Solving the multi-country real business cycle model using a monomial rule galerkin method

Paul Pichler

www.elsevier.com/locate/jedc

Solving the multi-country real business cycle model using a monomial rule galerkin method

Paul Pichler

PII: S0165-1889(10)00213-7
DOI: doi:10.1016/j.jedc.2010.09.009
Reference: DYNCON 2483

To appear in: *Journal of Economic Dynamics & Control*

Received date: 10 March 2010
Accepted date: 7 July 2010

Cite this article as: Paul Pichler, Solving the multi-country real business cycle model using a monomial rule galerkin method, *Journal of Economic Dynamics & Control*, doi:10.1016/j.jedc.2010.09.009

Solving the Multi-Country Real Business Cycle Model Using a Monomial Rule Galerkin Method

Paul Pichler
Department of Economics
University of Vienna
Hohenstaufengasse 9
A-1010 Vienna
Austria

Abstract

I propose a Galerkin projection method for solving dynamic economic models with many state variables. This method employs non-product monomial integration formulas for the computation of weighted residuals, and its computational cost therefore increases only polynomially in the model's dimensionality. I illustrate the practical implementation of the proposed algorithm by solving several specifications of the multi-country Real Business Cycle model described in Den Haan et al. [2010. Computational Suite of Models with Heterogeneous Agents: Multi-country Real Business Cycle Models, *Journal of Economic Dynamics and Control*, this issue], and briefly discuss two possible routes for further improving its numerical accuracy.

Keywords: Galerkin method, weighted residuals, monomial cubature rules, curse of dimensionality, Multi-Country Real Business Cycle Model

JEL classification: C63

1. Introduction

Projection methods (or weighted residual methods) have a long tradition in the natural sciences to solve systems of functional equations. In economics, they have first been introduced by Judd (1992) as a numerical solution method for aggregate growth models. Ever since, projection methods have become the arguably most popular approach for obtaining *globally accurate* numerical solutions to dynamic equilibrium models that lack analytical tractability.[1] Contributing largely to their success is the fact that projection methods are extremely flexible and often perform better than alternative strategies when the researcher seeks highly accurate numerical solutions (Christiano and Fisher, 2000; Aruoba et al., 2006).

The projection approach to solving stochastic recursive economic models proceeds in the following steps. First, the model's true equilibrium decision rules are replaced by parametric approximating functions which typically take the form of polynomials or spline functions. Second, a numerical integration method is used to approximate the conditional expectations in the model's intertemporal equilibrium conditions. Third, a set of grid points in the state space is selected, and the approximation error (*residual*) in the model's equilibrium conditions at each grid point is computed. Finally, the unknown parameters in the approximating functions are determined by minimizing the residuals subject to some loss criterion. According to the specific choice of loss criterion, projection methods fall into three categories: *collocation*, *least squares*, and *Galerkin* methods. Collocation selects as many grid points as there are unknown parameters in the approximating functions, and determines these parameters by equating the residual at each grid point to zero. The least squares and Galerkin approaches, in turn, use larger sets of grid points, i.e., more grid points than parameters. The parameters are pinned down by minimizing the sum of squared residuals, in the case of least squares, or by equating weighted averages of the residuals to zero, in the case of Galerkin. Computing these weighted averages amounts to solving a numerical integration problem, which is however of a different kind than the problem of approximating conditional expectations. A key step in the Galerkin algorithm is to choose the numerical integration methods to be used for both these tasks.

Computational economics textbooks (e.g., Judd, 1998; Marimon and Scott, 1999; Miranda and Fackler, 2002) present the Galerkin method resorting to Gaussian integration rules. In particular, Gauss-Chebyshev or Gauss-Legendre integration is used for computing weighted averages over residuals, whereas Gauss-Hermite integration is used for approximating conditional expectations. In low dimensional applications, Gaussian rules are attractive: they are relatively easy to code, fast, and accurate. In particular, univariate Gaussian quadrature methods can easily be extended to multidimensional settings using the tensor-product operator. While being conceptually straightforward, this approach is computationally costly as the number of grid points that have to be considered grows exponentially in the model's dimensionality (i.e., in the number of state variables). As a consequence, so does the computational cost associated with finding the unknown coefficients in the model's approximating decision rules, i.e., Galerkin methods using Gaussian product rules suf-

[1] See Reiter (2009), Niemann et al. (2010), Den Haan et al. (2010), Den Haan and Rendahl (2010), or Maliar et al. (2010) for just a few recent examples of their application in economics.

fer from the *curse of dimensionality*.[2] This property renders the "conventional" textbook Galerkin method infeasible for models with more than a few state variables.[3]

In the present paper I describe a Galerkin method that avoids the curse of dimensionality and therefore can be applied to models with many state variables. This method resorts to *non-product monomial cubature rules* for solving the numerical multi-dimensional integration problems arising in (i) the approximation of conditional expectations and (ii) the computation of weighted residuals. The computational cost of the described Galerkin method increases only polynomially (rather than exponentially) in the dimension of the model to be solved, and the method thus remains feasible even for relatively high-dimensional problems. I illustrate this key property by solving the multi-country Real Business Cycle model of the comparison project described in Den Haan et al. (2010). The model specifications solved involve up to twenty continuous-valued state variables.

While the present paper illustrates the methodology by means of a Real Business Cycle model, there are many other applications where the described method is of potential value. In particular, these include applications where resorting to local approximation methods may not be easily possible. For example, economic models featuring occasionally binding inequality constraints such as a zero-floor on nominal interest rates, capacity constraints, or investment irreversibility fall into this category. Moreover, local approximations around a non-stochastic steady state are typically infeasible for dynamic models with strategic interaction. In these models, optimal behavior is characterized by *generalized Euler equations* that involve the derivatives of some equilibrium decision rules, and thus it is impossible to compute the steady state independent of these rules (Ortigueira, 2006; Martin 2009). *Global* approximation methods, such as the Galerkin method described in the present paper, are clearly preferable to *local* methods in such applications.[4]

Finally, it is worth emphasizing that the present paper is not the first to suggest the use of non-product monomial integration formulas in the context of rational expectations models. The idea to employ such rules for approximating conditional expectations dates back to at least Den Haan et al. (2004). Moreover, Heer and Maussner (2006) have proposed a Galerkin approach based on a non-product monomial cubature rule by Mustard et al. (1963). Although their approach reduces the computational burden compared to the standard approach, it still suffers from the curse of dimensionality (i.e., the computational cost still increases exponentially in the state space dimension) and, consequently, the approach cannot be applied to relatively large models. Moreover, the method proposed by Heer and Maussner requires to evaluate the model's residual function at the boundaries of the relevant state space, which, as they point out, leads to numerical instabilities.

[2]The term *curse of dimensionality* was coined by Bellman (1961) and refers, in its most general interpretation, to the *exponential* increase in hypervolume when adding extra dimensions to a mathematical space. In economics, it is used mostly in the context of approximating functions (or integrands) arising in recursive dynamic models. Specifically, within these applications, the curse of dimensionality refers to the exponential rise in the computational cost associated with numerically approximating a certain function when the number of arguments of this function increases; see Doraszelski and Judd (2005) or Winschel and Kraetzig (2010).

[3]This is true also for the "conventional" collocation and least squares projection methods. These methods, too, employ tensor-grids of points in the state space at which the residuals have to be computed, and thus their computational cost increases exponentially in the model dimension.

[4]Other global solution methods that are of potential value within the outlined applications are described in the contributions to this special issue by Malin, Krueger, and Kubler (2010) and Maliar, Maliar, and Judd (2010).

The Galerkin method described in the present paper thus improves upon their contribution in two important ways. First, it remains applicable even for large models. Second, the solution algorithm is numerically stable and thus easy to use.

The remainder of this paper is organized as follows. Section 2 provides a brief introduction to solving multi-dimensional integration problems using monomial cubature rules. Section 3 presents the general description of a Galerkin weighted residual method employing non-product monomial rules. Section 4 illustrates the implementation of the Galerkin method in practical applications using the multi-country Real Business Cycle model by Den Haan et al. (2010). Finally, Section 5 briefly discusses two possible routes for further improving the numerical accuracy of the algorithm.

2. Monomial cubature rules

The key property of the proposed Galerkin method is its use of efficient multi-dimensional integration techniques, i.e., non-product *monomial cubature rules*, for computing weighted residuals.[5] Specifically, these rules are used for (i) the Galerkin integration step, i.e., computing weighted averages over residuals and (ii) approximating the conditional expectations in the model's dynamic equilibrium conditions. In this section I briefly overview monomial rules that can be applied for these two tasks. Most of the presented material is taken from Stroud (1971), Phillips (1980), and Judd (1998), and the interested reader is referred to these references for a rigorous treatment of monomial rules.

Consider the general integration problem

$$I(f) = \int_S f(x)dx \tag{1}$$

where $S \subseteq \mathbb{R}^n$ is a region in the n-dimensional space, and f is a real-valued function on S. Let $\hat{I}(f)$ be a numerical approximation to $I(f)$ of the form

$$\hat{I}(f) = \sum_i \omega_i f(x^{(i)}), \tag{2}$$

with real weights ω_i and grid points $x^{(i)} \in \mathbb{R}^n$.

Definition 1. *A monomial cubature rule of degree d is a numerical approximation $\hat{I}(f)$ that satisfies $\hat{I}(f) = I(f)$ whenever the function f is a monomial of degree not greater than d.*

A *monomial* of degree d is a function $x_1^{i_1}...x_n^{i_n}$ with $i_j \geq 0$ and $i_1 + ... + i_n = d$. Notice that a *polynomial* of degree d is a linear combination of monomials, with at least one monomial being of degree d and all others of degree not greater than d. It therefore follows trivially that a degree d monomial rule is exact for all *polynomials* of degree not greater than d. The parameter d is often referred to as the rule's *polynomial degree (of exactness)*.

[5] In the mathematics literature, the term *cubature* is typically used for multivariate integration whereas the term *quadrature* refers to univariate integration; see Krommer and Ueberhuber (1998).

In the mathematics literature there exists a large variety of monomial cubature rules for many common regions of integration, including the hypercube, the hypersphere, and the entire n-dimensional space with various weight functions. Stroud (1971) and Cools (2003) provide an encyclopedic treatment of several hundred rules for dozens of different regions. In the following, monomial rules for the hypercube and the n-dimensional space with weight function e^{-x^2} will be briefly examined. The first type of rules applies to the Galerkin integration step, while the second type of rules can be used to approximate conditional expectations.

2.1. Rules for the hypercube

Consider integration over the n-dimensional unit hypercube,

$$I(f) = \int_{[-1,1]^n} f(x)dx. \tag{3}$$

The choice of region $S = C_n = [-1,1]^n$ is not overly restrictive, as many common regions of integration can easily be transformed to C_n by change of variables techniques.[6] In particular, any n-dimensional hypercube $[a_1,b_1]\times[a_2,b_2]\times\cdots\times[a_n,b_n]$ can be transformed into C_n using the linear change of variables $y_j = 2(x_j - a_j)/(b_j - a_j) - 1$ for $j = 1,...,n$.

The arguably most simple and popular rule for the hypercube is the *Gauss-Chebyshev product rule*. For the integration problem (3), a Gauss-Chebyshev rule using m points in each dimension establishes an approximation of polynomial degree $2m-1$,

$$\hat{I}(f) = \left(\frac{\pi}{m}\right)^n \sum_{i_1=1}^{m}\sum_{i_2=1}^{m}\cdots\sum_{i_n=1}^{m} f(\hat{x}_{i_1},\hat{x}_{i_2},...,\hat{x}_{i_n})W(\hat{x}_{i_1},\hat{x}_{i_2},...,\hat{x}_{i_n}), \tag{4}$$

with $W(\hat{x}_{i_1},\hat{x}_{i_2},...,\hat{x}_{i_n}) = \prod_{j=1}^{n}(1-\hat{x}_{i_j}^2)^{\frac{1}{2}}$, and where the grid points $\hat{x}_k = \cos\left(\frac{2k-1}{2m}\pi\right)$, $k = 1,..,m$, are the roots of the univariate Chebyshev polynomial of order m. An attractive feature of the rule (4) is that a researcher can easily target the degree of polynomial exactness by choosing the number of points m used in each dimension. Its main weakness is that, due to its product form, the Gauss-Chebyshev rule suffers from the *curse of dimensionality*: for any given m, the number of grid points employed, m^n, grows exponentially in the dimension of the integration problem. As a consequence of this property, the rule is computationally infeasible for high-dimensional problems.

Certain *non-product monomial rules* avoid the curse of dimensionality by operating only on a sparse set of grid points rather than a full tensor-grid. In general, the construction of non-product rules can be difficult, because a degree d rule in n dimensions must exactly integrate $\binom{n+d}{n}$ monomials and thus the construction of such a rule typically involves solving a large system of non-linear equations. For symmetric regions such as the hypercube, however, some monomial rules can be derived fairly easily as the following theorem is available (Hammer and Stroud, 1957, p.63):

Theorem 1. *A fully symmetric rule $\hat{I}(f)$ applied to a fully symmetric n-dimensional region S is of degree d if and only if it is exact for all monomials of degree $\leq d$ of the form $x_1^{2i_1}x_2^{2i_2}\ldots x_n^{2i_n}$ with $i_1 \geq i_2 \geq \cdots \geq i_n \geq 0$.*

[6] See chapter 7.2 of Krommer and Ueberhuber (1998) or chapter 7 of Judd (1998) on the application of *change of variables* techniques in the context of numerical integration.

Theorem 1 is extremely helpful in constructing monomial rules, as it substantially reduces the number of monomials one has to consider. For example, a degree $d = 5$ rule in $n = 10$ dimensions has to exactly integrate all $\binom{n+d}{n} = \binom{15}{10} = 3003$ monomials of degree ≤ 5. By Theorem 1, this is automatically guaranteed if the rule is fully symmetric and exactly integrates the four monomials 1, x_1^2, $x_1^2 x_2^2$, and x_1^4.

Note that *fully symmetric* monomial rules are rules that employ grid points which are all elements of fully symmetric sets of points in S, with the same weight given to every point in a certain set.

Definition 2. *The fully symmetric set of* x, $(x_1, ..., x_n)_{FS}$, *is the set of all points* $(\pm x_{i_1}, \pm x_{i_2}, ..., \pm x_{i_n})$ *where* $(i_1, i_2, ..., i_n)$ *is any permutation of* $(1, 2, ..., n)$.

A region S, in turn, is fully symmetric if $(x_1, ..., x_n) \in S$ implies $(x_1, ..., x_n)_{FS} \subseteq S$. In what follows, I will use the shorter notation $\mathcal{G}_n^1(\alpha_1) = (\alpha_1, 0, 0, 0, ..., 0)_{FS}$, $\mathcal{G}_n^2(\alpha_1, \alpha_2) = (\alpha_1, \alpha_1, 0, 0, ..., 0)_{FS}$, $\mathcal{G}_n^3(\alpha_1, \alpha_2, \alpha_3) = (\alpha_1, \alpha_2, \alpha_3, 0, ..., 0)_{FS}$, etc. to denote fully symmetric sets.[7]

To emphasize how useful Theorem 1 is for constructing monomial rules, let me briefly sketch the derivation of one such rule for the hypercube. This rule, which is originally due to Hammer and Stroud (1958), will play a prominent role in the Galerkin weighted residual method to be developed in Section 3. Consider an approximation of the form

$$\hat{I}(f) = \omega_0 f(0) + \omega_1 \sum_{x \in \mathcal{G}_n^1(r)} f(x) + \omega_2 \sum_{x \in \mathcal{G}_n^2(r,r)} f(x). \tag{5}$$

Note that the summation is over $2n$ and $2n(n-1)$ grid points, respectively, and that, according to Theorem 1, the rule (5) is exact of polynomial degree $d = 5$ if it exactly integrates the four monomials 1, x_1^2, $x_1^2 x_2^2$, and x_1^4. It is straightforward to derive $I(1) = 2^n$, $I(x_1^2) = 2^n/3$, $I(x_1^4) = 2^n/5$, and $I(x_1^2 x_2^2) = 2^n/9$, such that the latter requirement translates into the following system of equations:

$$\begin{aligned}
2^n &= \omega_0 + 2n\omega_1 + 2n(n-1)\omega_2, && (6)\\
2^n/3 &= 2r^2\omega_1 + 4(n-1)r^2\omega_2, && (7)\\
2^n/5 &= 2r^4\omega_1 + 4(n-1)r^4\omega_2, && (8)\\
2^n/9 &= 4r^4\omega_2. && (9)
\end{aligned}$$

The solution of this system is $r = \sqrt{3/5}$, $\omega_0 = 2^n \frac{(25n^2 - 115n + 162)}{162}$, $\omega_1 = 2^n \frac{(70 - 25n)}{162}$, and $\omega_2 = 2^n \frac{25}{324}$. The following proposition summarizes these findings:

Proposition 1. $[C_n d5]$ *The numerical integration formula*

$$\hat{I}(f) = \omega_0 f(0) + \omega_1 \sum_{x \in \mathcal{G}_n^1(r)} f(x) + \omega_2 \sum_{x \in \mathcal{G}_n^2(r,r)} f(x),$$

with $r = \sqrt{3/5}$, $\omega_0 = 2^n \frac{(25n^2 - 115n + 162)}{162}$, $\omega_1 = 2^n \frac{(70 - 25n)}{162}$, *and* $\omega_2 = 2^n \frac{25}{324}$ *is a degree* 5 *monomial rule for* C_n *using* $2n^2 + 1$ *grid points.*

[7] For example, the set $\mathcal{G}_2^1(1) = (1, 0)_{FS}$ is comprised of the four points $(1, 0)$, $(-1, 0)$, $(0, 1)$ and $(0, -1)$; the set $\mathcal{G}_2^2(1, 2)$ is comprised of the eight points $(1, 2)$, $(-1, 2)$, $(1, -2)$, $(-1, -2)$, $(2, 1)$, $(-2, 1)$, $(2, -1)$, and $(-2, -1)$, respectively.

In a completely analogous way, one can construct other monomial rules for the hypercube. One such rule is the degree 7 rule due to Stenger (1971). The coefficients of this rule, however, are the solution to a non-linear system of equation, and thus cannot be obtained in closed form (i.e., as functions of the dimension n). Stenger (1971) provides the coefficients u, v, ω_0, ω_1, ω_2, ω_3, ω_4, and ω_5 in tabulated form for $2 \leq n \leq 20$.

Proposition 2. $[C_n d7]$ *The numerical integration formula*

$$\hat{I}(f) = \omega_0 f(0) + \omega_1 \sum_{x \in \mathcal{G}_n^1(u)} f(x) + \omega_2 \sum_{x \in \mathcal{G}_n^1(v)} f(x) + \omega_3 \sum_{x \in \mathcal{G}_n^2(u,u)} f(x) + \omega_4 \sum_{x \in \mathcal{G}_n^2(v,v)} f(x) + \omega_5 \sum_{x \in \mathcal{G}_n^3(u,u,u)} f(x),$$

with u, v, ω_0, ω_1, ω_2, ω_3, ω_4, and ω_5 such that $\hat{I}(f) = I(f)$ for $f \in \{1, x_1^2, x_1^4, x_1^6, x_1^2 x_2^2, x_1^4 x_2^2, x_1^2 x_2^2 x_3^2\}$, is a degree 7 monomial rule for C_n using $(4n^3 + 8n + 3)/3$ grid points.

Finally, notice that the number of grid points used by $[C_n d5]$ and $[C_n d7]$, respectively, grows only polynomially in the dimension n. Thus, both rules are not subject to the curse of dimensionality and, for large n, deliver a better trade-off between speed and accuracy (as measured by the polynomial degree) than Gaussian product rules.

2.2. Rules for the n-dimensional space with weight function e^{-x^2}

Integration over the entire n-dimensional Euclidian space with weight function e^{-x^2}, $E_n^{r^2}$, is encountered frequently in economic problems: it arises in the computation of the expected value of a function of normally distributed random variables. Specifically, if y is an n-dimensional vector of i.i.d. standard normal random variables and g is a non-linear function, then

$$\mathbb{E}(g(y_1, y_2, ..., y_n)) = (2\pi)^{-n/2} \int_{\mathbb{R}^n} g(y_1, y_2, ..., y_n) e^{-\sum_{j=1}^n \frac{y_j^2}{2}} dy \tag{10}$$

where $\mathbb{E}$ is the expectations operator. Using the change of variables $x_i = y_i/\sqrt{2}$, expression (10) can be written as

$$\mathbb{E}(g(y_1, y_2, ..., y_n)) = \pi^{-n/2} \int_{\mathbb{R}^n} \tilde{g}(x_1, x_2, ..., x_n) e^{-\sum_{j=1}^n x_j^2} dx = \pi^{-n/2} I(\tilde{g} \cdot \tilde{w}), \tag{11}$$

where $\tilde{g}(x_1, x_2, ..., x_n) = g(\sqrt{2}x_1, \sqrt{2}x_2, ..., \sqrt{2}x_n)$ and $\tilde{w}(x_1, x_2, ..., x_n) = e^{-\sum_{j=1}^n x_j^2} dx$.

If the dimension of integration is low, one possibility to approximate the integral on the right hand-side of (11) is by resorting to *Gauss-Hermite product rules*. A Gauss-Hermite rule with m points in each dimension gives a degree $2m - 1$ approximation of the form,

$$\hat{I}(\tilde{g} \cdot \tilde{w}) = \sum_{i_1=1}^m \sum_{i_2=1}^m \cdots \sum_{i_n=1}^m \omega_{i_1} \omega_{i_2} \ldots \omega_{i_n} \tilde{g}(\tilde{x}_{i_1}, \tilde{x}_{i_2}, ..., \tilde{x}_{i_n}), \tag{12}$$

where $\tilde{x}_j$, $j = 1, ..., m$, are the roots of the degree m Hermite polynomial H_m, and $\omega_j = \frac{2^{m-1} m! \sqrt{\pi}}{m^2 [H_{m-1}(\tilde{x}_j)]^2}$ is the weight associated with grid point $\tilde{x}_j$.

If the dimension of integration is relatively large, the curse of dimensionality impedes the use of Gauss-Hermite integration in practical applications. In such applications one can again resort to non-product monomial rules. Stroud and Secrest (1963) derive the following two rules (covered also in chapter 7 of Judd, 1998):

Proposition 3. $[E_n^{r^2} d3]$ *The numerical integration formula*

$$\hat{I}(\tilde{g} \cdot \tilde{w}) = \pi^{n/2} \frac{1}{2n} \sum_{x \in \mathcal{G}_n^1(r)} \tilde{g}(x),$$

with $r = \sqrt{n/2}$ *is a degree* 3 *monomial rule for* $E_n^{r^2}$ *using* $2n$ *grid points.*

Proposition 4. $[E_n^{r^2} d5]$ *The numerical integration formula*

$$\hat{I}(\tilde{g} \cdot \tilde{w}) = \omega_0 \tilde{g}(0) + \omega_1 \sum_{x \in \mathcal{G}_n^1(r)} \tilde{g}(x) + \omega_2 \sum_{x \in \mathcal{G}_n^2(s,s)} \tilde{g}(x), \tag{13}$$

with $r = \sqrt{\frac{n+2}{2}}$, $s = \sqrt{\frac{n+2}{4}}$, $\omega_0 = \pi^{n/2} \frac{2}{n+2}$, $\omega_1 = \pi^{n/2} \frac{4-n}{2(n+2)^2}$, *and* $\omega_2 = \pi^{n/2} \frac{1}{(n+2)^2}$ *is a degree* 5 *monomial rule for* $E_n^{r^2}$ *using* $2n^2 + 1$ *grid points.*

The number of grid points used by the degree $d = 3$ rule, $[E_n^{r^2} d3]$, grows only linearly in n. This is an attractive feature, as it implies that the rule is applicable even in very high-dimensional problems. The rule $[E_n^{r^2} d5]$, on the other hand, is of higher polynomial degree ($d = 5$) but the number of grid points it uses grows faster (quadratically) in n.

3. A non-product monomial-rule Galerkin method

Having discussed non-product monomial rules, I now turn to the description of a Galerkin projection method that employs these rules. The described method applies to recursive dynamic models that can be represented as

$$\mathcal{R}(f) = 0, \tag{14}$$

where $\mathcal{R} : B_1 \to B_2$ is a non-linear operator, B_1 and B_2 are function spaces, and $f : X \to D$ is a vector-valued decision rule that maps the state space $X \subseteq \mathbb{R}^{n_x}$ into the decision space $D \subseteq \mathbb{R}^{n_d}$. The proposed Galerkin method produces a global polynomial approximation $\hat{f} : X \to D$ to the true decision rule, such that $\mathcal{R}(\hat{f}) \approx 0$ over the relevant state space.

A brief outline of the algorithm is as follows:

1. Select approximating functions $\hat{f}_l(x; \kappa^l)$, $l = 1, ..., n_d$, with $x \in X$ being a vector of n_x state variables and κ^l being a vector of $\tilde{n}_\kappa$ coefficients.
2. Construct a computable operator $\hat{\mathcal{R}}$ by using a non-product monomial rule to approximate conditional expectations.
3. Use the basis functions of $\hat{f}_l$ as weighting functions $\omega_i(x)$, $i = 1, ..., \tilde{n}_\kappa$, construct the multi-dimensional approximating function

$$\hat{f}(x; \kappa) = \begin{pmatrix} \hat{f}_1(x; \kappa^1) \\ \vdots \\ \hat{f}_{n_d}(x; \kappa^{n_d}) \end{pmatrix},$$

where $\kappa = (\kappa^1, ..., \kappa^{n_d})'$, and compute the weighted residuals

$$\int_X \hat{\mathcal{R}}_l(\hat{f}(x;\kappa))\omega_i(x)dx, \qquad i = 1, ..., \tilde{n}_\kappa, \quad l = 1, ..., n_d,$$

using a non-product monomial rule for the integration step.[8]

4. Search for the $n_\kappa = n_d \times \tilde{n}_\kappa$ coefficients in κ that equate all weighted residuals to zero.

The following subsections discuss each step in greater detail.

3.1. Selection of approximating functions

The first step in implementing the Galerkin algorithm is to select the approximating functions, i.e., the basis and approximation degree. In principle, the method is compatible with many different sets of basis functions. The present paper employs as basis a complete set of Chebyshev polynomials (Judd, 1992; Gaspar and Judd, 1997). The approximation for each parameterized decision rule thus takes the form

$$\hat{f}_l(x;\kappa^l) = \sum_{i=1}^{\tilde{n}_\kappa} \kappa_i^l \psi_i(\xi(x)) \tag{15}$$

where ξ is a linear mapping from the state space X into $[-1,1]^{n_x}$ and $\psi_i(\xi(x)) \in \Psi$ with

$$\Psi = \left\{ \prod_{j=1}^{n_x} T_{i_j}(\xi(x_j)) \Big| \sum_{j=1}^{n_x} i_j \leq p, 0 \leq i_1, ..., i_{n_x} \right\}$$

being a complete set of Chebyshev polynomials of total degree p in n_x variables.[9] The choice of approximation order p is typically a matter of computational cost. For large models, such as the multi-country Real Business Cycle model with several countries, computing high-order approximations is impeded by the large associated computational cost, and the researcher is confined to work with linear or quadratic approximations, respectively.

3.2. Choice of numerical method to evaluate conditional expectations

If the model to be solved is stochastic and features random variables with continuous support, the operator $\mathcal{R}(\hat{f}(x;\kappa))$ in general cannot be evaluated without approximation error on a computer: it involves conditional expectations over non-linear functions of the model's variables which cannot be computed in closed form. $\mathcal{R}$ therefore has to be replaced with a computable operator $\hat{\mathcal{R}}$ by using a numerical method to approximate the conditional expectations. Section 2.2 has presented some methods that can be used for this purpose. In choosing between these methods, one typically faces a trade-off between speed and accuracy. For example, the degree five rule $[E_n^{r^2} d5]$ may be the optimal choice for medium scale models, but for larger models the degree three rule $[E_n^{r^2} d3]$ might be more attractive due to its low computational cost.

[8] $\hat{\mathcal{R}}_l$ denotes the l-th equation in the multi-dimensional operator $\hat{\mathcal{R}}$.

[9] Univariate Chebyshev polynomials are defined by the recursion $T_0(y) = 1$, $T_1(y) = y$, $T_j(y) = 2yT_{j-1}(y) - T_{j-2}(y)$, $j = 2, 3, ...$ Multivariate Chebyshev polynomials can be easily constructed as the products of the respective univariate polynomials.

3.3. Computation of weighted residuals

While the model's true solution satisfies $\mathcal{R}(f) = 0$ over the entire state space X, $\hat{\mathcal{R}}(\hat{f}(x;\kappa))$ is typically not equal to zero due to approximation error; $\hat{\mathcal{R}}$ is therefore often referred to as the model's residual function. The Galerkin method determines the coefficients κ of the approximating polynomial by constructing a system of weighted sums (or averages) over residuals and equating these sums to zero. As originally proposed by Galerkin (1915), the weighting functions are the basis functions of the approximating polynomial, $\omega_i(x) = \psi_i(\xi(x))$.[10] The weighted residuals are thus given by

$$\hat{u}_{l,i}(\kappa) = \int_X \hat{\mathcal{R}}_l(\hat{f}(x;\kappa))\omega_i(x)dx \tag{16}$$

for $i = 1, ..., \tilde{n}_\kappa$ and $l = 1, ..., n_d$. Section 2.1 has presented several methods to solve the integration problem in (16), including two non-product monomial rules of degrees $d = 5$ and $d = 7$, respectively. In order to apply these rules, the following identity can be exploited:

$$\int_X \hat{\mathcal{R}}_l(\hat{f}(x;\kappa))\psi_i(\xi(x))dx = \int_{C_n} g(z)dz$$

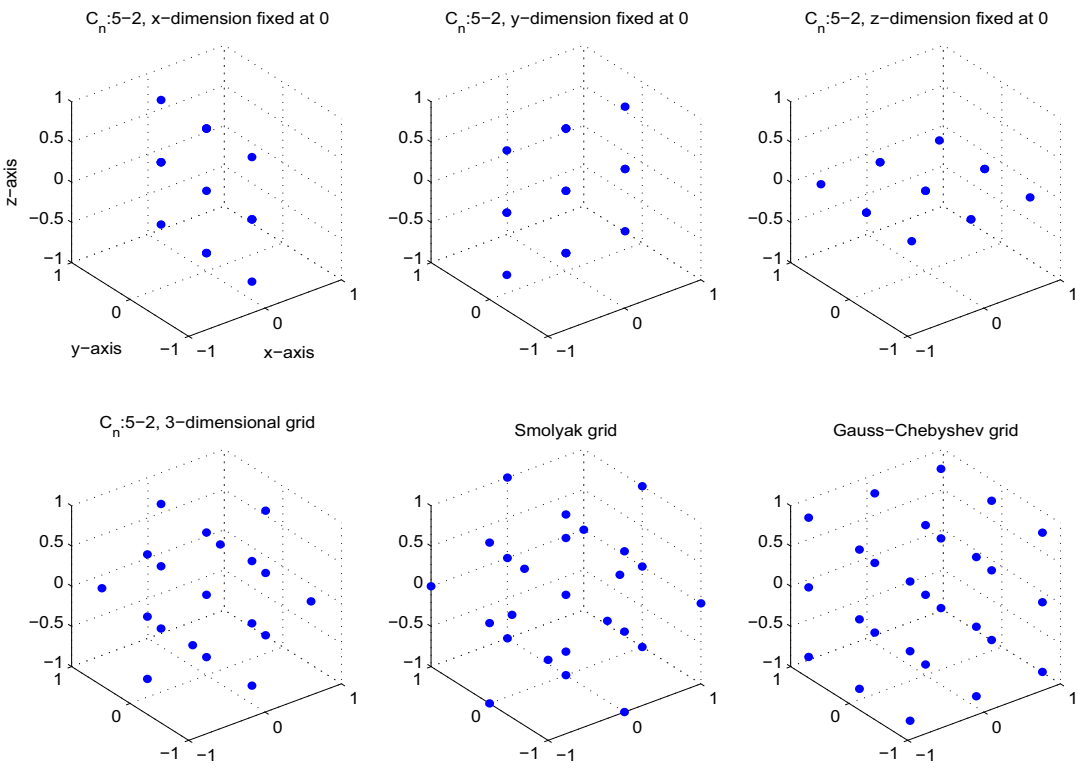
where $C_n = [-1, 1]^{n_x}$, $z = \xi(x)$, and $g(z) = \hat{\mathcal{R}}_l(\hat{f}(\xi^{-1}(z);\kappa))\psi_i(z)[\xi'(x)]^{-1}$.

Rule $[C_n d5]$ is particularly attractive for the integration step. First, it uses only points that are well inside the hypercube; this contributes to the numerical stability of the Galerkin algorithm by avoiding that variables get out of bounds as described in Heer and Maussner (2006). Second, the rule $[C_n d5]$ is quite efficient, i.e., it uses the lowest number of grid points of all degree 5 monomial rules currently known (Cools, 2003). Figure 1 shows the grid points employed by $[C_n d5]$ for three dimensions, and compares them to alternative approaches (the Smolyak algorithm used by Malin et al. (2010) and the Gauss-Chebyshev integration rule).

A drawback of rule $[C_n d5]$ is that it only allows for first and second-order approximations to the decision rules, i.e. $p \leq 2$. This is a consequence of its specific choice of grid points, causing the regressors in the approximate functions (15) to be linearly dependent for $p \geq 3$. As hinted before, this shortcoming is not overly restrictive for relatively large models (say, models with more than 10-15 state variables) as higher-order approximations are arguably infeasible for any global solution method due to their huge computational cost. For smaller models, however, we may want to compute such higher-order approximations. In particular, obtaining third-order approximations can be important in applications where a value function is parameterized, because third-order accurate value functions are often needed to obtain second-order accurate decision rules. In such applications, one can resort to the rule $[C_n d7]$. This rule does allow for an approximation order $p = 3$, but it is computationally more costly than $[C_n d5]$ as illustrated by Table 1.

[10] Since the basis functions are chosen from a complete set of functions, using them as weighting functions implies that the Galerkin solution approaches the true solution as more and more basis functions are included (as the order of approximation p goes to infinity). This argument employs the fact that the residual function $\hat{\mathcal{R}}$ is continuous over X, and therefore it is identically zero if it is orthogonal to every member of a complete set of functions; see McGrattan (1999, p. 117).

Figure 1: Grid points in three dimensions



Note: For clarity, the top three panels show the grid points of the rule $[C_n d5]$ in two of the three dimensions, holding the third dimension fixed at 0; the bottom left panel shows the three-dimensional grid. To ease comparison with alternative methods, the bottom middle panel and the bottom right panel illustrate the grids associated with the Smolyak and Gauss-Chebychev approach, respectively.

Table 1: Number of grid points used by different integration formulas

	Formula	Dimension (n_x)				
		2	3	4	12	20
Gaussian ($d = 5$)	3^{n_x}	9	27	81	531.441	3.486.784.401
$[C_n d5]$	$2{n_x}^2 + 1$	9	19	33	289	801
$[C_n d7]$	$(4{n_x}^3 + 8n_x + 3)/3$	17	45	97	2.337	10.721

3.4. Finding the solution

In the last step of the Galerkin algorithm, the parameter vector κ is determined such that weighted residuals equal zero. This amounts to solving a system of n_κ nonlinear equations in n_κ

unknowns,

$$\hat{u}_{l,i}(\kappa) = 0, \qquad i = 1, ..., \tilde{n}_\kappa, \quad l = 1, ..., n_d. \tag{17}$$

Given a proper initial guess, obtained for example from a log-linear solution of the model, it is conceptually straightforward to solve (17). Among other alternatives, one can use a Newton method as described in chapter 4 of Judd (1998). The present paper follows this strategy to determine the parameter vector such that (17) holds. Using this vector in (15) finally gives the model's approximate solution $\hat{f}$.

4. Implementing the algorithm to solve the multi-country Real Business Cycle model

In this section I describe the proposed Galerkin method within a particular application. Specifically, I illustrate the implementation of the method to solve the multi-country Real Business Cycle model in Den Haan et al. (2010). Juillard and Villemot (2010) provide a detailed description of this model, discussing four variants (*I-IV*) which differ with respect to the functional forms chosen for the utility and production functions. In the present paper, I provide only the minimal description of the economy sufficient to describe the implementation of the solution algorithm.

A key feature of the model to be solved is that the decentralized equilibrium is Pareto-optimal and therefore can be derived by solving an appropriately designed social planner's problem. The planner's optimality conditions are given by:

$$0 = \tau^j u'^j_c(c^j_t, \ell^j_t) - \lambda_t, \tag{18}$$

$$0 = \tau^j u'^j_\ell(c^j_t, \ell^j_t) + \lambda_t a^j_t f'^j_\ell(k^j_t, \ell^j_t), \tag{19}$$

$$\begin{aligned} 0 = {} & \lambda_t \left[1 + \phi\left(\frac{i^j_t}{k^j_t} - \delta\right)\right] \\ & - \beta \mathbb{E}_t \left\{ \lambda_{t+1} \left[1 + a^j_{t+1} f'^j_k(k^j_{t+1}, \ell l^j_{t+1}) + \phi\left(1 + \frac{1}{2}\left(\frac{i^j_{t+1}}{k^j_{t+1}} - \delta\right)\right)\left(\frac{i^j_{t+1}}{k^j_{t+1}} - \delta\right)\right]\right\}, \end{aligned} \tag{20}$$

$$0 = k^j_{t+1} - (1-\delta)k^j_t - i^j_t, \tag{21}$$

$$0 = \sum_{j=1}^{N} \left(c^j_t + i^j_t - \delta k^j_t\right) - \sum_{j=1}^{N} \left[a^j_t f^j(k^j_t, \ell^j_t) - \frac{\phi}{2} k^j_t \left(\frac{i^j_t}{k^j_t} - \delta\right)^2\right], \tag{22}$$

$$0 = \ln a^j_t - \rho \ln a^j_{t-1} - \sigma(e_t + e^j_t). \tag{23}$$

The parameter τ^j is the weight given to country $j \in \{1, ..., N\}$ in the planner's optimization problem, with N being the total number of countries; $u^j(c^j_t, \ell^j_t)$ gives country j's instantaneous utility as a function of its period-t consumption c^j_t and labor effort ℓ^j_t; u'^j_c and u'^j_ℓ denote marginal utilities; λ_t is the Lagrangian multiplier attached to the world budget constraint; a^j_t denotes country j's technology level; $a^j_t f^j(k^j_t, \ell^j_t)$ gives country j's production net of capital depreciation, with k^j_t being the capital stock employed in country j; f'^j_k and f'^j_ℓ denote marginal productivities; ϕ determines the size of capital adjustment costs; δ is the capital depreciation rate; β is the planner's time preference factor; ρ and σ determine the persistence and volatility of the countries' technology levels; e_t and e^j_t denote aggregate respectively idiosyncratic i.i.d. $N(0,1)$ technology innovations.

4.1. Solving variant I of the multi-country Real Business Cycle model

In the following, I describe the implementation of the Galerkin method to solve model I in Juillard and Villemot (2010). This model features inelastic labor supply (each country supplies one unit of labor), a CRRA utility function in consumption with γ_j denoting the inverse of the coefficient of relative risk aversion, and a Cobb-Douglas production function $f^j(k_t^j, \ell_t^j) = A(k_t^j)^\alpha$, with parameters $\alpha \in (0,1)$ and $A > 0$. As there is no labor choice, and eliminating the exogenous laws of motion for the productivity levels, one can characterize the model's equilibrium by the $3N + 1$ conditions:

$$0 = \tau^j (c_t^j)^{-\gamma_j} - \lambda_t, \tag{24}$$

$$\begin{aligned} 0 = \lambda_t \left[1 + \phi\left(\frac{i_t^j}{k_t^j} - \delta\right)\right] \\ - \beta \mathbb{E}_t \left\{ \lambda_{t+1} \left[1 + (a_t^j)^\rho e^{\sigma(e_{t+1} + e_{t+1}^j)} A\alpha (k_{t+1}^j)^{\alpha-1} + \frac{\phi}{2}\left(2 - \delta + \frac{i_{t+1}^j}{k_{t+1}^j}\right)\left(\frac{i_{t+1}^j}{k_{t+1}^j} - \delta\right) \right] \right\}, \end{aligned} \tag{25}$$

$$0 = k_{t+1}^j - (1-\delta) k_t^j - i_t^j, \tag{26}$$

$$0 = \sum_{j=1}^N \left(c_t^j + i_t^j - \delta k_t^j\right) - \sum_{j=1}^N \left[a_t^j f^j(k_t^j, \ell_t^j) - \frac{\phi}{2} k_t^j \left(\frac{i_t^j}{k_t^j} - \delta\right)^2 \right] \tag{27}$$

with $j = 1, ..., N$. Notice that, at the time when period-t decisions are made, the state vector composed of capital stocks and productivity levels, $x_t = (k_t^1, ..., k_t^N, a_t^1, ..., a_t^N)$, is known. The planner's optimal decisions can be expressed as functions of this state vector, $\lambda_t = \lambda(x_t)$, $c_t^j = \mathbf{c}^j(x_t)$, $k_{t+1}^j = \mathbf{k}'^j(x_t)$, and $i_t^j = \mathbf{i}^j(x_t)$ for $j = 1, ..., N$.

To solve the model (24)-(27) using the Galerkin method, it is useful to first represent it by an operator equation $\mathcal{R}(f)$ such that the arguments outlined in Section 3 directly apply. In constructing this representation, a researcher has several degrees of freedom. Specifically, there exists some freedom in choosing which functions to approximate by parametric forms and which equilibrium conditions to use as residual functions.

In the present application it turns out convenient to parameterize the capital decision rules and to use as residual functions the dynamic equilibrium conditions (25). First, the capital decision rules in the model under consideration are relatively linear and smooth, and therefore can be reasonably well approximated by polynomials.[11] Second, conditional upon parameterizing the future capital decision rules, the remaining choices can be computed in closed form (and thus it is relatively straightforward to construct the operator representation $\mathcal{R}(f)$). In particular, (26) can be solved to obtain the individual investment levels, which can then be plugged into the resource constraint (27) to yield aggregate consumption $C_t = \sum_j c_t^j$. By equation (24) for $j = 1, ..., N$, one can then obtain c_t^1 by numerically solving

$$C_t = c_t^1 + \sum_{j=2}^N \left[\frac{\tau^1}{\tau^j}(c_t^1)^{-\gamma^1}\right]^{-\frac{1}{\gamma^j}} \tag{28}$$

[11] See Malin, Krueger, and Kubler (2007) for a discussion.

for c_t^1. Upon knowing c_t^1, one can compute the remaining $N-1$ individual consumption levels and the Lagrangian multiplier from (24). Proceeding in this way, all control variables are solved for such that the static conditions (24), (26), and (27) hold exactly, and the operator $\mathcal{R}(f)$ therefore contains only the N dynamic equilibrium conditions (25).

Note that the described procedure involves numerically solving a non-linear equation for c_t^1, i.e., one needs to compute a non-linear function $c_t^1 = \tilde{c}(C_t)$ that satisfies (28). It is conceptually straightforward to do this using a numerical solver at each grid point and iteration step of the solution procedure, however, such an approach would be computationally costly. In light of this difficulty, the route taken in the present paper is to use (28) to construct a polynomial approximation $\hat{c}$ to the function $\tilde{c}$ over an appropriately chosen interval $[\underline{C}, \bar{C}]$, and use this approximation to compute consumption of country one as $c_t^1 = \hat{c}(C_t)$ within the Galerkin procedure. Importantly, as the function $\tilde{c}$ has only one argument and the approximation $\hat{c}$ needs to be computed only once before the Galerkin algorithm is initiated, one can use a very high order of approximation for $\hat{c}$, which allows for an approximation error close to machine precision. The idea to use an approximation to the function $\tilde{c}$ is not an innovation of the present paper, but was first pointed out and employed in an earlier version of Maliar et al. (2010). In that paper, Maliar and Maliar (2007), the authors computed consumption levels on a grid outside of the iterative circle and used an interpolation strategy to restore c_t^1 within the solution procedure. Finally, notice that alternative to approximating the function $\tilde{c}$ one could parameterize the decision rule for c_t^1, i.e., use a polynomial approximation to $\mathbf{c}^1(x_t)$. This approach is however sub-optimal: as $\mathbf{c}^1$ has $2N$ arguments, one would have to use a low-order approximation and thus introduce additional approximation error that can be avoided by resorting to the former strategy.

Once the operator representation $\mathcal{R}(f)$ is chosen, the Galerkin method can be used to compute an approximate model solution $\hat{f}$ as described in Section 3. This procedure involves choosing among several implementation details as laid out in Sections 3.1-3.4. The specific Galerkin implementation used in the present paper is characterized by the choices summarized in Table 2.

Table 2: Implementation details for model variant I

1. Approximating function	Complete Chebyshev polynomial of order $p = 2$
2. Method to evaluate conditional expectations	$[E_n^{r^2} d5]$ for $N = 2$, $[E_n^{r^2} d3]$ for $N > 2$
3. Method to compute weighted residuals	$[C_n d5]$
4. Solver to find the coefficient vector	Newton (starting from log-linear perturbation guess)

The rationale behind these choices is as follows. Regarding the functional form of the approximate decision rules, this paper follows Gaspar and Judd (1997) and uses

$$\hat{\mathbf{k}^{r}}^{j}(x_t; \kappa^j) = \sum_{i=1}^{\tilde{n}_\kappa} \kappa_i^j \psi_i(\xi(x_t)) \tag{29}$$

where $\psi_i(\xi(x_t)) \in \Psi$ with Ψ being a complete basis of Chebyshev polynomials up to the second order ($p = 2$). To map the state space X into the unit hypercube $[-1, 1]^{n_x}$, a linear transformation $\xi(x^i) = 2(x^i - \underline{x}^i)/(\bar{x}^i - \underline{x}^i) - 1$ is used which requires to postulate lower and upper bounds for each state variable. The specific choice of bounds is found to matter noticeably for accuracy. For

model variant I, a relatively narrow interval ($\underline{k} = 0.95k_{ss}$, $\bar{k} = 1.05k_{ss}$, $\underline{a} = 0.9a_{ss}$, $\bar{a} = 1.1a_{ss}$, with ss denoting the steady state value of a variable) is found to deliver very accurate results.

The specific choice of monomial rule for computing conditional expectations is mostly motivated by computational cost. The degree five rule $[E_n^{r^2}d5]$ is only used for the low-dimensional problems with $N = 2$, while $[E_n^{r^2}d3]$ is employed for the larger models ($N > 2$).

To compute the weighted residuals, the non-product rule $[C_nd5]$ is used for all model specifications. Gains from using the degree seven rule $[C_nd7]$ were found to be small for an approximation order of $p = 2$, and therefore did not justify the extra computational burden associated with choosing $[C_nd7]$ over $[C_nd5]$.

Finally, to equate the system of weighted residuals to zero, first an initial guess for the coefficient vector is obtained from a log-linear perturbation solution around the non-stochastic steady state using the method and computer code provided by Klein (2000). This guess is then used in a Newton solver[12] to find the solution to (17). Note that the solver searches over a high-dimensional parameter vector κ,[13] and at each iteration step computes a Jacobian. The residual function $\hat{\mathcal{R}}$ is therefore evaluated very frequently. To achieve reasonable computing times, it is extremely important to code the residual function such that its evaluation is cheap. The first prerequisite to achieve this goal is avoiding non-linear equations solving; this can generally be accomplished by choosing an appropriate set of parameterized decision rules. Finally, the computer code must also avoid for-loop structures as the execution of loops is very time-consuming in a matrix oriented environment like MATLAB.

4.2. Solving variants II-IV of the multi-country Real Business Cycle model

The model variants $II - IV$ described by Juillard and Villemot (2010) feature an endogenous labor choice and are therefore more challenging to solve. In the following, I briefly describe how the algorithm described in the previous section is modified to deal with these challenges.[14]

Model II employs a utility function that is separable in consumption and labor together with a Cobb-Douglas production function over capital and labor. Under this specification of functional forms, it is no longer possible to solve for aggregate consumption from the world budget constraint (22), given the parameterized decision rules for future capital holdings. To address this problem, I add the aggregate consumption policy to the set of parameterized decision rules and use the world budget constraint (27) as an additional residual function in $\mathcal{R}$. Apart from this modification, the same approach as laid out for model I is used solve model II.

Models III and IV feature non-separable utility. This further complicates matters, as one can no longer compute individual consumption levels from aggregate consumption using (18). Notice, however, that one can use (18) and (19) to express the labor supply of country j as a non-linear function $\tilde{\ell}_t^j(k_t^j, a_t^j, \lambda_t)$. As for the function $\tilde{c}$, a very accurate high-order polynomial approximation to each function $\tilde{\ell}_t^j$, $j = 1, ..., N$, can be computed outside of the Galerkin algorithm, and this

[12] To be precise, the fcsolve function developed by Chris Sims and later modified by Fabrice Collard is employed.

[13] The dimension of κ is $n_d(1 + 2N + N(2N + 1))$ where n_d gives the number of parameterized decision rules and N the number of countries. For model variant I where only the decision rules for future capital holdings are parameterized, $n_d = N$. This implies, for example, that $n_\kappa = 2.310$ when $N = 10$.

[14] See the MATLAB codes accompanying this paper for further details.

approximation can be used to compute labor supplies within the solution algorithm. Once the labor supplies are determined, the remaining control variables can again be computed easily from the equilibrium conditions. The strategy just described has been used for models III and IV. To make this strategy feasible at a reasonable computational cost, the decision rule for λ has to be added to the set of parameterized decision rules. In particular, the implementation used considers $f = (\log \lambda, \mathbf{k}'^1, ..., \mathbf{k}'^N)'$ together with the same operator representation used for model II. Notice that the log-approximation to λ is chosen as it gives higher accuracy than a level-approximation. Nevertheless, as $\log \lambda$ is still relatively non-linear, adding it to the parameterized decision rules comes at a cost in terms of accuracy.

5. Improving accuracy

The performance (as measured by the computational speed and numerical accuracy) of the proposed monomial rule Galerkin method is examined in detail by Kollmann et al. (2010). In particular, these authors present several accuracy tests and compare the Galerkin method to competing approaches. Out of space considerations, their findings are not replicated in the present paper. Rather I briefly examine two possible directions for further improving the numerical accuracy of the algorithm.

The first (obvious) direction for improving numerical accuracy is to increase the order of approximation in the parameterized decision rules, i.e., increase p. This comes at the expense of increased computational cost, in particular since $p \geq 3$ impedes the use of rule $[C_n d5]$ for computing weighted residuals.[15] Instead, computationally more intensive rules such as $[C_n d7]$ must be employed for this purpose. To gain some intuition about the trade-off between speed and accuracy when moving from second- to third-order approximations, consider as an example the asymmetric specification of model I with $N = 2$ countries. The second-order Galerkin approximation for this model has an associated CPU time of slightly less than a second, while the $\log_{10}$ measures of the average and maximum error on a stochastic simulation of 10.000 periods are given by -6.33 and -4.49, respectively.[16] In comparison, a third-order Galerkin approximation using $[C_n d7]$ has an associated CPU time of roughly 2 seconds, while the corresponding error measures are -7.64 and -5.56, respectively. For this relatively low-dimensional example, the accuracy improvement of more than one order of magnitude may likely compensate for the increase in computational cost by one second. Notice, however, that for models with larger N the difference in speed between a second- and third-order approximation will be significantly larger, while the difference in accuracy will presumably be roughly the same.

A second possible route for improving accuracy for models II-IV is to use an implementation of the Galerkin method that only parameterizes the capital decision rules. Such an implementation, however, requires that *all* decision rules other than the capital decision rules must be computed directly from the static equilibrium conditions, i.e., a large system of non-linear equations has to be solved repeatedly. Using a Newton method to solve this system grid point by grid point is conceptually straightforward, but in practice such an approach is impeded by the huge running time

[15] Recall the discussion in Section 3.3.

[16] See Kollmann et al. (2010).

associated with it. To address this difficulty, the contribution to this special issue by Maliar et al. (2010) develops a vectorized *iteration-on-allocation* strategy for solving the system of static equilibrium conditions at low computational cost. Using their approach within the proposed Galerkin method would arguably allow to achieve higher accuracy for models II-IV, while the running time of the algorithm would possibly be only moderately affected.

6. Acknowledgements

I thank Wouter Den Haan and Michel Juillard together with three anonymous referees for their comments and suggestions which have substantially improved the quality and clarity of this paper. Moreover, I thank Burkhard Heer, Dirk Krueger, Felix Kubler, Benjamin Malin, Stefan Niemann, Michael Reiter, Gerhard Sorger and the participants of the *1st Annual Workshop in Computational Financial Economics* in Zurich for helpful comments. Financial support from the Austrian Science Fund (FWF) under project number P19686 is gratefully acknowledged.

References

References

Algan, Y., Allais, O., Den Haan, W.J., 2010. Solving the incomplete markets model with aggregate uncertainty using parameterized cross-sectional distributions. Journal of Economic Dynamics and Control 34, 59–68.

Aruoba, S. B., Fernandez-Villaverde, J., Rubio-Ramirez, J., 2006. Comparing solution methods for dynamic equilibrium economies. Journal of Economic Dynamics and Control 30, 2477–2508.

Bellman, R., 1961. Adaptive Control Processes: A Guided Tour. Princeton University Press, Princeton, NJ.

Christiano, L.J., Fisher, J.D.M., 2000. Algorithms for solving dynamic models with occasionally binding constraints. Journal of Economic Dynamics and Control 24, 1179–1232.

Cools, R., 2003. An encyclopaedia of cubature formulas. Journal of Complexity 19, 445-453.

Den Haan, W., Judd, K.L., Juillard, M., 2004. Comparing Numerical Solutions of Models with Heterogenous Agents. Mimeo. LBS, Stanford University, and CEPREMAP.

Den Haan, W., Judd, K.L., Juillard, M. , 2010. Computational Suite of Models with Heterogeneous Agents: Multi-country Real Business Cycle Models. Journal of Economic Dynamics and Control, this issue.

Den Haan, W.J., Rendahl, P., 2010. Solving the incomplete markets model with aggregate uncertainty using explicit aggregation. Journal of Economic Dynamics and Control 34, 69–78.

Doraszelski, U., Judd, K.L., 2005. Avoiding the Curse of Dimensionality in Dynamic Stochastic Games. NBER Technical Working Papers 0304. National Bureau of Economic Research, Inc.

Galerkin, B., 1915. Rods and plates. Vestnik Inzh. 1, 897–908.

Gaspar, J., Judd, K.L., 1997. Solving Large-Scale Rational-Expectations Models. Macroeconomic Dynamics 1, 45–75.

Hammer, P. C., Stroud, A.H., 1957. Numerical evaluation of multiple integrals I. Mathematical Tables and Other Aids to Computation 11, 59–67.

Hammer, P. C., Stroud, A.H., 1958. Numerical evaluation of multiple integrals II. Mathematical Tables and Other Aids to Computation 12, 272–279.

Heer, B., Maussner, A., 2006. Projection methods and the curse of dimensionality. Mimeo.

Judd, K., 1992. Projection methods for solving aggregate growth models. Journal of Economic Theory 58, 410–452.

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

Juillard, M., Villemot, S., 2010. Multi-Country Real Business Cycle Models: Accuracy Tests and Testing Bench. Journal of Economic Dynamics and Control, this issue.

Klein, P., 2000. Using the generalized Schur form to solve a multivariate linear rational expectations model. Journal of Economic Dynamics and Control 24, 1405–1423.

Kollmann, R., Maliar, S., Malin, B., Pichler, P., 2010. Comparison of Solutions to the Multi-Country Real Business Cycle Model. Journal of Economic Dynamics and Control, this issue.

Krommer, A.R., Ueberhuber, C.W., 1998. Computational Integration. Society for Industrial Mathematics, Philadelphia, Pennsylvania.

Maliar, L., Maliar, S., 2007. Comparing numerical solutions of models with heterogeneous agents, Model A: a simulation - based parameterized expectations algorithm. Discussion paper.

Maliar, L., Maliar, S., Valli, F., 2010. Solving the incomplete markets model with aggregate uncertainty using the Krusell-Smith algorithm. Journal of Economic Dynamics and Control 34, 42–49.

Maliar, S., Maliar, L., Judd, K.L. , 2010. Solving the Multi-Country Real Business Cycle Model Using Ergodic Set Methods. Journal of Economic Dynamics and Control, this issue.

Malin, B., Krueger, D., Kubler, F., 2010. Solving the Multi-Country Real Business Cycle Model Using a Smolyak collocation method. Journal of Economic Dynamics and Control, this issue.

Malin, B., Krueger, D., Kubler, F., 2007. Computing Stochastic Dynamic Economic Models with a Large Number of State Variables: A Description and Application of a Smolyak-Collocation Method. NBER Technical Working Papers 0345. National Bureau of Economic Research, Inc.

Marimon, R., Scott, A., 1999. Computational Methods for the Study of Dynamic Economies. Oxford University Press, New York.

Martin, F., 2009. A Positive Theory of Government Debt. Review of Economic Dynamics 12, 608–631.

McGrattan, E., 1999. Application of weighted residual methods to dynamic economic models. In Computational Methods for the Study of Dynamic Economies, ed. by R. Marimon, A. Scott, pp. 114–142. Oxford University Press.

Miranda, M.J., Fackler, P.M., 2002 .Applied Computational Economics and Finance. The MIT Press, Cambridge, Massachusetts.

Mustard, D., Lyness, J., Blatt, J., 1963. Numerical quadrature in n dimensions. Computer Journal 6, 75–87.

Niemann, S., Pichler, P., Sorger, G., 2010. Central bank independence and the monetary instrument problem. Economics Discussion Papers 687. University of Essex, Department of Economics.

Ortigueira, S., 2006. Markov-Perfect Optimal Taxation. Review of Economic Dynamics 9, 153–178.

Phillips, G., 1980. A survey of one-dimensional and multidimensional numerical integration. Computer Physics Communications 20, 17–27.

Reiter, M., 2009. Solving heterogeneous-agent models by projection and perturbation.Journal of Economic Dynamics and Control 33, 649–665.

Stenger, F., 1971. Tabulation of Certain Fully Symmetric Numerical Integration Formulas of Degree 3, 5, 7, 9, and 11. Mathematics of Computation 25, 935.

Stroud, A., 1971. Approximate calculation of multiple integrals. Springer, Berlin.

Stroud, A.H., Secrest, D., 1963. Approximate integration formulas for certain spherical symmetric regions. Mathematics of Computation 17, 105–135.

Winschel, V., Kraetzig, M., 2010. Solving, Estimating, and Selecting Nonlinear Dynamic Models Without the Curse of Dimensionality. Econometrica 78, 803–821.