Journal of Economic Dynamics & Control 35 (2011) 229–239

Contents lists available at ScienceDirect

Journal of Economic Dynamics & Control

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

Solving the multi-country real business cycle model using a Smolyak-collocation method

Benjamin A. Malin[a], Dirk Krueger[b,*], Felix Kubler[c]

[a] *Federal Reserve Board of Governors, USA*
[b] *Department of Economics, University of Pennsylvania, 3718 Locust Walk, Philadelphia, PA 19104, CEPR and NBER, USA*
[c] *University of Zurich and Swiss Finance Institute, Switzerland*

ARTICLE INFO

Article history:
Received 11 March 2010
Accepted 25 August 2010
Available online 1 October 2010

JEL classification:
C68
C63
F41

Keywords:
Sparse grids
Collocation
International real business cycles

ABSTRACT

We describe a sparse-grid collocation method to compute recursive solutions of dynamic economies with a sizable number of state variables. We show how powerful this method can be in applications by computing the non-linear recursive solution of an international real business cycle model with a substantial number of countries, complete insurance markets and frictions that impede frictionless international capital flows. In this economy, the aggregate state vector includes the distribution of world capital across different countries as well as the exogenous country-specific technology shocks. We use the algorithm to efficiently solve models with up to 10 countries (i.e., up to 20 continuous-valued state variables).

1. Introduction

In this paper we propose a projection method based on Smolyak's (1963) algorithm to compute *globally accurate* solutions to models characterized by a sizeable number of continuous-valued state variables, such as international real business cycle models with a substantial number of countries. The use of a sparse grid constructed with Smolyak's algorithm allows us to handle (at least) 20 state variables, while solving the model with this many state variables is not possible when standard full grids are used.

Our objectives are twofold. First, we aim at providing an easily accessible general description of our algorithm, replacing and improving upon Krueger and Kubler (2004). Second, we show how powerful this method is by numerically solving an international real business cycle model with many countries and international capital market frictions.[1] The introductory articles to this issue, Denhaan et al. (this issue) and Juillard and Villemont (this issue), together provide a full description of the international real business cycle model that is to be solved. We therefore only repeat it here insofar as is needed for the description of the algorithm. Furthermore, Juillard and Villemont (this issue) describe the accuracy tests with which our and competing solution methods are evaluated, while the paper by Kollmann et al. (this issue-b) compares the

[*] Corresponding author. Tel.: +1 215 898 6691; fax: +1 215 573 2057.
E-mail address: dkrueger@econ.upenn.edu (D. Krueger).

[1] For a description and application of this class of models, see e.g. Backus et al. (1992, 1995).

doi:10.1016/j.jedc.2010.09.015

performance of our algorithm relative to these competing methods.[2] We therefore defer the detailed discussion of the performance of the algorithm to the latter paper. To summarize the main findings, our sparse grid projection method performs quite well for a wide variety of model specifications including models with up to 10 countries (i.e., 20 continuous-valued state variables), specifications that introduce a great deal of curvature into utility and production functions, and models with asymmetries between countries. Our method is also substantially more accurate than a linear approximation of the solution, especially when the exogenous shocks to the economy are large.

Section 2 provides a general description of our projection method, and Section 3 discusses key implementation details. The final section offers some short concluding remarks.

2. A sparse-grid collocation method

The model we solve, to demonstrate the scope as well as the advantages and shortcomings of our method, is the international real business cycle model with N countries and capital adjustment costs.

2.1. The application

Due to adjustment costs, the state variables for the recursive formulation of the social planner's problem consist of the vector of exogenous current productivity levels $a=(a^1,\ldots,a^N)$ and the vector of endogenous current capital stocks $k=(k^1,\ldots,k^N)$. Denote by $s=(k,a)$ the current state, which is of dimension $2N$. We defer a complete description of the model, the interpretation of the functional forms and their parameterization to the introductory article, Juillard and Villemont (this issue), and in this section only develop the notation needed to describe the application of the Smolyak algorithm to this model.

The planner's problem can be written recursively as

$$V(k,a)=\max_{\{c^j,l^j,k'j\}}\sum_{j=1}^{N}\tau^j u^j(c^j,l^j)+\beta\int V(k',a')g_a(a')\,da' \tag{1}$$

s.t.

$$\ln(a'j)=\rho\ln(a^j)+\sigma(e'j+e') \tag{2}$$

$$\sum_{j=1}^{N}\left(c^j+k'j+\frac{\phi}{2}\frac{(k'j-k^j)^2}{k^j}\right)=\sum_{j=1}^{N}(a^jf^j(k^j,l^j)+k^j) \tag{3}$$

where $g_a(a')$ denotes the probability density function over a', given a. We will now derive the system of functional equations used to compute this model. We seek functions $C^j(s)$, $L^j(s)$, and $K'j(s)$ for $j=1,\ldots,N$, mapping the current state $s=(k,a)$ into consumption and labor supply of each country today and its capital stock tomorrow. For future reference we define

$$C(s)=\sum_{j=1}^{N}C^j(s) \tag{4}$$

$$Y(s)=\sum_{j=1}^{N}a^jf^j(k^j,L^j(s)) \tag{5}$$

$$K=\sum_{j=1}^{N}k^j \tag{6}$$

$$K'(s)=(K'1(s),\ldots,K'N(s)) \tag{7}$$

Attaching Lagrange multiplier λ to the resource constraint, we find as first-order conditions

$$\tau^j u_c^j(c^j,l^j)=\lambda \quad \forall j \tag{8}$$

$$\frac{\tau^j u_l^j(c^j,l^j)}{-a^jf_l^j(k^j,l^j)}=\lambda \quad \forall j \tag{9}$$

$$\frac{\beta\int V_{k^j}(k',a')g_a(a')\,da'}{1+\frac{\phi(k'j-k^j)}{k^j}}=\lambda \quad \forall j \tag{10}$$

[2] Alternative algorithms described in this issue include those developed by Kollmann et al. (this issue-a), Maliar et al. (this issue), and Pichler (this issue).

where lower case letters attached to functions denote partial derivatives of the function with respect to the corresponding argument. The envelope conditions read as

$$V_{k^j}(k,a)=\lambda\left[(1+a^j f_k^j(k^j,l^j)+\frac{\phi}{2}\frac{(k'j-k^j)(k'j+k^j)}{{k^j}^2}\right] \tag{11}$$

$$=\tau^j u_c^j(c^j,l^j)\left[(1+a^j f_k^j(k^j,l^j)+\frac{\phi}{2}\frac{(k'j-k^j)(k'j+k^j)}{{k^j}^2}\right] \qquad \forall j \tag{12}$$

Combining the first-order conditions and the envelope conditions gives, replacing choices by policy functions and abusing notation by writing $s'=(K'(s),a')$:

$$\tau^j u_c^j(C^j(s),L^j(s))=\frac{\beta\int\tau^j u_c^j(C^j(s'),L^j(s'))\begin{bmatrix}1+a'jf_k^j(K'j(s),L^j(s'))\\+\dfrac{\phi}{2}\dfrac{(K'j(s')-K'j(s))(K'j(s')+K'j(s))}{K'j(s)^2}\end{bmatrix}g_a(a')\,da'}{1+\dfrac{\phi(K'j(s)-k^j)}{k^j}} \qquad \forall j \tag{13}$$

which together with

$$\tau^j u_c^j(C^j(s),L^j(s))=\tau^i u_c^i(C^i(s),L^i(s)) \quad \forall i,j \tag{14}$$

$$\frac{u_l^j(C^j(s),L^j(s))}{u_c^j(C^j(s),L^j(s))}=-a^j f_l^j(k^j,L^j(s)) \quad \forall j \tag{15}$$

$$C(s)+\sum_{j=1}^{N}\left(K'j(s)+\frac{\phi}{2}\frac{(K'j(s)-k^j)^2}{k^j}\right)=Y(s)+K \tag{16}$$

provide $3N$ functional equations to be jointly solved for the $3N$ functions $\{C^j(s),\ L^j(s),\ K'j(s)\}_{j=1}^N$.

We will solve for an approximate equilibrium of the international real business cycle model using a Smolyak collocation method. The basic idea of collocation methods is to approximate the policy functions for consumption, labor and tomorrow's capital stock, $\{C^j(s),L^j(s),K'j(s)\}_{j=1}^N$, by weighted sums of "easier functions", e.g. by polynomials.

In order to determine the unknown coefficients of the polynomials, collocation methods require that Eqs. (13)–(16) hold exactly at finitely many points — the so-called collocation points. While collocation methods are routinely used in economics to solve non-linear dynamic models, our innovation is to use a Smolyak sparse grid method which allows us to consider fairly high-dimensional problems. The use of sparse grids is well established in numerical analysis (see e.g. Bungartz and Griebel, 2004, for an overview) and was first introduced, as far as we know, to economics by Krueger and Kubler (2004).

From a technical point of view, standard collocation methods pose three challenges for the problem considered here. First, the state space is high-dimensional[3] which means that the policy functions, which have to be approximated, are high-dimensional. Second, the conditional expectations in agents' Euler equations are high-dimensional integrals which have to be evaluated very frequently in the solution procedure. Lastly, one has to solve a rather large system of non-linear equations to obtain the unknown coefficients. While the focus of this paper is on the first problem (the high dimensionality of the state space), we will also discuss the issues associated with the second problem and propose a solution based on monomial rules (see Judd, 1998). Regarding the third problem, we chose to use a simple time-iteration scheme rather than methods that are likely more efficient. We briefly discuss this issue in Section 2.4.

2.2. Smolyak sparse grids

In order to motivate the choice of Smolyak points as collocation points, we consider the abstract problem of how to approximate an unknown policy function $f:[-1,1]^d\rightarrow\mathbb{R}$ by interpolating a finite number of known function values. That is, we try to find a finite number of points $\mathcal{H}\subset[-1,1]^d$ and a function $\hat{f}$ such that, given the points $(x_i,y_i{=}f(x_i))$ with $x_i\in\mathcal{H}$, the function $\hat{f}$ satisfies $\hat{f}(x_i)=f(x_i)$ and such that $\hat{f}$ approximates f well on its entire domain $[-1,1]^d$. Here, d is the dimension of the problem, in our application the dimension of the state space. The remaining questions are then how to choose the interpolation points $\mathcal{H}$ and how to choose the interpolating function $\hat{f}$.

Smolyak's (1963) method provides both sets of points and formulas for the approximating functions. To describe Smolyak's method, adopted to the problem of high-dimensional interpolation on sparse grids by Barthelmann et al. (2000), we start by defining a set of points in $[-1,1]^d$ which can be interpolated by polynomials of relatively low degree. Then we give a formula for the interpolating polynomial. This description is meant to be a more accessible version of the discussion in Krueger and Kubler (2004), which also follows Barthelmann et al. (2000).

[3] As argued above, it consists, in general, of the capital stocks in all countries as well as all country-specific technology shocks.

Since we know from the structure of the economy that the true policy function f is smooth, we use a multi-variate polynomial to approximate it.[4] In one dimension, it is well known that one can interpolate n points by a univariate polynomial of degree $n-1$, i.e. by a polynomial with n terms of the form

$$p_{n-1}(x)=\sum_{j=1}^{n}\theta_j x^{j-1} \tag{17}$$

In order to find the unknown n coefficients $(\theta_1,\ldots,\theta_n)$, one can simply use the n equations (which are linear in the unknowns)

$$y_i=\sum_{j=1}^{n}\theta_j(x_i)^{j-1}\quad \text{for } i=1,\ldots,n \tag{18}$$

It is now common in economics to use the orthogonal Chebychev polynomials to express the approximating function and write

$$p_{n-1}(x)=\sum_{j=1}^{n}\theta_j T_j(x) \tag{19}$$

where the Chebychev polynomials $T_1(\cdot),T_2(\cdot),\ldots$can be evaluated recursively by $T_1(x)=1$, $T_2(x)=x$ and $T_{j+1}(x)=2xT_j(x)-T_{j-1}(x)$. While (19) is just a different way to express the same function as in (17), the advantage of using Chebyshev polynomials is that, as discussed in Judd (1998), their orthogonality properties make the interpolation problem better conditioned. Although we use Chebychev polynomials in our method, we want to highlight that Smolyak's algorithm in no way depends on this particular set of polynomials.

While approximation of a function by interpolation is fairly straightforward in one dimension, it is much more complicated in several dimensions. As in the one-dimensional case, uniform convergence can only be achieved if the interpolation points are placed in a particular fashion. Moreover, in the higher-dimensional case, it is not true (in fact, this is the exception rather than the rule) that with a polynomial of n terms one can interpolate arbitrary n points. Any scheme for higher-dimensional interpolation has to take these two issues into account.

The simplest approach to multi-variate interpolation that does the job is to span a rectangular grid with n values in each dimension and use a tensor product of one-dimensional polynomials as a set of approximating functions. Thus one would approximate a d-dimensional function $f:[-1,1]^d\rightarrow\mathbb{R}$ by interpolating the function values at the n^d grid points by a polynomial of total degree[5] $(n-1)d$. The problem with this approach is that it becomes infeasible very fast as d becomes large because the number of unknown coefficients grows exponentially with the dimension d. This is the well-known "curse of dimensionality." This argument makes clear that any rule to choose these interpolation points has to satisfy, in order to be of practical use for high-dimensional problems, that the number of interpolation points and the number of terms in the interpolating polynomial does not grow exponentially in the dimensionality of the problem. Note that the issue is *not* that, for a given grid, we cannot find a sufficiently rich polynomial that interpolates this grid. As long as one is willing to add sufficiently many terms in the multi-dimensional polynomial, this is always possible. The objective here is to propose a method in which the number of grid points *and* the associated number of terms of the approximating polynomial do not grow exponentially in the dimensionality of the problem and in which one achieves uniform convergence as the number of points increases. In this paper we therefore advocate, as an alternative to tensor methods, "Smolyak grid points" and linear combinations of polynomials which interpolate in between these grid points. Incidentally, through the judicious choice of grid points and interpolating polynomial, the number of grid points and terms in the polynomial coincide.

In order to understand the method, note that even in one dimension, in order to approximate a smooth function on the interval $[-1,1]$ by interpolating n of its function values, one needs to carefully choose the nodes $x_1,\ldots,x_n$. It is well known (see e.g. Rivlin, 1969) that both the extrema and the zeros of the Chebychev polynomials have the following important uniform convergence properties. Given any continuously differentiable function $f:[-1,1]\rightarrow\mathbb{R}$, let p_{n-1} denote the unique

[4] It therefore might be problematic to apply our method to economies in which the functions that need to be approximated are not smooth (e.g. in instances in which the policy functions have kinks due to occasionally binding inequality constraints).

[5] The *degree* of a multi-variate polynomial is defined to be the maximal degree across all monomials. If the set of one-dimensional polynomials of degree $n-1$ (which has n terms) along dimension i is denoted by $P_{n-1}=\{p_{n-1}(x_i)\}$, the tensor product for d dimensions is given by the set

$$P_{n-1}^d=\left\{p(x)|p(x)=\prod_{i=1}^{d}p_{n-1}(x_i)\text{ for }p_{n-1}(x_i)\in P_{n-1}\right\}$$

For example, if $d=3$ and $n=2$, the set is given by

$$P_1^3=\{c,x_1,x_2,x_3,x_1x_2,x_1x_3,x_2x_3,x_1x_2x_3\}$$

and the *total degree* of the highest-order polynomial in P_1^3 is 3.

polynomial that interpolates f at the n Chebychev extrema (or the Chebychev zeros) and $\omega(\delta)$ be the modulus of continuity of the function f, i.e.

$$\omega(\delta) = \sup_{x_1,x_2 \in [-1,1], \|x_1-x_2\| \le \delta} \|f(x_1)-f(x_2)\|$$

Then we have

$$\max_{x \in [-1,1]} |p_{n-1}(x)-f(x)| \le 6(1+\Lambda_{n-1})\omega\left(\frac{1}{n-1}\right) \tag{20}$$

with $\Lambda_{n-1} \le C+(2/\pi)\log(n)$, where C is independent of n and of f. Unlike interpolation at uniformly spaced points, if the function f is interpolated at Chebychev zeros, the interpolating polynomial converges uniformly to the function as the number of points increases.

Following Barthelmann et al. (2000), we use the extrema of Chebychev polynomials as our basis for the grid of points $\mathcal{H}$. Denote $\mathcal{G}^1 = \{0\}$ and for n=2,..., define the sets $\mathcal{G}^n = \{\zeta_1,\dots,\zeta_n\} \subset [-1,1]$ as the set of the extrema of the Chebychev polynomials

$$\zeta_j = -\cos\left(\frac{\pi(j-1)}{n-1}\right), \quad j=1,\dots,n \tag{21}$$

Define a sequence of positive integers by $m(1)$=1 and $m(i)$=2^{i-1}+1 for i=2,3,.... It is easy to see, and this turns out to be crucial for Smolyak's construction, that the sets of interpolation points satisfy $\mathcal{G}^{m(i)} \subset \mathcal{G}^{m(i+1)}$ for all i. The construction of the integers $m(i)$ is therefore crucial in assuring that the sets of Chebychev extrema are *nested* as long as only sets of size $m(i)$ are permitted.

Smolyak's method uses this fact to build a hierarchical sparse grid out of combinations of the grids $\mathcal{G}^{m(i)}$ for different values of i. We first present a simple, albeit still abstract, three-dimensional example to illustrate the intuition of this idea and then move to a general description of the method.

2.2.1. The three-dimensional case

We choose three dimensions because we can represent our selection of grid points graphically, and two dimensions are not sufficient to clarify how exactly the method avoids the curse of dimensionality.

We want to approximate a smooth function $f:[-1,1]^3 \to \mathbb{R}$ by a polynomial of relatively small degree with few monomial terms, and we are looking for a method that is flexible in the sense that it is easy to add terms of higher degree and thereby increase the quality of the approximation. Let $\mu \in \mathbb{Z}_{++}$ denote a parameter that measures the fineness of the grid to be constructed. We then define a three-dimensional level$-\mu$ grid as follows:

$$\mathcal{H}^{3,\mu} = \bigcup_{(i_1,i_2,i_3) \in \mathbb{Z}^3_{++}:\ i_1+i_2+i_3 = 3+\mu} \mathcal{G}^{m(i_1)} \times \mathcal{G}^{m(i_2)} \times \mathcal{G}^{m(i_3)} \tag{22}$$

Recall that for $m(i)$ as defined above, the grids $\mathcal{G}^{m(i)}$ are nested for i=1,2,... . In order to understand how the formula constructs the interpolation points, it is useful to go through the cases $\mu=1,2,3$ one by one. Recall that $\mathcal{G}^{m(1)}=\{0\}$ and that $\mathcal{G}^{m(2)}$ consists of three points, $\{-1,0,1\}$. Then, according to Eq. (22),

$$\mathcal{H}^{3,1} = \mathcal{G}^{m(2)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(1)} \cup \mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)} \times \mathcal{G}^{m(1)} \cup \mathcal{G}^{m(1)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)}$$

So the first level grid consists of the seven points: $(-1,0,0)$, $(0,0,0)$, $(1,0,0)$, $(0,-1,0)$, $(0,1,0)$, $(0,0,-1)$ and $(0,0,1)$.

Now, let us move to the case $\mu=2$. Recall that $\mathcal{G}^{m(3)}$ consists of five points. Eq. (22) gives

$$\begin{aligned}\mathcal{H}^{3,2} = {} & \mathcal{G}^{m(3)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(1)} \cup \mathcal{G}^{m(1)} \times \mathcal{G}^{m(3)} \times \mathcal{G}^{m(1)} \cup \mathcal{G}^{m(1)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(3)} \\ & \cup \mathcal{G}^{m(2)} \times \mathcal{G}^{m(2)} \times \mathcal{G}^{m(1)} \cup \mathcal{G}^{m(2)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)} \cup \mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)} \times \mathcal{G}^{m(2)}\end{aligned}$$

Fig. 1 shows where these points are located in the three-dimensional cube $[-1,1]^3$. It first shows, for clarity, the point grids in two of the three dimensions, holding the third dimension fixed at 0, that is, at $\mathcal{G}^{m(1)}$. The lower-right panel then shows the three-dimensional grid, which is generated as the union of the three two-dimensional sets (see Eq. (22)). The figure also aims to clarify how the Smolyak grid achieves sparsity. To see this, we start from the two-dimensional grid in the upper-left panel (i.e., the x–y plane) and circle points that are added as we include the third dimension (i.e., the z-axis). The upper-right panel shows eight additional points, corresponding to the four non-origin points in $\mathcal{G}^{m(1)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(3)}$ and the four corner points of the $\mathcal{G}^{m(2)} \times \mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)}$ tensor grid. The lower-left panel shows four more additional points, the corner points of the $\mathcal{G}^{m(1)} \times \mathcal{G}^{m(2)} \times \mathcal{G}^{m(2)}$ tensor grid. The lower-right panel sums up by circling all 12 points that are added in moving from 2 to 3 dimensions. The key of the Smolyak construction is that there are relatively few of these added points as the dimension increases by one. For $\mu=2$, moving from $d-1$ to d dimensions requires $4d$ additional points: 4 points for the $\mathcal{G}^{m(3)}$ grid in the new dimension and 4 points for each of the new $d-1$ tensor grids. Thus, the number of grid points grows quadratically in d for $\mu=2$. More generally, the construction of the Smolyak grid guarantees that the number of grid points grows polynomially (i.e., quadratically for $\mu=2$, cubically for $\mu=3$), but not exponentially, in the dimension of the problem.

Note that $\mathcal{H}^{3,1} \subset \mathcal{H}^{3,2}$ and in general, as one can verify from Eq. (22), $\mathcal{H}^{3,\mu} \subset \mathcal{H}^{3,\mu+1}$. Passing from level μ to the next level of approximation $\mu+1$ can therefore be viewed as adding new points to the existing grid. This property is displayed in Fig. 2, which replicates Fig. 1 by plotting $\mathcal{H}^{3,2}$ but denotes all points $\mathcal{H}^{3,1} \subset \mathcal{H}^{3,2}$ with a circle, clarifying that $\mathcal{H}^{3,1}$ is nested within $\mathcal{H}^{3,2}$, and more generally, that $\mathcal{H}^{d,\mu} \subset \mathcal{H}^{d,\mu+1}$.

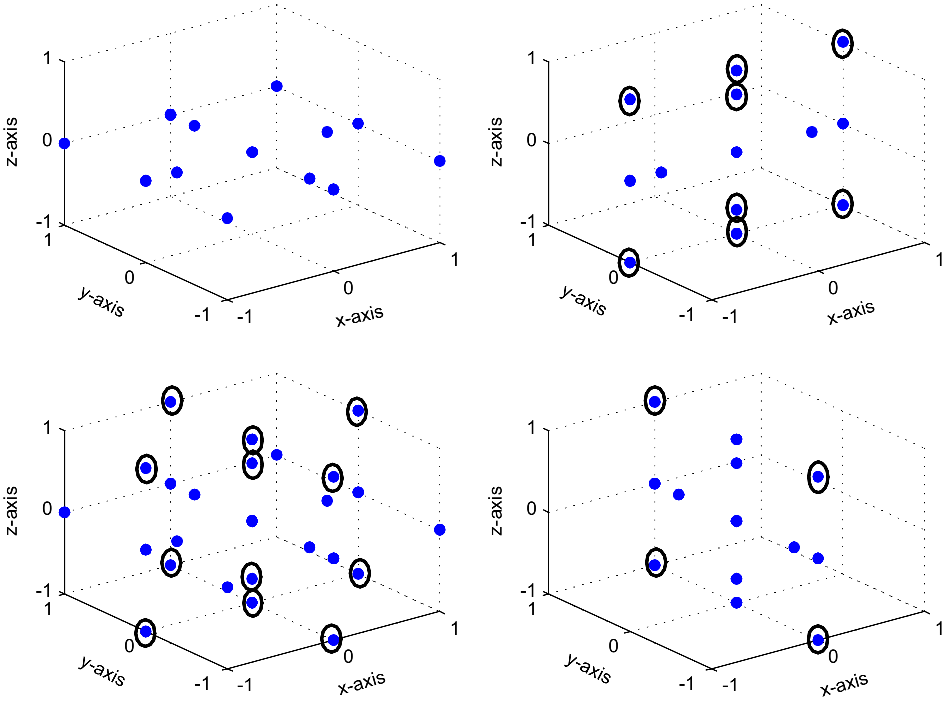

Fig. 1. Collocation points in three dimensions.

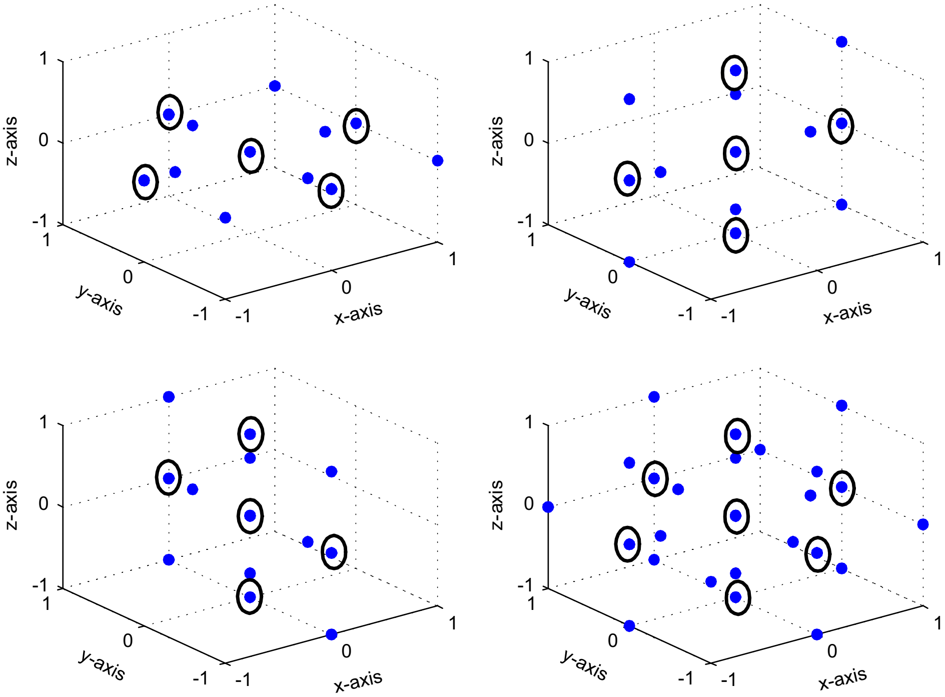

Fig. 2. $\mathcal{H}^{3,1}$ and $\mathcal{H}^{3,2}$.

At the risk of boring the reader, for clarification let us consider one more level of approximation, i.e. $\mu=3$ which leads to $m(4)$=9 points along each dimension (holding the two others fixed at zero). We have as the third level Smolyak grid

$$\begin{aligned}\mathcal{H}^{3,3} = {} & \mathcal{G}^{m(4)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(4)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(4)}\\ & \cup\mathcal{G}^{m(3)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(2)}\times\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}\\ & \cup\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(2)}\cup\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(3)}\\ & \cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(3)}\times\mathcal{G}^{m(2)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(3)}\\ & \cup\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\end{aligned}$$

2.2.2. *Grids in arbitrary dimensions*

In general, for approximation in the d-dimensional hypercube, we can construct grids $\mathcal{H}^{d,\mu}$ in exactly the same fashion. In order to give the general formula, for arbitrary dimension d and arbitrary approximation level μ, define a multi-index to be a vector of positive integers $\mathbf{i}=(i_1,\dots,i_d)\in\mathbb{Z}^d_{++}$ and let $|\mathbf{i}|=i_1+\cdots+i_d$. For integers $\mu\geq 1$, we can then define a *sparse* grid of points in $[-1,1]^d$ as follows:

$$\mathcal{H}^{d,\mu}=\bigcup_{\mathbf{i}:\ |\mathbf{i}|=d+\mu}(\mathcal{G}^{m(i_1)}\times\cdots\times\mathcal{G}^{m(i_d)}) \tag{23}$$

It can be easily verified that, for d=3 and $\mu=1,2,3$, this gives the sets of points which we described in the previous example. It turns out that for the international real business cycle model solved in this paper, a level $\mu=2$ construction is sufficient to obtain fairly high accuracy. Thus, for arbitrary dimension d, we always consider $\mathcal{H}^{d,2}$ as our set of points. Note that this is simply the union of d-dimensional sets of the form

$$\begin{aligned}&\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}\times\cdots\times\mathcal{G}^{m(3)}\times\cdots\times\mathcal{G}^{m(1)}\\&\mathcal{G}^{m(1)}\times\mathcal{G}^{m(2)}\times\cdots\times\mathcal{G}^{m(2)}\times\cdots\times\mathcal{G}^{m(1)}\end{aligned}$$

Since a complete enumeration of these sets is straightforward, the construction of $\mathcal{H}^{d,2}$ is very simple for arbitrary dimension d.

2.2.3. *Interpolation*

Given the construction of these points, we now briefly describe an easy way to construct an interpolating polynomial. Smolyak's method interpolates at nodes in $\mathcal{H}$, using weighted sums of polynomials which interpolate subsets of $\mathcal{H}$. Define $p^{\mathbf{i}}$ to be the tensor-product multi-variate polynomial which interpolates on $\mathcal{G}^{m(i_1)}\times\cdots\times\mathcal{G}^{m(i_d)}$. As pointed out above, we represent this in Chebychev form (that is, use Chebychev polynomials), i.e.

$$p^{\mathbf{i}}_\theta(s)=\sum_{l_1=1}^{m(i_1)}\cdots\sum_{l_d=1}^{m(i_d)}\theta_{l_1\dots l_d}T_{l_1}(s_1)\cdots T_{l_d}(s_d) \tag{24}$$

The coefficients $\theta_{l_1\dots l_d}$ can be efficiently computed as follows. Consider an arbitrary grid in d dimensions, with $k_1,\dots,k_d>1$ points along each dimension. Then the coefficients of an interpolating tensor product of the form in (24) are given in closed form as

$$\theta_{l_1\dots l_d}=\frac{2^d}{(k_1-1)\cdots(k_d-1)}\frac{1}{c_{l_1}\cdots c_{l_d}}\sum_{j_1=1}^{k_1}\cdots\sum_{j_d=1}^{k_d}\frac{T_{l_1}(\zeta_{j_1})\cdots T_{l_d}(\zeta_{j_d})\cdot f(\zeta_{j_1},\dots,\zeta_{j_d})}{c_{j_1}\cdots c_{j_d}} \tag{25}$$

with $c_1=c_{k_d}=2$ and c_j=1 for j=2,…,k_d-1.

Note that for our Smolyak construction, directions with only one point are then simply dropped, in the sense that if $m(i_d)$=1, we do not include this dimension in the computation of the coefficients in the formula above.

To return briefly to the three-dimensional example, $p^{(i,1,1)}$ is the polynomial of degree 2^{i-1} which interpolates $m(i)$=2^{i-1}+1 points in the first direction and is constant along the second and third dimensions. $p^{(2,2,1)}$ is the tensor product of two univariate polynomials of degree $m(2)$–1=2 and interpolates the function on the 3 by 3 grid $\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}$. For $\mu=2$, it then might seem that the interpolating polynomial for the entire grid $\mathcal{H}^{3,2}$ should be a weighted sum of the univariate polynomials in each direction as well as the two-dimensional tensor product on each plane. However, it turns out that things are not quite as simple, and that one also needs to include the polynomials that interpolate $\mathcal{H}^{3,1}$.

To see this, note that in order to interpolate the points in $\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}$, one would have to weight the polynomial $p^{2,2,1}$ with one. (Why? Because $p^{2,2,1}$ is the only interpolating polynomial that goes through the corner points of the tensor grid in the upper-left panel of Fig. 2.) But then, to interpolate points in $\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}$ and $\mathcal{G}^{m(1)}\times\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}$ which are not in the previous tensor grid, one would have to weight the polynomials $p^{3,1,1}$ and $p^{1,3,1}$ with one. (Why? Because these polynomials are the only ones that go through the non-circled interior grid points in the upper-left panel of Fig. 2.) However, if we weight each of these polynomials by one, then we need to subtract some polynomials as well.

(Why? Because the circled grid points in the upper-left panel of Fig. 2 have already shown up in multiple polynomials, each weighted by one.) The solution is to take the weighted sum not only of polynomials associated with $\mathcal{H}^{3,2}$ but also of those associated with $\mathcal{H}^{3,1}$ (i.e., the polynomials $p^{2,1,1}$, $p^{1,2,1}$, $p^{1,1,2}$) and the constant $p^{1,1,1}$. It turns out that by including these, it is possible to find the correct weights to interpolate all points in $\mathcal{H}^{3,2}$.

In general, the Smolyak function which interpolates on $\mathcal{H}^{d,\mu}$ is given by the weighted sum of low dimensional tensor products. Denote by $q=\max(d,\mu+1)$. At a point $x\in[-1,1]^d$, we then approximate $f(x)$ by

$$\hat{f}^{d,\mu}(x)=\sum_{q\le|\mathbf{i}|\le d+\mu}(-1)^{d+\mu-|\mathbf{i}|}\binom{d-1}{d+\mu-|\mathbf{i}|}p^{\mathbf{i}}(x) \tag{26}$$

The weights, $(-1)^{d+\mu-|\mathbf{i}|}\binom{d-1}{d+\mu-|\mathbf{i}|}$, are chosen to ensure that the weighted sum of polynomials which interpolate on subsets of $\mathcal{H}$ interpolates on the entire set.[6]

As with the construction of the Smolyak grid, it is instructive to work through an example to see what order of polynomials are used in the overall approximation (and which subset of polynomials from the set of complete polynomials is omitted). We constrain ourselves to the example of $d=3$ and $\mu=2$. In this case, recall that the set $\mathcal{H}^{3,2}$ consisted of 25 points, formed by the union of the sets

$$\begin{aligned}\mathcal{H}^{3,2}&=\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(3)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(3)}\\&\cup\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}\cup\mathcal{G}^{m(2)}\times\mathcal{G}^{m(1)}\times\mathcal{G}^{m(2)}\cup\mathcal{G}^{m(1)}\times\mathcal{G}^{m(2)}\times\mathcal{G}^{m(2)}\end{aligned}$$

In this case, $q=3$ and the overall interpolating function consists of the weighted sum of the following polynomials[7]: $|\mathbf{i}|=3$:

$$\begin{aligned}p_\theta^{1,1,1}&=\theta_{1,1,1}\\p_\theta^{2,1,1}&=\theta_{1,1,1}+\theta_{2,1,1}T_2(s_1)+\theta_{3,1,1}T_3(s_1)\end{aligned}$$

$|\mathbf{i}|=4$:

$$\begin{aligned}p_\theta^{1,2,1}&=\theta_{1,1,1}+\theta_{1,2,1}T_2(s_2)+\theta_{1,3,1}T_3(s_2)\\p_\theta^{1,1,2}&=\theta_{1,1,1}+\theta_{1,1,2}T_2(s_3)+\theta_{1,1,3}T_3(s_3)\\p_\theta^{3,1,1}&=\theta_{1,1,1}+\theta_{2,1,1}T_2(s_1)+\theta_{3,1,1}T_3(s_1)+\theta_{4,1,1}T_4(s_1)+\theta_{5,1,1}T_5(s_1)\\p_\theta^{1,3,1}&=\theta_{1,1,1}+\theta_{1,2,1}T_2(s_2)+\theta_{1,3,1}T_3(s_2)+\theta_{1,4,1}T_4(s_2)+\theta_{1,5,1}T_5(s_2)\\p_\theta^{1,1,3}&=\theta_{1,1,1}+\theta_{1,1,2}T_2(s_3)+\theta_{1,1,3}T_3(s_3)+\theta_{1,1,4}T_4(s_3)+\theta_{1,1,5}T_5(s_3)\end{aligned}$$

$$\begin{aligned}p_\theta^{2,2,1}&=\theta_{1,1,1}+\theta_{2,1,1}T_2(s_1)+\theta_{3,1,1}T_3(s_1)+\theta_{1,2,1}T_2(s_2)+\theta_{1,3,1}T_3(s_2)+\theta_{2,2,1}T_2(s_1)T_2(s_2)\\&\quad+\theta_{3,2,1}T_3(s_1)T_2(s_2)+\theta_{2,3,1}T_2(s_1)T_3(s_2)+\theta_{3,3,1}T_3(s_1)T_3(s_2)\end{aligned}$$

$|\mathbf{i}|=5$:

$$\begin{aligned}p_\theta^{2,1,2}&=\theta_{1,1,1}+\theta_{2,1,1}T_2(s_1)+\theta_{3,1,1}T_3(s_1)+\theta_{1,1,2}T_2(s_3)+\theta_{1,1,3}T_3(s_3)+\theta_{2,1,2}T_2(s_1)T_2(s_3)\\&\quad+\theta_{3,1,2}T_3(s_1)T_2(s_3)+\theta_{2,1,3}T_2(s_1)T_3(s_3)+\theta_{3,1,3}T_3(s_1)T_3(s_3)\end{aligned}$$

$$\begin{aligned}p_\theta^{1,2,2}&=\theta_{1,1,1}+\theta_{1,2,1}T_2(s_2)+\theta_{1,3,1}T_3(s_3)+\theta_{1,1,2}T_2(s_3)+\theta_{1,1,3}T_3(s_3)+\theta_{1,2,2}T_2(s_2)T_2(s_3)\\&\quad+\theta_{1,3,2}T_3(s_2)T_2(s_3)+\theta_{1,2,3}T_2(s_2)T_3(s_3)+\theta_{1,3,3}T_3(s_2)T_3(s_3)\end{aligned}$$

This tedious but conceptually straightforward enumeration shows that the number of unknown coefficients[8] exactly coincides with the number of 25 grid points in $\mathcal{H}^{3,2}$.

[6] See Barthelmann et al. (2000) for a proof that this procedure indeed works. For the example below with $d=3$ and $\mu=2$, the weights on the polynomials with $|\mathbf{i}|=3$ and $|\mathbf{i}|=5$ equal 1, whereas the weights for the polynomials with $|\mathbf{i}|=4$ equal -2.

[7] Note that $T_1(s)\equiv1$.

[8] In the interest of completeness, the coefficients are

$$\theta_{1,1,1},\theta_{2,1,1},\theta_{3,1,1},\theta_{4,1,1},\theta_{5,1,1}$$

$$\theta_{1,2,1},\theta_{1,3,1},\theta_{1,4,1},\theta_{1,5,1}$$

$$\theta_{1,1,2},\theta_{1,1,3},\theta_{1,1,4},\theta_{1,1,5}$$

$$\theta_{2,2,1},\theta_{3,2,1},\theta_{2,3,1},\theta_{3,3,1}$$

$$\theta_{2,1,2},\theta_{3,1,2},\theta_{2,1,3},\theta_{3,1,3}$$

$$\theta_{1,2,2},\theta_{1,2,3},\theta_{1,3,2},\theta_{1,3,3}$$

This example also shows that the set of polynomials whose weighted sum makes up $\hat{f}^{3,2}$ is significantly smaller than the complete set of polynomials of total degree 4 (all polynomials with $|\mathbf{i}|=5$ have total degree 4).[9] For example, the Chebychev polynomial $T_2(s_1)T_2(s_2)T_3(s_3)$ has total degree 4 and would thus be part of the complete set of polynomials of degree 4, but it is not part of the set of polynomials whose weighted sum form the approximating function $\hat{f}^{3,2}$. We want to stress that the choice of polynomials forming $\hat{f}^{3,2}$, and $\hat{f}^{d,\mu}$ in general, is by no means arbitrary: the Smolyak method simultaneously constructs the grid $\mathcal{H}^{d,\mu}$ and chooses the polynomials whose weighted sum fits exactly through these interpolation points.

2.2.4. *Some properties of Smolyak's method*

Without getting into mathematical details, we want to briefly discuss the advantages of Smolyak's method. The first obvious advantage is that the number of grid points does not grow exponentially with the dimension. It can be verified that the number of points in $\mathcal{H}^{d,\mu}$ is $1+2d$ for $\mu=1$, $1+4d+4d(d-1)/2$ for $\mu=2$ (the level of approximation used for the RBC model), and $1+8d+12d(d-1)/2+8d(d-1)(d-2)/6$ for $\mu=3$. The nestedness of the nodes, $\mathcal{G}^{m(i)}$, implies that the number of points in $\mathcal{H}^{d,\mu}$ grows only polynomially in d, taking μ as fixed.[10]

Note that the number of points does grow quickly in μ, but our application to the international RBC model will show that very good approximations are achieved even for $\mu=2$. Moreover, it can be shown that $\hat{f}^{d,\mu}$ exactly replicates any polynomial function with monomials of degree less than or equal to μ, in the sense that $\hat{f}^{d,\mu}$ will be identically equal to such a function if $\hat{f}^{d,\mu}$ interpolates it at the Smolyak points. This might seem a bit disappointing at first glance. After all, $\hat{f}^{d,\mu}$ is a polynomial of degree 2^{μ}. However, because the ratio between the degree of $\hat{f}$ and the degree of any polynomial that can be replicated by $\hat{f}$ is independent of the dimension d, the algorithm is regarded as *nearly optimal*. In general, better schemes are not known. Moreover, this makes clear that using a level-2 (i.e., $\mu=2$) Smolyak approximation is at least as good as (and often strictly better than) using any second-degree polynomial approximation.

2.3. *Integration*

Once we approximate the unknown policy functions by Smolyak polynomials, we require that the unknown coefficients ensure that Eqs. (1)–(4) hold exactly at the collocation points. To solve for the coefficients, we obviously need a way to evaluate the integral in Eq. (1). Since u_c, f_k and the probability density function for the exogenous state variables are not polynomials, we need to approximate the integral numerically. It is well known that for integration in relatively low dimensions (say around 10–15), if the integrand is sufficiently smooth, routines based on interpolatory cubature rules turn out to deliver much more accurate results than Monte Carlo or quasi-Monte Carlo methods (see Cools, 2002; Schürer, 2003). Since Judd's (1998) textbook contains an excellent description of these various rules, we do not discuss them in detail here. In our computations, we use a degree 5 rule for an integrand on an unbounded range weighted by a standard normal. Note that we can easily rewrite the integral in Eq. (13) as a function of the underlying innovations, which are standard normal. Thus, the integral is of the form $\int_{\mathbb{R}^d} f(x)e^{-\sum_{i=1}^{d} x_i^2}\,dx$ and can be approximated by this degree 5 rule.[11]

2.4. *Finding the unknown coefficients*

Using Smolyak's polynomials to approximate the policy functions and using the cubature rule to approximate the integrals now allows us to consider a finite system of non-linear equations whose solutions are the unknown coefficients θ of the approximate policy functions. In principle, this system can be solved directly by modern non-linear equation solvers which are variations of Newton's method. However, the system to be solved for the unknown θ is large as the dimensionality $d=2N$ of the problem increases, consisting of $3N$ times $\#\mathcal{H}^{d,\mu}$ equations in as many unknowns. Therefore we chose to solve for the coefficients by a "time-iteration" algorithm. In each iteration m a system of $3N$ equations in as many unknowns $y \in \Re^{3N}$ has to be solved, for each of the $\#\mathcal{H}^{d,\mu}$ points $x \in \mathcal{H}^{d,\mu}$ in the grid. Once the (x,y) pairs are determined, the updated coefficients θ_m then follow straightforwardly from Eq. (25). Thus we trade off solving non-linear systems of much smaller size by the necessity to having to solve the systems many times (as opposed to only once). For problems of the dimensionality considered in this paper, the time-iteration algorithm converged reliably for all parameterizations.

[9] Note that, in our notation, T_1 is the Chebychev polynomial of order 0 (i.e., a constant), and thus T_5 is a one-dimensional Chebychev polynomial of order 4.

[10] As pointed out above, the number of grid points and the number of different polynomials used to interpolate between the grid points exactly coincide. Thus for $\mu=2$, the number of distinct polynomials is quadratic in the dimensionality d of the problem. The associated complete set of polynomials of total degree 4 is quartic in d, and thus contains many more polynomials, especially as the dimensionality of the problem increases.

[11] In order to verify the quality of approximation, we compared the results with a simple Monte-Carlo method that uses 10,000 draws. In all cases, the differences were on the order of at most 10^{-5}.

First, we guess policy functions $K_0^j(s),L_0^j(s)$ and $C_0^j(s)$ for all $j \in [1\dots N]$. For a given iteration $m-1$ and associated policy functions $\{K_{m-1}^j(s),L_{m-1}^j(s),C_{m-1}^j(s)\}_{j=1\dots N}$ the iteration m policy functions are, for all $s=(k,a)$, defined by the $3N$ equations[12]

$$\tau^j u_c^j(C_m^j(s),L_m^j(s)) = \frac{\beta \int \tau^j u_c^j(C_{m-1}^j(s'),L_{m-1}^j(s')) \left[\begin{array}{c} 1+a'jf_k^j(K_m^j(s),L_{m-1}^j(s')) \\ +\dfrac{\phi}{2}\dfrac{(K_{m-1}^j(s')-K_m^j(s))(K_{m-1}^j(s')+K_m^j(s))}{K_m^j(s)^2} \end{array} \right] g_a(a')da'}{1+\dfrac{\phi(K_m^j(s)-k^j)}{k^j}} \quad \forall j \tag{27}$$

and

$$\tau^j u_c^j(C_m^j(s),L_m^j(s)) = \tau^i u_c^i(C_m^i(s),L_m^i(s)) \quad \forall i,j \tag{28}$$

$$\frac{u_l^j(C_m^j(s),L_m^j(s))}{u_c^j(C_m^j(s),L_m^j(s))} = -a^j f_l^j(k^j,L_m^j(s)) \quad \forall j \tag{29}$$

$$C_m(s)+\sum_{j=1}^{N}\left(K_m^j(s)+\frac{\phi}{2}\frac{(K_m^j(s)-k^j)^2}{k^j}\right) = Y_m(s)+K \tag{30}$$

Once the policy functions are determined for all points *s on the grid,* Eq. (25) then determines the coefficients θ that define the policy functions in the m-th step off the grid.

In terms of running-times, this method is obviously not comparable to Newton's method. However, it has the advantage that it can easily handle very large systems. Moreover, it has a nice economic interpretation in that it can be viewed as approximating the infinite horizon economy by an economy with a large finite horizon.

3. Implementation details

In this section, we describe some key details of our implementation of Smolyak's method to solve the multi-country RBC model considered in this project. These details provide necessary information for interpreting the speed and accuracy results described in Kollmann et al. (this issue-b).

First, our time-iteration procedure requires an initial guess for the policy functions. For this initialization, we simply take a (log-)linearized solution of the model around its non-stochastic steady state. Linearized solution routines are easily available (e.g. the first-order perturbation method of Kollmann et al., this issue-a) and, for the model specifications we consider, produce an initial guess in less than a second.

A second implementation detail involves choosing bounds for the state variables. Recall that, at least for theoretical results, Smolyak's method is defined over a closed hypercube. As a practical matter, the interpolation algorithm applies outside the hypercube as well, although accuracy may suffer at such points. A trade-off exists: with tighter bounds, the accuracy at points inside the bounds will be higher but, because the state variables will be more likely to run out-of-bounds, overall accuracy could decline. For this project, we simply set the capital bounds to (roughly) 20% above and below the steady-state capital level $[\underline{k}^j,\overline{k}^j]=[0.8,1.2]$, while the productivity bounds were set to be $[\ln(\underline{a}^j),\ln(\overline{a}^j)]=[-0.8\sigma/(1-\rho),0.8\sigma/(1-\rho)]$. We did not systematically investigate if these particular bounds were optimal but did find that changing the bounds made only small differences in the accuracy measures.

A final important implementation detail involves deciding which of the policy functions to explicitly parameterize. In the previous section, we described our solution method as approximating all $3N$ policy functions by Smolyak polynomials, but given the model structure, we actually take a slightly more flexible approach. Specifically, note that, given the state variables and consumption of one country, say $C_m^1(s)$, Eqs. (28) and (29) provide $2N-1$ conditions for determining the remaining consumption decisions and labor supplies of all countries. Thus, we can specify each of these variables in a "non-parametric" form; given $C_m^1(s)$, the variables are simply equal to whatever values satisfy Eqs. (28) and (29) (as determined by a non-linear equation solver, namely Fortran subroutine HYBRD, with fairly high accuracy).[13] As the state s is varied, this approach thus traces out flexible policy functions for $\{C^j(s)\}_{j=2}^N,\{L^j(s)\}_{j=1}^N$. The other $N+1$ policy functions, $\{K'j(s)\}_{j=1}^N$ and $C^1(s)$, *are* explicitly parameterized as weighted functions of low-order polynomials according to the Smolyak interpolation algorithm.

When deciding which policy functions to parameterize explicitly, there is an intuitive trade-off. On the one hand, the use of a very flexible, non-parametric form for some policy functions means that some equilibrium conditions will always hold with very high accuracy. Indeed, for our method, this is true for Eqs. (14) and (15), or equivalently, Eqs. (28) and (29). On the other hand, specifying the functions in a non-parametric form requires using a non-linear equation solver every time one needs to know the value of the function, which can be very costly. As noted above, we balance this trade-off by

[12] Again, we abuse notation and let $s'=(K_m'(s),a')$.

[13] Similarly, Eqs. (28) and (29) determine $\{C_{m-1}^j(s')\}_{j=2}^N$, $\{L_{m-1}^j(s')\}_{j=1}^N$ given $C_{m-1}^1(s')$.

explicitly parameterizing all capital decision rules and the consumption of one country, which allows for the remaining decision rules to either be expressed in closed form or be solved for using small systems of non-linear equations. Alternatively, we could have specified the consumption and labor supply of *all* countries non-parametrically. Doing so would improve the accuracy of our solution method (by eliminating the errors in the world resource constraint, which is the equilibrium condition that always produces the maximum error for our method) but at a computational cost. We judged the computational cost to be too large, but other project participants have found clever ways to reduce these computational costs. As mentioned in the comparison paper by Kollmann et al. (this issue-b), the ‘‘iteration-on-allocation’’ approach of Maliar et al. (this issue) could be merged with our solution method, thus allowing us to explicitly parameterize fewer policy functions and improve accuracy, while possibly even reducing the time it takes to find a solution.

4. Conclusion

In this paper we described and used a projection method based on Smolyak's algorithm to compute globally accurate solutions to models featuring a sizeable number of continuous-valued state variables. The method was applied to solving a wide variety of international real business cycle model specifications. As documented by Kollmann et al. (this issue-b), the method delivers high accuracy and reasonable running times and appears to be a viable solution method for use in a wide class of economic models. One goal of this paper has been to make this solution method more accessible to other economists. To this end, we provided both a general description of the method and a discussion of some of the practical details of implementing it. We have made available corresponding computer code for the Smolyak method,[14] which should enable any researcher armed with a set of optimality conditions to approximate the corresponding model solution without having to undertake the fixed costs of constructing the Smolyak grid and interpolation algorithm.

Acknowledgments

We thank Ken Judd for clarifying discussions about the scope and focus of this paper. We also wish to thanks seminar participants at the 2006 Cleveland FED conference on international macroeconomics, the 2007 Heterogeneity and Macrodynamics conference in Paris, and the 2009 conference on computational economics in Zurich, as well as Wouter Denhaan, Karl Schmedders, Paul Pichler, Michael Reiter and an anonymous referee for helpful comments. Krueger and Kubler gratefully acknowledge financial support under NSF Grant SES-0004376. The views expressed in this paper are solely our own and should not be interpreted as reflecting those of the Board of Governors or the staff of the Federal Reserve System.

References

Backus, D., Kehoe, P., Kydland, F., 1992. International real business cycles. Journal of Political Economy 100, 745–775.

Backus, D., Kehoe, P., Kydland, F., 1995. International business cycles, theory and evidence. In: Cooley, T. (Ed.), Frontiers of Business Cycle Research. Princeton University Press, Princeton, pp. 331–356.

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

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

Cools, R., 2002. Advances in multidimensional integration. Journal of Computational and Applied Mathematics 149, 1–12.

Denhaan, W., Judd, K., Juillard, M., this issue. Computational suite of models with heterogenous agents: multi-country real business cycle models. Journal of Economic Dynamics and Control, doi: 10.1016/j.jedc.2010.09.010.

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

Juillard, M., Villemont, S., this issue. Multi-country real business cycle models: accuracy tests and testing bench. Journal of Economic Dynamics and Control, doi:10.1016/j.jedc.2010.09.011.

Kollmann, R., Kim, S., Kim, J., this issue-a. Solving the multi-country real business cycle model using a perturbation method. Journal of Economic Dynamics and Control, doi:10.1016/j.jedc.2010.09.012.

Kollmann, R., Maliar, S., Malin, B., Pichler, P., this issue-b. Comparison of solutions to the multi-country real business cycle model. Journal of Economic Dynamics and Control, doi:10.1016/j.jedc.2010.09.013.

Krueger, D., Kubler, F., 2004. Computing equilibrium in OLG models with stochastic production. Journal of Economic Dynamics and Control 28, 1411–1436.

Maliar, S., Maliar, L., Judd, K., this issue. Solving the multi-country real business cycle model using ergodic set methods. Journal of Economic Dynamics and Control, doi:10.1016/j.jedc.2010.09.014.

Pichler, P., this issue. Solving the multi-country real business cycle model using a monomial-rule Galerkin method. Journal of Economic Dynamics and Control, doi:10.1016/j.jedc.2010.09.009.

Rivlin, T.J., 1969. An Introduction to the Approximation of Functions. Dover Publications, Mineola, NY.

Schürer, R., 2003. A comparison between quasi-Monte Carlo and cubature rule based methods for solving high-dimensional integration problems. Mathematics and Computers in Simulation 62, 509–517.

Smolyak, S., 1963. Quadrature and interpolation formulas for tensor products of certain classes of functions. Soviet Mathematics, Doklady 4, 240–243.

[14] The programs are available at http://www.econ.upenn.edu/~dkrueger/research.php.