_{\mathrm{B}}^{\mathrm{s},p,\mathrm{cc}}\|_{L^\infty} \approx 9.51\,\mathrm{N}, \tag{7.15}$$

where $\|F_X^{\mathrm{ref}} - F_X^{\mathrm{s},p,\mathrm{cc}}\|_{L^\infty} := \max_{(\theta,\beta_X)} |F_X^{\mathrm{ref}}(\theta, \beta_X) - F_X^{\mathrm{s},p,\mathrm{cc}}(\theta, \beta_X)|$ (since the functions are continuous). If we restrict the domain to $[31°, 129°] \times [0.15, 0.85]$ by omitting 15 % on each side of the original domain, then the maximal absolute errors drop to only 6.73 N (triceps) and 0.967 N (biceps), which is small compared to maximal possible forces of around 1 kN.

[9] It would have been possible to construct F_X^{s} as a sparse grid interpolant of F_X^{ref}. However, building a spline surrogate (F_X^{s}) of another spline surrogate (F_X^{ref}) would skew the results.





A $|F_X^{\text{ref}} - F_X^{s,p}|$ for $X = \text{T}$ *(left)* and $X = \text{B}$ *(right)*. **B** $|F_X^{\text{ref}} - F_X^{s,p,\text{cc}}|$ for $X = \text{T}$ *(left)* and $X = \text{B}$ *(right)*.

FIGURE 7.4 Absolute error of muscle forces F_T, F_B for modified cubic B-splines ($p = 3$) on sparse grids of uniform type *(left two plots)* and of Clenshaw–Curtis type *(right two plots)* together with the points of the sparse grid *(dots)*.

Error of the equilibrium elbow angle. The relative L^2 errors $\|\theta_{F_\text{L}}^{\text{ref}} - \theta_{F_\text{L}}^{s}\|_{L^2} / \|\theta_{F_\text{L}}^{\text{ref}}\|_{L^2}$ of the equilibrium elbow angle function are shown in Tab. 7.2a for the load of $F_\text{L} = 22\,\text{N}$. Modified cubic Clenshaw–Curtis B-splines achieve the best results. Therefore, we use this type of hierarchical basis for the remainder of this chapter. Pointwise plots of the absolute error $|\theta_{F_\text{L}}^{\text{ref}} - \theta_{F_\text{L}}^{s,p,\text{cc}}|$ are presented in Fig. 7.5. Again, the maximal error is comparatively small: For $F_\text{L} = 22\,\text{N}$, it is only 0.886°. If we restrict the domain to $[0.15, 0.85]^2$, then this maximal error drops to 0.103° (or 6.18′), as the areas near the boundary of $[\mathbf{0}, \mathbf{1}]$ contribute the most to the error.

7.3.4 Test Scenario

Definition of the test scenario. In the following, we want to assess the performance of the sparse grid interpolants for the optimization problems O1 and O2. For this goal, we create a test scenario [Vale18b] that simulates a pseudo-dynamic sequence of motions by varying the load force and/or the target elbow angle in discrete time steps t as seen in Fig. 7.6A. The test scenario is as follows:

1. Find a feasible initial solution for problem O1 with $F_\text{L}(t_0) := 22\,\text{N}$ and $\theta^*(t_0) := 75°$.
2. Apply O1 with $F_\text{L}(t_1) := 22\,\text{N}$ and $\theta^*(t_1) := 75°$.
3. Apply O2 with $F_\text{L}(t_2) := 22\,\text{N}$ and $\theta^*(t_2) := 60°$ (changed target angle).
4. Apply O2 with $F_\text{L}(t_3) := 30\,\text{N}$ and $\theta^*(t_3) := 60°$ (changed load).
5. Apply O2 with $F_\text{L}(t_4) := 40\,\text{N}$ and $\theta^*(t_4) := 50°$ (changed load and target angle).

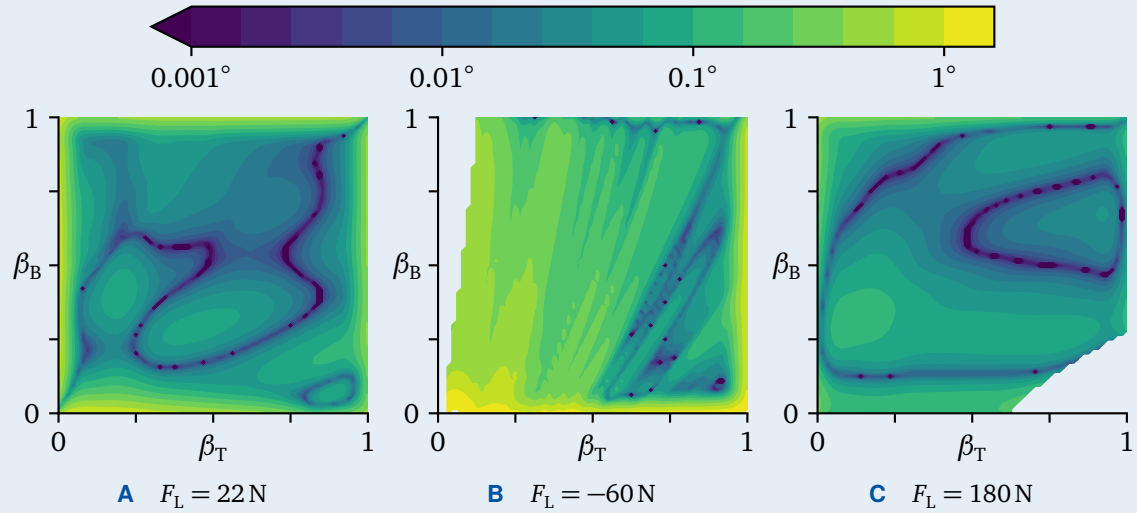


FIGURE 7.5 Absolute error $|\theta_{F_L}^{\text{ref}} - \theta_{F_L}^{s,p,cc}|$ of the equilibrium elbow angle for modified hierarchical cubic Clenshaw–Curtis B-splines ($p = 3$) for different loads F_L. In the empty areas, at least one of $\theta_{F_L}^{\text{ref}}$ and $\theta_{F_L}^{s,p,cc}$ is not well-defined (see Eq. (7.5)).

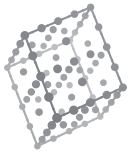
For each of the steps 2 to 5, the activation levels β_T, β_B obtained in the previous step (i.e., either the feasible initial solution of step 1 or the optimal solution of steps 2 to 4) are used as the input of the optimization problem O1 or O2. The feasible initial solution in step 1 is determined as explained in Sec. 5.1.3.

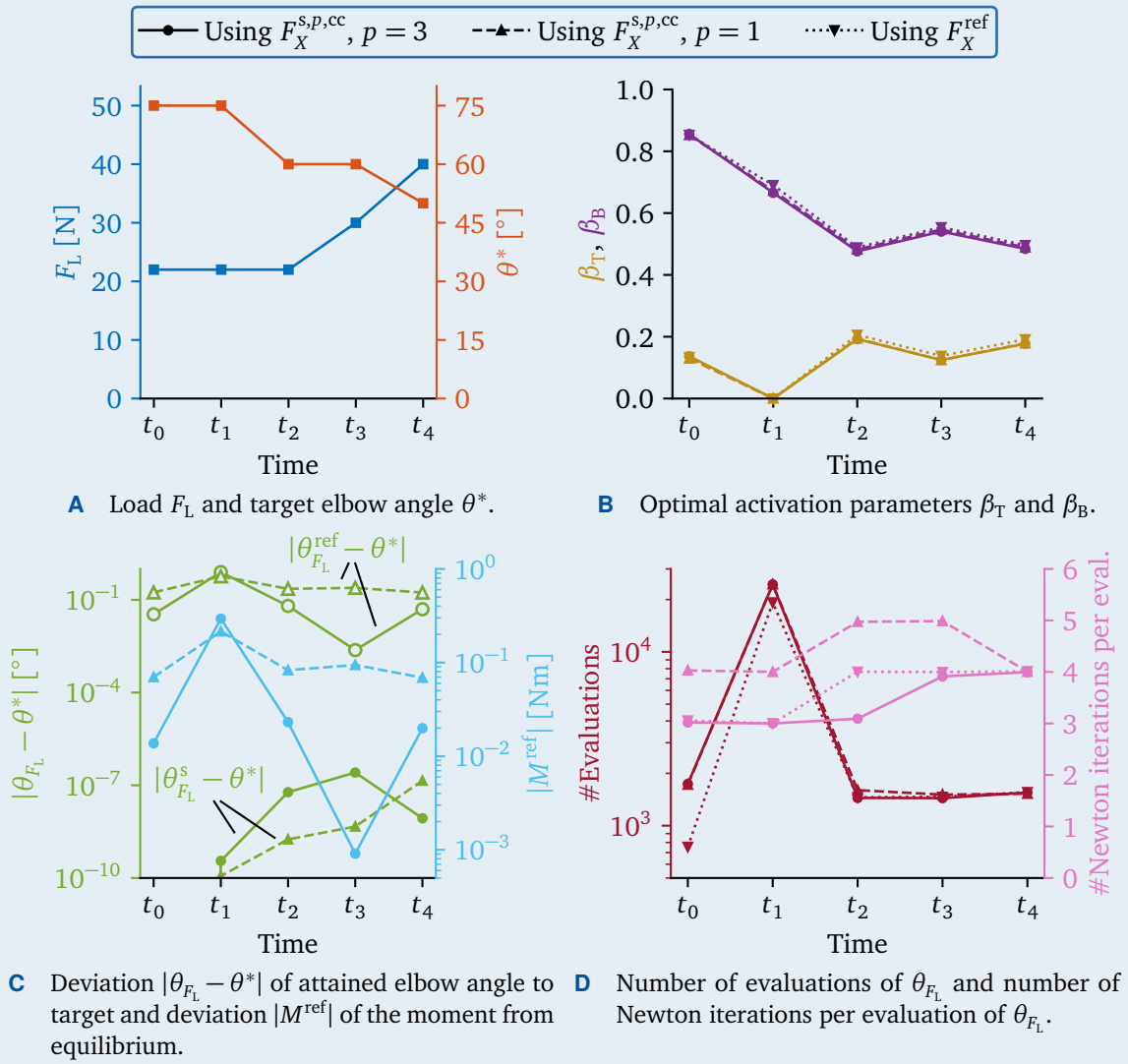
Solutions of problem O1. We note that independently of F_L and θ^*, every solution (β_T, β_B) of problem O1 will be on the boundary part of the domain $[\mathbf{0}, \mathbf{1}]$, on which at least one activation parameter vanishes, i.e.,

$$\{(\beta_T, \beta_B) \in [\mathbf{0}, \mathbf{1}] \mid (\beta_T = 0) \vee (\beta_B = 0)\}. \tag{7.16}$$

The reason is that the two muscles triceps and biceps are antagonistic (see Sec. 7.1.1), meaning that they work against each other. If both $\beta_T > 0$ and $\beta_B > 0$, then the body will waste energy, as the same target elbow angle can be attained by reducing both β_T and β_B simultaneously, thus requiring less energy. A visual example for this is Fig. 7.3, where the contour lines generally go from the bottom left (small β_T, β_B) to the top right (large β_T, β_B). This issue may be prevented by either more complicated musculoskeletal models with more than two muscles or different optimization problems such as problem O2, where the objective function differs.

Plots of optimization results. Figures 7.6B to 7.6D show the results of the test scenario using the muscle forces $F_X^{s,p,cc}$ obtained by interpolating with modified hierarchical cubic





A Load F_L and target elbow angle θ^*.

B Optimal activation parameters β_T and β_B.

C Deviation $|\theta_{F_L} - \theta^*|$ of attained elbow angle to target and deviation $|M^{ref}|$ of the moment from equilibrium.

D Number of evaluations of θ_{F_L} and number of Newton iterations per evaluation of θ_{F_L}.

FIGURE 7.6 Setting (a) of the test scenario and corresponding results (b, c, d).

Clenshaw–Curtis B-splines (solid lines, $p = 3$). As comparison, we repeat the solution process with the forces obtained by interpolating with the corresponding hierarchical piecewise linear basis (dashed lines, $p = 1$) and with the reference forces F_X^{ref} (dotted lines). For the piecewise linear basis, we use exactly the same method as for the cubic case (Newton method for $\theta_{F_L}^s$, Augmented Lagrangian with adaptive gradient descent for the solution of problems O1 and O2), although the derivatives of the muscle forces are discontinuous. For the reference forces, we use the fact that the reference surrogates are full grid spline interpolants, which can be explicitly differentiated. Without the full grid interpolants, we would have to approximate the derivatives with finite differences.

Equilibrium elbow angle. In Fig. 7.6B, we see that the activation levels of all three methods are more or less the same. However, Fig. 7.6C reveals that even these small differences lead to deviations of the resulting equilibrium elbow angle to the target angle that differ by up to two orders of magnitude. The two green lines with filled markers at the bottom of Fig. 7.6C show the error of the equilibrium elbow angle $\theta^{s}_{F_L}$ using sparse grid interpolation to the desired target angle θ^*. Unsurprisingly, this error is very small as it is minimized by the optimizer as part of the constraint. The true error, which is obtained by using the reference equilibrium elbow angle $\theta^{\text{ref}}_{F_L}$, is in general much larger (top two green lines in Fig. 7.6C with hollow markers). We see that the cubic B-splines decrease the error by up to two orders of magnitude compared to the piecewise linear basis. There are two reasons for this: First, the error of θ_{F_L} is generally smaller when using higher-order B-splines as we have seen above. Second, higher-order B-splines are continuously differentiable, which makes them suitable for gradient-based optimization. In contrast, the surrogates obtained by piecewise linear interpolation have kinks, which may complicate finding optimal points in the augmented Lagrangian and Newton methods.

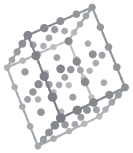
Number of evaluations and Newton iterations. This is supported by Fig. 7.6D, which shows the number of evaluations of θ_{F_L} during the optimization and the average number of Newton iterations per evaluation. While the number of total evaluations is similar for all three methods, the number of required Newton iterations to achieve convergence is in general around 50 % larger for the piecewise linear basis functions.

7.3.5 Spatial Adaptivity

Generation of a spatially adaptive sparse grid. As mentioned in [Vale18b], spatial adaptivity may be employed to reduce the number of necessary muscle force samples even further, especially for more complicated musculoskeletal systems with more parameters. To verify this statement, we remove all grid points $(\theta^{(k,\text{cc})}, \beta_X^{(k,\text{cc})})$ from the regular sparse Clenshaw–Curtis grid that satisfy

$$\frac{|\alpha_{\text{T}}^{(k,p,\text{cc})}|}{\max_{k'} |\alpha_{\text{T}}^{(k',p,\text{cc})}|} < 1\,\% \quad \text{and} \quad \frac{|\alpha_{\text{B}}^{(k,p,\text{cc})}|}{\max_{k'} |\alpha_{\text{B}}^{(k',p,\text{cc})}|} < 1\,\%, \tag{7.17}$$

where $\alpha_X^{(k,p,\text{cc})}$ ($X \in \{\text{T}, \text{B}\}$) is the hierarchical surplus of the basis function $\varphi_k^{p,\text{cc,mod}}$ corresponding to $(\theta^{(k,\text{cc})}, \beta_X^{(k,\text{cc})})$. For higher-dimensional models, one would of course not sample muscle data on a regular sparse grid and then coarsen the data by removing points, but rather use an a posteriori adaptivity criterion to decide which grid points to refine iteratively.



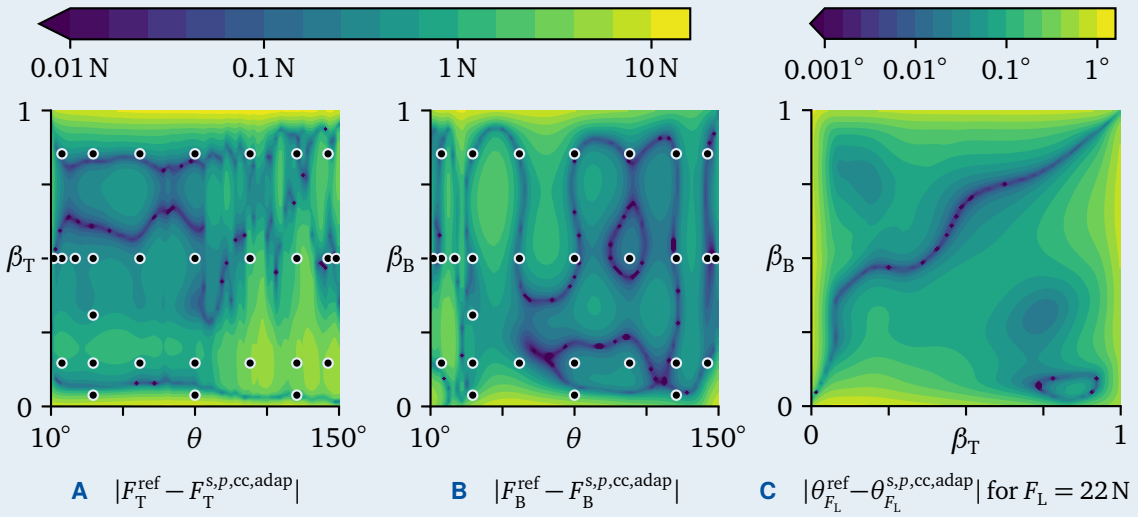


FIGURE 7.7 Errors of muscle forces and equilibrium elbow angle for the spatially adaptive case (modified hierarchical cubic Clenshaw–Curtis B-splines, i.e., $p = 3$) together with the points of the spatially adaptive sparse grid *(dots)*.

Comparison with the regular case. For the cubic case $p = 3$, the resulting force interpolants $F_X^{s,p,cc,adap}$ together with the spatially adaptive sparse grid (which has been coarsened from 49 to 28 points) and equilibrium elbow angle $\theta_{F_L}^{s,p,cc,adap}$ for $F_L = 22\,N$ are shown in Fig. 7.7. The sparse grid is almost dimensionally adaptive, as F_X^{ref} seems to be almost linear in the β_X direction for both $X = T$ and $X = B$. The errors increase slightly: The relative L^2 force errors for (T, B) increase from $(3.31‰, 3.56‰)$ to $(3.36‰, 4.43‰)$, and the absolute L^∞ errors increase from $(10.6\,N, 9.51\,N)$ to $(12.3\,N, 9.57\,N)$. In addition, the relative L^2 and absolute L^∞ errors for θ_{F_L} increase from 2.83 ‰ and 0.886° to 4.12 ‰ and 1.09°, respectively. While all these errors are somewhat larger than for the regular sparse grid, they are still at an acceptable level, but the number of necessary muscle force evaluations is halved compared to the regular case. Additionally, the solution of the test scenario doesn't change significantly due to the similar errors of F_X and θ_{F_L}.

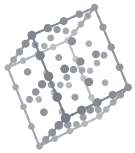
Application 3: Dynamic Portfolio Choice Models

> *The goal is to buy as many iPads as possible during your lifetime.*
>
> — In a talk at the 5th Workshop on Sparse Grids and Applications

Surrogates based on B-splines on sparse grids can also be used for our third application, which stems from mathematical finance. In this application, we optimize financial decisions of an individual over their lifetime in discrete time steps or iterations $t = 0, \ldots, T$ (for example, years $t = 0, \ldots, 80$, where $20 + t$ is the age of the individual), depending on internal and external factors. There are three types of variables:

- *State variables* $\boldsymbol{x}_t$ such as the individual's wealth w_t and their income cannot be controlled directly by the individual. Instead, the individual's decisions may influence the value of state variables of future iterations.[1]
- *Policy variables* $\boldsymbol{y}_t$ such as consumption c_t and the amount of stocks to buy or sell represent the investment decisions the individual can make in each iteration. They

[1] The time t can also be regarded as a state variable.



are subject to specific constraints (for instance, you cannot spend more money than you have, if you do not allow debts).

- *Stochastic variables* $\boldsymbol{\omega}_t$ such as return rates of stocks and inflation cannot be controlled by the individual at all. Therefore, statements about optimal investment conditions are usually made for expected values instead of exact values.

We discretize the state space with a spatially adaptive sparse grid. For each state grid point, an optimization problem over the policy variables has to be solved, where the objective function depends on the expected value over the stochastic variables. By using B-splines as hierarchical basis functions, the accuracy of the interpolants is increased and the explicitly known gradients enable the usage of gradient-based optimization methods, thus accelerating convergence. The process is repeated for each time step, which is possible due to the Bellman principle, which implies that the objective functions occurring at time t depend on the interpolant of the next iteration $t+1$. Hence, the problem has to be solved backward in time via a scheme that closely resembles dynamic programming.

The outline of this chapter is as follows: In Sec. 8.1, we formalize the framework of dynamic portfolio choice models and describe our approach. Afterwards, we explain in Sec. 8.2 the necessary algorithms for implementing the solution of these models. Section 8.3 introduces the transaction costs problem as an example application of the general framework presented in Sec. 8.1. Finally, in Sec. 8.4, we study numerical results.

This chapter is based on a collaboration with Prof. Dr. Raimond Maurer and Peter Schober (both Goethe University Frankfurt, Germany). In previous work, Peter Schober treated the solution of dynamic portfolio choice models with piecewise linear basis functions on spatially adaptive sparse grids [Schob18]. The original contribution of this thesis is the introduction of higher-order B-splines for the solution of these problems. The author of this thesis contributed the methodology of hierarchical B-splines and large parts of the implementation. The contributions of the collaborators at Goethe University Frankfurt are the financial models, the literature review of related work, and the assessment of the quality of results.

8.1 Solving the Bellman Equation

In this section, we give a mathematical framework for dynamic portfolio choice models, briefly mention related literature, and explain where B-splines on sparse grids come into play. Table 8.1 summarizes

IN THIS SECTION

t	Time	w_t	Wealth	u	Utility fcn.
$\boldsymbol{x}_t$	State variables	c_t	Consumption	$\boldsymbol{\psi}_t$	State transition fcn.
$\boldsymbol{y}_t$	Policy variables	b_t	Bond investment	J_t	Value fcn.
$\boldsymbol{\omega}_t$	Stochastic variables	$\tilde{J}_t^s$	Interpolated certainty-equivalent-transf. value fcn.		
γ	Risk aversion	$s_{t,j}$	Stock holding	$\boldsymbol{y}_t^{\text{opt}}$	Optimal policy fcn.
ϱ	Patience factor	$\delta_{t,j}^{\pm}$	Stock buy/sell	$\hat{(\cdot)}$	Normalized quantity
r_t	Bond return rate	$\lambda_{t,j}$	Stock return rate	η_t	Wealth ratio
τ	Transaction cost rate	$\varepsilon_t^{\text{w,Eu}}$	Weighted Euler equation error		

TABLE 8.1 Glossary of the notation for dynamic portfolio choice models.

the symbols that are introduced in this chapter. Rust provides a more detailed introduction to dynamic portfolio choice models [Rus18].

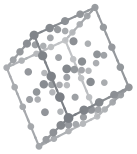
8.1.1 Bellman Equation

Utility maximization. In the following, dynamic portfolio choice models aim to maximize the expected *discounted time-additive utility* over the lifetime of the individual, where the terminal utility is derived solemnly from consumption (i.e., no inheritance motive). If we neglect stochastic factors, then these models solve

$$(\boldsymbol{y}_0^{\text{opt}}, \dots, \boldsymbol{y}_T^{\text{opt}}) = \operatorname*{arg\,max}_{\boldsymbol{y}_0,\dots,\boldsymbol{y}_T} \sum_{t=0}^{T} \varrho^t u(c_t(\boldsymbol{x}_t, \boldsymbol{y}_t)) \quad \text{s.t. specific constraints.} \tag{8.1}$$

Here, $\boldsymbol{x}_t \in [\mathbf{0}, \mathbf{1}] \subseteq \mathbb{R}^d$ and $\boldsymbol{y}_t \in \mathbb{R}^{m_y}$ are the state[2] and policy of time $t = 0, \dots, T$, respectively. The constraints ensure that for instance, we do not spend more money than we actually have. Starting from a given initial state $\boldsymbol{x}_0$, the state $\boldsymbol{x}_{t+1}$ of time $t+1$ can be computed from $\boldsymbol{x}_0$ and $\boldsymbol{y}_0, \dots, \boldsymbol{y}_t$ with a *state transition function* $(\boldsymbol{x}_t, \boldsymbol{y}_t) \mapsto \boldsymbol{x}_{t+1}$. As shown in Fig. 8.1, in each time step, a fraction of the available wealth is consumed *(consumption c_t)*, which can be computed from the state $\boldsymbol{x}_t$ and the policy $\boldsymbol{y}_t$. The individuals rate the consumption with a *utility function* $u(c_t)$. A common choice for u is the *constant relative risk aversion (CRRA) utility* $u(c_t) := c_t^{1-\gamma}/(1-\gamma)$ with the *risk aversion* $\gamma \in \mathbb{R} \setminus \{1\}$. Positive and negative values of γ correspond to risk-averse and risk-affine individuals, respectively. The factor $\varrho \in]0, 1]$ is the *patience* or *time discount factor*.

[2] We assume that each state variable $x_{t,o}$ $(o = 1, \dots, d)$ is bounded, since the state space will be discretized with sparse grids. Without loss of generality, we may then assume that $\boldsymbol{x}_t \in [\mathbf{0}, \mathbf{1}]$. If some state variables are unbounded in reality, then extrapolation is necessary, which will be explained in Sec. 8.2.5.



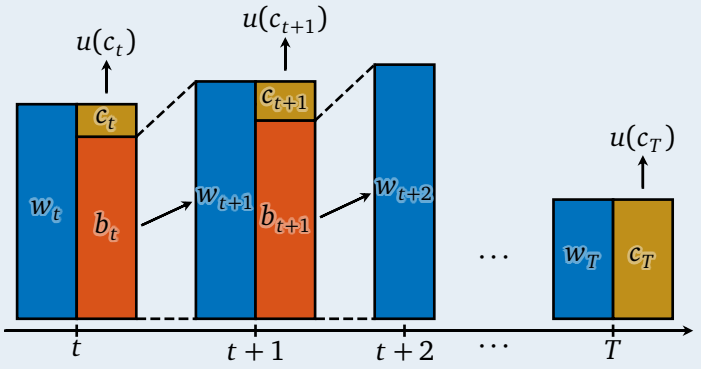


FIGURE 8.1
Example of a dynamic portfolio choice model. The available wealth w_t is either invested into risk-free bonds (b_t) or consumed (c_t), resulting in utility $u(c_t)$. In the last time step T *(far right)*, the optimal solution is to consume the whole wealth, if we do not take inheritance into account.

Limitations of naive utility maximization. When solving the utility maximization problem in Eq. (8.1), there are two issues. First, solving Eq. (8.1) for all times t at once implies solving a $(T+1)m_y$-dimensional optimization problem, which is usually computationally infeasible. Second, Eq. (8.1) does not take stochastic variables $\boldsymbol{\omega}_t$ such as stock return rates into account. These variables influence the state transition, i.e., $(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t) \mapsto \boldsymbol{x}_{t+1}$. Consequently, $\boldsymbol{x}_{t+1}$ cannot be computed from $\boldsymbol{x}_0$ and $\boldsymbol{y}_0, \dots, \boldsymbol{y}_t$ alone, which complicates the solution of Eq. (8.1) even for expected values.

Bellman principle. To resolve the first issue, Bellman's principle of optimality [Bel57] can be applied to problems like Eq. (8.1) that are said to have *optimal substructure*. The principle states that the optimal policy for all times $t = 0, \dots, T$ is also optimal with respect to $t = 1, \dots, T$, i.e.,

$$\max_{\boldsymbol{y}_0,\dots,\boldsymbol{y}_T} \sum_{t=0}^{T} \varrho^t u(c_t(\boldsymbol{x}_t, \boldsymbol{y}_t)) = \max_{\boldsymbol{y}_0} \left(u(c_0(\boldsymbol{x}_0, \boldsymbol{y}_0)) + \varrho \max_{\boldsymbol{y}_1,\dots,\boldsymbol{y}_T} \sum_{t=1}^{T} \varrho^{t-1} u(c_t(\boldsymbol{x}_t, \boldsymbol{y}_t)) \right), \tag{8.2}$$

where we omitted the constraints for brevity. The inner maximum problem over $\boldsymbol{y}_1, \dots, \boldsymbol{y}_T$ has the same structure as the problem on the left-hand side (LHS). With the *value function* $J_t \colon [\mathbf{0}, \mathbf{1}] \to \mathbb{R}$, $J_t(\boldsymbol{x}_t) := \max_{\boldsymbol{y}_t,\dots,\boldsymbol{y}_T} \sum_{t'=t}^{T} \varrho^{t'-t} u(c_{t'}(\boldsymbol{x}_{t'}, \boldsymbol{y}_{t'}))$, this can be rewritten as

$$J_0(\boldsymbol{x}_0) = \max_{\boldsymbol{y}_0} \big(u(c_0(\boldsymbol{x}_0, \boldsymbol{y}_0)) + \varrho J_1(\boldsymbol{x}_1) \big) \quad \text{s.t. specific constraints,} \tag{8.3}$$

where $\boldsymbol{x}_1$ is the result of the state transition starting from $(\boldsymbol{x}_0, \boldsymbol{y}_0)$.

General Bellman equation. If we formulate Eq. (8.3) for arbitrary times t and consider constraints, state transition, and stochastic variables, we obtain the *Bellman equation:*

$$J_t(\boldsymbol{x}_t) = \max_{\boldsymbol{y}_t} \big(u(c_t(\boldsymbol{x}_t, \boldsymbol{y}_t)) + \varrho \mathbb{E}_t\big[J_{t+1}(\boldsymbol{\psi}_t(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t)) \big] \big), \quad t = 0, \dots, T, \tag{8.4a}$$

$$\boldsymbol{y}_t \in \mathbb{R}^{m_y} \text{ s.t. } \boldsymbol{g}_t(\boldsymbol{x}_t, \boldsymbol{y}_t) \leq \mathbf{0}, \tag{8.4b}$$

where $J_{T+1} :\equiv 0$ for simplicity, $\boldsymbol{\psi}_t : [\mathbf{0},\mathbf{1}] \times \mathbb{R}^{m_y} \times \Omega \to [\mathbf{0},\mathbf{1}]$, $(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t) \mapsto \boldsymbol{x}_{t+1}$, is the *state transition function,* $\boldsymbol{g}_t : [\mathbf{0},\mathbf{1}] \times \mathbb{R}^{m_y} \to \mathbb{R}^{m_g}$ is the *constraint function,* and

$$\mathbb{E}_t\big[J_{t+1}(\boldsymbol{\psi}_t(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t))\big] := \int_\Omega J_{t+1}(\boldsymbol{\psi}_t(\boldsymbol{x}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t)) P_{t,\boldsymbol{\omega}}(\boldsymbol{\omega}_t)\,\mathrm{d}\boldsymbol{\omega}_t \tag{8.5}$$

with the probability density function $P_{t,\boldsymbol{\omega}} : \Omega \to \mathbb{R}_{\geq 0}$ of $\boldsymbol{\omega}_t$.[3] We denote the location of the maximum of (8.4) as the optimal policy $\boldsymbol{y}_t^{\mathrm{opt}}$, which may be regarded as a function $\boldsymbol{y}_t^{\mathrm{opt}} : [\mathbf{0},\mathbf{1}] \to \mathbb{R}^{m_y}$, $\boldsymbol{x}_t \mapsto \boldsymbol{y}_t^{\mathrm{opt}}(\boldsymbol{x}_t)$.

Dynamic programming scheme. The Bellman equation (8.4) can be solved backwards in time with a dynamic programming scheme. Starting from the solution J_T and $\boldsymbol{y}_T^{\mathrm{opt}}$ of time T, which is determined by maximizing the utility for the terminal time step, we can determine J_t and $\boldsymbol{y}_t^{\mathrm{opt}}$ from J_{t+1} and $\boldsymbol{y}_{t+1}^{\mathrm{opt}}$ for $t = T-1, T-2, \ldots, 0$ with the Bellman equation. This way, we only have to solve $T+1$ separate m_y-dimensional optimization problems instead of a single large $(T+1)m_y$-dimensional problem. Often, the terminal solutions J_T and $\boldsymbol{y}_T^{\mathrm{opt}}$ are explicitly known. In our case, the optimal terminal solution is to consume the whole wealth w_T (see Fig. 8.1).

Implementation and interpolation. For the implementation of (8.4), we discretize the state space $[\mathbf{0},\mathbf{1}]$ into N_t grid points $\boldsymbol{x}_t^{(k)}$, $k = 1, \ldots, N_t$, and we tabulate the values of J_t and $\boldsymbol{y}_t^{\mathrm{opt}}$ at $\boldsymbol{x}_t^{(k)}$ for all $t = 0, \ldots, T$ and $k = 1, \ldots, N_t$. However, in general, the next state $\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t)$ does not correspond to a grid point $\boldsymbol{x}_{t+1}^{(k')}$, which means that we cannot lookup the value of J_{t+1} at $\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t)$. Therefore, we have to interpolate J_{t+1} at the grid points, obtaining the interpolant J_{t+1}^{s} as a result:

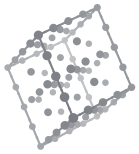
$$J_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)}) = \max_{\boldsymbol{y}_t} \big(u(c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t)) + \varrho\, \mathbb{E}_t\big[J_{t+1}^{\mathrm{s}}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t))\big]\big), \quad k = 1, \ldots, N_t, \tag{8.6}$$

where $J_{T+1}^{\mathrm{s}} :\equiv 0$ for simplicity. As J_{t+1}^{s} on the right-hand side (RHS) is only an approximation to J_{t+1}, the values $J_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)})$ on the LHS are approximations, too. Since we are mainly interested in the optimal policy decisions $\boldsymbol{y}_t^{\mathrm{opt}}$, we have to interpolate them as well, i.e.,

$$\boldsymbol{y}_t^{\mathrm{opt,s}}(\boldsymbol{x}_t^{(k)}) = \operatorname*{\mathbf{arg\,max}}_{\boldsymbol{y}_t} \big(u(c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t)) + \varrho\, \mathbb{E}_t\big[J_{t+1}^{\mathrm{s}}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t))\big]\big). \tag{8.7}$$

Note that the employed grids for $\boldsymbol{y}_t^{\mathrm{opt,s}}$ may be different from the grids for J_t^{s}.

[3] While the state $\boldsymbol{x}_t \in [\mathbf{0},\mathbf{1}]$ is continuous in this thesis, *Markov-chain discrete states* $\boldsymbol{\theta}_t \in \Theta$ such as alive/dead (i.e., Θ is the Cartesian product of finite sets) can be incorporated into (8.4). The objective function of $J_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t)$ then equals $u(c_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t, \boldsymbol{y}_t)) + \varrho\, \mathbb{E}_t\big[J_{t+1}(\boldsymbol{\psi}_t(\boldsymbol{x}_t, \boldsymbol{\theta}_t, \boldsymbol{y}_t, \boldsymbol{\omega}_t), \boldsymbol{\theta}_{t+1}) \mid \boldsymbol{\theta}_t\big]$.



8.1.2 Solution with B-Spline Surrogates on Sparse Grids

Sparse grids for dynamic models and related work. As interpolation approaches for J_t based on full grids suffer from the curse of dimensionality, we want to use interpolation on spatially adaptive sparse grids instead. Recently, sparse grids have found increasing interest in the solution of dynamic models in finance [Bru17; Jud14; Schob18; Win10]. For example in [Bru17], discrete choices in the value iteration are computed using piecewise linear basis functions on spatially adaptive sparse grids. Schober employs spatially adaptive sparse grids for the interpolation of dynamic portfolio choice models, but uses piecewise linear basis functions [Schob18]. Judd et al. use global polynomials on sparse Clenshaw–Curtis grids for the interpolation of higher-dimensional economic models [Jud14].

B-splines on sparse grids for dynamic portfolio choice models. The shortcomings of the two approaches of piecewise linear functions [Bru17; Schob18] or global polynomials [Jud14] are evident: Piecewise linear functions are not continuously differentiable, impeding convergence of interpolation errors (see Sec. 5.4.1) and prohibiting the use of gradient-based optimization methods to solve Eq. (8.6). The reason for the latter statement is that gradient-based optimizers require the derivatives of the objective function of Eq. (8.6) with respect to the entries $y_{t,j}$ of $\boldsymbol{y}_t$ $(j = 1, \ldots, m_y)$, i.e.,

$$
\begin{aligned}
&u'(c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t)) \frac{\partial}{\partial y_{t,j}} c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t) \\
&+ \varrho \mathbb{E}_t \left[\left(\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} J_{t+1}^s(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t)) \right)^{\mathrm{T}} \frac{\partial}{\partial y_{t,j}} \boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t) \right],
\end{aligned} \tag{8.8}
$$

which involves the gradient $\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} J_{t+1}^s$ of the value function interpolant J_{t+1}^s. Gradient-based optimization methods do not converge fast if this gradient is discontinuous. Moreover, piecewise linear basis functions introduce many additional local minima. In contrast, global polynomials only work well on Clenshaw–Curtis grids with Chebyshev-distributed nodes due to Runge's phenomenon.

In the following, we use higher-order B-splines as basis functions for the interpolation of J_t and $\boldsymbol{y}_t^{\mathrm{opt}}$. This method has two advantages: First, B-splines of degree $p > 1$ are continuously differentiable, increasing the order of convergence and enabling gradient-based optimization for solving Eq. (8.6). Second, B-splines are defined for arbitrary knot sequences, leading to a greater flexibility when compared to global polynomials.

8.2 Algorithms

This section gives an overview of the algorithms that we use to implement the solution process after discretization of the Bellman equation (8.6). In the following, we assume that the probability density functions of the stochastic variables are known.

IN THIS SECTION

8.2.1 General Structure

The general approach to solve dynamic portfolio choice models is as follows:

1. Generation of value function interpolants J_t^s
2. Generation of optimal policy interpolants $\boldsymbol{y}_t^{\text{opt,s}}$
3. Post-processing, e.g., Monte Carlo simulation

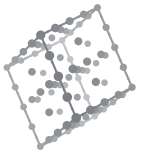
The separation of the solution processes for the value function interpolants J_t^s and the optimal policy interpolants $\boldsymbol{y}_t^{\text{opt,s}}$ enables the generation of different spatially adaptive sparse grids for the value function and the optimal policies. This is useful if the shapes of value function and optimal policies have different characteristics.

In the following Sections 8.2.2 to 8.2.6, we describe the algorithmic details of `solveValueFunction` (step 1). The treatment of the other steps `solvePolicy` (step 2) and post-processing (step 3) follows with Sec. 8.2.7 and Sec. 8.2.8, respectively.

We track two interpolants $J_t^{s,1}$ and $J_t^{s,p}$ for each $t = 0, \ldots, T$. The former interpolates value function data at the grid points with the hierarchical piecewise linear basis (used for the surplus-based grid generation), while the latter interpolates the same data with hierarchical B-splines of degree $p > 1$. Each $J_t^{s,*}$ $(* \in \{1, p\})$ additionally stores the grid points $\boldsymbol{x}_t^{(k)}$ and the optimal policies $\boldsymbol{y}_t^{\text{opt,s}}(\boldsymbol{x}_t^{(k)})$ at the grid points $(k = 1, \ldots, N_t)$. For simplicity, we do not pass them as separate data to the algorithms.

8.2.2 Solution for the Value Function

`solveValueFunction` algorithm. Algorithm 8.1 shows `solveValueFunction`, which generates the value function interpolants $J_t^{s,1}$ and $J_t^{s,p}$ $(t = 0, \ldots, T)$. The algorithm follows a simple optimize–refine–interpolate scheme, which is visualized in Fig. 8.2: First, the Bellman equation (8.6) is solved on an initial sparse grid (`optimize`). Then, we `re`–



```
1 function (J_t^{s,p})_{t=0,...,T} = solveValueFunction()
2   J_{T+1}^{s,p} ← ∅                          ⇝ dummy variable (is not used)
3   for t = T, T−1, ..., 0 do
4     J_t^{s,1} ← Initial regular sparse grid with no values
5     J_t^{s,1} ← optimize(t, J_t^{s,1}, J_{t+1}^{s,p})
6     J_t^{s,1} ← refine(t, J_t^{s,1}, J_{t+1}^{s,p})
7     J_t^{s,p} ← interpolate(J_t^{s,1})
```

ALGORITHM 8.1 Generation of value function interpolants. The output is the higher-order B-spline interpolant $J_t^{s,p}$ for all $t = 0, \ldots, T$.

`fine` the grid spatially adaptively. Finally, the resulting grid point data are `interpolated` with hierarchical higher-order B-splines.

At the beginning of every iteration t, the grid of the piecewise linear interpolant is reset to an initial, possibly regular sparse grid. It would also be possible to reuse the grid from the previous iteration $t+1$. Nevertheless, the results we then obtain become worse, likely due to the different characteristics of $J_t^{s,1}$ for different t (e.g., kinks).

The higher-order B-spline interpolant $J_{t+1}^{s,p}$ of the previous iteration $t+1$ is used for the RHS of the Bellman equation (8.6), if $t < T$. In the first iteration $t = T$, there is no such interpolant. However, the terminal solution J_T is usually a known function.

8.2.3 Optimization

`optimize` algorithm. The `optimize` step is given as Alg. 8.2. The grid of the argument $J_t^{s,1}$ is some spatially adaptive sparse grid $\Omega_t^s = \{\boldsymbol{x}_t^{(k)} \mid k = 1, \ldots, N_t\}$, where the function values $J_t^{s,1}(\boldsymbol{x}_t^{(k)})$ may already be known for some of the grid points $\boldsymbol{x}_t^{(k)}$, if `optimize` is called from within `refine`. The function `optimize` computes the missing value function values. For $t = T$, we assume that the terminal solution J_T can be computed by some function `computeKnownTerminalSolution`.[4] Otherwise, for $t < T$, we solve the Bellman equation (8.6) by using the higher-order B-spline interpolant $J_{t+1}^{s,p}$ of the previous iteration $t+1$ (`optimizeSinglePoint`). The computations for the different $\boldsymbol{x}_t^{(k)}$ are independent of each other, which means that they can be computed in parallel [Hor16].[5] After generating all missing data, we update the hierarchical surpluses of the piecewise linear interpolant $J_t^{s,1}$ to interpolate the new data at all grid points of Ω_t^s.

[4] In any case, the terminal solution may be computed as the solution of the corresponding single-time optimization problem, e.g., $J_T(\boldsymbol{x}_T^{(k)}) = \max_{\boldsymbol{y}_T} u(c_T(\boldsymbol{x}_T^{(k)}, \boldsymbol{y}_T))$.

[5] Such a problem is usually referred to as *embarrassingly parallel*.

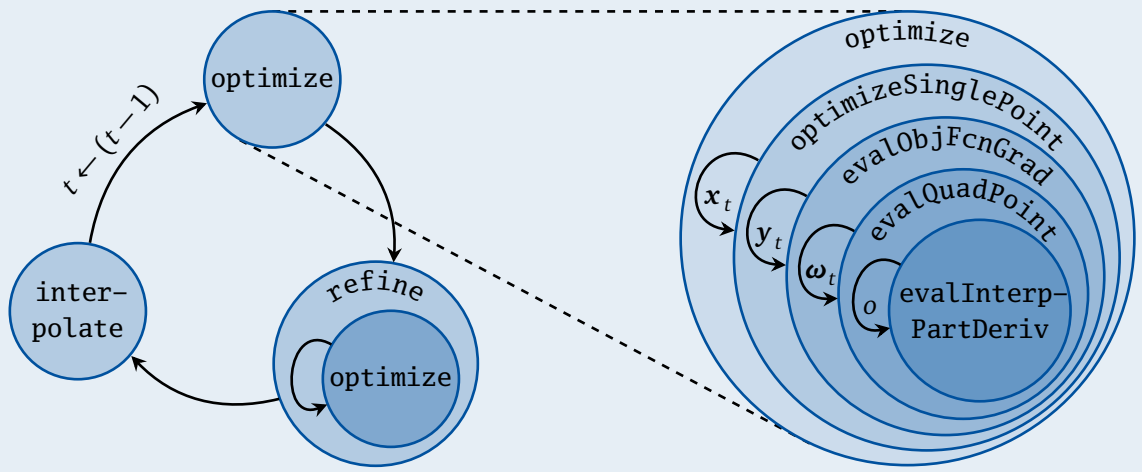
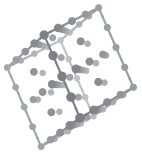


FIGURE 8.2 Scheme of the generation of value function interpolants with `solveValueFunction` (Alg. 8.1, *left*), which repeatedly calls the `optimize` routine (Alg. 8.2, *right*), which in turn consists of various sub-functions. The function `optimize` iterates over all state grid points $\boldsymbol{x}_t = \boldsymbol{x}_t^{(k)}$ $(k = 1, \ldots, N_t)$ and calls `optimizeSinglePoint` for each point. The optimization method evaluates the objective function and its gradient at a sequence of different policy points $\boldsymbol{y}_t$ to find $\boldsymbol{y}_t^{\text{opt,s}}(\boldsymbol{x}_t^{(k)})$. This evaluation (denoted by `evalObjFcnGrad`) has to compute the expectation in Eq. (8.6), which is done using a quadrature rule. For every quadrature point $\boldsymbol{\omega}_t = \boldsymbol{\omega}_t^{(j)}$ $(j = 1, \ldots, m_\zeta)$, `evalQuadPoint` computes the corresponding value of the expression in the expectation. Finally, `evalInterpPartDeriv` evaluates the interpolant $J_{t+1}^{\text{s,p}}$ and its partial derivatives, for which we have to loop over the state dimensions $o = 1, \ldots, d$.

```
1 function J_t^{s,1} = optimize(t, J_t^{s,1}, J_{t+1}^{s,p})
2   (x_t^{(k)})_{k=1,...,N_t} ← grid of J_t^{s,1}
3   for k = 1,...,N_t do
4     if J_t^{s,1}(x_t^{(k)}) not previously computed then
5       if t = T then J_T^{s,1}(x_T^{(k)}) ← computeKnownTerminalSolution(x_T^{(k)})
6       else J_t^{s,1}(x_t^{(k)}) ← optimizeSinglePoint(t, x_t^{(k)}, J_{t+1}^{s,p})
7   Re-interpolate (J_t^{s,1}(x_t^{(k)}))_{k=1,...,N_t} with piecewise linear functions
```

ALGORITHM 8.2 Evaluation of the value function at all grid points $\boldsymbol{x}_t^{(k)}$ of $J_t^{\text{s,1}}$ at which the value function has not been evaluated yet. Inputs are the time t, the piecewise linear interpolant $J_t^{\text{s,1}}$ of the current iteration t (with the underlying sparse grid and corresponding function values, possibly unset), and the higher-order B-spline interpolant $J_{t+1}^{\text{s,p}}$ of the previous iteration $t+1$ (not used if $t = T$). The output is the updated piecewise linear interpolant $J_t^{\text{s,1}}$, where all missing function values at grid points have been computed.



Certainty-equivalent transformation. For utility functions of CRRA-type, i.e., of the form $u(c_t) = c_t^{1-\gamma}/(1-\gamma)$, the curvature of the objective function in the Bellman equation (8.6) can be very high (depending on the risk aversion parameter γ), which may impede convergence of the optimizer. As a remedy, we transform the value function J_t^s with the *certainty-equivalent transformation* $J_t^s \mapsto \tilde{J}_t^s := ((1-\gamma)J_t^s)^{1/(1-\gamma)}$ if $\gamma > 1$. Equation (8.6) then becomes $\tilde{J}_T^s(\boldsymbol{x}_T^{(k)}) = \max_{\boldsymbol{y}_T} c_T(\boldsymbol{x}_T^{(k)}, \boldsymbol{y}_T)$ for $t = T$ and

$$\tilde{J}_t^s(\boldsymbol{x}_t^{(k)}) = \max_{\boldsymbol{y}_t} \left(\left(c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t)^{1-\gamma} + \varrho \mathbb{E}_t \left[\left(\tilde{J}_{t+1}^s(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t)) \right)^{1-\gamma} \right] \right)^{1/(1-\gamma)} \right) \tag{8.9}$$

for $t < T$, since for $\gamma > 1$, $(\cdot)^{1/(1-\gamma)}$ is strictly monotonously decreasing and $(1-\gamma) < 0$. The notation in the remainder of this section does not distinguish between $J_t^{s,*}$ and $\tilde{J}_t^{s,*}$ and uses $J_t^{s,*}$ for both if it is not relevant whether the value function is transformed.

8.2.4 Quadrature

We need to approximate the expectation in Eq. (8.9) by quadrature,

$$\mathbb{E}_t \left[\left(\tilde{J}_{t+1}^{s,p}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t)) \right)^{1-\gamma} \right] \approx \sum_{j=1}^{m_\zeta} \zeta_t^{(j)} \left(\tilde{J}_{t+1}^{s,p}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t^{(j)})) \right)^{1-\gamma}, \tag{8.10}$$

for some weights $\zeta_t^{(j)} \in \mathbb{R}$ and nodes $\boldsymbol{\omega}_t^{(j)} \in \Omega$ $(j = 1, \ldots, m_\zeta)$. Since the stochastic domain $\Omega \subseteq \mathbb{R}^{m_\omega}$ might be high-dimensional as well, full grid quadrature rules suffer from the curse of dimensionality. Therefore, we use sparse grid quadrature rules based on Gauss–Hermite quadrature [Ger98; Hor16]. Note that this sparse grid in the stochastic space Ω is independent of the sparse grid in the state space $[\mathbf{0}, \mathbf{1}]$. However, it would also be feasible to employ Monte Carlo quadrature, albeit usually far more expensive.

8.2.5 Interpolation and Extrapolation

Sparse grid interpolation. As already mentioned, $J_t^{s,1}$ is constructed as the sparse grid interpolant of the grid data $\boldsymbol{x}_t^{(k)}$ $(k = 1, \ldots, N_t)$ using the hierarchical piecewise linear basis. For $J_t^{s,p}$, we use cubic hierarchical weakly fundamental not-a-knot splines (see Sec. 4.5.4). The not-a-knot boundary conditions help to decrease the interpolation error (see Sec. 5.4.1), while the weakly fundamental property eases the hierarchization complexity by enabling us to use the unidirectional principle (see Sections 4.5 and 5.4.2).

Extrapolation. Unfortunately, for many dynamic portfolio choice models, the state transition is not a function $\boldsymbol{\psi}_t : [\mathbf{0}, \mathbf{1}] \times \mathbb{R}^{m_y} \times \Omega \to [\mathbf{0}, \mathbf{1}]$, especially if the state space is actually unbounded. It may then happen that $\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t^{(j)}) \notin [\mathbf{0}, \mathbf{1}]$ for some quadrature nodes $\boldsymbol{\omega}_t^{(j)} \in \Omega$ in Eq. (8.10). Hence, we might not be able to evaluate the value function interpolant $J_{t+1}^{\mathrm{s},p}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t^{(j)}))$, as it is only defined on $[\mathbf{0}, \mathbf{1}]$. Scaling of the domain is not an option due to the dynamic nature of the problem.

Instead, we extend the interpolant $J_{t+1}^{\mathrm{s},p}$ to $\mathbb{R}^d$ by an extrapolation method based on Taylor approximation. First, we crop the evaluation point $\boldsymbol{x}_{t+1} \in \mathbb{R}^d \setminus [\mathbf{0}, \mathbf{1}]$ to a point $\boldsymbol{x}_{t+1}^{\mathrm{in}} = \boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t^{(j)}) \in [\mathbf{0}, \mathbf{1}]$ with $\boldsymbol{x}_{t+1}^{\mathrm{in}} := \mathbf{min}(\mathbf{max}(\boldsymbol{x}_{t+1}, \mathbf{0}), \mathbf{1})$ (component-wise minimum/maximum). The extrapolation type, which may be `constant`, `linear`, and `quadratic`, determines the degree of the Taylor approximation:

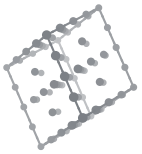
$$
\begin{aligned}
J_{t+1}^{\mathrm{s},p}(\boldsymbol{x}_{t+1}) \approx\ & J_{t+1}^{\mathrm{s},p}(\boldsymbol{x}_{t+1}^{\mathrm{in}}) + (\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} J_{t+1}^{\mathrm{s},p}(\boldsymbol{x}_{t+1}^{\mathrm{in}}))^{\mathrm{T}}(\boldsymbol{x}_{t+1} - \boldsymbol{x}_{t+1}^{\mathrm{in}}) \\
& + (\boldsymbol{x}_{t+1} - \boldsymbol{x}_{t+1}^{\mathrm{in}})^{\mathrm{T}}(\boldsymbol{\nabla}^2_{\boldsymbol{x}_{t+1}} J_{t+1}^{\mathrm{s},p}(\boldsymbol{x}_{t+1}^{\mathrm{in}}))(\boldsymbol{x}_{t+1} - \boldsymbol{x}_{t+1}^{\mathrm{in}}),
\end{aligned} \tag{8.11}
$$

where `constant` and `linear` only use the first summand and first two summands, respectively. Since hierarchical B-splines enable us to exactly and efficiently compute the gradient $\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} J_{t+1}^{\mathrm{s},p}$ and the Hessian $\boldsymbol{\nabla}^2_{\boldsymbol{x}_{t+1}} J_{t+1}^{\mathrm{s},p}$, we do not have to approximate the derivatives with finite differences.

8.2.6 Grid Generation

`refine` algorithm. Algorithm 8.3 shows how to generate the spatially adaptive sparse grid in `solveValueFunction` (Alg. 8.1). The underlying criterion is the common surplus-based refinement criterion [Pfl13]. As for the application in topology optimization (see Chap. 6), we use the piecewise linear interpolant for the surplus-based grid generation, since the surpluses are easier to compute in the piecewise linear case, and they are more meaningful due to the integral representation formula (2.25). Parameters for Alg. 8.3 are the tolerance $\kappa_t \in \mathbb{R}_{\geq 0}$, by which the set of grid points to be refined is determined, and the number $q_t \in \mathbb{N}_0$ of refinement iterations. These parameters may depend on the time t, since it might be beneficial to change the adaptivity of the grid over time.

Gradient grids. The classical surplus-refinement criterion focuses on regions where the mixed second derivative $\frac{\partial^{2d}}{\partial x_{t,1}^2 \cdots \partial x_{t,d}^2} J_t^{\mathrm{s},1}$ of $J_t^{\mathrm{s},1}$ has large absolute values, i.e., where $J_t^{\mathrm{s},1}$ has large high-frequency oscillations. In gradient-based optimization, it might be advisable to apply this criterion also to the partial derivatives $\frac{\partial}{\partial x_{t,o}} J_t^{\mathrm{s},1}$ of $J_t^{\mathrm{s},1}$ $(o = 1, \ldots, d)$, since the optimizer depends on the accuracy of the gradient. In this case, we have to track in Alg. 8.1 additional sparse grid interpolants for every partial derivative $\frac{\partial}{\partial x_{t,o}} J_t^{\mathrm{s},1}$ that



```
function J_t^{s,1} = refine(t, J_t^{s,1}, J_{t+1}^{s,p})
  for j = 1,...,q_t do
    N_t ← number of grid points of J_t^{s,1}
    for k = 1,...,N_t do α_t^{(k)} ← surplus of x_t^{(k)} in J_t^{s,1}
    K_refine ← {k = 1,...,N_t | |α_t^{(k)}| ≥ κ_t}
    if K_refine = ∅ then break
    Refine all grid points in {x_t^{(k)} | k ∈ K_refine}
    J_t^{s,1} ← optimize(t, J_t^{s,1}, J_{t+1}^{s,p})
```

ALGORITHM 8.3 In-place refinement of the value function $J_t^{s,1}$. Inputs are the time t, the piecewise linear interpolant $J_t^{s,1}$ of the current iteration t, and the higher-order B-spline interpolant $J_{t+1}^{s,p}$ of the previous iteration $t+1$ (not used if $t = T$). The output is the updated piecewise linear interpolant $J_t^{s,1}$ with the refined sparse grid.

is affected by a policy variable. This possibility is omitted from the algorithms in this section, as it would unnecessarily complicate their presentation.

8.2.7 Solution for Optimal Policies

`solvePolicies` algorithm. After explaining the generation of the value function interpolants $J_t^{s,p}$ $(t = 0,\dots,T)$, we move on to step 2 of the general structure of our method (see Sec. 8.2.1), which is the generation of optimal policy interpolants. The corresponding Alg. 8.4 is similar to `solveValueFunction` (Alg. 8.1), except that it operates on the policy instead of the value function interpolants. The functions `optimize`, `refine`, and `interpolate` have been replaced by corresponding policy versions `optimizePolicy`, `refinePolicy`, and `interpolatePolicy` that work very much like their value function counterparts. `optimizePolicy` only has to generate new values if the initial regular sparse grid for the policies is not contained in the grid of $J_t^{s,p}$. The policy grid is then refined and interpolated independently of the value function grid. The iterations over time are independent of each other, which means that they can be parallelized.

8.2.8 Post-Processing

Monte Carlo simulation. There are various ways to assess whether the resulting optimal policy B-spline interpolants $(y_t^{\mathrm{opt},s,p})_{t=0,\dots,T}$ are reasonable. One possibility is a Monte

```
function (y_t^{opt,s})_{t=0,...,T} = solvePolicies((J_t^{s,p})_{t=0,...,T})
  J_{T+1}^{s,p} ← ∅                                  ⇝ dummy variable (is not used)
  for t = 0,...,T do
    y_t^{opt,s,1} ← Initial regular sparse grid, retrieve values from J_t^{s,p}
    y_t^{opt,s,1} ← optimizePolicy(t, y_t^{opt,s,1}, J_{t+1}^{s,p})
    y_t^{opt,s,1} ← refinePolicy(t, y_t^{opt,s,1}, J_{t+1}^{s,p})
    y_t^{opt,s,p} ← interpolatePolicy(y_t^{opt,s,1})
```

ALGORITHM 8.4 Generation of interpolants for optimal policies. The input is the higher-order B-spline interpolant $J_t^{s,p}$ of the value function for all $t = 0, \dots, T$. The output is the higher-order B-spline interpolant $\boldsymbol{y}_t^{\text{opt,s,}p}$ of the optimal policies for all $t = 0, \dots, T$.

Carlo simulation, where we calculate the mean optimal policy

$$\bar{\boldsymbol{y}}_t^{\text{opt}} = \frac{1}{m_{\text{MC}}} \sum_{j=1}^{m_{\text{MC}}} \boldsymbol{y}_{t,(j)} \tag{8.12}$$

for $m_{\text{MC}} \in \mathbb{N}$ individuals. The optimal policies $\boldsymbol{y}_{t,(j)}$ of the individuals ($t = 0, \dots, T$ and $j = 1, \dots, m_{\text{MC}}$) are determined by

$$\boldsymbol{y}_{t,(j)} := \boldsymbol{y}_t^{\text{opt,s,}p}(\boldsymbol{x}_{t,(j)}), \tag{8.13a}$$

$$\boldsymbol{x}_{t,(j)} := \boldsymbol{\psi}_{t-1}(\boldsymbol{x}_{t-1,(j)}, \boldsymbol{y}_{t-1,(j)}, \boldsymbol{\omega}_{t-1,(j)}), \quad t > 0, \qquad \boldsymbol{x}_{0,(j)} \sim P_{0,\boldsymbol{x}}, \tag{8.13b}$$

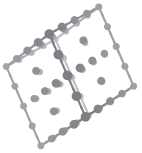
$$\boldsymbol{\omega}_{t,(j)} \sim P_{t,\boldsymbol{\omega}}, \tag{8.13c}$$

i.e., the initial state $\boldsymbol{x}_{0,(j)}$ and the stochastic variables $\boldsymbol{\omega}_{t,(j)}$ are samples of random variables. Monte Carlo simulations enable us to draw macro-economic conclusions, e.g., the evolution of the amount of consumption of the average individual over time.

8.3 Transaction Costs Problem

IN THIS SECTION

Description. In the *transaction costs problem,* the individual can invest their money risk-free in bonds (with a fixed interest rate similar to a bank account) or in $m_s \in \mathbb{N}$ different risk-affected stocks [Schob18]. Every stock transaction, i.e., buy $\delta_{t,j}^+$ or sell $\delta_{t,j}^-$, inflicts transaction costs $\tau \delta_{t,j}^{\pm}$ ($\tau \in \mathbb{R}_{\geq 0}$) proportional to the amount $\delta_{t,j}^{\pm}$ bought or sold ($j = 1, \dots, m_s$). The individual only wants to invest a fixed amount w_0 in stocks, i.e., we omit the individual's income.



8.3.1 Unnormalized Problem

Consumption and state transition. In the following, $s_{t,j}$ denotes the fraction of the total wealth w_t that is invested in the j-th stock. We combine these *stock fractions* $s_{t,j}$ in a vector $\boldsymbol{s}_t := (s_{t,1}, \dots, s_{t,m_s})$; similarly, $\boldsymbol{\delta}_t^{\pm} := (\delta_{t,1}^{\pm}, \dots, \delta_{t,m_s}^{\pm})$ combines buy and sell amounts. Then, the consumption can be computed as a residual variable (i.e., a variable that can be fully computed from $\boldsymbol{x}$ and $\boldsymbol{y}$ and is thus omitted from $\boldsymbol{y}$), which is given by

$$c_t := (1 - \Sigma(\boldsymbol{s}_t))w_t - b_t - (1+\tau)\,\Sigma(\boldsymbol{\delta}_t^{+}) + (1-\tau)\,\Sigma(\boldsymbol{\delta}_t^{-}), \tag{8.14}$$

where $\Sigma(\boldsymbol{a}) := \mathbf{1}^{\mathrm{T}}\boldsymbol{a}$ is the sum of all entries of $\boldsymbol{a}$. The state transition is computed by adding the returns of bonds and stocks:

$$w_{t+1} := b_t r_t + (\boldsymbol{s}_t w_t + \boldsymbol{\delta}_t^{+} - \boldsymbol{\delta}_t^{-})^{\mathrm{T}}\boldsymbol{\lambda}_t, \qquad \boldsymbol{s}_{t+1} := \frac{(\boldsymbol{s}_t w_t + \boldsymbol{\delta}_t^{+} - \boldsymbol{\delta}_t^{-}) \odot \boldsymbol{\lambda}_t}{w_{t+1}}, \tag{8.15}$$

where $r_t \in \mathbb{R}$ is the bond interest rate, $\boldsymbol{\lambda}_t = (\lambda_{t,1}, \dots, \lambda_{t,m_s}) \in \mathbb{R}^{m_s}$ is the vector of (stochastic) stock return rates, and $\odot$ is component-wise multiplication.

8.3.2 Normalization

State transition. The above equations can be normalized with respect to the wealth w_t: By setting $\hat{c}_t := c_t/w_t$, $\hat{b}_t := b_t/w_t$, and $\hat{\boldsymbol{\delta}}_t^{\pm} := \boldsymbol{\delta}_t^{\pm}/w_t$, we obtain

$$\hat{c}_t = (1 - \Sigma(\boldsymbol{s}_t)) - \hat{b}_t - (1+\tau)\,\Sigma(\hat{\boldsymbol{\delta}}_t^{+}) + (1-\tau)\,\Sigma(\hat{\boldsymbol{\delta}}_t^{-}), \tag{8.16a}$$

$$\eta_{t+1} := \hat{b}_t r_t + (\boldsymbol{s}_t + \hat{\boldsymbol{\delta}}_t^{+} - \hat{\boldsymbol{\delta}}_t^{-})^{\mathrm{T}}\boldsymbol{\lambda}_t, \qquad (= w_{t+1}/w_t) \tag{8.16b}$$

$$\boldsymbol{s}_{t+1} = \frac{(\boldsymbol{s}_t + \hat{\boldsymbol{\delta}}_t^{+} - \hat{\boldsymbol{\delta}}_t^{-}) \odot \boldsymbol{\lambda}_t}{\eta_{t+1}}, \tag{8.16c}$$

where $\hat{c}_t$ and η_{t+1} are residual variables that specify *normalized consumption* and *wealth ratio,* respectively. All in all, the resulting dynamic portfolio choice model has the following variables:

- $d = m_s$ state variables $\hat{\boldsymbol{x}}_t$: Stock fractions $s_{t,1}, \dots, s_{t,m_s}$
- $m_y = 2m_s + 1$ policy variables $\hat{\boldsymbol{y}}_t$: Normalized bonds $\hat{b}_t$, normalized buy amounts $\hat{\delta}_{t,1}^{+}, \dots, \hat{\delta}_{t,m_s}^{+}$ and normalized sell amounts $\hat{\delta}_{t,1}^{-}, \dots, \hat{\delta}_{t,m_s}^{-}$
- $m_{\boldsymbol{\omega}} = m_s$ stochastic variables $\boldsymbol{\omega}_t$: Stock return rates $\lambda_{t,1}, \dots, \lambda_{t,m_s}$

The state space and policy space constraints are given by

$$\boldsymbol{s}_t \geq \mathbf{0}, \qquad \Sigma(\boldsymbol{s}_t) \leq 1, \qquad \hat{b}_t \geq 0, \qquad \hat{\boldsymbol{\delta}}_t^{\pm} \geq \mathbf{0}, \qquad \hat{\boldsymbol{\delta}}_t^{-} \leq \boldsymbol{s}_t, \tag{8.17a}$$

$$\hat{c}_{\min} + \hat{b}_t + (1+\tau)\,\Sigma(\hat{\boldsymbol{\delta}}_t^{+}) - (1-\tau)\,\Sigma(\hat{\boldsymbol{\delta}}_t^{-}) \leq 1 - \Sigma(\boldsymbol{s}_t), \tag{8.17b}$$

where $\hat{c}_{\min} \in \mathbb{R}_{\geq 0}$ is some minimal consumption that must be maintained.

Bellman equation. Consequently, the Bellman equation (8.9) after the certainty-equivalent transformation has to be normalized as well. By setting $\hat{\tilde{J}}_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)}) := \tilde{J}_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)})/w_t$, we obtain

$$\hat{\tilde{J}}_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)}) = w_t^{-1}\tilde{J}_t^{\mathrm{s}}(\boldsymbol{x}_t^{(k)}) \tag{8.18a}$$

$$= \max_{\boldsymbol{y}_t}\left(\left(\left(w_t^{-1}c_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t)\right)^{1-\gamma} + \varrho\,\mathbb{E}_t\Big[\left(w_t^{-1}\tilde{J}_{t+1}^{\mathrm{s}}(\boldsymbol{\psi}_t(\boldsymbol{x}_t^{(k)}, \boldsymbol{y}_t, \boldsymbol{\omega}_t))\right)^{1-\gamma}\Big]\right)^{1/(1-\gamma)}\right) \tag{8.18b}$$

$$= \max_{\hat{\boldsymbol{y}}_t}\left(\left(\hat{c}_t(\boldsymbol{x}_t^{(k)}, \hat{\boldsymbol{y}}_t)^{1-\gamma} + \varrho\,\mathbb{E}_t\Big[\left(\eta_{t+1}\hat{\tilde{J}}_{t+1}^{s}(\hat{\boldsymbol{\psi}}_t(\boldsymbol{x}_t^{(k)}, \hat{\boldsymbol{y}}_t, \boldsymbol{\omega}_t))\right)^{1-\gamma}\Big]\right)^{1/(1-\gamma)}\right). \tag{8.18c}$$

This means that compared with (8.9), the value function in the expectation has to be multiplied by the wealth ratio η_{t+1} introduced above in (8.16). Since there is no inheritance, the optimal terminal solution is to sell all stocks and consume everything:

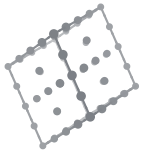
$$\hat{\tilde{J}}_t^{\mathrm{s}}(\boldsymbol{x}_T^{(k)}) = 1 - \tau\,\Sigma(\boldsymbol{s}_T^{(k)}), \quad \hat{b}_T^{\mathrm{opt}}(\boldsymbol{x}_T^{(k)}) = 0, \quad \hat{\boldsymbol{\delta}}_T^{+,\mathrm{opt}}(\boldsymbol{x}_T^{(k)}) = \mathbf{0}, \quad \hat{\boldsymbol{\delta}}_T^{-,\mathrm{opt}}(\boldsymbol{x}_T^{(k)}) = \boldsymbol{s}_T^{(k)}. \tag{8.19}$$

8.3.3 State Space Cropping

Sparse grids on non-rectangular domains. Unfortunately, the constraint $\Sigma(\boldsymbol{s}_t) \leq 1$ from Eq. (8.17) limits the feasible state space region to a proper subset (which is the unit simplex) of the unit hypercube $[\mathbf{0}, \mathbf{1}]$, which impedes the direct application of sparse grids. There are three possible remedies: transforming the unit hypercube to the feasible state space, applying extrapolation techniques as discussed in Sec. 8.2.5, or choosing a model-tailored approach to obtain function values outside the feasible state space.

Virtual selling of stocks. We choose the third remedy and *virtually sell,* if $\Sigma(\boldsymbol{s}_t) > 1$, as many stocks as needed to meet the constraint $\Sigma(\boldsymbol{s}_t) \leq 1$. We already might need to sell stocks even if $\Sigma(\boldsymbol{s}_t)$ is smaller but close to one in order to satisfy the minimum consumption requirement (8.17b). In detail, we replace $\boldsymbol{s}_t$ by $\hat{\beta}\boldsymbol{s}_t$ whenever $\hat{\beta} < 1$, where $\hat{\beta} \in \mathbb{R}_{>0}$ is a *cropping factor* that is determined by

$$\Big[1 - \tau\left(\Sigma(\boldsymbol{s}_t) - \Sigma(\hat{\beta}\boldsymbol{s}_t)\right)\Big] \cdot \left(1 - \Sigma(\hat{\beta}\boldsymbol{s}_t)\right) = \hat{c}_{\min}. \tag{8.20}$$



Here, $\big(\Sigma(\boldsymbol{s}_t) - \Sigma(\hat{\beta}\boldsymbol{s}_t)\big)$ is the amount of virtually sold stocks. Hence, the term in square brackets is the fraction of wealth that is still available after deducting the induced transaction costs. The product of this term with $\big(1-\Sigma(\hat{\beta}\boldsymbol{s}_t)\big)$ is the fraction of wealth that can be consumed after the virtual selling, which needs to be at least $\hat{c}_{\min}$. Solving Eq. (8.20) for $\hat{\beta}$ and choosing the positive solution, we finally obtain

$$\hat{\beta} := \frac{\tau\big(1+\Sigma(\boldsymbol{s}_t)\big) - 1 + \sqrt{\tau^2\big(1-\Sigma(\boldsymbol{s}_t)\big)^2 - 2\tau\big(2\hat{c}_{\min} - 1 + \Sigma(\boldsymbol{s}_t)\big) + 1}}{2\tau\,\Sigma(\boldsymbol{s}_t)}. \tag{8.21}$$

8.3.4 Euler Equation Errors

Motivation. Due to the curse of dimensionality, reasonably accurate full grid reference solutions of the transaction costs problem can only be computed if the number m_s of stocks is small. Mainly (but not only) in higher-dimensional settings, a different means of assessing the quality of sparse grid solutions is desirable. We use Euler equation errors to measure the deviation in the first-order optimality conditions.

Derivation. In the following, we fix the state $\hat{\boldsymbol{x}}_t \in [\mathbf{0},\mathbf{1}]$ for which we want to compute the Euler equation error. We abbreviate the value function interpolant $\hat{\tilde{J}}_t^s := \hat{\tilde{J}}_t^s(\hat{\boldsymbol{x}}_t)$, the state transition function $\hat{\boldsymbol{\psi}}_t := \hat{\boldsymbol{\psi}}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t, \boldsymbol{\omega}_t)$, the wealth ratio $\eta_{t+1} := \eta_{t+1}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t, \boldsymbol{\omega}_t)$, and the consumption $\hat{c}_t := \hat{c}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t)$. The Lagrangian of the optimization problem corresponding to the Bellman equation (8.18c) of the normalized transaction costs problem vanishes with respect to the problem's constraints (8.17) is given by

$$\begin{aligned}\mathscr{L}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t, \boldsymbol{\mu}) := \Big((\hat{c}_t)^{1-\gamma} + \varrho\,\mathbb{E}_t\Big[\big(\eta_{t+1}\,\hat{\tilde{J}}_{t+1}^s(\hat{\boldsymbol{\psi}}_t)\big)^{1-\gamma}\Big]\Big)^{1/(1-\gamma)} \\ - \mu_1 \hat{b}_t - \boldsymbol{\mu}_2^{\mathrm{T}}\hat{\boldsymbol{\delta}}_t^{+} - \boldsymbol{\mu}_3^{\mathrm{T}}\hat{\boldsymbol{\delta}}_t^{-} + \boldsymbol{\mu}_4^{\mathrm{T}}\,(\hat{\boldsymbol{\delta}}_t^{-} - \boldsymbol{s}_t) + \mu_5\,(\hat{c}_{\min} - \hat{c}_t)\end{aligned} \tag{8.22}$$

with $\boldsymbol{\mu} := (\mu_1, \boldsymbol{\mu}_2, \boldsymbol{\mu}_3, \boldsymbol{\mu}_4, \mu_5)$, $\mu_1, \mu_5 \in \mathbb{R}$, and $\boldsymbol{\mu}_2, \boldsymbol{\mu}_3, \boldsymbol{\mu}_4 \in \mathbb{R}^{m_s}$. According to the first-order conditions *(Karush–Kuhn–Tucker (KKT) conditions),* the partial derivative $\frac{\partial}{\partial \hat{b}_t}\mathscr{L}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t, \boldsymbol{\mu})$ with respect to $\hat{b}_t$ vanishes in the exact optimum $\hat{\boldsymbol{y}}_t = \hat{\boldsymbol{y}}_t^{\mathrm{opt}} := \hat{\boldsymbol{y}}_t^{\mathrm{opt}}(\hat{\boldsymbol{x}}_t)$, i.e.,

$$\frac{\partial}{\partial \hat{b}_t}\Big((\hat{c}_t^{\mathrm{opt}})^{1-\gamma} + \varrho\,\mathbb{E}_t\Big[\big(\eta_{t+1}^{\mathrm{opt}}\,\hat{\tilde{J}}_{t+1}^s(\hat{\boldsymbol{\psi}}_t^{\mathrm{opt}})\big)^{1-\gamma}\Big]\Big)^{1/(1-\gamma)} - \mu_1 - \mu_5\frac{\partial}{\partial \hat{b}_t}\hat{c}_t^{\mathrm{opt}} = 0, \tag{8.23}$$

where $\hat{\boldsymbol{\psi}}_t^{\mathrm{opt}} := \hat{\boldsymbol{\psi}}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\mathrm{opt}}, \boldsymbol{\omega}_t)$, $\eta_{t+1}^{\mathrm{opt}} := \eta_{t+1}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\mathrm{opt}}, \boldsymbol{\omega}_t)$, and $\hat{c}_t^{\mathrm{opt}} := \hat{c}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\mathrm{opt}})$. We now neglect binding constraints, i.e., we assume that $\mu_1 = \mu_5 = 0$, otherwise we cannot compute the error. After calculating the derivatives, Eq. (8.23) becomes

$$\varrho\, r_t \cdot \mathbb{E}_t\Big[\big(\hat{\tilde{J}}_t^s - (\boldsymbol{\nabla}_{\hat{\boldsymbol{x}}_t}\hat{\tilde{J}}_t^s)^{\mathrm{T}}\hat{\boldsymbol{\psi}}_t^{\mathrm{opt}}\big)\cdot\big(\eta_{t+1}^{\mathrm{opt}}\,\hat{\tilde{J}}_t^s\big)^{-\gamma}\Big] = (\hat{c}_t^{\mathrm{opt}})^{-\gamma}. \tag{8.24}$$

This equation can be used as an error measure by substituting $\hat{\boldsymbol{y}}_t^{\text{opt}}$ for the interpolated optimum $\hat{\boldsymbol{y}}_t^{\text{opt,s}} = \hat{\boldsymbol{y}}_t^{\text{opt,s}}(\hat{\boldsymbol{x}}_t)$. By multiplying the resulting equation by $(\hat{c}_t^{\text{opt,s}})^\gamma := (\hat{c}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\text{opt,s}}))^\gamma$, we obtain the *unit-free Euler equation errors* $\varepsilon_t^{\text{Eu}}(\hat{\boldsymbol{x}}_t)$ with respect to $\hat{b}_t$:

$$\varepsilon_t^{\text{Eu}}(\hat{\boldsymbol{x}}_t) := \left| 1 - \left(\varrho r_t (\hat{c}_t^{\text{opt,s}})^\gamma \cdot \mathbb{E}_t \left[\left(\hat{\tilde{J}}_t^{\text{s}} - (\boldsymbol{\nabla}_{\hat{\boldsymbol{x}}_t} \hat{\tilde{J}}_t^{\text{s}})^{\mathrm{T}} \hat{\boldsymbol{\psi}}_t^{\text{opt,s}} \right) \cdot \left(\eta_{t+1}^{\text{opt,s}} \hat{\tilde{J}}_t^{\text{s}} \right)^{-\gamma} \right] \right)^{-1/\gamma} \right| \tag{8.25}$$

with $\hat{\boldsymbol{\psi}}_t^{\text{opt,s}} := \hat{\boldsymbol{\psi}}_t(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\text{opt,s}}, \boldsymbol{\omega}_t)$ and $\eta_{t+1}^{\text{opt,s}} := \eta_{t+1}(\hat{\boldsymbol{x}}_t, \hat{\boldsymbol{y}}_t^{\text{opt,s}}, \boldsymbol{\omega}_t)$.

Weighted Euler equation errors. However, the state space cropping as introduced above distorts Euler equation errors: The error $\varepsilon_t^{\text{Eu}}(\hat{\boldsymbol{x}}_t)$ does not vanish even for the exact solution and even inside the feasible state space. This is because the cropping already occurs for large stock holdings $\Sigma(\hat{\boldsymbol{x}}_t)$ that are less than one, as stocks have to be sold to maintain minimum consumption $\hat{c}_{\min}$. Numerical experiments show that due to this issue, the error attains large values in the region near the hyperplane $\Sigma(\hat{\boldsymbol{x}}_t) = 1$. Economically, this region is not significant as such large stock fractions are highly unusual, which is confirmed by Monte Carlo simulations. We therefore use the *weighted Euler equation error*

$$\varepsilon_t^{\text{w,Eu}}(\hat{\boldsymbol{x}}_t) := \left(1 - \Sigma(\hat{\boldsymbol{x}}_t)\right) \cdot \varepsilon_t^{\text{Eu}}(\hat{\boldsymbol{x}}_t) \tag{8.26}$$

instead of $\varepsilon_t^{\text{Eu}}$, although other strategies exist such as restricting the state domain where the error is computed or weighting the error with the probability that a given state occurs in Monte Carlo simulations.

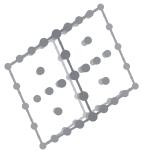
8.4 Implementation and Numerical Results

8.4.1 Implementation

IN THIS SECTION

Parameter values. We used a risk aversion factor of $\gamma := 3.5$, a patience factor of $\varrho := 0.97$, a transaction cost rate of $\tau := 1\,\%$, and a minimum consumption of $\hat{c}_{\min} := 0.001$. The bond and stock return rates r_t and $\boldsymbol{\lambda}_t$ were taken from [Cai10]; the log-normally distributed stock return rates were generalized from the three-stock case to five stocks via $\ln \boldsymbol{\lambda}_t \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, where

$$\boldsymbol{\mu} := \begin{pmatrix} 0.0572 \\ 0.0638 \\ 0.07 \\ 0.0764 \\ 0.0828 \end{pmatrix}, \quad \boldsymbol{\Sigma} := 10^{-2} \begin{pmatrix} 2.56 & 0.576 & 0.288 & 0.176 & 0.096 \\ 0.576 & 3.24 & 0.90432 & 1.0692 & 1.296 \\ 0.288 & 0.90432 & 4 & 1.32 & 1.68 \\ 0.176 & 1.0692 & 1.32 & 4.84 & 2.112 \\ 0.096 & 1.296 & 1.68 & 2.112 & 5.76 \end{pmatrix}. \tag{8.27}$$



The models were solved for $T = 6$ time steps; this number suffices to show all relevant numerical effects and results, while keeping the computational effort at a reasonable level. As initial grids, we employed regular sparse grids $\Omega_{n,d}^{s(b)}$ with $b = 1$ to decrease the number of grid points (see Sec. 2.4.1).

Software. The dynamic portfolio choice models were solved using a self-written MATLAB framework. The object-oriented framework was designed in such a way that not only transaction costs problems, but many other types of dynamic portfolio choice models can be handled. For instance, the base class `LifecycleProblem` provides an interface with abstract functions such as `computeTerminalValueFunction` and `computeStateTransition`. The actual functionality implemented in the base class strongly resembles the algorithms presented in Sec. 8.2. This is not only desirable from a modeling perspective, but also facilitates future usage by other researchers. For creating (i.e., hierarchizing) and evaluating sparse grid interpolants, the sparse grid toolbox SG^{++} was used [Pfl10].[6] The emerging optimization problems were solved using sequential quadratic programming methods supplied by the NAG Toolbox for MATLAB.[7] To avoid being stuck in local minima, we repeated the optimization process for a varying number of initial multi-start points (in the range of a few dozens). All computation times were measured on a shared-memory computer with 144 threads on 4x Intel Xeon E7-8880v3 (72 cores, 144 threads).

8.4.2 Error Sources and Error Measure

Error sources. In this application, there are the following error sources:

E1. Interpolation of the value function (i.e., $\hat{\tilde{J}}^{s}_{t+1} \neq \hat{\tilde{J}}_{t+1}$)

E2. Interpolation of the policy functions (i.e., $\hat{\boldsymbol{y}}_t^{\text{opt,s}} \neq \hat{\boldsymbol{y}}_t^{\text{opt}}$)

E3. Extrapolation (i.e., $\hat{\tilde{J}}^{s}_{t+1}(\boldsymbol{x}_{t+1}) \neq \hat{\tilde{J}}_{t+1}(\boldsymbol{x}_{t+1})$)

E4. State space cropping (i.e., Euler errors do not vanish for exact solution)

E5. Optimization (i.e., the minimum found by the optimizer is inaccurate or not global)

E6. Quadrature ($\mathbb{E}_t[\cdots] \neq \sum_{j=1}^{m_\zeta} \zeta_t^{(j)} \cdot [\cdots](\boldsymbol{\omega}_t^{(j)})$)

E7. Floating-point rounding errors (i.e., arithmetical operations are inaccurate)

Due to the dynamic programming scheme, the combination of all errors accumulates over t. For instance, if the optimization does not find the global optimum exactly or it only

[6] `http://sgpp.sparsegrids.org/`
[7] `https://www.nag.com/`

finds a local one for $t+1$, the error propagates from the interpolant $\hat{\tilde{J}}^s_{t+1}$ on the right-hand side of the Bellman equation (8.18c) to $\hat{\tilde{J}}^s_t$ on the left-hand side, and so on. If the system does not damp these errors, the error steadily becomes larger backwards in time t.

Error measure. We use the weighted Euler equation error $\varepsilon_t^{\mathrm{w,Eu}}(\hat{\boldsymbol{x}}_t)$ to assess the quality of the resulting policies (L^2 norm or pointwise). As the errors generally grow backwards in time, it suffices to consider $t = 0$. However, since Euler equation errors can only be evaluated at points in the simplex $\Omega_{\mathrm{simplex}} := \{\hat{\boldsymbol{x}}_t \in [\mathbf{0}, \mathbf{1}] \mid \Sigma(\hat{\boldsymbol{x}}_t) \leq 1\}$, the L^2 norm would quickly converge to zero with growing dimensionality, even if the mean error stayed constant. Therefore, we normalize the L^2 norm:

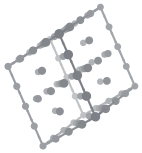
$$\varepsilon_t^{\mathrm{w,Eu},L^2} := \sqrt{d!} \cdot \|\varepsilon_t^{\mathrm{w,Eu}}\|_{L^2} = \sqrt{\frac{1}{\mathrm{vol}(\Omega_{\mathrm{simplex}})} \int_{\Omega_{\mathrm{simplex}}} \varepsilon_t^{\mathrm{w,Eu}}(\hat{\boldsymbol{x}}_t)^2 \,\mathrm{d}\hat{\boldsymbol{x}}_t}, \tag{8.28}$$

where the expression under the root sign is approximated via Monte Carlo quadrature as the mean of samples of $\varepsilon_t^{\mathrm{w,Eu}}(\hat{\boldsymbol{x}}_t)^2$.

8.4.3 Numerical Results

Full grid solution. We show in Fig. 8.3 a full grid solution for the case of $d = 2$ stocks, i.e., $\{\boldsymbol{x}_t^{(k)} \mid k = 1, \ldots, N_t\} = \Omega_{n,d}$ for some fixed level $n \in \mathbb{N}$ (here, $n = 7$ and $N_t = (2^7+1)^2 = 16\,641$) and for all $t = 0, \ldots, T$. Obviously, this is only computationally feasible for low dimensionalities d due to the curse of dimensionality. The two-dimensional solution of level $n = 7$ took over nine hours to compute. The solution of the next level is estimated to already take one week. Hence, full grid solutions can only be computed up to $d = 3$ due to excessive computation time for $d \geq 4$. This underlines the need for sophisticated discretization techniques such as sparse grids.

Convergence of the weighted Euler equation error. Figure 8.4 shows the convergence of the L^2 norm $\varepsilon_0^{\mathrm{w,Eu},L^2}$ weighted Euler equation error for $t = 0$ for regular sparse grids and spatially adaptive sparse grids for the cases of $d = 1, \ldots, 4$ stocks. For this and the following plots, the value function grid is left unchanged (usually a slightly refined regular sparse grid), while the mean number N_t of policy grid points increases with decreasing refinement threshold κ_t. This is because the value function grid does not seem to have a great influence on the convergence of the Euler equation errors. Compared to regular grids, the spatial adaptivity decreases the error by two orders of magnitude in one dimension. The gain is smaller for higher dimensionalities d, but spatially adaptive grids still outperform regular grids. For $d = 2$, we observe that the error saturates at $N_t \approx 4000$



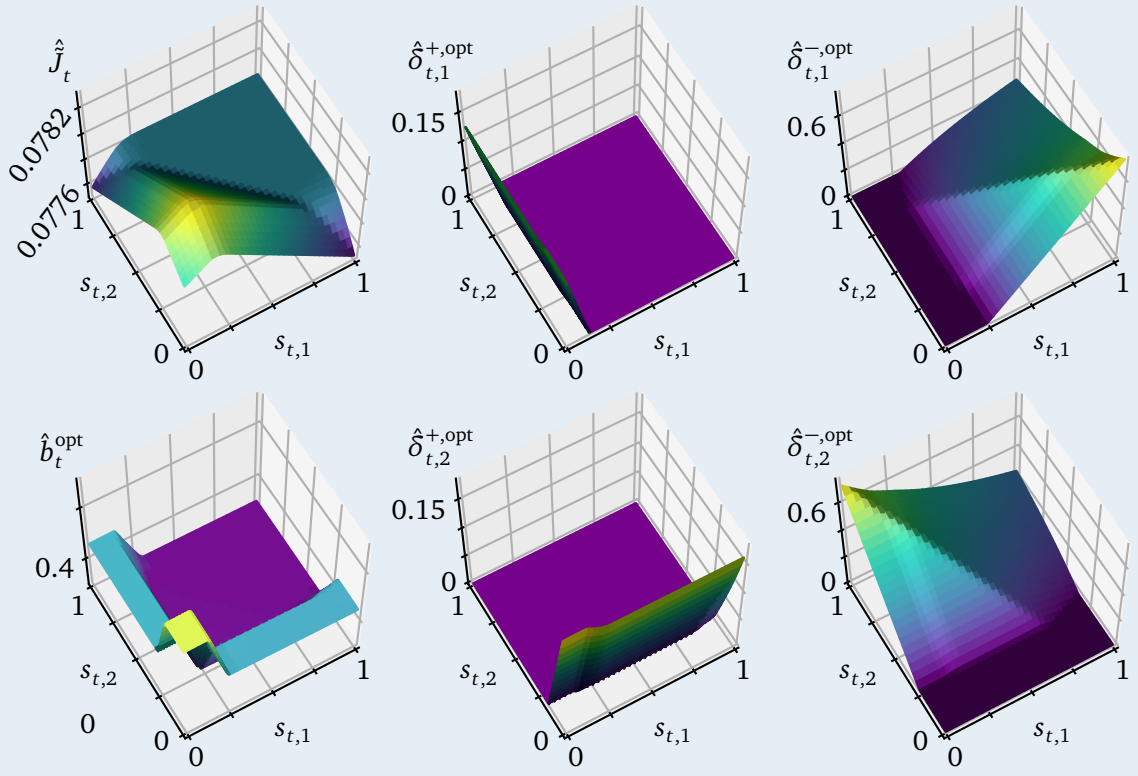


FIGURE 8.3 Full grid solution for the transaction costs problem with $d = 2$ stocks. Shown are the value function $\hat{\tilde{J}}_t$ *(top left)* and the optimal policy $\hat{\boldsymbol{y}}_t^{\text{opt}}$ for $t = 0$.

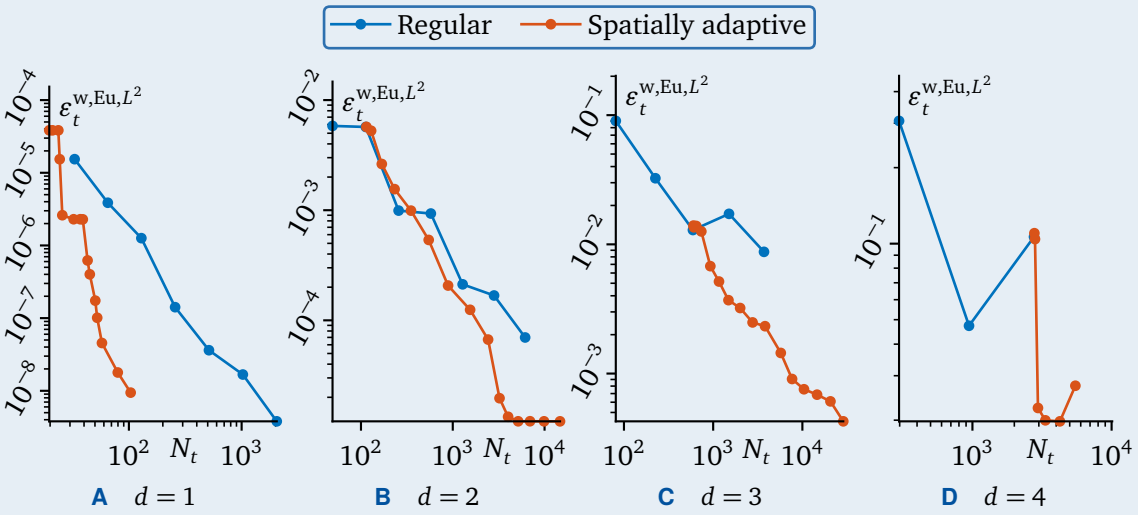


FIGURE 8.4 Convergence of the L^2 norm $\varepsilon_t^{\text{w,Eu},L^2}$ of the weighted Euler equation error for $t = 0$ for regular sparse grids *(blue)* and spatially adaptive sparse grids *(red)*. The number N_t is the mean number $\frac{1}{m_y}\sum_{j=1}^{m_y} N_{t,j}$ of grid points over all policy grids for $t = 0$, where $N_{t,j}$ is the number of grid points of the j-th policy entry.

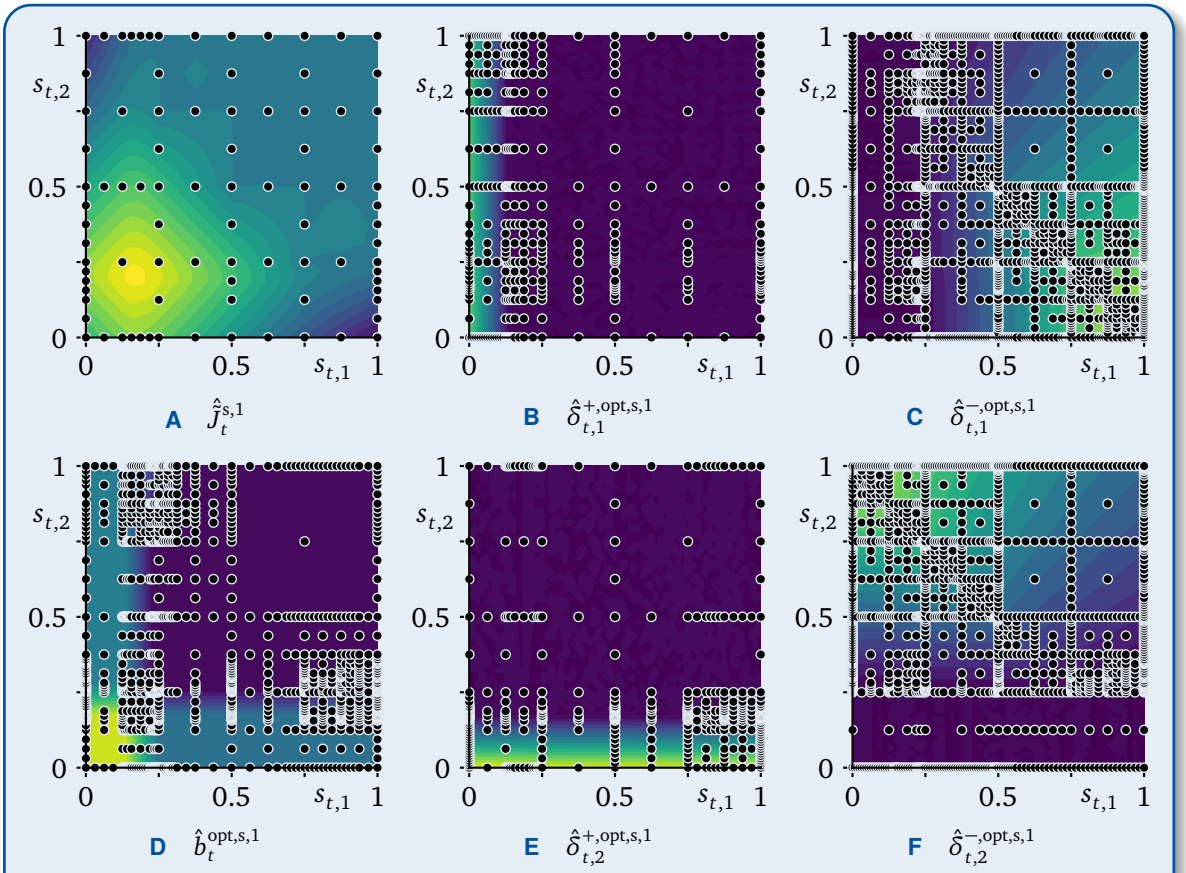
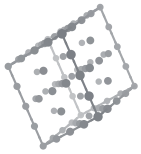


FIGURE 8.5 Spatially adaptive sparse grid solution for the transaction costs problem with $d = 2$ stocks. Shown are the value function $\hat{\hat{J}}_t^s$ *(top left)* and the optimal policy $\hat{\boldsymbol{y}}_t^{\mathrm{opt},s}$ for the initial time step $t = 0$, together with the corresponding grid points *(dots)*. The color coding is the same as in Fig. 8.3.

points just above 10^{-5}. This is most likely due to the parts E3 to E7 of the error that are not influenced by sparse grid interpolation. In addition, convergence significantly decelerates starting with $d = 4$. For $d = 4$, spatially adaptive sparse grids are able to achieve a weighted Euler equation error of $\varepsilon_t^{\mathrm{w,Eu},L^2} \approx 2.0 \cdot 10^{-2}$ for $t = 0$ (with a mean number $N_0 = 4252$ of policy grid points). For $d = 5$, we are still able to achieve a small error of $\varepsilon_t^{\mathrm{w,Eu},L^2} \approx 1.9 \cdot 10^{-2}$ for $t = 0$ with spatially adaptive sparse grids with a mean number $N_0 = 12\,572$ of policy grid points. While we cannot detect any convergence for this dimensionality yet, this is still a major result as such high-dimensional models could not be solved up to now with conventional methods.

Optimal policies in 2D and 5D. Figures 8.5 and 8.6 each display the value function and the optimal policies corresponding to sparse grid solutions for $d = 2$ stocks with $N_0 = 879$ policy grid points or $d = 5$ stocks with $N_0 = 12\,572$ policy grid points. Obviously, most



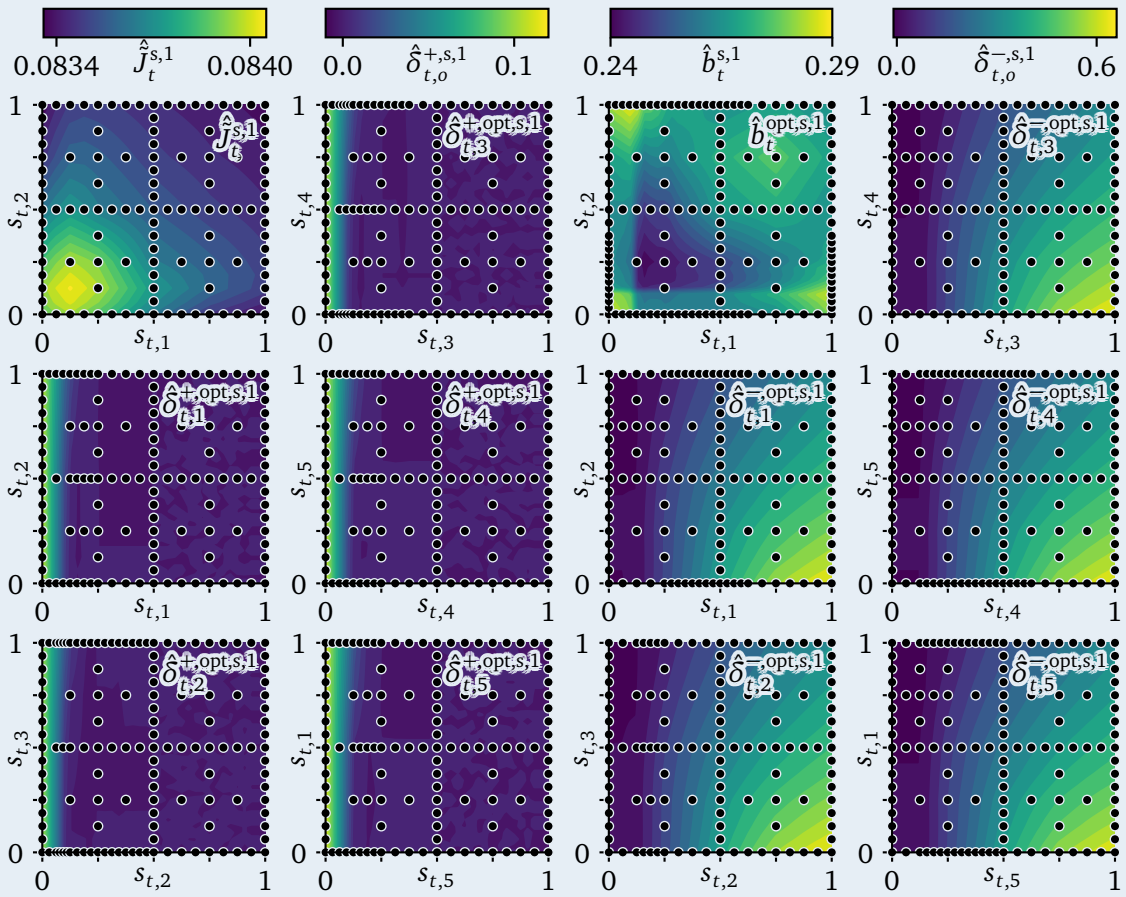


FIGURE 8.6 Spatially adaptive sparse grid solution for the transaction costs problem with $d = 5$ stocks. Shown are slice plots of the value function $\hat{\tilde{J}}_t^s$ *(top left)* and the optimal policy $\hat{\boldsymbol{y}}_t^{\text{opt,s}}$ for the initial time step $t = 0$, where for each function, a pair (o_1, o_2) of dimensions to be plotted was chosen, and the stock fractions $s_{t,o}$ of the other dimensions o are set to 0.1. In addition, the corresponding grid points *(dots)* are shown as the projection onto the s_{t,o_1}-s_{t,o_2} plane.

grid points are placed along the various kinks in the policies. Interestingly, experiments show that the surplus-based refinement criterion does not place more grid points along the perfectly diagonal kink caused by the cropping of the state space (i.e., along $\Sigma(\boldsymbol{s}_t) = 1$). It is possible to circumvent this issue by either transforming the domain (e.g., rotations as in [Boh18]) or directly incorporating the distance to the diagonal into the refinement criterion for the value function. However, we refrain from doing so here as this does not seem to drastically improve results. Again, this might be due to the domination of the overall error by the parts E3 to E7 that are not related to interpolation.

Pointwise error. Pointwise plots of the weighted Euler equation error as in Fig. 8.7 for two stocks reveal that there are two types of regions where the error is large: The first type

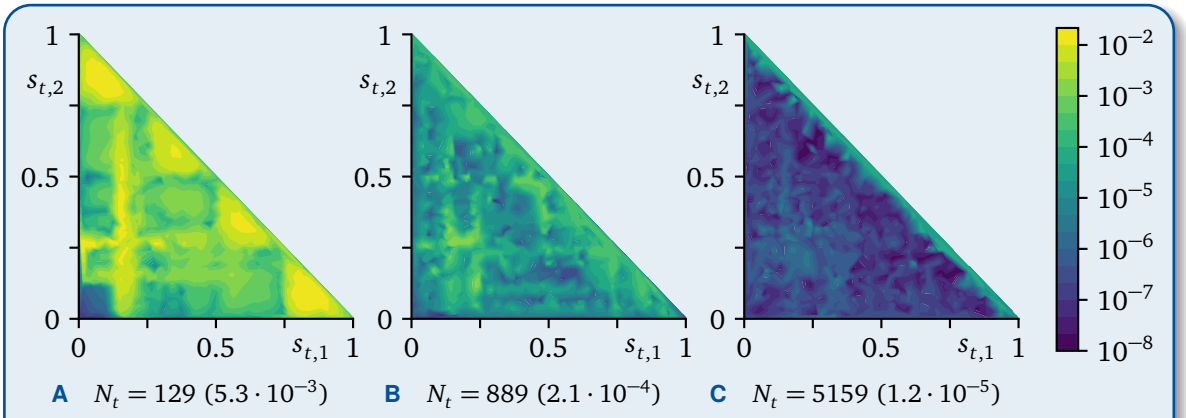


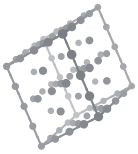
FIGURE 8.7 Pointwise weighted Euler equation error $\varepsilon_t^{\mathrm{w,Eu}}(\boldsymbol{s}_t)$ $(t = 0)$ for the two-dimensional transaction costs problem and different spatially adaptive sparse grids. The L^2 error $\varepsilon_t^{\mathrm{w,Eu},L^2}$ is given in parentheses.

of region is the neighborhood of the aforementioned diagonal boundary $\Sigma(\boldsymbol{s}_t) = 1$ of the uncropped region, where the cropping distorts the error despite the weights. The second type of region are kinks of the optimal policy functions, which is most visible for coarse grids (e.g., Fig. 8.7A). When increasing the number of grid points (e.g., Figures 8.7A and 8.7B), the error decreases quickly in the whole domain.

Monte Carlo simulation. As explained in Sec. 8.2.8, we perform a multi-agent Monte Carlo simulation and plot the resulting mean state and policy in Fig. 8.8 for $d = 3$, 4, and 5 stocks. In addition, this figure contains the evolution of the weighted Euler equation error $\varepsilon_t^{\mathrm{w,Eu},L^2}$ over time. We perform a two-part assessment of the simulated results. First, consumption should ideally be constant over time from a finance perspective. We measure this by calculating the coefficients of variation (ratio of the standard deviation of the c_t values to their mean), which is 2.76 %, 2.68 %, and 2.58 % for $d = 3$, 4, and 5, respectively. This indicates that the variation of the consumption over time is indeed small. Second, we consider the so-called *Sharpe ratios* [Sharp66]. The ratios are stock fractions $\boldsymbol{s}$ that are determined such that the excess stock return (compared to risk-free investment) per unit of risk is maximized:[8]

$$\operatorname*{arg\,max}_{\boldsymbol{s}\in[0,1]^d} \frac{\boldsymbol{\mu}_{1:d}^{\mathrm{T}}\boldsymbol{s} - r}{\sqrt{\boldsymbol{s}^{\mathrm{T}}\boldsymbol{\Sigma}_{1:d,1:d}\boldsymbol{s}}}, \tag{8.29}$$

[8]The Sharpe ratios per se are derived for non-skewed stock return rate distributions. Our stock return rates are log-normally distributed and thus skewed, but the deviation should be small after six time steps. However, there are variants that take skewed distributions into account [Mül15].



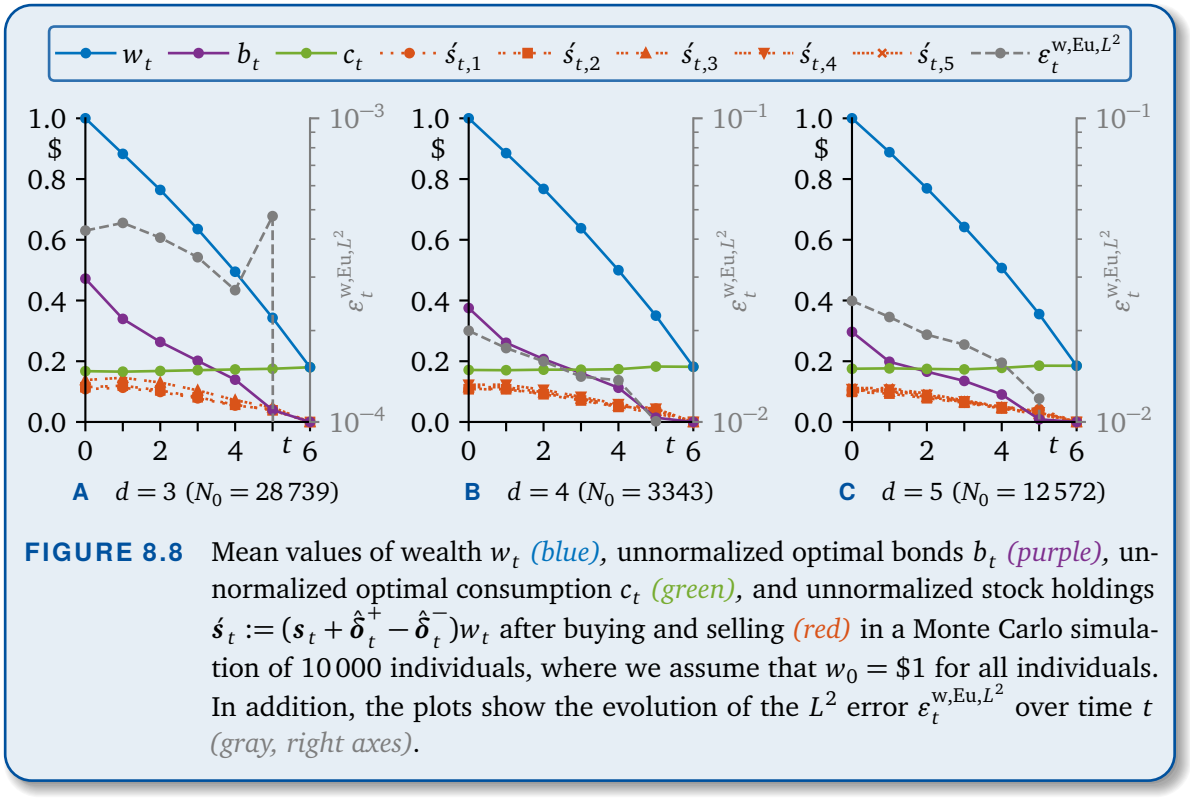


FIGURE 8.8 Mean values of wealth w_t *(blue)*, unnormalized optimal bonds b_t *(purple)*, unnormalized optimal consumption c_t *(green)*, and unnormalized stock holdings $\acute{\boldsymbol{s}}_t := (\boldsymbol{s}_t + \hat{\hat{\boldsymbol{\delta}}}_t^+ - \hat{\hat{\boldsymbol{\delta}}}_t^-)w_t$ after buying and selling *(red)* in a Monte Carlo simulation of 10 000 individuals, where we assume that $w_0 = \$1$ for all individuals. In addition, the plots show the evolution of the L^2 error $\varepsilon_t^{\mathrm{w,Eu},L^2}$ over time t *(gray, right axes)*.

where $\boldsymbol{\mu}_{1:d}$ and $\boldsymbol{\Sigma}_{1:d,1:d}$ are the first d entries of $\boldsymbol{\mu}$ and the principal minor of order d of $\boldsymbol{\Sigma}$ as given in Eq. (8.27). We compare these theoretical Sharpe ratios (left) with the simulated stock fractions $\acute{s}_{t,o}/\Sigma(\acute{\boldsymbol{s}}_t)$ for $t = 0$ (right):

(8.30a) $d = 3$: $(0.314, 0.302, 0.384)$, $(0.300, 0.317, 0.383)$,

(8.30b) $d = 4$: $(0.275, 0.185, 0.250, 0.289)$, $(0.239, 0.238, 0.253, 0.270)$,

(8.30c) $d = 5$: $(0.275, 0.122, 0.176, 0.203, 0.223)$, $(0.199, 0.188, 0.197, 0.205, 0.212)$.

The simulated stock fractions match the predicted Sharpe ratios well for $d = 3$, while the deviation for $d \geq 4$ is larger. However, as the simulated stock fractions do not change much over time, we may suspect that the skewness of the distribution of the stock return rates limits the applicability of the Sharpe ratios to these cases.

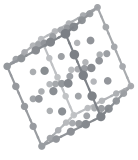
Complexity and computation time. A complexity analysis reveals that the difficulty of solving transaction cost problems quickly grows with the dimensionality d: As shown in Fig. 8.2, the number of necessary arithmetic operations grows like

$$\Theta(T \cdot N_t \cdot \#\text{optimizer iterations} \cdot \underbrace{m_\zeta \cdot \overbrace{m_y \cdot N_{t+1} \cdot m_x \cdot p}^{\text{one evaluation of interpolant}}}_{\text{one evaluation of objective gradient}}), \tag{8.31}$$

where $m_x, m_y \in \Theta(d)$ and $m_\zeta, N_t, N_{t+1} \in \Theta(2^n n^{d-1})$ if regular sparse grids of level n without boundary points are used for state and stochastic grids (due to $m_\omega = d$). In addition, the number of optimizer iterations is likely superlinear in d, as this depends on the dimensionality m_y of the search space as well as on the number of multi-start points (which also grows with m_y). This means that the complexity is at least cubic in d, quadratic in the mean number N of employed state grid points, and linear in the number m_ζ of quadrature points. Figure 8.9 confirms these observations with experimental data. For fixed d, the total time required by the optimization process grows quadratically with the number N of grid points. The time for one solution of the Bellman equation, the time for one optimizer iteration, and the time for one evaluation of the interpolant are all linear in N, as the number of optimizer iterations is constant for fixed d. If d increases, then the number of interpolant evaluations per optimizer iteration (i.e., the number of quadrature points) increases as well. Surprisingly, the number of optimizer iterations per grid point and the time per evaluation are not monotonously increasing. The latter observation might be due to vectorization effects.

Comparison to piecewise linear functions. Hierarchical B-splines introduce two major benefits to the solution of dynamic portfolio choice models. The first benefit are the smooth objective functions: When repeating the computations with piecewise linear functions (i.e., $p = 1$), one obtains almost the same weighted Euler equation errors as in the cubic case (except for the case of $d = 1$, where the error is one order of magnitude greater than in the cubic case). However, as we see in Fig. 8.9, the total computation time is several times larger (e.g., more than five times for $d = 3$) for piecewise linear functions, although evaluations are cheaper than for B-splines. The main reason is that the number of required optimizer iterations is for $p = 1$ almost seven times as high as in the cubic case, since the optimizer has to deal with kinks in the objective function. Experiments show that beginning with $d = 4$, the total optimization time required to solve the transaction costs problem is one whole order of magnitude shorter for cubic B-splines than for piecewise linear functions.

Comparing exact gradients to finite differences. The second benefit is the availability of exact gradients: Figure 8.9 also contains computation times of the solution process if we artificially do not use exact gradients of the objective functions, but rather approximate them with finite differences. For each evaluation of the objective gradient, at least m_y additional evaluations of the objective function have to performed to compute the finite differences ($2m_y$ if central differences are used). Consequently, while the resulting weighted Euler equation errors are similar, the total optimization time roughly increases by a factor of up to five if we do not use exact gradients.



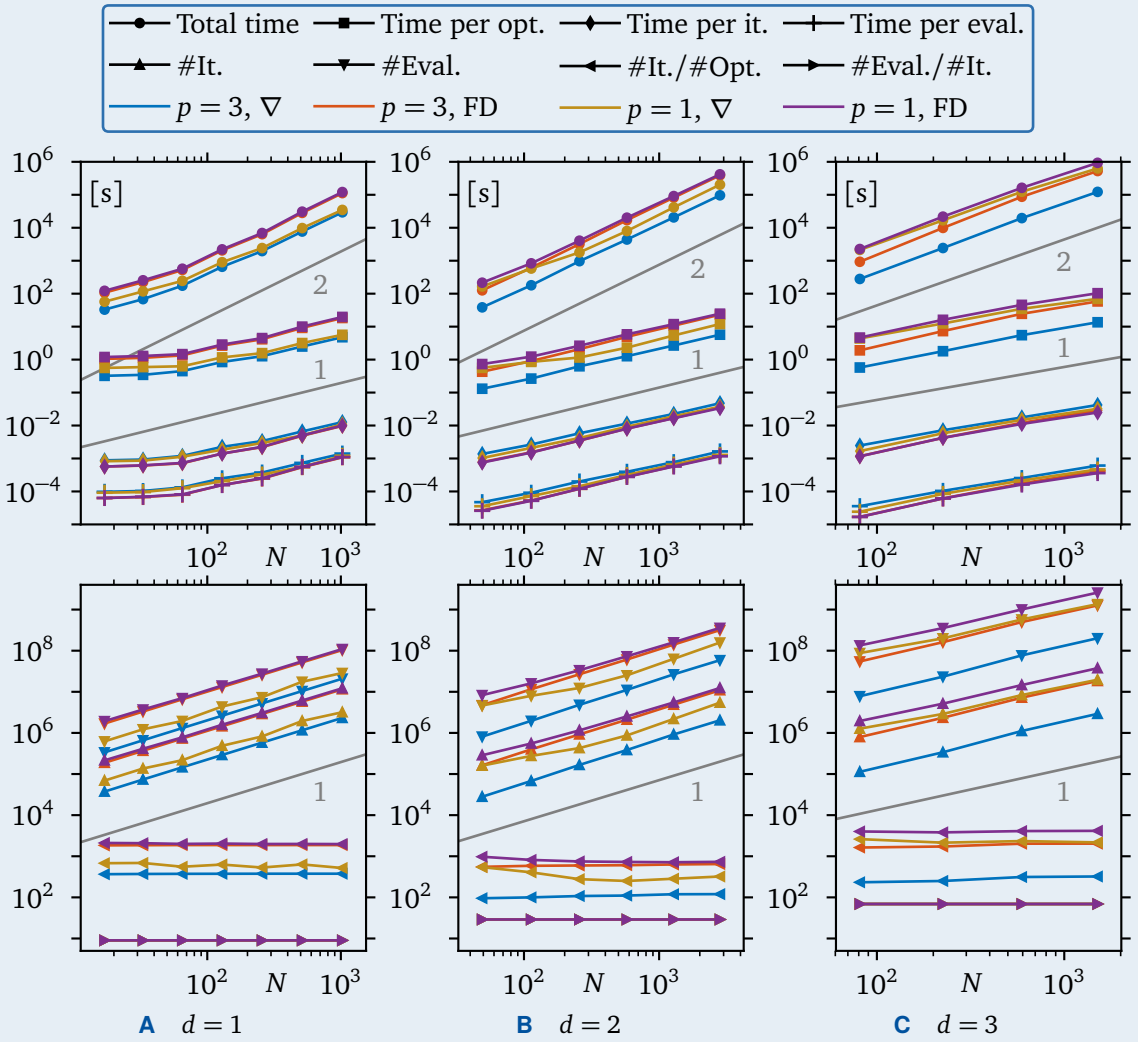
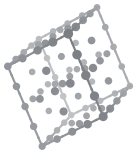


FIGURE 8.9 Computation times *(top)* and numbers of iterations and evaluations of the interpolant *(bottom)* for the transaction costs problem on static regular sparse grids (i.e., without refinement). "Total time" is the serial time required to solve all emerging optimization problems. "Time per opt." is this time divided by the number #Opt. $= TN$ of optimization problems. "Time per it." is the time divided by the number #It. of optimization iterations, each of which is assumed to correspond to exactly one combined evaluation of objective function and gradient (the latter only if gradients are used). "Time per eval." is the time divided by the number #Eval. of evaluations of the sparse grid interpolant and its gradient. The colors correspond to B-spline degrees $p = 3$ or $p = 1$ and to using gradients ("∇") or finite differences ("FD").

Conclusion

Finally, we conclude the thesis by summarizing its results and by giving an outlook on possible future work. In particular, we highlight key contributions of the thesis to research, give recommendations for future applications of the presented method, and state possible downsides and limitations.

Summary of the thesis. The contribution of this thesis consisted of two major parts. In the first part, hierarchical B-splines on sparse grids were comprehensively presented and embedded in a sparse grid framework with general tensor product basis functions. The advantage of this approach was that the framework could be reused for different hierarchical bases (as for the various spline bases derived in this thesis) and that it clarified which properties only held for the classical piecewise linear bases and not for other tensor product bases. We saw that standard hierarchical B-splines suffer from approximation issues near the boundary, and we resolved these issues by incorporating not-a-knot boundary conditions into the hierarchical B-spline basis. In the further course of the thesis, the focus was put on the algorithmic implications of the novel bases, taking the hierarchization problem as an example. We looked at requirements that had to be satisfied by grids and bases to enable efficient hierarchization algorithms such as breadth-first search and unidirectional principle, for which we gave clear formulations and formal correctness proofs. As a result, a whole "zoo" of hierarchical (B-)spline functions has been derived in this thesis. The main types were standard hierarchical B-splines, modified hierarchical B-splines, hierarchical not-a-knot B-splines, hierarchical fundamental splines, and hierarchical weakly fundamental splines (where the first two are not novel). Modified, not-a-knot, and (weakly) fundamental splines could be combined almost arbitrarily to tailor the ansatz functions to suit one's specific needs.



Category	Topology opt.	Biomechanics	Finance
Interpolated quantities	Elasticity tensors	Muscle forces	Value functions
SG dimensionality	5	2	5
#Optimization variables	40 000	2	11
Time per evaluation	30 s	30 min	—
#Eval. per opt. iteration	8000	4	150
Objective function type	Non-linear	Linear/non-linear	Non-linear
Constraint function type	Non-linear	Non-linear/—	Linear
Optimization method	SQP	Augm. Lagrangian	SQP

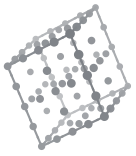
TABLE 9.1 Summary of characteristics of the applications presented in this thesis. The given values are rough example values that represent possible application test cases.

The second, more practical part of the thesis was dedicated to transferring the newly gained theoretical knowledge to academic and real-life application test cases. We verified that only with the new hierarchical not-a-knot conditions, one is able to obtain the best possible order of convergence $\mathcal{O}(h_h^{p+1})$ for interpolation (B-spline degree p, fixed dimensionality d). Using the Novak–Ritter criterion, which was specifically designed for optimization, we were able to achieve optimization gaps that were for some test functions up to six orders of magnitude smaller for cubic B-splines than for standard piecewise linear functions. We transferred the Novak–Ritter criterion to uncertainty quantification and obtained similarly strong results for the propagation of fuzzy uncertainties with the fuzzy extension principle. Furthermore, we successfully showed the suitability of hierarchical B-splines for three real-world applications, which are summarized in Tab. 9.1. First, by interpolating Cholesky factors of elasticity tensors, we accomplished to efficiently solve topology optimization problems in three spatial dimensions with complex microcell structures. Second, in the biomechanical application, we dramatically reduced the computational time to solve test scenarios by up to 99 % by using sparse grid surrogates with B-splines instead of the exact continuum-mechanical model. Third, we were able to solve dynamic portfolio choice problems with five state variables and eleven policy variables with unprecedented precision, as one could only speculate how the solution looked like with state-of-the-art methods. In all of these applications, the advantages of B-splines were made clear by comparing the results to the classical piecewise linear basis. The implementation of hierarchical B-splines of sparse grids is publicly available as part of the sparse grid toolbox SG^{++} under a free and open-source license.[1]

[1] `http://sgpp.sparsegrids.org/`

Recommendations (advantages and disadvantages). Despite its broad applicability, the presented method of B-splines on sparse grids is of course not suited for all possible scenarios. One must be able to sample the objective function at arbitrary locations in some hyper-rectangle in order to use sparse grids; a prescribed point cloud of scattered data does not suffice. Moreover, the problem should not have more than ten dimensions, since convergence notably slows down as the dimensionality grows, although spatially adaptive approaches might still be feasible for higher dimensionalities [Pfl10]. In addition, the objective function should be "as smooth as possible" in order to benefit from higher-order B-splines. This means "continuous" at the very least, but twice continuous differentiability is more desirable. The general rule is that the employed basis functions should be at least as smooth as the objective function in order to obtain optimal convergence results. The concrete choice of basis (general type and degree) depends on the application: Not-a-knot B-splines are well-suited for objective functions with dominating near-polynomial parts. Fundamental splines may be used to accelerate the process of hierarchization by enabling breadth-first search in quadratic time. With weakly fundamental splines, this can be further reduced to linear time using the unidirectional principle. However, the additional grid points that have to be inserted have to be taken into account as well. The rule of thumb is that the more spatially adaptive a sparse grid is (i.e., only few high-level grid points), the more points have to be inserted. In general, it does not hurt to try the different available B-spline types and degrees, since most function values can simply be reused once the objective function has been sampled.

Outlook and future work. Finally, we briefly give suggestions for possible future work. A major topic of interest is that of refinement criteria and adaptivity. Besides the Novak–Ritter criterion, there are other refinement criteria that are tailored to optimization such as simultaneous optimistic optimization [Wan14]. In addition, nested methods for hierarchical optimization could use multiple interpolants with different resolutions on different grids [Delb14]. Criteria that directly incorporate constraints would improve results in constrained optimization settings. With respect to adaptivity, there is also much work left to do. This thesis focused on spatial adaptivity for its applications, but there are interesting applications that greatly benefit from dimensional adaptivity, for example plasma physics [Pfl14]. Another key task would be the introduction of h-p-adaptivity to B-splines on sparse grids, which would greatly enhance the applicability of B-splines in non-smooth scenarios. As a simple special case, one could investigate different B-spline degrees in different dimensions. However, true h-p-adaptivity would allow to locally choose both the spatial resolution h and the B-spline degree p, adapting them according to the local smoothness of the function.



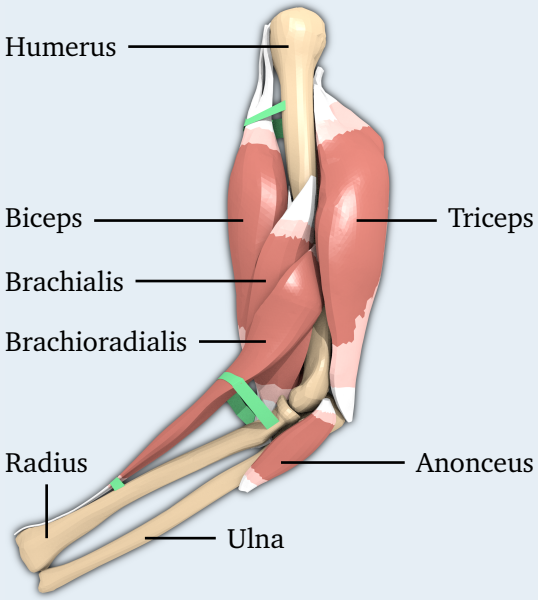


FIGURE 9.1
Extended model of the human upper limb with the three bones humerus, radius, and ulna *(light brown)* and the five muscles triceps, biceps, brachialis, brachioradialis, and anonceus *(red)*. Each muscle has associated tendon *(white)* and muscle-tendon complexes *(light red)*. Biceps and brachioradialis have each been fixed with one and two bands *(green)* to simulate the effect of the missing skin. Without bands, the muscles would unnaturally raise from the bones.

With regard to the application side, there are also quite a few possibilities for future work. The sparse grids in the biomechanical application we considered in this thesis were only two-dimensional. This is not in the range of dimensions in which sparse grids demonstrate their full strength, although the two-dimensional surrogates were already able to drastically reduce the required computation time compared to full grids. Currently, an extended model with five muscles and therefore five-dimensional sparse grids is being considered (see Fig. 9.1). Here, it will be mandatory to employ spatial adaptivity to cope with the increased dimensionality. In the application of topology optimization, more complicated micro-cell models and more complicated settings could be studied. For example, the widths of the diagonal macro-bars could be constrained [All16]. The dynamic portfolio choice models in the financial application were quite limited. For example, there was no inheritance motive (i.e., bequest), the model did not contain the individual's regular income, and the model did not account for savings for large necessary investments (e.g., cars or houses), which seems unnecessarily unrealistic. Finally, one could consider many other real-world optimization problems or other application fields of B-splines on sparse grids, for instance, data mining or uncertainty quantification. If objective gradients are available besides function values, it might be feasible to directly incorporate the gradients into the interpolation scheme [Baa15].

This extensive but by no means exhaustive list of possible future work can be seen as an inspiration and starting point for new and interesting applications of B-splines for sparse grids.

Proofs

This chapter contains proofs that were too long or too technical to include them in the main text. For convenience, the corresponding propositions and theorems are repeated before the proofs, using the same numbering as in their original chapter.

A.1 Proofs for Chapter 2

A.1.1 Proof of the Size of the Regular Sparse Grid with Coarse Boundaries

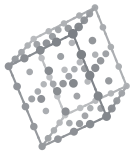
PROPOSITION 2.10 (number of regular sparse grid points with coarse boundary)

(2.38) $$|\Omega_{n,d}^{\mathrm{s}(b)}| = |\mathring{\Omega}_{n,d}^{\mathrm{s}}| + \sum_{q=1}^{d} 2^q \binom{d}{q} |\mathring{\Omega}_{n-q-b+1,d-q}^{\mathrm{s}}|, \quad b \in \mathbb{N}$$

PROOF Note that the outer union in the definition of $L_{n,d}^{\mathrm{s}(b)}$ in (2.37b) is indeed disjoint. Therefore,

(A.1) $$|\Omega_{n,d}^{\mathrm{s}(b)}| = \sum_{\substack{\boldsymbol{\ell} \in \mathbb{N}^d \\ \|\boldsymbol{\ell}\|_1 \le n}} |I_{\boldsymbol{\ell}}| + \sum_{\substack{\boldsymbol{\ell} \in \mathbb{N}_0^d \setminus \mathbb{N}^d \\ (\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \le n-b+1) \vee (\boldsymbol{\ell} = \mathbf{0})}} |I_{\boldsymbol{\ell}}|.$$

The first sum is the number $|\mathring{\Omega}_{n,d}^{\mathrm{s}}|$ of interior grid points in $\Omega_{n,d}^{\mathrm{s}}$. The second sum can be split into summands with the same number q of zero entries, which we count with



$N_{\boldsymbol{\ell}} := |\{t \mid \ell_t = 0\}| = \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 - \|\boldsymbol{\ell}\|_1$, and the same level sum $m = \|\boldsymbol{\ell}\|_1$:

$$|\Omega_{n,d}^{\mathrm{s}(b)}| = |\mathring{\Omega}_{n,d}^{\mathrm{s}}| + 2^d + \sum_{q=1}^{d-1} \sum_{m=d-q}^{n-b-q+1} \sum_{\substack{\boldsymbol{\ell} \in \mathbb{N}_0^d \\ N_{\boldsymbol{\ell}} = q,\, \|\boldsymbol{\ell}\|_1 = m}} |I_{\boldsymbol{\ell}}|, \tag{A.2}$$

where 2^d is the summand for $\boldsymbol{\ell} = \mathbf{0}$ (number $|I_{\mathbf{0}}|$ of corners of $[\mathbf{0}, \mathbf{1}]$). The limits of the sum over m are $d-q$, since there are $d-q$ entries ≥ 1 in a level vector with q zero entries, and $n-b-q+1$, since $m = \|\boldsymbol{\ell}\|_1 = \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 - N_{\boldsymbol{\ell}} \leq n-b+1-N_{\boldsymbol{\ell}} = n-b-q+1$.

In general, the innermost summand $|I_{\boldsymbol{\ell}}|$ equals $|I_{\boldsymbol{\ell}}| = \prod_{\{t|\ell_t \geq 1\}} 2^{\ell_t - 1} \cdot \prod_{\{t|\ell_t = 0\}} 2 = 2^{\|\boldsymbol{\ell}\|_1 - d + 2N_{\boldsymbol{\ell}}}$. The number of innermost summands is given by

$$|\{\boldsymbol{\ell} \in \mathbb{N}_0^d \mid N_{\boldsymbol{\ell}} = q,\ \|\boldsymbol{\ell}\|_1 = m\}| = \binom{d}{q} |\{\boldsymbol{\ell} \in \mathbb{N}^{d-q} \mid \|\boldsymbol{\ell}\|_1 = m\}| = \binom{d}{q} \binom{m-1}{d-q-1}. \tag{A.3}$$

This can be seen by first putting q zeros in d places, for which there are $\binom{d}{q}$ possibilities, and then counting all positive vectors of length $d-q$ with level sum m, which can be done in $\binom{m-1}{d-q-1}$ ways. Thus,

$$|\Omega_{n,d}^{\mathrm{s}(b)}| = |\mathring{\Omega}_{n,d}^{\mathrm{s}}| + 2^d + \sum_{q=1}^{d-1} \binom{d}{q} \sum_{m=d-q}^{n-b-q+1} 2^{m-d+2q} \binom{m-1}{d-q-1}. \tag{A.4a}$$

After shifting the index $m \to (m+d-q)$ and slightly rearranging the terms, we obtain

$$\cdots = |\mathring{\Omega}_{n,d}^{\mathrm{s}}| + 2^d + \sum_{q=1}^{d-1} 2^q \binom{d}{q} \sum_{m=0}^{n-d-b+1} 2^m \binom{(d-q)-1+m}{(d-q)-1}. \tag{A.4b}$$

We can now use Lemma 2.8 (number of interior regular sparse grid points) to conclude that

$$\cdots = |\mathring{\Omega}_{n,d}^{\mathrm{s}}| + \sum_{q=1}^{d} 2^q \binom{d}{q} |\mathring{\Omega}_{n-q-b+1,d-q}^{\mathrm{s}}| \tag{A.4c}$$

as desired. ∎

A.1.2 Correctness Proof of the Construction of the Regular Sparse Grid with Coarse Boundaries

PROPOSITION 2.11 (invariant of SG generation with coarse boundary)
After iteration t of Alg. 2.1 ($t = 1, \dots, d$), it holds

$$L^{(t)} = \{\boldsymbol{\ell} \in \mathbb{N}^t \mid \|\boldsymbol{\ell}\|_1 \leq n-d+t\} \\ \dot\cup \left(\{\boldsymbol{\ell} \in \mathbb{N}_0^t \setminus \mathbb{N}^t \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n-d+t-b+1\} \cup \{\mathbf{0}\}\right). \tag{2.39}$$

PROOF First, we show that every inserted level $\boldsymbol{\ell}' \in \mathbb{N}_0^t$ in the inner loop can be found on the right-hand side of (2.39). If $\boldsymbol{\ell}' := (\boldsymbol{\ell}, 0)$ is inserted for some $\boldsymbol{\ell} \in L^{(t-1)}$, then we have $\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n-d+t-b$ or $\boldsymbol{\ell} = \mathbf{0}$ by line 6 of Alg. 2.1. In the first case, we have

$$\|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 = \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 + 1 \leq n-d+t-b+1, \tag{A.5}$$

and in the second case $\boldsymbol{\ell}' = \mathbf{0}$. In either case, $\boldsymbol{\ell}'$ is contained in the right-hand side (RHS) of (2.39).

If $\boldsymbol{\ell}' := (\boldsymbol{\ell}, \ell_t)$ is inserted for some $\boldsymbol{\ell} \in L^{(t-1)}$ and $\ell_t \in \{1, \dots, \ell^*\}$, then there are, depending on whether $\boldsymbol{\ell} \in \mathbb{N}^{t-1}$, two cases:

- If $\boldsymbol{\ell} \in \mathbb{N}^{t-1}$, then $\boldsymbol{\ell}' \in \mathbb{N}^t$ and $\|\boldsymbol{\ell}'\|_1 \leq \|\boldsymbol{\ell}\|_1 + \ell^* = n-d+t$ due to line 9, i.e., $\boldsymbol{\ell}'$ is contained in the first set of the RHS of (2.39).
- If $\boldsymbol{\ell} \notin \mathbb{N}^{t-1}$, then $\boldsymbol{\ell}' \notin \mathbb{N}^t$ and $\|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 \leq \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 + \ell^* = n-d+t-b+1$ due to line 11, i.e., $\boldsymbol{\ell}$ is contained in the second set of the RHS of (2.39).

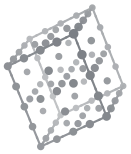
Thus, all levels that the algorithm inserts into $L^{(t)}$ can be found on the RHS of (2.39).

It remains to prove that all levels on the RHS of (2.39) are eventually inserted by the algorithm into $L^{(t)}$. We prove this by induction over $t = 1, \dots, d$. For $t = 1$, the RHS of (2.39) equals $\{\ell \in \mathbb{N}_0 \mid \ell \leq n-d+1\}$, which is just $L^{(1)}$ (see line 2 of Alg. 2.1). For the induction step $(t-1) \to t$, we assume the validity of the induction hypothesis

$$L^{(t-1)} = \{\boldsymbol{\ell} \in \mathbb{N}^{t-1} \mid \|\boldsymbol{\ell}\|_1 \leq n-d+t-1\} \dot\cup \\ \left(\{\boldsymbol{\ell} \in \mathbb{N}_0^{t-1} \setminus \mathbb{N}^{t-1} \mid \|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 \leq n-d+t-b\} \cup \{\mathbf{0}\}\right). \tag{A.6}$$

The RHS of (2.39) has three parts, so we check for elements $\boldsymbol{\ell}' \in \mathbb{N}_0^t$ of each of the three sets that they are appended to $L^{(t)}$ eventually.

First, let $\boldsymbol{\ell}' = (\boldsymbol{\ell}, \ell_t)$ be in the first set of the RHS, i.e., $\boldsymbol{\ell}' \in \mathbb{N}^t$ (in particular $\ell_t \geq 1$) and $\|\boldsymbol{\ell}'\|_1 \leq n-d+t$. Note that $\boldsymbol{\ell}$ will be encountered in the inner loop, as $\boldsymbol{\ell} \in \mathbb{N}^{t-1}$ and $\|\boldsymbol{\ell}\|_1 = \|\boldsymbol{\ell}'\|_1 - \ell_t \leq n-d+t-1$, which implies $\boldsymbol{\ell} \in L^{(t-1)}$ by the induction hypothesis



(A.6). Since $1 \le \ell_t \le \ell^*$ (due to $\ell_t = \|\boldsymbol{\ell}'\|_1 - \|\boldsymbol{\ell}\|_1 \le n-d+t-\|\boldsymbol{\ell}\|_1 = \ell^*$), the level $\boldsymbol{\ell}'$ is inserted into $L^{(t)}$ during the innermost loop in line 12 of Alg. 2.1.

Second, let $\boldsymbol{\ell}' = (\boldsymbol{\ell}, \ell_t)$ be in the second set of the RHS, i.e., we have $\boldsymbol{\ell}' \notin \mathbb{N}^t$ and $\|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 \le n-d+t-b+1$. Here, there are three cases:

1. $\ell_t \ge 1$: This implies $\boldsymbol{\ell} \notin \mathbb{N}^{t-1}$ and $\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 = \|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 - \ell_t \le n-d+t-b$. Consequently, $\boldsymbol{\ell} \in L^{(t-1)}$ by the induction hypothesis (A.6). As $1 \le \ell_t \le \ell^*$ (due to $\ell_t \le n-d+t-b+1-\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 = \ell^*$), $\boldsymbol{\ell}$ is added to $L^{(t)}$ in line 12.
2. $\ell_t = 0$ and $\boldsymbol{\ell} \in \mathbb{N}^{t-1}$: This implies $\|\boldsymbol{\ell}\|_1 = \|\boldsymbol{\ell}'\|_1 = \|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 - 1 \le n-d+t-b \le n-d+t-1$ since $b \ge 1$. Again, by the induction hypothesis (A.6), $\boldsymbol{\ell}$ is added to $L^{(t)}$ in line 7 due to $\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 = \|\boldsymbol{\ell}\|_1 \le n-d+t-b$.
3. $\ell_t = 0$ and $\boldsymbol{\ell} \notin \mathbb{N}^{t-1}$: This implies $\|\mathbf{max}(\boldsymbol{\ell}, \mathbf{1})\|_1 = \|\mathbf{max}(\boldsymbol{\ell}', \mathbf{1})\|_1 - 1 \le n-d+t-b$. Again, by the induction hypothesis (A.6), $\boldsymbol{\ell}$ is added to $L^{(t)}$ in line 7.

Third, let $\boldsymbol{\ell} = (\mathbf{0}, 0) \in \mathbb{N}_0^t$ be in the third set of the RHS. This level is appended in line 7 to $L^{(t)}$, since $\boldsymbol{\ell}' = \mathbf{0} \in \mathbb{N}_0^{t-1}$ is in $L^{(t-1)}$ by the induction hypothesis (A.6). ■

A.2 Proofs for Chapter 3

A.2.1 Proof of the Linear Independence of Hierarchical B-Splines

PROPOSITION 3.5 (hierarchical B-splines are linearly independent)
The hierarchical B-splines $\varphi^p_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$) are linearly independent.

PROOF The proof is rigorous for the common B-spline degrees of $p \in \{1,3,5,7\}$. For higher degrees, the proof has to be viewed as a sketch.

We follow the presentation in [Vale16] and prove the assertion by induction over $\ell \in \mathbb{N}_0$. For $\ell = 0$, the B-splines $\varphi^p_{0,i'}$ with $i' \in \{0,1\}$ are linearly independent. For the induction step $(\ell-1) \to \ell$, let

$$\sum_{\ell'=0}^{\ell} \sum_{i' \in I_{\ell'}} \alpha_{\ell',i'} \varphi^p_{\ell',i'} \equiv 0 \tag{A.7}$$

be a linear combination of the zero function. We separate the summands of level ℓ from the summands of coarser levels $\ell' < \ell$:

$$\sum_{i \in I_\ell} \alpha_{\ell,i} \varphi^p_{\ell,i} =: g_1 \equiv g_2 := -\sum_{\ell'=0}^{\ell-1} \sum_{i' \in I_{\ell'}} \alpha_{\ell',i'} \varphi^p_{\ell',i'}. \tag{A.8}$$

The right-hand side g_2 is smooth in every grid point $x_{\ell,i}$ of level ℓ ($i \in I_\ell$), since these grid points are not knots of the hierarchical B-splines $\varphi^p_{\ell',i'}$ of level $\ell' < \ell$ ($i' \in I_{\ell'}$). This implies that the left-hand side g_1 must be smooth there as well:

$$\underbrace{\sum_{i\in I_\ell} \alpha_{\ell,i}\, \partial_-^p \varphi^p_{\ell,i}(x_{\ell,i'})}_{=\partial_-^p g_1(x_{\ell,i'})} = \underbrace{\sum_{i\in I_\ell} \alpha_{\ell,i}\, \partial_+^p \varphi^p_{\ell,i}(x_{\ell,i'})}_{=\partial_+^p g_1(x_{\ell,i'})}, \quad i' \in I_\ell, \tag{A.9}$$

where ∂_-^p and ∂_+^p denote the left and right derivative of order p, respectively. By repeated application of (3.3), one can show that

$$\partial_-^p b^p(k+1) = (-1)^k \binom{p}{k} = \partial_+^p b^p(k), \quad k \in \mathbb{Z}, \tag{A.10}$$

where $\binom{p}{k} = 0$ for $k < 0$ or $k > p$ [Höl13]. We can insert this relation into (A.9) and use (3.5) to obtain

$$\sum_{i\in I_\ell} \alpha_{\ell,i} (-1)^{k-1} \binom{p}{k-1} = \sum_{i\in I_\ell} \alpha_{\ell,i} (-1)^k \binom{p}{k}, \quad i' \in I_\ell, \quad k := \frac{p+1}{2} + i' - i. \tag{A.11}$$

As $\binom{p}{k-1} + \binom{p}{k} = \binom{p+1}{k}$ and $(-1)^k$ is constant for $i \in I_\ell$ when i' is fixed, this is equivalent to

$$\sum_{i\in I_\ell} \alpha_{\ell,i} \binom{p+1}{\frac{p+1}{2} + i' - i} = 0, \quad i' \in I_\ell. \tag{A.12}$$

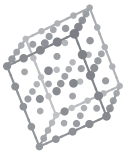
This is a square system of linear equations whose system matrix $\boldsymbol{A}(p)$ is a banded symmetric Toeplitz matrix[1] of size $2^{\ell-1} \times 2^{\ell-1}$ with bandwidth $\lceil \frac{p-1}{4} \rceil$. The non-zero values of $\boldsymbol{A}(p)$ are tabulated for some degrees p in Tab. A.1. For $p = 1, 3, 5, 7$, the corresponding matrices are diagonally dominant and therefore regular. For higher B-spline degrees p, the regularity of $\boldsymbol{A}(p)$ has to be shown differently.

Due to the regularity of $\boldsymbol{A}(p)$, we infer from (A.12) that $\alpha_{\ell,i} = 0$ for $i \in I_\ell$. According to (A.7), we obtain a linear combination of the zero function with the hierarchical B-splines of level $< \ell$, i.e.,

$$\sum_{\ell'=0}^{\ell-1} \sum_{i'\in I_{\ell'}} \alpha_{\ell',i'} \varphi^p_{\ell',i'} = 0, \tag{A.13}$$

which implies $\alpha_{\ell',i'} = 0$ for all $\ell' < \ell$ and $i' \in I_{\ell'}$ by the induction hypothesis. Thus, the

[1] The entries $A_{k,j}$ of a Toeplitz matrix $\boldsymbol{A}$ solely depend on $k - j$, i.e., $A_{k,j} = c_{k-j}$ for some vector $\boldsymbol{c}$.



	$k=0$	$k=1$	$k=2$	$k=3$
$p=1$	2			
$p=3$	6	1		
$p=5$	20	6		
$p=7$	70	28	1	
$p=9$	252	120	10	
$p=11$	924	495	66	1

TABLE A.1 Non-zero values $A_{j,j+k}(p)$ of the diagonals of $\boldsymbol{A}(p)$ obtained in (A.12).

hierarchical B-splines $\varphi^p_{\ell',i'}$ ($\ell' \le \ell$, $i' \in I_{\ell'}$) are linearly independent. ■

A.3 Proofs for Chapter 4

A.3.1 Combinatorial Proof of the Combination Technique

DEFINITION A.1 (binomial coefficient for integer parameters)
The binomial coefficient $\binom{n}{k}$ is defined for $n \in \mathbb{N}_0$ and $k \in \mathbb{Z}$ as

$$\binom{n}{k} := \begin{cases} \frac{n(n-1)\cdots(n-(k-1))}{k!}, & 0 < k < n, \\ 1, & (k=0) \vee (k=n), \\ 0, & (k<0) \vee (k>n). \end{cases} \tag{A.14}$$

LEMMA A.2 (inclusion-exclusion counting lemma)
For $a \in \mathbb{N}_0$, $r \ge a$, and $s \in \mathbb{Z}$, we have

$$\sum_{q=0}^{a} (-1)^q \binom{a}{q}\binom{r-q}{s} = \binom{r-a}{s-a}. \tag{A.15}$$

PROOF We apply the upper negation formula (see Eq. (5.14) of [Gra94]) to the second binomial of the left-hand side (LHS):

$$\sum_{q=0}^{a} (-1)^q \binom{a}{q}\binom{r-q}{s} = (-1)^s \sum_{q=0}^{a} \binom{a}{0+q}\binom{(s-r-1)+q}{s}(-1)^q. \tag{A.16a}$$

This sum can be simplified using the identity in Eq. (5.24) of [Gra94] (the sum has already been written in the same way as in [Gra94]):

$$\cdots = (-1)^{s+a}\binom{s-r-1}{s-a}. \tag{A.16b}$$

Applying the upper negation formula again,

$$\cdots = \binom{r-a}{s-a}, \tag{A.16c}$$

we obtain the desired quantity. ∎

PROPOSITION 4.4 (inclusion-exclusion principle)
For every $\boldsymbol{x}_{\ell,i} \in \Omega^s_{n,d}$, we have

$$\sum_{q=0}^{d-1}(-1)^q\binom{d-1}{q}\cdot|\{\boldsymbol{\ell}' \mid \|\boldsymbol{\ell}'\|_1 = n-q,\ \Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\ell,i}\}| = 1. \tag{4.19}$$

PROOF Let $q = 0,\dots,d-1$ and $\boldsymbol{x}_{\ell,i} \in \Omega^s_{n,d}$, i.e., $\|\boldsymbol{\ell}\|_1 \le n$ and $\boldsymbol{i} \in I_\ell$. Note that for $\boldsymbol{\ell}' \in \mathbb{N}_0^d$, we have $\Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\ell,i} \iff \boldsymbol{\ell}' \ge \boldsymbol{\ell}$. Hence,

$$|\{\boldsymbol{\ell}' \mid \|\boldsymbol{\ell}'\|_1 = n-q,\ \Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\ell,i}\}| = |\{\boldsymbol{\ell}' \mid \|\boldsymbol{\ell}'\|_1 = n-q,\ \boldsymbol{\ell}' \ge \boldsymbol{\ell}\}| \tag{A.17a}$$

$$= |\{\boldsymbol{a} \in \mathbb{N}_0^d \mid \|\boldsymbol{a}\|_1 = n-q-\|\boldsymbol{\ell}\|_1\}| \tag{A.17b}$$

by mapping $\boldsymbol{a} := \boldsymbol{\ell}' - \boldsymbol{\ell}$. The size of the last set is known as the number of *weak compositions* of $n-q-\|\boldsymbol{\ell}\|_1$ into d parts and can be computed as

$$\cdots = \binom{n-q-\|\boldsymbol{\ell}\|_1+d-1}{d-1}, \tag{A.17c}$$

see Theorem 2.2 of [Bón15]. Now, we can use Lemma A.2 with the values $a := s := d-1$ and $r := n-\|\boldsymbol{\ell}\|_1+d-1$ to conclude that the LHS of the assertion (4.19) equals

$$\sum_{q=0}^{d-1}(-1)^q\binom{d-1}{q}\cdot\binom{n-q-\|\boldsymbol{\ell}\|_1+d-1}{d-1} = \binom{n-\|\boldsymbol{\ell}\|_1}{0} = 1, \tag{A.18}$$

proving the proposition. ∎

LEMMA A.3 *$\sim$ is an equivalence relation.*

PROOF We check reflexivity, symmetry, and transitivity of $\sim$:

- *Reflexivity:* Using the same level $\boldsymbol{\ell}' = \boldsymbol{\ell}''$ implies $T_{\boldsymbol{\ell}',\boldsymbol{\ell}'} = \{t \mid \ell'_t < \ell_t\}$. For all $t \notin T_{\boldsymbol{\ell}',\boldsymbol{\ell}'}$, we have $\ell'_t \geq \ell_t$. Consequently, $\boldsymbol{\ell}' \sim \boldsymbol{\ell}'$.
- *Symmetry:* We have $\boldsymbol{\ell}' \sim \boldsymbol{\ell}'' \iff \boldsymbol{\ell}'' \sim \boldsymbol{\ell}'$, since $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''} = T_{\boldsymbol{\ell}'',\boldsymbol{\ell}'}$ and $\min\{\ell'_t, \ell''_t\} = \min\{\ell''_t, \ell'_t\}$.
- *Transitivity:* Let $\boldsymbol{\ell}' \sim \hat{\boldsymbol{\ell}}$, $\hat{\boldsymbol{\ell}} \sim \boldsymbol{\ell}''$, and $t \notin T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$. From the definition of $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$, it holds that either $\ell'_t \neq \ell''_t$ or $\ell'_t = \ell''_t \geq \ell_t$. As $\ell'_t = \ell''_t \geq \ell_t$ already implies $\min\{\ell'_t, \ell''_t\} \geq \ell_t$, we assume that $\ell'_t \neq \ell''_t$. Here, we have three cases:
  - *Case 1:* $\ell'_t \neq \hat{\ell}_t = \ell''_t$. $t \notin T_{\boldsymbol{\ell}',\hat{\boldsymbol{\ell}}}$ implies $\ell'_t \geq \ell_t$ and $\ell''_t = \hat{\ell}_t \geq \ell_t$. Therefore, $\min\{\ell'_t, \ell''_t\} \geq \ell_t$.
  - *Case 2:* $\ell'_t = \hat{\ell}_t \neq \ell''_t$. Analogously to the first case, we conclude $\min\{\ell'_t, \ell''_t\} \geq \ell_t$.
  - *Case 3:* $\ell'_t \neq \hat{\ell}_t \neq \ell''_t$. $t \notin T_{\boldsymbol{\ell}',\hat{\boldsymbol{\ell}}}$ implies $\ell'_t \geq \ell_t$ and $t \notin T_{\boldsymbol{\ell}'',\hat{\boldsymbol{\ell}}}$ implies $\ell''_t \geq \ell_t$. Hence, $\min\{\ell'_t, \ell''_t\} \geq \ell_t$.

  Therefore, it holds that $\min\{\ell'_t, \ell''_t\} \geq \ell_t$ for all $t \notin T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$, i.e., $\boldsymbol{\ell}' \sim \boldsymbol{\ell}''$.

This shows that $\sim$ is an equivalence relation. ■

LEMMA 4.6 *Let $\boldsymbol{\ell}', \boldsymbol{\ell}'' \in L$ with $\boldsymbol{\ell}' \sim \boldsymbol{\ell}''$. Then, $f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\boldsymbol{\ell},i}) = f_{\boldsymbol{\ell}''}(\boldsymbol{x}_{\boldsymbol{\ell},i})$.*

PROOF First, we note that $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''} \neq \emptyset$. Otherwise, for $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''} = \emptyset$, we have $\min\{\ell'_t, \ell''_t\} \geq \ell_t$ for all $t = 1, \ldots, d$, which implies $\boldsymbol{\ell}' \geq \boldsymbol{\ell}$, i.e., $\Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\boldsymbol{\ell},i}$. This contradicts the fact that $\boldsymbol{\ell}' \in L$, where L is defined in (4.20) (which holds as our equivalence relation is only defined on L). Therefore, $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''} \neq \emptyset$ must hold. Without loss of generality, we assume that $T_{\boldsymbol{\ell}',\boldsymbol{\ell}''} = \{1, \ldots, m\}$ for some $m \in \{1, \ldots, d\}$.

Let

$$S := \boldsymbol{x}_{\boldsymbol{\ell},i} + \operatorname{span}\{\boldsymbol{e}_1, \ldots, \boldsymbol{e}_m\} = \{\boldsymbol{x}_{\boldsymbol{\ell},i} + \textstyle\sum_{t=1}^{m} c_t \boldsymbol{e}_t \mid c_1, \ldots, c_m \in \mathbb{R}\} \tag{A.19}$$

be the m-dimensional affine subspace of $\mathbb{R}^d$ through $\boldsymbol{x}_{\boldsymbol{\ell},i}$ parallel to the dimensions $1, \ldots, m$, where $\boldsymbol{e}_t$ is the t-th standard basis vector. It holds that $S \cap \Omega_{\boldsymbol{\ell}'} = S \cap \Omega_{\boldsymbol{\ell}''}$ due to $\ell'_t = \ell''_t$ for $t \leq m$.[2]

On this m-dimensional grid $S \cap \Omega_{\boldsymbol{\ell}'} = S \cap \Omega_{\boldsymbol{\ell}''}$, the full grid interpolants $f_{\boldsymbol{\ell}'}$ and $f_{\boldsymbol{\ell}''}$ coincide, as both interpolate the function values given by the objective function f:

$$f_{\boldsymbol{\ell}'}|_{S \cap \Omega_{\boldsymbol{\ell}'}} = f|_{S \cap \Omega_{\boldsymbol{\ell}'}} = f_{\boldsymbol{\ell}''}|_{S \cap \Omega_{\boldsymbol{\ell}''}}. \tag{A.20}$$

[2] In more detail: If we have an $\boldsymbol{x}_{\hat{\boldsymbol{\ell}},\hat{\boldsymbol{i}}} \in S \cap \Omega_{\boldsymbol{\ell}'}$, then $\forall_{t \leq m}\, \hat{\ell}_t \leq \ell'_t = \ell''_t$ and $\forall_{t > m}\, \hat{\ell}_t = \ell_t \leq \ell''_t$, i.e., $\hat{\boldsymbol{\ell}} \leq \boldsymbol{\ell}''$ and therefore $\boldsymbol{x}_{\hat{\boldsymbol{\ell}},\hat{\boldsymbol{i}}} \in S \cap \Omega_{\boldsymbol{\ell}''}$.

However, this does not suffice to conclude $f_{\ell'}(\boldsymbol{x}_{\ell,i}) = f_{\ell''}(\boldsymbol{x}_{\ell,i})$, since $\boldsymbol{x}_{\ell,i} \notin \Omega_{\ell'}$.

To this end, we recall from (2.11) that

$$f_{\ell'} = \sum_{\boldsymbol{i}'=\boldsymbol{0}}^{2^{\ell'}} c_{\ell',\boldsymbol{i}'} \varphi_{\ell',\boldsymbol{i}'}, \quad c_{\ell',\boldsymbol{i}'} \in \mathbb{R}. \tag{A.21}$$

This implies that the m-variate restricted interpolant $f_{\ell'}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]}$ can be written as

$$(f_{\ell'}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]})(\boldsymbol{x}_{1:m}) = \sum_{\boldsymbol{i}'_{1:m}=\boldsymbol{0}}^{2^{\ell'_{1:m}}} \tilde{c}_{\ell'_{1:m},\boldsymbol{i}'_{1:m}} \varphi_{\ell'_{1:m},\boldsymbol{i}'_{1:m}}(\boldsymbol{x}_{1:m}), \quad \boldsymbol{x}_{1:m} \in [0,1]^m, \tag{A.22a}$$

$$\tilde{c}_{\ell'_{1:m},\boldsymbol{i}'_{1:m}} := \sum_{\boldsymbol{i}'_{m+1:d}=\boldsymbol{0}}^{2^{\ell'_{m+1:d}}} c_{\ell',\boldsymbol{i}'} \varphi_{\ell'_{m+1:d},\boldsymbol{i}'_{m+1:d}}(\boldsymbol{x}_{\ell_{m+1:d},\boldsymbol{i}_{m+1:d}}) \tag{A.22b}$$

by factoring out tensor product factors corresponding to dimensions $m+1,\dots,d$. The subscripts $1:m$ and $m+1:d$ denote the entries with respect to the dimensions $1,\dots,m$ and $m+1,\dots,d$, respectively. As a result, both $f_{\ell'}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]}$ and, analogously, $f_{\ell''}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]}$ are interpolants of f in $V_{\ell'_{1:m}} = V_{\ell''_{1:m}}$. Due to Lemma 2.1 (linear independence of tensor products), it follows from (A.20) that they must be the same:

$$f_{\ell'}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]} = f_{\ell''}|_{S\cap[\boldsymbol{0},\boldsymbol{1}]}. \tag{A.23}$$

Consequently, $f_{\ell'}(\boldsymbol{x}_{\ell,i}) = f_{\ell''}(\boldsymbol{x}_{\ell,i})$ as $\boldsymbol{x}_{\ell,i} \in S \cap [\boldsymbol{0},\boldsymbol{1}]$. ■

LEMMA 4.7 (characterization of equivalence classes)
Let $L_0 \in L/\!\sim$ be an equivalence class of $\sim$. If we define

$$T_{L_0} := \{t \mid \exists_{\ell_t^* < \ell_t} \forall_{\ell' \in L_0} \ell'_t = \ell_t^*\} \tag{4.22}$$

as the set of dimensions t in which all levels in L_0 have the same entry $\ell_t^* < \ell_t$, then

$$L_0 = \{\boldsymbol{\ell}' \in L \mid \forall_{t\in T_{L_0}} \ell'_t = \ell_t^*,\ \forall_{t\notin T_{L_0}} \ell'_t \geq \ell_t\}. \tag{4.23}$$

PROOF "$\subseteq$": Let $\boldsymbol{\ell}' \in L_0$. We have to prove that $\forall_{t\in T_{L_0}} \ell'_t = \ell_t^*$ and $\forall_{t\notin T_{L_0}} \ell'_t \geq \ell_t$. The first statement is clear by the definition of T_{L_0}. Therefore, let $t \notin T_{L_0}$. By the definition of T_{L_0}, we have either $\exists_{\hat{\boldsymbol{\ell}}\in L_0} \ell'_t \neq \hat{\ell}_t$ or $\forall_{\hat{\boldsymbol{\ell}}\in L_0} \ell'_t = \hat{\ell}_t \geq \ell_t$. In the latter case, we obtain $\ell'_t \geq \ell_t$ (e.g., by setting $\hat{\boldsymbol{\ell}}$ to $\boldsymbol{\ell}'$). In the former case, there is an $\hat{\boldsymbol{\ell}} \in L_0$ such that $\ell'_t \neq \hat{\ell}_t$. Due to $\boldsymbol{\ell}' \sim \hat{\boldsymbol{\ell}}$ (since $\boldsymbol{\ell}'$ and $\hat{\boldsymbol{\ell}}$ are both contained in the same equivalence class L_0) and $t \notin T_{\boldsymbol{\ell}',\hat{\boldsymbol{\ell}}}$, we have $\min\{\ell'_t, \hat{\ell}_t\} \geq \ell_t$. This implies $\forall_{t\notin T_{L_0}} \ell'_t \geq \ell_t$, as desired.

"$\supseteq$": Let $\boldsymbol{\ell}' \in L$ such that $\forall_{t \in T_{L_0}} \ell'_t = \ell^*_t$ and $\forall_{t \notin T_{L_0}} \ell'_t \geq \ell_t$. Furthermore, let $\boldsymbol{\ell}'' \in L_0$ be an arbitrary representative of L_0. We prove that $\boldsymbol{\ell}' \sim \boldsymbol{\ell}''$ (i.e., $\boldsymbol{\ell}' \in L_0$). Note that $T_{L_0} \subseteq T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$, as $t \in T_{L_0}$ implies $\ell''_t = \ell^*_t < \ell_t$, which can be combined with $\ell'_t = \ell^*_t$ to $\ell'_t = \ell''_t < \ell_t$, i.e., $t \in T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$.

To prove the equivalence of $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$, let $t \notin T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$, i.e., $t \notin T_{L_0}$. By assumption on $\boldsymbol{\ell}'$, it holds $\ell'_t \geq \ell_t$. Hence, it remains to show that $\ell''_t \geq \ell_t$ as well. Again, by definition of T_{L_0}, we have either $\exists_{\hat{\boldsymbol{\ell}} \in L_0} \ell''_t \neq \hat{\ell}_t$ or $\forall_{\hat{\boldsymbol{\ell}} \in L_0} \ell''_t = \hat{\ell}_t \geq \ell_t$. In the second case, it holds $\ell''_t \geq \ell_t$. In the first case, there is an $\hat{\boldsymbol{\ell}} \in L_0$ such that $\ell''_t \neq \hat{\ell}_t$. Due to $\boldsymbol{\ell}'' \sim \hat{\boldsymbol{\ell}}$ (since $\boldsymbol{\ell}''$ and $\hat{\boldsymbol{\ell}}$ are both contained in the same equivalence class L_0) and $t \notin T_{\boldsymbol{\ell}'',\hat{\boldsymbol{\ell}}}$, we have $\min\{\ell''_t, \hat{\ell}_t\} \geq \ell_t$. In particular, $\ell''_t \geq \ell_t$. In total, we have $\min\{\ell'_t, \ell''_t\} \geq \ell_t$ for all $t \notin T_{\boldsymbol{\ell}',\boldsymbol{\ell}''}$, proving that $\boldsymbol{\ell}'$ and $\boldsymbol{\ell}''$ are equivalent, as asserted. ■

PROPOSITION 4.8 (function value cancellation)
For every $\boldsymbol{x}_{\ell,i} \in \Omega^s_{n,d}$, we have

$$\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\boldsymbol{\ell}'} \not\ni \boldsymbol{x}_{\ell,i}}} f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\ell,i}) = 0. \tag{4.24}$$

PROOF Lemma 4.6 implies that the summands $f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\ell,i})$ corresponding to levels $\boldsymbol{\ell}'$ of the same equivalence class $L_0 \in L/\!\sim$ are identical. Let f_{L_0} denote the common function value. The sum in the LHS of the assertion can be reordered to combine levels of the equivalence classes $L_0 \in L/\!\sim$:

$$\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \sum_{\substack{\|\boldsymbol{\ell}'\|_1 = n-q \\ \Omega_{\boldsymbol{\ell}'} \not\ni \boldsymbol{x}_{\ell,i}}} f_{\boldsymbol{\ell}'}(\boldsymbol{x}_{\ell,i}) \tag{A.24a}$$

$$= \sum_{L_0 \in L/\sim} f_{L_0} \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot |\{\boldsymbol{\ell}' \in L_0 \mid \|\boldsymbol{\ell}'\|_1 = n-q\}|. \tag{A.24b}$$

It now suffices to show that the inner sum vanishes for every equivalence class $L_0 \in L/\!\sim$.

To this end, we have to calculate $|\{\boldsymbol{\ell}' \in L_0 \mid \|\boldsymbol{\ell}'\|_1 = n-q\}|$ for a fixed equivalence class L_0. Without loss of generality, let $T_{L_0} = \{1, \ldots, m\}$ in the notation of Lemma 4.7 (characterization of equivalence classes) with $1 \leq m \leq d$. Note that the case $m = 0$ is impossible: Otherwise, $T_{L_0} = \emptyset$ implies $\forall_{\boldsymbol{\ell}' \in L_0} \boldsymbol{\ell}' \geq \boldsymbol{\ell}$ by Lemma 4.7, and as equivalence classes are non-empty, there is at least one $\boldsymbol{\ell}' \in L_0$ with $\boldsymbol{\ell}' \geq \boldsymbol{\ell}$. However, this is equivalent to $\Omega_{\boldsymbol{\ell}'} \ni \boldsymbol{x}_{\ell,i}$, which contradicts $\boldsymbol{\ell}' \in L$. Hence, we have $m > 0$.

To enumerate all levels $\boldsymbol{\ell}' \in L_0$ with $\|\boldsymbol{\ell}'\|_1 = n-q$, we exploit the characterization of

L_0 of Lemma 4.7. For notational convenience, we define the vector

$$\hat{\boldsymbol{\ell}} := (\ell_1^*, \ldots, \ell_m^*, \ell_{m+1}, \ldots, \ell_d), \tag{A.25}$$

where $\boldsymbol{\ell}^*$ is given as in Lemma 4.7. We show that $\boldsymbol{a} := (\ell'_t - \ell_t)_{t=m+1,\ldots,d}$ constitutes a bijection between

$$\{\boldsymbol{\ell}' \in L_0 \mid \|\boldsymbol{\ell}'\|_1 = n-q\} \quad \text{and} \quad \{\boldsymbol{a} \in \mathbb{N}_0^{d-m} \mid \|\boldsymbol{a}\|_1 = n-q-\|\hat{\boldsymbol{\ell}}\|_1\}: \tag{A.26}$$

- Let $\boldsymbol{\ell}' \in L_0$ with $\|\boldsymbol{\ell}'\|_1 = n-q$. Then, $\forall_{t=m+1,\ldots,d}\ \ell'_t - \ell_t \geq 0$ (by Lemma 4.7), i.e., $\boldsymbol{a} \in \mathbb{N}_0^{d-m}$, and

$$\|\boldsymbol{a}\|_1 = \sum_{t=m+1}^{d} (\ell'_t - \ell_t) = \left(\|\boldsymbol{\ell}'\|_1 - \sum_{t=1}^{m} \ell'_t\right) - \sum_{t=m+1}^{d} \hat{\ell}_t = n-q-\|\hat{\boldsymbol{\ell}}\|_1. \tag{A.27}$$

- Conversely, let $\boldsymbol{a} \in \mathbb{N}_0^{d-m}$ with $\|\boldsymbol{a}\|_1 = n-q-\|\hat{\boldsymbol{\ell}}\|_1$. If we define $\boldsymbol{\ell}'$ as

$$\boldsymbol{\ell}' = (\ell_1^*, \ldots, \ell_m^*, a_1 + \ell_{m+1}, \ldots, a_{d-m} + \ell_d), \tag{A.28}$$

  then $\forall_{t=1,\ldots,m}\ \ell'_t = \ell_t^* < \ell_t$ and $\forall_{t=m+1,\ldots,d}\ \ell'_t \geq \ell_t$. By Lemma 4.7, we obtain $\boldsymbol{\ell}' \in L_0$ and

$$\|\boldsymbol{\ell}'\|_1 = \|\hat{\boldsymbol{\ell}}\|_1 + \|\boldsymbol{a}\|_1 = n-q. \tag{A.29}$$

This bijection implies that

$$|\{\boldsymbol{\ell}' \in L_0 \mid \|\boldsymbol{\ell}'\|_1 = n-q\}| = |\{\boldsymbol{a} \in \mathbb{N}_0^{d-m} \mid \|\boldsymbol{a}\|_1 = n-q-\|\hat{\boldsymbol{\ell}}\|_1\}|. \tag{A.30a}$$

This is the number of weak decompositions of $n-q-\|\hat{\boldsymbol{\ell}}\|_1$ into $d-m$ parts:

$$\cdots = \binom{n-q-\|\hat{\boldsymbol{\ell}}\|_1 + d-m-1}{d-m-1}, \tag{A.30b}$$

see Theorem 2.2 of [Bón15]. We insert this quantity into the inner sum of (A.24b):

$$\sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot |\{\boldsymbol{\ell}' \in L_0 \mid \|\boldsymbol{\ell}'\|_1 = n-q\}| \tag{A.31a}$$

$$= \sum_{q=0}^{d-1} (-1)^q \binom{d-1}{q} \cdot \binom{n-q-\|\hat{\boldsymbol{\ell}}\|_1 + d-m-1}{d-m-1}. \tag{A.31b}$$

Again, we apply Lemma A.2 (inclusion-exclusion counting lemma) with the values $a := d-1$, $r := n - \|\hat{\boldsymbol{\ell}}\|_1 + d - m - 1$, and $s := d - m - 1$ to infer that as claimed, (A.31) is equal to

$$\binom{n - \|\hat{\boldsymbol{\ell}}\|_1 - m}{-m} = 0 \tag{A.32}$$

by the convention for binomial coefficients in Def. A.1 as $-m < 0$.

Note that for the calculation in Equations (A.31) and (A.32) to be correct, we have to ensure that $n - \|\hat{\boldsymbol{\ell}}\|_1 - m \geq 0$; otherwise, the binomial coefficients would not be well-defined. This is a direct consequence of the fact that $\ell_t^* < \ell_t$ for all $t = 1, \ldots, m$ (see Lemma 4.7) as

$$n - \|\hat{\boldsymbol{\ell}}\|_1 - m = n - \sum_{t=1}^{m} \ell_t^* - \sum_{t=m+1}^{d} \ell_t - m \geq n - \sum_{t=1}^{m} (\ell_t - 1) - \sum_{t=m+1}^{d} \ell_t - m = n - \|\boldsymbol{\ell}\|_1 \geq 0, \tag{A.33}$$

where we have used $\ell_t^* \leq \ell_t - 1$ for $t = 1, \ldots, m$ and $\|\boldsymbol{\ell}\|_1 \leq n$. ■

A.3.2 Correctness Proof of the Method of Residual Interpolation

PROPOSITION 4.10 (invariant of residual interpolation)
For $j = 1, \ldots, m$, it holds

$$r^{(j-1)}_{\boldsymbol{\ell}^{(j)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0, \qquad \boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j')},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}},\quad j' = 1, \ldots, j-1, \tag{4.28a}$$

$$r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0, \qquad \boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j')},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}},\quad j' = 1, \ldots, j, \tag{4.28b}$$

$$r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - f^{s,(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad \boldsymbol{\ell} \in L,\quad \boldsymbol{i} \in I_{\boldsymbol{\ell}}, \tag{4.28c}$$

$$\text{where} \quad f^{s,(j)} := \sum_{\boldsymbol{\ell}' \in L} \sum_{\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}} \left(\sum_{j'=1}^{j} \alpha_{\boldsymbol{\ell}',\boldsymbol{i}'}^{(j')} \right) \varphi_{\boldsymbol{\ell}',\boldsymbol{i}'}. \tag{4.29}$$

PROOF We prove the assertion by induction over $j = 1, \ldots, m$. We will need the following two equations that directly follow from the algorithm (lines 6 to 8, respectively):

$$r^{(j-1)}_{\boldsymbol{\ell}^{(j)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = r^{(j-1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \qquad \boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j)},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}}, \tag{A.34a}$$

$$r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = r^{(j-1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - r^{(j-1)}_{\boldsymbol{\ell}^{(j)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad \boldsymbol{\ell} \in L,\quad \boldsymbol{i} \in I_{\boldsymbol{\ell}}. \tag{A.34b}$$

Induction base case: For $j = 1$, there is nothing to show for (4.28a). Equation (4.28b) can be proven as follows:

$$r^{(1)}(\boldsymbol{x}_{\ell,i}) \overset{\text{(A.34b)}}{=} r^{(0)}(\boldsymbol{x}_{\ell,i}) - r^{(0)}_{\boldsymbol{\ell}^{(1)}}(\boldsymbol{x}_{\ell,i}) \overset{\text{(A.34a)}}{=} 0, \qquad \boldsymbol{\ell} \leq \boldsymbol{\ell}^{(1)},\ \boldsymbol{i} \in I_{\boldsymbol{\ell}}. \tag{A.35}$$

Equation (4.28c) holds as $r^{(0)}_{\boldsymbol{\ell}^{(1)}} = f^{\mathrm{s},(1)}$ (by line 7 in Alg. 4.3) and, therefore,

$$r^{(1)}(\boldsymbol{x}_{\ell,i}) \overset{\text{(A.34b)}}{=} r^{(0)}(\boldsymbol{x}_{\ell,i}) - r^{(0)}_{\boldsymbol{\ell}^{(1)}}(\boldsymbol{x}_{\ell,i}) = f(\boldsymbol{x}_{\ell,i}) - f^{\mathrm{s},(1)}(\boldsymbol{x}_{\ell,i}), \quad \boldsymbol{\ell} \in L,\ \boldsymbol{i} \in I_{\boldsymbol{\ell}}. \tag{A.36}$$

Induction step case: We show the three statements for the induction step $j \to (j+1)$.

- *Showing* (4.28a) *for* $j+1$*:* Let $j' = 1, \dots, j$, $\boldsymbol{\ell} \leq \boldsymbol{\ell}^{(j')}$, and $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$. Due to the ordering of the levels $\boldsymbol{\ell}^{(1)}, \dots, \boldsymbol{\ell}^{(m)}$, we can conclude from $j+1 > j'$ that $\|\boldsymbol{\ell}^{(j+1)}\|_1 \leq \|\boldsymbol{\ell}^{(j')}\|_1$. This implies that there must be a $t' \in \{1, \dots, d\}$ such that $\ell^{(j+1)}_{t'} \leq \ell^{(j')}_{t'}$. Let S be the line in $\mathbb{R}^d$ defined by

  $$S := \boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} + \operatorname{span}\{\boldsymbol{e}_{t'}\}, \tag{A.37}$$

  where $\boldsymbol{e}_{t'}$ is the t'-th standard basis vector. It holds that $S \cap \Omega_{\boldsymbol{\ell}^{(j+1)}} \subseteq \Omega_{\boldsymbol{\ell}^{(j')}}$. To show this, let $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in S \cap \Omega_{\boldsymbol{\ell}^{(j+1)}}$ be arbitrary (with $\boldsymbol{i}' \in I_{\boldsymbol{\ell}'}$). Then, $\forall_{t \neq t'} \ell'_t = \ell_t \leq \ell^{(j')}_t$ (due to $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in S$) and $\ell'_{t'} \leq \ell^{(j+1)}_{t'} \leq \ell^{(j')}_{t'}$ (due to $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in \Omega_{\boldsymbol{\ell}^{(j+1)}}$). This means that $\boldsymbol{\ell}' \leq \boldsymbol{\ell}^{(j')}$, which implies that $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in \Omega_{\boldsymbol{\ell}^{(j')}}$. As $\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ is arbitrary, this shows $S \cap \Omega_{\boldsymbol{\ell}^{(j+1)}} \subseteq \Omega_{\boldsymbol{\ell}^{(j')}}$.

  Thus, we infer

  $$r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}) \overset{\text{(A.34a)}}{=} r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'}) \overset{\text{(4.28b)}}{=} 0, \quad \boldsymbol{x}_{\boldsymbol{\ell}',\boldsymbol{i}'} \in S \cap \Omega_{\boldsymbol{\ell}^{(j+1)}} \subseteq \Omega_{\boldsymbol{\ell}^{(j')}},\ \boldsymbol{i}' \in I_{\boldsymbol{\ell}'}, \tag{A.38}$$

  with the induction hypothesis (4.28b) for j. Unfortunately, this does not suffice to directly conclude that $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0$ as $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ is in general not contained in $\Omega_{\boldsymbol{\ell}^{(j+1)}}$.

  As in the proof of Lemma 4.6, we exploit the tensor product nature of the basis functions and restrict $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}$ to $S \cap [0,1]$:

  $$(r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}|_{S\cap[0,1]})(x_{t'}) = \sum_{\ell'_{t'}=0}^{\ell^{(j+1)}_{t'}} \sum_{i'_{t'} \in I_{\ell'_{t'}}} \tilde{\alpha}^{(j+1)}_{\ell'_{t'},i'_{t'}} \varphi_{\ell'_{t'},i'_{t'}}(x_{t'}), \quad x_{t'} \in [0,1], \tag{A.39a}$$

  $$\tilde{\alpha}^{(j+1)}_{\ell'_{t'},i'_{t'}} := \sum_{\boldsymbol{\ell}'_{-t'}=\boldsymbol{0}}^{\boldsymbol{\ell}^{(j+1)}_{-t'}} \sum_{\boldsymbol{i}'_{-t'} \in I_{\boldsymbol{\ell}'_{-t'}}} \alpha^{(j+1)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi_{\boldsymbol{\ell}'_{-t'},\boldsymbol{i}'_{-t'}}(\boldsymbol{x}_{\boldsymbol{\ell}_{-t'},\boldsymbol{i}_{-t'}}), \tag{A.39b}$$

  where the subscript $-t'$ indicates all entries but the t'-th. As a consequence, this

shows that $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}|_{S\cap[0,1]} \in V_{\ell^{(j+1)}_{t'}}$ is an interpolant of the zero function (by (A.38)). Due to the linear independence of the univariate basis functions, we conclude

$$r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}|_{S\cap[0,1]} \equiv 0. \tag{A.40}$$

Consequently, we obtain $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0$ as $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \in S \cap [\mathbf{0},\mathbf{1}]$.

- *Showing* (4.28b) *for* $j+1$*:* Let $j' = 1,\dots,j+1$, $\boldsymbol{\ell} \le \boldsymbol{\ell}^{(j')}$, and $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$. For the case $j' \le j$, we obtain

$$r^{(j+1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \overset{(A.34b)}{=} r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0 \tag{A.41}$$

due to $r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0$ by induction hypothesis (Eq. (4.28b)) and $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0$ as shown above (Eq. (4.28a) for $j+1$).

For the case $j' = j+1$, Eq. (A.41) still holds as the difference between $r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ and $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ vanishes due to (A.34a) for $j+1$ (here, we need $\boldsymbol{\ell} \le \boldsymbol{\ell}^{(j+1)}$).

- *Showing* (4.28c) *for* $j+1$*:* Let $\boldsymbol{\ell} \in L$ and $\boldsymbol{i} \in I_{\boldsymbol{\ell}}$. Then,

$$r^{(j+1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \overset{(A.34b)}{=} r^{(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - r^{(j)}_{\boldsymbol{\ell}^{(j+1)}}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{A.42a}$$

The first term can replaced with the induction hypothesis ((4.28c) for j). For the second term, note that $r^{(j)}_{\boldsymbol{\ell}^{(j+1)}} = \sum_{\boldsymbol{\ell}'\in L}\sum_{\boldsymbol{i}'\in I_{\boldsymbol{\ell}'}} \alpha^{(j+1)}_{\boldsymbol{\ell}',\boldsymbol{i}'}\varphi_{\boldsymbol{\ell}',\boldsymbol{i}'} = f^{s,(j+1)} - f^{s,(j)}$ by definition. Hence, we obtain as desired

$$\cdots = (f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - f^{s,(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})) - (f^{s,(j+1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - f^{s,(j)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})) \tag{A.42b}$$

$$= f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - f^{s,(j+1)}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{A.42c}$$

This shows the validity of the statements in (4.28) for $j+1$. ∎

A.3.3 Correctness Proof of Hierarchization with Breadth-First Search

PROPOSITION 4.13 (invariant of breadth-first-search hierarchization)
Under the assumption (4.38)*, it holds after* **pop***ping all grid points with level sum* $< q$ *from the queue* Q *in Alg. 4.4:*

$$y_{\boldsymbol{\ell},\boldsymbol{i}} = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\substack{(\boldsymbol{\ell}',\boldsymbol{i}')\in K \\ \|\boldsymbol{\ell}'\|_1 < q}} y_{\boldsymbol{\ell}',\boldsymbol{i}'}\varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad (\boldsymbol{\ell},\boldsymbol{i}) \in K,\ \|\boldsymbol{\ell}\|_1 = q. \tag{4.39}$$

PROOF We start with two observations:

- First, due to the breadth-first search nature of Alg. 4.4 and the hierarchical relation (4.37), all grid points with level sum $< q$ are popped before the first point with level sum $\geq q$ is popped.
- Second, after popping all grid points with level sum $< q$, the output values of the grid points with level sum $\leq q$ remain unchanged for the rest of the algorithm: If line 8 of the algorithm updates the output value of a point $(\boldsymbol{\ell}, \boldsymbol{i})$ in the iteration of $(\boldsymbol{\ell}', \boldsymbol{i}') \in K$ with $\|\boldsymbol{\ell}'\|_1 \geq q$, then line 7 implies $\boldsymbol{\ell} \geq \boldsymbol{\ell}'$ and thus, $\|\boldsymbol{\ell}\|_1 \geq \|\boldsymbol{\ell}'\|_1 \geq q$. However, $\|\boldsymbol{\ell}\|_1 = q$ is not possible as this would imply that $\|\boldsymbol{\ell}\|_1 = \|\boldsymbol{\ell}'\|_1 \implies (\boldsymbol{\ell}, \boldsymbol{i}) = (\boldsymbol{\ell}', \boldsymbol{i}')$ by line 7, but $(\boldsymbol{\ell}, \boldsymbol{i}) = (\boldsymbol{\ell}', \boldsymbol{i}')$ is explicitly excluded in the **for** loop of line 7. Therefore, we must have $\|\boldsymbol{\ell}\|_1 > q$. Hence, if a point $(\boldsymbol{\ell}, \boldsymbol{i})$ with level sum $\geq q$ has been popped, only surpluses of points with level sum $> q$ may be updated.

Now, we prove the asserted claim by induction over q.
Induction base case: For $q = 0$, Alg. 4.4 sets $y_{\boldsymbol{\ell},\boldsymbol{i}}$ to $f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}})$ in line 2. As the sum in (4.39) is empty, the claim is correct for $q = 0$.
Induction step case: Let $y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ and $y^{(q+1)}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ be the surpluses after popping all grid points with level sum $< q$ and $< q+1$, respectively. We show the induction step $q \to (q+1)$, i.e., we assume that the assertion is true for q and prove that after popping all grid points with level sum $< q+1$, it holds

$$y^{(q+1)}_{\boldsymbol{\ell},\boldsymbol{i}} = f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\|\boldsymbol{\ell}'\|_1 < q+1} y^{(q+1)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}), \quad (\boldsymbol{\ell}, \boldsymbol{i}) \in K, \ \|\boldsymbol{\ell}\|_1 = q+1. \tag{A.43}$$

Therefore, let $(\boldsymbol{\ell}, \boldsymbol{i}) \in K$ fulfill $\|\boldsymbol{\ell}\|_1 = q+1$. The update in line 8 can safely be applied with all grid points $(\boldsymbol{\ell}', \boldsymbol{i}')$ with level sum q. The grid points $(\boldsymbol{\ell}', \boldsymbol{i}')$ that do not satisfy the relation in the set in line 7 do not contribute as $\varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) = 0$ due to the necessary condition (4.32). By summing all updates from line 8, we obtain

$$y^{(q+1)}_{\boldsymbol{\ell},\boldsymbol{i}} = y^{(q)}_{\boldsymbol{\ell},\boldsymbol{i}} - \sum_{\|\boldsymbol{\ell}'\|_1 = q} y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{A.44a}$$

After inserting the induction hypothesis for the first $y^{(q)}_{\boldsymbol{\ell},\boldsymbol{i}}$, we have

$$\cdots = \left(f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\|\boldsymbol{\ell}'\|_1 < q} y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \right) - \sum_{\|\boldsymbol{\ell}'\|_1 = q} y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) \tag{A.44b}$$

$$= f(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}) - \sum_{\|\boldsymbol{\ell}'\|_1 < q+1} y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'} \varphi^{\mathrm{f}}_{\boldsymbol{\ell}',\boldsymbol{i}'}(\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}). \tag{A.44c}$$

As noted above, we have $\forall_{(\boldsymbol{\ell}',\boldsymbol{i}'),\|\boldsymbol{\ell}'\|_1<q+1}\ y^{(q)}_{\boldsymbol{\ell}',\boldsymbol{i}'} = y^{(q+1)}_{\boldsymbol{\ell}',\boldsymbol{i}'}$ (the values of points with level sum $< q+1$ do not change after popping all points with level sum $< q$). This shows the induction claim (A.43). ■

A.3.4 Proof for the Correctness of the Unidirectional Principle on Spatially Adaptive Sparse Grids

LEMMA 4.22 (sufficient condition for chain existence)
If $(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$ *for some* $j = 0,\ldots,d$, *then the grid* K *contains the chain from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_j)$.

PROOF We prove the assertion by induction over $j = 0,\ldots,d$.

For $j = 0$, the operator $\mathfrak{L}^{(\emptyset)}$ is by definition (4.68) the identity operator id. The assumption $(\mathfrak{L}^{(\emptyset)})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$ implies that $\boldsymbol{k}' = \boldsymbol{k}''$, since the identity matrix is diagonal. Therefore, the chain $(\boldsymbol{k}^{(0)})$ from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ is given by $\boldsymbol{k}^{(0)} = \boldsymbol{k}'$, which is contained in K.

For the induction step $j \to (j+1)$, we split Eq. (4.68), i.e.,

$$\mathfrak{L}^{(t_1,\ldots,t_{j+1})} = \mathfrak{L}^{(t_{j+1})}\mathfrak{L}^{(t_j)}\ldots\mathfrak{L}^{(t_1)} = \mathfrak{L}^{(t_{j+1})}\mathfrak{L}^{(t_1,\ldots,t_j)}, \tag{A.45}$$

and infer by assumption

$$0 \neq (\mathfrak{L}^{(t_1,\ldots,t_{j+1})})_{\boldsymbol{k}'',\boldsymbol{k}'} = \sum_{\boldsymbol{k}\in K} (\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}'',\boldsymbol{k}} (\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k},\boldsymbol{k}'}. \tag{A.46}$$

Consequently, there is at least one summation index $\boldsymbol{k}$ for which both factors do not vanish. The first factor $(\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}'',\boldsymbol{k}}$ can by definition only be non-zero if $\boldsymbol{k} \sim_{t_{j+1}} \boldsymbol{k}''$. The second factor $(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k},\boldsymbol{k}'}$ being non-zero implies that by induction hypothesis, K contains the chain $(\boldsymbol{k}^{(0)},\ldots,\boldsymbol{k}^{(j)})$ from $\boldsymbol{k}'$ to $\boldsymbol{k}$ with respect to $(t_1,\ldots,t_j)$. The combination of both statements leads to the chain $(\boldsymbol{k}^{(0)},\ldots,\boldsymbol{k}^{(j)},\boldsymbol{k}^{(j+1)})$ from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1,\ldots,t_{j+1})$. All points of the chain are contained in K. ■

LEMMA 4.23 (necessary condition for chain existence)
If the grid K *contains the chain* $(\boldsymbol{k}^{(0)},\ldots,\boldsymbol{k}^{(j)})$ *from* $\boldsymbol{k}'$ *to* $\boldsymbol{k}''$ *with respect to* $(t_1,\ldots,t_j)$ *for some* $j = 0,\ldots,d$, *then*

$$(\mathfrak{L}^{(t_1,\ldots,t_j)})_{\boldsymbol{k}^{(j)},\boldsymbol{k}'} = (\mathfrak{L}^{(t_1),[\boldsymbol{k}^{(1)}]_{\sim_{t_1}}})_{k''_{t_1},k'_{t_1}} \cdots (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim_{t_j}}})_{k''_{t_j},k'_{t_j}}. \tag{4.72}$$

PROOF Again, we prove the claim by induction over $j = 0,\ldots,d$.

For $j = 0$, the operator $\mathfrak{L}^{(\emptyset)}$ is the identity operator. Therefore, the LHS of (4.72) is one (due to $\boldsymbol{k}^{(0)} = \boldsymbol{k}'$). The RHS is by convention also one, as it is an empty product.

For the induction step $j \to (j+1)$, we consider again

$$(\mathfrak{L}^{(t_1,\dots,t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}'} = \sum_{\boldsymbol{k}\in K} (\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}} (\mathfrak{L}^{(t_1,\dots,t_j)})_{\boldsymbol{k},\boldsymbol{k}'} \tag{A.47}$$

similar to (A.46). Recall that

$$k_t^{(j)} = k_t'' \text{ and } k_t^{(j+1)} = k_t'' \quad \text{for } t \in \{t_1,\dots,t_j\}, \tag{A.48a}$$

$$k_t^{(j)} = k_t' \text{ and } k_t^{(j+1)} = k_t'' \quad \text{for } t = t_{j+1}, \tag{A.48b}$$

$$k_t^{(j)} = k_t' \text{ and } k_t^{(j+1)} = k_t' \quad \text{for } t \notin \{t_1,\dots,t_j,t_{j+1}\}. \tag{A.48c}$$

We now argue that all summands of (A.47) vanish, except the summand with index $\boldsymbol{k}^{(j)}$. There are two cases for the summation index $\boldsymbol{k}$, if we assume $\boldsymbol{k} \neq \boldsymbol{k}^{(j)}$:

- If there is a $t \in \{t_1,\dots,t_j\}$ with $k_t \neq k_t''$, then we have $k_t \neq k_t'' = k_t^{(j+1)}$. Consequently, $\boldsymbol{k}^{(j+1)} \not\sim_{t_{j+1}} \boldsymbol{k}$ and $(\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}} = 0$ due to (4.67), i.e., the first factor of the $\boldsymbol{k}$-th summand in (A.47) vanishes.

- If there is a $t \notin \{t_1,\dots,t_j\}$ with $k_t \neq k_t'$, then the second factor $(\mathfrak{L}^{(t_1,\dots,t_j)})_{\boldsymbol{k},\boldsymbol{k}'}$ of the $\boldsymbol{k}$-th summand in (A.47) vanishes. Indeed, if we assume the contrary, then Lemma 4.22 implies that there is a chain from $\boldsymbol{k}'$ to $\boldsymbol{k}$ with respect to $(t_1,\dots,t_j)$. However, by definition of the chain, this means that $\boldsymbol{k}'$ and $\boldsymbol{k}$ coincide in all other dimensions (which are not in $\{t_1,\dots,t_j\}$). This contradicts $k_t \neq k_t'$ and therefore $(\mathfrak{L}^{(t_1,\dots,t_j)})_{\boldsymbol{k},\boldsymbol{k}'}$ must vanish.

We infer that only the summand $\boldsymbol{k} = \boldsymbol{k}^{(j)}$ remains in (A.47):

$$(\mathfrak{L}^{(t_1,\dots,t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}'} = (\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}^{(j)}} (\mathfrak{L}^{(t_1,\dots,t_j)})_{\boldsymbol{k}^{(j)},\boldsymbol{k}'}. \tag{A.49}$$

The first factor equals

$$(\mathfrak{L}^{(t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}^{(j)}} = (\mathfrak{L}^{(t_{j+1}),[\boldsymbol{k}^{(j+1)}]_{\sim t_{j+1}}})_{k_{t_{j+1}}^{(j+1)},k_{t_{j+1}}^{(j)}} = (\mathfrak{L}^{(t_{j+1}),[\boldsymbol{k}^{(j+1)}]_{\sim t_{j+1}}})_{k_{t_{j+1}}'',k_{t_{j+1}}'} \tag{A.50}$$

by Equations (4.67) and (A.48b) (and due to $\boldsymbol{k}^{(j)} \sim_{t_{j+1}} \boldsymbol{k}^{(j+1)}$). The second factor equals

$$(\mathfrak{L}^{(t_1,\dots,t_j)})_{\boldsymbol{k}^{(j)},\boldsymbol{k}'} = (\mathfrak{L}^{(t_1),[\boldsymbol{k}^{(1)}]_{\sim t_1}})_{k_{t_1}'',k_{t_1}'} \cdots (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim t_j}})_{k_{t_j}'',k_{t_j}'} \tag{A.51}$$

by induction hypothesis. Hence, the product of both factors is

$$(\mathfrak{L}^{(t_1,\dots,t_{j+1})})_{\boldsymbol{k}^{(j+1)},\boldsymbol{k}'} = (\mathfrak{L}^{(t_1),[\boldsymbol{k}^{(1)}]_{\sim t_1}})_{k_{t_1}'',k_{t_1}'} \cdots (\mathfrak{L}^{(t_{j+1}),[\boldsymbol{k}^{(j+1)}]_{\sim t_{j+1}}})_{k_{t_{j+1}}'',k_{t_{j+1}}'}. \tag{A.52}$$

■

PROPOSITION 4.24 (characterization of the correctness of the UP)
Let $\mathfrak{L}$ have tensor product structure: For all $\boldsymbol{k}', \boldsymbol{k}'' \in K$ with the chain $(\boldsymbol{k}^{(0)}, \dots, \boldsymbol{k}^{(d)})$ from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$, we assume that

$$(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} = \prod_{j=1}^{d} (\mathfrak{L}^{(t_j),[\boldsymbol{k}^{(j)}]_{\sim t_j}})_{k''_{t_j},k'_{t_j}}. \tag{4.73}$$

Then the unidirectional principle (UP) is correct for $\mathfrak{L}$ and $(t_1, \dots, t_d)$ if and only if the grid K contains the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$ for all $\boldsymbol{k}', \boldsymbol{k}'' \in K$ for which $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$.

PROOF "$\Longrightarrow$": Let the UP be correct for $\mathfrak{L}$ and $(t_1, \dots, t_d)$ and $\boldsymbol{k}', \boldsymbol{k}'' \in K$ with $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$. Then, we obtain

$$(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'} = (\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0. \tag{A.53}$$

By Lemma 4.22, this implies that K contains the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$.

"$\Longleftarrow$": For the converse direction, we assume that there are chains from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$ for all $\boldsymbol{k}', \boldsymbol{k}'' \in K$ with $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$. Let $\boldsymbol{k}', \boldsymbol{k}'' \in K$ be arbitrary. There are two cases:

- $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$: By assumption, K contains the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$. We apply Lemma 4.23 with $j = d$ to infer

  $$(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'} = (\mathfrak{L}^{(t_1),[\boldsymbol{k}^{(1)}]_{\sim t_1}})_{k''_{t_1},k'_{t_1}} \cdots (\mathfrak{L}^{(t_d),[\boldsymbol{k}^{(d)}]_{\sim t_d}})_{k''_{t_d},k'_{t_d}} = (\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} \tag{A.54}$$

  by the assumption (4.73) on the tensor product structure of $\mathfrak{L}$.

- $(\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} = 0$: In this case, $(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'}$ must vanish as well. Indeed, if we assume the contrary $(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'} \neq 0$, then we can apply Lemma 4.22 to obtain that K contains the chain from $\boldsymbol{k}'$ to $\boldsymbol{k}''$ with respect to $(t_1, \dots, t_d)$. We conclude with Lemma 4.23 as in the first case that $(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'} = (\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'} = 0$, which is a contradiction.

In any case, we obtain $(\mathfrak{L}^{(t_1,\dots,t_d)})_{\boldsymbol{k}'',\boldsymbol{k}'} = (\mathfrak{L})_{\boldsymbol{k}'',\boldsymbol{k}'}$, from which follows the correctness of the UP, as $\boldsymbol{k}'$ and $\boldsymbol{k}''$ are arbitrary. ■

A.3.5 Correctness Proof of Hermite Hierarchization

PROPOSITION 4.27 (invariant of Hermite hierarchization)
In Alg. 4.6, it holds for $\ell = 0,\dots,n$ *and* $i = 0,\dots,2^\ell$

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_\ell(x_{\ell,i}) = \sum_{\ell'=0}^{\ell} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{\ell',i'}(x_{\ell,i}), \quad q = 0,\dots,\frac{p-1}{2}. \tag{4.84}$$

PROOF We prove the assertion by induction over $\ell = 0,\dots,n$.

For the induction base case $\ell = 0$ and $i \in \{0,1\}$, we have

$$\sum_{i'=0}^{1} y_{0,i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{0,i'}(x_{0,i}) = \delta_{q,0} \cdot f(x_{0,i}) + \delta_{q,1} \cdot (f(x_{0,1}) - f(x_{0,0})) = \frac{\mathrm{d}^q}{\mathrm{d}x^q} f_0(x_{0,i}) \tag{A.55}$$

for $q = 0,\dots,\frac{p-1}{2}$ by lines 3 and 4 of Alg. 4.6.

For the induction step case $(\ell-1) \to \ell$, it suffices to show that

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i}) \overset{!}{=} \sum_{\ell'=0}^{\ell-1} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{\ell',i'}(x_{\ell,i}), \quad i = 0,\dots,2^\ell, \ q = 0,\dots,\frac{p-1}{2}. \tag{A.56}$$

Indeed, if (A.56) holds, then we obtain by lines 9 and 13 of Alg. 4.6

$$\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_\ell(x_{\ell,i}) = \frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i}) + \frac{\mathrm{d}^q}{\mathrm{d}x^q} r^{(\ell)}_\ell(x_{\ell,i}) \tag{A.57a}$$

$$= \sum_{\ell'=0}^{\ell-1} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{\ell',i'}(x_{\ell,i}) + \sum_{i' \in I_\ell} y_{\ell,i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{\ell,i'}(x_{\ell,i}) \tag{A.57b}$$

$$= \sum_{\ell'=0}^{\ell} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \frac{\mathrm{d}^q}{\mathrm{d}x^q} \varphi^{p,\mathrm{wfs}}_{\ell',i'}(x_{\ell,i}), \tag{A.57c}$$

which is the desired relation (4.84) for level ℓ.

To prove (A.56), we separate two cases:

- $i \notin I_\ell$: In this case, the "true" level of $x_{\ell,i}$ is actually $\le \ell-1$. Therefore, we can apply the induction hypothesis for Eq. (4.84) to obtain (A.56).
- $i \in I_\ell$: In this case, we cannot directly apply the induction hypothesis, as it only holds for grid points of levels $\le \ell-1$. However, we note that in (A.56), the term $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i})$ is the q-th derivative of the Hermite interpolant of the data $\frac{\mathrm{d}^{q'}}{\mathrm{d}x^{q'}} f_{\ell-1}(x_{\ell,i\pm1})$ $(q' = 0,\dots,\frac{p-1}{2})$, as determined in line 7 of Alg. 4.6. The "true" level of the grid points $x_{\ell,i\pm1}$ is actually $\le \ell-1$ due to $i \in I_\ell$. Hence, we can apply the induction hypothesis

for Eq. (4.84) to conclude that the interpolated data of $\frac{\mathrm{d}^q}{\mathrm{d}x^q} f_{\ell-1}(x_{\ell,i})$ are given by

$$\text{(A.58)} \qquad \frac{\mathrm{d}^{q'}}{\mathrm{d}x^{q'}} f_{\ell-1}(x_{\ell,i\pm1}) = \frac{\mathrm{d}^{q'}}{\mathrm{d}x^{q'}} \left[\sum_{\ell'=0}^{\ell-1} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \varphi_{\ell',i'}^{p,\mathrm{wfs}} \right] (x_{\ell,i\pm1}), \quad q' = 0, \dots, \frac{p-1}{2}.$$

The linear combination in square brackets is a polynomial of degree $\leq p$ on the interval $[x_{\ell,i-1}, x_{\ell,i+1}]$ by construction of the hierarchical basis functions $\varphi_{\ell',i'}^{p,\mathrm{wfs}}$ ($\ell' = 0, \dots, \ell-1$, $i' \in I_{\ell'}$). Due to the uniqueness of Hermite interpolation (Lemma 4.26), the interpolation polynomial of the data must coincide on $[x_{\ell,i-1}, x_{\ell,i+1}]$ with the term in square brackets. In particular, as $x_{\ell,i} \in [x_{\ell,i-1}, x_{\ell,i+1}]$, we obtain the claim (A.56).

In both cases, we obtain the desired relation (A.56). ■

COROLLARY 4.28 *Algorithm 4.6 is correct.*

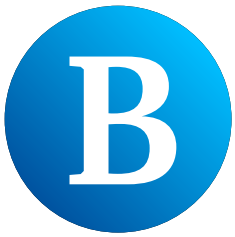
PROOF By Prop. 4.27 ($q = 0$), we have

$$\text{(A.59)} \qquad \sum_{\ell'=0}^{n} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \varphi_{\ell',i'}^{p,\mathrm{wfs}}(x_{\ell,i}) = \sum_{\ell'=0}^{\ell} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \varphi_{\ell',i'}^{p,\mathrm{wfs}}(x_{\ell,i}) = f_\ell(x_{\ell,i}), \quad \ell \leq n,\ i \in I_\ell,$$

as $\varphi_{\ell',i'}^{p,\mathrm{wfs}}(x_{\ell,i}) = 0$ if $\ell' > \ell$ (weakly fundamental property (4.77)). Line 13 of Alg. 4.6 implies $f_\ell(x_{\ell,i}) = f_{\ell-1}(x_{\ell,i}) + r_\ell^{(\ell)}(x_{\ell,i})$, and by lines 8 and 10, the second summand $r_\ell^{(\ell)}(x_{\ell,i})$ equals $f(x_{\ell,i}) - f_{\ell-1}(x_{\ell,i})$, which cancels out the first summand, resulting in $f_\ell(x_{\ell,i}) = f(x_{\ell,i})$. Combining these statements, we obtain

$$\text{(A.60)} \qquad f^{\mathrm{s}}(x_{\ell,i}) = f(x_{\ell,i}), \quad \ell \leq n,\ i \in I_\ell, \quad \text{where} \quad f^{\mathrm{s}} := \sum_{\ell'=0}^{n} \sum_{i' \in I_{\ell'}} y_{\ell',i'} \varphi_{\ell',i'}^{p,\mathrm{wfs}}.$$

This means that f^{s} is the correct hierarchical interpolant of the given function values (see Eq. (4.2)). Due to the uniqueness of hierarchical surpluses, the coefficients $y_{\ell,i}$ (which are the output of Alg. 4.6) must coincide with the surpluses $\alpha_{\ell,i}$. ■



Test Problems for Optimization

In the following, we give formal definitions of the test problems mentioned in Sec. 5.3. For each problem, we state the objective function $\bar{f} : [\boldsymbol{a}, \boldsymbol{b}] \to \mathbb{R}$, $\bar{\boldsymbol{x}} \mapsto \bar{f}(\bar{\boldsymbol{x}})$, its domain $[\boldsymbol{a}, \boldsymbol{b}]$ (where $\bar{\boldsymbol{x}} \in [\boldsymbol{a}, \boldsymbol{b}]$), the location $\bar{\boldsymbol{x}}^{\text{opt}} \in [\boldsymbol{a}, \boldsymbol{b}]$ of its global minimum, and its minimal value $\bar{f}(\bar{\boldsymbol{x}}^{\text{opt}})$ (and the constraint function $\bar{\boldsymbol{g}} : [\boldsymbol{a}, \boldsymbol{b}] \to \mathbb{R}^{m_{\bar{g}}}$, if any). Plots of the test problems are given in Figures 5.3 and 5.4.

B.1 Unconstrained Problems

B.1.1 Bivariate Unconstrained Problems

Branin02. The function originates from [Mun98]. Compared to [Mun98], we changed the domain from $[-5, 10] \times [0, 15]$ to $[-5, 15]^2$, which seems more common in recent literature [Gav13]. In addition, [Mun98] uses the reciprocal function value, while searching for the maximum instead of the minimum.

$$\bar{f}_{\text{Bra02}}(\bar{\boldsymbol{x}}) := \left(-\frac{51\bar{x}_1^2}{40\pi^2} + \frac{5\bar{x}_1}{\pi} + \bar{x}_2 - 6\right)^2 + \left(10 - \frac{5}{4\pi}\right)\cos(\bar{x}_1)\cos(\bar{x}_2) + \ln(\bar{x}_1^2 + \bar{x}_2^2 + 1) + 10, \tag{B.1a}$$

$$\bar{\boldsymbol{x}} \in [-5, 15]^2, \quad \bar{\boldsymbol{x}}^{\text{opt}} = (-3.196988424804, 12.52625788532), \tag{B.1b}$$

$$\bar{f}_{\text{Bra02}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 5.558914403894 \tag{B.1c}$$

GoldsteinPrice. This function originates from [Gol71], where the function was stated without bounds for the optimization domain. We took the domain $[-2,2]^2$ from [Gav13]. In addition, we scaled the function values by the factor 10^{-4} for the sake of plotting.

$$\bar{f}_{\text{GoP}}(\bar{\boldsymbol{x}}) := 10^{-4} \cdot \big(1 + (\bar{x}_1 + \bar{x}_2 + 1)^2 (19 - 14\bar{x}_1 + 3\bar{x}_1^2 - 14\bar{x}_2 + 6\bar{x}_1\bar{x}_2 + 3\bar{x}_2^2)\big) \cdot \big(30 + (2\bar{x}_1 - 3\bar{x}_2)^2 (18 - 32\bar{x}_1 + 12\bar{x}_1^2 + 48\bar{x}_2 - 36\bar{x}_1\bar{x}_2 + 27\bar{x}_2^2)\big), \tag{B.2a}$$

$$\bar{\boldsymbol{x}} \in [-2,2]^2, \quad \bar{\boldsymbol{x}}^{\text{opt}} = (0,-1), \tag{B.2b}$$

$$\bar{f}_{\text{GoP}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 3 \cdot 10^{-4} \tag{B.2c}$$

Schwefel06. This function originates from [Schw77]. We changed the domain from $[-3,5] \times [-1,7]$ to $[-6,4]^2$, such that the optimum point is not located at the center of the optimization domain.

$$\bar{f}_{\text{Sch06}}(\bar{\boldsymbol{x}}) := \max(|\bar{x}_1 + 2\bar{x}_2 - 7|, |2\bar{x}_1 + \bar{x}_2 - 5|), \tag{B.3a}$$

$$\bar{\boldsymbol{x}} \in [-6,4]^2, \quad \bar{\boldsymbol{x}}^{\text{opt}} = (1,3), \quad \bar{f}_{\text{Sch06}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 0 \tag{B.3b}$$

B.1.2 d-Variate Unconstrained Problems

Ackley. The form of this function originates from [Ack87], where it was stated only for two variables. We use the generalization to d variables from [Gav13]. The optimization domain $[1.5, 6.5]^d$ was chosen such that it does not contain $\mathbf{0}$, where the gradient of the objective function becomes singular. Otherwise, the function would not be continuously differentiable, which would be a disadvantage for spline-based approaches (see Schwefel06 and Schwefel22 for functions with discontinuous derivatives).

$$\bar{f}_{\text{Ack}}(\bar{\boldsymbol{x}}) := -20 \exp\left(-\frac{\|\bar{\boldsymbol{x}}\|_2}{5\sqrt{d}}\right) - \exp\left(\frac{1}{d}\sum_{t=1}^{d} \cos(2\pi\bar{x}_t)\right) + 20 + \mathrm{e}, \tag{B.4a}$$

$$\bar{\boldsymbol{x}} \in [1.5, 6.5]^d, \quad \bar{\boldsymbol{x}}^{\text{opt}} = 1.974451986484 \cdot \mathbf{1}, \quad \bar{f}_{\text{Ack}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 6.559645375628 \tag{B.4b}$$

Alpine02. This function originates from [Cle99]. We changed the domain from $[0,10]^d$ to $[2,10]^d$ to exclude the singularities of the derivative of the objective function at $\bar{x}_t = 0$. In addition, the author of [Cle99] searched for maximal points. For minimization, we changed the sign of the objective function.

$$\bar{f}_{\text{Alp02}}(\bar{\boldsymbol{x}}) := -\prod_{t=1}^{d} \sqrt{\bar{x}_t} \sin(\bar{x}_t), \qquad \bar{\boldsymbol{x}} \in [2,10]^d, \tag{B.5a}$$

$$\bar{\boldsymbol{x}}^{\text{opt}} = 7.917052684666 \cdot \mathbf{1}, \quad \bar{f}_{\text{Alp02}}(\bar{\boldsymbol{x}}^{\text{opt}}) = -2.808131180070^d \tag{B.5b}$$

Schwefel22. This function originates from [Schw77]. We changed the domain from $[-10,10]^d$ to $[-3,7]^d$, such that the optimum point is not located at the center of the optimization domain.

$$\bar{f}_{\text{Sch22}}(\bar{\boldsymbol{x}}) := \sum_{t=1}^{d} |\bar{x}_t| + \prod_{t=1}^{d} |\bar{x}_t|, \qquad \bar{\boldsymbol{x}} \in [-3,7]^d, \tag{B.6a}$$

$$\bar{\boldsymbol{x}}^{\text{opt}} = \mathbf{0}, \quad \bar{f}_{\text{Sch22}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 0 \tag{B.6b}$$

B.2 Constrained Problems

G08. This problem originates from [Schoena93]. We changed the domain from $[0,10]^2$ to $[0.5,2.5]\times[3,6]$ to increase the size of feasible region. In addition, we use different frequencies for the sine terms as in [Gav13].

$$\bar{f}_{\text{G08}}(\bar{\boldsymbol{x}}) := -\frac{\sin^3(2\pi\bar{x}_1)\sin(2\pi\bar{x}_2)}{\bar{x}_1^3(\bar{x}_1+\bar{x}_2)}, \quad \bar{\boldsymbol{g}}_{\text{G08}}(\bar{\boldsymbol{x}}) := \begin{pmatrix} \bar{x}_1^2 - \bar{x}_2 + 1 \\ 1 - \bar{x}_1 + (\bar{x}_2-4)^2 \end{pmatrix}, \tag{B.7a}$$

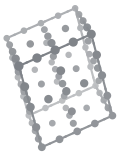
$$\bar{\boldsymbol{x}} \in [0.5,2.5]\times[3,6], \quad \bar{\boldsymbol{x}}^{\text{opt}} = (1.227971358337, 4.245373366474), \tag{B.7b}$$

$$\bar{f}_{\text{G08}}(\bar{\boldsymbol{x}}^{\text{opt}}) = -0.09582504141804 \tag{B.7c}$$

G04Squared. This problem is based on a problem from [Col68] with the objective function $\bar{f}_{\text{G04}}(\bar{\boldsymbol{x}}) := 5.3578547\bar{x}_3^2 + 0.8356891\bar{x}_1\bar{x}_5 + 37.293239\bar{x}_1 - 40792.141$ and the same constraints $\bar{\boldsymbol{g}}_{\text{G04}}(\bar{\boldsymbol{x}}) := \bar{\boldsymbol{g}}_{\text{G04Sq}}(\bar{\boldsymbol{x}})$. However, hierarchical cubic not-a-knot B-splines are able to exactly represent the polynomial $\bar{f}_{\text{G04}}$ of coordinate degree two on the whole domain $[\mathbf{0},\mathbf{1}]$, if the level of the sparse grids is high enough, see Corollary 3.11 (sparse grid with not-a-knot B-splines contains polynomials). Therefore, we modified the original G04 problem by squaring the objective function. To ensure that this does not change the location of the global minimum, we added a constant before squaring such that the shifted function is non-negative on $[\mathbf{0},\mathbf{1}]$.

$$\bar{f}_{\text{G04Sq}}(\bar{\boldsymbol{x}}) := (5.3578547\bar{x}_3^2 + 0.8356891\bar{x}_1\bar{x}_5 + 37.293239\bar{x}_1 - 10120)^2, \tag{B.8a}$$

$$\bar{\boldsymbol{g}}_{\text{G04Sq}}(\bar{\boldsymbol{x}}) := 10^{-3}\begin{pmatrix} 85334.407 + 5.6858\bar{x}_2\bar{x}_5 + 0.6262\bar{x}_1\bar{x}_4 - 2.2053\bar{x}_3\bar{x}_5 - 92000 \\ -85334.407 - 5.6858\bar{x}_2\bar{x}_5 - 0.6262\bar{x}_1\bar{x}_4 + 2.2053\bar{x}_3\bar{x}_5 \\ 80512.49 + 7.1317\bar{x}_2\bar{x}_5 + 2.9955\bar{x}_1\bar{x}_2 + 2.1813\bar{x}_3^2 - 110000 \\ -80512.49 - 7.1317\bar{x}_2\bar{x}_5 - 2.9955\bar{x}_1\bar{x}_2 - 2.1813\bar{x}_3^2 + 90000 \\ 9300.961 + 4.7026\bar{x}_3\bar{x}_5 + 1.2547\bar{x}_1\bar{x}_3 + 1.9085\bar{x}_3\bar{x}_4 - 25000 \\ -9300.961 - 4.7026\bar{x}_3\bar{x}_5 - 1.2547\bar{x}_1\bar{x}_3 - 1.9085\bar{x}_3\bar{x}_4 + 20000 \end{pmatrix}, \tag{B.8b}$$



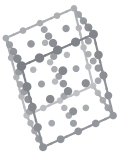
$$\bar{\boldsymbol{x}} \in [78, 102] \times [33, 45] \times [27, 45]^3, \tag{B.8c}$$

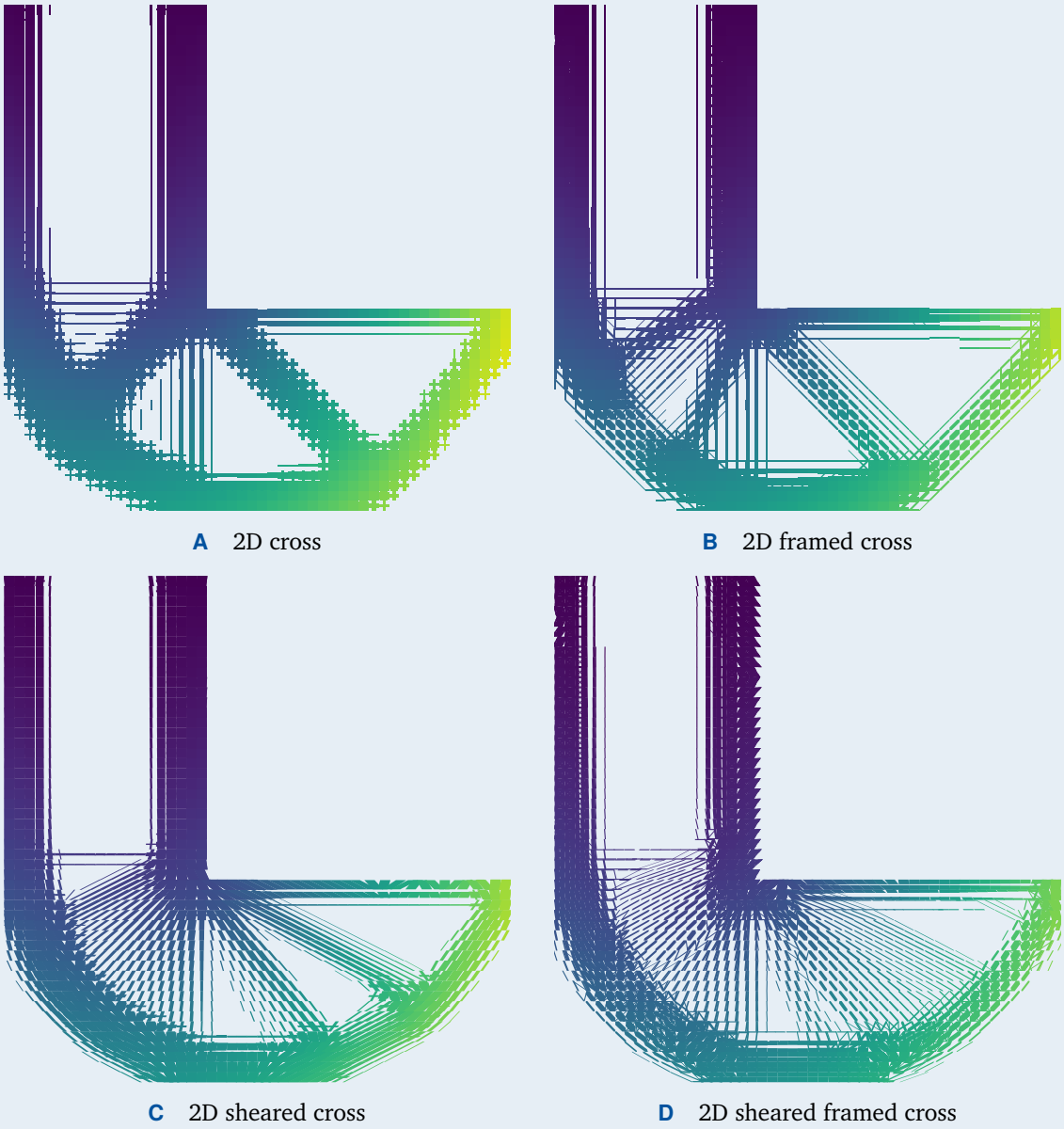
$$\bar{\boldsymbol{x}}^{\text{opt}} = (78, 33, 29.995256025682, 45, 36.775812905788), \tag{B.8d}$$

$$\bar{f}_{\text{G04Sq}}(\bar{\boldsymbol{x}}^{\text{opt}}) = 43.590737882363 \tag{B.8e}$$

Detailed Results for Topology Optimization

This appendix complements Sec. 6.4. It contains visualizations of the topologically optimal structures in the 2D L-shape and the 3D scenarios in Fig. C.1 and Fig. C.2, respectively. In addition, we report details of the corresponding optimization runs in Tab. C.1 and information about the employed spatially adaptive sparse grids in Tab. 6.2. The computation times were measured on a shared-memory computer with 4x Intel Xeon E7-8880v3 (72 cores, 144 threads).





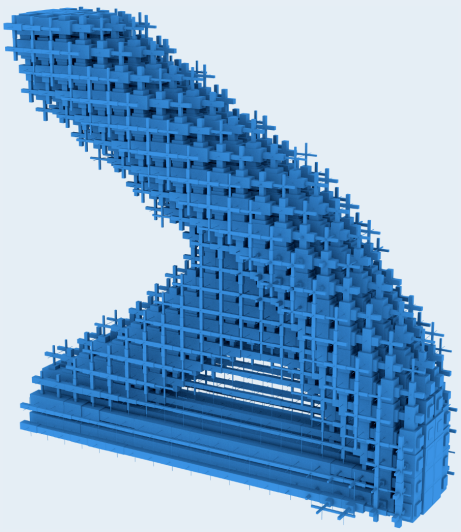
A 2D cross

B 2D framed cross

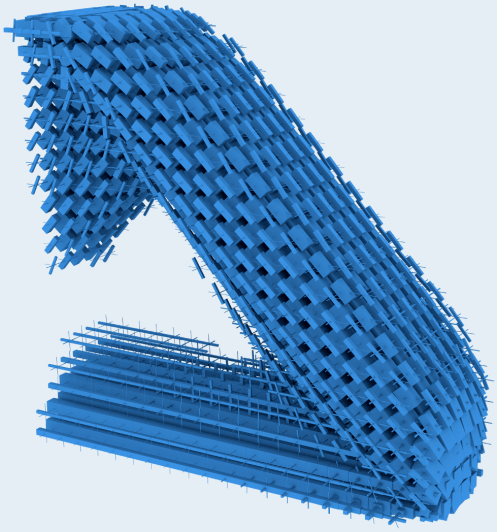
C 2D sheared cross

D 2D sheared framed cross

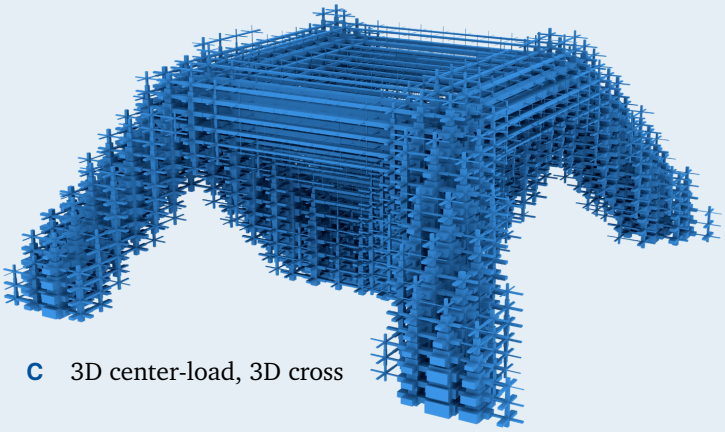
FIGURE C.1 Topologically optimal structures in the 2D L-shape scenario for different micro-cell models using cubic B-splines (spatially adaptive grids with around 10 000 points). The colors indicate the length of the displacement, where dark regions correspond to weak displacements and bright regions to strong displacements. The color map is the same as in Fig. 6.9. Only bars with widths ≥ 0.1 are shown. More details can be found in Tab. C.1.



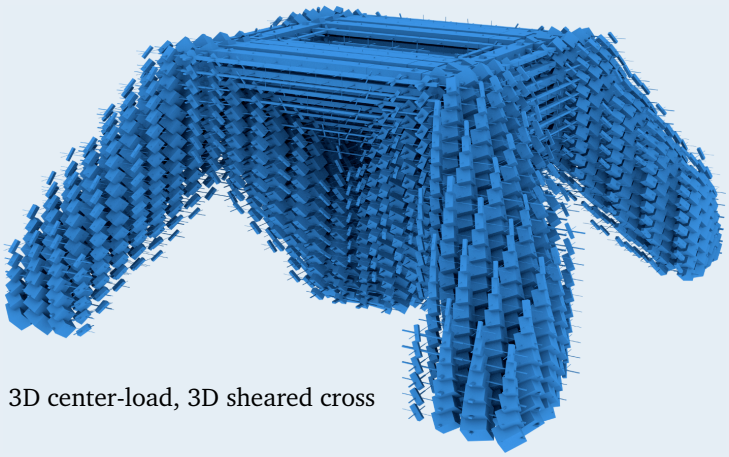
A 3D cantilever, 3D cross



B 3D cantilever, 3D sheared cross

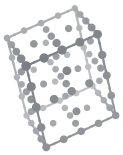


C 3D center-load, 3D cross



D 3D center-load, 3D sheared cross

FIGURE C.2 Topologically optimal structures in the 3D cantilever and center-load scenarios for different micro-cell models using cubic B-splines (spatially adaptive grids with around 10 000 points). More details can be found in Tab. C.1.



Scenario	Model	#Iter.	$J^{\text{opt},*}$	$J^{\text{s,opt},*}$	O.-i. gap	Time
2D cantilever	2D-C	547	74.974	74.974	$3.67 \cdot 10^{-5}$	5 min
	2D-FC	249	70.816	69.409	$1.41 \cdot 10^{0}$	14 min
	2D-SC	2196	67.809	67.804	$5.21 \cdot 10^{-3}$	1 h 06 min
	2D-SFC	749	68.602	65.201	$3.40 \cdot 10^{0}$	15 min
2D L-shape	2D-C	289	183.68	183.68	$9.06 \cdot 10^{-5}$	3 min
	2D-FC	602	177.51	174.49	$3.02 \cdot 10^{0}$	17 min
	2D-SC	1609	169.60	169.60	$7.33 \cdot 10^{-3}$	33 min
	2D-SFC	574	174.55	158.19	$1.64 \cdot 10^{1}$	8 min
3D cantilever	3D-C	39	247.60	247.49	$1.13 \cdot 10^{-1}$	10 min
	3D-SC	608	162.59	159.33	$3.25 \cdot 10^{0}$	3 h 17 min
3D center-load	3D-C	35	169.27	169.27	$3.31 \cdot 10^{-3}$	4 min
	3D-SC	1026	46.171	45.571	$6.00 \cdot 10^{-1}$	2 h 25 min

TABLE C.1 Detailed information about the optimization runs corresponding to Tab. 6.2 and Figures 6.9, C.1, and C.2, which employs cubic B-splines ($p = 3$) on the spatially adaptive grids listed in Tab. 6.2. From left to right, the columns contain the optimization scenario, the micro-cell model, the number of optimization iterations, the actual compliance value $J^{\text{opt},*} := J(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)})$, the approximated compliance value $J^{\text{s,opt},*} := J^{\text{s}}(\boldsymbol{x}^{\text{opt},*,(1)}, \dots, \boldsymbol{x}^{\text{opt},*,(M)})$ as reported by the optimizer, the optimality-interpolation gaps $|J^{\text{opt},*} - J^{\text{s,opt},*}|$, and the computation time of the online phase (without the time to generate the sparse grid data).

Model	d	N	Threshold	Rel. err.	Eval. time
2D-C	2	10 197	$2.15 \cdot 10^{-5}$	$2.24 \cdot 10^{-5}$	6.96 s
2D-FC	4	10 502	$7.94 \cdot 10^{-1}$	$1.81 \cdot 10^{-2}$	7.45 s
2D-SC	3	10 723	$4.64 \cdot 10^{-3}$	$1.49 \cdot 10^{-3}$	8.72 s
2D-SFC	5	10 694	$5.01 \cdot 10^{0}$	$4.82 \cdot 10^{-2}$	7.45 s
3D-C	3	9207	$7.94 \cdot 10^{-2}$	$3.18 \cdot 10^{-3}$	33.5 s
3D-SC	5	15 389	$5.01 \cdot 10^{0}$	$4.95 \cdot 10^{-2}$	40.0 s

TABLE C.2 Detailed information about the spatially adaptive sparse grids used for Tables 6.2 and C.1 and Figures 6.9, C.1, and C.2. The columns correspond to the micro-cell model, the number d of micro-cell parameters, the number N of sparse grid points, the threshold κ used in the grid generation algorithm, the relative L^2 spectral interpolation error $\left\| \|\boldsymbol{E}(\cdot) - \boldsymbol{E}^{\text{chol,s}}(\cdot)\|_2 \right\|_{L^2} / \left\| \|\boldsymbol{E}(\cdot)\|_2 \right\|_{L^2}$, and the time needed to evaluate the elasticity tensor $\boldsymbol{E}(\boldsymbol{x}_k)$ at a single grid point $\boldsymbol{x}_k$.

Bibliography

[Ack87] **Ackley**, D. H.: *A Connectionist Machine for Genetic Hillclimbing*, Kluwer Academic Publishers, 1987, isbn:978-0-89838-236-5

[All04] **Allaire**, G.: *Topology optimization with the homogenization and the level-set methods*, Nonlinear Homogenization and its Applications to Composites, Polycrystals and Smart Materials, ed. by **Ponte Castaneda**, P.; **Telega**, J. J.; **Gambin**, B., NATO Science Series II: Mathematics, Physics and Chemistry 170, Springer, 2004, pp. 1–13, doi:10.1007/1-4020-2623-4_1

[All16] **Allaire**, G.; **Jouve**, F.: *Towards efficient and reliable topology optimization of structures*, ECCOMAS Newsletter – June 2016, ed. by **Ramm**, E. et al., 2016, pp. 6–9, https://web.archive.org/web/20181004085628/http://www.eccomas.org/cvdata/cntr1/spc22/dtos/img/mdia/ECCOMAS-NL-2016-(1).pdf

[An84] **An**, K. N. et al.: *Determination of muscle orientations and moment arms*, Journal of Biomechanical Engineering 106.3, 1984, pp. 280–282, doi:10.1115/1.3138494

[Ani00] **Anile**, A. M. et al.: *Modeling uncertain data with fuzzy B-splines*, Fuzzy Sets and Systems 113.3, 2000, pp. 397–410, doi:10.1016/S0165-0114(98)00146-8

[Baa15] **Baar**, J. H. S. de; **Harding**, B.: *A gradient-enhanced sparse grid algorithm for uncertainty quantification*, International Journal for Uncertainty Quantification 5.5, 2015, pp. 453–468, doi:10.1615/Int.J.UncertaintyQuantification.2015014394

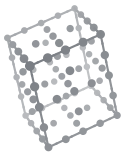
[Bac01] **Bachau**, H. et al.: *Applications of B-splines in atomic and molecular physics*, Reports on Progress in Physics 64.12, 2001, pp. 1815–1942, doi:10.1088/0034-4885/64/12/205

[Bal94] **Balder**, R.: *Adaptive Verfahren für elliptische und parabolische Differentialgleichungen auf dünnen Gittern*, PhD thesis, Technical University of Munich, Institute of Computer Science, 1994

[Bel57] **Bellman**, R.: *Dynamic Programming*, Princeton University Press, 1957, isbn:978-0-691-07951-6

[Bel61] **Bellman**, R.: *Adaptive Control Processes: A Guided Tour*, Princeton University Press, 1961, isbn:978-1-4008-7466-8

[Ben24] **Benoît**, E.: *Note sur une méthode de résolution des équations normales provenant de l'application de la méthode des moindres carrés à un système d'équations linéaires en nombre inférieur à celui des inconnues, Application de la méthode a la résolution d'un systeme défini d'équations linéaires*, Bulletin Géodésique 2, 1924, pp. 67–77, doi:10.1007/BF03031308



[Boh18] **Bohn**, B.; **Griebel**, M.; **Oettershagen**, J.: *Optimally Rotated Coordinate Systems for Adaptive Least-Squares Regression on Sparse Grids*, 2018, https://arxiv.org/abs/1810.06749v1

[Bón15] **Bóna**, M.: *Introduction to Enumerative and Analytic Combinatorics*, 2nd ed., CRC Press, 2015, isbn:978-1-4822-4909-5

[Boor16] **Boor**, C. de: *A comment on Ewald Quak's "About B-splines"*, Journal of Numerical Analysis and Approximation Theory 45.1, 2016, pp. 84–86

[Boor72] **Boor**, C. de: *On calculating with B-splines*, Journal of Approximation Theory 6.1, 1972, pp. 50–62, doi:10.1016/0021-9045(72)90080-9

[Boor76] **Boor**, C. de: *Splines as linear combinations of B-splines. A survey*, Approximation Theory II, ed. by **Lorentz**, G. G.; **Chui**, C. K.; **Schumaker**, L. L., Academic Press, 1976

[Boos85] **Boos**, D. D.: *A converse to Scheffé's theorem*, The Annals of Statistics 13.1, 1985, pp. 423–427, doi:10.1214/aos/1176346604

[Boy04] **Boyd**, S.; **Vandenberghe**, L.: *Convex Optimization*, Cambridge University Press, 2004, isbn:978-0-521-83378-3

[Bru17] **Brumm**, J.; **Scheidegger**, S.: *Using adaptive sparse grids to solve high-dimensional dynamic models*, Econometrica 85.5, 2017, pp. 1575–1612, doi:10.3982/ECTA12216

[Buc90] **Buckley**, J. J.; **Qu**, Y.: *On using α-cuts to evaluate fuzzy equations*, Fuzzy Sets and Systems 38.3, 1990, pp. 309–312, doi:10.1016/0165-0114(90)90204-J

[Bun04] **Bungartz**, H.-J.; **Griebel**, M.: *Sparse grids*, Acta Numerica 13, 2004, pp. 147–269, doi:10.1017/S0962492904000182

[Bun14] **Bungartz**, H.-J. et al.: *Modeling and Simulation, An Application-Oriented Introduction*, Springer Undergraduate Texts in Mathematics and Technology, Springer, 2014, isbn:978-3-642-39523-9

[Bun92] **Bungartz**, H.-J.: *Dünne Gitter und deren Anwendung bei der adaptiven Lösung der dreidimensionalen Poisson-Gleichung*, PhD thesis, Technical University of Munich, Institute of Computer Science, 1992

[Bun98] **Bungartz**, H.-J.: *Finite Elements of Higher Order on Sparse Grids*, Habilitation thesis, Technical University of Munich, Institute of Computer Science, 1998

[Cai10] **Cai**, Y.; **Judd**, K. L.: *Stable and efficient computational methods for dynamic programming*, Journal of the European Economic Association 8.2–3, 2010, pp. 626–634, doi:10.1111/j.1542-4774.2010.tb00532.x

[Chu92] **Chui**, C. K.: *An Introduction to Wavelets*, Wavelet Analysis and Its Applications 1, Academic Press, 1992, isbn:978-0-12-174584-4

[Cle99] **Clerc**, M.: *The swarm and the queen: Towards a deterministic and adaptive particle swarm optimization*, Proceedings of the 1999 Congress on Evolutionary Computation – CEC99, IEEE, 1999, pp. 1951–1957, doi:10.1109/CEC.1999.785513

[Coh01] **Cohen**, E.; **Riesenfeld**, R. F.; **Elber**, G.: *Geometric Modeling with Splines: An Introduction*, A K Peters, 2001, isbn:978-1-56881-137-6

[Col68] **Colville**, A. R.: *A comparative study on nonlinear programming codes*, technical report 320-2949, IBM New York Scientific Center, 1968

[Cot09] **Cottrell**, J. A.; **Hughes**, T. J. R.; **Bazilevs**, Y.: *Isogeometric Analysis, Toward Integration of CAD and FEA*, John Wiley & Sons, 2009, isbn:978-0-470-74873-2

[Cox72] **Cox**, M. G.: *The numerical evaluation of B-splines*, IMA Journal of Applied Mathematics 10.2, 1972, pp. 134–149, doi:10.1093/imamat/10.2.134

[Delb14] **Delbos**, F.; **Dumas**, L.; **Echagüe**, E.: *Global Optimization Based on Sparse Grid Surrogate Models for Black-Box Expensive Functions*, 2014, https://web.archive.org/web/20181121161555/http://dumas.perso.math.cnrs.fr/JOGO.pdf

[Delv82] **Delvos**, F.-J.: *d-variate Boolean interpolation*, Journal of Approximation Theory 34, 1982, pp. 99–114, doi:10.1016/0021-9045(82)90085-5

[Delv89] **Delvos**, F.-J.; **Schempp**, W.: *Boolean Methods in Interpolation and Approximation*, Pitman Research Notes in Mathematics 230, Longman, 1989, isbn:978-0-470-21583-8

[Don09] **Donahue**, M. M.; **Buzzard**, G. T.; **Rundell**, A. E.: *Robust parameter identification with adaptive sparse grid-based optimization for nonlinear systems biology models*, Proceedings of the 2009 American Control Conference, St. Louis: IEEE, 2009, pp. 5055–5060, doi:10.1109/ACC.2009.5160512

[Fere05] **Ferenczi**, I.: *Globale Optimierung unter Nebenbedingungen mit dünnen Gittern*, Diploma thesis, Technical University of Munich, Department of Mathematics, 2005

[Fern05] **Fernandez**, J. W.; **Hunter**, P. J.: *An anatomically based patient-specific finite element model of patella articulation: Towards a diagnostic tool*, Biomechanics and Modeling in Mechanobiology 4.1, 2005, pp. 20–38, doi:10.1007/s10237-005-0072-0

[Fra16] **Franzelin**, F.; **Pflüger**, D.: *From data to uncertainty: An efficient integrated data-driven sparse grid approach to propagate uncertainty*, Sparse Grids and Applications – Stuttgart 2014, ed. by **Garcke**, J.; **Pflüger**, D., Lecture Notes in Computational Science and Engineering 109, Springer, 2016, pp. 29–49, doi:10.1007/978-3-319-28262-6_2

[Fra17] **Franzelin**, F.: *Data-Driven Uncertainty Quantification for Large-Scale Simulations*, PhD thesis, University of Stuttgart, Department of Computer Science, IPVS, 2017

[Fre07] **Freund**, R. W.; **Hoppe**, R. H. W.: *Stoer/Bulirsch: Numerische Mathematik 1*, 10th ed., Springer, 2007, isbn:978-3-540-45389-5

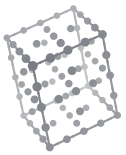
[Gao12] **Gao**, F.; **Han**, L.: *Implementing the Nelder-Mead simplex algorithm with adaptive parameters*, Computational Optimization and Applications 51.1, 2012, pp. 256–277, doi:10.1007/s10589-010-9329-3

[Gar01] **Garcke**, J.; **Griebel**, M.; **Thess**, M.: *Data mining with sparse grids*, Computing 67.3, 2001, pp. 225–253, doi:10.1007/s006070170007

[Gar13] **Garcke**, J.: *Sparse grids in a nutshell*, Sparse Grids and Applications, ed. by **Garcke**, J.; **Griebel**, M., Lecture Notes in Computational Science and Engineering 88, Springer, 2013, pp. 57–80, doi:10.1007/978-3-642-31703-3_3

[Gav13] **Gavana**, A.: *Global Optimization Benchmarks and AMPGO, Test Functions Index*, 2013, https://web.archive.org/web/20171217080109/http://infinity77.net/global_optimization/test_functions.html

[Ger98] **Gerstner**, T.; **Griebel**, M.: *Numerical integration using sparse grids*, Numerical Algorithms 18.3–4, 1998, pp. 209–232, doi:10.1023/A:1019129717644



[Gol71] **Goldstein**, A. A.; **Price**, J. F.: *On descent from local minima*, Mathematics of Computation 25.115, 1971, pp. 569–574, doi:10.2307/2005219

[Gra94] **Graham**, R. L.; **Knuth**, D. E.; **Patashnik**, O.: *Concrete Mathematics, A Foundation of Computer Science*, 2nd ed., Addison-Wesley, 1994, isbn:978-0-201-55802-9

[Gri10] **Griebel**, M.; **Hegland**, M.: *A finite element method for density estimation with gaussian process priors*, SIAM Journal on Numerical Analysis 47.6, 2010, pp. 4759–4792, doi:10.1137/080736478

[Gri92] **Griebel**, M.; **Schneider**, M.; **Zenger**, C.: *A combination technique for the solution of sparse grid problems*, Proceedings of the IMACS International Symposium on Iterative Methods in Linear Algebra, ed. by **Groen**, P. de; **Beauwens**, R., North Holland, 1992, pp. 263–281, isbn:978-0-444-89248-5

[Hanse03] **Hansen**, N.; **Müller**, S. D.; **Koumoutsakos**, P.: *Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES)*, Evolutionary Computation 11.1, 2003, pp. 1–18, doi:10.1162/106365603321828970

[Hanss05] **Hanss**, M.: *Applied Fuzzy Arithmetic, An Introduction with Engineering Applications*, Springer, 2005, isbn:978-3-540-24201-7

[Hee18] **Heene**, M.: *A Massively Parallel Combination Technique for the Solution of High-Dimensional PDEs*, PhD thesis, University of Stuttgart, Institute for Parallel and Distributed Systems, 2018, doi:10.18419/opus-9893

[Heg07] **Hegland**, M.; **Garcke**, J.; **Challis**, V.: *The combination technique and some generalisations*, Linear Algebra and its Applications 420.2–3, 2007, pp. 249–275, doi:10.1016/j.laa.2006.07.014

[Hei14] **Heidlauf**, T.; **Röhrle**, O.: *A multiscale chemo-electro-mechanical skeletal muscle model to analyze muscle contraction and force generation for different muscle fiber arrangements*, Frontiers in Physiology 5, 498, 2014, pp. 1–14, doi:10.3389/fphys.2014.00498

[Hig02] **Higham**, N. J.: *Accuracy and Stability of Numerical Algorithms*, 2nd ed., SIAM, 2002, isbn:978-0-89871-521-7

[Höl03] **Höllig**, K.: *Finite Element Methods with B-Splines*, SIAM, 2003, isbn:978-0-89871-699-3

[Höl12] **Höllig**, K.; **Hörner**, J.; **Hoffacker**, A.: *Finite element analysis with B-splines: Weighted and isogeometric methods*, Curves and Surfaces: 7th International Conference on Curves and Surfaces, ed. by **Boissonnat**, J.-D. et al., Lecture Notes in Computer Science 6920, Springer, 2012, doi:10.1007/978-3-642-27413-8_21

[Höl13] **Höllig**, K.; **Hörner**, J.: *Approximation and Modeling with B-Splines*, SIAM, 2013, isbn:978-1-611972-94-8

[Hor16] **Horneff**, V.; **Maurer**, R.; **Schober**, P.: *Efficient parallel solution methods for dynamic portfolio choice models in discrete time*, working paper, SSRN, 2016, doi:10.2139/ssrn.2665031

[Hüb14] **Hübner**, D.: *Mehrdimensionale Parametrisierung der Mikrozellen in der Zwei-Skalen-Optimierung*, Master's thesis, FAU Erlangen-Nürnberg, Department of Mathematics, 2014

[Jud14] **Judd**, K. L. et al.: *Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain*, Journal of Economic Dynamics and Control 44, 2014, pp. 92–123, doi:10.1016/j.jedc.2014.03.003

[Kal10] **Kaltenbacher**, M.: *Advanced simulation tool for the design of sensors and actuators*, Procedia Engineering 5, 2010: Eurosensor XXIV Conference, 5–8 September 2010, Linz, Austria, ed. by **Jakoby**, B.; **Vellekoop**, M. J., pp. 597–600, doi:10.1016/j.proeng.2010.09.180

[Ken95] **Kennedy**, J.; **Eberhart**, R.: *Particle swarm optimization*, 1995 IEEE International Conference on Neural Networks, vol. 4, IEEE, 1995, pp. 1942–1948, doi:10.1109/ICNN.1995.488968

[Kir14] **Kiranyaz**, S.; **Ince**, T.; **Gabbouj**, M.: *Multidimensional Particle Swarm Optimization for Machine Learning and Pattern Recognition*, Adaptation, Learning, and Optimization 15, Springer, 2014, isbn:978-3-642-37845-4

[Kli05] **Klimke**, A.; **Wohlmuth**, B.: *Algorithm 847: `spinterp`, Piecewise multilinear hierarchical sparse grid interpolation in matlab*, ACM Transactions on Mathematical Software 31.4, 2005, pp. 561–579, doi:10.1145/1114268.1114275

[Kli06] **Klimke**, W. A.: *Uncertainty Modeling Using Fuzzy Arithmetic and Sparse Grids*, Industriemathematik und Angewandte Mathematik, Shaker Verlag, 2006, isbn:978-3-8322-4766-9

[Knu74] **Knuth**, D. E.: *Structured programming with `go to` statements*, ACM Computing Surveys 6.4, 1974, pp. 261–301, doi:10.1145/356635.356640

[Knu77] **Knuth**, D. E.: *Notes on the van Emde Boas Construction of Priority Deques: An Instructive Use of Recursion*, 1977, https://web.archive.org/web/20180116050102/https://staff.fnwi.uva.nl/p.vanemdeboas/knuthnote.pdf, published as: The Correspondence between Donald E. Knuth and Peter van Emde Boas on Priority Deques During the Spring of 1977

[Kud95] **Kudryavtsev**, L. D.: *Implicit function*, Encyclopaedia of Mathematics, vol. 3: Heaps and Semi-Heaps–Moments, Method of (in Probability Theory), ed. by **Hazewinkel**, M., Kluwer Academic Publishers, 1995, pp. 145–147, isbn:978-1-55608-010-4

[Laa87] **Laarhoven**, P. J. M. van; **Aarts**, E. H. L.: *Simulated Annealing: Theory and Applications*, Mathematics and Its Applications, Kluwer, 1987, isbn:978-90-481-8438-5

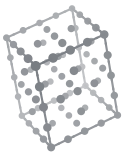
[Lee09] **Lee**, S.-H.; **Sifakis**, E.; **Terzopoulos**, D.: *Comprehensive biomechanical modeling and simulation of the upper body*, ACM Transactions on Graphics 28.4, 99, 2009, pp. 1–17, doi:10.1145/1559755.1559756

[Lem05] **Lemos**, R. R. et al.: *Modeling and simulating the deformation of human skeletal muscle based on anatomy and physiology*, Computer Animation and Virtual Worlds 16.3–4, 2005, pp. 319–330, doi:10.1002/cav.83

[Mar16] **Martin**, F.: *Formoptimierung elastischer Bauteile mit gewichteten B-Splines*, BestMasters, Springer Spektrum, 2016, isbn:978-3-658-13293-4

[Mar17] **Martin**, F.: *WEB-Spline Approximation and Collocation for Singular and Time-Dependent Problems*, Shaker Verlag, 2017, isbn:978-3-8440-5428-6

[McC04] **McCurdy**, C. W.; **Martín**, F.: *Implementation of exterior complex scaling in B-splines to solve atomic and molecular collision problems*, Journal of Physics B: Atomic,



Molecular and Optical Physics 37.4, 2004, pp. 917–936, doi:10.1088/0953-4075/37/4/017

[McK98] **McKinnon**, K. I. M.: *Convergence of the Nelder–Mead simplex method to a non-stationary point*, SIAM Journal on Optimization 9.1, 1998, pp. 148–158, doi:10.1137/s1052623496303482

[Mor87] **Moré**, J. J.; **Wright**, S. J.: *Optimization Software Guide*, Frontiers in Applied Mathematics 14, SIAM, 1987, isbn:978-0-89871-322-0

[Mül15] **Müller-Freitag**, J.: *Ansätze zur Berücksichtigung von Schätzrisiken in der Asset Allocation*, Bachelor's thesis, Goethe University Frankfurt, Faculty of Economics and Business Administration, Department of Finance, 2015

[Mun98] **Munteanu**, C.; **Lazarescu**, V.: *Global search using a new evolutionary framework: The adaptive reservoir genetic algorithm*, Complexity International 5, 1998, https://web.archive.org/web/20110405204539/http://www.complexity.org.au/ci/vol05/munteanu/munteanu.html

[Nel65] **Nelder**, J. A.; **Mead**, R.: *A simplex method for function minimization*, The Computer Journal 7.4, 1965, pp. 308–313, doi:10.1093/comjnl/7.4.308

[Nob16] **Nobile**, F. et al.: *An adaptive sparse grid algorithm for elliptic PDEs with lognormal diffusion coefficient*, Sparse Grids and Applications – Stuttgart 2014, ed. by **Garcke**, J.; **Pflüger**, D., Lecture Notes in Computational Science and Engineering 109, Springer, 2016, pp. 191–220, doi:10.1007/978-3-319-28262-6_8

[Noc99] **Nocedal**, J.; **Wright**, S. J.: *Numerical Optimization*, 2nd ed., Springer Series in Operations Research, Springer, 1999, isbn:978-0-387-98793-4

[Nov96] **Novak**, E.; **Ritter**, K.: *Global optimization using hyperbolic cross points*, State of the Art in Global Optimization, Computational Methods and Applications, ed. by **Floudas**, C. A.; **Pardalos**, P. M., 1996, pp. 19–33, isbn:978-1-4613-3439-2

[Pan08] **Pandey**, D.: *Regression with Spatially Adaptive Sparse Grids in Financial Applications*, Master's thesis, Technical University of Munich, Institute of Computer Science, 2008

[Peh14] **Peherstorfer**, B.; **Pflüger**, D.; **Bungartz**, H.-J.: *Density estimation with adaptive sparse grids for large data sets*, Proceedings of the 2014 SIAM International Conference on Data Mining, ed. by **Zaki**, M. et al., SIAM, 2014, pp. 443–451, doi:10.1137/1.9781611973440.51

[Pfl10] **Pflüger**, D.: *Spatially Adaptive Sparse Grids for High-Dimensional Problems*, Verlag Dr. Hut, 2010, isbn:978-3-86853-555-6

[Pfl13] **Pflüger**, D.: *Spatially adaptive refinement*, Sparse Grids and Applications, ed. by **Garcke**, J.; **Griebel**, M., Lecture Notes in Computational Science and Engineering 88, Springer, 2013, pp. 243–262, doi:10.1007/978-3-642-31703-3_12

[Pfl14] **Pflüger**, D. et al.: *EXAHD: An exa-scalable two-level sparse grid approach for higher-dimensional problems in plasma physics and beyond*, Euro-Par 2014: Parallel Processing Workshops, Revised Selected Papers, Part II, ed. by **Lopes**, L. et al., Lecture Notes in Computer Science 8806, Springer, 2014, pp. 565–576, doi:10.1007/978-3-319-14313-2_48

[Pfl16] **Pflüger**, D.; **Mehl**, M.; **Valentin**, J., et al.: *The scalability-efficiency/maintainability-portability trade-off in simulation software engineering: Examples and a preliminary*

systematic literature review, Proceedings of the 2016 Fourth International Workshop on Software Engineering for High Performance Computing in Computational Science and Engineering (SE-HPCCSE 2016), Held in Conjunction with SC16, Salt Lake City, Utah, IEEE, 2016, pp. 26–34, doi:10.1109/SE-HPCCSE.2016.008

[Pol71] **Polak**, E.: *Computational Methods in Optimization, A Unified Approach*, Mathematics in Science and Engineering 77, Academic Press, 1971, isbn:978-0-12-559350-2

[Pre07] **Press**, W. H. et al.: *Numerical Recipes, The Art of Scientific Computing*, 3rd ed., Cambridge University Press, 2007, isbn:978-0-521-88068-8

[Qia13] **Qian**, X.: *Topology optimization in B-spline space*, Computer Methods in Applied Mechanics and Engineering 265, 2013, pp. 15–35, doi:10.1016/j.cma.2013.06.001

[Qua16] **Quak**, E.: *About B-splines. Twenty answers to one question: What is the cubic B-spline for the knots* $-2, -1, 0, 1, 2$*?* Journal of Numerical Analysis and Approximation Theory 45.1, 2016, pp. 37–83

[Rei13] **Reinhardt**, R.; **Hoffmann**, A.; **Gerlach**, T.: *Nichtlineare Optimierung, Theorie, Numerik und Experimente*, Springer Spektrum, 2013, isbn:978-3-8274-2948-3

[Rie93] **Riedmiller**, M.; **Braun**, H.: *A direct adaptive method for faster backpropagation learning: The RPROP algorithm*, 1993 IEEE International Conference on Neural Networks, vol. 1, IEEE, 1993, pp. 586–591, doi:10.1109/ICNN.1993.298623

[Röh16] **Röhrle**, O.; **Sprenger**, M.; **Schmitt**, S.: *A two-muscle, continuum-mechanical forward simulation of the upper limb*, Biomechanics and Modeling in Mechanobiology 16.3, 2016, pp. 743–762, doi:10.1007/s10237-016-0850-x

[Run00] **Runarsson**, T. P.; **Yao**, X.: *Stochastic ranking for constrained evolutionary optimization*, IEEE Transactions on Evolutionary Computation 4.3, 2000, pp. 284–294, doi:10.1109/4235.873238

[Rus18] **Rust**, J.: *Dynamic programming*, The New Palgrave Dictionary of Economics, 3rd ed., Palgrave Macmillan, 2018, pp. 3133–3158, isbn:978-1-349-95188-8

[Schn03] **Schneider**, P. J.; **Eberly**, D. H.: *Geometric Tools for Computer Graphics*, Morgan Kaufmann, 2003, isbn:978-1-55860-594-7

[Schob18] **Schober**, P.: *Solving dynamic portfolio choice models in discrete time using spatially adaptive sparse grids*, Sparse Grids and Applications – Miami 2016, ed. by **Garcke**, J. et al., Lecture Notes in Computational Science and Engineering 123, Springer, 2018, pp. 135–173, doi:10.1007/978-3-319-75426-0_7

[Schoena93] **Schoenauer**, M.; **Xanthakis**, S.: *Constrained GA optimization*, Proceedings of the 5th International Conference on Genetic Algorithms, ed. by **Forrest**, S., Morgan Kaufmann, 1993, pp. 573–580, isbn:978-1-55860-299-1

[Schoenb46] **Schoenberg**, I. J.: *Contributions to the problem of approximation of equidistant data by analytic functions*, Quarterly Applied Mathematics 4, 1946, pp. 45–99, 112–141

[Schoenb67] **Schoenberg**, I. J.: *On spline functions*, Inequalities, Proceedings of a Symposium Held at Wright-Patterson Air Force Base, Ohio, August 19–27, 1965, ed. by **Shisha**, O., Academic Press, 1967, pp. 255–291, isbn:978-0-126-40350-3

[Schoenb72] **Schoenberg**, I. J.: *Cardinal interpolation and spline functions: II, Interpolation of data of power growth*, Journal of Approximation Theory 6.4, 1972, doi:10.1016/0021-9045(72)90048-2

[Schoenb73] **Schoenberg**, I. J.: *Cardinal Spline Interpolation*, CBMS-NSF Regional Conference Series in Applied Mathematics 12, SIAM, 1973, isbn:978-0-89871-009-0

[Schw77] **Schwefel**, H.-P.: *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*, Mit einer vergleichenden Einführung in die Hill-Climbing- und Zufallsstrategien, Interdisciplinary Systems Research 26, Birkhäuser, 1977, isbn: 978-3-7643-0876-6

[Shara11] **Sharafi**, B. et al.: *Strains at the myotendinous junction predicted by a micromechanical model*, Journal of Biomechanics 44.16, 2011, pp. 2795–2801, doi:10.1016/j.jbiomech.2011.08.025

[Sharp66] **Sharpe**, W. F.: *Mutual fund performance*, The Journal of Business 39.1, 1966, pp. 119–138, doi:10.1086/294846

[Sic11] **Sickel**, W.; **Ullrich**, T.: *Spline interpolation on sparse grids*, Applicable Analysis 90.3–4, 2011, pp. 337–383, doi:10.1080/00036811.2010.495336

[Sig01] **Sigmund**, O.: *A 99 line topology optimization code written in Matlab*, Structural and Multidisciplinary Optimization 21.2, 2001, pp. 120–127, doi:10.1007/s001580050176

[Smo63] **Smolyak**, S. A.: *Quadrature and interpolation formulas for tensor products of certain classes of functions*, trans. from the Russian by **Brown**, J. R., Soviet Mathematics Doklady 4, 1963, pp. 240–243, Russian original: Doklady Akademii Nauk SSSR 148.5, 1963, pp. 1042–1045

[Sön13] **Sönnerlind**, H.: *Why All These Stresses and Strains?* COMSOL Inc., 2013, https://web.archive.org/web/20150920210345/https://www.comsol.com/blogs/why-all-these-stresses-and-strains/

[Spi96] **Spitzer**, V. et al.: *The Visible Human Male: A technical report*, Journal of the American Medical Informatics Association 3.2, 1996, pp. 118–130, doi:10.1136/jamia.1996.96236280

[Spr15] **Sprenger**, M.: *A 3D Continuum-Mechanical Model for Forward-Dynamics Simulations of the Upper Limb*, PhD thesis, University of Stuttgart, Institute of Applied Mechanics (Civil Engineering), 2015, doi:10.18419/opus-8777

[Sri10] **Srinivas**, N. et al.: *Gaussian process optimization in the bandit setting: No regret and experimental design*, Proceedings of the 27th International Conference on Machine Learning (ICML'10), Omnipress, 2010, pp. 1015–1022, isbn:978-1-60558-907-7

[Stor97] **Storn**, R.; **Price**, K.: *Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces*, Journal of Global Optimization 11.4, 1997, pp. 341–359, doi:10.1023/A:1008202821328

[Stoy18] **Stoyanov**, M.: *User Manual: Toolkit for Adaptive Stochastic Modeling and Non-Intrusive Approximation (TASMANIAN)*, version 5.1, ORNL/TM-2015/596, 2018

[Tem82] **Temljakov**, V. N.: *Approximation of periodic functions of several variables with bounded mixed difference*, trans. from the Russian by **Cooke**, R. L., Mathematics of the USSR Sbornik 41.1, 1982, doi:10.1070/SM1982v041n01ABEH002220, Russian original: Matematicheskii Sbornik 113(155).1(9), 1980, pp. 65–80

[Tou15] **Toussaint**, M.: *Introduction to Optimization*, lecture slides and exercices, 2015, https://web.archive.org/web/20180619123151/https://ipvs.informatik.uni-stuttgart.de/mlr/marc/teaching/15-Optimization/15-Optimization-script.pdf

[Ulb12] **Ulbrich**, M.; **Ulbrich**, S.: *Nichtlineare Optimierung*, Mathematik Kompakt, Birkhäuser, 2012, isbn:978-3-0346-0142-9

[Uns92] **Unser**, M.; **Aldroubi**, A.; **Eden**, M.: *On the asymptotic convergence of B-spline wavelets to gabor functions*, IEEE Transactions on Information Theory 38.2, 1992, pp. 864–872, doi:10.1109/18.119742

[Vald17] **Valdez**, S. I. et al.: *Topology optimization benchmarks in 2D: Results for minimum compliance and minimum volume in planar stress problems*, Archives of Computational Methods in Engineering 24.4, 2017, pp. 803–839, doi:10.1007/s11831-016-9190-3

[Vale12] **Valentin**, J.: *Spline-Approximation unregelmäßig verteilter Daten*, Bachelor's thesis, University of Stuttgart, Department of Mathematics, IMNG, 2012, doi:10.18419/opus-5143

[Vale14] **Valentin**, J.: *Hierarchische Optimierung mit Gradientenverfahren auf Dünngitterfunktionen*, Master's thesis, University of Stuttgart, Department of Computer Science, IPVS, 2014, doi:10.18419/opus-3462

[Vale16] **Valentin**, J.; **Pflüger**, D.: *Hierarchical gradient-based optimization with B-splines on sparse grids*, Sparse Grids and Applications – Stuttgart 2014, ed. by **Garcke**, J.; **Pflüger**, D., Lecture Notes in Computational Science and Engineering 109, Springer, 2016, pp. 315–336, doi:10.1007/978-3-319-28262-6_13

[Vale18a] **Valentin**, J.; **Pflüger**, D.: *Fundamental splines on sparse grids and their application to gradient-based optimization*, Sparse Grids and Applications – Miami 2016, ed. by **Garcke**, J. et al., Lecture Notes in Computational Science and Engineering 123, Springer, 2018, pp. 229–251, doi:10.1007/978-3-319-75426-0_10

[Vale18b] **Valentin**, J. et al.: *Gradient-based optimization with B-splines on sparse grids for solving forward-dynamics simulations of three-dimensional, continuum-mechanical musculoskeletal system models*, International Journal for Numerical Methods in Biomedical Engineering 34.5, e2965, 2018, pp. 1–21, doi:10.1002/cnm.2965

[Wal16] **Walz**, N.-P.: *Fuzzy Arithmetical Methods for Possibilistic Uncertainty Analysis*, Shaker Verlag, 2016, isbn:978-3-8440-4911-4

[Wan14] **Wang**, Z. et al.: *Bayesian multi-scale optimistic optimization*, Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics, Proceedings of Machine Learning Research 33, 2014, pp. 1005–1014

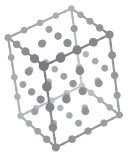
[Wer11] **Werner**, D.: *Funktionalanalysis*, 7th ed., Springer, 2011, isbn:978-3-642-21016-7

[Win10] **Winschel**, V.; **Krätzig**, M.: *Solving, estimating, and selecting nonlinear dynamic models without the curse of dimensionality*, Econometrica 78.2, 2010, pp. 803–821, doi:10.3982/ECTA6297

[Wol97] **Wolpert**, D. H.; **Macready**, W. G.: *No free lunch theorems for optimization*, IEEE Transactions on Evolutionary Computation 1.1, 1997, pp. 67–82, doi:10.1109/4235.585893

[Wu13] **Wu**, T. et al.: *Modelling facial expressions: A framework for simulating nonlinear soft tissue deformations using embedded 3D muscles*, Finite Elements in Analysis and Design 76, 2013, pp. 63–70, doi:10.1016/j.finel.2013.08.002

[Xu16] **Xu**, K.: *The Chebyshev points of the first kind*, Applied Numerical Mathematics 102, 2016, pp. 17–30, doi:10.1016/j.apnum.2015.12.002



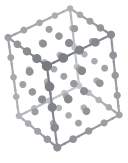
[Zad75] **Zadeh**, L. A.: *The concept of a linguistic variable and its application to approximate reasoning—I*, Information Sciences 8.3, 1975, pp. 199–249, doi:10.1016/0020-0255(75)90036-5

[Zak14] **Zakaria**, R.; **Wahab**, A. F.; **Gobithaasan**, R. U.: *Fuzzy B-spline surface modeling*, Journal of Applied Mathematics 2014, 285045, 2014, pp. 1–8, doi:10.1155/2014/285045

[Zen91] **Zenger**, C.: *Sparse grids*, Parallel Algorithms for Partial Differential Equations: Proceedings of the Sixth GAMM-Seminar, ed. by **Hackbusch**, W., Notes on Numerical Fluid Mechanics 31, Vieweg, 1991, pp. 241–251, isbn:978-3-528-07631-3

[Zha17] **Zhang**, W. et al.: *Topology optimization with closed B-splines and Boolean operations*, Computer Methods in Applied Mechanics and Engineering 315, 2017, pp. 652–670, doi:10.1016/j.cma.2016.11.015

[Zie09] **Zielinski**, K.: *Optimizing Real-World Problems with Differential Evolution and Particle Swarm Optimization*, PhD thesis, University of Bremen, Department of Physics, Electrical Engineering, and Information Engineering, 2009




interpolation
problem
grid
sparse
level
boundary
solution
basis
first
opt
linear
interpolant
value
different
dimension
objective
optimization
hierarchical
number
two
case
hierarchization
time
tensor
function
error
space
Fig.
method
spline
regular
one
use
B-spline
condition
model
adaptive
Sec.
degree
fundamental
point
fuzzy
not-a-knot
micro-cell
algorithm