Journal of Economic Dynamics & Control 44 (2014) 92–123

Contents lists available at ScienceDirect

Journal of Economic Dynamics & Control

journal homepage: www.elsevier.com/locate/jedc

Smolyak method for solving dynamic economic models: Lagrange interpolation, anisotropic grid and adaptive domain

Kenneth L. Judd [a], Lilia Maliar [b,c,*], Serguei Maliar [c,d], Rafael Valero [c]

[a] Hoover Institution, 434 Galvez Mall, Stanford University, Stanford, CA 94305-6010, USA
[b] Department of Economics, 579 Serra Mall, Stanford University, Stanford, CA 94305-6072, USA
[c] Department of Fundamentos del Análisis Económico, University of Alicante, Campus de San Vicente, 03080 Alicante, Spain
[d] Leavey School of Business, Lucas Hall 124, Santa Clara University, Santa Clara, CA, 95053, USA

ARTICLE INFO

Article history:
Received 17 August 2013
Received in revised form
8 March 2014
Accepted 9 March 2014
Available online 3 April 2014

JEL classification:
C63
C68

Keywords:
Smolyak method
Sparse grid
Adaptive domain
Projection
Anisotropic grid
High-dimensional problem

ABSTRACT

We show how to enhance the performance of a Smolyak method for solving dynamic economic models. First, we propose a more efficient implementation of the Smolyak method for interpolation, namely, we show how to avoid costly evaluations of repeated basis functions in the conventional Smolyak formula. Second, we extend the Smolyak method to include anisotropic constructions that allow us to target higher quality of approximation in some dimensions than in others. Third, we show how to effectively adapt the Smolyak hypercube to a solution domain of a given economic model. Finally, we argue that in large-scale economic applications, a solution algorithm based on Smolyak interpolation has substantially lower expense when it uses derivative-free fixed-point iteration instead of standard time iteration. In the context of one- and multi-agent optimal growth models, we find that the proposed modifications to the conventional Smolyak method lead to substantial increases in accuracy and speed.

1. Introduction

In a seminal paper, Smolyak (1963) proposed a sparse-grid method that allows to efficiently represent, integrate and interpolate functions on multidimensional hypercubes. The Smolyak method is not subject to the curse of dimensionality and can be used to solve large-scale applications. A pioneering work of Krueger and Kubler (2004) introduced the Smolyak method to economics in the context of a projection-style iterative method for solving multi-period overlapping generation models. Smolyak methods have been also used to solve portfolio-choice problems (Gavilan-Gonzalez and Rojas, 2009); to develop state-space filters tractable in large-scale problems (Winschel and Krätzig, 2010); to solve models with infinitely lived heterogenous agents (Malin et al., 2011; Gordon, 2011; Brumm and Scheidegger, 2013); and to solve new Keynesian models (Fernández-Villaverde et al., 2012).

While the Smolyak method enables us to study far larger problems than do tensor-product methods, its computational expense still grows rapidly with the dimensionality of the problem. In particular, Krueger and Kubler (2004) and Malin et al. (2011) document a high computational cost of their solution methods when the number of state variables exceeds twenty.

* Corresponding author at: Department of Economics, 579 Serra Mall, Stanford University, Stanford, CA 94305-6072, USA. Tel.: +1 6507259069.
E-mail address: maliarl@stanford.edu (L. Maliar).

http://dx.doi.org/10.1016/j.jedc.2014.03.003
0165-1889/

In the paper, we show a more efficient implementation of the Smolyak method that reduces its computational expense, and we propose extensions of the Smolyak method that enable us to more effectively solve dynamic economic models.[1]

First, the conventional Smolyak formula is inefficient. To interpolate a function in a given point, it first causes the computer to create and evaluate a long list of repeated basis functions, and it then constructs linear combinations of such functions to get rid off repetitions. In high-dimensional problems, the number of repetitions is large and slows down computations dramatically. We offer a way to avoid costly evaluations of the repeated basis functions: Instead of conventional nested-set generators, we introduce disjoint-set generators. Nested sets include one another, and as a result, their tensor products contain repeated elements but our disjoint sets do not. This is why our implementation of the Smolyak formula does not have repetitions.[2]

An efficient implementation of the Smolyak method is especially important in the context of numerical methods for solving dynamic economic models which require us to interpolate decision and value functions a very large number of times, e.g., in each grid point, integration node or time period. We save on cost every time when we perform an evaluation of the Smolyak interpolant.

To compute the interpolation coefficients, we use a universal Lagrange interpolation technique instead of the conventional closed-form expressions. Namely, we proceed in three steps: (i) construct M Smolyak grid points; (ii) construct M corresponding Smolyak basis functions; and (iii) interpolate the values of the true function at the grid points using the basis functions. We then solve a system of M linear equations in M unknowns. The cost of solving this system can be high but it is a fixed cost in the context of iterative methods for solving dynamic economic models. Namely, we argue that an expensive inverse in the Lagrange inverse problem can be precomputed up-front (as it does not change along iterations). Furthermore, to ensure numerical stability of a solution to the Lagrange inverse problem, we use families of orthogonal basis functions, such as a Chebyshev family.

Second, the conventional Smolyak formula is symmetric in a sense that it has the same number of grid points and basis functions for all variables. To increase the quality of approximation, one must equally increase the number of grid points and basis functions for all variables, which may be costly or even infeasible in large-scale applications. In the paper, we present an anisotropic version of the Smolyak method that allows for asymmetric treatments of variables, namely, it enables us to separately choose the accuracy level for each dimension with the aim of increasing the quality of approximation. In economic applications, variables do not enter symmetrically: decision or value functions may have more curvature in some variables than in others; some variables may have larger ranges of values than others; and finally, some variables may be more important than the others. For example, in heterogeneous-agent economies, an agent's decision functions may depend more on her own capital stock than on the other agents' capital stocks (e.g., Kollmann et al., 2011); also, we may need more grid points for accurate approximation of endogenous than exogenous state variables (e.g., models based on Tauchen and Hussy's, 1991 approximation of shocks). An anisotropic version of the Smolyak method allows us to take into account a specific structure of decision or value functions to solve economic models more efficiently.

Third, the Smolyak method constructs grid points within a normalized multidimensional hypercube. In economic applications, we must also specify how the model's state variables are mapped into the Smolyak hypercube. The way in which this mapping is constructed can dramatically affect the effective size of a solution domain, and hence, the quality of approximation. In the paper, we show how to effectively adapt the Smolyak grid to a solution domain of a given economic model. We specifically construct a parallelotope that encloses a high-probability area of the state space of the given model, and we reduce the size of the parallelotope to minimum by reorienting it with a principle-component transformation of state variables. Judd et al. (2011) find that solution algorithms that focus on "a relevant domain" are more accurate (in that domain) than solution algorithms that focus on a larger (and partially irrelevant) domain and that face therefore a trade off between the fit inside and outside the relevant domain. For the same reason, an adaptive domain increases the accuracy of the Smolyak method.

Finally, the Smolyak method for interpolation is just one ingredient of a numerical method for solving dynamic economic models. In particular, Krueger and Kubler (2004) and Malin et al. (2011) complemented Smolyak interpolation with other computational techniques that are tractable in large-scale problems, such as Chebyshev polynomials, monomial integration and a learning-style procedure for finding polynomial coefficients. Nonetheless, there is one technique – time iteration – that is expensive in their version of their numerical procedure. Time iteration is traditionally used in dynamic programming: given functional forms for future value function, it solves for current value function using a numerical solver. It works similarly in the context of the Euler equation methods: given functional forms for future decision functions, it solves for current decision functions using a numerical solver. However, there is a simple derivative-free alternative to time iteration – fixed point iteration – that can solve large systems of equations using only straightforward calculations, avoiding thus the need of a numerical solver. Fixed-point iteration is commonly used in the context of solution methods for dynamic economic models; see, e.g., Wright and Williams (1984), Miranda and Helmberger (1988), Marcet (1988), Den Haan (1990), Marcet and Lorenzoni (1999), Gaspar and Judd (1997), Judd et al. (2010, 2011), Maliar and Maliar (2014b). We assess how the cost of a Smolyak-based projection method for solving dynamic economic models depends on a specific iterative scheme used. We find that time iteration may dominate fixed-point iteration in small-scale applications but the situation reverses

[1] We provide a MATLAB code that implements the computational techniques developed in the paper. Also, Coleman and Lyon (2013) provide Python and Julia codes that implement some of these techniques.

[2] Our exposition was informed by our personal communication with Sergey Smolyak.

when the dimensionality increases. Therefore, we advocate the use of fixed-point iteration instead of time iteration in large-scale applications.

We assess the performance of the Smolyak-based projection method in the context of one- and multi-agent neoclassical stochastic growth models with up to 20 state variables. Our analysis shows that there are substantial accuracy gains from using anisotropic grid and adaptive domain even in the simplest case with two state variables: the maximum residuals in the Euler equations can be reduced by an order of magnitude compared to those produced by the baseline isotropic Smolyak method with the standard hypercube domain (holding the number of the coefficients roughly the same). In multi-dimensional problems – the real interest of our analysis – the accuracy gains from using anisotropic grid and adaptive domain reach two orders of magnitude in some examples. Our cost grows fairly slowly with the dimensionality of the problem. In particular, our MATLAB code delivers a second-level Smolyak approximation to a model with ten countries (twenty state variables) in about 45 min. For comparison, a Fortran code of Malin et al. (2011), based on the conventional Smolyak method, solves a similar model in about 10 h. Moreover, we are able to produce a very accurate third-level polynomial approximation to a ten-country model; however, such an approximation is computationally demanding even for our efficient implementation of the Smolyak method (our running time increases to nearly 45 h).

The rest of the paper is organized as follows: in Section 2, we illustrate the inefficiency of the conventional Smolyak formula. In Section 3, we introduce an alternative implementation of the Smolyak method based on disjoint-set unidimensional generators and Lagrange interpolation. In Sections 4 and 5, we develop versions of the Smolyak method with anisotropic grid and adaptive domain, respectively. In Section 6, we assess the performance of the Smolyak-based projection method in the context of one- and multi-agent optimal growth models. Finally, in Section 7, we conclude.

2. Conventional Smolyak method for interpolation

In this section, we describe the conventional Smolyak method for interpolation. In Section 2.1, we outline the idea of the Smolyak method and review the related literature. In Sections 2.2, 2.3 and 2.4, we show how to construct the Smolyak grid points, Smolyak polynomial and Smolyak interpolating coefficients, respectively. Finally, in Section 2.5, we argue that the conventional Smolyak method is inefficient.

2.1. Smolyak method at glance

The problem of representing and interpolating multidimensional functions commonly arises in economics. In particular, when solving dynamic economic models, one needs to represent and interpolate decision and value functions in terms of state variables. With few state variables, one can use tensor-product rules but such rules become intractable when the number of state variables increases. For example, if there are five grid points per each variable, a tensor product grid for d variables contains 5^d grid points, which is a large number even for moderately large d. Bellman (1961) referred to exponential growth in complexity as a curse of dimensionality.

Smolyak (1963) introduced a numerical technique for representing multidimensional functions, which is tractable in problems with high dimensionality. The key idea of Smolyak's (1963) analysis is that some elements produced by tensor-product rules are more important for representing multidimensional functions than the others. The Smolyak method orders all elements produced by a tensor-product rule by their potential importance for the quality of approximation and selects a relatively small number of the most important elements. A parameter, called a *level of approximation*, controls how many tensor-product elements are included into the Smolyak grid. By increasing the level of approximation, one can add new elements and improve the quality of approximation.[3]

Examples of Smolyak grids under approximation levels $\mu = 0, 1, 2, 3$ are illustrated in Fig. 1 for the two-dimensional case. For comparison, we also show a tensor-product grid of 5^2 points.

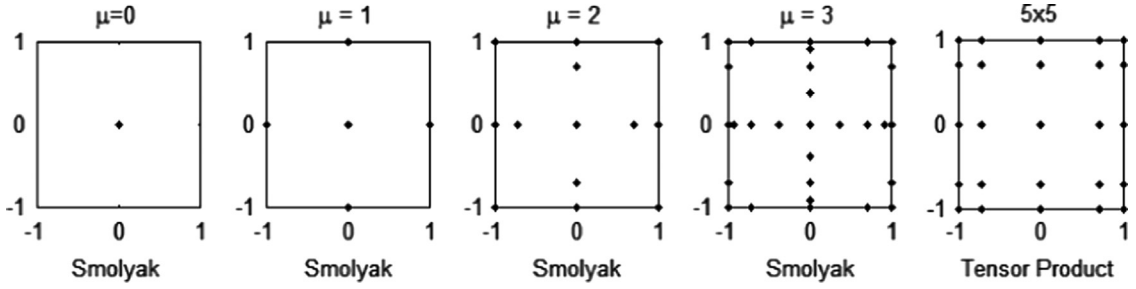

Fig. 1. Smolyak grids versus a tensor-product grid.

[3] The level of approximation plays the same role in the Smolyak construction as the order of expansion in the Taylor series, i.e., we include the terms up to a given order, and we neglect the remaining high-order terms.

Table 1
Number of grid points: tensor-product grid with 5 points in each dimension versus Smolyak grids.

d	Tensor-product grid with 5^d points	Smolyak grid		
		$\mu=1$	$\mu=2$	$\mu=3$
1	5	3	5	9
2	25	5	13	29
10	9,765,625	21	221	1581
20	95, 367, 431, 640, 625	41	841	11,561

In Table 1, we compare the number of points in the Smolyak grid and that in the tensor-product grid with five grid points in each dimension. The number of points in a Smolyak grid grows polynomially with dimensionality d, meaning that the Smolyak method is not subject to the curse of dimensionality. In particular, for $\mu=1$ and $\mu=2$, the number of the Smolyak grid points grows as $1+2d$ and $1+4d+(4d(d-1))/2$, i.e., linearly and quadratically, respectively. A relatively small number of points in Smolyak grids sharply contrasts with a huge number of points in tensor-product grids in a high-dimensional case. Because of this, Smolyak grids are also called *sparse grids*.

To interpolate multidimensional functions off the Smolyak grid, two broad classes of interpolants are used in mathematical literature. One class uses hierarchical surplus formulas and piecewise local basis functions; see, e.g., Griebel (1998), Bungartz and Griebel (2004), Klimke and Wohlmutch (2005) and Jakeman and Roberts (2011) for related mathematical results, and see Brumm and Scheidegger (2013) for an economic application. Piecewise functions are very flexible and make it possible to vary the quality of approximations over different areas of the state space as needed. However, the resulting approximations are non-smooth and non-differentiable and also, they have a high computational expense (this interpolation technique is still subject to the curse of dimensionality).

The other class of Smolyak interpolants includes global polynomial functions; see, e.g., Delvos (1982), Wasilkowski and Woźniakowski (1995) and Barthelmann et al. (2000) for a mathematical background. Global polynomial approximations are smooth and continuously differentiable and also, they are relatively inexpensive. However, their flexibility and adaptivity are limited. In economics, global polynomial approximation is used in Krueger and Kubler (2004), Gavilan-Gonzalez and Rojas (2009), Winschel and Krätzig (2010), Gordon (2011), Malin et al. (2011) and Fernández-Villaverde et al. (2012). In the present paper, we also confine our attention to global polynomial approximations. Below, we show the conventional Smolyak method for interpolation in line with Malin et al. (2011).

2.2. Construction of Smolyak grids using unidimensional nested sets

To construct a Smolyak grid, we generate unidimensional sets of grid points, construct tensor products of unidimensional sets and select a subsets of grid points satisfying the Smolyak rule.

2.2.1. Unidimensional nested sets of points

The Smolyak construction begins with one dimension. To generate unidimensional grid points, we use extrema of Chebyshev polynomials (also known as Chebyshev–Gauss–Lobatto points or Clenshaw–Curtis points); see Appendix A. We do not use all consecutive extrema but those that form a sequence $S_1, S_2, \ldots$ satisfying two conditions:

Condition 1: A set S_i, $i=1,2,\ldots$, has $m(i)=2^{i-1}+1$ points for $i\geq 2$ and $m(1)\equiv 1$.

Condition 2: Each subsequent set contains all points of the previous set, $S_i \subset S_{i+1}$. Such sets are called *nested*.[4]

Below, we show the first four nested sets composed of extrema of Chebyshev polynomials:

$$i=1: S_1=\{0\};$$

$$i=2: S_2=\{0,-1,1\};$$

$$i=3: S_3=\left\{0,-1,1,\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right\};$$

$$i=4: S_4=\left\{0,-1,1,\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}},\frac{-\sqrt{2+\sqrt{2}}}{2},\frac{-\sqrt{2-\sqrt{2}}}{2},\frac{\sqrt{2-\sqrt{2}}}{2},\frac{\sqrt{2+\sqrt{2}}}{2}\right\}.$$

[4] There are many other ways to construct sets of points that have a nested structure. For example, we can use subsets of equidistant points; see Appendix A for a discussion. Gauss–Patterson points also lead to nested sets, however, the number of points in such sets is different, namely, $m(i)=2^{i-1}-1$; see Patterson (1968).

2.2.2. Tensor products of unidimensional nested sets of points

Next, we construct tensor products of unidimensional sets of points. As an illustration, we consider a two-dimensional case with $i=1,2,3$ in each dimension.

In Table 2, i_1 and i_2 are indices that correspond to dimensions one and two, respectively; a column S_{i_1} and a raw S_{i_2} (see $S_{i_1}|S_{i_2}$) show the sets of unidimensional elements that correspond to dimensions one and two, respectively.

2.2.3. Smolyak sparse grids

Smolyak (1963) offers a rule that tells us which tensor products must be selected from the table. For the two-dimensional case, we must select tensor products (cells of Table 2) for which the following condition is satisfied:

$$d \le i_1 + i_2 \le d + \mu, \tag{1}$$

where $\mu \in \{0,1,2,\ldots\}$ is the approximation level, and d is the dimensionality (in Table 2, $d=2$). In other words, the sum of a column i_1 and a raw i_2 must be between d and $d+\mu$.

Let $\mathcal{H}^{d,\mu}$ denote a Smolyak grid for a problem with dimensionality d and approximation level μ. Let us construct Smolyak grids for $\mu=0,1,2$ and $d=2$ using Smolyak rule (1).

- If $\mu=0$, then $2 \le i_1+i_2 \le 2$. The only cell in Table 2 that satisfies this restriction is $i_1=1$ and $i_2=1$, so that the Smolyak grid has just one grid point,
$$\mathcal{H}^{2,0} = \{(0,0)\}. \tag{2}$$

- If $\mu=1$, then $2 \le i_1+i_2 \le 3$. The three cells that satisfy this restriction are (a) $i_1=1$, $i_2=1$; (b) $i_1=1$, $i_2=2$; (c) $i_1=2$, $i_2=1$, and the corresponding five Smolyak grid points are
$$\mathcal{H}^{2,1} = \{(0,0),(-1,0),(1,0),(0,-1),(0,1)\}. \tag{3}$$

- If $\mu=2$, then $2 \le i_1+i_2 \le 4$. There are six cells that satisfy this restriction: (a) $i_1=1$, $i_2=1$; (b) $i_1=1$, $i_2=2$; (c) $i_1=2$, $i_2=1$; (d) $i_1=1$, $i_2=3$; (e) $i_1=2$, $i_2=2$; (f) $i_1=3$, $i_2=1$, and there are thirteen Smolyak grid points,
$$\mathcal{H}^{2,2} = \Big\{(-1,1),(0,1),(1,1),(-1,0),(0,0),(1,0),(-1,-1),(0,-1),(1,-1), \left(\frac{-1}{\sqrt{2}},0\right),\left(\frac{1}{\sqrt{2}},0\right),\left(0,\frac{-1}{\sqrt{2}}\right),\left(0,\frac{1}{\sqrt{2}}\right)\Big\}. \tag{4}$$

Smolyak grids $\mathcal{H}^{2,0}$, $\mathcal{H}^{2,1}$ and $\mathcal{H}^{2,2}$ are those that are shown in the first three subplots of Fig. 1.

Table 2
Tensor products of disjoint sets of unidimensional grid points for the two-dimensional case.

	$S_{i_1}\|S_{i_2}$	$i_2=1$	$i_2=2$	$i_2=3$
		0	$0,-1,1$	$0,-1,1,\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}$
$i_1=1$	0	$(0,0)$	$(0,0),(0,-1),(0,1)$	$(0,0),(0,-1),(0,1),\left(0,\frac{-1}{\sqrt{2}}\right),\left(0,\frac{1}{\sqrt{2}}\right)$
$i_1=2$	0 −1 1	$(0,0)$ $(-1,0)$ $(1,0)$	$(0,0),(0,-1),(0,1)$ $(-1,0),(-1,-1),(-1,1)$ $(1,0,),(1,-1),(1,1)$	$(0,0),(0,-1),(0,1),\left(0,\frac{-1}{\sqrt{2}}\right),\left(0,\frac{1}{\sqrt{2}}\right)$ $(-1,0),(-1,-1),(-1,1),\left(-1,\frac{-1}{\sqrt{2}}\right),\left(-1,\frac{1}{\sqrt{2}}\right)$ $(1,0),(1,-1),(1,1),\left(1,\frac{-1}{\sqrt{2}}\right),\left(1,\frac{1}{\sqrt{2}}\right)$
$i_1=3$	0 −1 1 $\frac{-1}{\sqrt{2}}$ $\frac{1}{\sqrt{2}}$	$(0,0)$ $(-1,0)$ $(1,0)$ $\left(\frac{-1}{\sqrt{2}},0\right)$ $\left(\frac{1}{\sqrt{2}},0\right)$	$(0,0),(0,-1),(0,1)$ $(-1,0),(-1,-1),(-1,1)$ $(1,0),(1,-1),(1,1)$ $\left(\frac{-1}{\sqrt{2}},0\right),\left(\frac{-1}{\sqrt{2}},-1\right),\left(\frac{-1}{\sqrt{2}},1\right)$ $\left(\frac{1}{\sqrt{2}},0\right),\left(\frac{1}{\sqrt{2}},-1\right),\left(\frac{1}{\sqrt{2}},1\right)$	$(0,0),(0,-1),(,1),\left(0,\frac{-1}{\sqrt{2}}\right),\left(0,\frac{1}{\sqrt{2}}\right)$ $(-1,0),(-1,-1),(-1,1),\left(-1,\frac{-1}{\sqrt{2}}\right),\left(-1,\frac{1}{\sqrt{2}}\right)$ $(1,0),(1,-1),(1,1),\left(1,\frac{-1}{\sqrt{2}}\right),\left(1,\frac{1}{\sqrt{2}}\right)$ $\left(\frac{-1}{\sqrt{2}},0\right),\left(\frac{-1}{\sqrt{2}},-1\right),\left(\frac{-1}{\sqrt{2}},1\right),\left(\frac{-1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right),\left(\frac{-1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$ $\left(\frac{1}{\sqrt{2}},0\right),\left(\frac{1}{\sqrt{2}},-1\right),\left(\frac{1}{\sqrt{2}},1\right),\left(\frac{1}{\sqrt{2}},\frac{-1}{\sqrt{2}}\right),\left(\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}\right)$

2.3. Smolyak formula for interpolation using unidimensional nested sets

The conventional technique for constructing a Smolyak polynomial function also builds on unidimensional nested sets and mimics the construction of a Smolyak grid.

2.3.1. Smolyak polynomial

Let $\widehat{f}^{d,\mu}$ denote a Smolyak polynomial function (interpolant) in dimension d, with approximation level μ. The Smolyak interpolant is a linear combination of tensor-product operators $p^{|i|}$ given by

$$\widehat{f}^{d,\mu}(x_1,\ldots,x_d;b)=\sum_{\max(d,\mu+1)\le|i|\le d+\mu}(-1)^{d+\mu-|i|}\binom{d-1}{d+\mu-|i|}p^{|i|}(x_1,\ldots,x_d), \tag{5}$$

where $|i|$ is defined as $|i|\equiv i_1+\cdots+i_d$ and $(-1)^{d+\mu-|i|}\binom{d-1}{d+\mu-|i|}$ is a counting coefficient. For each $|i|$ satisfying $\max(d,\mu+1)\le|i|\le d+\mu$, a tensor-product operator $p^{|i|}(x_1,\ldots,x_d)$ is defined as

$$p^{|i|}(x_1,\ldots,x_d)=\sum_{i_1+\cdots+i_d=|i|}p^{i_1,\ldots,i_d}(x_1,\ldots,x_d), \tag{6}$$

and $p^{i_1,\ldots,i_d}$ is defined as

$$p^{i_1,\ldots,i_d}(x_1,\ldots,x_d)=\sum_{\ell_1=1}^{m(i_1)}\cdots\sum_{\ell_d=1}^{m(i_d)}b_{\ell_1\ldots\ell_d}\psi_{\ell_1}(x_1)\ldots\psi_{\ell_d}(x_d), \tag{7}$$

where $m(i_j)$ is the number of basis functions in dimension j, with $m(i_j)\equiv 2^{i_j-1}+1$ for $i_j\ge 2$ and $m(1)\equiv 1$; $\psi_{\ell_j}(x_j)$ is a ℓ_jth unidimensional basis function in dimension j with $\ell_j=1,\ldots,m(i_j)$; $\psi_{\ell_1}(x_1)\cdots\psi_{\ell_d}(x_d)$ is a d-dimensional basis function and $b_{\ell_1\ldots\ell_d}$s are the corresponding polynomial coefficients.

2.3.2. Example of Smolyak polynomial under $d=2$ and $\mu=1$

We now illustrate the construction of Smolyak polynomial function (5) under $d=2$ and $\mu=1$; in Appendix B, we show the construction of such a function under $d=2$ and $\mu=2$.

For the case of $\mu=1$, we have that $2\le|i|\le 3$. This is satisfied in three cases: (a) $i_1=i_2=1$; (b) $i_1=1$, $i_2=2$; (c) $i_1=2$, $i_2=1$. From (7), we obtain

$$\text{(a)}\quad p^{1,1}=\sum_{\ell_1=1}^{m(1)}\sum_{\ell_2=1}^{m(1)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}, \tag{8}$$

$$\text{(b)}\quad p^{1,2}=\sum_{\ell_1=1}^{m(1)}\sum_{\ell_2=1}^{m(2)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}+b_{12}\psi_2(y)+b_{13}\psi_3(y), \tag{9}$$

$$\text{(c)}\quad p^{2,1}=\sum_{\ell_1=1}^{m(2)}\sum_{\ell_2=1}^{m(1)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x), \tag{10}$$

where we assume that $\psi_1(x)=\psi_1(y)\equiv 1$. Collecting the elements p^{i_1,i_2} with the same sum $i_1+i_2\equiv|i|$, we obtain

$$p^{|2|}\equiv p^{1,1}, \tag{11}$$

$$p^{|3|}\equiv p^{2,1}+p^{1,2}. \tag{12}$$

Smolyak polynomial function (5) for the case of $d=2$ and $\mu=1$ is given by

$$\begin{aligned}\widehat{f}^{2,1}(x,y;b)&=\sum_{\max(d,\mu+1)\le|i|\le d+\mu}(-1)^{d+\mu-|i|}\binom{d-1}{d+\mu-|i|}p^{|i|}\\&=\sum_{2\le|i|\le 3}(-1)^{3-|i|}\binom{1}{3-|i|}p^{|i|}=\sum_{2\le|i|\le 3}(-1)^{3-|i|}\frac{1}{(3-|i|)!}p^{|i|}\\&=(-1)\cdot p^{|2|}+1\cdot p^{|3|}\\&=(-1)\cdot p^{1,1}+1\cdot(p^{2,1}+p^{1,2})\\&=-b_{11}+b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{11}+b_{12}\psi_2(y)+b_{13}\psi_3(y)\\&=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{12}\psi_2(y)+b_{13}\psi_3(y).\end{aligned} \tag{13}$$

By construction, the number of basis functions in Smolyak polynomial $\widehat{f}^{2,1}(x,y;b)$ is equal to the number of points in Smolyak grid $\mathcal{H}^{2,1}$. The same is true for a Smolyak grid $\mathcal{H}^{d,\mu}$ and Smolyak polynomial $\widehat{f}^{d,\mu}$ under any $d\ge 1$ and $\mu\ge 0$.

2.4. Smolyak interpolation coefficients

Polynomial coefficients $b_{\ell_1\ldots\ell_d}$s in (5) must be constructed so that Smolyak polynomial $\widehat{f}^{d,\mu}$ matches the true function f in all points of Smolyak grid $H^{d,\mu}$. In the present paper, we construct multidimensional Smolyak polynomials using unidimensional Chebyshev polynomial basis functions.

2.4.1. Closed-form expression for Smolyak interpolation coefficients

There is a closed-form formula for the polynomial coefficients in (5) if multidimensional Smolyak grid points and basis functions are constructed using unidimensional Chebyshev polynomials and their extrema, respectively; see Quarteroni et al. (2000) for a derivation of such formulas. Consider a grid that has $m(i_1),\ldots,m(i_d)$ grid points and basis functions in dimensions $1,\ldots,d$, respectively. Then, the corresponding coefficients are given by

$$b_{\ell_1\ldots\ell_d}=\frac{2^d}{(m(i_1)-1)\cdots(m(i_d)-1)}\cdot\frac{1}{c_{\ell_1}\cdots c_{\ell_d}}\times\sum_{j_1=1}^{m(i_1)}\cdots\sum_{j_d=1}^{m(i_d)}\frac{\psi_{\ell_1}(\zeta_{j_1})\cdots\psi_{\ell_d}(\zeta_{j_d})\cdot f(\zeta_{j_1},\ldots,\zeta_{j_d})}{c_{j_1}\cdots c_{j_d}},\tag{14}$$

where $\zeta_{j_1},\ldots,\zeta_{j_d}$ are grid points in dimensions $j_1,\ldots,j_d$, respectively; $c_2=1$; $c_2=c_3=2$; and $c_j=1$ for $j=4,\ldots,m(i_d)$. If along any dimension d, we have $m(i_d)=1$, this dimension is dropped from computation, i.e., $m(i_d)-1$ is set to 1 and $c_{j_d}=c_1$ is set to 1.

We shall comment on the following notational issue. We ordered the Smolyak elements so that we do not need to distinguish between grid points and basis functions corresponding to different levels of approximation. This simplifies notation considerably. For example, for $i_d=1$, the node is $\zeta_1^1=0$, and for $i_2=2$, we write the nodes as $\{\zeta_1^2,\zeta_2^2,\zeta_3^2\}=\{0,-1,1\}$; we set $\zeta_1^1=\zeta_1^2\equiv\zeta_1$ and we omit the approximation-level superscript. We do the same for the basis functions, for example, we set $\psi_1^1=\psi_1^2\equiv\psi_1$, etc. Finally, we define $c_{j_1}\cdots c_{j_d}$ in a way that allows us to omit the approximation-level superscript. Note that under other orderings of Smolyak elements we need to keep the approximation-level superscript, for example, if we set $\zeta_1^1=0$, and $\{\zeta_1^2,\zeta_2^2,\zeta_3^2\}=\{-1,0,1\}$, then we have $\zeta_1^1\neq\zeta_1^2$.

2.4.2. Example of the Smolyak coefficients under $d=2$ and $\mu=1$

We must compute the coefficients $\{b_{11},b_{21},b_{31},b_{12},b_{13}\}$ so that polynomial function $\widehat{f}^{2,1}$, given by (13), matches true function f on Smolyak grid $\mathcal{H}^{2,1}$ given by (3). For $\mu=1$, the set of Chebyshev polynomial basis are $\{\psi_1(x),\psi_2(x),\psi_3(x)\}=\{1,x,2x^2-1\}$ (and we have the same polynomial basis for y, namely, $\{1,y,2y^2-1\}$); the extrema of Chebyshev polynomials are $\{\zeta_1,\zeta_2,\zeta_3\}=\{0,-1,1\}$; and finally, we have $c_1=1$, $c_2=c_3=2$.

For b_{21}, formula (14) implies

$$b_{21}=\frac{2}{3-1}\cdot\frac{1}{c_2}\sum_{j_1=1}^{3}\frac{\psi_2(\zeta_{j_1})\cdot f(\zeta_{j_1},0)}{c_{j_1}\cdot 1}=\frac{\psi_2(\zeta_1)\cdot f(\zeta_1,0)}{c_1}+\frac{\psi_2(\zeta_2)\cdot f(\zeta_2,0)}{c_2}+\frac{\psi_2(\zeta_3)\cdot f(\zeta_3,0)}{c_3}=\frac{-1\cdot f(-1,0)}{2}+\frac{1\cdot f(1,0)}{2};$$

similarly, for b_{12}, we get

$$b_{12}=-\frac{f(0,-1)}{2}+\frac{f(0,1)}{2}.$$

Coefficient b_{31} is given by

$$b_{31}=\frac{2}{3-1}\cdot\frac{1}{c_3}\sum_{j_1=1}^{3}\frac{\psi_3(\zeta_{j_1})\cdot f(\zeta_{j_1},0)}{c_{j_1}}=\frac{1}{2}\left[\frac{1\cdot f(-1,0)}{2}-f(0,0)+\frac{1\cdot f(1,0)}{2}\right]=-\frac{f(0,0)}{2}+\frac{f(-1,0)+f(1,0)}{4},$$

and b_{13} is obtained similarly

$$b_{13}=-\frac{f(0,0)}{2}+\frac{f(0,-1)+f(0,1)}{4}.$$

Formula (14) does not apply to a constant term b_{11}. To find b_{11}, observe that (13) implies

$$\widehat{f}^{2,1}(0,0;b)=b_{11}+\frac{f(0,0)}{2}-\frac{f(-1,0)+f(1,0)}{4}+\frac{f(0,0)}{2}-\frac{f(0,-1)+f(0,1)}{4}.$$

Since under interpolation, we must have $\widehat{f}^{2,1}(0,0;b)=f(0,0)$, the last formula yields

$$b_{11}=\frac{f(-1,0)+f(1,0)+f(0,-1)+f(0,1)}{4}.$$

Note that to compute the coefficients, we need to evaluate function f in five Smolyak grid points of $\mathcal{H}^{2,1}$.

2.5. Shortcomings of the conventional Smolyak method

The conventional Smolyak method using nested sets is inefficient. First, it creates a list of tensor products with many repeated elements and then, it eliminates the repetitions. In high-dimensional applications, the

number of repetitions is large and increases with both μ and d, which leads to a considerable increase in computational expense.

Repetitions of grid points can be appreciated by looking at Table 2. For example, when constructing $\mathcal{H}^{2,1}$, we list a grid point $(0,0)$ in three different cells, and hence, we must eliminate two grid points out of seven; when constructing $\mathcal{H}^{2,2}$, we must eliminate thirteen repeated points out of twenty six points, etc. However, the repeated grid points are not a critical issue for computational expense. This is because grid points must be constructed just once (it is a one-time fixed cost), and it is not so important if they are constructed efficiently or not.

Unfortunately, the Smolyak formula (5) involves the same kind of repetitions, and it is not a fixed cost. For example, $\widehat{f}^{2,1}$, given by (13), lists seven basis functions $\{1\}$, $\{1,\psi_2(x),\psi_3(x)\}$ and $\{1,\psi_2(y),\psi_3(y)\}$ in (8)–(10), respectively, and eliminates two repeated functions $\{1\}$ by assigning a weight (-1) to $p^{|2|}$; furthermore, $\widehat{f}^{2,2}$, derived in Appendix B, creates a list of twenty six basis functions and removes thirteen repeated basis function by assigning appropriate weights, etc. We suffer from repetitions every time we evaluate a Smolyak polynomial function. This is an especially important issue in the context of numerical methods for solving dynamic economic models, since we must interpolate decision and value functions in a very large number of points, e.g., grid points, integration nodes and time periods. Moreover, we must repeat interpolation each time when the decision and value functions change in the iterative cycle. The overall cost of repetitions in the Smolyak formula can be very large.

3. Efficient implementation of the Smolyak method for interpolation

We have argued that the Smolyak (1963) sparse-grid structure is an efficient choice for high-dimensional interpolation. However, the existing implementation of the Smolyak method does not arrive to this structure directly. Instead, it produces such a structure using a linear combinations of sets with repeated elements, which is inefficient and expensive. In this section, we propose a more efficient implementation of the Smolyak method that avoids costly repetitions of elements and arrives to the Smolyak structure directly. Our key novelty is to replace the conventional nested-set generators with equivalent disjoint-set generators. We use the disjoint-set generators not only for constructing Smolyak grids but also for constructing Smolyak basis functions; as a result, we do not make use of the conventional interpolation formula of type (7). Furthermore, to identify the interpolating coefficients, we use a canonical Lagrange interpolation; thus, we also do not make use of formula (14) for the coefficient. We find it easiest to present our implementation of the Smolyak method starting from a description of the Lagrange interpolation framework.

3.1. Multidimensional Lagrange interpolation

We consider the following interpolation problem. Let $f:[-1,1]^d\to\mathbb{R}$ be a smooth function defined on a normalized d-dimensional hypercube, and let $\widehat{f}(\cdot;b)$ be a polynomial function of the form

$$\widehat{f}(x;b)=\sum_{n=1}^{M} b_n\Psi_n(x), \tag{15}$$

where $\Psi_n:[-1,1]^d\to\mathbb{R}$, $n=1,\ldots,M$, are d-dimensional basis functions, and $b\equiv(b_1,\ldots,b_M)$ is a coefficient vector.

We construct a set of M grid points $\{x_1,\ldots,x_M\}$ in $[-1,1]^d$, and we compute b so that the true function, f, and its approximation, $\widehat{f}(\cdot;b)$ coincide in all grid points:

$$\begin{bmatrix} f(x_1)\\ \cdots\\ f(x_M)\end{bmatrix}=\begin{bmatrix}\widehat{f}(x_1;b)\\ \cdots\\ \widehat{f}(x_M;b)\end{bmatrix}=\begin{bmatrix}\Psi_1(x_1) & \cdots & \Psi_M(x_1)\\ \cdots & \ddots & \cdots\\ \Psi_1(x_M) & \cdots & \Psi_M(x_M)\end{bmatrix}\cdot\begin{bmatrix} b_1\\ \cdots\\ b_M\end{bmatrix}. \tag{16}$$

Approximation $\widehat{f}(\cdot;b)$ is used to interpolate (infer, reconstruct) f in any point $x\in[-1,1]^d$.

To implement the above interpolation method, we must perform three steps:

(i) Choose M grid points $\{x_1,\ldots,x_M\}$.

(ii) Choose M basis functions for forming $\widehat{f}(x;b)$.

(iii) Compute b that makes f and $\widehat{f}(\cdot;b)$ coincide in all grid points.

Lagrange interpolation allows for many different choices of grid points and basis functions. We will use grid points and basis functions produced by the Smolyak method. In Section 3.2, we construct the Smolyak grid points; in Section 3.3, we produce the Smolyak basis functions; in Section 3.4, we identify the interpolation coefficients; in Section 3.5, we compare our implementation of the Smolyak method with the conventional implementation described in Section 2; and finally, in Section 3.6, we show an efficient formula for Smolyak interpolation.

3.2. Construction of Smolyak grids using unidimensional disjoint sets

To construct a Smolyak grid, we proceed as in the conventional Smolyak method, namely, we produce sets of unidimensional grid points, compute tensor products of such sets and select an appropriate subsets of tensor-product

elements for constructing a multidimensional grid. The difference is that we operate with unidimensional disjoint sets instead of unidimensional nested sets. This allows us to avoid repetitions of grid points.

3.2.1. Unidimensional disjoint sets of grid points

Let us define a sequence of disjoint sets $A_1, A_2, \ldots$ using the sequence of nested sets $S_1, S_2, \ldots$ of Section 2.2.1 such that $A_1 = S_1$ and $A_i = S_i \backslash S_{i-1}$ for $i \geq 2$:

$$i = 1: A_1 = \{0\};$$

$$i = 2: A_2 = \{-1, 1\};$$

$$i = 3: A_3 = \left\{\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right\};$$

$$i = 4: A_4 = \left\{\frac{-\sqrt{2+\sqrt{2}}}{2}, \frac{-\sqrt{2-\sqrt{2}}}{2}, \frac{\sqrt{2-\sqrt{2}}}{2}, \frac{\sqrt{2+\sqrt{2}}}{2}\right\}.$$

By definition, A_i is a set of points in S_i but not in S_{i-1}, i.e., $S_i \backslash S_{i-1}$ The constructed sets are disjoint, $A_i \cap A_j = \{\varnothing\}$ for any $i \neq j$ and their unions satisfy $A_1 \cup \cdots \cup A_i = S_i$. The number of elements in A_i is $m(i) - m(i-1) = 2^{i-2}$ points for $i \geq 3$, and the number of elements in A_1 and A_2 is 1 and 2, respectively.

3.2.2. Tensor products of unidimensional disjoint sets of points

Next, we construct tensor products of disjoint sets of unidimensional grid points. Again, we consider the two-dimensional case, with $i = 1, 2, 3$ in each dimension.

In Table 3, i_1 and i_2 are indices that correspond to dimensions one and two, respectively; a column A_{i_1} and a raw A_{i_2} (see $A_{i_1} \backslash A_{i_2}$) show the sets of unidimensional elements that correspond to dimensions one and two, respectively; (ζ_ℓ, ζ_n) denotes a two-dimensional grid point obtained by combining a grid point ζ_ℓ in dimension one and a grid point ζ_n in dimension two. Thus, the table shows incremental grid points, and we can easily see which grid points are added when we increase the approximation level.

3.2.3. Smolyak sparse grids

We use the same Smolyak rule (1) for constructing multidimensional grid points. That is, we select elements that belong to the cells in Table 3 for which the sum of indices of a column and a raw, $i_1 + i_2$, is between d and $d+\mu$. This leads to the same Smolyak grids $\mathcal{H}^{2,0}$, $\mathcal{H}^{2,1}$ and $\mathcal{H}^{2,2}$ as is shown in (2), (3), and (4), respectively. However, in our case, no grid point is repeated in Table 3. Furthermore, note that the multidimensional grids $\mathcal{H}^{2,0}$, $\mathcal{H}^{2,1}$ and $\mathcal{H}^{2,2}$ are nested $\mathcal{H}^{2,0} \subset \mathcal{H}^{2,1} \subset \mathcal{H}^{2,2}$ even though their unidimensional generators are disjoint (i.e., not nested).

3.3. Construction of Smolyak polynomials using unidimensional disjoint sets

Our construction of Smolyak polynomials parallels our construction of Smolyak grids using unidimensional disjoint sets. To be specific, we produce disjoint sets of unidimensional basis functions, compute tensor products of such sets and select an appropriate subset of tensor-product elements for constructing a multidimensional polynomial function. Again, using disjoint-set generators instead of nested-set generators allows us to avoid repetitions of basis functions.

Table 3
Tensor products of disjoint sets of unidimensional grid points for the two-dimensional case.

	$A_{i_1} \vert A_{i_2}$	$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
		0	$-1, 1$	$\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}$
$i_1 = 1$	0	$(0,0)$	$(0,-1), (0,1)$	$\left(0, \frac{-1}{\sqrt{2}}\right), \left(0, \frac{1}{\sqrt{2}}\right)$
$i_1 = 2$	-1	$(-1,0)$	$(-1,-1), (-1,1)$	$\left(-1, \frac{-1}{\sqrt{2}}\right), \left(-1, \frac{1}{\sqrt{2}}\right)$
	1	$(1,0)$	$(1,-1), (1,1)$	$\left(1, \frac{-1}{\sqrt{2}}\right), \left(1, \frac{1}{\sqrt{2}}\right)$
$i_1 = 3$	$\frac{-1}{\sqrt{2}}$	$\left(\frac{-1}{\sqrt{2}}, 0\right)$	$\left(\frac{-1}{\sqrt{2}}, -1\right), \left(\frac{-1}{\sqrt{2}}, 1\right)$	$\left(\frac{-1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right), \left(\frac{-1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$
	$\frac{1}{\sqrt{2}}$	$\left(\frac{1}{\sqrt{2}}, 0\right)$	$\left(\frac{1}{\sqrt{2}}, -1\right), \left(\frac{1}{\sqrt{2}}, 1\right)$	$\left(\frac{1}{\sqrt{2}}, \frac{-1}{\sqrt{2}}\right), \left(\frac{1}{\sqrt{2}}, \frac{1}{\sqrt{2}}\right)$

3.3.1. Unidimensional disjoint sets of basis functions

We first construct disjoint sets $F_1, \ldots, F_i, \ldots$ that contain unidimensional basis functions:

$i = 1: F_1 = \{1\};$

$i = 2: F_2 = \{\psi_2(x), \psi_3(x)\};$

$i = 3: F_3 = \{\psi_4(x), \psi_5(x)\}.$

$i = 4: F_4 = \{\psi_6(x), \psi_7(x), \psi_8(x), \psi_9(x)\}.$

3.3.2. Tensor products of unidimensional disjoint sets of basis functions

We next construct the two-dimensional basis functions using tensor products of unidimensional basis functions.

By construction, all elements in Table 4 appear just once and therefore, are non-repeated. Note that Table 4 looks exactly like Table 3 (but for basis functions).

3.3.3. Smolyak polynomial basis functions

We apply the same Smolyak rule (1) to produce a list of basis function as we used for producing grid points. Let $\mathcal{P}^{d,\mu}$ denote a Smolyak basis function with dimensionality d and approximation level μ.

- If $\mu = 0$, then $2 \le i_1 + i_2 \le 2$. The only cell that satisfies this restriction is $i_1 = 1$ and $i_2 = 1$, so that the set of Smolyak basis functions has just one element
$$\mathcal{P}^{2,0} = \{1\}. \tag{17}$$

- If $\mu = 1$, then $2 \le i_1 + i_2 \le 3$. The three cells that satisfy this restriction are (a) $i_1 = 1$, $i_2 = 1$; (b) $i_1 = 1$, $i_2 = 2$; (c) $i_1 = 2$, $i_2 = 1$, and the corresponding five Smolyak basis functions are
$$\mathcal{P}^{2,1} = \{1, \psi_2(x), \psi_3(x), \psi_2(y), \psi_3(y)\}. \tag{18}$$

- If $\mu = 2$, then $2 \le i_1 + i_2 \le 4$. There are six cells that satisfy this restriction: (a) $i_1 = 1$, $i_2 = 1$; (b) $i_1 = 1$, $i_2 = 2$; (c) $i_1 = 2$, $i_2 = 1$; (d) $i_1 = 1$, $i_2 = 3$; (e) $i_1 =$, $i_2 = 2$; (f) $i_1 = 3$, $i_2 = 1$, and there are thirteen Smolyak basis functions
$$\mathcal{P}^{2,2} = \{1, \psi_2(x), \psi_3(x), \psi_2(y), \psi_3(y), \psi_4(x), \psi_5(x), \psi_4(y), \psi_5(y)\psi_2(x)\psi_2(y), \psi_2(x)\psi_3(y), \psi_3(x)\psi_2(y), \psi_3(x)\psi_3(y)\}. \tag{19}$$

The sets of Smolyak basis functions $\mathcal{P}^{2,0}$, $\mathcal{P}^{2,1}$ and $\mathcal{P}^{2,2}$ defined in (17), (18), and (19) are used, respectively, with the grids $\mathcal{H}^{2,0}$, $\mathcal{H}^{2,1}$ and $\mathcal{H}^{2,2}$ defined in (2), (3) and (4). To form a Smolyak polynomial function, one just needs to use the elements satisfying condition (1) since no element is repeated in Table 4 by construction.

3.4. Construction of Smolyak coefficients using Lagrange interpolation

Recall that coefficients $b_{\ell_1 \ldots \ell_d}$s in (5) must be constructed so that the Smolyak polynomial $\widehat{f}^{d,\mu}$ matches the true function f on the Smolyak grid $\mathcal{H}^{d,\mu}$. We construct the Smolyak interpolating coefficients solving the inverse problem (16) numerically.

Table 4
Tensor products of disjoint sets of Chebyshev polynomial basis for the two-dimensional case.

	$F_{i_1} \vert F_{i_2}$	$i_2 = 1$	$i_2 = 2$	$i_2 = 3$
		1	$\psi_2(y),\ \psi_3(y)$	$\psi_4(y),\ \psi_5(y)$
$i_1 = 1$	1	1	$\psi_2(y), \psi_3(y)$	$\psi_4(y), \psi_5(y)$
$i_1 = 2$	$\psi_2(x)$ $\psi_3(x)$	$\psi_2(x)$ $\psi_3(x)$	$\psi_2(x)\psi_2(y), \psi_2(x)\psi_3(y)$ $\psi_3(x)\psi_2(y), \psi_3(x)\psi_3(y)$	$\psi_2(x)\psi_4(y), \psi_2(x)\psi_5(y)$ $\psi_3(x)\psi_4(y), \psi_3(x)\psi_5(y)$
$i_1 = 3$	$\psi_4(x)$ $\psi_5(x)$	$\psi_4(x)$ $\psi_5(x)$	$\psi_4(x)\psi_2(y), \psi_4(x)\psi_3(y)$ $\psi_5(x)\psi_2(y), \psi_5(x)\psi_3(y)$	$\psi_4(x)\psi_4(y), \psi_4(x)\psi_5(y)$ $\psi_5(x)\psi_4(y), \psi_5(x)\psi_5(y)$

3.4.1. Solution to the inverse problem

Provided that the matrix of basis functions in the right side of (16) has full rank, we obtain a system of M linear equations with M unknowns that admits a unique solution for b:

$$\begin{bmatrix} b_1 \\ \cdots \\ b_M \end{bmatrix} = \begin{bmatrix} \Psi_1(x_1) & \cdots & \Psi_M(x_1) \\ \cdots & \ddots & \cdots \\ \Psi_1(x_M) & \cdots & \Psi_M(x_M) \end{bmatrix}^{-1} \begin{bmatrix} f(x_1) \\ \cdots \\ f(x_M) \end{bmatrix}. \tag{20}$$

By construction, the approximating polynomial $\widehat{f}$ coincides with the true function f in all grid points, i.e., $\widehat{f}(x_n; b) = f(x_n)$ for all $x_n \in \{x_1, \ldots, x_M\}$.

3.4.2. Example of interpolation coefficients under $d=2$ and $\mu=1$ revisited

Let us now construct the Smolyak polynomial coefficients under $d=2$ and $\mu = 1$ by solving the inverse problem as shown in (20). We again use unidimensional Chebyshev polynomials and extrema of Chebyshev polynomials. As follows from (18), the Smolyak polynomial function is given by

$$\widehat{f}(x, y; b) \equiv b_{11} \cdot 1 + b_{21}x + b_{31}(2x^2 - 1) + b_{12}y + b_{13}(2y^2 - 1), \tag{21}$$

where $b \equiv (b_{11}, b_{21}, b_{31}, b_{12}, b_{13})$ is a vector of five unknown coefficients on five basis functions. We identify the coefficients such that the approximation $\widehat{f}(x, y; b)$ matches the true function $f(x, y)$ in five Smolyak grid points distinguished in (3), namely, $\{(0,0), (-1,0), (1,0), (0,-1), (0,1)\}$.

This yields a system of linear equations $Bb = w$, where

$$\mathcal{B} \equiv \begin{bmatrix} 1 & 0 & -1 & 0 & -1 \\ 1 & -1 & 1 & 0 & -1 \\ 1 & 1 & 1 & 0 & -1 \\ 1 & 0 & -1 & -1 & 1 \\ 1 & 0 & -1 & 1 & 1 \end{bmatrix}; \quad b \equiv \begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{13} \end{bmatrix}; \quad w \equiv \begin{bmatrix} f(0,0) \\ f(-1,0) \\ f(1,0) \\ f(0,-1) \\ f(0,1) \end{bmatrix}. \tag{22}$$

The solution to this system is given by $b = \mathcal{B}^{-1} w$,

$$\begin{bmatrix} b_{11} \\ b_{21} \\ b_{31} \\ b_{12} \\ b_{13} \end{bmatrix} = \begin{bmatrix} 0 & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} & \frac{1}{4} \\ 0 & -\frac{1}{2} & \frac{1}{2} & 0 & 0 \\ -\frac{1}{2} & \frac{1}{4} & \frac{1}{4} & 0 & 0 \\ 0 & 0 & 0 & -\frac{1}{2} & \frac{1}{2} \\ -\frac{1}{2} & 0 & 0 & \frac{1}{4} & \frac{1}{4} \end{bmatrix} \begin{bmatrix} f(0,0) \\ f(-1,0) \\ f(1,0) \\ f(0,-1) \\ f(0,1) \end{bmatrix} = \begin{bmatrix} \frac{f(-1,0)+f(1,0)+f(0,-1)+f(0,1)}{4} \\ \frac{-f(-1,0)+f(1,0)}{2} \\ -\frac{f(0,0)}{2} + \frac{f(-1,0)+f(1,0)}{4} \\ \frac{-f(0,-1)+f(0,1)}{2} \\ -\frac{f(0,0)}{2} + \frac{f(0,-1)+f(0,1)}{4} \end{bmatrix} \tag{23}$$

As expected, the coefficients in (23) coincide with those produced by conventional formula (14).

3.5. Comparison to the conventional Smolyak method

We compare our implementation of the Smolyak method with the conventional implementation described in Section 2. First, we quantify the reduction in cost of the Smolyak interpolant's evaluation that we achieve by avoiding the repetitions and then, we compare the Lagrange interpolation method with explicit formulas for the interpolating coefficients.

3.5.1. Nested-set versus disjoint-set constructions of the Smolyak interpolant

First, consider the conventional construction of the Smolyak polynomial in (5) based on unidimensional nested sets; see Section 2.3.1. By (5), the number of terms which we list to evaluate $\widehat{f}^{d,\mu}$ is $\sum_{\max(d,\mu+1) \le |i| \le d+\mu} \prod_{j=1}^{d} m(i_j)$. Note that the counting coefficient $(-1)^{d+\mu-|i|}\binom{d-1}{d+\mu-|i|}$ is not relevant for computation of the number of terms because it does not add new terms but only counts the number of repetitions to be cancelled out.

Second, consider our alternative construction of the Smolyak polynomial function based on unidimensional disjoint sets; see Section 3.3. The number of basis functions in $\mathcal{P}^{d,\mu}$ is equal to $\sum_{d \le |i| \le d+\mu} \prod_{j=1}^{d} [m(i_j) - m(i_j - 1)]$. To assess the difference in costs between the two constructions, we consider a ratio of the number of terms under the two constructions:

$$R^{d,\mu} \equiv \frac{\sum_{\max(d,\mu+1) \le |i| \le d+\mu} \prod_{j=1}^{d} m(i_j)}{\sum_{d \le |i| \le d+\mu} \prod_{j=1}^{d} [m(i_j) - m(i_j - 1)]} = \frac{\sum_{\max(d,\mu+1) \le |i| \le d+\mu} \prod_{j=1}^{d} m(i_j)}{\sum_{d \le |i| \le d+\mu} \prod_{j=1}^{d} m(i_j) \prod_{j=1}^{d} \left(1 - \frac{m(i_j - 1)}{m(i_j)}\right)}, \tag{24}$$

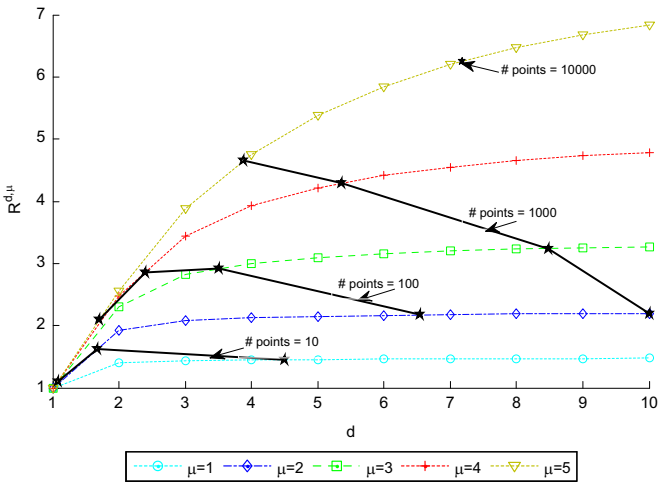

Fig. 2. The ratio of the number of basis functions under two alternative implementations of the Smolyak method.

where $m(0)=0$. In Fig. 2, we represent the ratio $R^{d,\mu}$ for $1\le d\le 10$ and $0\le \mu\le 5$. The "iso-function-count" in the graph shows the number of the Smolyak elements.

The higher is the level of approximation, the larger are the savings due to a more efficient evaluation of the Smolyak interpolant. In particular, under $\mu=1$, the conventional nested-set construction of the Smolyak interpolant is up to 50% more expensive than our construction, while under $\mu=5$, it can be more than 700% more expensive. For high-dimensional applications, the tractable level of approximation is $1\le\mu\le 3$, so that the savings vary from 1.5 to 4. However, we shall emphasize that our construction saves on cost every time when the Smolyak interpolant is evaluated, i.e., in every grid point, integration node or time period.

Some qualitative assessment of $R^{d,\mu}$ can be derived for the case when $\max(d,\mu+1)=d$ (this is the most relevant case for high-dimensional problems in which high-order polynomial approximations are infeasible). Consider the term

$$\prod_{j=1}^{d}\left(1-\frac{m(i_j-1)}{m(i_j)}\right)$$

in the denominator of (24). If $i_j=1$, we have

$$1-\frac{m(0)}{m(1)}=1,$$

and if $i_j\ge 2$, we have

$$1-\frac{m(i_j-1)}{m(i_j)}=\left(1-\frac{2^{i_j-2}+1}{2^{i_j-1}+1}\right)=\frac{2^{i_j-2}}{2^{i_j-1}+1}.$$

Observe that

$$\frac{1}{3}\le\frac{2^{i_j-2}}{2^{i_j-1}+1}\le\frac{1}{2}\quad\text{for } i_j\ge 2,$$

and that the limits are reached under $i_j=2$ and $i_j\to\infty$. This means that for any $i_j\ge 2$, our disjoint-set construction reduces the number of terms by at least a factor of $\frac{1}{3}$ compared to the conventional nested-set construction. For higher approximation levels, the savings are larger, for example, for $i_j\ge 3$, we have

$$\frac{2}{5}\le\frac{2^{i_j-2}}{2^{i_j-1}+1}\le\frac{1}{2},\ \text{etc.}$$

3.5.2. Analytical expression for interpolation coefficients versus numerical Lagrange interpolation

The Lagrange interpolation method is universal: it can be applied to any sets of grid points and basis functions provided that the inverse problem (20) is well-defined (the matrix of basis functions evaluated in grid points has full rank). In turn, the formula of type (14) is a special case of Lagrange interpolation in which the solution to the inverse problem can be derived in a closed form. Using closed-form expressions to calculate the Smolyak interpolation coefficients is relatively inexpensive. Furthermore, Petras (2001) proposes a "divide and conquer" method that allows to reduce even further the cost of evaluating closed-form expressions for the coefficients. His method exploits the fact that expressions of type (14) contain repeated computations; it stores the repeated elements in a tree and uses them as needed.

Our numerical implementation of Lagrange interpolation using (20) has two potential shortcomings compared to the closed-form expression: first, it can be numerically unstable and second, it can be expensive.

To attain numerical stability, we use families of grid points and basis functions that do not lead to ill-conditioned inverse problems. Chebyshev polynomials and their extrema are one example but many other choices are possible.[5] In Fig. 3, we report a condition number of the matrix of basis functions in the associated inverse problem. As a condition number, we use a ratio of the largest to the smallest eigenvalue. In all the cases considered, the condition number is always smaller than 10^5 while an inverse problem is ill-conditioned when the condition number is of order 10^{15} on a 16-digit machine.

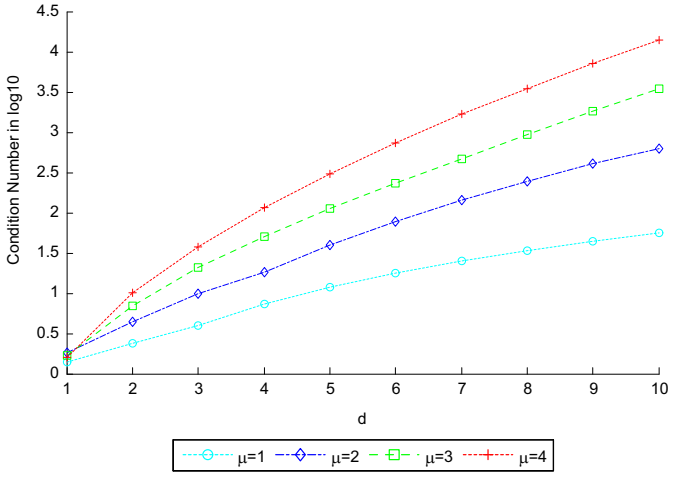

Fig. 3. Condition number of the matrix of basis functions in the inverse problem.

To reduce the computational expense, we use the following precomputation technique in the context of numerical algorithm for solving dynamic economic models. We compute an inverse of the matrix of basis functions in (20) at the initialization stage, before entering the main iterative cycle. In this way, the expensive part of Lagrange interpolation becomes a fixed cost. In the main iterative cycle, computing the interpolation coefficients requires only inexpensive matrix multiplications, which can be easily parallelized for a further reduction in cost if needed. This precomputation technique allows us to reconstruct the polynomial coefficients at a low cost each time when decision and value functions change along iterations.[6]

3.6. *Smolyak formula for interpolation revisited*

It is relatively straightforward to write a computer code that automates the construction of the Smolyak interpolant under our disjoint-set generators described in Section 3.3. We just have to sum up all the basis functions that appear in a table like Table 4 since no basis function is repeated by construction. Nonetheless, we were also able to derive a formula for Smolyak interpolation using disjoint sets, which is parallel to the conventional formula (5) using nested sets. We show such a formula below.

3.6.1. *Smolyak polynomial: efficient construction*

The formula for constructing a Smolyak polynomial function using unidimensional disjoint sets is as follows:

$$\widehat{f}^{d,\mu}(x_1,\ldots,x_d;b)=\sum_{d\le|i|\le d+\mu} q^{|i|}(x_1,\ldots,x_d); \tag{25}$$

for each $|i|$ satisfying $d\le|i|\le d+\mu$, a tensor-product operator $q^{|i|}(x_1,\ldots,x_d)$ is defined as

$$q^{|i|}(x_1,\ldots,x_d)=\sum_{i_1+\cdots+i_d=|i|} q^{i_1,\ldots,i_d}(x_1,\ldots,x_d), \tag{26}$$

[5] See Judd et al. (2011) for a discussion of numerical techniques for dealing with ill-conditioned problems.

[6] See Maliar et al. (2011) and Judd et al. (2011) for other precomputation techniques that reduce the computational expense of numerical solution methods, precomputation of intertemporal choice functions and precomputation of integrals.

and $q^{i_1,\ldots,i_d}(x_1,\ldots,x_d)$ is defined as

$$q^{i_1,\ldots,i_d}(x_1,\ldots,x_d)=\sum_{\ell_1=m(i_1-1)+1}^{m(i_1)}\cdots\sum_{\ell_d=m(i_d-1)+1}^{m(i_d)}b_{\ell_1\ldots\ell_d}\psi_{\ell_1}(x_1)\cdots\psi_{\ell_d}(x_d),\tag{27}$$

where $\psi_{\ell_1}(x_1),\ldots,\psi_{\ell_d}(x_d)$ are unidimensional basis functions, in dimensions $1,\ldots,d$, respectively; $\psi_{\ell_1}(x_1)\cdots\psi_{\ell_d}(x_d)$ is a d-dimensional basis function; $\ell_d=1,\ldots,m(i_d)$; $b_{\ell_1\ldots\ell_d}$ is a coefficients vector. We use the convention that $m(0)=0$ and $m(1)=1$. By construction of (27), there are no repeated terms across different $q^{i_1,\ldots,i_d}$'s.

Formula (25) sums up all the terms in Table 4 (recall that all the sets in Table 4 are disjoint and basis functions are never repeated by construction). The economization from our more efficient alternative Smolyak formula is described in (24) and is shown in Fig. 2. To compute coefficients $b_{\ell_1\ldots\ell_d}$s, we can use either the conventional closed-form expression (14) or a numerical solution using (20) to the Lagrange interpolation problem.

3.6.2. Example of Smolyak polynomial under $d=2$ and $\mu=1$ revisited

We now illustrate the construction of the Smolyak polynomial function (25) by revisiting an example with $d=2$ and $\mu=1$ studied in Section 2.3.2 (in Appendix B, we also show the construction of such a function with $d=2$ and $\mu=2$).

For the case of $\mu=1$, we have $i_1+i_2\le 3$. This restriction is satisfied in three cases: (a) $i_1=i_2=1$; (b) $i_1=1$, $i_2=2$; (c) $i_1=2$, $i_2=1$. Thus, using (27), we obtain

$$\text{(a)}\quad q^{1,1}=\sum_{\ell_1=m(0)+1}^{m(1)}\sum_{\ell_2=m(0)+1}^{m(1)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11},\tag{28}$$

$$\text{(b)}\quad q^{1,2}=\sum_{\ell_1=m(0)+1}^{m(1)}\sum_{\ell_2=m(1)+1}^{m(2)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{12}\psi_2(y)+b_{13}\psi_3(y),\tag{29}$$

$$\text{(c)}\quad q^{2,1}=\sum_{\ell_1=m(1)+1}^{m(2)}\sum_{\ell_2=m(0)+1}^{m(1)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{21}\psi_2(x)+b_{31}\psi_3(x).\tag{30}$$

Collecting elements q^{i_1,i_2} with the same $i_1+i_2\equiv|i|$, we have

$$q^{|2|}\equiv q^{1,1},\tag{31}$$

and

$$q^{|3|}\equiv q^{2,1}+q^{1,2}.\tag{32}$$

The Smolyak polynomial function (27) for the case of $\mu=1$ is given by

$$\mathcal{P}^{2,1}(x,y;b)=\sum_{|i|\le d+\mu}q^{|i|}=q^{1,1}+q^{2,1}+q^{1,2}=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{12}\psi_2(y)+b_{13}\psi_3(y).\tag{33}$$

This formula coincides with (13). Thus, we obtain the same Smolyak polynomial function as the one produced by the conventional Smolyak formula but we avoid forming lists of repeated basis functions.

4. Anisotropic Smolyak method

In economic applications, variables often enter asymmetrically in decision or value functions. For example, in a heterogeneous-agent economy studied in Kollmann et al. (2011), the individual choices depend more on her own variables than on variables of other agents; and the literature, based on Tauchen and Hussy's (1991) discretizations uses many grid points for endogenous state variables and few grid points for exogenous state variables.

However, the Smolyak method, studied in Sections 2 and 3, treats all dimensions equally in the sense that it uses the same number of grid points and basis functions for all variables. To increase the quality of approximation under such a method, we must equally increase the number of grid points and basis functions in all dimensions. This may be too costly to do in large-scale applications. An alternative is to increase the number of grid points and basis functions only in those dimensions that are most important for the overall quality of approximation. In this section, we show a variant of the Smolyak method that allows for a differential treatment of variables, specifically, it enables us to separately choose accuracy levels for each dimension by taking into account a specific structure of decision functions in a given economic model. We refer to such a method as a *Smolyak method with anisotropic grid*.

4.1. Definition of anisotropy

The term *anisotropic* comes from mathematical literature and refers to functions that are "asymmetric" in some respect, for example, have more curvature in some variables than in others, or have a larger range in some variables than in others. Gerstner and Griebel (1998, 2003) propose dimension-adaptive grids for numerical integration of multivariate anisotropic functions; see also Bungartz and Griebel (2004), Garcke (2011) and Jakeman and Roberts (2011). In particular, the last paper

develops a sparse grid technique for both integration and interpolation that combines local and dimension adaptivities. We introduce the Smolyak anisotropic constructions for interpolation in line with this literature.

As a starting point, consider a standard (isotropic) Smolyak grid with a level of approximation μ, as defined in Sections 2 and 3. Observe that for each dimension $j=1,\ldots,d$, index i_j varies from 1 to $\mu+1$. For example, in Table 2, i_1 and i_2 vary from 1 to 3, and μ varies from 0 to 2, respectively (i.e., μ is equal to a maximum index minus 1).

For the anisotropic case, let us denote by μ_j an approximation level in dimension j. If in a dimension j, the maximum index admitted is $i_j^{\max}$, i.e., $i_j=1,\ldots,i_j^{\max}$, then $\mu_j=i_j^{\max}-1$. A Smolyak grid is called *anisotropic* if there is at least one two-dimensional pair j,k such that $\mu_j\neq\mu_k$; otherwise, it is called *isotropic*. We denote an anisotropic Smolyak grid, Smolyak basis functions and Smolyak interpolant as $\mathcal{H}^{d,(\mu_1,\ldots,\mu_d)}$, $\mathcal{P}^{d,(\mu_1,\ldots,\mu_d)}$ and $\widehat{f}^{\,d,(\mu_1,\ldots,\mu_d)}$, respectively.

4.2. Anisotropic Smolyak grid

Our design of an anisotropic variant of the Smolyak method parallels the design of the isotropic Smolyak method described in Sections 3. Namely, we first produce an anisotropic Smolyak grid, we then produce an anisotropic Smolyak polynomial function, and we finally compute polynomial coefficients using Lagrange interpolation. Our anisotropic construction also builds on disjoint-set generators, which allow us to avoid costly repetitions of elements in the Smolyak interpolant.

4.2.1. Tensor products of unidimensional sets of points

The two-dimensional tensor products, constructed from the unidimensional sets up to $i=4$ and $i=2$ in the dimensions x and y, respectively, are summarized in Table 5. By construction, the table contains only non-repeated elements.

4.2.2. Smolyak sets of multidimensional elements under anisotropic construction

The Smolyak rule tells us which tensor products must be selected. For the two-dimensional case, it is as follows: Select elements that belong to the cells in Table 5 for which the following conditions are satisfied:

$$d\leq i_1+i_2\leq d+\mu^{\max}, \tag{34}$$

$$i_1\leq\mu_1+1,\quad i_2\leq\mu_2+1, \tag{35}$$

where $\mu^{\max}\equiv\max\{\mu_1,\mu_2\}$ is a maximum level of approximation across the two dimensions, and the dimensionality is $d=2$. In other words, the sum of indices of a column and a raw, i_1+i_2, must be between d and $d+\mu^{\max}$ subject to additional dimension-specific restrictions $i_1\leq\mu_1+1$, $i_2\leq\mu_2+1$.

Table 5
Tensor products of unidimensional disjoint sets of grid points in the two-dimensional case.

	$A_{i_1}\vert A_{i_2}$	$i_2=1$	$i_2=2$
		0	$-1,1$
$i_1=1$	0	$(0,0)$	$(0,-1),(0,1)$
$i_1=2$	-1 1	$(-1,0)$ $(1,0)$	$(-1,-1),(-1,1)$ $(1,-1),(1,1)$
$i_1=3$	$\frac{-1}{\sqrt{2}}$ $\frac{1}{\sqrt{2}}$	$\left(\frac{-1}{\sqrt{2}},0\right)$ $\left(\frac{1}{\sqrt{2}},0\right)$	$\left(\frac{-1}{\sqrt{2}},-1\right),\left(\frac{-1}{\sqrt{2}},1\right)$ $\left(\frac{1}{\sqrt{2}},-1\right),\left(\frac{1}{\sqrt{2}},1\right)$
$i_1=4$	$\frac{-\sqrt{2+\sqrt{2}}}{2}$ $\frac{-\sqrt{2-\sqrt{2}}}{2}$ $\frac{\sqrt{2-\sqrt{2}}}{2}$ $\frac{\sqrt{2+\sqrt{2}}}{2}$	$\left(\frac{-\sqrt{2+\sqrt{2}}}{2},0\right)$ $\left(\frac{-\sqrt{2-\sqrt{2}}}{2},0\right)$ $\left(\frac{\sqrt{2-\sqrt{2}}}{2},0\right)$ $\left(\frac{\sqrt{2+\sqrt{2}}}{2},0\right)$	$\left(\frac{-\sqrt{2+\sqrt{2}}}{2},-1\right),\left(\frac{-\sqrt{2+\sqrt{2}}}{2},1\right)$ $\left(\frac{-\sqrt{2-\sqrt{2}}}{2},-1\right),\left(\frac{-\sqrt{2-\sqrt{2}}}{2},1\right)$ $\left(\frac{\sqrt{2-\sqrt{2}}}{2},-1\right),\left(\frac{\sqrt{2-\sqrt{2}}}{2},1\right)$ $\left(\frac{\sqrt{2+\sqrt{2}}}{2},-1\right),\left(\frac{\sqrt{2+\sqrt{2}}}{2},1\right)$

Let us construct examples of Smolyak anisotropic grids using the anisotropic version of the Smolyak rule (34).

- If $(\mu_1,\mu_2)=(1,0)$, then $\mu^{\max}=1$ and the anisotropic Smolyak rule implies $2\le i_1+i_2\le 3$, $i_1\le 2$ and $i_2\le 1$. The two cells that satisfy this restriction are (a) $i_1=1$, $i_2=1$; (b) $i_1=2$, $i_2=1$, and the corresponding three Smolyak grid points are
$$\mathcal{H}^{2,(1,0)}=\{(0,0),(-1,0),(1,0)\}. \tag{36}$$

- If $(\mu_1,\mu_2)=(2,1)$, then $\mu^{\max}=2$ and $2\le i_1+i_2\le 4$, $i_1\le 3$ and $i_2\le 2$. There are five cells that satisfy this restriction (a) $i_1=1$, $i_2=1$; (b) $i_1=1$, $i_2=2$; (c) $i_1=2$, $i_2=1$; (d) $i_1=3$, $i_2=1$; (e) $i_1=2$, $i_2=2$; and there are eleven Smolyak grid points
$$\mathcal{H}^{2,(2,1)}=\left\{(-1,1),(0,1),(1,1),(-1,0),(0,0),(1,0),(-1,-1),(0,-1),(1,-1),\left(\frac{-1}{\sqrt{2}},0\right),\left(\frac{1}{\sqrt{2}},0\right)\right\}. \tag{37}$$

- If $(\mu_1,\mu_2)=(3,1)$, then $\mu^{\max}=3$ and $2\le i_1+i_2\le 5$, $i_1\le 4$ and $i_2\le 2$. There are seven cells in the table that satisfy this restriction, and $\mathcal{H}^{3,(3,1)}$ consists of nineteen grid points (see Table 5).

The three Smolyak anisotropic grids constructed in the above two-dimensional example are shown in Fig. 4.

We do not elaborate the construction of the anisotropic Smolyak polynomial function because such a construction trivially repeats the construction of grid points. Furthermore, to find the interpolation coefficients, we use the Lagrange interpolation approach described in Section 3.4, which applies to anisotropic construction without changes. Finally, we can adapt Smolyak interpolation formula (25) to the anisotropic case by imposing restrictions on $i_1,\ldots,i_d$.

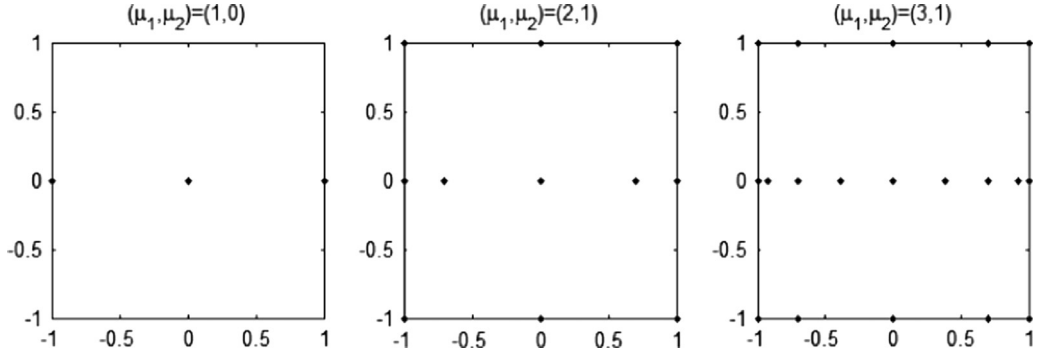

Fig. 4. Examples of Smolyak anisotropic grids.

4.3. Advantages of anisotropic Smolyak method

The anisotropic class of methods involves the following trade-off: from one side, decreasing the number of grid points and basis functions reduces the computational expense but from the other side, it also reduces the quality of approximation. This trade-off suggests that anisotropic methods must be used when the former effect overweighs the latter. Using anisotropic methods in practice would require us either to have certain knowledge of the function that we want to interpolate or to do some experimentation. Namely, we can keep adding or removing grid points and basis functions in different dimensions until the target level of accuracy is attained. Maliar et al. (2014) construct error bounds for a variant of the Smolyak anisotropic method.

Let us compare the number of elements in an isotropic Smolyak grid $\mathcal{H}^{d,\mu}$ with the number of elements in an anisotropic Smolyak grid $\mathcal{H}^{d,(\mu_1,\ldots,\mu_d)}$; we assume that $\mu=\max\{\mu_1,\ldots,\mu_d\}$; and in both cases, we use the disjoint-set construction with no elements repeated:

$$R^{d,\mu}=\frac{\sum_{d\le|i|\le d+\mu}\prod_{j=1}^{d}m(i_j)\left(1-\frac{m(i_j-1)}{m(i_j)}\right)}{\sum_{d\le|i|\le d+\mu,\{i_j\le\mu_j+1\}_{j=1}^{d}}\prod_{j=1}^{d}m(i_j)\left(1-\frac{m(i_j-1)}{m(i_j)}\right)}.$$

The above ratio depends on the specific assumption about the approximation level in each dimension $(\mu_1,\ldots,\mu_d)$. Let us assume that the true function is completely flat in all dimensions except in dimension one. To accurately approximate such a

function, we can use an anisotropic Smolyak method in which $\mu_1 = \mu$ in dimension 1 and $\mu_j = 1$ for all other dimensions, $j = 2, \ldots, d$. In Fig. 5, we show the corresponding ratio of the number of points under isotropic and anisotropic versions of the Smolyak method. Again, the "iso-function-count" curves show the number of the Smolyak elements.

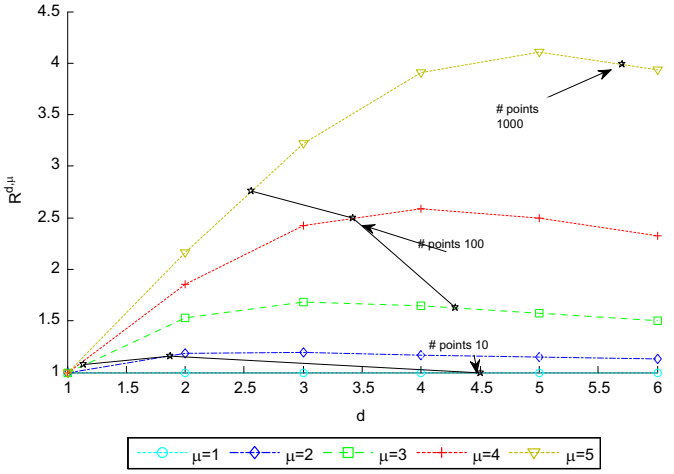

Fig. 5. The ratio of the number of basis functions under isotropic and anisotropic versions of the Smolyak method.

Gains from anisotropic constructions may vary depending on the anisotropy of specific decision or value functions in economic models. In our example, the savings are sizable when d and μ are large. For example, when $d=6$ and $\mu = 5$ for dimension 1, the number of grid points and basis functions is reduced by about 4 times compared to the isotropic grid.

5. Smolyak method with adaptive domain

The Smolyak construction tells us how to represent and interpolate functions defined on a normalized d-dimensional hypercube. However, the solution domain of a typical dynamic economic model does not have the shape of a hypercube but can be a set of any shape in a d-dimensional space. We now describe how to effectively adapt a multidimensional Smolyak hypercube to an unstructured solution domain of a given economic model.

5.1. Adaptive parallelotope

The ergodic set (i.e., the support of the ergodic distribution) of a dynamic economic model can have any shape in a d-dimensional space. It may be even an unbounded set such as $\mathbb{R}^d_+$. We must first construct a d-dimensional parallelotope to enclose the relevant area of the state space of the studied model, typically, a high-probability area of the ergodic set. We must then match the parallelotope to a normalized hypercube $[-1, 1]^d$ used by the Smolyak method.

As an example, in Fig. 6, we plot a simulation of 10,000 observations for capital and productivity level in a representative-agent neoclassical stochastic growth model with a closed-form solution (see Section 6.1 for a detailed description of this model).

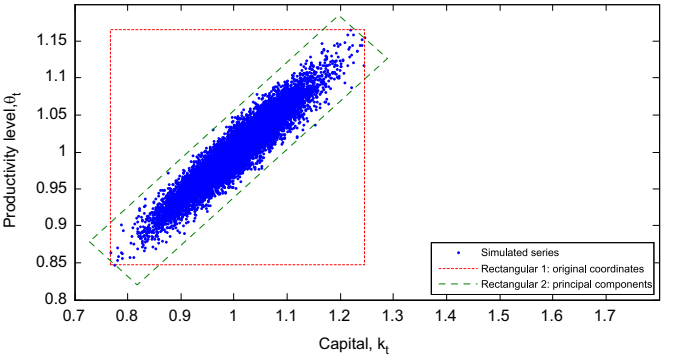

Fig. 6. Two rectangular domains enclosing a set of simulated points.

We show two possible rectangles in the figure that enclose the given set of simulated point: one is a conventional rectangle in the original coordinates, and the other is a rectangle obtained after a change of variables (both rectangles are minimized subject to including all the simulated points).

Our example shows that the way in which the rectangle is adapted to the state space of an economic model can matter a lot for the size of the resulting rectangle. How much the rectangle can be reduced by changing the system of coordinates will depend on the shape of the high-probability set. In particular, all rectangles will have the same size when simulated data are of the shape of a perfect sphere, however, the reduction in size can be fairly large if such high-probability set is inclined (as is shown in the figure). The reduction in size can be especially large in problems with high dimensionality.

5.2. Smolyak grid on principal components

There are many ways to match a parallelotope into a normalized hypercube in the context of the Smolyak method. We propose an approach that relies on a principle component (PC) transformation and that is convenient to use in economic applications. Namely, we first use simulation to determine the high-probability area of the model's state space (for a sufficiently accurate initial guess), and we then build a parallelotope surrounding the cloud of simulated points. The PC approach was previously used by Judd et al. (2010) and Maliar and Maliar (2014b) in the context of simulation-based methods in order to define a distance between simulated points. In the present paper, the PC approach is used to re-orientate the parallelotope.

Let $X \equiv (x^1, \ldots, x^{\mathcal{L}}) \in \mathbb{R}^{T\times\mathcal{L}}$ be a set of simulated data, i.e., we have T observations on $\mathcal{L}$ variables. Let the variables $(x^1, \ldots, x^{\mathcal{L}})$ be normalized to zero mean and unit variance. Consider the singular value decomposition of X, defined as $X = USV^\top$, where $U \in \mathbb{R}^{T\times\mathcal{L}}$ and $V \in \mathbb{R}^{\mathcal{L}\times\mathcal{L}}$ are orthogonal matrices, and $S \in \mathbb{R}^{\mathcal{L}\times\mathcal{L}}$ is a diagonal matrix with diagonal entries $s_1 \geq s_2 \geq \cdots \geq s_{\mathcal{L}} \geq 0$, called *singular values* of X. Perform a linear transformation of X using the matrix of singular vectors V as follows: $Z \equiv XV$, where $Z = (z^1, \ldots, z^{\mathcal{L}}) \in \mathbb{R}^{T\times\mathcal{L}}$. The variables $z^1, \ldots, z^{\mathcal{L}}$ are called *principal components* of X, and are orthogonal (uncorrelated), $(z^{\ell'})^\top z^{\ell} = 0$ for any $\ell' \neq \ell$ and $(z^{\ell})^\top z^{\ell} = s_\ell^2$. The sample variance of z^{ℓ} is s_ℓ^2/T, and, thus, z^1 and $z^{\mathcal{L}}$ have the largest and smallest sample variances, respectively.

In Fig. 7, we illustrate the four steps of construction of the adaptive parallelotope using PCs.

In the upper-left panel, we show a set of simulated points; in the upper-right panel, we translate the origin to the center of the cloud, rotate the system of coordinates, renormalize the principal components to have a unit variance in both dimensions and surround it with a hypercube $[-1, 1]^2$. In the lower-left panel, we show thirteen Smolyak points with the approximation level $\mu = 2$ for the PCs of the data. Finally, in the lower-right panel, we plot the corresponding thirteen Smolyak points for the original data after computing an inverse PC transformation. In Appendix C, we provide additional details on how to construct the rectangular domain for this particular example.

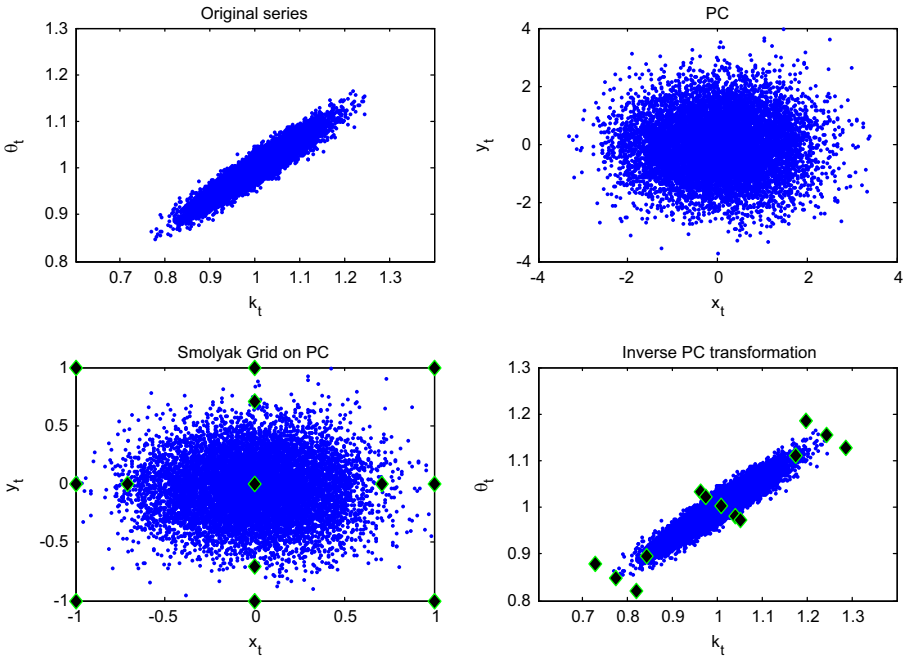

Fig. 7. Smolyak grid on PCs.

5.3. Advantages and possible pitfalls of adaptive parallelotopes

The size of a parallelotope on which a function is approximated is translated into either higher quality or lower costs of the resulting approximation. For a fixed cost of approximation (i.e., for a given level of approximation), fitting a polynomial on a relevant domain gives us a better fit inside the relevant domain than fitting a polynomial on a large but partially irrelevant domain; this is because in the latter case we face a trade-off between the fit inside and outside the relevant domain. In turn, if we solve the model on a larger domain, we can attain the given quality of approximation in the relevant domain by computing a more expensive approximation (characterized by a higher approximation level).

However, the technique of an adaptive parallelotope has also a potential pitfall. Specifically, in addition to the size of the domain on which a function is approximated, the accuracy of the Smolyak method depends critically on the relative importance of cross terms. In particular, a Smolyak method with the level of approximation $\mu=1$ has no cross terms and cannot accurately approximate decision or value functions with non-negligible cross-variable interactions. In certain economic models, such interactions can be either absent or small in the original coordinates but may become important when we rotate the system of coordinates and as a result, the quality of approximation can decrease rather than increase. It is also possible to have the opposite effect, namely, the importance of cross terms can decrease after we rotate the system of coordinates. Some experimentation may help us to decide whether to use the adaptive parallelotope technique or not. Finally, we should point out that it is possible to combine an asymmetric treatment of variables with an adaptive parallelotope. This could be a potentially useful extension for some applications but we do not pursue it in the present paper.

6. Smolyak method for solving dynamic economic models

The Smolyak method for interpolation is just one specific ingredient of a method for solving dynamic economic models. We need to complement it with other ingredients, such as a procedure for approximation of integrals, a procedure that solves for fixed point coefficients, a procedure that updates the functions along iterations, a procedure that maps the state space of a given economic model into the Smolyak hypercube, etc. In this section, we incorporate the Smolyak method for interpolation into a projection methods for solving dynamic economic models. We assess the performance of the studied Smolyak-based solution method in the context of one- and multi-agent optimal growth models.

6.1. The representative agent model

Our first example is the standard representative agent neoclassical stochastic growth model:

$$\max_{\{c_t,k_{t+1}\}_{t=0}^{\infty}} E_0 \sum_{t=1}^{\infty} \beta^t u(c_t) \tag{38}$$

$$\text{s.t.} c_t + k_{t+1} = (1-\delta)k_t + \theta_t f(k_t), \tag{39}$$

$$\ln \theta_t = \rho \ln \theta_{t-1} + \sigma\varepsilon_t, \quad \varepsilon_t \sim \mathcal{N}(0,\sigma^2), \tag{40}$$

where $c_t, k_{t+1} \geq 0$, and k_0 and θ_0 are given. Here, c_t and k_t are consumption and capital, respectively; $\beta \in (0,1)$ is the discount factor; $u(c_t)$ is the utility function, which is assumed to be increasing and concave; $\delta \in (0,1]$ is the depreciation rate of capital; $f(k_t,\theta_t)$ is the production function with $\alpha \in (0,1)$ being the capital share in production; and E_t is the operator of expectation conditional on state (k_t,θ_t). The productivity level θ_t in (40) follows a first-order autoregressive process with $\rho \in (-1,1)$ and $\sigma > 0$.

6.2. Time iteration versus fixed-point iteration

Our implementation of the Smolyak method also differs from the one in Krueger and Kubler (2004) and Malin et al. (2011) in the technique that we use to iterate on decision functions. Specifically, they use time iteration that solves a system of non-linear equations using a numerical solver, whereas we use derivative-free fixed-point iteration that does so using only straightforward calculations. As an illustration, suppose we need to solve a non-linear equation $f(x)=x$; then time iteration finds $\min_x |f(x)-x|$ using a solver, while fixed-point iteration constructs a sequence like $x^{(i+1)}=f(x^{(i)})$, $i=0,1,\ldots$, starting from some initial guess $x^{(0)}$ with the hope that this sequence will converge to a true solution. See Wright and Williams (1984), Miranda and Helmberger (1988), Marcet (1988) for early applications of fixed-point iteration to economic problems. Den Haan (1990) proposed a way to implement fixed-point iteration in models with multiple Euler equations; see also Marcet and Lorenzoni (1999) for related examples. Gaspar and Judd (1997) pointed out that fixed-point iteration is a cheap alternative to time iteration in high-dimensional applications. Finally, Judd et al. (2010, 2011) and Maliar and Maliar (2014b) show a variant of fixed-point iteration, which performs particularly well in the context of the studied models; we

adopt their variant in the present paper. Below, we illustrate the difference between time-iteration and fixed-point iteration methods using the model (38)–(40) as an example.

6.2.1. Time iteration

Time iteration solves a system of three equilibrium conditions with respect to $\widehat{k}'$ in each point of the Smolyak grid. It parameterizes the capital decision function by Smolyak polynomial $\widehat{k}' = K(k,\theta;b)$, where b is the coefficients vector. It iterates on b by solving for current capital $\widehat{k}'$ given the Smolyak interpolant for future capital $K(\widehat{k}',\theta';b)$ (it mimics time iteration in dynamic programming where we solve for a current value function given a future value function):

$$u'(c) = \beta E[u'(c'_j)(1-\delta+\theta'_j f'(\widehat{k}'))], \tag{41}$$

$$c = (1-\delta)k+\theta f(k)-\widehat{k}', \tag{42}$$

$$c'_j = (1-\delta)\widehat{k}'+\theta'_j f(\widehat{k}')-K(\widehat{k}',\theta'_j;b). \tag{43}$$

The system (41)–(43) must be solved with respect to $\widehat{k}'$ using a numerical solver. Observe that Smolyak interpolation $K(\widehat{k}',\theta'_j;b)$ must be performed for each subiteration on $\widehat{k}'$ using a numerical solver, which is expensive. Time iteration has a high cost even in a simple unidimensional problem.

Time iteration becomes far more expensive in more complex settings. For example, in the multi-agent version of the model, one needs to solve a system of N Euler equations with respect to N unknown capital stocks. A high cost of time iteration procedure accounts for a rapid increase in the cost of the Smolyak method of Malin et al. (2011) with the dimensionality of the problem.

6.2.2. Fixed-point iteration

We also parameterize the capital function using Smolyak polynomial $\widehat{k}' = K(k,\theta;b)$. Before performing any computation, we rewrite the Euler equation of the problem (38)–(40) in a way, which is convenient for implementing a fixed-point iteration:

$$k' = \beta E\left[\frac{u'(c')}{u'(c)}(1-\delta+\theta' f'(k'))k'\right]. \tag{44}$$

In the true solution, k' on both sides of (44) takes the same values and thus, cancels out. In the fixed-point iterative process, k' on the two sides of (44) takes different values. To proceed, we substitute $k' = \widehat{K}(\cdot;b)$ in the right side of (44), and we get a different value in the left side of (44); we perform iterations until the two sides coincide.[7]

Using parameterization (44), we represent the system of (equations (41)–(43)) as follows:

$$k' = K(k,\theta;b) \quad \text{and} \quad k''_j = K(k',\theta';b), \tag{45}$$

$$c = (1-\delta)k+\theta f(k)-k', \tag{46}$$

$$c' = (1-\delta)k'+\theta' f(k')-k'', \tag{47}$$

$$\widehat{k}' = \beta E\left[\frac{u'(c')}{u'(c)}(1-\delta+\theta' f'(k'))k'\right]. \tag{48}$$

In each iteration, given b, we compute k', k'', c, c', substitute them into (48), get $\widehat{k}'$ and continue iterating until convergence is achieved.

In Appendix D, this approach is extended to a multi-agent version of the model to perform iterations on N Euler equations. Even in the multidimensional case, our fixed-point iterative procedure requires only trivial calculations and avoids the need of a numerical solver, unlike the time-iteration method.

Some theoretical arguments suggest that time iteration may possess better convergence properties than fixed-point iteration. In particular, for some simple models, it is possible to show that time iteration has a contraction mapping property locally, which is similar to the one observed for value function iteration; see Judd (1998, p.553) for details. However, the local contraction mapping property is not preserved in more complicated models like the multi-agent model studied later in the paper. It is therefore unknown which iterative scheme has better convergence properties in general. Our simple fixed-point iteration method was reliable and stable in all experiments if appropriate damping was used.

[7] This kind of parameterization was used by Den Haan (1990) as a technique to implement the parameterized expectations algorithm in a model with several Euler equations.

6.3. Algorithm

Below, we show a Smolyak-based projection algorithm for solving the model (38)–(40).

Algorithm 1. Smolyak-based projection method.

Initialization.

a. Choose the approximation level, μ.
b. Construct the Smolyak grid $\mathcal{H}^{2,\mu}=\{(x_n,y_n)\}_{n=1,\ldots,M}$ on $[-1,1]^2$.
c. Compute the Smolyak basis functions $\mathcal{P}^{2,\mu}$ in each grid point n. The resulting $M\times M$ matrix is $\mathcal{B}$.
d. Fix $\Phi:(k,\theta)\to(x,y)$, where $(k,\theta)\in\mathbb{R}^2_+$ and $(x,y)\in[-1,1]^2$. Use Φ^{-1} to compute (k_n,θ_n) that corresponds to (x_n,y_n) in $\mathcal{H}^{2,\mu}$.
e. Choose integration nodes, ϵ_j, and weights, ω_j, $j=1,\ldots,J$.
f. Construct future productivities, $\theta'_{n,j}=\theta_n^{\rho}\exp(\epsilon_j)$ for all j;
g. Choose an initial guess $b^{(1)}$.

Step 1. *Computation of a solution for K.*

a. At iteration i, for $n=1,\ldots,M$, compute
- $k'_n=\mathcal{B}_n b^{(i)}$, where $\mathcal{B}_n$ is the nth raw of $\mathcal{B}$.
- $(x'_n,y'_{n,j})$ that corresponds to $(k'_n,\theta'_{n,j})$ using Φ.
- Compute the Smolyak basis functions in each point $(x'_n,y'_{n,j})$
- The resulting $M\times M\times J$ matrix is $\mathcal{B}'$.
- $k''_{n,j}=\mathcal{B}'_{n,j}b^{(i)}$, where $\mathcal{B}'_{n,j}$ is the nth raw of $\mathcal{B}'$ in state j.
- $c_n=(1-\delta)k_n+\theta_n f(k_n)-k'_n$;
- $c'_{n,j}=(1-\delta)k'_n+\theta_n^{\rho}\exp(\epsilon_j)f(k'_n)-k''_{n,j}$ for all j;
- $\widehat{k}'_n\equiv\beta\sum_{j=1}^{J}\omega_j\cdot\left[\frac{u'(c'_{n,j})}{u'(c_n)}[1-\delta+\theta_n^{\rho}\exp(\epsilon_j)f'(k'_n)]k'_n\right]$

b. Find b that solves the system in Step 1a.
- Compute $\widehat{b}$ that solves $\widehat{k}'=\mathcal{B}\widehat{b}$, i.e., $\widehat{b}=\mathcal{B}^{-1}\widehat{k}'_n$.
- Use damping to compute $b^{(i+1)}=(1-\xi)b^{(i)}+\xi\widehat{b}$, where $\xi\in(0,1]$.
- Check for convergence: end Step 1 if $\frac{1}{M\xi}\sum_{n=1}^{M}\left|\frac{(k'_n)^{(i+1)}-(k'_n)^{(i)}}{(k'_n)^{(i)}}\right|<\varpi$.

Iterate on Step 1 until convergence.

6.4. Relation to other solution methods in the literature

We now describe the relation between the Smolyak solution method and other numerical methods for solving dynamic economic models in the literature; see Maliar and Maliar (2014a) for a survey of numerical methods for solving large-scale dynamic economic models. First, the baseline version of the Smolyak method is similar to conventional projection methods in that it relies on a fixed grid, orthogonal polynomials and deterministic integration methods; see Judd (1992), Gaspar and Judd (1997), Christiano and Fisher (2000), and Aruoba et al. (2006), among others. The difference is that conventional projection methods build on tensor-product rules and their cost grows rapidly with the dimensionality of the problem, whereas the Smolyak method uses non-product rules and its cost grows far more slowly. Second, the anisotropic variant of the Smolyak method is similar to solution methods that use different number of grid points for different variables (e.g., few grid points for shocks and many grid points for capital as in Aiyagari, 1994, Huggett, 1993, Rios-Rull, 1997, and Krusell and Smith, 1998) and other numerical methods that rely on discretization of shocks in line with Tauchen and Hussey (1991). Third, the variant of the Smolyak method with an adaptive domain is related to simulation-based and learning methods; see, e.g., Marcet (1988), Marcet and Sargent (1989), Smith (1991, 1993), Rust (1997), Maliar and Maliar (2005) and Powell (2011). The advantage of simulation-based methods is that they focus on the relevant area of the state space; the shortcoming is that their accuracy is limited by a low rate of convergence of Monte Carlo integration. The Smolyak method with an adaptive domain does not rely on Monte Carlo integration: it uses simulations only for constructing the solution domain, and it uses projection techniques to accurately solve the model on this domain; in this respect, it is similar to combinations of projection and simulation methods studied in Judd et al. (2010) and Maliar and Maliar (2005, 2014b). Fourth, in contrast to perturbation methods, the Smolyak method is a global solution method: its accuracy does not decline rapidly away from the steady state, and it can be used to accurately solve models with strong non-linearities. Fifth, in our examples, we focus on equilibrium problems, however, the techniques described in the paper can be also used in the context of dynamic programming problems. Winschel and Krätzig (2010) show how to use the conventional Smolyak method in the context of canonical value function iteration, however, the Smolyak method can be used in the context of an endogenous grid method of Carroll (2005) and an envelope condition method of Maliar and Maliar (2013) both of which can significantly reduce the cost of conventional value function iteration. Finally, Brumm and Scheidegger (2013) parallelize a sparse grid

method using piecewise local basis functions, and Valero et al. (2013) parallelize a Smolyak method using a global polynomial function. In particular, the latter paper parallelizes the Smolyak method on a desktop computer using multiple CPUs and GPUs and on Blacklight supercomputer; see Maliar (2013) for additional experiments assessing gains from parallelization on Blacklight.

Finally, the anisotropic-grid and adaptive-domain constructions, which we propose in the context of the Smolyak method, will also work for any projection solution method that operates on a hypercube domain including those based on tensor-product rules, e.g., Judd (1992). Also, these construction can be effectively used in the context of low-dimensional problems. However, there are many ways to increase accuracy in problems with low dimensionality, in particular, one can increase the degree of polynomial approximations. In high-dimensional applications, increasing the polynomial degree might be too costly. Anisotropic grid or adaptive domain or their combination may be the only feasible alternative. This is why we advocate these techniques in the context of the Smolyak method applied to problems with high dimensionality.

6.5. Implementation details

Our algorithm builds on techniques that are used in the related literature. To approximate expectation functions, we use a ten-node Gauss–Hermite quadrature rule. We find that for this particular class of examples, the choice of the number of integration nodes plays a very minor effect on the properties of the solution, for example, the results will be the same up to 6 digits of precision if instead of ten integration nodes we use just two. We could have used Monte Carlo integration but this would reduce the accuracy dramatically; see Judd et al. (2011) for a discussion.

We consider two mappings $\Phi: X \to [-1,1]^2$ that transform each possible value of state variables $(k,\theta) \in X \subseteq \mathbb{R}^2$ into a hypercube (which is a square in the two-dimensional case) $(x,y) \in [-1,1]^2$. One is a conventional rectangular domain for capital and productivity, and the other is a rectangular domain constructed on principal components of simulated series. The rectangles are chosen to enclose the cloud of simulated data as shown in Fig. 6. (That is, we solve the model two times: we first compute a solution starting from some initial guess about the ergodic range, we then simulate time series, and we finally recompute solutions more accurately using the rectangular domains that enclose the cloud of simulated data). In Appendix C, we provide further details on the construction of the two mappings.

We use extrema of Chebyshev polynomials as unidimensional grid points, and we use a Chebyshev polynomial family as unidimensional basis functions; in this respect, we are similar to Krueger and Kubler (2004) and Malin et al. (2011). We use the damping parameter $\xi = 0.05$, and we use the convergence criterion $\varpi = 10^{-7}$.

Finally, as a measure of accuracy, we report the mean and maximum of unit-free Euler equation errors on a stochastic simulation of 10,000 observations. Our computer code is written in MATLAB 2012a, and we use a desktop computer with Intel(R) Core(TM) i7-2600 CPU (3.40 GHz) with RAM 12 GB. At the initial stage of this project, we benefited from consulting the Fortran code of Malin et al. (2011).

6.6. Results for the representative agent model

We parameterize the model (38)–(40) by $u(c) = (c^{1-\gamma}-1)/(1-\gamma)$, $f(k) = k^{\alpha}$ and $\ln \theta' = \rho \ln \theta + \sigma\varepsilon$, where the parameters are fixed at $\rho = 0.95$, $\beta = 0.99$ and $\alpha = 1/3$. We set the benchmark values of $\delta = 1$, $\gamma = 1$ and $\sigma = 0.01$, and we consider variations in γ, σ and δ one-by-one, holding the remaining parameters at the benchmark values. Specifically, we consider

$$\delta = \left\{ \begin{matrix} 0, 0.01, 0.02, 0.025, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, \\ 0.09, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1 \end{matrix} \right\},$$

$$\gamma = \{1, 5, 10, 15, 20\},$$

$$\sigma = \{0.001, 0.005, 0.01, 0.02, 0.03, 0.04, 0.05\}.$$

We use these parameterizations to test the performance of different versions of the Smolyak method introduced in the paper.

6.6.1. Conventional isotropic Smolyak grids under different approximation levels

We first solve the model (38)–(40) using a baseline version of the Smolyak method under four approximation levels $\mu = 1, 2, 3, 4$. Our baseline version is isotropic, i.e., has the same number of grid points for capital and productivity, and it operates on a rectangular domain in the original system of coordinates. The algorithm was able to converge in a wide range of the parameters and to produce highly accurate solutions.

In Fig. 8, we report the (unit-free) maximum residuals in the Euler equation (44) (expressed in log 10 units). The residuals vary from about 1% under $\mu = 1$–10^{-8}% under $\mu = 4$. Therefore, the quality of approximation consistently increases with the value of μ.

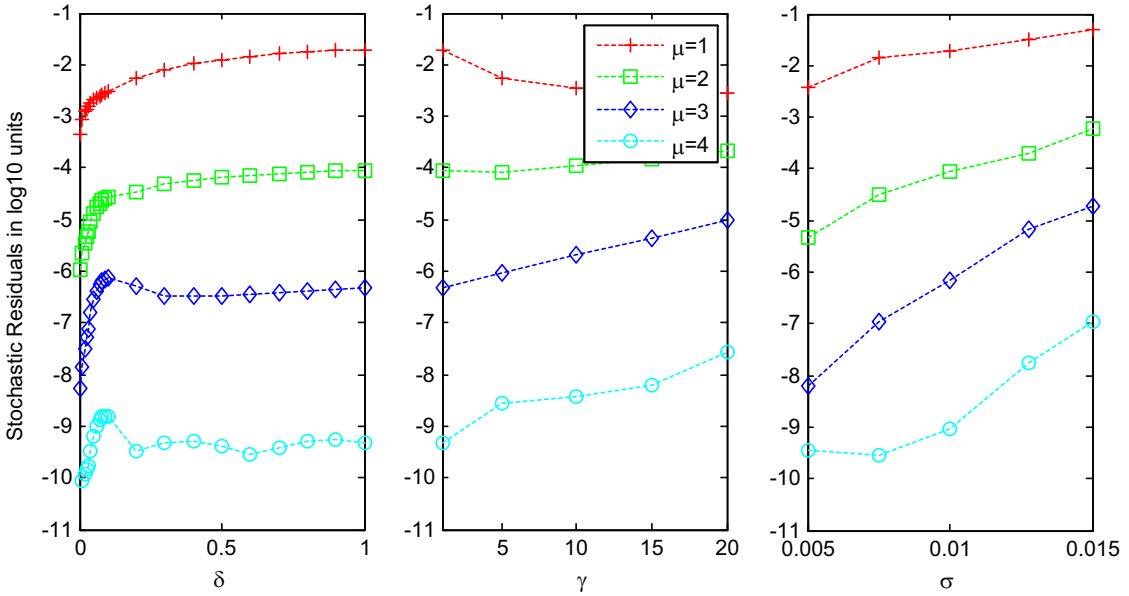

Fig. 8. Accuracy of the Smolyak method under different approximation levels.

6.6.2. Anisotropic Smolyak grids

We next consider anisotropic variants of the Smolyak method that use different numbers of grid points for different variables. We consider two possibilities $(\mu_1, \mu_2) = (3, 1)$ and $(\mu_1, \mu_2) = (1, 3)$. With these constructions, we have nine elements in the first dimension and three elements in the second dimension, which results in nineteen elements in total (i.e., nineteen grid points and nineteen basis functions); these are the elements distinguished in Section 4.2.2.

In Fig. 9, we compare the maximum residuals in the Euler equation with anisotropic grids and isotropic grids. The medium line (the one with triangles) is our benchmark isotropic case $\mu = 2$ that contains 13 polynomial terms.

We observe that if we use more grid points in dimension of capital than in dimension of productivity, the anisotropic Smolyak method produces more accurate solutions than the benchmark isotropic Smolyak method, but if we have more grid points in productivity than in capital, the opposite is true. The difference in accuracy between two anisotropic solutions can be as large as two orders of magnitude. These results illustrate the potential usefulness of anisotropic grids in economic applications.

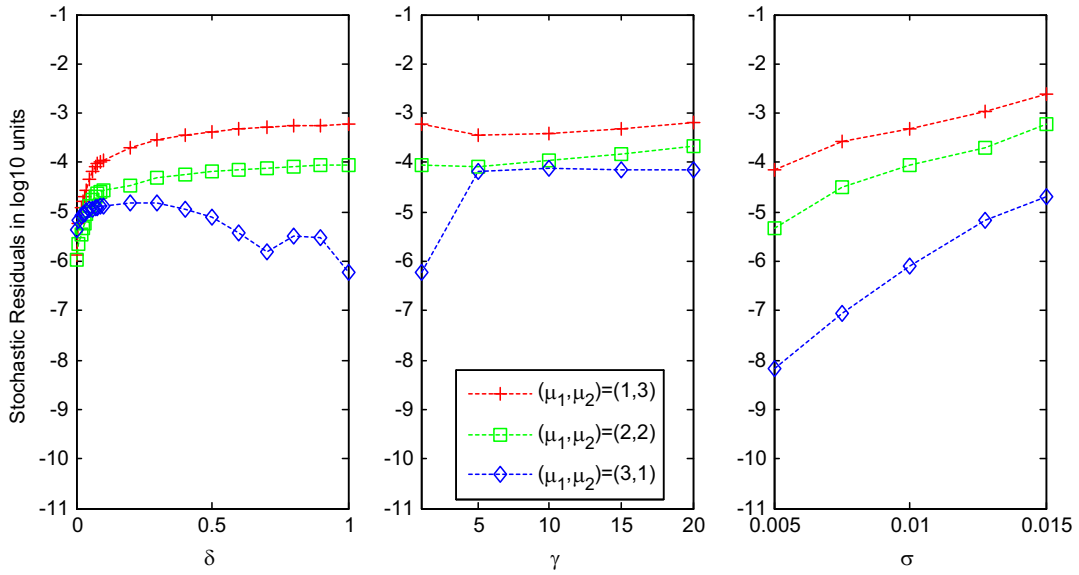

Fig. 9. Accuracy of the Smolyak method with anisotropic grid.

6.6.3. Adaptive domain

We next assess how the performance of the Smolyak method depends on the choice of the solution domain. We compare the accuracy of solutions of the Smolyak method that operates on the conventional hypercube with that of the Smolyak technique that operates on the adaptive rectangular domain. We compare the maximum residuals in the Euler equation under the conventional and adaptive domains in Fig. 10.

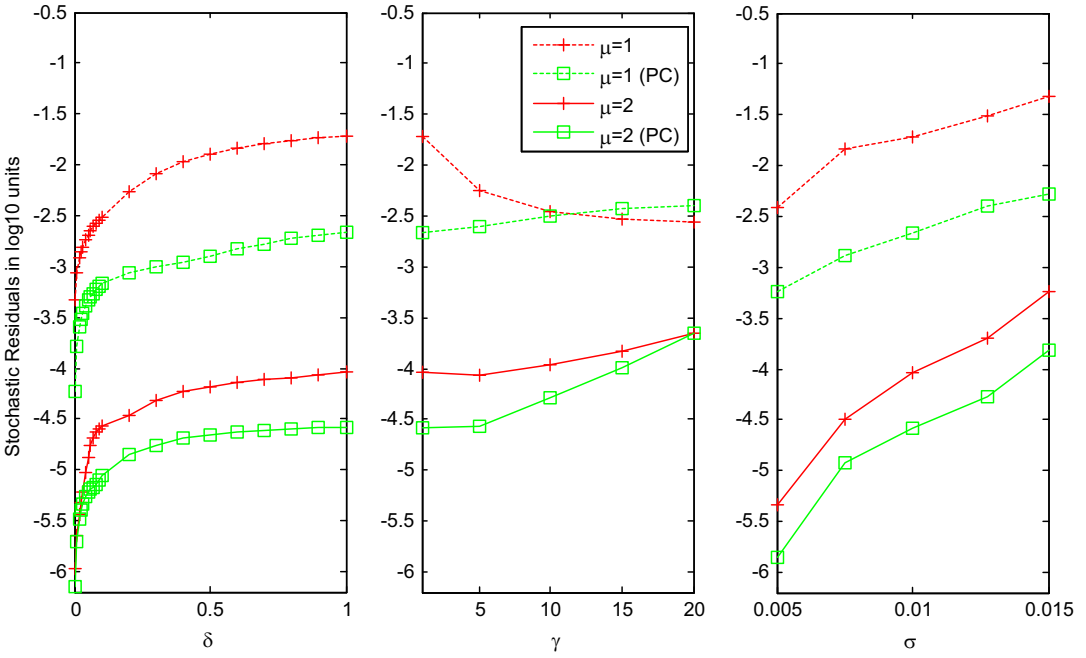

Fig. 10. Accuracy of the Smolyak method with adaptive domain.

The maximum residuals in the Euler equation are about a half order of magnitude (i.e., about 5 times) lower under the adaptive Smolyak domain than under the conventional Smolyak domain for the range of values for γ, σ and δ considered. We observe a visible improvement in the quality of approximation due to an adaptive domain under both $\mu=1$ and $\mu=2$. In Section 3.5, we argued that there could be a potentially negative effect of cross term on accuracy after we rotate the system of coordinates. Our numerical results suggest that this effect does not play a significant role in our example except when γ becomes very large.

6.6.4. Computational expense

We finally assess savings due to our more efficient disjoint-set implementation of the Smolyak method (recall that we avoid forming lists with repeated basis functions when constructing the Smolyak interpolant). In the comparison analysis, we apply an isotropic variant of the Smolyak method to the model parameterized by $\delta=1$, $\gamma=1$ and $\sigma=0.01$. To implement the conventional Smolyak method with a nested-set construction, we use software by Gordon (2011), which we find reliable and easy to use; see https://sites.google.com/site/greygordon/code.

In our experiments, we also study how the savings depend on a specific iterative scheme, namely, we compare time iteration in line with Malin et al. (2011) to our baseline fixed point iteration, see Sections 6.2.1 and 6.2.2, respectively. To solve for capital choices under time iteration, we use a solver "csolve.m" by Chris Sims. In Fig. 11, we show the running time for four considered cases referred to as "FPI with disjoint sets", "FPI with nested sets", "TI with disjoint sets" and "TI with nested sets", where FPI and TI stand for fixed-point iteration and time iteration, respectively.

Two tendencies can be seen from the figure. First, the conventional nested-set implementation of the Smolyak method is always more expensive than our disjoint-set implementation. This is true under both time iteration and fixed-point iteration schemes. In the case of fixed point iteration, the savings increase with the level of approximation μ and can reach two orders of magnitude even in our simple two-dimensional example. For the time iteration scheme, the savings are more modest but still visible. Second, time iteration performs faster than fixed-point iteration in our example. This is because the cost of a numerical solver is relatively low for small problems like ours, while the convergence rate of time iteration is higher than that of fixed-point iteration. This finding is in line with the analysis of Gaspar and Judd (1997) and Malin et al. (2011) in which time iteration methods are very fast in small-scale models but their cost increases dramatically with the size of the problem. Fixed-point iteration will be a competitive alternative for larger problems as we will see in the next section, Section 6.7.

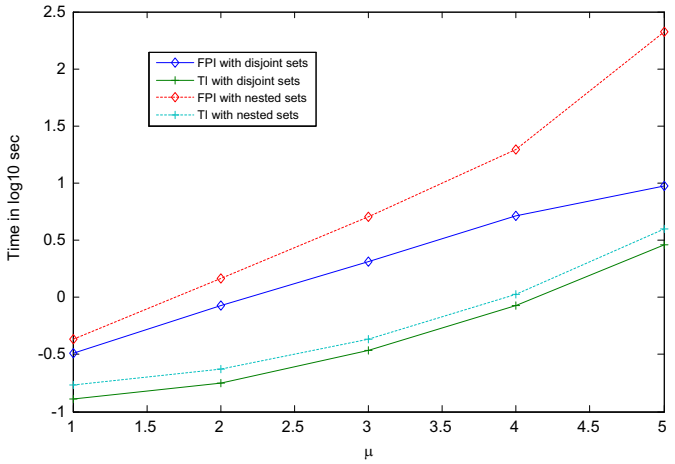

Fig. 11. Running time of the Smolyak method using fixed-point and time iteration with disjoint and nested sets.

6.7. Results for the multicountry model

We now explore the performance of the Smolyak-based projection method in the context of problems with high dimensionality. To this purpose, we extend the one-agent model (38)–(40) to include multiple agents. This is a simple way to expand the size of the problem and to have a control over its dimensionality.

There are N agents, interpreted as countries, that differ in initial capital endowment and productivity level. The countries' productivity levels are affected by both country-specific and worldwide shocks. We study the social planner's problem. If agents are identical in preferences, the planner will allocate identical consumption to all agents. However, we do not make use of the symmetric structure of the economy, and we approximate the planner's solution in the form of N capital policy functions, each of which depends on $2N$ state variables (N capital stocks and N productivity levels). We solve for N policy functions $(k')^h = \mathcal{B}b^h$, where $\mathcal{B}$ is a matrix of Smolyak basis functions evaluated in the Smolyak grid points, and b^h is a vector of the polynomial coefficients for $h = 1, \ldots N$. For each country, we use essentially the same computational procedure as the one used for the representative-agent model. The Gauss–Hermite quadrature method builds on product rules and is not tractable in problems with high dimensionality. We replace it with monomial rules combined with Cholesky decomposition; see Judd et al. (2011) for a detailed description of these techniques. For a description of the multicountry model and details of the computational procedure, see Appendix D.

In Fig. 12, we compare four different solutions to the multicountry model, namely, the conventional solutions with $\mu = 2$ and $\mu = 3$, the anisotropic solution with $\mu_1^h = 3$ for the capital stocks and $\mu_2^h = 2$ for the productivity levels of all countries, $h = 1, \ldots N$ (in the figure, we denote this case as $(\mu_1, \mu_2) = (3, 2)$ which means $(\mu_1^1, \ldots, \mu_1^N, \mu_2^1, \ldots, \mu_2^N) = (3, \ldots, 3, 2, \ldots, 2)$). Finally, we report the solution with $\mu = 2$ for the Smolyak method with an adaptive domain.

We observe the following results from the figure. First, the difference between the isotropic solution with $\mu = 3$ and that with $\mu = 2$ is about two orders of magnitude. Second, by using an anisotropic grid, we can make half of the way between $\mu = 2$ and $\mu = 3$ in terms of the average residuals, and we obtain essentially identical maximum residuals. Third, the effect of an adaptive domain is also quite sizable, namely, we can reduce the residuals up to one order of magnitude. Finally,

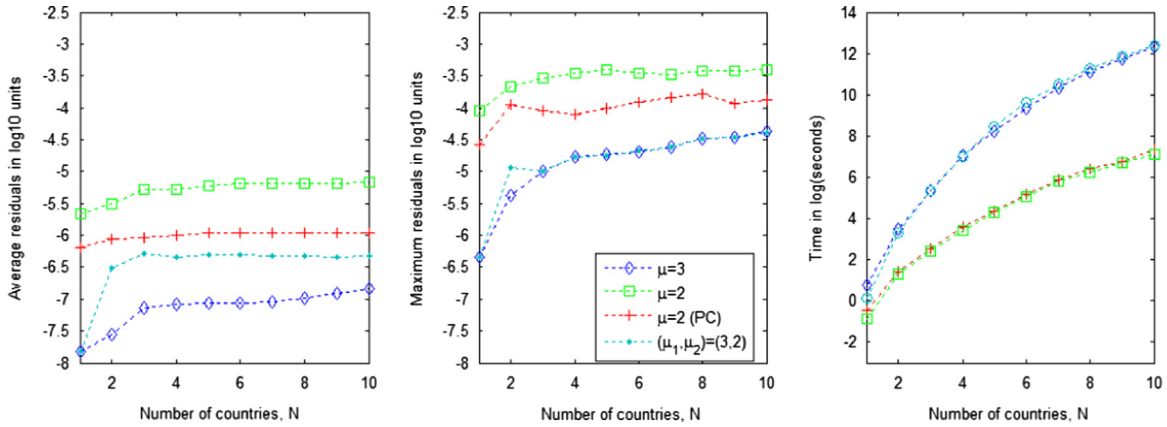

Fig. 12. Accuracy and running time for the multicountry model.

we should draw attention to the computational expense of our solution method. We observe a clearly concave pattern for the running time in the logarithmic scale. This means that the expense grows slower than an exponential function, i.e., our implementation does not appear to be subject to the curse of dimensionality in the sense of Bellman (1961).

Malin et al. (2011) also solve models with up to 10 countries (20 state variables); in particular, a symmetric specification of Model I in their analysis is similar to the model studied in the present paper. For this model, their cost in seconds is 1, 59, 916, 7313, 38150 when the number of countries is 2, 4, 6, 8, 10, respectively, which grows faster than the exponential function. They use the approximation level $\mu = 2$, and their program is written in Fortran. We differ in two respects: first, we implement the Smolyak method for interpolation by avoiding the repetitions, and second, we use a much cheaper version of fixed-point iteration than time iteration used in Malin et al. (2011). We solve a similar model with 10 countries in about 45 min using MATLAB. However, the third-level Smolyak approximation is expensive even for our efficient implementation: we need almost 45 h to solve a model with 10 countries, which increases accuracy by two orders of magnitude.

Thus, the adaptive domain allows us to make about 1/3 way in terms of accuracy between $\mu = 2$ and $\mu = 3$ without a visible increase in cost. The anisotropic grid with $(\mu_1, \mu_2) = (3, 2)$ gives us essentially the same accuracy as $\mu = 3$ but uses a considerably smaller number of grid points and basis function. However, our current implementation of the anisotropic Smolyak method does not allow us to translate a reduction in the number of Smolyak elements in a sizable cost reduction in this particular example. We first construct an isotropic Smolyak interpolant, and we next remove the terms accordingly to arrive to the target anisotropic construction. This procedure requires additional running time which reduces the savings from anisotropy.

7. Conclusion

The Smolyak method is designed to deal with high-dimensional applications. However, the cost of the Smolyak method still grows rapidly with dimensionality of the problem. In this paper, we propose a more efficient implementation of the Smolyak method that reduces its computational expense, and we present extensions of the Smolyak method that allow us to increase its accuracy level while maintaining a fixed computational cost. The analytical and numerical techniques developed in the present paper are not limited to economic applications but can be used in other fields.

First, we propose a more efficient implementation of the Smolyak method than the conventional one, namely, we avoid unnecessary repeated evaluations of basis functions when forming a Smolyak interpolant. Efficient implementation of interpolation is especially important in the context of numerical methods for solving dynamic economic models in which decision or value functions must be interpolated a very large number of times during the solution procedure, i.e., in each grid point, integration node or time period.

Second, we propose an anisotropic version of the Smolyak grid which allows us to vary the number of grid points and basic functions by dimension. In a typical economic application, we know some properties of decision and value functions, and we may use this knowledge to represent such functions more efficiently using the proposed anisotropic constructions. Maliar et al. (2014) discuss further strategies for improving the performance of the anisotropic Smolyak method.

Third, we show an effective transformation of the state space of a given economic model into a normalized hypercube used by the Smolyak method. We find that the best accuracy of approximations is attained when we use a minimum hypercube that encloses the high-probability set of a given economic model.

The above three improvements are related to interpolation. Our last improvement is concerned with an iterative procedure for solving dynamic economic models. Time iteration used in the existing Smolyak methods relies on a numerical solver while a version of fixed-point iteration used in the present paper involves only a straightforward computation. This improvement, although minor in substance, allows us to achieve substantial economizing on cost, especially, in high-dimensional applications.

Acknowledgments

We thank Editor Chris Otrok and two anonymous referees for many useful comments and suggestions. We also benefited from comments of participants of the 2012 CFE-ERCIM conference and the Summer 2013 Computation in CA Workshop at Stanford University. Lilia Maliar and Serguei Maliar acknowledge support from the Hoover Institution and Department of Economics at Stanford University, University of Alicante, Ivie, MECD and FEDER funds under the projects SEJ-2007-62656 and ECO2012-36719. Rafael Valero acknowledges support from MECD under the FPU program. Errors are ours.

Appendix

This section contains some supplementary results. In Appendix A, we describe how to construct unidimensional Smolyak grid points and basis functions. In Appendix B, we develop the formula for the Smolyak interpolant in the two-dimensional example with $\mu = 2$. In Appendix C, we show an example of constructing adaptive domain. Finally, in Appendix D, we describe the multicountry model and the solution algorithm.

Table 6
Extrema of Chebyshev polynomials and construction of Smolyak points in the unidimensional case.

n	Chebyshev polynomial of degree $n-1$ $T_{n-1}(x)=\cos((n-1)\cos^{-1}(x))$	n extrema of the polynomial of degree $n-1$ $\zeta_j^n=-\cos\left(\frac{\pi(j-1)}{n-1}\right), j=1,\ldots,n$
1	1	$\boxed{0}$
2	x	$-1,\quad 1$
3	$2x^2-1$	$\boxed{0\quad -1\quad 1}$
4	$4x^3-3x$	$-1,\quad 1,\quad -\frac{1}{2},\quad \frac{1}{2}$
5	$8x^4-8x^2+1$	$\boxed{0\quad -1\quad 1\quad -\frac{1}{\sqrt{2}}\quad \frac{1}{\sqrt{2}}}$

Appendix A. Unidimensional Smolyak grid points and basis functions

To construct multidimensional Smolyak grid points and basis functions, we must first specify unidimensional grid points and basis functions. Many choice are possible. For example, we can consider a family of ordinary polynomials, $\{1,x,x^2,\ldots\}$ and grid points generated by dividing the interval $[-1,1]$ into 2^{i-1} equal parts, $i\geq 2$ (for $i=1$ we assume a grid point 0). In this manner, for $i=2$, we have grid points $\{0,-1,1\}$ and we have basis functions $\{1,x,x^2\}$; for $i=3$ we have grid points $\{0,-1,1,-1/2,1/2\}$, and we use basis functions $\{1,x,x^2,x^3,x^4\}$, etc.

Another possibility is to use Chebyshev polynomials as basis functions and extrema of such polynomials as grid points. Approximations based on Chebyshev polynomials have two useful properties. First, there always exists a unique set of coefficients such that a Chebyshev polynomial function matches M given values at M grid points. Second, approximations are uniformly accurate, and error bounds are established. We stick to this choice in our analysis.

Chebyshev polynomials are defined in the interval $[-1,1]$ with a recursive relation: $T_0(x)=1$, $T_1(x)=x$, and $T_n(x)=2xT_{n-1}(x)-T_{n-2}(x)$ for $n\geq 2$.[8] Chebyshev polynomial of degree $n-1$ has n extrema. Let ζ_j^n be a jth extremum of Chebyshev polynomial of degree $n-1$ with $j=1,\ldots,n$,

$$\zeta_j^n=-\cos\left(\frac{\pi(j-1)}{n-1}\right).$$

Table 6 presents Chebyshev polynomials of degree $n-1$ and their n extrema (for the polynomial of degree 0, the extremum is assumed to be 0). Note that the sequence of unidimensional Chebyshev polynomials and their extrema cannot be used in Smolyak formula (5) because such sequence does not satisfy Conditions 1 and 2 of Section 2.5, namely, the number of extrema is equal to i, with $i=1,2,3,\ldots$, and not to $2^{i-1}+1$ as required by Condition 1, and the consecutive sets are not nested as required by Condition 2. However, there is a subsequence of this sequence that satisfies both Conditions 1 and 2, and is suitable for the conventional interpolation formula. Namely, we select a subsequence in which the number of extrema is $m(i)=1,3,5,9,17,\ldots$ for $i=1,2,3,4,5\ldots$, respectively (the first three sets of such a sequence are in-boxed elements in the last column of the table).

Therefore, the unidimensional Smolyak basis functions and grid points are as follows: for $i=1$, a grid point is $\{0\}$ and a basis function is $\{1\}$; for $i=2$, grid points are $\{0,-1,1\}$ and basis functions are $\{1,x,2x^2-1\}$; for $i=3$, grid points are $\{0,-1,1,-1/\sqrt{2},1/\sqrt{2}\}$ and basis functions are $\{1,x,2x^2-1,4x^3-3x,8x^4-8x^2+1\}$, etc.

Appendix B. Smolyak interpolant under $\mu=2$

We compare two alternative formulas for Smolyak interpolation in the two-dimensional case under the approximation level $\mu=2$. One is the conventional formula with repeated basis functions and the other is an alternative formula introduced in the present paper.

Conventional Smolyak interpolation formula: We consider the conventional Smolyak formula for interpolation in the two-dimensional case, $d=2$, under the approximation level $\mu=2$. Condition $\max(d,\mu+1)\leq|i|\leq\mu+d$ in (5) becomes $2\leq i_1+i_2\leq 4$. We use (7) to form p^{i_1,i_2}. In particular, $p^{1,1}$, $p^{1,2}$ and $p^{2,1}$ are given by (8)–(10), respectively. For the remaining polynomials, $p^{2,2}$, $p^{3,1}$ and $p^{1,3}$, we have

$$p^{2,2}=\sum_{\ell_1=1}^{m(2)}\sum_{\ell_2=1}^{m(2)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{12}\psi_2(y)+b_{22}\psi_2(x)\psi_2(y)+b_{32}\psi_3(x)\psi_2(y)$$
$$+b_{13}\psi_3(y)+b_{23}\psi_2(x)\psi_3(y)+b_{33}\psi_3(x)\psi_3(y),$$
$$p^{3,1}=\sum_{\ell_1=1}^{m(3)}\sum_{\ell_2=1}^{m(1)}b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{41}\psi_4(x)+b_{51}\psi_5(x),$$

[8] In terms of basis function ψ that was used in the main text and that is used later in Appendix B, we have $T_{n-1}(x)=\psi_n(x)$.

$$p^{1,3}=\sum_{\ell_1=1}^{m(1)}\sum_{\ell_2=1}^{m(3)} b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{11}+b_{12}\psi_2(y)+b_{13}\psi_3(y)+b_{14}\psi_4(y)+b_{15}\psi_5(y).$$

Furthermore, $p^{|2|}$ and $p^{|3|}$ are defined before in (11) and (12), respectively. A new combination of polynomials with $|i|=i_1+i_2=4$ is given by

$$p^{|4|}\equiv p^{2,2}+p^{3,1}+p^{1,3}.$$

Smolyak polynomial function (7) for the case $\mu=2$ is given by

$$\begin{aligned}\widehat{f}^{2,2}(x,y;b)&=\sum_{\max(d,\mu+1)\le|i|\le d+\mu}(-1)^{d+\mu-|i|}\binom{d-1}{d+\mu-|i|}p^{|i|}\\&=\sum_{3\le|i|\le 4}(-1)^{4-|i|}\binom{1}{4-|i|}p^{|i|}=\sum_{3\le|i|\le 4}(-1)^{4-|i|}\frac{1}{(4-|i|)}p^{|i|}\\&=-1\cdot p^{|3|}+1\cdot p^{|4|}=-1\cdot(p^{2,1}+p^{1,2})+1\cdot(p^{2,2}+p^{3,1}+p^{1,3})\\&=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{12}\psi_2(y)+b_{22}\psi_2(x)\psi_2(y)\\&\quad+b_{32}\psi_3(x)\psi_2(y)+b_{13}\psi_3(y)+b_{23}\psi_2(x)\psi_3(y)+b_{33}\psi_3(x)\psi_3(y)\\&\quad+b_{41}\psi_4(x)+b_{51}\psi_5(x)+b_{14}\psi_4(y)+b_{15}\psi_5(y).\end{aligned}$$

As expected, the conventional Smolyak formula gives us the same thirteen basis functions as distinguished in (19).

Smolyak interpolation formula without repeated basis functions: Let us now illustrate the use of interpolation formula (25) without repetitions. Here, we have $d\le|i|\le\mu+d$, which means $2\le i_1+i_2\le 4$. We use formula (27) to form q^{i_1,i_2}. In particular, $q^{1,1}$, $q^{1,2}$ and $q^{2,1}$ are given by (28)–(30), respectively. For the remaining polynomials, $q^{2,2}$, $q^{3,1}$ and $q^{1,3}$, we obtain

$$\begin{aligned}q^{2,2}&=\sum_{\ell_1=m(1)+1}^{m(2)}\sum_{\ell_2=m(1)+1}^{m(2)} b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{22}\psi_2(x)\psi_2(y)+b_{32}\psi_3(x)\psi_2(y)\\&\quad+b_{13}\psi_3(y)+b_{23}\psi_2(x)\psi_3(y)+b_{33}\psi_3(x)\psi_3(y),\\q^{3,1}&=\sum_{\ell_1=m(2)+1}^{m(3)}\sum_{\ell_2=m(0)+1}^{m(1)} b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{41}\psi_4(x)+b_{51}\psi_5(x),\\q^{1,3}&=\sum_{\ell_1=m(0)+1}^{m(1)}\sum_{\ell_2=m(2)+1}^{m(3)} b_{\ell_1\ell_2}\psi_{\ell_1}(x)\psi_{\ell_2}(y)=b_{14}\psi_4(y)+b_{15}\psi_5(y).\end{aligned}$$

Furthermore, $q^{|2|}$ and $q^{|3|}$ are defined before in (31) and (32), respectively. A new sum with $|i|=i_1+i_2=4$ is given by

$$q^{|4|}\equiv q^{2,2}+q^{3,1}+q^{1,3}.$$

The Smolyak polynomial function (27) for the case of $\mu=2$ is given by

$$\begin{aligned}\widehat{f}(x,y;b)&=\sum_{d\le|i|\le d+\mu}q^{|i|}=q^{|2|}+q^{|3|}+q^{|4|}=b_{11}+b_{21}\psi_2(x)+b_{31}\psi_3(x)+b_{12}\psi_2(y)+b_{22}\psi_2(x)\psi_2(y)\\&\quad+b_{32}\psi_3(x)\psi_2(y)+b_{13}\psi_3(y)+b_{23}\psi_2(x)\psi_3(y)+b_{33}\psi_3(x)\psi_3(y)\\&\quad+b_{41}\psi_4(x)+b_{51}\psi_5(x)+b_{14}\psi_4(y)+b_{15}\psi_5(y).\end{aligned}$$

Again, as expected, our interpolation formula gives the same 13 basis functions as those distinguished in (19) for the conventional Smolyak interpolation formula.

Appendix C. Adaptive domain

We show how to adapt the Smolyak hypercube to the high-probability area of the state space in the context of the representative agent model (38)–(40). Our objective is to define a mapping Φ that transforms each possible value of state variables (k,θ) into $(x,y)\in[-1,1]^2$. Below, we show two ways of constructing the mapping Φ, one uses the original coordinates and the other uses principal components of the original coordinates.

Standard hypercube: Consider a cloud of simulated data $\{k_t,\theta_t\}_{t=1,\dots,T}$ shown in Fig. 6. Let us define $[\underline{k},\overline{k}]$ and $[\underline{\theta},\overline{\theta}]$ as intervals for the state variables that we observe in simulation (the rectangle $[\underline{k},\overline{k}]\times[\underline{\theta},\overline{\theta}]$ is the larger rectangle shown in Fig. 6. Consider the following linear transformation of $(k,\theta)\in[\underline{k},\overline{k}]\times[\underline{\theta},\overline{\theta}]$ into $(x,y)\in[-1,1]^2$

$$x=2\frac{k-\underline{k}}{\overline{k}-\underline{k}}-1\quad\text{and}\quad y=2\frac{\theta-\underline{\theta}}{\overline{\theta}-\underline{\theta}}-1. \tag{49}$$

By using (49) and its inverse, we can move from $[\underline{k},\overline{k}]\times[\underline{\theta},\overline{\theta}]$ to $[-1,1]^2$. Malin et al. (2011) set bounds exogenously at $[\underline{k},\overline{k}]=[0.8,1.2]$ and $[\underline{\theta},\overline{\theta}]=[\exp(-0.8\sigma/(1-\rho)),\exp(0.8\sigma/(1-\rho))]$, where the steady state of both capital and productivity level is one. Our own analysis shows that the best accuracy is attained if the intervals $[\underline{k},\overline{k}]$ and $[\underline{\theta},\overline{\theta}]$ are chosen to enclose the set of simulated data as shown in Fig. 6.

Adaptive parallelotope: We now describe how to construct the adaptive (inclined) rectangle in Fig. 6. To map the state variables into a unit square, we use a principal component (PC) transformation of the time series as described in Section 5

and illustrated in Fig. 7. We first normalize the simulated data $\{k_t,\theta_t\}_{t=1,\ldots,T}$ to have zero mean and unit variance by

$$\tilde{k}_t=\frac{k_t-\mu_k}{\sigma_k}\quad\text{and}\quad\tilde{\theta}_t=\frac{\theta_t-\mu_\theta}{\sigma_\theta},\tag{50}$$

where μ_k and μ_θ are means, and σ_k and σ_θ are standard deviations of of capital and shock, respectively. We then compute the SVD decomposition $Y=USV^\top$, where

$$Y\equiv\begin{bmatrix}\tilde{k}_1 & \tilde{\theta}_1\\ \vdots & \vdots\\ \tilde{k}_T & \tilde{\theta}_T\end{bmatrix},$$

$U\in\mathbb{R}^{T\times 2}$ and $V\in\mathbb{R}^{2\times 2}$ are orthogonal matrices, and $S\in\mathbb{R}^{2\times 2}$ is a diagonal matrix of singular values. We find $Z\equiv YV$, where

$$Z=\begin{bmatrix}z_1^x & z_1^y\\ \vdots & \vdots\\ z_T^x & z_T^y\end{bmatrix}\in\mathbb{R}^{T\times 2}.$$

Let $[\underline{z}^x,\overline{z}^x]$ and $[\underline{z}^y,\overline{z}^y]$ be the intervals for the resulting principal components $\{z_t^x,z_t^y\}_{t=1,\ldots,T}$. We map each $(z^x,z^y)\in[\underline{z}^x,\overline{z}^x]\times[\underline{z}^y,\overline{z}^y]$ into $(x,y)\in[-1,1]^2$ using

$$x=2\frac{z^x-\underline{z}^x}{\overline{z}^x-\underline{z}^x}-1\quad\text{and}\quad y=2\frac{z^y-\underline{z}^y}{\overline{z}^y-\underline{z}^y}-1.\tag{51}$$

To go back to the state space of the model, we first find (z^x,z^y) that correspond to a given pair (x,y) using (51), we then apply an inverse PC transformation to get $(\tilde{k},\tilde{\theta})^\top=V^{-1}(z^x,z^y)^\top$ and finally, we compute (k,θ) using the inverse of (50).

In our experiments, we typically recompute solutions two times: first, we solve the model using the standard rectangular domain in the original system of coordinates for some guess $[\underline{k},\overline{k}]\times[\underline{\theta},\overline{\theta}]$. After the decision functions were computed, we simulate the model and use the time series solution $\{k_t,\theta_t\}_{t=1,\ldots,T}$ either to refine the guess on the bounds $\underline{k}$, $\overline{k}$, $\underline{\theta}$ and $\overline{\theta}$ for constructing the conventional domain or to compute μ_k, μ_θ, σ_k, σ_θ, V, $\underline{z}^x$, $\overline{z}^x$, $\underline{z}^y$ and $\overline{z}^y$ for constructing the adaptive domain.

Appendix D. Smolyak-based projection methods for problems with high dimensionality

We describe the multicountry model studied in Section 6.7 and provide further computational details about the Smolyak method that was used to produce numerical solutions in that section.

D.1. Multicountry model

We test the performance of the Smolyak-based projection method in the context of the multi-agent model studied in Judd et al. (2010, 2011) and Maliar and Maliar (2014b). This allows us to compare the performance of the Smolyak method with other approaches tractable in high-dimensional applications.

A world economy consists of a finite number of agents N, interpreted as countries. Each country $h\in\{1,\ldots,N\}$ is populated by a representative consumer. A social planner solves the following maximization problem:

$$\max_{\{c_t^h,k_{t+1}^h\}_{t=0,\ldots,\infty}^{h=1,\ldots,N}}E_0\sum_{h=1}^{N}\lambda^h[\sum_{t=0}^{\infty}\beta^t u^h(c_t^h)]\tag{52}$$

subject to the aggregate resource constraint,

$$\sum_{h=1}^{N}c_t^h+\sum_{h=1}^{N}k_{t+1}^h=\sum_{h=1}^{N}k_t^h(1-\delta)+\sum_{h=1}^{N}\theta_t^h Af^h(k_t^h),\tag{53}$$

and to the process for the countries' productivity levels,

$$\ln\theta_{t+1}^h=\rho\ln\theta_t^h+\epsilon_{t+1}^h,\quad h=1,\ldots,N,\tag{54}$$

where initial condition $\{k_0^h,\theta_0^h\}^{h=1,\ldots,N}$ is exogenously given, and the productivity shocks follow a multivariate Normal distribution $(\epsilon_{t+1}^1,\ldots,\epsilon_{t+1}^N)^\top\sim\mathcal{N}(0_N,\Sigma)$ with $0_N\in\mathbb{R}^N$ being a vector of zero means and $\Sigma\in\mathbb{R}^{N\times N}$ being a variance-covariance matrix. We assume that shocks of different countries are given by $\epsilon_{t+1}^h=\varsigma_t^h+\varsigma_t$, $h=1,\ldots,N$, where $\varsigma_t^h\sim\mathcal{N}(0,\sigma^2)$ is a country-specific component, and $\varsigma_t\sim\mathcal{N}(0,\sigma^2)$ is a worldwide component. The resulting variance covariance matrix is

$$\Sigma=\begin{pmatrix}2\sigma^2 & \cdots & \sigma^2\\ \cdots & \ddots & \cdots\\ \sigma^2 & \cdots & 2\sigma^2\end{pmatrix}.$$

In the problem (52)–(54), E_t is the operator of conditional expectation; c_t^h, k_t^h, θ_t^h and λ^h are a country's h consumption, capital, productivity level and welfare weight, respectively; $\beta\in(0,1)$ is the discount factor; $\delta\in(0,1]$ is the

depreciation rate; A is a normalizing constant in the production function; $\rho \in (-1,1)$ is the autocorrelation coefficient. The utility and production functions, u^h and f^h, respectively, are strictly increasing, continuously differentiable and concave. We assume that all countries have identical preferences and technology, i.e. $u^h = u$ and $f^h = f$ for all h. Under these assumptions, the planner assigns equal weights, $\lambda^h = 1$, and therefore, equal consumption to all countries, $c_t^h = c_t$ for all $h = 1, \ldots, N$.

D.2. Algorithm for the multicountry model

The solution to the model (52)–(54) satisfies N Euler equations:

$$k_{t+1}^h = E_t\left\{\beta\frac{u'(c_{t+1})}{u'(c_t)}\left[1-\delta+\theta_{t+1}^h A f'\left(k_{t+1}^h\right)\right]k_{t+1}^h\right\}, \quad h = 1, \ldots, N, \tag{55}$$

where u' and f' are the first derivatives of u and f, respectively.

We approximate the planner's solution as N capital policy functions $K^h(\{k_t^h, \theta_t^h\}^{h=1,\ldots,N})$. Note that our approximating functions $\widehat{K}^h(\{k_t^h, \theta_t^h\}^{h=1,\ldots,N}; b^h)$, $h = 1, \ldots, N$, are country-specific. In effect, we treat countries as completely heterogeneous even if they are identical in fundamentals and have identical optimal policy functions. This allows us to assess costs associated with computing solutions to models with heterogeneous preferences and technology.

Steps of the Smolyak-based projection method: The Smolyak method, described for the representative agent model in the main text, can be readily extended to the multicountry case.

- *Initialization*:

  (a) Choose the level of approximation, μ.

  (b) Parameterize $(k^h)' = K^h(\{k_t^h, \theta_t^h\}^{h=1,\ldots,N})$ with $\widehat{K}^h(\{k_t^h, \theta_t^h\}^{h=1,\ldots,N}; b^h)$ which is a Smolyak interpolant constructed using Chebyshev unidimensional basis functions.

  (c) Construct a Smolyak grid $\mathcal{H}^{2N,\mu} = \{x_n^h, y_n^h\}_{n=1,\ldots,M}^{h=1,\ldots,N}$ on the hypercube $[-1,1]^{2N}$, as described in Sections 3 and 4 for isotropic and anisotropic cases, respectively.

  (d) Compute Smolyak basis functions $\mathcal{P}^{2N,\mu}$ in each grid point n as described in Section 3 for the isotropic case or in Section 4 for the anisotropic case. The resulting $M \times M$ matrix is $\mathcal{B}$.

  (e) Choose the relevant ranges of values for $\{k_t^h, \theta_t^h\}^{h=1,\ldots,N}$ on which a solution is computed. The resulting hypercube is $[\underline{k}^1, \overline{k}^1] \times \cdots \times [\underline{\theta}^N, \overline{\theta}^N]$.

  (f) Construct a mapping between points $\{k_n^h, \theta_n^h\}^{h=1,\ldots,N}$ in the original hypercube $[\underline{k}^1, \overline{k}^1] \times \cdots \times [\underline{\theta}^N, \overline{\theta}^N]$ and points $\{x_n^h, y_n^h\}^{h=1,\ldots,N}$ in the normalized hypercube $[-1,1]^{2N}$ using either a linear change of variables of type (49) or principal component transformation of type (51); see Section 5.

$$\Phi: [\underline{k}^1, \overline{k}^1] \times \cdots \times [\underline{\theta}^N, \overline{\theta}^N] \to [-1,1]^{2N}. \tag{56}$$

  (g) Choose integration nodes, $\{\epsilon_j^h\}^{h=1,\ldots,N}$, and weights, ω_j, $j = 1, \ldots, J$.

  (h) Construct next-period productivity, $\{(\theta_{n,j}^h)'\}^{h=1,\ldots,N}$ with $(\theta_{n,j}^h)' = (\theta_n^h)^\rho \exp(\epsilon_j^h)$ for all j and n.

  (k) Make an initial guess on the coefficients vectors $(b^1)^{(1)}, \ldots, (b^N)^{(1)}$.

- *Iterative cycle*: *At iteration i, given $(b^1)^{(i)}, \ldots, (b^N)^{(i)}$, perform the following steps.*
- *Step* 1: *Computation of the capital choice.*
  Compute $(k_n^h)' = \mathcal{B}_n (b^h)^{(i)}$, where $\mathcal{B}_n$ is the nth raw of $\mathcal{B}$ for $n = 1, \ldots, M$.
- *Step* 2: *Computation of the consumption choice.*
  Compute $\{c_n^h\}^{h=1,\ldots,N}$ satisfying (53) given $\{k_n^h, \theta_n^h, (k_n^h)'\}^{h=1,\ldots,N}$ for $n = 1, \ldots, M$.
- *Step* 3: *Approximation of conditional expectation.*
  For $n = 1, \ldots, M$,

  (a) compute:

  - $\{(x_{n,j}^h)', (y_{n,j}^h)'\}^{h=1,\ldots,N}$ that correspond to $\{(k_n^h)', (\theta_{n,j}^h)'\}^{h=1,\ldots,N}$ using the inverse of transformation (56);
  - Smolyak basis functions $\mathcal{P}^{2N,\mu}$ in each point $\{(x_{n,j}^h)', (y_{n,j}^h)'\}^{h=1,\ldots,N}$; the resulting $M \times M \times J$ matrix is $\mathcal{B}'_{n,j}$;
  - $(k_{n,j}^h)'' = \mathcal{B}'_{n,j}(b^h)^{(i)}$, where $\mathcal{B}'_{n,j}$ are basis functions evaluated in $\{(k_n^h)', (\theta_{n,j}^h)'\}^{h=1,\ldots,N}$ using the transformation (56) for all j;
  - $\{(c_{n,j}^h)'\}^{h=1,\ldots,N}$ satisfying (53) given $\{(k_n^h)', (\theta_{n,j}^h)', (k_{n,j}^h)''\}^{h=1,\ldots,N}$ for $n = 1, \ldots, M$;

evaluate conditional expectation:

(b) $$e_n^h \equiv \beta \sum_{j=1}^{J} \left\{ \omega_j \left(\frac{u_1^h((c_{n,j}^h)')}{u1(c_n^h)} \left[1-\delta+(\theta_{n,j}^h)' f_1^h\left((k_n^h)'\right) \right] (k_n^h)' \right) \right\}.$$

Step 4: *Computation of the coefficients.*

- Find $\{\widehat{b}^h\}^{h=1,\ldots,N}$ that solves $e_n^h = \mathcal{B}_n \widehat{b}^h$, i.e., $\widehat{b}^h = \mathcal{B}_n^{-1} e_n^h$.
- *Step* 5: *Updating of the coefficients vectors.*
  For each $h=1,\ldots,N$, compute the coefficients vector for the subsequent iteration $i+1$ using fixed-point iteration,
  $$(b^h)^{(i+1)} = (1-\xi)(b^h)^{(i)} + \xi \widehat{b}^h. \tag{57}$$

where $\xi \in (0,1)$ is a damping parameter.

Iterate on Steps 1–5 until convergence of the solution,

$$\frac{1}{MN\xi} \sum_{n=1}^{M} \sum_{h=1}^{N} \left| \frac{((k_n^h)')^{(i+1)} - ((k_n^h)')^{(i)}}{((k_n^h)')^{(i)}} \right| < \varpi, \tag{58}$$

where $((k_n^h)')^{(i+1)}$ and $((k_n^h)')^{(i)}$ are the hth country's capital choices on the grid obtained on iterations $i+1$ and i, respectively, and $\vartheta > 0$.

Computational details: To solve the model, we assume $u(c_t) = (c_t^{1-\gamma}-1)/(1-\gamma)$, $f(k_t) = k_t^{\alpha}$ with $\alpha = 0.36$, $\beta = 0.99$, $\rho = 0.95$ and we vary γ, δ and σ. We start iterations from an arbitrary initial guess on the capital decision function, $k_{t+1}^h = 0.9k_t^h + 0.1\theta_t^h$ for all $h=1,\ldots N$ (this guess matches the steady-state level of capital). To approximate integrals, we use a monomial integration rule with $2N$ nodes combined with Cholesky decomposition; see Judd et al. (2011) for a description of these techniques. We set the damping parameter in (57) at $\xi = 0.05$, and we set the tolerance parameter in convergence criterion (58) at ϖ.

References

Aruoba, S.B., Fernández-Villaverde, J., Rubio-Ramírez, J., 2006. Comparing solution methods for dynamic equilibrium economies. J. Econ. Dyn. Control 30, 2477–2508.

Aiyagari, R., 1994. Uninsured idiosyncratic risk and aggregate saving. Q. J. Econ. 109 (3), 659–684.

Barthelmann, V., Novak, E., Ritter, K., 2000. High-dimensional polynomial interpolation on sparse grids. Adv. Comput. Math. 12, 273–288.

Bellman, R.E., 1961. Adaptive Control Processes. Princeton University Press, Princeton, NJ.

Brumm, J., Scheidegger, S., 2013. Using Adaptive Sparse Grids to Solve High-Dimensional Dynamic Models. Manuscript. University of Zurich.

Bungartz, H., Griebel, M., 2004. Sparse grids. Acta Numer. 13, 147–269.

Carroll, K., 2005. The method of endogenous grid points for solving dynamic stochastic optimal problems. Econ. Lett. 91, 312–320.

Christiano, L., Fisher, D., 2000. Algorithms for solving dynamic models with occasionally binding constraints. J. Econ. Dyn. Control 24, 1179–1232.

Coleman, C., Lyon, S., 2013. Efficient Implementations of Smolyak's Algorithm for Function Approximation in Python and Julia. ⟨https://github.com/EconForge/Smolyak⟩.

Delvos, F., 1982. *d*-variate Boolean interpolation. J. Approx. Theory 34, 99–114.

Den Haan, W., 1990. The optimal inflation path in a Sidrauski-type model with uncertainty. J. Monet. Econ. 25, 389–409.

Fernández-Villaverde, J., Gordon, G., Guerrón-Quintana, P., Rubio-Ramírez, J. 2012. Nonlinear Adventures at the Zero Lower Bound. NBER Working Paper 18058.

Garcke, J., 2011. Sparse Grid Tutorial. Manuscript.

Gaspar, J., Judd, K., 1997. Solving large-scale rational-expectations models. Macroecon. Dyn. 1, 45–75.

Gavilan-Gonzalez, A., Rojas, J., 2009. Solving Portfolio Problems with the Smolyak-Parameterized Expectations Algorithm. Banco de España Working Paper 0838.

Gerstner, T., Griebel, M., 1998. Numerical integration using sparse grids. Numer. Algorithm 18, 209–232.

Gerstner, T., Griebel, M., 2003. Dimension-adaptive tensor-product quadrature. Computing 71, 65–87.

Griebel, M., 1998. Adaptive sparse grid multilevel methods for elliptic PDEs based on finite differences. Computing 61, 151–179.

Gordon, G., 2011. Computing Dynamic Heterogeneous-Agent Economies. Penn Institute for Economic Research Working Paper 11-018. Software: ⟨https://sites.google.com/site/greygordon/code⟩.

Huggett, M., 1993. The risk-free rate in heterogeneous-agent incomplete-insurance economies. J. Econ. Dyn. Control 17, 953–969.

Jakeman, J.D., Roberts, S.G., 2011. Local and Dimension Adaptive Sparse Grid Interpolation and Quadrature. Manuscript.

Judd, K., 1992. Projection methods for solving aggregate growth models. J. Econ. Theory 58, 410–452.

Judd, K., 1998. Numerical Methods in Economics. MIT Press, Cambridge, MA.

Judd, K., Maliar, L., Maliar, S., 2010. A Cluster-Grid Projection Method: Solving Problems with High Dimensionality. NBER Working Paper 15965.

Judd, K., Maliar, L., Maliar, S., 2011. Numerically stable and accurate stochastic simulation approaches for solving dynamic models. Quant. Econ. 2, 173–210.

Klimke, A., Wohlmutch, B., 2005. Algorithm 847: spinterp: piecewise multilinear hierarchical sparse grid interpolation in MATLAB. ACM Trans. Math. Softw. 31, 561–579.

Kollmann, R., Maliar, S., Malin, B., Pichler, P., 2011. Comparison of solutions to the multi-country real business cycle model. J. Econ. Dyn. Control 35, 186–202.

Krueger, D., Kubler, F., 2004. Computing equilibrium in OLG models with production. J. Econ. Dyn. Control 28, 1411–1436.

Krusell, P., Smith, A., 1998. Income and wealth heterogeneity in the macroeconomy. J. Polit. Econ. 106, 868–896.

Maliar, L., 2013. Assessing Gains from Parallel Computation on Supercomputers. SSRN 2270804.

Maliar, L., Maliar, S., 2005. Solving nonlinear stochastic growth models: iterating on value function by simulations. Econ. Lett. 87, 135–140.

Maliar, L., Maliar, S., 2013. Envelope condition method versus endogenous grid method for solving dynamic programming problems. Econ. Lett. 120, 262–266.

Maliar, L., Maliar, S., 2014a. Numerical methods for large scale dynamic economic models. In: Schmedders, K., Judd, K. (Eds.), Handbook of Computational Economics, vol. 3. , Elsevier Science, Amsterdam.

Maliar, L., Maliar, S., 2014b. Merging simulation and projection approaches to solve high-dimensional problems with an application to a new Keynesian model. Quant. Econ., forthcoming.

Maliar, S., Maliar, L., Judd, K., 2011. Solving the multi-country real business cycle model using ergodic set methods. J. Econ. Dyn. Control 35, 207–228.

Maliar, L., Maliar, S., Valero, R., 2014. Error Bounds for Anisotropic Smolyak Formula. Manuscript.
Malin, B., Krueger, D., Kubler, F., 2011. Solving the multi-country real business cycle model using a Smolyak-collocation method. J. Econ. Dyn. Control 35, 229–239. Software: ⟨http://economics.sas.upenn.edu/~dkrueger/research.php⟩.
Marcet, A., 1988. Solving Non-Linear Models by Parameterizing Expectations. Manuscript, Carnegie Mellon University, Graduate School of Industrial Administration.
Marcet, A., Lorenzoni, G., 1999. The parameterized expectation approach: some practical issues. In: Marimon, R., Scott, A. (Eds.), Computational Methods for Study of Dynamic Economies, Oxford University Press, New York, pp. 143–171.
Marcet, A., Sargent, T., 1989. Convergence of least-squares learning in environments with hidden state variables and private information. J. Polit. Econ. 97, 1306–1322.
Miranda, M., Helmberger, P., 1988. The effects of commodity price stabilization programs. Am. Econ. Rev. 78, 46–58.
Patterson, T., 1968. The optimum addition of points to quadrature formulae. Math. Comput. 22, 847–856.
Petras, K., 2001. Fast calculation of coefficients in the Smolyak algorithm. Numer. Algorithms 26, 93–109.
Powell, W., 2011. Approximate Dynamic Programming. Hoboken, New Jersey, Wiley.
Quarteroni, A., Sacco, R., Saleri, F., 2000. Numerical Mathematics. Springer, New York.
Rios-Rull, J.V., 1997. Computing of Equilibria in Heterogeneous Agent Models. Federal Reserve Bank of Minneapolis Staff Report 231.
Rust, J., 1997. Using randomization to break the curse of dimensionality. Econometrica 65, 487–516.
Smith, A., 1991. Solving Stochastic Dynamic Programming Problems Using Rules of Thumb. Queen's University. Economics Department. Discussion Paper 816.
Smith, A., 1993. Estimating nonlinear time-series models using simulated vector autoregressions. J. Appl. Econ. 8, S63–S84.
Smolyak, S., 1963. Quadrature and interpolation formulas for tensor products of certain classes of functions. Dokl. Akad. Nauk 148, 1042–1045.
Tauchen, G., Hussey, R., 1991. Quadrature-based methods for obtaining approximate solutions to nonlinear asset pricing models. Econometrica 59, 371–396.
Valero, R., Maliar, L., Maliar, S., 2013. Parallel Speedup or Parallel Slowdown: Is Parallel Computation Useful for Solving Large-Scale Dynamic Economic Models? Manuscript.
Wasilkowski, G., Woźniakowski, H., 1995. Explicit cost bounds of algorithms for multivariate tensor-product problems. J. Complex. 11, 1–56.
Winschel, V., Krätzig, M., 2010. Solving, estimating and selecting nonlinear dynamic models without the curse of dimensionality. Econometrica 78 (2), 803–821.
Wright, B., Williams, J., 1984. The welfare effects of the introduction of storage. Q. J. Econ. 99, 169–192.