Computational Economics (2022) 59:185–224
https://doi.org/10.1007/s10614-020-10061-x

Solving High-Dimensional Dynamic Portfolio Choice Models with Hierarchical B-Splines on Sparse Grids

Peter Schober[1] · **Julian Valentin**[2] · **Dirk Pflüger**[2]

Accepted: 15 October 2020 / Published online: 4 January 2021

Abstract
Discrete time dynamic programming to solve dynamic portfolio choice models has three immanent issues: firstly, the curse of dimensionality prohibits more than a handful of continuous states. Secondly, in higher dimensions, even regular sparse grid discretizations need too many grid points for sufficiently accurate approximations of the value function. Thirdly, the models usually require continuous control variables, and hence gradient-based optimization with smooth approximations of the value function is necessary to obtain accurate solutions to the optimization problem. For the first time, we enable accurate and fast numerical solutions with gradient-based optimization while still allowing for spatial adaptivity using hierarchical B-splines on sparse grids. When compared to the standard linear bases on sparse grids or finite difference approximations of the gradient, our approach saves an order of magnitude in total computational complexity for a representative dynamic portfolio choice model with varying state space dimensionality, stochastic sample space, and choice variables.

Keywords Curse of dimensionality · Dynamic portfolio choice · Discrete time dynamic programming · Gradient-based optimization · Spatially adaptive sparse grids · Hierarchical B-splines

✉ Peter Schober
schober@finance.uni-frankfurt.de

Julian Valentin
julian.valentin@ipvs.uni-stuttgart.de

Dirk Pflüger
dirk.pflueger@ipvs.uni-stuttgart.de

[1] Finance Department, Goethe University Frankfurt, Theodor-W.-Adorno-Platz 3, 60323 Frankfurt am Main, Germany

[2] Institute for Parallel and Distributed Systems, University of Stuttgart, Universitätsstraße 38, 70569 Stuttgart, Germany

1 Introduction

A common approach to solve dynamic portfolio choice models in discrete time is dynamic programming, iterating over the value function backwards in time. Starting from the known value function at final time T, the value function is approximated on a state space grid, assuming that the state space is continuous. To determine the next iterate of the value function at each grid point at time $T-1$ we have to solve an optimization problem that depends on the value function of the previous iterate at time T. When a tensor product approximation is used, this approach suffers from the curse of dimensionality as the number of grid points of the approximation grows exponentially with the dimensionality of the state space. In addition, solving for the current value function iterate at a grid point relies on an accurate solution of the underlying optimization problem. When the portfolio choice is continuous, e.g., choosing the investment amount in stocks, bonds, etc., the computation of the optimal solution can be greatly accelerated by gradient-based optimization routines if the gradient of the objective function is available.

Recently, sparse grids have been successfully employed to break the curse of dimensionality in high-dimensional dynamic models (Brumm and Scheidegger 2017; Judd et al. 2014; Schober 2018; Winschel and Krätzig 2010).[1] A standard d-dimensional tensor product grid on the unit hypercube $[0,1]^d$ with mesh size 2^{-n}, $n \in \mathbb{N}$, and no points on the boundary contains $2^n - 1$ grid points per coordinate direction and thus $\mathcal{O}(2^{nd})$ points in total, growing exponentially with the dimensionality d. In contrast, a regular sparse grid with the same mesh size contains only $\mathcal{O}(2^n n^{d-1})$ points. The error of the sparse grid approximation of a function with homogeneous boundary conditions using piecewise linear basis functions is $\mathcal{O}(2^{-2n} n^{d-1})$ with respect to the L^2 and L^∞ norm if the approximated function has bounded mixed second derivatives (Bungartz and Griebel 2004; Zenger 1991). This is only slightly worse than the corresponding error $\mathcal{O}(2^{-2n})$ for the case of full tensor product grids.

In higher dimensions, even regular sparse grids need too many grid points for a sufficiently accurate approximation when solving high-dimensional dynamic models (Brumm and Scheidegger 2017). Fortunately, for approximations in the standard piecewise linear basis, the hierarchical structure of sparse grids allows for spatially adaptive refinement of the grid by inserting the $2d$ children of only certain leaves in the hierarchical structure. Spatially adaptive refinement has successfully been employed to solve high-dimensional dynamic models by Brumm and Scheidegger (2017) and Schober (2018). Unfortunately, approximations of the value function using the standard piecewise linear basis are not continuously differentiable and, hence, have discontinuous gradients. This poses a problem to gradient-based optimization techniques, which rely on a twice continuously differentiable approximation of the objective function to ensure convergence (Schober 2018).

[1] See Cai (2019) for a discussion on alternative approximation methods for dynamic economic models that also do not suffer from the curse of dimensionality.

Global polynomial approximations have shown to work well with value function iteration and continuous choices for solving dynamic economic models (Cai and Judd 2015; Judd et al. 2014) as they are globally smooth. Smolyak's formula can be used to construct sparse grid approximations (Barthelmann et al. 2000) on global polynomial bases, which can be refined adaptively with regard to specific dimensions of the state space (Judd et al. 2014) and with regard to the hierarchical surpluses, i.e., locally adaptively (Stoyanov 2017). Value function iteration with the use of gradient information to approximate the value function more accurately with global polynomials is also possible (Cai and Judd 2015).

However, B-splines are much more flexible than global polynomials (Valentin and Pflüger 2016; Valentin 2019). While global polynomial approximations are bound to certain grid structures to avoid Runge's phenomenon or similar issues, B-spline basis functions can be employed on any nested spatially adaptive grid hierarchy. They allow for simultaneous local- and degree-adaptive refinement (*hp*-adaptivity), implying that one could use a smaller or larger mesh size and/or degree of the B-spline basis functions in certain regions of the state space, e.g., to resolve kinks. In addition, the local basis functions are faster to evaluate than the conventional global polynomial basis functions. Approximations with B-splines of cubic degree (or higher) are twice continuously differentiable, and readily supply smooth and explicit approximations of both, the value function and the gradient. Compared to approximating the derivatives with finite differences, the optimization is not only more accurate but also significantly faster, especially when the number of optimization variables is large (Valentin 2019). B-splines have thus proven useful for computing numerical solutions to numerous dynamic models when finding the root of the gradient is required (Chu et al. 2013; Habermann and Kindermann 2007; Judd and Solnick 1994; Philbrick and Kitanidis 2001).

In total, three issues with discrete time dynamic programming for dynamic portfolio choice models with continuous choices emerge: the curse of dimensionality, the lack of spatial adaptivity, and the lack of continuous gradients. It is apparent that current economic literature deals with these issues only in isolation, e.g., by combining sparse grids with global polynomial basis functions, or using sparse grids with non-smooth local linear basis functions to allow for spatial adaptivity. These approaches are hence computationally inefficient in accurately solving high-dimensional dynamic portfolio choice models or any high-dimensional dynamic economic model that requires smooth approximations or gradient-based optimization.

This paper is the first to address all of these issues at once by combining hierarchical B-splines with sparse grids to approximate the value function and its gradient. Thus, we enable accurate and fast numerical solutions using gradient-based optimization while still allowing for spatial adaptivity (Pflüger 2010; Valentin and Pflüger 2016). The hierarchical grid structure allows us to develop an algorithm that uses the local adaptivity similar to Brumm and Scheidegger (2017) and Schober (2018), but interpolates the value function and its gradient with a B-spline basis. Therefore, we create a sparse grid for the value function, for which we interpolate the value function in the piecewise linear basis. We then refine the grid using the standard hierarchical surplus-volume-based refinement

criterion. Finally, we interpolate the value function with hierarchical B-spline basis functions on the spatially adaptively refined sparse grid.

We focus our study on the numerical accuracy of our approach. Therefore, we choose a dynamic portfolio choice model with multiple stocks, one bond, and consumption. For buying and selling the stocks, linear transaction costs have to be deducted. The resulting optimization problem is high-dimensional in terms of the state space, stochastic sample space, and choice variables. Hence, this problem is especially suited for a complexity analysis. At the same time, this model is similar to models from a vast strand of state-of-the-art literature on dynamic portfolio choice, e.g., Barberis and Huang (2009), Cocco et al. (2005), De Giorgi and Legg (2012), Horneff et al. (2010), Horneff et al. (2008), Hubener et al. (2016), Hubener et al. (2014), Inkmann et al. (2011). Consequently, our approach can be generalized to a broad class of dynamic portfolio choice models with only minor modifications.

Dynamic portfolio choice models with transaction costs have been studied economically, e.g., by Abrams and Karmarkar (1980), Kamin (1975), Liu and Loewenstein (2002), Magill and Constantinides (1976), and extensively numerically by Cai (2009), Cai and Judd (2010), Cai et al. (2015), Cai et al. (2020). The latter report computational times and economic solutions for higher-dimensional transaction costs problems. They employ polynomial interpolation with only few polynomial nodes and parallelization to solve these problems with and without consumption using value function iteration in discrete time. Cai et al. (2020) present convergence results and computational times for the three-dimensional transaction costs problem with consumption and numerical errors for the four-dimensional problem using complete Chebyshev polynomials to approximate the value function. However, only we have employed spatially adaptive sparse grids to the transaction costs problem, which induces optimization on continuous choices (Schober 2018). We also apply local adaptivity to compute the optimal policies from the solution to the underlying optimization problem as earlier suggested by us (Schober 2018) and Brumm and Grill (2014).

A complexity analysis reveals that cubic B-splines save more than one order of magnitude in computational effort compared to the state of the art with the linear basis (Brumm and Scheidegger 2017) and/or finite difference approximations of the gradient on regular sparse grids. Using spatially adaptive refinement of the optimal policy, we solve the problem for up to five dimensions where highly accurate solutions on regular sparse grids would require hundreds of thousands of grid points, and are thus no longer suitable. Here, spatially adaptive refinement allows a comparably low base resolution in the solution process of the value function, and, in a second step, adds grid points in the optimal policies where required. By this, we obtain low reported unit-free Euler equation errors for the transaction costs problem.

The rest of this article is structured as follows: In Sect. 2, we define the general class of dynamic portfolio choice models for which our approach is applicable. Section 3 introduces hierarchical B-splines on spatially adaptive sparse grids, leading to the definition of hierarchical weakly fundamental not-a-knot splines. Algorithms for solving dynamic portfolio choice models with B-splines on spatially adaptive sparse grids are discussed in Sect. 4. We analyze the complexity and demonstrate the numerical

accuracy of our approach solving the transaction costs problem in Sect. 5 before concluding in Sect. 6.

2 Discrete Time Dynamic Portfolio Choice Models

We consider discrete time dynamic portfolio choice models with finite time horizon T in which the investor seeks to maximize additive expected life-time utility u from consumption c_t:

$$\mathbb{E}_0\left[\sum_{t=0}^{T} \rho^t u(c_t(\boldsymbol{p}_t, \boldsymbol{x}_t))\right]. \tag{1}$$

Here, she has $m_{\boldsymbol{p}}$ continuous choices $\boldsymbol{p}_t \in \Psi \subset \mathbb{R}^{m_{\boldsymbol{p}}}$ (e.g., investment amounts in stocks and bonds) with respect to d continuous states $\boldsymbol{x}_t \in \Omega \subset \mathbb{R}^d$ (e.g., current financial wealth or labor income) she can reside in at time t. In addition, the transition from state $\boldsymbol{x}_t$ to $\boldsymbol{x}_{t+1}$ does not only depend on her choices and her state, but may also be subject to $m_{\boldsymbol{\zeta}}$ random shocks $\boldsymbol{\zeta}_t \in Z$ (such as stock returns or labor income shocks), which are drawn from the sample space $Z \subset \mathbb{R}^{m_{\boldsymbol{\zeta}}}$. The random variable $\boldsymbol{f}_t : \Psi \times \Omega \times Z \to \Omega, (\boldsymbol{p}_t, \boldsymbol{x}_t, \boldsymbol{\zeta}_t) \mapsto \boldsymbol{x}_{t+1}$, then describes the continuous state dynamics between t and $t+1$. We denote by $\rho < 1$ the subjective time discount factor, and we assume the utility function to be of *Constant Relative Risk Aversion* type with risk aversion $\gamma > 1$:

$$u(c_t) = \frac{1}{1-\gamma} c_t^{1-\gamma}. \tag{2}$$

It is also straightforward to include discrete choices (compare also, e.g., Brumm and Scheidegger 2017) in the trivial way and discrete states (Schober 2018) in this model setup. Furthermore, the model can be generalized to further utility functions, e.g., to Epstein and Zin (1989) utility and to utility functions with a narrow framing component (Barberis and Huang 2009). In this paper, we disregard these modeling choices purely for simplicity.

By the Bellman principle (Bellman 1954), this utility maximization problem can be reformulated in terms of the value function j_t for $t = 0, \dots, T$ with known terminal utility v

$$j_t(\boldsymbol{x}_t) = \max_{\boldsymbol{p}_t} \left\{ u(c_t(\boldsymbol{p}_t, \boldsymbol{x}_t)) + \rho \mathbb{E}_t\left[j_{t+1}(\boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t, \boldsymbol{\zeta}_t)) \right] \right\}, \quad t < T, \tag{3a}$$

$$j_T(\boldsymbol{x}_T) = v(\boldsymbol{x}_T), \tag{3b}$$

subject to the $m_{\boldsymbol{g}}$ possibly non-linear inequality constraints $\boldsymbol{g}_t : \Psi \times \Omega \to \mathbb{R}^{m_{\boldsymbol{g}}}$:

$$\boldsymbol{g}_t(\boldsymbol{p}_t, \boldsymbol{x}_t) \geq \boldsymbol{0}. \tag{3c}$$

For two vectors $\boldsymbol{a}, \boldsymbol{b}$, we define $\boldsymbol{a} \geq \boldsymbol{b}$ if $a_i \geq b_i$ for all i (and "$\leq$" analogously). The corresponding expected value of j_{t+1} is

$$\mathbb{E}_t\left[j_{t+1}(\boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t, \boldsymbol{\zeta}_t))\right] := \int_Z j_{t+1}(\boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t, \boldsymbol{\zeta}_t))\, \mathrm{d}\boldsymbol{\Phi}_t(\boldsymbol{\zeta}_t \mid \boldsymbol{x}_t)\,. \tag{4}$$

Here, $\boldsymbol{\Phi}_t(\cdot \mid \boldsymbol{x}_t)$ denotes the conditional distribution of $\boldsymbol{\zeta}_t$. Note that we only need the value function at time $t+1$ to determine the value function at time t via maximization.

To numerically solve the optimization problem (3), discrete-time dynamic programming iterating over the value function is common (Judd 1998; Rust 2008). Therefore, the value function j_t is restricted to a finite grid on the (truncated) state space with N_t grid points $\boldsymbol{x}_t^{(k)}$, $k = 1, \ldots, N_t$. The value function values in between grid points are interpolated by

$$j_t^{\mathrm{S}}(\boldsymbol{x}_t) := \sum_{k=1}^{N_t} \alpha_k \varphi_k(\boldsymbol{x}_t)\,, \tag{5}$$

with basis functions φ_k and coefficients α_k, which are chosen in such a way that the interpolant j_t^{S} fits the known function values at all grid points $\boldsymbol{x}_t^{(k)}$. Beginning with the known final solution at time T, the Bellman equation is solved backwards in time until the value function is computed for each grid point at each $t = T-1, \ldots, 0$.

For $t < T$, let us define the interpolant of the objective function of the maximization in Eq. (3a) for grid point $\boldsymbol{x}_t^{(k)}$ by:

$$\tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}) := u(c_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)})) + \rho \mathbb{E}_t\left[j_{t+1}^{\mathrm{S}}(\boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}, \boldsymbol{\zeta}_t))\right]. \tag{6}$$

The maximization of this target function (6) with respect to $\boldsymbol{p}_t$ can then be performed using *Sequential quadratic programming (SQP)* routines, see "Appendix A.1".

To compute the expectation with respect to $\mathrm{d}\boldsymbol{\Phi}$, numerical integration can be used if the conditional distributions $\boldsymbol{\Phi}(\cdot \mid \boldsymbol{x}_t^{(k)})$ are known.[2]

3 Hierarchical B-Splines on Sparse Grids

As discussed in Sect. 1, hierarchical B-splines on sparse grids provide numerous advantages over other basis choices, especially in the context of optimization (Valentin 2019; Valentin and Pflüger 2016). In addition, they allow for spatially adaptive refinement by applying the standard surplus-based refinement criterion. To compute the coefficients of the B-spline approximation, usually a computationally expensive linear system has to be solved. Therefore, we determine the underlying grid structure by applying the surplus-based refinement criterion on the piecewise linear basis and interpolating with B-splines on the resulting grid. As proven in our previous work

[2] In this paper, we assume that numerical quadrature is possible. The applicability of the algorithms does not change if the distribution is generated by a different method, e.g., a Monte Carlo simulation. However, due to their slow convergence, Monte Carlo integration methods need a large simulated sample to obtain the accuracy required for the numerical optimization routine, see, e.g., the discussion by Cai (2019).

(Valentin 2019), the computational effort needed for the computation of the coefficients can be further reduced by using the unidirectional principle. This is facilitated by weakly fundamental not-a-knot splines and the insertion of some additional grid points.

3.1 Not-A-Knot B-Spline Basis

Let $p \in \mathbb{N}_0$, $m \in \mathbb{N}$, and $\boldsymbol{\xi} = (\xi_0, \dots, \xi_{m+p})$ be an increasing sequence of real numbers. The *B-spline* $b^p_{k,\boldsymbol{\xi}}$ of degree p for the knot sequence $\boldsymbol{\xi}$ is defined via the Cox–de Boor recurrence (Cox 1972; de Boor 1972; Höllig and Hörner 2013)

$$\begin{aligned} b^p_{k,\boldsymbol{\xi}}(x) &:= \frac{x - \xi_k}{\xi_{k+p} - \xi_k} b^{p-1}_{k,\boldsymbol{\xi}}(x) + \frac{\xi_{k+p+1} - x}{\xi_{k+p+1} - \xi_{k+1}} b^{p-1}_{k+1,\boldsymbol{\xi}}(x), \\ b^0_{k,\boldsymbol{\xi}}(x) &:= \begin{cases} 1 & \text{if } x \in [\xi_k, \xi_{k+1}), \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{7}$$

where $k = 0, \dots, m-1$.

It can be shown that for $m > p$, the B-splines $b^p_{0,\boldsymbol{\xi}}, \dots, b^p_{m-1,\boldsymbol{\xi}}$ form a basis of the spline space $S^p_{\boldsymbol{\xi}} := \operatorname{span}\{b^p_{k,\boldsymbol{\xi}} \mid k = 0, \dots, m-1\}$ on $D^p_{\boldsymbol{\xi}} := [\xi_p, \xi_m]$ (Höllig and Hörner 2013). The space $S^p_{\boldsymbol{\xi}}$ contains exactly those functions $s : D^p_{\boldsymbol{\xi}} \to \mathbb{R}$ which are piecewise polynomial of degree smaller or equal to p on every knot interval $[\xi_k, \xi_{k+1}]$ in the interior of $D^p_{\boldsymbol{\xi}}$ ($k = p, \dots, m-1$) and at least $p-1$ times continuously differentiable at every knot ξ_k in the interior of $D^p_{\boldsymbol{\xi}}$, $k = p+1, \dots, m-1$ (Höllig and Hörner 2013).

For simplicity, we restrict the considerations and results in this paper to cubic B-splines (i.e., $p = 3$) although it is important to note that our method can be generalized to arbitrary odd B-spline degrees. A common special case is the case of linear B-splines ($p = 1$, so called *hat functions*), which are commonly used as basis functions for sparse grids.

We consider equidistant grid points $x_{\ell,i} := ih_\ell$ on the unit interval [0, 1] where $\ell \in \mathbb{N}_0$ is the level, $i = 0, \dots, 2^\ell$ is the index, and $h_\ell := 2^{-\ell}$ is the mesh size. We want to find basis functions $\varphi_{\ell,i} : [0,1] \to \mathbb{R}$ such that we can interpolate a given objective function $f : [0,1] \to \mathbb{R}$ on the equidistant grid of level ℓ by a linear combination of the basis functions:

$$\tilde{f}(x_{\ell,j}) = f(x_{\ell,j}) \quad \text{for all } j = 0, \dots, 2^\ell, \quad \text{where} \quad \tilde{f} := \sum_{i=0}^{2^\ell} \alpha_{\ell,i} \varphi_{\ell,i} \tag{8}$$

with some $\alpha_{\ell,i} \in \mathbb{R}$.

The most straightforward choice of B-splines for $\varphi_{\ell,i}$ are *uniform B-splines* that are scaled and translated versions of the cardinal B-spline $b^3 := b^3_{0,(0,1,2,3,4)}$ (Pflüger 2010; Valentin 2019):

$$\varphi^{\text{unif}}_{\ell,i}(x) := b^3(2^\ell x + 2 - i)\,. \tag{9}$$

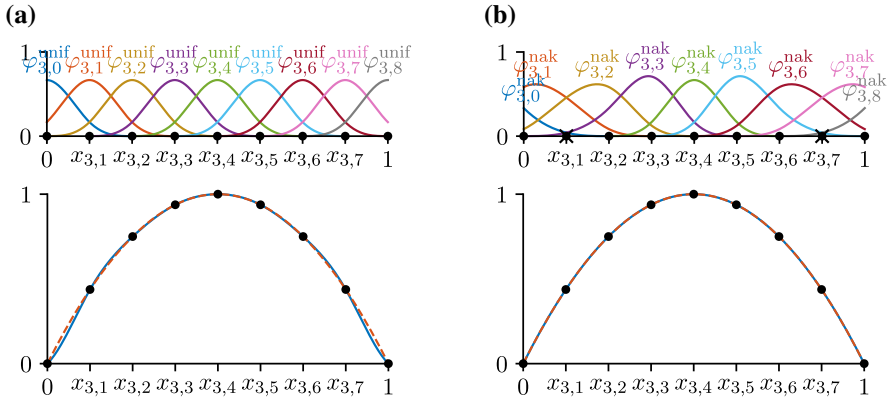


Fig. 1 Nodal B-spline bases and interpolation of parabola. **a** Nodal uniform B-spline basis of level 3 in 1D and interpolation of the parabola $f(x) = 4(x - 0.5)^2$ with this basis, resulting in oscillations near the boundary. **b** The corresponding nodal not-a-knot B-spline basis interpolates the parabola f exactly. The knots at the left-most and right-most inner grid points $x_{\ell,1}$ and $x_{\ell,7}$ (crosses) are removed

The resulting uniform B-splines $\varphi^{\text{unif}}_{\ell,i}$, $i = 0, \ldots, 2^\ell$, are exactly the B-splines $b^3_{k,\xi^{\text{unif}}}$, $k = 0, \ldots, m-1$, that arise from Eq. (7) when choosing the uniform knot sequence $\xi^{\text{unif}} := (x_{\ell,-2}, x_{\ell,-1}, \ldots, x_{\ell,2^\ell+2})$ and $m := 2^\ell + 1$.

However, the interpolation domain on which the B-splines span the spline space would only be $D^3_{\xi^{\text{unif}}} = [\xi_p, \xi_m] = [x_{\ell,1}, x_{\ell,2^\ell-1}] = [2^{-\ell}, 1 - 2^{-\ell}]$. This interval does not contain the two boundary grid points $x_{\ell,0} = 0$ and $x_{\ell,2^\ell} = 1$. This leads to interpolation problems since the spline space on [0, 1] is not contained in the spanned space of the basis functions $\varphi^{\text{unif}}_{\ell,i}$. Even simple polynomials such as $f(x) = 4(x - 0.5)^2$ cannot be represented exactly with the B-spline basis on the whole domain [0, 1] as shown by Fig. 1a and Valentin and Pflüger (2016). Consequently, the approximation quality for more complex functions like the value functions we interpolate in this paper deteriorates unnecessarily, which means that the economic results are not as conclusive as they could be.

As a remedy, we impose so-called not-a-knot boundary conditions by removing the left-most and right-most inner grid points $x_{\ell,1}$ and $x_{\ell,2^\ell-1}$ from the knot sequence (Höllig and Hörner 2013; Valentin 2019). To keep the number $m = 2^\ell + 1$ of B-splines the same, we have to insert two additional knots outside the domain:

$$\xi^{\text{nak}} := (x_{\ell,-3},\ \ldots,\ x_{\ell,0},\ x_{\ell,2},\ \ldots,\ x_{\ell,2^\ell-2},\ x_{\ell,2^\ell},\ \ldots,\ x_{\ell,2^\ell+3})\,. \tag{10}$$

The new interpolation domain $D^3_{\xi^{\text{nak}}} = [x_{\ell,0}, x_{\ell,2^\ell}]$ is now the whole unit interval [0, 1], containing all grid points at which we interpolate. As a result, the not-a-knot B-spline functions

$$\varphi^{\text{nak}}_{\ell,i} := b^3_{i,\xi^{\text{nak}}}, \quad i = 0, \ldots, 2^\ell\,, \tag{11}$$

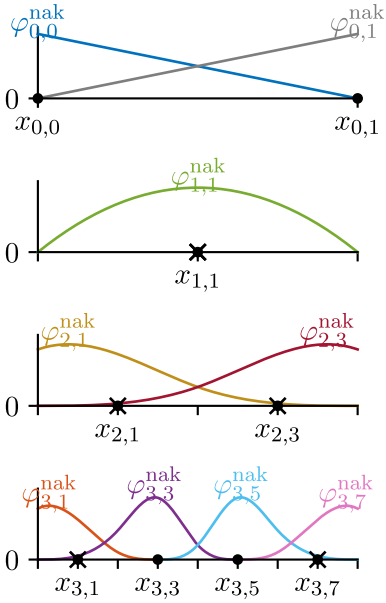


Fig. 2 Hierarchical not-a-knot B-spline basis in 1D up to level 3

form a basis of the spline space on $[0, 1]$ corresponding to the grid $\{x_{\ell,i} \mid i = 0, \dots, 2^\ell\} \backslash \{x_{\ell,1}, x_{\ell,2^\ell-1}\}$ and, consequently, the not-a-knot basis is able to reproduce all polynomials of degree smaller or equal to p on $[0, 1]$, see Fig. 1b, Höllig and Hörner (2013), and Valentin (2019). Note that the removal of the two grid points $x_{\ell,1}$ and $x_{\ell,2^\ell-1}$ is only possible if $\ell \geq 2$. If $\ell = 0$ or $\ell = 1$, we define $\varphi^{\text{nak}}_{\ell,i}$ as the Lagrange polynomial corresponding to the data $\{(i', \delta_{i,i'}) \mid i' = 0, \dots, 2^\ell\}$ where $\delta_{i,i'}$ is the Kronecker delta, e.g.,

$$\varphi^{\text{nak}}_{0,0}(x) := 1 - x\,, \quad \varphi^{\text{nak}}_{0,1}(x) := x\,, \quad \varphi^{\text{nak}}_{1,1}(x) := 4x(x-1)\,. \tag{12}$$

3.2 Hierarchical Not-A-Knot B-Splines

The construction of sparse grids needs a hierarchical splitting of the nodal basis. Therefore, we define the so-called nodal subspaces V^{nak}_ℓ and hierarchical subspaces W^{nak}_ℓ by

$$V^{\text{nak}}_\ell := \operatorname{span}\{\varphi^{\text{nak}}_{\ell,i} \mid i = 0, \dots, 2^\ell\}\,, \quad W^{\text{nak}}_\ell := \operatorname{span}\{\varphi^{\text{nak}}_{\ell,i} \mid i \in I_\ell\}\,, \tag{13}$$

where

$$I_\ell := \begin{cases} \{i = 1, \dots, 2^\ell - 1 \mid i \text{ odd}\} & \ell > 0, \\ \{0, 1\} & \ell = 0. \end{cases} \tag{14}$$

The bases of the hierarchical subspaces are shown in Fig. 2.

It can be shown that the basis functions of W_ℓ^{nak} for $\ell = 0, \dots, n$ and $n \in \mathbb{N}_0$ are linearly independent on [0, 1] (Valentin 2019). This means that the sum $\text{span}\{\varphi_{\ell,i}^{\text{nak}} \mid \ell = 0, \dots, n,\ i \in I_\ell\}$ of the subspaces $W_0^{\text{nak}}, \dots, W_n^{\text{nak}}$ is direct (i.e., $W_\ell^{\text{nak}} \cap W_{\ell'}^{\text{nak}} = \{0\}$ for $\ell \neq \ell'$) and can be written as $\bigoplus_{\ell=0}^{n} W_\ell^{\text{nak}}$. Due to dimensional arguments, the direct sum coincides with the nodal space,

$$\bigoplus_{\ell=0}^{n} W_\ell^{\text{nak}} = V_n^{\text{nak}}. \tag{15}$$

Both sides are equal to the not-a-knot spline space described before.

3.3 Sparse Grids

We generalize the univariate hierarchical basis to d-variate functions with a tensor product approach:

$$\varphi_{\boldsymbol{\ell},\boldsymbol{i}}^{\text{nak}} : [0,1]^d \to \mathbb{R}, \quad \varphi_{\boldsymbol{\ell},\boldsymbol{i}}^{\text{nak}}(\boldsymbol{x}) := \prod_{t=1}^{d} \varphi_{\ell_t,i_t}^{\text{nak}}(x_t), \tag{16}$$

where level and index are multi-indices $\boldsymbol{\ell} = (\ell_1, \dots, \ell_d) \in \mathbb{N}_0^d$ and $\boldsymbol{i} = (i_1, \dots, i_d)$ with $i_t \in \{0, \dots, 2^{\ell_t}\}$ for $t = 1, \dots, d$. The corresponding grid points are given by

$$\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} := (x_{\ell_1,i_1}, \dots, x_{\ell_d,i_d}) \in [0,1]^d, \tag{17}$$

and nodal and hierarchical subspaces are defined by

$$V_{\boldsymbol{\ell}}^{\text{nak}} := \text{span}\{\varphi_{\boldsymbol{\ell},\boldsymbol{i}}^{\text{nak}} \mid \boldsymbol{0} \le \boldsymbol{i} \le \boldsymbol{2^n}\}, \quad W_{\boldsymbol{\ell}}^{\text{nak}} := \text{span}\{\varphi_{\boldsymbol{\ell},\boldsymbol{i}}^{\text{nak}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}, \tag{18}$$

where $\boldsymbol{0} \le \boldsymbol{i} \le \boldsymbol{2^n}$ is to be read component-wise ($0 \le i_t \le 2^{n_t}$ for all $t = 1, \dots, d$) and $I_{\boldsymbol{\ell}} = I_{\ell_1} \times \cdots \times I_{\ell_d}$ with the Cartesian product $\times$. The nodal subspace of level $\boldsymbol{n}$ can be split into hierarchical subspaces by the d-dimensional generalization of Eq. (15):

$$\bigoplus_{\boldsymbol{\ell} \le \boldsymbol{n}} W_{\boldsymbol{\ell}}^{\text{nak}} = V_{\boldsymbol{n}}^{\text{nak}}, \tag{19}$$

where $\boldsymbol{n} \in \mathbb{N}_0^d$. In the following, we assume that the level is equal for every dimension: $\boldsymbol{n} := (n, \dots, n) = n \cdot \boldsymbol{1}$.

Sparse grids provide a method for the interpolation of objective functions $f : [0,1]^d \to \mathbb{R}$. The common approach is to use the nodal space $V_{\boldsymbol{n}}^{\text{nak}}$ for interpolation. However, the corresponding full grid of level n,

$$\{\boldsymbol{x}_{\boldsymbol{n},\boldsymbol{i}} \mid \boldsymbol{0} \le \boldsymbol{i} \le \boldsymbol{2^n}\}, \tag{20}$$

contains $(2^n + 1)^d = \Omega(2^{nd})$ grid points. If we interpolated with $V_{\boldsymbol{n}}^{\text{nak}}$, we would have to evaluate the objective function $\Omega(2^{nd})$ times, a number that grows exponentially with the dimensionality d. This fact is known as the curse of dimensionality (Bellman 1961). If evaluations of the objective function are computationally expensive, then dimensionalities of $d \ge 4$ usually prohibit full grid approaches. For

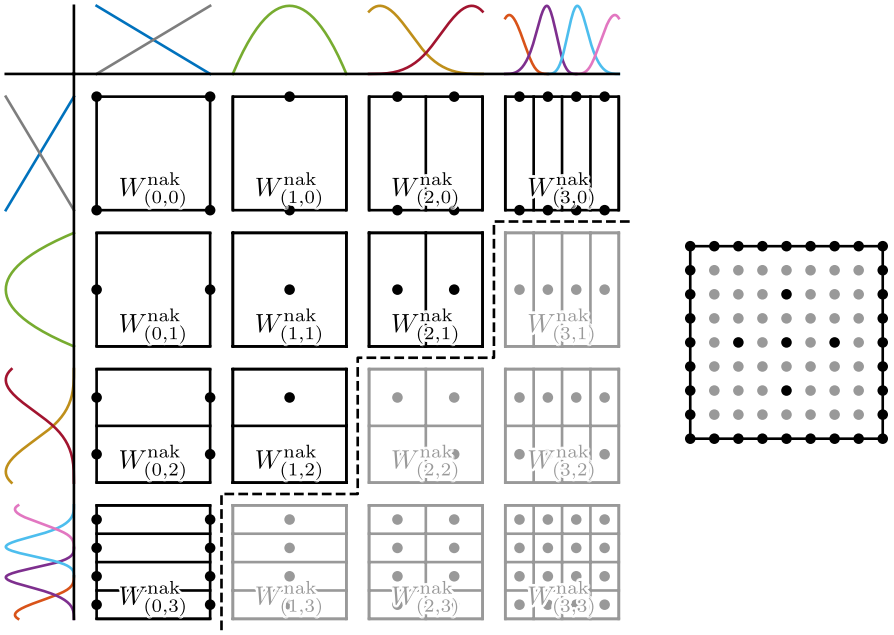


Fig. 3 Selection of hierarchical subspaces for the regular sparse grid of level $n = 3$ in 2D; the gray subspaces are omitted. For each subspace $W^{\text{nak}}_{\boldsymbol{\ell}}$, the grid points $\{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \boldsymbol{i} \in I_{\boldsymbol{\ell}}\} \in W^{\text{nak}}_{\boldsymbol{\ell}}$ are shown. The rectangular regions are the supports for the piecewise linear case; for the cubic case, they can be seen as a rough indication of the support sizes. To the left and top, the one-dimensional not-a-knot B-spline functions are plotted. The resulting grid is shown on the right

dynamic portfolio choice models, this means that only very coarse full grids may be employed in the state space if the number d of state variables is large.

Sparse grids exploit the splitting (19) to select only some hierarchical subspaces such that the number of necessary evaluations for interpolation is drastically reduced, but the interpolation error deteriorates only slightly. The subspace selection can be formulated as an optimization problem for the piecewise linear case ($p = 1$) on the L^2 and L^∞ interpolation error as detailed by Bungartz and Griebel (2004). The basic idea is to select those hierarchical subspaces that contribute most to the interpolation, assuming the objective function is sufficiently smooth. The optimal a priori selection for hat functions is given by the regular sparse grid of level n:

$$V_n^{\text{S,nak}} := \bigoplus_{\|\boldsymbol{\ell}\|_1 \le n} W^{\text{nak}}_{\boldsymbol{\ell}}, \tag{21}$$

where the 1-norm $\|\cdot\|_1$ is given by $\|\boldsymbol{\ell}\|_1 = \sum_{t=1}^{d} |\ell_t|$. This definition is illustrated in Fig. 3. It can be seen as an analogue to Eq. (19), which we obtain by replacing the 1-norm on the right-hand side of Eq. (21) with the ∞-norm $\|\boldsymbol{\ell}\|_\infty := \max\{|\ell_t| \mid t = 1, \ldots, d\}$.

Although the definition is motivated by the piecewise linear case ($p = 1$), using other basis functions such as higher-order B-splines has proven useful in various

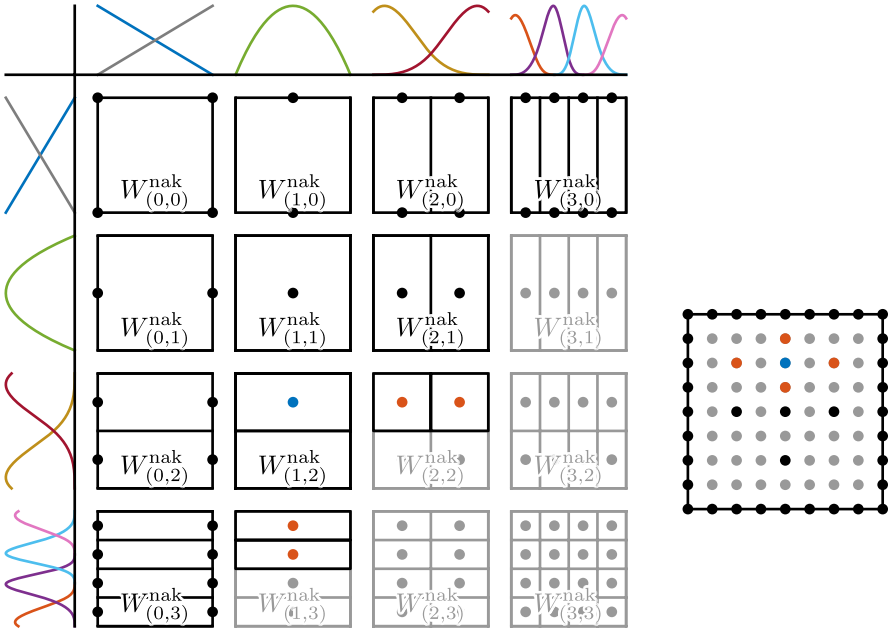


Fig. 4 Spatially adaptive refinement of the hierarchical subspace $W_{(1,2)}$ for the regular sparse grid of level $n = 3$ in 2D. A refinable grid point (blue) is refined by adding its $2d = 4$ children (red). The resulting grid is shown on the right

applications (Pflüger 2010; Valentin and Pflüger 2016; Valentin et al. 2018). For $p = 1$ and homogeneous boundary conditions, the L^2 interpolation error on the full grid of level n is given by $\mathcal{O}(2^{-2n})$ and the number of grid points (i.e., the required function evaluations) is $\mathcal{O}(2^{nd})$. In contrast, the sparse grid L^2 interpolation error is $\mathcal{O}(2^{-2n}n^{d-1})$ and therefore only slightly worse (by a factor which is polynomial in n) while requiring $\mathcal{O}(2^n n^{d-1})$ grid points (see the proof in Bungartz and Griebel 2004). The number of grid points does not depend on $2nd$ anymore, which means that significantly less grid points than in the full grid case are required. For B-splines of degree p, the interpolation error has been proven by Sickel and Ullrich (2011) to be in the order of $\mathcal{O}(2^{-(p+1)n}n^{d-1})$, which differs from the corresponding full grid error $\mathcal{O}(2^{-(p+1)n})$ (see Höllig and Hörner 2013 for a proof) by the same polynomial factor n^{d-1}.

These a priori estimates are based on the assumption that the interpolated function has continuous mixed second derivatives. If this is not the case or if it contains oscillations with high frequencies, then spatial adaptivity must be employed. Therefore, grid points are refined a posteriori according to suitable refinement criteria, see Fig. 4. This is of particular importance for the scope of this paper as spatial adaptivity enables us to increase the accuracy in regions of interest while simultaneously keeping the number of grid points at an acceptable level. The idea of the common surplus-based refinement criterion is that in the piecewise linear basis, the *hierarchical surpluses* correspond to the integral of the mixed second derivative of the interpolated function (Bungartz and Griebel 2004). If the absolute value $|\alpha_{\boldsymbol{\ell},\boldsymbol{i}}|$ of the hierarchical surplus of a grid point $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$

is larger than a certain tolerance ϵ, then the $2d$ children of $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ are inserted to improve the accuracy of the interpolation in the proximity of $\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}}$ (Pflüger 2012). This criterion is only motivated for piecewise linear basis functions. Therefore, and for reasons of complexity, we use the piecewise linear basis to determine the grid points to be refined, and interpolate with the B-spline basis on the refined grid.

3.4 Weakly Fundamental Not-A-Knot Splines

Let $\Omega^{\mathrm{S}} \subset [0,1]^d$ be the set of grid points of the sparse grid, for example $\Omega^{\mathrm{S}} = \{\boldsymbol{x}_{\boldsymbol{\ell},\boldsymbol{i}} \mid \|\boldsymbol{\ell}\|_1 \leq n\,,\ \boldsymbol{i} \in I_{\boldsymbol{\ell}}\}$ for the regular sparse grid of level n (but dimensionally or spatially adaptive sparse grids are possible as well). In this setting, the task of interpolation is usually called *hierarchization* for the basis functions $\varphi_{\boldsymbol{\ell},\boldsymbol{i}}$. The resulting coefficients $\alpha_{\boldsymbol{\ell},\boldsymbol{i}}$ for Eq. (8) are the hierarchical surpluses.

Conventional B-spline bases, such as the not-a-knot B-splines described before, share the drawback that the hierarchization is in general computationally expensive. In the case of the common piecewise linear basis ($p = 1$), the hierarchical surpluses can be calculated in $\mathcal{O}(|\Omega^{\mathrm{S}}| \cdot d)$ time with the so-called unidirectional principle (Pflüger 2010). For B-splines, usually the solution of a linear system with $|\Omega^{\mathrm{S}}|$ unknowns is required, which is generally much slower as this needs $\mathcal{O}(|\Omega^{\mathrm{S}}|^3)$ time (where, e.g., $|\Omega^{\mathrm{S}}| = \mathcal{O}(2^n n^{d-1})$ for regular sparse grids of level n if we omit points on the boundary).

In one dimension, the reason is the additional couplings between the basis functions of different levels introduced by the wider support of the cubic B-splines compared to the piecewise linear functions. To mitigate this issue, we linearly combine as few neighboring nodal not-a-knot B-splines as possible such that the resulting combination $\varphi_{l,i}^{\mathrm{wfnak}}$ satisfies

$$\varphi_{l,i}^{\mathrm{wfnak}}(x_{k,j}) = 0 \quad \text{for all } k < l \text{ and } j \in I_k\,, \tag{22}$$

which we call the *weakly fundamental property*. The resulting basis functions are plotted in Fig. 5. This enables the efficient unidirectional principle for the hierarchization with the resulting weakly fundamental not-a-knot spline basis if specific points are inserted beforehand (for details, see Valentin 2019). Therefore, we use weakly fundamental not-a-knot splines on sparse grids for the rest of the paper.

4 B-Splines on Spatially Adaptive Sparse Grids for Dynamic Portfolio Choice Models

To solve the Bellman problem (3) numerically, we compute the value function interpolants (5) by solving

$$j_t^{\mathrm{S}}(\boldsymbol{x}_t^{(k)}) = \max_{\boldsymbol{p}_t} \left\{\tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)})\right\} \tag{23}$$

at all grid points $\boldsymbol{x}_t^{(k)}$ $(k = 1, \ldots, N_t)$, using higher-order B-splines on sparse grids for j_{t+1}^{S} in the right-hand side of target function (6). This basis choice readily provides

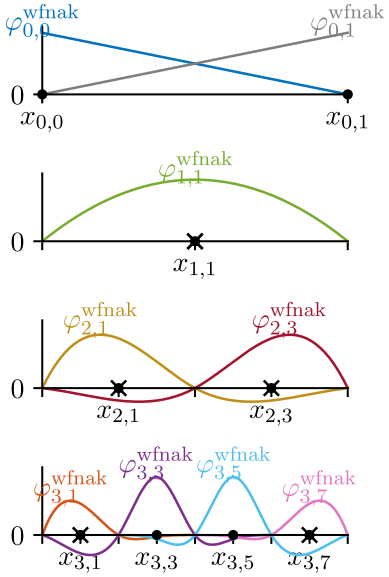


Fig. 5 Hierarchical weakly fundamental not-a-knot spline basis in 1D up to level 3

the gradient of the target function (6) at each $\boldsymbol{x}_t^{(k)}$, such that we can supply it to any SQP routine. As a result of the SQP optimization, we obtain the values of the interpolant $j_t^{\mathrm{S}}(\boldsymbol{x}_t^{(k)})$ and the optimal policies $\boldsymbol{p}_t^{\mathrm{opt,S}}(\boldsymbol{x}_t^{(k)})$ at these grid points for all $t < T$ and $k = 1, \ldots, N_t$:

$$\boldsymbol{p}_t^{\mathrm{opt,S}}(\boldsymbol{x}_t^{(k)}) = \underset{\boldsymbol{p}_t}{\mathbf{argmax}} \left\{ \tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}) \right\}. \tag{24}$$

In general, the shapes of the value function and the optimal policies have different characteristics. The sufficiently accurate resolution of the optimal policy is already relevant to achieve plausible economic results if full grid solutions are computed (Brumm and Grill 2014). On sparse grids, this is even more important as kinks in the optimal policies can deteriorate the numerical error drastically (Schober 2018). Hence, as proposed by Schober (2018), in a subsequent step, optimal policy interpolants $\boldsymbol{p}_t^{\mathrm{opt,S}}$ are computed by adaptively refining the respective policy grids and re-optimizing for the refined grid points if the added grid point is not yet part of the solution from the first step.

We track two interpolants $j_t^{\mathrm{S},1}$ and $j_t^{\mathrm{S},p}$ for each $t = 0, \ldots, T$. The former interpolates the value function data at the grid points $\boldsymbol{x}_t^{(k)}$ $(k = 1, \ldots, N_t)$ with the hierarchical piecewise linear basis (used for the surplus-based grid generation) while the latter interpolates the data with cubic hierarchical weakly fundamental not-a-knot splines of degree $p = 3$. Each $j_t^{\mathrm{S},*}$ $(* \in \{1, p\})$ additionally stores the grid points $\boldsymbol{x}_t^{(k)}$ and the optimal policies $\boldsymbol{p}_t^{\mathrm{opt}}(\boldsymbol{x}_t^{(k)})$ at the grid points. For simplicity, we do not pass them explicitly to the algorithms.

In the following Sects. 4.1–4.3, we describe the algorithmic details of the generation of the value function interpolant (23). The generation of the optimal policy

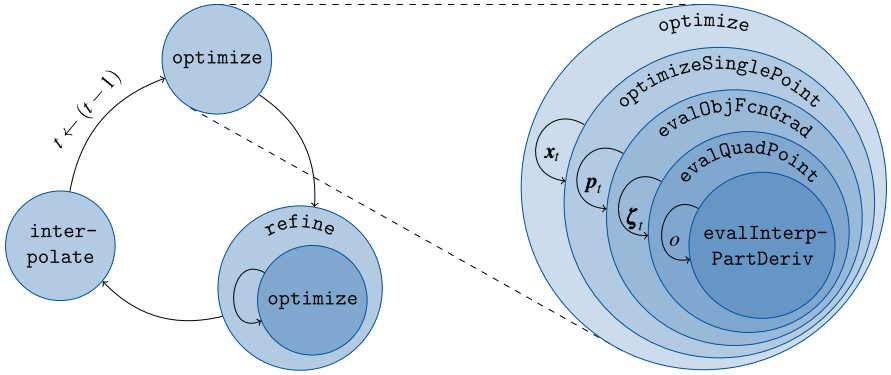


Fig. 6 Scheme of the generation of value function interpolants with `solveValueFunction` (left, Algorithm 1), which repeatedly calls the `optimize` algorithm (right, Algorithm 2), which in turn consists of various sub-functions. The function `optimize` iterates over all state grid points $\boldsymbol{x}_t = \boldsymbol{x}_t^{(k)}$ ($k = 1, \ldots, N_t$) and calls `optimizeSinglePoint` for each point. The optimization method evaluates the objective function and its gradient at a sequence of different policy points $\boldsymbol{p}_t$ to find $\boldsymbol{p}_t^{\text{opt,S}}(\boldsymbol{x}_t^{(k)})$. This evaluation (denoted by `evalObjFcnGrad`) has to implicitly compute the expectation in Eq. (23), which is done using a quadrature rule. For every quadrature point $\boldsymbol{\zeta}_t = \boldsymbol{\zeta}_t^{(j)}$ ($j = 1, \ldots, Q_t$), `evalQuadPoint` computes the corresponding value of the expression in the expectation. Finally, `evalInterpPartDeriv` evaluates the interpolant $j_{t+1}^{S,p}$ and its partial derivatives for which we have to loop over the state dimensions $o = 1, \ldots, d$

interpolant (24) follows in Sect. 4.4. Major parts from the Sects. 4.1–4.4 are taken from our previous work in the recently submitted Ph.D. thesis of Valentin (2019).

4.1 Solution for the Value Function

Algorithm 1 shows `solveValueFunction`, generating 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 presented in Fig. 6: First, Eq. (23) is solved on an initial sparse grid (`optimize`). Then, we `refine` the grid spatially adaptively. Finally, the resulting grid 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 $t + 1$. However, the results we then obtain become worse, likely due to the different characteristics of $j_t^{S,1}$ for different t (e.g., kinks).

4.2 Optimization

The `optimize` step can be seen in Algorithm 2. This algorithm accepts in $j_t^{S,1}$ a 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 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`.[3] Otherwise, for $t < T$, we solve the maximization problem (23) by using the higher-order B-spline interpolant $j_{t+1}^{\mathrm{S},p}$ 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 (Cai et al. 2015; Horneff et al. 2016).[4] After generating all missing data, we update the hierarchical surpluses of the piecewise linear interpolant $j_t^{\mathrm{S},1}$ to interpolate the new data at all grid points of Ω_t^{S}.

Algorithm 1 Generation of value function interpolants. The output is the higher-order B-spline interpolant $j_t^{\mathrm{S},p}$ for all $t = 0, \ldots, T$.

function $(j_t^{\mathrm{S},p})_{t=0,\ldots,T}$ = `solveValueFunction`()
  $j_{T+1}^{\mathrm{S},p} \leftarrow \emptyset$ ⇝ *dummy variable (value is not used)*
  for $t = T, T-1, \ldots, 0$ **do**
    $j_t^{\mathrm{S},1} \leftarrow$ Initial regular sparse grid with no values
    $j_t^{\mathrm{S},1} \leftarrow$ `optimize`$(t,\ j_t^{\mathrm{S},1},\ j_{t+1}^{\mathrm{S},p})$
    $j_t^{\mathrm{S},1} \leftarrow$ `refine`$(t,\ j_t^{\mathrm{S},1},\ j_{t+1}^{\mathrm{S},p})$
    $j_t^{\mathrm{S},p} \leftarrow$ `interpolate`$(j_t^{\mathrm{S},1})$

Algorithm 2 Computation of the value function at all grid points $\boldsymbol{x}_t^{(k)}$ of $j_t^{\mathrm{S},1}$ at which the value function values have not been computed yet. Inputs are the time t, the piecewise linear interpolant $j_t^{\mathrm{S},1}$ of the current iteration t (with the underlying sparse grid and possibly set value function values when `optimize` is called from within `refine`), and the higher-order B-spline interpolant $j_{t+1}^{\mathrm{S},p}$ of the previous iteration $t+1$ (not used if $t = T$). The output is the updated piecewise linear interpolant $j_t^{\mathrm{S},1}$ where all missing function values at grid points have been computed.

function $j_t^{\mathrm{S},1}$ = `optimize`$(t, j_t^{\mathrm{S},1}, j_{t+1}^{\mathrm{S},p})$
  $(\boldsymbol{x}_t^{(k)})_{k=1,\ldots,N_t} \leftarrow$ grid of $j_t^{\mathrm{S},1}$
  for $k = 1, \ldots, N_t$ **do**
    if $j_t^{\mathrm{S},1}(\boldsymbol{x}_t^{(k)})$ not previously computed **then**
      if $t = T$ **then** $j_T^{\mathrm{S},1}(\boldsymbol{x}_T^{(k)}) \leftarrow$ `computeKnownTerminalSolution`$(\boldsymbol{x}_T^{(k)})$
      else $j_t^{\mathrm{S},1}(\boldsymbol{x}_t^{(k)}) \leftarrow$ `optimizeSinglePoint`$(t,\ \boldsymbol{x}_t^{(k)},\ j_{t+1}^{\mathrm{S},p})$
  Re-interpolate $(j_t^{\mathrm{S},1}(\boldsymbol{x}_t^{(k)}))_{k=1,\ldots,N_t}$ with piecewise linear functions

4.3 Refinement

For adaptive refinement, the criterion is the common surplus-volume (Pflüger 2010). We use the piecewise linear interpolant for the surplus-based grid generation as the surpluses are easier to compute in the piecewise linear case, and as they are more meaningful due to the integral representation formula (Bungartz and Griebel

[3] In any case, the terminal solution may be computed as the solution of the corresponding single-time optimization problem, e.g., $j_T(\boldsymbol{x}_T^{(k)}) = \max_{\boldsymbol{p}_T} \{u(c_T(\boldsymbol{x}_T^{(k)}, \boldsymbol{p}_T))\}$.

[4] Such a problem is usually referred to as *embarrassingly parallel.*

2004). Algorithm 3 shows how to generate the spatially adaptive sparse grid in `solveValueFunction` (Algorithm 1). Parameters are the tolerance $\varepsilon \in \mathbb{R}_{\geq 0}$ by which the set of grid points to be refined is determined and the number $q \in \mathbb{N}_0$ of refinement iterations.

Algorithm 3 In-place refinement of the value function $j_t^{\mathrm{S},1}$. Inputs are the time t, the piecewise linear interpolant $j_t^{\mathrm{S},1}$ of the current iteration t, and the higher-order B-spline interpolant $j_{t+1}^{\mathrm{S},p}$ of the previous iteration $t+1$ (not used if $t = T$). The output is the updated piecewise linear interpolant $j_t^{\mathrm{S},1}$ with the refined sparse grid.

function $j_t^{\mathrm{S},1} =$ `refine`$(t, j_t^{\mathrm{S},1}, j_{t+1}^{\mathrm{S},p})$
  for $j = 1, \ldots, q$ **do**
    $N_t \leftarrow$ number of grid points of $j_t^{\mathrm{S},1}$
    for $k = 1, \ldots, N_t$ **do** $\alpha_t^{(k)} \leftarrow$ surplus of $\boldsymbol{x}_t^{(k)}$ in $j_t^{\mathrm{S},1}$
    $K_{\mathrm{refine}} \leftarrow \left\{k = 1, \ldots, N_t \;\middle|\; 2^{-\|\boldsymbol{\ell}\|_1} |\alpha_t^{(k)}| \geq \varepsilon\right\}$
    if $K_{\mathrm{refine}} = \emptyset$ **then break**
    Refine all grid points in $\{\boldsymbol{x}_t^{(k)} \mid k \in K_{\mathrm{refine}}\}$
    $j_t^{\mathrm{S},1} \leftarrow$ `optimize`$(t,\ j_t^{\mathrm{S},1},\ j_{t+1}^{\mathrm{S},p})$

4.4 Solution for the Optimal Policies

To construct the optimal policies, we use the higher-order B-spline interpolant $j_t^{\mathrm{S},p}$ and the optimal policies $\boldsymbol{p}_t^{\mathrm{opt}}(\boldsymbol{x}_t^{(k)})$ at the grid points $\boldsymbol{x}_t^{(k)}$ $(k = 1, \ldots, N_t)$ obtained from Algorithm 1. We then spatially adaptively refine the grid for each policy to construct a policy interpolant of degree one, $\boldsymbol{p}_t^{\mathrm{opt,S},1}$, for each $t = 1, \ldots, T$. The corresponding Algorithm 1 is similar to `solveValueFunction` (Algorithm 4), except that it operates on the policy interpolants instead of the value function interpolant. The functions `optimize` and `refine` have been replaced by corresponding policy versions `optimizePolicy` and `refinePolicy` that work very much like their value function counterpart. In the optimization step, `optimizePolicy` only has to generate new values if the initial regular sparse grid for the policies is not contained in the grid of $j_t^{\mathrm{S},p}$. The policy grid is then refined independently of the value function grid. The iterations are independent of each other, which means that they can be parallelized.[5]

[5] In principle, one could generate policy interpolants of degree p, $\boldsymbol{p}_t^{\mathrm{opt,S},p}$, by adding an extra interpolation step after `refinePolicy` (Valentin 2019).

Algorithm 4 Generation of interpolants for optimal policies. The input is the higher-order B-spline interpolant $j_t^{\mathrm{S},p}$ of the value function for all $t = 0, \ldots, T$. The output is the piecewise linear interpolant $\boldsymbol{p}_t^{\mathrm{opt,S,1}}$ of the optimal policies for all $t = 0, \ldots, T$.

function $(\boldsymbol{p}_t^{\mathrm{opt,S,1}})_{t=0,\ldots,T} = \texttt{solvePolicies}((j_t^{\mathrm{S},p})_{t=0,\ldots,T})$
  $j_{T+1}^{\mathrm{S},p} \leftarrow \emptyset$ ⇝ *dummy variable (value is not used)*
  for $t = 0, \ldots, T$ **do**
    $\boldsymbol{p}_t^{\mathrm{opt,S,1}} \leftarrow$ Initial regular sparse grid, retrieve values from $j_t^{\mathrm{S},p}$
    $\boldsymbol{p}_t^{\mathrm{opt,S,1}} \leftarrow \texttt{optimizePolicy}(t,\ \boldsymbol{p}_t^{\mathrm{opt,S,1}},\ j_{t+1}^{\mathrm{S},p})$
    $\boldsymbol{p}_t^{\mathrm{opt,S,1}} \leftarrow \texttt{refinePolicy}(t,\ \boldsymbol{p}_t^{\mathrm{opt,S,1}},\ j_{t+1}^{\mathrm{S},p})$

5 Application: Transaction Costs Problem

First, we introduce the dynamic portfolio choice model with transaction costs (Sect. 5.1). We then describe its numerical solution in Sect. 5.2 and derive our error measure (unit-free Euler equation errors) in Sect. 5.3. We verify our solution economically on a two-dimensional full grid (Sect. 5.4). Then, we analyze the time complexity of the solution approach and show the impact of the choice of basis functions on the computational complexity in Sect. 5.5. Therefore, we solve the problem with B-splines of cubic degree on a regular sparse grid and compare them to the linear approach as used by Brumm and Scheidegger (2017). We find that we save approximately one order of magnitude in computational complexity with cubic B-splines already for three dimensions. Furthermore, we compare the results of our approach based on analytical gradients with the results we obtain with finite differences. For $d > 3$, solutions on regular sparse grids are no longer feasible to compute in suitable numerical accuracy. In Sect. 5.6, we illustrate how spatial adaptivity allows us to solve the transaction costs problem up to $d = 5$ accurately by showing pointwise error decay and convergence of our approach. Finally, we present in Sect. 5.7 economic results for the transaction costs problem in higher dimensions. The results in these sections are also contained in the recently submitted Ph.D. thesis of Valentin (2019).

5.1 Transaction Costs Problem

As the transaction costs problem is easiest described in vector notation, let us denote the unit vector with $\mathbf{1}$. For two vectors $\boldsymbol{a}$, $\boldsymbol{b}$, we define the Hadamard product as $(\boldsymbol{a} \odot \boldsymbol{b})_i := a_i b_i$ for all i.

In the transaction costs problem the investor maximizes expected utility from consumption (1). Therefore, at time t, she tracks her wealth $W_t \in \mathbb{R}_{\geq 0}$ and fractions of wealth $\boldsymbol{x}_t \in [0,1]^d$ invested in stocks. Her choices are how much to buy of the d stocks $\boldsymbol{\Delta}_t^+ \in \mathbb{R}_{\geq 0}^d$ with transaction costs $\tau \boldsymbol{\Delta}_t^+$ or sell $\boldsymbol{\Delta}_t^- \in \mathbb{R}_{\geq 0}^d$ with transaction costs $\tau \boldsymbol{\Delta}_t^-$ where $\tau > 0$ is a cost factor. Additionally, she can invest in a transaction-cost-free money market account B_t, yielding a risk-free return $r_f \in \mathbb{R}_{\geq 0}$. We assume the returns $\boldsymbol{r}_t \in \mathbb{R}_{\geq 0}$ that the d stocks earn from t to $t+1$ are independent and identically

lognormally distributed with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$: $\boldsymbol{r}_t \sim LN(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ (Cai 2009; Cai and Judd 2010). The investor's consumption C_t in period t is the residual of her wealth that is not invested in stocks or bonds, reduced by the transaction costs for rearranging her portfolio in this period:

$$C_t = \left(1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t\right) W_t - B_t - (1+\tau)\mathbf{1}^\top \cdot \boldsymbol{\Delta}_t^+ - (\tau - 1)\mathbf{1}^\top \cdot \boldsymbol{\Delta}_t^- . \tag{25}$$

The state dynamics from t to $t+1$ are thus given by:

$$W_{t+1} = B_t r_f + \left(\boldsymbol{x}_t W_t + \boldsymbol{\Delta}_t^+ - \boldsymbol{\Delta}_t^-\right)^\top \cdot \boldsymbol{r}_t , \tag{26a}$$

$$\boldsymbol{x}_{t+1} = \frac{\left(\boldsymbol{x}_t W_t + \boldsymbol{\Delta}_t^+ - \boldsymbol{\Delta}_t^-\right) \odot \boldsymbol{r}_t}{W_{t+1}} . \tag{26b}$$

The investor faces the optimization problem

$$J_t(W_t, \boldsymbol{x}_t) = \max_{B_t, \boldsymbol{\Delta}_t^+, \boldsymbol{\Delta}_t^-} \left\{ u(C_t) + \rho \mathbb{E}_t \left[J_{t+1}\left(W_{t+1}, \boldsymbol{x}_{t+1}\right) \right] \right\} , \quad t < T , \tag{27a}$$

$$J_T(W_T, \boldsymbol{x}_T) = u\left(\left(1 - \tau \mathbf{1}^\top \cdot \boldsymbol{x}_T\right) W_T\right) , \tag{27b}$$

with utility function u from Eq. (2) subject to the constraint for all $t = 0, \ldots, T$,

$$B_t + (1+\tau)\mathbf{1}^\top \cdot \boldsymbol{\Delta}_t^+ + (\tau - 1)\mathbf{1}^\top \cdot \boldsymbol{\Delta}_t^- \leq \left(1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t\right) W_t - C_{\min} , \tag{27c}$$

where $\boldsymbol{\Delta}_t^+ \geq \mathbf{0}$, $\boldsymbol{\Delta}_t^- \in [\mathbf{0}, \boldsymbol{x}_t W_t]$, $B_t \geq 0$, and $\mathbf{1}^\top \cdot \boldsymbol{x}_t \leq 1$. Here, a minimum consumption level $C_{\min}$ must be maintained, and the final stock holdings $\boldsymbol{x}_T W_T$ are assumed to be sold before they can be consumed. In addition, at no point in time t the investor can sell more of the stocks than her current holdings $\boldsymbol{x}_t W_t$.

The problem can be simplified by normalizing the value function $j_t = J_t / W_t$, consumption $c_t = C_t / W_t$, and investment choices $b_t = B_t / W_t$, $\boldsymbol{\delta}_t^+ = \boldsymbol{\Delta}_t^+ / W_t$, $\boldsymbol{\delta}_t^- = \boldsymbol{\Delta}_t^- / W_t$ with respect to wealth W_t for each t. The investor's normalized consumption c_t in period t is then

$$c_t = 1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t - b_t - (1+\tau)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^+ - (\tau - 1)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^- . \tag{28}$$

The state dynamics from t to $t+1$ can be expressed in terms of the portfolio value

$$\pi_{t+1} := b_t r_f + (\boldsymbol{x}_t + \boldsymbol{\delta}_t^+ - \boldsymbol{\delta}_t^-)^\top \cdot \boldsymbol{r}_t \tag{29}$$

in $t+1$:

$$W_{t+1} = W_t \pi_{t+1} , \tag{30a}$$

$$\boldsymbol{x}_{t+1} = \frac{\left(\boldsymbol{x}_t + \boldsymbol{\delta}_t^+ - \boldsymbol{\delta}_t^-\right) \odot \boldsymbol{r}_t}{\pi_{t+1}} . \tag{30b}$$

With this normalization, the solution to problem (27a–27c) can be expressed as $J_t = W_t^{1-\gamma} j_t$ for each $t = 0, \dots, T$ with

$$j_t(\boldsymbol{x}_t) = \max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ u(c_t) + \rho \mathbb{E}_t \left[\pi_{t+1}^{1-\gamma} j_{t+1}(\boldsymbol{x}_{t+1}) \right] \right\}, \quad t < T, \tag{31a}$$

$$j_T(\boldsymbol{x}_T) = u\left(1 - \tau \mathbf{1}^\top \cdot \boldsymbol{x}_T\right), \tag{31b}$$

subject to the constraints for all $t = 0, \dots, T$,

$$b_t + (1+\tau)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^+ + (\tau - 1)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^- \leq 1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t - c_{\min}, \tag{31c}$$

$$\boldsymbol{\delta}_t^+ \geq \mathbf{0}, \tag{31d}$$

$$\boldsymbol{\delta}_t^- \geq \mathbf{0}, \tag{31e}$$

$$\boldsymbol{\delta}_t^- \leq \boldsymbol{x}_t, \tag{31f}$$

$$b_t \geq 0, \tag{31g}$$

$$\mathbf{1}^\top \cdot \boldsymbol{x}_t \leq 1, \tag{31h}$$

where the minimum consumption level $c_{\min} = C_{\min}/W_t$ is also normalized with respect to wealth (see "Appendix A.2"). Now the investor's optimization problem no longer depends on W_t, and hence one state variable can be eliminated. The non-normalized optimal choices can be obtained by multiplication with a given wealth W_t for any t and state $\boldsymbol{x}_t$.

5.2 Numerical Solution

To compute the solution to the transaction costs problem, we use the certainty equivalent transformation $\hat{j}_t$ of the normalized value function j_t,

$$\hat{j}_t(\boldsymbol{x}_t) = \left((1-\gamma) j_t(\boldsymbol{x}_t)\right)^{\frac{1}{1-\gamma}}, \tag{32}$$

which reduces the curvature of the value function when the utility is of Constant Relative Risk Aversion type, Eq. (2) (Garlappi and Skoulakis 2009). Since this transform is strictly monotone, any maximizer of $\hat{j}_t$ also maximizes j_t. The optimization problem then reads

$$\hat{j}_t(\boldsymbol{x}_t) = \max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ \left(c_t^{1-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}(\boldsymbol{x}_{t+1}) \right)^{1-\gamma} \right] \right)^{\frac{1}{1-\gamma}} \right\}, \tag{33a}$$

$$\hat{j}_T(\boldsymbol{x}_T) = 1 - \tau \mathbf{1}^\top \cdot \boldsymbol{x}_T\,, \tag{33b}$$

with constraints Eq. (31c) to (31h) (see "Appendix A.3").

The optimization problem for a given state $\boldsymbol{x}_t$ is solved with the SQP solver SNOPT (Gill et al. 2005), see "Appendix A.4" for the specific objective function and its gradient. Since the distribution of the returns $\boldsymbol{r}_t$ is multivariate log-normal and state-independent, we can compute the expectation in Eq. (6) using Gauss–Hermite quadrature. For this, we also use a sparse grid quadrature rule, thus breaking the curse of dimensionality when including stochastic risk factors (see the appendix of Horneff et al. 2016 for details).

The constraint (31h) constrains the state space. The resulting eligible subspace $\Omega_{\text{Simplex}} = \{\boldsymbol{x}_t \in [0,1]^d \mid \mathbf{1}^\top \cdot \boldsymbol{x}_t \leq 1\} \subset [0,1]^d$ is a d-dimensional simplex, not a rectangular domain as needed for the sparse grid approximation. We solve this problem by assuming that any state attained that is not eligible is cropped to an eligible state by selling all stock holdings pro rata until all constraints are satisfied. That is, money is transferred from stocks to wealth, for which the proportionate transaction costs are deducted (see "Appendix A.5"). The approximation of the value function is then evaluated at this eligible state.

The optimization ran over an investment horizon of $T = 6$ years and had a fixed period length of 1 year. The risk aversion $\gamma = 3.5$, the risk-free rate $r_f = 4\%$, and the transaction costs factor $\tau = 1\%$ were taken from Cai and Judd (2010). We extended the return distribution parametrization of Cai and Judd (2010) to five dimensions:

$$\boldsymbol{\mu} := \begin{pmatrix} 0.0572 \\ 0.0638 \\ 0.07 \\ 0.0764 \\ 0.0828 \end{pmatrix}, \quad \boldsymbol{\Sigma} := 10^{-2} \begin{pmatrix} 2.56 & 0.576 & 0.288 & 0.176 & 0.096 \\ 0.576 & 3.24 & 0.90432 & 1.0692 & 1.296 \\ 0.288 & 0.90432 & 4 & 1.32 & 1.68 \\ 0.176 & 1.0692 & 1.32 & 4.84 & 2.112 \\ 0.096 & 1.296 & 1.68 & 2.112 & 5.76 \end{pmatrix}, \tag{34}$$

and set the time discount factor to $\rho = 0.97$ and $c_{\min} = 0.001$ to ensure that minimal consumption was taking place. For d stocks we used the first d entries of $\boldsymbol{\mu}$ and the elements $\boldsymbol{\Sigma}_{i,j}$, $i,j \leq d$, as the return distribution parametrization. As initial grids, we used regular sparse grids $\Omega^{\text{S}}_{n,d}$ of level n and dimension d.

The code was written in MATLAB where the interpolation on sparse and full grids was implemented by a MEX file interface to the sparse grids C++ toolbox SG^{++} (sgpp.sparsegrids.org, Pflüger 2010). The quadrature routine was implemented by a MEX file interface to the TASMANIAN sparse grids C++ toolbox (Stoyanov 2017) as TASMANIAN allows us to integrate a real valued function over a Gaussian density using Hermite polynomials on sparse grids (see the appendix of Horneff et al. 2016). We used the SNOPT implementation of the Numerical Algorithms Group (www.nag.co.uk). If convergence of the optimizer was not observed, we stopped the optimization after 100 iterations. 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 computations were performed on the compute cluster LOEWE-CSC (csc.uni-frankfurt.de) where we exclusively allocated three compute nodes with two Intel Xeon

E5-2670 v2 CPUs (ten cores at 2.5 GHz, 20 threads) each, i.e., 120 threads in total, and 4000 MB RAM per thread.

5.3 Error Measurement

Any optimal policy $\boldsymbol{p}_t^{\text{opt}} := (b_t^{\text{opt}}, \boldsymbol{\delta}_t^{+,\text{opt}}, \boldsymbol{\delta}_t^{-,\text{opt}})^\top$ must satisfy the first order conditions of the Lagrangian at any given state $\boldsymbol{x}_t$ for each $t < T$. Specifically, for the transaction costs problem—when neglecting binding constraints—we obtain from the first-order condition with regard to the optimal bond policy b_t^{opt}:

$$-c_t^{\text{opt}-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1}^{\text{opt}} \hat{j}_{t+1}^{\text{S}} \right)^{-\gamma} r_f \left(\hat{j}_{t+1}^{\text{S}} - \left(\nabla_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\text{S}} \right)^\top \cdot \boldsymbol{x}_{t+1}^{\text{opt}} \right) \right] = 0 \,, \tag{35}$$

where

$$c_t^{\text{opt}} = 1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t - b_t^{\text{opt}} - (1+\tau)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^{+,\text{opt}} - (\tau - 1)\mathbf{1}^\top \cdot \boldsymbol{\delta}_t^{-,\text{opt}} \,, \tag{36a}$$

$$\pi_{t+1}^{\text{opt}} = b_t^{\text{opt}} r_f + (\boldsymbol{x}_t + \boldsymbol{\delta}_t^{+,\text{opt}} - \boldsymbol{\delta}_t^{-,\text{opt}})^\top \cdot \boldsymbol{r}_t \,, \tag{36b}$$

$$\boldsymbol{x}_{t+1}^{\text{opt}} = \frac{\left(\boldsymbol{x}_t + \boldsymbol{\delta}_t^{+,\text{opt}} - \boldsymbol{\delta}_t^{-,\text{opt}} \right) \odot \boldsymbol{r}_t}{\pi_{t+1}^{\text{opt}}} \,. \tag{36c}$$

Rearranging Eq. (35) and taking the root $(\cdot)^{-1/\gamma}$ yields the *unit-free Euler equation error*,

$$\varepsilon_t(\boldsymbol{x}_t) = \left(\rho \mathbb{E}_t \left[\left(\pi_{t+1}^{\text{opt}} \hat{j}_{t+1}^{\text{S}} \right)^{-\gamma} r_f \left(\hat{j}_{t+1}^{\text{S}} - \left(\nabla_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\text{S}} \right)^\top \cdot \boldsymbol{x}_{t+1}^{\text{opt}} \right) c_t^{\text{opt}\gamma} \right] \right)^{-\frac{1}{\gamma}} - 1 \,, \tag{37}$$

which should be 0 for any given state $\boldsymbol{x}_t$ in the eligible domain $\boldsymbol{\Omega}_{\text{Simplex}}$.

However, the state space cropping distorts unit-free Euler equation errors. This is due to three sources: Firstly, the cropping already occurs for large stock holdings $\mathbf{1}^\top \cdot \boldsymbol{x}_t$ that are less than one as stocks have to be sold to maintain minimum consumption $c_{\min}$. Secondly, transaction costs for selling the stocks are deducted. Thirdly, even if neither minimum consumption is required, nor transaction costs are incurred the error at the hyperplane $\mathbf{1}^\top \cdot \boldsymbol{x}_t = 1$ does not vanish even for full grid solutions. Only in the limit, as the resolution of the grid goes to infinity, the error will vanish. Economically, the region near this hyperplane is not significant as such large stock fractions are unusual, which is confirmed by Monte Carlo simulations. We therefore use the *weighted Euler equation error*

$$\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}_t) := \left(1 - \mathbf{1}^\top \cdot \boldsymbol{x}_t\right)\varepsilon_t(\boldsymbol{x}_t) \tag{38}$$

instead of ε_t. Alternatives would be restricting the state domain in which the error is computed or weighting the error with the probability that a given state occurs in a Monte Carlo simulation.[6]

We then choose the same $N = 1000$ points $\boldsymbol{x}^{(k)} \in \Omega_{\text{Simplex}}$ $(k = 1, \ldots, N)$ for all times $t = 0, \ldots, T-1$ and compute the errors $\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}^{(k)})$ for each t.[7] We report the L^2 norm scaled by $\sqrt{d!}$ and the L^∞ norm for each t:

$$\varepsilon_t^{\mathrm{w},L^2} := \sqrt{\frac{1}{N}\sum_{k=1}^{N} |\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}^{(k)})|^2}\,, \tag{39a}$$

$$\varepsilon_t^{\mathrm{w},L^\infty} := \max\{|\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}^{(k)})| \mid k = 1, \ldots, N\}\,. \tag{39b}$$

For details on the error derivation see "Appendix A.6".

In principle, we could also compare the solutions $\hat{j}_t^{\mathrm{S}}$ and optimal policies $\boldsymbol{p}_t^{\mathrm{opt,S}}$ obtained on sparse grids with the full grid solution, e.g., in a point-wise way. However, full grid solutions with acceptable resolutions are computationally infeasible already in $d > 2$. In addition, the Euler equation error does not compare numerical solutions with each other, but rather measures the accuracy of any solution, regardless of whether it is obtained numerically or analytically.

5.4 Economical Verification

We show in Fig. 7 a full grid solution for the case of $d = 2$ stocks, i.e., $\{\boldsymbol{x}_t^{(k)} \mid k = 1, \ldots, N_t\} = \{0, 2^{-n}, \ldots, 1\}^d$ for some fixed level $n \in \mathbb{N}$ (here, $n = 7$ and $N_t = (2^7 + 1)^2 = 16641$) and for all $t = 0, \ldots, T$. The red dot $(x_{t,1}, x_{t,2}) = (0.1509, 0.1831)$ shows the so-called *Merton point*

$$\boldsymbol{x}_t^{\mathrm{opt}} := \frac{\boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu} - \mathbf{1}r_f)}{\gamma}\,, \tag{40}$$

for which Merton (1969) derives that in the case of $\tau = 0$ the optimal stock fractions $\boldsymbol{x}_t^{\mathrm{opt}}$ are constant over time and wealth. When faced with transaction costs, Magill and Constantinides (1976) find that the investor must weigh up the benefits of improved diversification against the associated transaction costs for rebalancing the portfolio. This leads to the *no-trade region* (red outline). If $\boldsymbol{x}_t^{\mathrm{opt}}$ lies within this region, the investor does not alter her portfolio. In discrete time consumption and portfolio choice, the no-trade region is known to be a convex set, and, if the current stock fraction is outside this region, the optimal policy is to move to the convex hull

[6] Another possibility is to transform the non-rectangular domain to the unit hypercube, e.g., by analyzing the principal components of the ergodic distribution (Judd et al. 2014).

[7] We choose $1000 \times d!$ points from $[0, 1]^d$ and discard all points that are not in Ω_{Simplex}. Here, the $d!$ corrects for the volume of the d-dimensional simplex obtained by cropping. Thus, there are only $N \approx 1000$ points in Ω_{Simplex}.

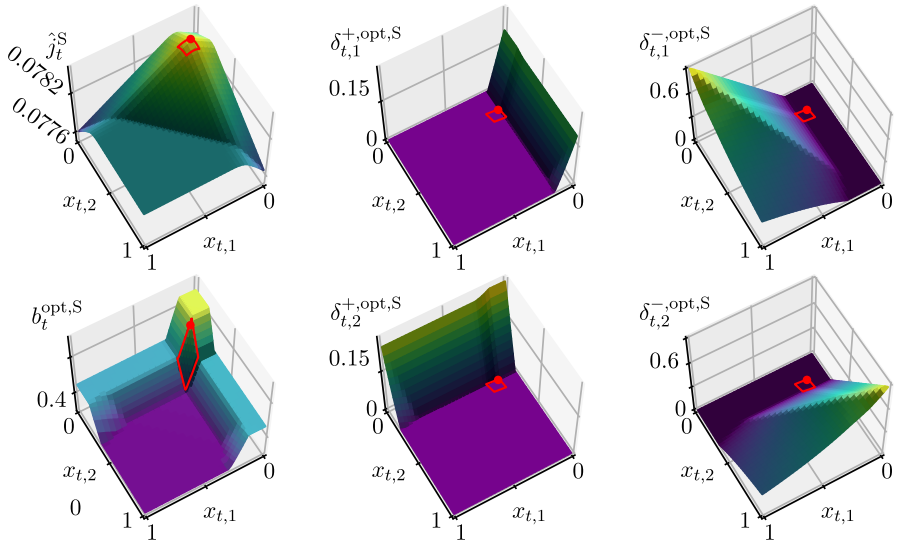


Fig. 7 Full grid solution for the transaction costs problem with $d = 2$ stocks. Shown are the certainty equivalent value function $\hat{j}_t^{\mathrm{S}}(\boldsymbol{x}_t)$ (top left) and the optimal policies $\boldsymbol{p}_t^{\mathrm{opt,S}}(\boldsymbol{x}_t)$ for $t = 0$. Also shown are the Merton point $\boldsymbol{x}_t^{\mathrm{opt}}$ (red dot) and the no-trade region (red outline)

of the set (Abrams and Karmarkar 1980; Constantinides 1979). Naturally, the Merton point lies inside the no-trade region. We can confirm this result for our optimal policies, i.e., if we choose any point outside the no-trade region (but within the eligible domain $\Omega_{\mathrm{Simplex}}$), computing $\boldsymbol{x}_t + \boldsymbol{\delta}_t^{+,\mathrm{opt}} - \boldsymbol{\delta}_t^{-,\mathrm{opt}}$ then results in a boundary point of the no-trade region. Figure 7 also shows the impact of the state space cropping as the eligible subspace $\Omega_{\mathrm{Simplex}} \subset [0,1]^d$ is not a rectangular domain as needed for the sparse grid interpolation. This is the reason why the certainty equivalent value function $\hat{j}_t^{\mathrm{S}}$ is zero in $[0,1]^d \backslash \Omega_{\mathrm{Simplex}}$ and the optimal sell policies $\boldsymbol{\delta}_t^{-,\mathrm{opt,S}}$ contain a diagonal kink at the hyperplane $\mathbf{1}^\top \cdot \boldsymbol{x}_t$.

Obviously, computing full grid solutions 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 on the LOEWE-CSC cluster with 120 threads. 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 prohibitively long computational times for $d \geq 4$. This underlines the need for sophisticated discretization techniques such as sparse grids.

5.5 Savings in Complexity Using B-Splines

A complexity analysis reveals that the difficulty of solving dynamic portfolio choice models quickly grows with the dimensionality d: The number of necessary arithmetic operations grows like (see Fig. 6)

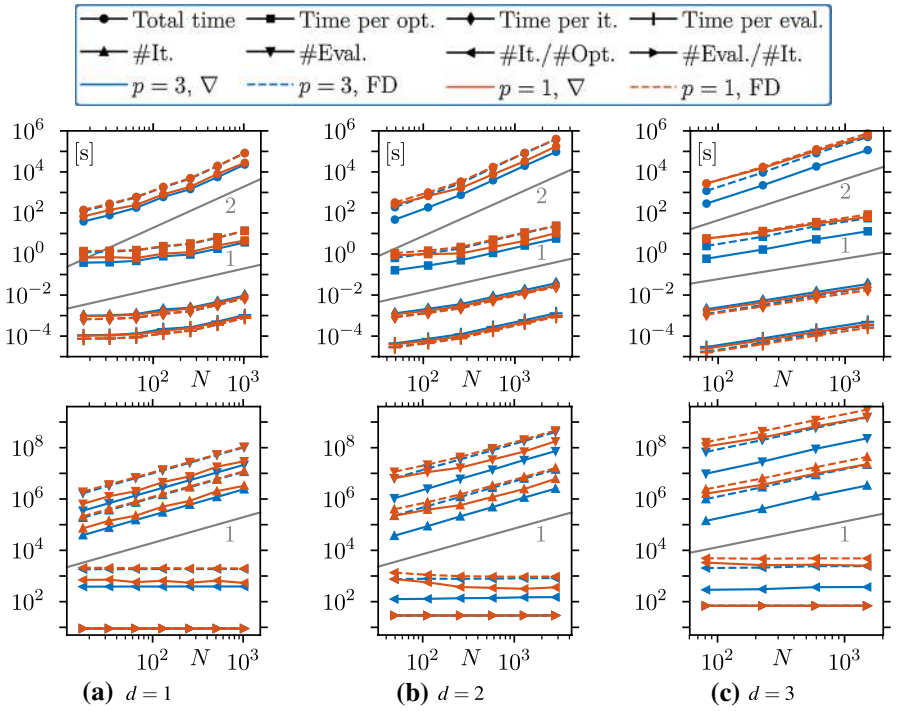


Fig. 8 Computational times (top) and numbers of iterations and evaluations of the interpolant (bottom) for the transaction costs problem on regular sparse grids without refinement. "Total time" is the serial time required to solve all emerging optimization problems. "Time per opt." is this time divided by the number #Opt. $= TN$ of optimization problems. "Time per it." is the total time divided by the number #It. of optimizer 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 total 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 gradients ("∇") or finite differences ("FD")

$$\Theta\Bigg(T \cdot N_t \cdot \#\,\text{optimizer iterations} \cdot \underbrace{Q_t \cdot \overbrace{m_{\boldsymbol{p}} \cdot N_{t+1} \cdot d \cdot p}^{\text{one evaluation of interpolant}}}_{\text{one evaluation of objective gradient}}\Bigg), \tag{41}$$

where for the transaction costs problem $m_{\boldsymbol{p}} = 2d + 1$ and $Q_t, N_t, N_{t+1} \in \Theta(2^n n^{d-1})$ if regular sparse grids of level n without boundary points would be used for state and stochastic grids (due to $m_\xi = d$). In addition, the number of optimizer iterations is likely superlinear in d as this depends on the dimensionality $m_{\boldsymbol{p}}$ of the search space as well as on the number of multi-start points (which also grows with $m_{\boldsymbol{p}}$). This means that the complexity is at least cubic in d, quadratic in the average number N_t of employed state grid points, and linear in the number Q_t of quadrature points. Figure 8 confirms these observations with experimental data using regular sparse grids without spatially adaptive refinement. 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 code optimization effects such as vectorization.

5.5.1 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 larger than in the cubic case). However, as we see in Fig. 8, the total computational time is several times larger for piecewise linear functions although evaluations are cheaper than for B-splines. The main reason is that the number of required optimizer iterations for piecewise linear basis functions is almost seven times as high as in the cubic case since the optimizer has to deal with kinks in the objective function. Our experiments show that beginning with $d = 4$, the total optimization time required to solve the transaction costs problem will be one whole order of magnitude shorter for cubic B-splines than for piecewise linear functions.

5.5.2 Comparing Exact Gradients to Finite Differences

The second benefit is the availability of exact gradients: Figure 8 also contains computational 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_p additional evaluations of the objective function have to be performed to compute the finite differences ($2m_p$ if central differences are used). Consequently, while the resulting weighted Euler equation errors are similar, the total optimization time increases by a factor of up to five if we do not use exact gradients.

5.6 Accuracy Through Spatial Adaptivity

Figure 9 shows the convergence of the scaled L^2 norm $\varepsilon_t^{\mathrm{w},L^2}$ and the L^∞ norm $\varepsilon_t^{\mathrm{w},L^\infty}$ of the 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. We look at the error for $t = 0$ throughout the remaining sections as numerical inaccuracies accrue from time t to time $t - 1$ by the dynamic programming nature of the problem (3). For Fig. 9 and the following plots, the value function grid is left unchanged while the average number N_t of policy grid points increases with decreasing refinement

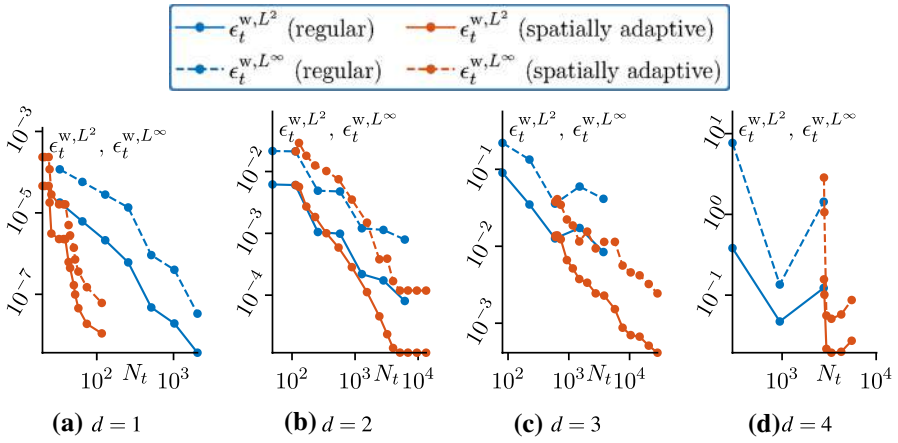
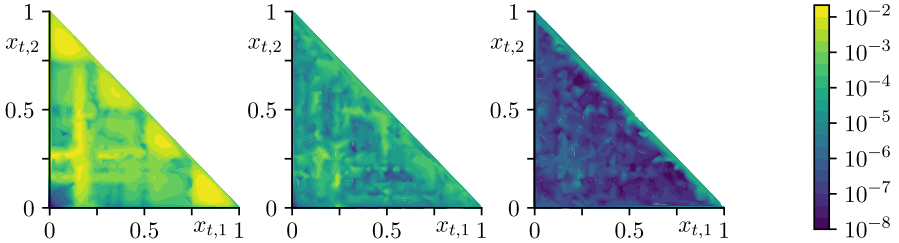


Fig. 9 Convergence of the scaled L^2 norm ε_t^{w,L^2} (solid) and L^∞ norm $\varepsilon_t^{w,L^\infty}$ (dashed) of the weighted Euler equation error for $t = 0$ for regular sparse grids (blue) and spatially adaptive sparse grids (red). The number N_t is the average number $1/m_p \sum_{j=1}^{m_p} N_{t,j}$ 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

threshold ε. 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 sparse 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 sparse grids still outperform regular sparse grids. For $d = 2$, we observe that the error saturates at $N_0 \approx 4000$ points. This is most likely due to floating-point rounding errors 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 $\varepsilon_0^{w,L^2} \approx 1.99e{-}02$ and $\varepsilon_0^{w,L^\infty} \approx 5.76e{-}02$ (with an average number $N_0 = 4252$ of policy grid points). For $d = 5$, we are still able to achieve a small error of $\varepsilon_0^{w,L^2} \approx 2.67e{-}02$ and $\varepsilon_0^{w,L^\infty} \approx 6.37e{-}02$, respectively, with spatially adaptive sparse grids with an average number $N_0 = 12572$ 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 that accurately up to now with conventional methods.

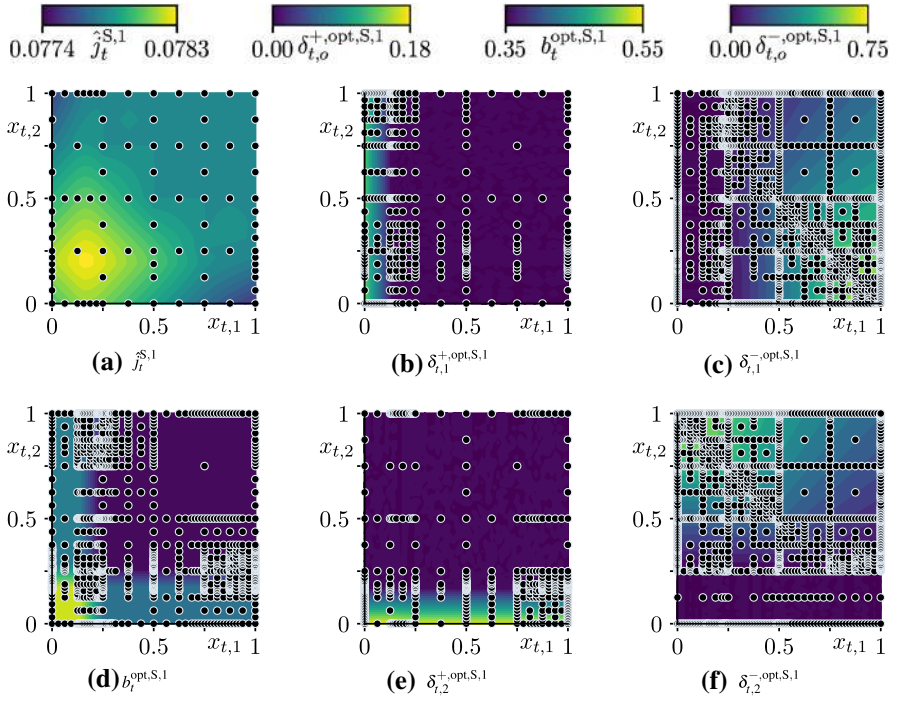
Pointwise plots of the weighted Euler equation error as in Fig. 10 for two stocks reveal that there are two types of regions where the error is large: The first type of region is the neighborhood of the aforementioned diagonal boundary $\mathbf{1}^\top \cdot \boldsymbol{x}_t = 1$ of the uncropped region 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. 10a). When increasing the number of grid points (e.g., Fig. 10b, c), the error decreases quickly in the whole domain.

All in all, Figs. 9 and 10 show that there are two necessary conditions to compute accurate solutions in higher dimensions: Firstly, reliable optimization enabled through B-spline interpolants of the value function and, secondly, spatial



(a) $N_0 = 129$ $(\varepsilon_0^{\mathrm{w},L^2} = 5.3 \cdot 10^{-3})$ **(b)** $N_0 = 889$ $(\varepsilon_0^{\mathrm{w},L^2} = 2.1 \cdot 10^{-4})$ **(c)** $N_0 = 5159$ $(\varepsilon_0^{\mathrm{w},L^2} = 1.2 \cdot 10^{-5})$

Fig. 10 Pointwise weighted Euler equation error $\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}_t)$ for the two-dimensional transaction costs problem and different spatially adaptive sparse grids at $t = 0$



(a) $\hat{j}_t^{\mathrm{S},1}$ **(b)** $\delta_{t,1}^{+,\mathrm{opt},\mathrm{S},1}$ **(c)** $\delta_{t,1}^{-,\mathrm{opt},\mathrm{S},1}$

(d) $b_t^{\mathrm{opt},\mathrm{S},1}$ **(e)** $\delta_{t,2}^{+,\mathrm{opt},\mathrm{S},1}$ **(f)** $\delta_{t,2}^{-,\mathrm{opt},\mathrm{S},1}$

Fig. 11 Spatially adaptive sparse grid solution for the transaction costs problem with $d = 2$ stocks. Shown are the linear interpolant $\hat{j}_t^{\mathrm{S},1}$ for the certainty equivalent value function $\hat{j}_t^{\mathrm{S}}$ (top left) and the linear interpolants $\boldsymbol{p}_t^{\mathrm{opt},\mathrm{S},1}$ for the optimal policies $\boldsymbol{p}_t^{\mathrm{opt},\mathrm{S}}$ at the initial time step $t = 0$ as obtained from Algorithm 1 and 4. The corresponding grid points (dots) are plotted onto the $x_{t,1}$–$x_{t,2}$ plane

adaptive refinement of the policy grids. The latter condition was originally proposed in our previous work (Schober 2018).

Figures 11 and 12 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 $N_0 = 12572$ policy grid points. Obviously, most grid

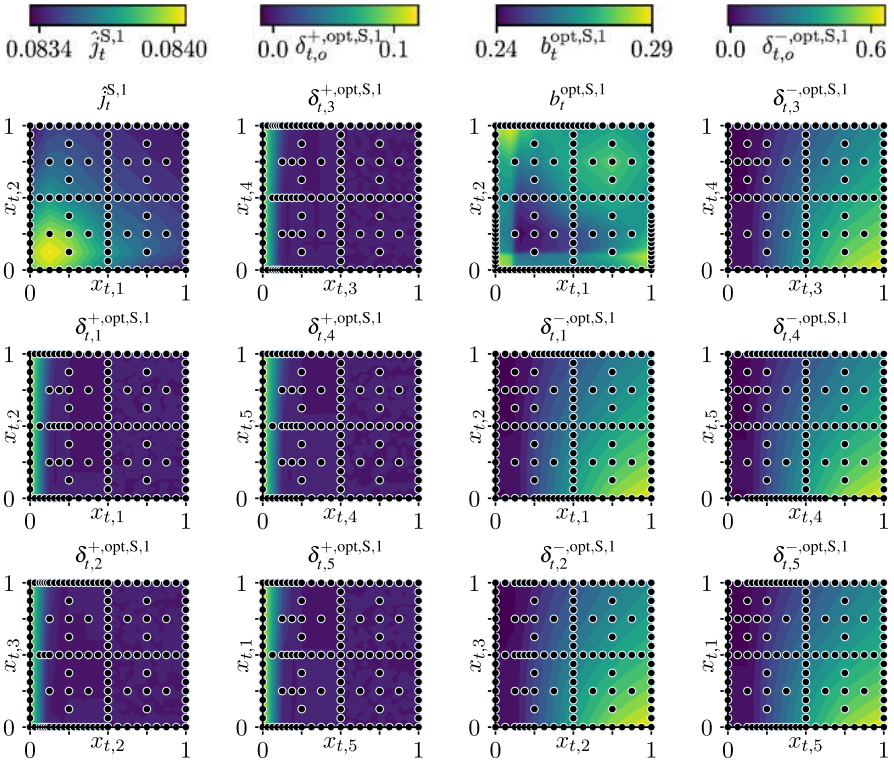


Fig. 12 Spatially adaptive sparse grid solution for the transaction costs problem with $d = 5$ stocks. Shown are slice plots of the linear interpolant $\hat{j}_t^{\mathrm{S},1}$ for the certainty equivalent value function $\hat{j}_t^{\mathrm{S}}$ (top left) and the linear interpolants $\boldsymbol{p}_t^{\mathrm{opt},\mathrm{S},1}$ for the optimal policies $\boldsymbol{p}_t^{\mathrm{opt},\mathrm{S}}$ at the initial time step $t = 0$. For each function a pair (o_1, o_2) of dimensions was chosen, and the stock fractions $x_{t,o}$ of the other dimensions $o \in \{1, \ldots, d\} \setminus \{o_1, o_2\}$ were set to 0.1. In addition, the corresponding grid points (dots) are shown as the projection onto the x_{t,o_1}–x_{t,o_2} plane

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 $\mathbf{1}^\top \cdot \boldsymbol{x}_t = 1$). It is possible to circumvent this issue by either rotating the domain 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 general floating-point rounding errors.

5.7 Solutions in Higher Dimensions

For higher dimensions, economic results for our transaction costs problem (31a–31h) scarcely exist. In continuous time, analytical solutions for special cases (Liu 2004; Liu and Loewenstein 2002) and numerical solutions with finite

element methods (Muthuraman and Kumar 2006) have been discussed. Dynamic programming solutions with value function iteration in discrete time have been studied by Cai (2009), Cai and Judd (2010), Cai et al. (2020) for up to six stocks and one bond without consumption choice.

The purpose of this paper is to show the numerical accuracy of our approach. Rather than analyzing the economic implications of the solution to the higher-dimensional transaction costs problem, we limit ourselves to assessing the resulting optimal policy interpolants $(\boldsymbol{p}_t^{\text{opt,S,1}})_{t=0,\dots,T}$ in a Monte Carlo simulation setup. We calculate the average optimal policy

$$\bar{\boldsymbol{p}}_t^{\text{opt}} := \frac{1}{m_{\text{MC}}} \sum_{j=1}^{m_{\text{MC}}} \boldsymbol{p}_{t,(j)}^{\text{opt}} \tag{42}$$

for $m_{\text{MC}} \in \mathbb{N}$ individuals where $\boldsymbol{p}_{t,(j)}^{\text{opt}} = (b_{t,(j)}^{\text{opt}}, \boldsymbol{\delta}_{t,(j)}^{+,\text{opt}}, \boldsymbol{\delta}_{t,(j)}^{-,\text{opt}})^\top$ denotes the optimal policies of the individuals ($t = 0, \dots, T$ and $j = 1, \dots, m_{\text{MC}}$). They are determined by

$$b_{t,(j)}^{\text{opt}} := b_t^{\text{opt,S,1}}(\boldsymbol{x}_{t,(j)}), \tag{43a}$$

$$\boldsymbol{\delta}_{t,(j)}^{+,\text{opt}} := \boldsymbol{\delta}_t^{+,\text{opt,S,1}}(\boldsymbol{x}_{t,(j)}), \tag{43b}$$

$$\boldsymbol{\delta}_{t,(j)}^{-,\text{opt}} := \boldsymbol{\delta}_t^{-,\text{opt,S,1}}(\boldsymbol{x}_{t,(j)}), \tag{43c}$$

$$\pi_{t,(j)}^{\text{opt}} := b_{t-1,(j)}^{\text{opt}} r_f + (\boldsymbol{x}_{t-1,(j)} + \boldsymbol{\delta}_{t-1,(j)}^{+,\text{opt}} - \boldsymbol{\delta}_{t-1,(j)}^{-,\text{opt}})^\top \cdot \boldsymbol{r}_{t-1,(j)}, \quad t > 0, \tag{43d}$$

$$\boldsymbol{x}_{t,(j)} := \frac{\left(\boldsymbol{x}_{t-1,(j)} + \boldsymbol{\delta}_{t-1,(j)}^{+,\text{opt}} - \boldsymbol{\delta}_{t-1,(j)}^{-,\text{opt}}\right) \odot \boldsymbol{r}_{t-1,(j)}}{\pi_{t-1,(j)}^{\text{opt}}}, \quad t > 0, \quad \boldsymbol{x}_{0,(j)} = \mathbf{1}, \tag{43e}$$

$$\boldsymbol{r}_{t,(j)} \sim LN(\boldsymbol{\mu}, \boldsymbol{\Sigma}). \tag{43f}$$

We plot the resulting average state and policies in Fig. 13 for $d = 2, 3, 4$, and 5 stocks for $m_{\text{MC}} = 10^5$ individuals. In addition, this figure contains the evolution of the weighted Euler equation error $\varepsilon_t^{\text{w},L^2}$ over time. We perform a two-part assessment of the simulation results: First, consumption is slightly increasing over time, which is plausible given the solution, e.g., by Merton (1969) in the finite-horizon case. Second, we compare the stock fractions implied by the Merton points with the simulated stock fractions $\acute{x}_{t,o}/(\mathbf{1}^\top \cdot \acute{\boldsymbol{x}}_t)$ ($o = 1, \dots, d$) for $t = 0$ in Table 1. We observe that the simulated stock fractions deviate from the Merton points' stock fractions as expected. However, after computing the simulated stock fractions for all times, we see that they do not change much over time. This is in line with the buy-and-hold characteristics of solutions to portfolio choice models with transaction costs (e.g., Liu and Loewenstein 2002).

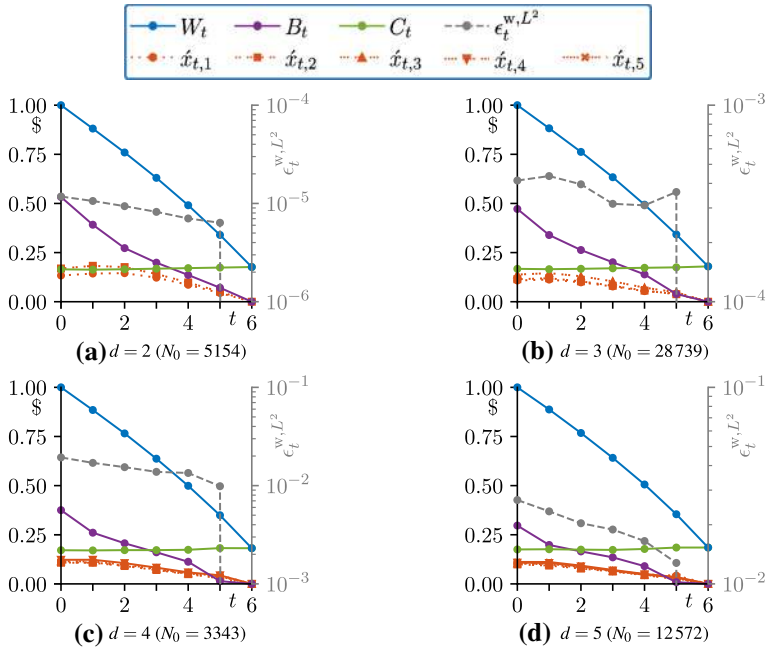


Fig. 13 Average values of wealth W_t (blue), unnormalized optimal bonds B_t (purple), unnormalized optimal consumption C_t (green), and unnormalized stock holdings $\hat{\boldsymbol{x}}_t := (\boldsymbol{x}_t + \boldsymbol{\delta}_t^+ - \boldsymbol{\delta}_t^-)W_t$ after buying and selling (red) in a Monte Carlo simulation of 10^5 individuals where we assume that $W_0 = \$1$ for all individuals. In addition, the plots show the evolution of the scaled L^2 error $\varepsilon_t^{\mathrm{w},L^2}$ over time t (gray, right axes)

Table 1 Simulated stock fractions $\hat{x}_{t,o}/(\mathbf{1}^\top \cdot \hat{\boldsymbol{x}}_t)$ for $t = 0$ and stock fractions implied by the Merton points $x_{t,o}^{\mathrm{opt}}/(\mathbf{1}^\top \cdot \boldsymbol{x}_t^{\mathrm{opt}})$ ($o = 1, \ldots, d$) for the Monte Carlo simulations obtained by evaluating the optimal policy interpolants computed on spatially adaptive sparse grids

d	$\hat{x}_{t,o}/(\mathbf{1}^\top \cdot \hat{\boldsymbol{x}}_t)$	$x_{t,o}^{\mathrm{opt}}/(\mathbf{1}^\top \cdot \boldsymbol{x}_t^{\mathrm{opt}})$
2	(0.441, 0.559)	(0.452, 0.548)
3	(0.300, 0.317, 0.383)	(0.314, 0.302, 0.384)
4	(0.239, 0.238, 0.253, 0.270)	(0.275, 0.185, 0.250, 0.289)
5	(0.199, 0.188, 0.197, 0.205, 0.212)	(0.275, 0.122, 0.176, 0.203, 0.223)

Finally, we present in Table 2 the computational times and numerical errors of the sparse grid solutions underlying Fig. 13 and Table 1.

Table 2 Number of stocks d, base level n, refine tolerance ϵ, grid points of the base grid $|\Omega^{\mathrm{S}}_{n,d}|$ for d stocks and level n, grid points N_t of the refined grid, added points Δ_{N_t}, computational time and weighted Euler equation errors $\varepsilon_t^{\mathrm{w},L^2}$, $\varepsilon_t^{\mathrm{w},L^\infty}$, at $t = 0$. For the optimal policy $\boldsymbol{p}_t^{\mathrm{opt,S,1}}$ rows, the number N_t is the average number $1/m_{\boldsymbol{p}} \sum_{j=1}^{m_{\boldsymbol{p}}} N_{t,j}$ 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

d	Interp.	n	ϵ	$\lvert\Omega^{\mathrm{S}}_{n,d}\rvert$	N_t	Δ_{N_t}	Time	$\varepsilon_t^{\mathrm{w},L^2}$	$\varepsilon_t^{\mathrm{w},L^\infty}$
2	$\hat{j}_t^{\mathrm{S},1}$	4	∞	113	113	0	0 min	8.9 e^{-06}	9.0 e^{-05}
	$\boldsymbol{p}_t^{\mathrm{opt,S,1}}$	4	3 e^{-09}	113	5154	5041	3 min		
3	$\hat{j}_t^{\mathrm{S},1}$	4	∞	593	593	0	3 min	3.7 e^{-04}	3.6 e^{-03}
	$\boldsymbol{p}_t^{\mathrm{opt,S,1}}$	4	2 e^{-07}	593	28735	28142	5 h 19 min		
4	$\hat{j}_t^{\mathrm{S},1}$	4	∞	2769	2769	0	3 h 41 min	1.5 e^{-02}	3.9 e^{-02}
	$\boldsymbol{p}_t^{\mathrm{opt,S,1}}$	4	2 e^{-04}	2769	3343	574	3 h 30 min		
5	$\hat{j}_t^{\mathrm{S},1}$	4	∞	12033	12033	0	9 h 23 min	2.0 e^{-02}	4.8 e^{-02}
	$\boldsymbol{p}_t^{\mathrm{opt,S,1}}$	4	3 e^{-04}	12033	12572	539	3 h 5 min		

6 Conclusion

In this paper, we are the first to develop an approach to accurately solve high-dimensional dynamic portfolio choice models in discrete time that require smooth approximations or gradient-based optimization. With our approach, we have addressed the three key issues of solving these models by means of value function iteration: the curse of dimensionality, the lack of spatial adaptivity, and the lack of continuous gradients all at once by using B-splines on sparse grids with spatially adaptive refinement. We have solved a dynamic portfolio and consumption choice model with transaction costs to study the numerical accuracy of our approach. Solutions to the transaction costs problem with value function iteration have achieved economically acceptable results already for lower resolutions of the interpolation grid than in our presented example. Our approach, however, can easily be applied to other dynamic portfolio choice models or any high-dimensional economic model that require such a high resolution.

We have solved the transaction costs problem with up to five stocks and one risk-free bond, i.e., a five-dimensional interpolation and an eleven-dimensional optimization problem per time step. Using spatially adaptive refinement of the optimal policies, we have obtained maximum unit-free Euler equation errors around 5% for the five-dimensional problem and even lower maximum errors for lower-dimensional problems. This showcases the high accuracy of the proposed spatially adaptive solution scheme for the optimization of continuous choices, which relies on smooth approximations of the value function and the gradient. We have shown convergence of our approach in up to four dimensions. Here, spatially adaptive refinement of the optimal policies decreased the maximum

Euler equation error by nearly two orders of magnitude in the four-dimensional case compared to regular sparse grids without spatially adaptive refinement. This has shown that only with spatial adaptivity, high-dimensional problems can be solved accurately. Finally, we have given a rigorous analysis of the complexity of our approach for dynamic portfolio choice models in general, not only for the transaction costs problem for which we have verified our analysis of complexity with measurements of computational time. We have found that the sole availability of the gradient for the optimization process has saved nearly one order of magnitude in computational complexity in the three-stock case. We expect even larger reductions of the computational complexity for higher-dimensional problems. Compared to finite differences with interpolation on hat functions as used by Brumm and Scheidegger (2017), we have saved considerably more than one order of magnitude in computational complexity and one order of magnitude in total computational time in three dimensions.

There are certain limitations to the applicability of spatially adaptive sparse grids to solve high-dimensional dynamic economic models: Firstly, sparse grid approximations are not shape-preserving, which is especially of importance for value function iteration with interpolation (Cai and Judd 2012). Secondly, the calculation of the coefficients of the B-spline interpolant is time-consuming and not trivial to parallelize since the solution to a system of linear equations has to be computed in every time step. Thirdly, the exact choice of the refinement tolerance for value function and policy interpolants is subject to trial-and-error. Choosing a refinement tolerance that is too low will lead to too many points that are inserted and may cause instability of the entire scheme if the optimizer does not give perfect results.

Future improvements of our approach may lie in the use of problem-tailored adaptivity criteria (Brumm and Scheidegger 2017; Pflüger 2012) instead of the simple surplus-based refinement criterion.

Appendix A

A.1 Sequential Quadratic Programming (SQP)

SQP methods are well-suited to compute the solution to problem (3). These methods use the linearization of the Lagrangian

$$\mathscr{L}_t(\boldsymbol{p}_t, \lambda_t, \boldsymbol{x}_t^{(k)}) := \tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}) + \lambda_t^{\top} \cdot \boldsymbol{g}_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}), \tag{44}$$

with Kuhn-Tucker multipliers $\lambda_t \in \mathbb{R}^{m_g}$ to set up a quadratic programming problem

$$\max_{\boldsymbol{d}_t^{(i)}} \left\{ \nabla_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t^{(i)}, \boldsymbol{x}_t^{(k)})^{\top} \cdot \boldsymbol{d}_t^{(i)} + \frac{1}{2} \boldsymbol{d}_t^{(i)\top} \cdot \nabla_{\boldsymbol{p}_t}^2 \tilde{j}_t^{\mathrm{S}}(\boldsymbol{p}_t^{(i)}, \boldsymbol{x}_t^{(k)}) \cdot \boldsymbol{d}_t^{(i)} \right\}, \tag{45}$$

which finds the search direction $\boldsymbol{d}_t^{(i)}$ for the current iterate $\boldsymbol{p}_t^{(i)}$ starting from an initial guess $\boldsymbol{p}_t^{(0)} \in \Psi \subset \mathbb{R}^{m_p}$.[8] This problem can hence be solved by means of standard quadratic programming where frequently the Hessian $\nabla^2_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}}$ is approximated by the BFGS method (Fletcher 2013). Finally, the next iterate $\boldsymbol{p}_t^{(i+1)}$ is chosen by an appropriate line search procedure over the step length σ_i:

$$\boldsymbol{p}_t^{(i+1)} := \boldsymbol{p}_t^{(i)} + \sigma_i \boldsymbol{d}_t^{(i)} . \tag{46}$$

If the gradient $\nabla_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}}$ is not available, implementations of SQP methods approximate it by finite differences. This leads to $m_{\boldsymbol{p}}$ additional evaluations of $\tilde{j}_t^{\mathrm{S}}$ per quadratic programming iteration and to $2m_{\boldsymbol{p}}$ additional evaluations if central finite differences are used. Some SQP routines allow the user to choose finite difference approximations, some automatically use central finite differences in certain situations to achieve higher accuracy when needed. However, almost all SQP routines allow the user to provide the gradient $\nabla_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}}$ (and many also the gradients of the constraints) to save computing time and to increase accuracy. For details see Fletcher (2013), Gill et al. (2005).

The gradient $\nabla_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}}$ of the target function (6) at the grid point $\boldsymbol{x}_t^{(k)}$,

$$\nabla_{\boldsymbol{p}_t} \tilde{j}_t^{\mathrm{S}} = \nabla_{\boldsymbol{p}_t} u(c_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)})) + \rho \mathbb{E}_t \left[\nabla_{\boldsymbol{f}_t} j_{t+1}^{\mathrm{S}}(\boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}, \boldsymbol{\zeta}_t))^{\top} \cdot \nabla_{\boldsymbol{p}_t} \boldsymbol{f}_t(\boldsymbol{p}_t, \boldsymbol{x}_t^{(k)}, \boldsymbol{\zeta}_t) \right], \tag{47}$$

can be supplied to the SQP routine by evaluating the approximation of the gradient $\nabla_{\boldsymbol{f}_t} j_{t+1}^{\mathrm{S}}$ rather than using finite differences.

A.2 Proof of the Normalization

Theorem 1 *For all* $t = 0, \ldots, T$,

$$J_t(W_t, \boldsymbol{x}_t) = W_t^{1-\gamma} j_t(\boldsymbol{x}_t) , \tag{48}$$

where J_t *is the solution of problem* (27a–27c) *and* j_t *the solution of problem* (31a–31h).

Proof We have $u(C_t) = u(W_t c_t) = W_t^{1-\gamma} u(c_t)$ for all t due to the choice of the utility function (2).

Base case $t = T$: Because of Eqs. (27b) and (31b) it is

$$\begin{aligned} J_T(W_T, \boldsymbol{x}_T) &= u\big(\big(1 - \tau \mathbf{1}^{\top} \cdot \boldsymbol{x}_T\big) W_T\big) = W_T^{1-\gamma} u\big(1 - \tau \mathbf{1}^{\top} \cdot \boldsymbol{x}_T\big) \\ &= W_T^{1-\gamma} j_T(\boldsymbol{x}_t) . \end{aligned} \tag{49}$$

Inductive hypothesis: For $t+1$ it is $J_{t+1}\big(W_{t+1}, \boldsymbol{x}_{t+1}\big) = W_{t+1}^{1-\gamma} j_{t+1}(\boldsymbol{x}_{t+1})$.

[8] We define $\nabla_{\boldsymbol{x}} \boldsymbol{f} := (\partial f_j / \partial x_i)_{i,j}$ (i.e., the transposed Jacobian) and "$\cdot$" denotes the matrix-vector product.

Inductive step $t+1 \to t$: By the inductive hypothesis and Eq. (30a) it is

$$\begin{aligned} J_t(W_t, \boldsymbol{x}_t) &= \max_{B_t, \boldsymbol{\Delta}_t^+, \boldsymbol{\Delta}_t^-} \left\{ u(C_t) + \rho \mathbb{E}_t \left[W_{t+1}^{1-\gamma} j_{t+1}(\boldsymbol{x}_{t+1}) \right] \right\} \\ &= W_t^{1-\gamma} \max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ u(c_t) + \rho \mathbb{E}_t \left[\pi_{t+1}^{1-\gamma} j_{t+1}(\boldsymbol{x}_{t+1}) \right] \right\} \\ &= W_t^{1-\gamma} j_t(\boldsymbol{x}_t) \,. \end{aligned} \tag{50}$$

□

A.3 Proof of the Certainty Equivalent Transformation

Theorem 2 *For all* $t = 0, \ldots, T$,

$$j_t(\boldsymbol{x}_t) = \frac{1}{1-\gamma} \hat{j}_t(\boldsymbol{x}_t)^{1-\gamma} \,, \tag{51}$$

where j_t *the solution of problem* (31a–31h) *and* $\hat{j}_t$ *the solution of problem* (33a, 33b).

Proof **Base case** $t = T$: Because of Eqs. (31b) and (33b) it is

$$\begin{aligned} j_T(\boldsymbol{x}_T) &= u\left(1 - \tau \mathbf{1}^\top \cdot \boldsymbol{x}_T\right) = \frac{1}{1-\gamma} \left(1 - \tau \mathbf{1}^\top \cdot \boldsymbol{x}_T\right)^{1-\gamma} \\ &= \frac{1}{1-\gamma} \hat{j}_T(\boldsymbol{x}_t)^{1-\gamma} \,. \end{aligned} \tag{52}$$

Inductive hypothesis: For $t+1$ it is $j_{t+1}(\boldsymbol{x}_{t+1}) = 1/(1-\gamma) \hat{j}_{t+1}(\boldsymbol{x}_{t+1})^{1-\gamma}$.

Inductive step $t+1 \to t$: By the inductive hypothesis and Eq. (33a) it is

$$\begin{aligned} j_t(\boldsymbol{x}_t) &= \max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ u(c_t) + \rho \mathbb{E}_t \left[\pi_{t+1}^{1-\gamma} j_{t+1}(\boldsymbol{x}_{t+1}) \right] \right\} \\ &= \max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ \frac{1}{1-\gamma} c_t^{1-\gamma} + \rho \mathbb{E}_t \left[\pi_{t+1}^{1-\gamma} \frac{1}{1-\gamma} \hat{j}_{t+1}(\boldsymbol{x}_{t+1})^{1-\gamma} \right] \right\} \\ &= \frac{1}{1-\gamma} \min_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ c_t^{1-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}(\boldsymbol{x}_{t+1}) \right)^{1-\gamma} \right] \right\} \\ &= \frac{1}{1-\gamma} \left(\max_{b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-} \left\{ \left(c_t^{1-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}(\boldsymbol{x}_{t+1}) \right)^{1-\gamma} \right] \right)^{\frac{1}{1-\gamma}} \right\} \right)^{1-\gamma} \\ &= \frac{1}{1-\gamma} \hat{j}_t(\boldsymbol{x}_t)^{1-\gamma} \,, \end{aligned} \tag{53}$$

where we used that $(\cdot)^{1/(1-\gamma)}$ is a strictly monotonously decreasing function and $1-\gamma < 0$. □

A.4 Analytical Gradients

For $t < T$ at state $\boldsymbol{x}_t$, let us define the objective function (6) for the certainty equivalent formulation of the transaction costs problem (33a, 33b) by

$$\tilde{\hat{j}}_t^{\mathrm{S}}(b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{x}_t) := \left\{ c_t^{1-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}^{\mathrm{S}} \right)^{1-\gamma} \right] \right\}^{\frac{1}{1-\gamma}}, \tag{54}$$

where $\hat{j}_t^{\mathrm{S}}$ denotes the sparse grid B-spline approximation of Eq. (33a) evaluated at $\boldsymbol{x}_t$. With the sparse grid B-spline approximation of the gradient with respect to $\boldsymbol{x}_t$ evaluated at $\boldsymbol{x}_t$, $\boldsymbol{\nabla}_{\boldsymbol{x}_t} \hat{j}_t^{\mathrm{S}}$, the gradient of the objective function (47) with respect to the policies b_t, $\boldsymbol{\delta}_t^+$, and $\boldsymbol{\delta}_t^-$ is:

$$\boldsymbol{\nabla}_{b_t} \tilde{\hat{j}}_t^{\mathrm{S}} = \tilde{\hat{j}}_t^{\mathrm{S}\gamma} \left(-c_t^{-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}^{\mathrm{S}} \right)^{-\gamma} r_f \left(\hat{j}_{t+1}^{\mathrm{S}} - \left(\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\mathrm{S}} \right)^{\top} \cdot \boldsymbol{x}_{t+1} \right) \right] \right), \tag{55a}$$

$$\boldsymbol{\nabla}_{\boldsymbol{\delta}_t^+} \tilde{\hat{j}}_t^{\mathrm{S}} = \tilde{\hat{j}}_t^{\mathrm{S}\gamma} \left(-(1+\tau) \mathbf{1} c_t^{-\gamma} + \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}^{\mathrm{S}} \right)^{-\gamma} \boldsymbol{r}_t \odot \left(\mathbf{1} \hat{j}_{t+1}^{\mathrm{S}} + \boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\mathrm{S}} - \mathbf{1} \left(\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\mathrm{S}} \right)^{\top} \cdot \boldsymbol{x}_{t+1} \right) \right] \right), \tag{55b}$$

$$\boldsymbol{\nabla}_{\boldsymbol{\delta}_t^-} \tilde{\hat{j}}_t^{\mathrm{S}} = \tilde{\hat{j}}_t^{\mathrm{S}\gamma} \left(-(\tau-1) \mathbf{1} c_t^{-\gamma} - \rho \mathbb{E}_t \left[\left(\pi_{t+1} \hat{j}_{t+1}^{\mathrm{S}} \right)^{-\gamma} \boldsymbol{r}_t \odot \left(\mathbf{1} \hat{j}_{t+1}^{\mathrm{S}} + \boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\mathrm{S}} - \mathbf{1} \left(\boldsymbol{\nabla}_{\boldsymbol{x}_{t+1}} \hat{j}_{t+1}^{\mathrm{S}} \right)^{\top} \cdot \boldsymbol{x}_{t+1} \right) \right] \right). \tag{55c}$$

A.5 State Space Cropping

To obtain function values outside the feasible state space we *virtually sell,* if $\mathbf{1}^{\top} \cdot \boldsymbol{x}_t > 1$ as many stocks as needed to meet the constraint $\mathbf{1}^{\top} \cdot \boldsymbol{x}_t \leq 1$. We already might need to sell stocks even if $\mathbf{1}^{\top} \cdot \boldsymbol{x}_t$ is smaller but close to one in order to satisfy the minimum consumption requirement (31c). In detail, we replace $\boldsymbol{x}_t$ by $\hat{\beta} \boldsymbol{x}_t$ whenever $\hat{\beta} < 1$ where $\hat{\beta} > 0$ is a *cropping factor* that is determined by

$$\left[1 - \tau \left(\mathbf{1}^{\top} \cdot \boldsymbol{x}_t - \mathbf{1}^{\top} \cdot (\hat{\beta} \boldsymbol{x}_t) \right) \right] \cdot \left(1 - \mathbf{1}^{\top} \cdot (\hat{\beta} \boldsymbol{x}_t) \right) = c_{\min}. \tag{56}$$

Here, $\left(\mathbf{1}^{\top} \cdot \boldsymbol{x}_t - \mathbf{1}^{\top} \cdot (\hat{\beta} \boldsymbol{x}_t) \right)$ 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 $\left(1 - \mathbf{1}^{\top} \cdot (\hat{\beta} \boldsymbol{x}_t) \right)$ is the fraction of wealth that can be consumed after the virtual selling, which needs to be at least $c_{\min}$. Solving Eq. (56) for $\hat{\beta}$ and choosing the positive solution, we finally obtain

$$\hat{\beta} = \frac{\tau \left(1 + \mathbf{1}^{\top} \cdot \boldsymbol{x}_t \right) - 1 + \sqrt{\tau^2 \left(1 - \mathbf{1}^{\top} \cdot \boldsymbol{x}_t \right)^2 - 2\tau \left(2c_{\min} - 1 + \mathbf{1}^{\top} \cdot \boldsymbol{x}_t \right) + 1}}{2\tau \mathbf{1}^{\top} \cdot \boldsymbol{x}_t}. \tag{57}$$

A.6 Euler Equation Error Derivation

The Lagrangian (44) for the transaction costs problem in certainty equivalent formulation (33a, 33b) is given by:

$$\mathscr{L}_t(b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{\lambda}_t, \boldsymbol{x}_t) := \tilde{\tilde{j}}_t^{\mathrm{S}}(b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{x}_t) + \boldsymbol{\lambda}_t^\top \cdot \boldsymbol{g}_t(b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{x}_t)\,, \tag{58}$$

with $\tilde{\tilde{j}}_t^{\mathrm{S}}$ from Eq. (54), $\boldsymbol{\lambda} \in \mathbb{R}^{3d+2}$ and $\boldsymbol{g}_t(b_t, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{x}_t) = (c_t - c_{\min}, \boldsymbol{\delta}_t^+, \boldsymbol{\delta}_t^-, \boldsymbol{x}_t - \boldsymbol{\delta}_t^-, b_t)^\top$ from constraints Eqs. (31c) to (31g).

The first order condition with regard to b_t is

$$\boldsymbol{\nabla}_{b_t} \mathscr{L}_t = \boldsymbol{\nabla}_{b_t} \tilde{\tilde{j}}_t^{\mathrm{S}} - \lambda_t^1 \boldsymbol{\nabla}_{b_t} c_t^{\mathrm{opt}} - \lambda_t^{3d+2} = 0\,. \tag{59}$$

We neglect binding constraints, i.e., we assume $\lambda_t^1 = \lambda_t^{3d+2} = 0$, and set the error $\varepsilon_t^{\mathrm{w}}(\boldsymbol{x}_t) = \mathrm{NaN}$ whenever $\lambda_t^1 \neq 0$ or $\lambda_t^{3d+2} \neq 0$ for any $\boldsymbol{x}_t$. Thus, we take the NaN-mean in Eq. (39a) and the NaN-maximum in Eq. (39b).

Assuming $\lambda_t^1 = \lambda_t^{3d+2} = 0$ in Eq. (59), plugging in Eq. (55a) for $\boldsymbol{\nabla}_{b_t} \tilde{\tilde{j}}_t^{\mathrm{S}}$ at the optimum $(b_t^{\mathrm{opt}}, \boldsymbol{\delta}_t^{+,\mathrm{opt}}, \boldsymbol{\delta}_t^{-\mathrm{opt}})^\top$ and dividing both sides by $\tilde{\tilde{j}}_t^{\mathrm{S}\,\gamma}$ yields Eq. (35).

Acknowledgements We thank the initiative High Performance Computing in Hessen for granting us computing time at the LOEWE-CSC cluster and the Lichtenberg High Performance Computer. We appreciate valuable remarks to improve this paper from two anonymous referees. Helpful comments have been provided by Johannes Brumm, Yannick Dillschneider, Kenneth Judd, and Alexander Ludwig. We thank Stefan Zimmer for helping us develop the weakly fundamental not-a-knot splines. Finally, Peter Schober thanks Raimond Maurer for supporting this research in every way possible.

Funding Open Access funding enabled and organized by Projekt DEAL. This work was supported by funding from the German Investment and Asset Management Association (BVI), the Juniorprofessurenprogramm of the Landesstiftung Baden-Württemberg, and the DFG (Cluster of Excellence SimTech EXC310/EXC2075).

Availability of data and materials/Code availability The code used to generate the results presented in this paper and all input data is publicly available at trix0r/BBSG.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

References

Abrams, R. A., & Karmarkar, U. S. (1980). Optimal multiperiod investment-consumption policies. *Econometrica*, *48*(2), 333–353.

Barberis, N., & Huang, M. (2009). Preferences with frames: A new utility specification that allows for the framing of risks. *Journal of Economic Dynamics and Control*, *33*(8), 1555–1576.

Barthelmann, V., Novak, E., & Ritter, K. (2000). High dimensional polynomial interpolation on sparse grids. *Advances in Computational Mathematics*, *12*(4), 273–288.

Bellman, R. (1954). The theory of dynamic programming. *Bulletin of the American Mathematical Society*, *60*(6), 503–515.

Bellman, R. (1961). *Adaptive control processes: A guided tour*. Princeton: Princeton University Press. https://doi.org/10.1515/9781400874668.

Brumm, J., & Grill, M. (2014). Computing equilibria in dynamic models with occasionally binding constraints. *Journal of Economic Dynamics and Control*, *38*, 142–160.

Brumm, J., & Scheidegger, S. (2017). Using adaptive sparse grids to solve high-dimensional dynamic models. *Econometrica*, *85*(5), 1575–1612.

Bungartz, H.-J., & Griebel, M. (2004). Sparse grids. *Acta Numerica*, *13*, 147–269.

Cai, Y. (2009). *Dynamic programming and its application in economics and finance*. Ph.D. thesis, Stanford University. https://purl.stanford.edu/zd335yg6884.

Cai, Y. (2019). Computational methods in environmental and resource economics. *Annual Review of Resource Economics*, *11*(1), 59–82.

Cai, Y., & Judd, K. L. (2010). Stable and efficient computational methods for dynamic programming. *Journal of the European Economic Association*, *8*(2–3), 626–634.

Cai, Y., & Judd, K. L. (2012). Shape-preserving dynamic programming. *Mathematical Methods of Operations Research*, *77*(3), 407–421.

Cai, Y., & Judd, K. L. (2015). Dynamic programming with Hermite approximation. *Mathematical Methods of Operations Research*, *81*(3), 245–267.

Cai, Y., Judd, K. L., Thain, G., & Wright, S. J. (2015). Solving dynamic programming problems on a computational grid. *Computational Economics*, *45*(2), 261–284.

Cai, Y., Judd, K. L., & Xu, R. (2020). *Numerical solution of dynamic portfolio optimization with transaction costs*. Working paper. EID arXiv:2003.01809.

Chu, M. T., Kuo, C.-H., & Lin, M. M. (2013). Tensor spline approximation in economic dynamics with uncertainties. *Computational Economics*, *42*(2), 175–198.

Cocco, J. F., Gomes, F. J., & Maenhout, P. J. (2005). Consumption and portfolio choice over the life cycle. *Review of Financial Studies*, *18*(2), 491–533.

Constantinides, G. M. (1979). Multiperiod consumption and investment behavior with convex transactions costs. *Management Science*, *25*(11), 1127–1137.

Cox, M. G. (1972). The numerical evaluation of B-splines. *IMA Journal of Applied Mathematics*, *10*(2), 134–149.

de Boor, C. (1972). On calculating with B-splines. *Journal of Approximation Theory*, *6*(1), 50–62.

De Giorgi, E. G., & Legg, S. (2012). Dynamic portfolio choice and asset pricing with narrow framing and probability weighting. *Journal of Economic Dynamics and Control*, *36*(7), 951–972.

Epstein, L. G., & Zin, S. E. (1989). Substitution, risk aversion, and the temporal behavior of consumption and asset returns: A theoretical framework. *Econometrica*, *57*(4), 937–969.

Fletcher, R. (2013). *Practical methods of optimization*. New York: Wiley. https://doi.org/10.1002/9781118723203.

Garlappi, L., & Skoulakis, G. (2009). Numerical solutions to dynamic portfolio problems: The case for value function iteration using Taylor approximation. *Computational Economics*, *33*(2), 193–207.

Gill, P. E., Murray, W., & Saunders, M. A. (2005). SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, *47*(1), 99–131.

Habermann, C., & Kindermann, F. (2007). Multidimensional spline interpolation: Theory and applications. *Computational Economics*, *30*(2), 153–169.

Horneff, V., Maurer, R., & Schober, P. (2016). *Efficient parallel solution methods for dynamic portfolio choice models in discrete time*. Working paper 2665031. Available at SSRN. Goethe University Frankfurt. https://doi.org/10.2139/ssrn.2665031.

Horneff, W. J., Maurer, R., & Rogalla, R. (2010). Dynamic portfolio choice with deferred annuities. *Journal of Banking & Finance*, *34*(11), 2652–2664.
Horneff, W. J., Maurer, R., & Stamos, M. Z. (2008). Life-cycle asset allocation with annuity markets. *Journal of Economic Dynamics and Control*, *32*(11), 3590–3612.
Hubener, A., Maurer, R., & Mitchell, O. S. (2016). How family status and social security claiming options shape optimal life cycle portfolios. *Review of Financial Studies*, *29*(4), 937–978.
Hubener, A., Maurer, R., & Rogalla, R. (2014). Optimal portfolio choice with annuities and life insurance for retired couples. *Review of Finance*, *18*(1), 147–188.
Höllig, K., & Hörner, J. (2013). Approximation and modeling with B-splines. *SIAM*. http://bookstore.siam.org/ot132/.
Inkmann, J., Lopes, P., & Michaelides, A. (2011). How deep is the annuity market participation puzzle? *Review of Financial Studies*, *24*(1), 279–319.
Judd, K. L. (1998). *Numerical methods in economics*. Cambridge: MIT Press. https://mitpress.mit.edu/books/numericalmethods-economics.
Judd, K. L., Maliar, L., Maliar, S., & Valero, R. (2014). Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain. *Journal of Economic Dynamics and Control*, *44*, 92–123.
Judd, K. L., & Solnick, A. (1994). *Numerical dynamic programming with shape-preserving splines*. Unpublished manuscript, Hoover Institution. https://web.stanford.edu/~judd/papers/dpshape.pdf.
Kamin, J. H. (1975). Optimal portfolio revision with a proportional transaction cost. *Management Science*, *21*(11), 1263–1271.
Liu, H. (2004). Optimal consumption and investment with transaction costs and multiple risky assets. *Journal of Finance*, *59*(1), 289–338.
Liu, H., & Loewenstein, M. (2002). Optimal portfolio selection with transaction costs and finite horizons. *Review of Financial Studies*, *15*(3), 805–835.
Magill, M. J., & Constantinides, G. M. (1976). Portfolio selection with transactions costs. *Journal of Economic Theory*, *13*(2), 245–263.
Merton, R. C. (1969). Lifetime portfolio selection under uncertainty: The continuous-time case. *Review of Economics and Statistics*, *51*(3), 247–257.
Muthuraman, K., & Kumar, S. (2006). Multidimensional portfolio optimization with proportional transaction costs. *Mathematical Finance*, *16*(2), 301–335.
Pflüger, D. (2010). Spatially adaptive sparse grids for high-dimensional problems. Verlag Dr. Hut. https://www5.in.tum.de/pub/pflueger10spatially.pdf.
Pflüger, D. (2012). Spatially adaptive refinement. In J. Garcke & M. Griebel (Eds.), *Sparse grids and applications. Lecture Notes in Computational Science and Engineering* (Vol. 88, pp. 243–262). Berlin: Springer.
Philbrick, C. R, Jr., & Kitanidis, P. K. (2001). Improved dynamic programming methods for optimal control of lumped-parameter stochastic systems. *Operations Research*, *49*(3), 398–412.
Rust, J. (2008). Dynamic programming. In S. N. Durlauf & L. E. Blume (Eds.), *The new Palgrave dictionary of economics* (Vol. 1-8, pp. 1471–1489). London: Palgrave Macmillan.
Schober, P. (2018). Solving dynamic portfolio choice models in discrete time using spatially adaptive sparse grids. In J. Garcke, D. Pflüger, C. Webster, & G. Zhang (Eds.), *Sparse grids and applications-Miami 2016. Lecture Notes in Computational Science and Engineering* (Vol. 123, pp. 135–173). Berlin: Springer.
Sickel, W., & Ullrich, T. (2011). Spline interpolation on sparse grids. *Applicable Analysis*, *90*(3–4), 337–383.
Stoyanov, M. (2017). User manual: TASMANIAN sparse grids v4.0. Technical report, Oak Ridge National Laboratory. https://tasmanian.ornl.gov/documents/UserManual.pdf.
Valentin, J. (2019). *B-splines for sparse grids: Algorithms and application to higher-dimensional optimization*. Ph.D. thesis. University of Stuttgart.
Valentin, J., & Pflüger, D. (2016). Hierarchical gradient-based optimization with B-splines on sparse grids. In J. Garcke & D. Pflüger (Eds.), *Sparse grids and applications-Stuttgart 2014. Lecture Notes in Computational Science and Engineering* (Vol. 109, pp. 315–336). Berlin: Springer.
Valentin, J., Sprenger, M., Pflüger, D., & Röhrle, O. (2018). 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), 1–21.

Winschel, V., & Krätzig, M. (2010). Solving, estimating, and selecting nonlinear dynamic models without the curse of dimensionality. *Econometrica*, *78*(2), 803–821.

Zenger, C. (1991). Sparse grids. In W. Hackbusch (Ed.), *Parallel algorithms for partial differential equations. Notes on Numerical Fluid Mechanics* (Vol. 31, pp. 241-251). Braunschweig: Vieweg. http://www5.in.tum.de/pub/zenger91sg.pdf.