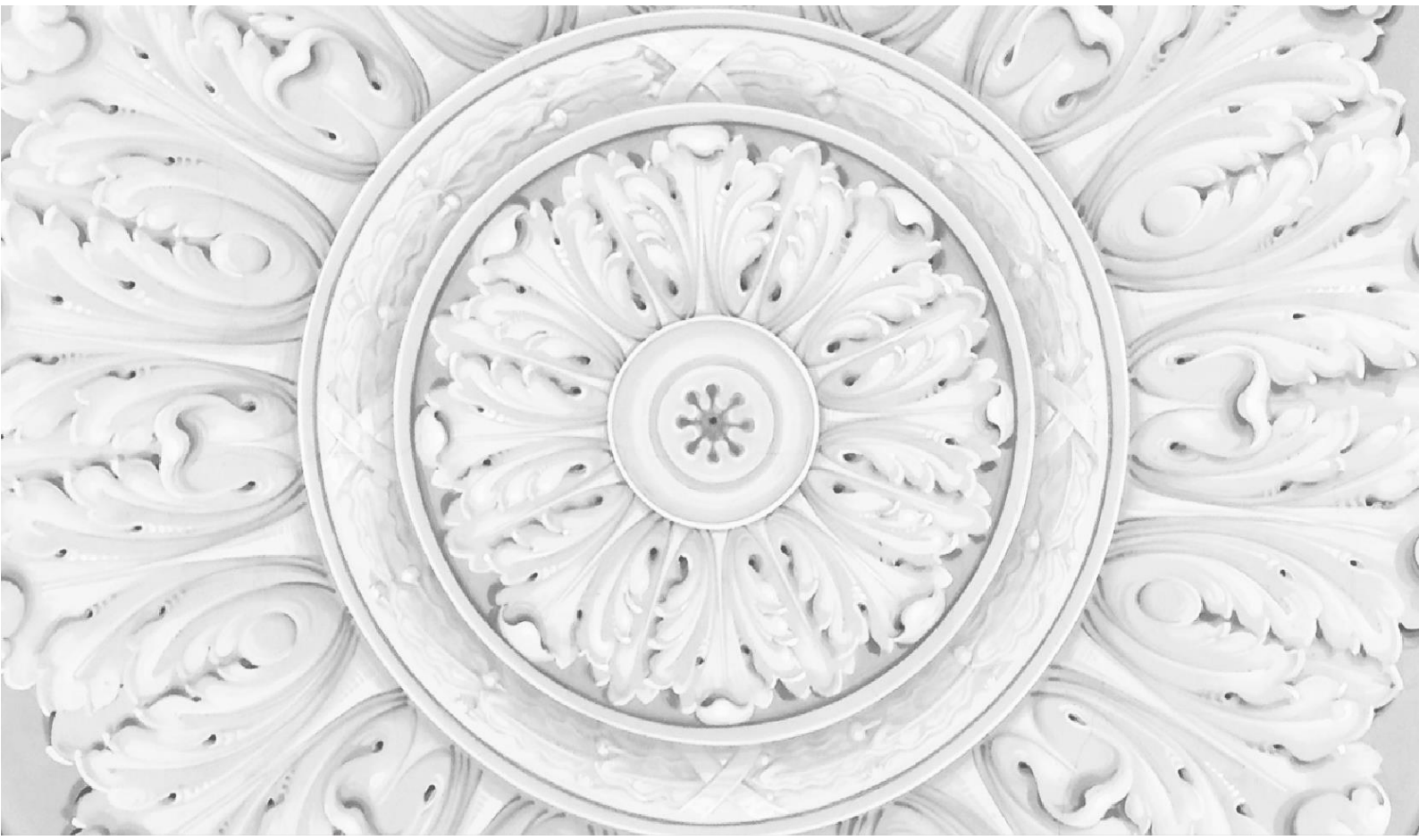
Bank of Russia

The Central Bank of the Russian Federation



WORKING PAPER SERIES

Sergei Seleznev

Solving DSGE models with stochastic trends

No. 15 / September 2016

Sergei Seleznev
E-mail: SeleznevSM@cbr.ru

I am grateful to Bulat Gafarov, Sergey Ivaschenko, Dmitry Kreptsev, Nikita Kulagin, Pavel Molyakov, Andrei Polbin, Alexey Ponomarenko and Sergey Slobodyan for their helpful comments and suggestions. Any errors or discrepancies belong to the authors.

© Bank of Russia, 2016

Postal address 12 Neglinnaya Street, Moscow, 107016
Telephone +7 495 771-91-00, +7 495 621-64-65 (fax)
Website www.cbr.ru

All rights reserved. The views expressed in this paper (Bank of Russia working paper series) are solely those of the author (authors) and do not necessarily reflect the official position of the Bank of Russia. The Bank of Russia does not hold responsibility for the con-tent of the paper (Bank of Russia working paper series). Any reproduction, publication and reprint in the form of a different publication, in whole or in part, is permitted only with the explicit authorisation of the authors.

Abstract

We propose an algorithm for solving DSGE models with stochastic trends. Several implementations help us to solve the model with a small number of stochastic trends in the absence of a balanced growth path fast and allow us to control the accuracy of approximation in a certain range. Taking into account the fact that many implementations can be easily parallelized, this algorithm enables the estimation of models in the absence of a balanced growth path. We also provide a number of possible methods for estimation.

Keywords: Non-stationary DSGE, stochastic trends, Smolyak's algorithm, perturbation method.
JEL classification: C61, C63

CONTENTS

1. INTRODUCTION

Over the past decade, DSGE models have gained popularity among macroeconomists. In the presence of theoretical interpretation, they can be fitted to data and used, for example, for forecasting or analysing optimal policy rules. To work with the model and fit it to data, it is necessary to solve the model, and there is a vast literature focusing on the solution and estimation of DSGE models (see, for instance, the review by Fernandez-Villaverde et al. (2016)). However, fewer papers consider non-stationary models, while part of the observed data is non-stationary in nature (GDP, investment, consumption, etc.). Models that can be reduced by transforming the variables into stationary ones (for example Fernandez-Villaverde and Rubio-Ramirez (2007)) are standard, but such a class leaves out many interesting models. Models in which the dynamics of trends is predetermined (and/or expectations about the dynamics of these trends are formed) have also been developed. Such models are usually solved using backward recursion, such as that of Kulish and Pagan (2014).

Sometimes, to fit the model to data, trends are removed using other models or various filters. However, such an approach can lead to biased estimates of model parameters.[1] Furthermore, Canova (2014) describes the hybrid approach in which the trend and the cycle are modelled separately. Such a method can lead to the loss of theoretical interpretation of trends and their inconsistency with microfounded models.

A number of algorithms have been proposed for a model with stochastic trends (unit root processes). The perturbation algorithm based on Taylor expansion around any point is discussed by Kim et al. (2008) and Lan and Meyer-Gohde (2014). In Evans and Phillips (2015) the algorithm of linearization about the current state is offered. The algorithm is applicable to simulations, and it is difficult to use this algorithm for estimation because of its computational complexity.

In this work we introduce an algorithm for the solution of models with stochastic trends, which, like the algorithms of Kim et al. (2008), Lan and Meyer-Gohde (2014) and Evans and Phillips (2015), does not assume the existence of a balanced growth path. We generalize the algorithms of Kim et al. (2008) and Lan and Meyer-Gohde (2014) and add elements of linearization about the current state. The described algorithm consists of two steps. The dependence between a steady state and stochastic trends is approximated in the first step. In the second step, the deviation from a steady state is approximated. This deviation depends on the point at which the approximation is made.

[1] See Gorodnichenko and Ng (2010).

For each of the two steps, we present several possible implementations, all of which do not depend on shock realization (the solution does not depend on the trajectory by which the system reached the current point). The trend part is approximated by the grid search, the solution in a finite number of points, the linear approximation, Smolyak's algorithm and the grid search using differential equations. For the approximation of the cyclical part, we use the same algorithms, except for the grid search using differential equations, and we add constant approximation.

Despite the fact that, in a number of implementations, this algorithm is more computationally difficult than the algorithms of Kim et al. (2008) and Lan and Meyer-Gohde (2014), it allows us to model the non-linear dependences, which are a consequence of the change in the stochastic trends. The running time of the majority of these implementations is acceptable for the estimation of models with a small number of trends. In addition, most of the procedures proposed here can easily be parallelized, significantly reducing the time of the algorithm.

This algorithm can be applied, for example, to models in which the technology is a process with a unit root and the utility function is not logarithmic (as an alternative to adding trends directly to a utility function; see, for instance, Aruoba et al. (2016)), for models of oil-exporting countries, where the real oil price follows a random walk, or for models of small open economies, where the domestic and foreign technologies follow various stochastic trends.

The rest of the paper is organized as follows. Section 2 describes the two-step algorithm. Section 3 discusses the estimation procedures. Various implementations and results are presented in Section 4. Section 5 concludes.

2. SOLUTION ALGORITHM

We analyse a system of equilibrium conditions of the following form:

$$E_t f(y_{t+1}, y_t, y_{t-1}, A_t, A_{t-1}, \varepsilon_{t+1}, \varepsilon_t, e_{t+1}, e_t) = 0_{(n_y+n_A)\times 1} \quad (1)$$

The last n_A equations are:

$$A_t = A_{t-1} + e_t \quad (2)$$

where A_t is an $n_A \times 1$ vector of stochastic trends, y_t is an $n_y \times 1$ vector of endogenous and exogenous variables except for trends, e_t is an $n_A \times 1$ vector of trend shocks and ε_t is an $n_\varepsilon \times 1$ vector of other shocks. We assume that e_t and ε_t are i.i.d. with a finite diagonal covariance matrix.

We also assume continuous differentiability of f, full rank of $f_1' + f_2' + f_3'$ and the existence of the unique and bounded steady state $y_{ss}(A_t)$ for each A_t in the area of interest:

$$f_{1:n_y}(y_{ss}, y_{ss}, y_{ss}, A_t, A_t, 0, 0, 0, 0) = 0_{n_y\times 1} \quad (3)$$

Our approximation algorithm is an adaptation of the perturbation methods described by Kim et al. (2008) and Lan and Meyer-Gohde (2014). The authors also discuss the motivation for the approximation of non-stationary models and fitting models to data. We will try to find the solution of the form[2]:

$$y_t = g(y_{t-1}, A_t, \varepsilon_t, e_t) \quad (4)$$

The approximation algorithm can be divided into two steps. In the first system (3) is solved for various values of A_t. The second step involves the approximation about trends and adjustment to trend changes.

Step 1

This is a standard procedure to solve system (3) for a fixed value of A_t. In the presence of a balanced growth path, a change of variable $\hat{y}_t = \frac{y_t}{F(A_t)}$ (where $F(A_t)$ is a function depending on the values of the trends) transforms the system into stationary representation. If possible, we assume preliminary replacement for a subset of A_t (excluding A_t from the system) to accelerate our procedure.

After that, system (3) is solved for a fixed A_t or dependence between y_{ss} and A_t is approximated by one of the methods described in Section 3. Thus, after this step there is a solution for some values of trends. Further, we will show that the methods of approximation offered in this paper require solution only in a number of points, which is proportional to the data length.

Step 2

In step 2 we linearize (1) except for (2) about $y_{ss}(A_t)$ for some values of A_t. The approximation has the following form[3]:

$$y_t - y_{ss}(A_t) = g_1'(A_t)\big(y_{t-1} - y_{ss}(A_t)\big) + g_3'(A_t)\varepsilon_t + g_4'(A_t)e_t \quad (5)$$

Adjusting to the change in $y_{ss}(A_t)$ and denoting $y_t - y_{ss}(A_t)$ by $\hat{y}_t$, we have that:

$$\hat{y}_t = g_1'(A_t)\big(\hat{y}_{t-1} + y_{ss}(A_{t-1}) - y_{ss}(A_t)\big) + g_3'(A_t)\varepsilon_t + g_4'(A_t)e_t \quad (6)$$

and $y_{ss}(A_{t-1}) - y_{ss}(A_t)$ is the output of the first step.

[2] For simplicity we assume that for each y_{ss} in area of interest system (1) has unique no bubble solution. For more details on non-unique solutions, see Lubik and Schorfheide (2003) and Ascari et al. (2016).
[3] See Appendix A

3. SOME ESTIMATION ALGORITHMS

The measurement equation can be expressed as:

$$y_t^{obs} = c_0(A_t, A_{t-1}, \dots) + c_1(A_t, A_{t-1}, \dots)y_t + e_t^{obs} \quad (7)$$

where y_t^{obs} is a vector of observables, $c_0(A_t, A_{t-1}, \dots)$ and $c_1(A_t, A_{t-1}, \dots)$ are coefficients and e_t^{obs} is a vector of measurement errors.

System (6)-(7) can be estimated using particle filter for likelihood calculation. It is also easy to see that, for given values of the trend, the system is linear. Under the error normality assumption, for example, the system can be estimated using conditional particle filter.[4] In fact, when the particle filter is applied, it is necessary to approximate the solution in NT points (where N is a number of particles and T is a number of observed periods) for likelihood calculation. In the case of Bayesian estimation, using estimates of the likelihood function and sampling techniques such as MH (Metropolis–Hastings) and SMC (Sequential Monte Carlo), it is possible to obtain the posterior distribution of the model parameters.[5,6,7] When a maximum likelihood estimator is the goal, the problem with the dependence of likelihood approximation on random variables arises. To avoid this problem, the algorithm with continuous resampling of Malik and Pitt (2011) can be used.

This paper is devoted to model solution, so we will not estimate models; however, we provide another sampling algorithm in addition to those described above. We hope that this algorithm will be useful for models with stochastic trends.

1. Choose initial point θ_0, A^0 for model parameters and trends
2. For $n = 1, \dots, N$:

   2.a. For $t = 1, \dots, T$:

[4] For more information about the particle filter and conditional particle filter, see, for instance, Doucet and Johansen (2011) and Herbst and Schorfheide (2015).

[5] It should be noted that using estimates of the likelihood function instead of the true values can lead to sampling points that are not from the posterior distribution. The validity of MH and SMC algorithms is provided by Andrieu et al. (2010) and Chopin et al. (2012).

[6] Another possible algorithm is IS with accurately constructed proposal density. For example, posterior of linear approximation can be used to construct proposal. Calculations related to nonlinear model can be done in parallel way.

[7] A number of improvements to standard algorithms have been used for estimation and likelihood evaluation. IS^2 algorithm for state space model estimation is proposed by Tran et al. (2016). In addition, the choice of the optimal number of particles for time reduction is discussed. Doucet et al. (2015) can be seen as an example of the papers devoted to the choice of the optimal number of particles in the context of PMMH. The addition of correlated states can also improve the properties of algorithms (see Dahlin et al. (2015) and Deligiannidis et al. (2016)). Algorithms like the auxiliary particle filter (Doucet and Johansen (2011)), tempered particle filter (Herbst and Schorfheide (2016)) under the condition that likelihood is unbiased and particle EIS (Scharth and Kohn (2016)) can be applied to reduce the likelihood estimation variability. To improve the proposal density properties, a number of techniques exist (see Giordani and Kohn (2010), Hoogerheide et al. (2012), Herbst and Schorfheide (2015)). There are also several extensions of the standard SMC (see Herbst and Schorfheide (2015)), for example SQMC (Gerber and Chopin (2014)).

Sample A_t^n from $p(A_t^n|A_{-t}, \theta_{n-1}, y_{1:T})$ using MH algorithm[8]

2.b. Sample θ_n from $p(\theta_n|A_{1:T}^n, y_{1:T})$ using MH algorithm

The remainder of the paper is devoted to the comparison of different solution algorithms. The key characteristics are speed and accuracy of solution approximation.

Before we move on to the evaluation of the properties of algorithms, we note that, in the case of the observed trends, the likelihood function can be found exactly.[9] In this case the likelihood function can be factorized as follows:

$$p(Y_{1:T}|\theta) = p(Y_1|\theta)\prod_{t=2}^{T} p(Y_t|Y_{1:t-1}, \theta)$$

$$= p(Y_1^{-A}|\theta, A_1)p(A_1|\theta)\prod_{t=2}^{T} p(Y_t^{-A}|Y_{1:t-1}, A_t, \theta)p(A_t|Y_{1:t-1}, \theta)$$

$$= \left(p(Y_1^A|\theta)\prod_{t=2}^{T} p(Y_t^A|Y_{1:t-1}, \theta)\right)\left(p(Y_1^{-A}|\theta, Y_1^A)\prod_{t=2}^{T} p(Y_t^{-A}|Y_{1:t-1}, Y_t^A, \theta)\right)$$

The first multiplier reflects the likelihood of trends and can be calculated easily, taking into account the fact that the trends are observable. The second factor can easily be calculated using Kalman filter (under the assumption of normally distributed errors).

4. ALGORITHM IMPLEMENTATIONS

For the comparison of the algorithms, we will use the model described in appendix B. It is a modified model without the balanced growth path of Evans and Phillips (2015). We add an endowment good. The production of this good follows a random walk. The model calibration is provided in Table 1.

4.1. Solution algorithms

The implementation of the algorithm is divided into two parts: the approximation of the trend and the cyclical part. For the approximation of the trend part, the following algorithms will be used: the grid search, solution in a finite number of points, linear approximation, Smolyak's algorithm and the grid search using differential equations.[10]

[8] Modification with additional state sampling can be used.
[9] The model in which the price of oil follows a random walk is an example of such a type of model.
[10] See Appendix C

The grid search is a basic algorithm, although the computing complexity of this algorithm grows exponentially with the number of trends. Smolyak's algorithm works on a sparse grid and partially solves this problem, but its complexity also increases with the number of trends. Linear approximation and the solution in a finite number of points are more robust in this context. The first of these, however, can be inaccurate in problems with strong nonlinearities, and the complexity of the second grows with the number of approximation points (usually the number of approximation points is proportional to the number of observed periods). The algorithm using differential equations, in the form in which it is applied in this paper, suffers from "curse of dimensionality" too, but using this algorithm can lead to a considerable reduction in the number of numerical solutions of the steady-state system.

For the approximation of the cyclical part, we use the same algorithms except for the grid search using differential equations, and we add constant approximation.

4.1.1. Trend

SMC and MH algorithms require a solution in the NT point to calculate the likelihood, and the proposed two-step algorithm requires a solution in 2T points for one iteration, so we compare the solution algorithms in a finite number of points with the others for NT and 2T points.

We use MATLAB for the realization of the algorithms,[11] and at the same time we do not use parallel programming for the algorithms that can be applied to the majority of them. For the solution in point, the *fsolve*[12] function is used. We obtain an initial guess from the decision in the previous point, if it is possible. A more detailed implementation description can be found in appendix C.

We use 99 points for the grid search with a step equal to the standard deviation of trend logarithms. As a result, we obtain a solution in $[logA_0 - 49\sigma_1; logA_0 + 49\sigma_1] \times [logd_0 - 49\sigma_2; logd_0 + 49\sigma_2]$. We use the same grid (99×99) for the approximation of trends in differential equations and a 9×9 grid with the same boundaries.[13] The same area is used for Smolyak's algorithm. For speed comparison in 2T points, we solve twice the model with T=100.

We solve models following equations (B.7.ss)–(B.13.ss). Numerically we solve only equation (B.10.ss), so we compare the accuracy of this equation. Two measures of accuracy are

[11] We use a desktop PC with the following characteristics: Intel(R) Core(TM) i5-4210U CPU @ 1.70 GHz 2.40 GHz and RAM 4 GB.
[12] The accuracy is equal to 10^{-6}, which partially determines the accuracy of the algorithms that use this function.
[13] We choose the bilinear function to interpolate the trends inside a rectangle.

used: the difference between the logarithm of the numerical solution and approximation; and the error in (B. 10.ss) divided by the sum of the modules of all the terms.[14] For comparison, the values are calculated on a 999×999 grid.

The results are presented in table 2. The grid search takes about 30 seconds and is the slowest algorithm (except for the approximation in a finite number of points for NT points), because it is necessary to solve equation (B.10.ss) many times. The running time of all the other algorithms is less than two seconds. The linear approximation with a numerical derivative solves equation (B.10.ss) three times; however, it can easily be modified to an algorithm with the use of differential equations (the tangent plane in the initial point) by finding the derivative analytically. The presented linear approximation takes from 0.02 to 0.11 sec depending on the number of points.[15] The running time of the grid search using differential equations for 200 points is 0.01 sec for the 9×9 grid and 0.27 sec for the 99×99 grid. The same times for 100,000 points are 1.37 and 1.62 sec. Approximation in a finite number of points also requires equation (B.10.ss) to be solved many times. For 200 points the solution takes 0.61 sec, and for 100,000 it takes 305 sec. Smolyak's algorithm works in 0.11 and 1.29 sec for 200 and 100,000 points with 29 nodes. The same times for Smolyak's algorithm with the use of 13(5) points are 0.05(0.02) and 1.2(1.15) sec.

It is natural that in most cases the price of speed is accuracy. The algorithm of approximation in a finite number of points is the most accurate. It differs from true because of numerical solver tolerance; therefore, we do not give the accuracy values of this algorithm. In a case in which the size of the grid exceeds the number of approximation points, the grid search is less precise but takes more time (as in an example with 200 points). The grid search is the second of the presented algorithms for accuracy. The maximal error of logarithm deviation from a steady state is about 0.0001. For the algorithm using differential equations on the same grid, the error is about 0.025, and for Smolyak's algorithm with 29 points, it is 0.04. The other algorithms are less precise. At the same time, the average errors for the presented algorithms are more than a half-order less.

4.1.2. Cycle

We use several modifications of standard linearization about the steady state described in appendix C. For linearization we use *gensys* described by Sims (2002). As well as for the approximation of a trend, the same grid is used. We do not demonstrate the accuracy of a cycle

[14] We will see later that the accuracy of (B.3.ss) depends on these errors.
[15] 0.01 in the case of an analytical derivative

because it is difficult to divide a cycle and a trend. Instead, we present the accuracy of the full solution for various combinations of trend and cycle approximations.

The running times of various cycle approximations are presented in table 3.[16] All the steady states presented in system (B.1.ss)–(B.7.ss) are preliminarily computed.

As well as for algorithms of trend approximation, the grid search takes the most time (9 and 12 sec for 200 and 100,000 points). Approximation in a finite number of points for 100,000 points takes more time since the number of system (B.1.lin)–(B.7.lin) solutions is far larger than that for the grid search algorithm. The running time is 0.19 sec for 200 points. Smolyak's algorithm with the use of 5, 13 and 29 nodes for approximation in 200 points takes 0.006, 0.02 and 0.03 sec, respectively, and 0.5, 0.56 and 0.6 sec for 100,000. The running times of linear approximation are 0.004 and 0.04 sec.

4.1.3. Full solution

We combine identical approximation algorithms of the cyclic and trend part to compare the accuracy of the full solution. Other combinations are also possible; however, we do not consider them in this work. For an accuracy comparison of the full solution, 1000 random points were generated for trend values in the area of interest. In addition, given that $A_{ss} = 1$ and $d_{ss} = 1$, 100 points were generated. In these 100,000 points, we estimate the accuracy of the algorithms, calculating the logarithm of unit-free error[17] for (B.1)–(B.6). We follow Tauchen (1986) in numerical integration.

We run this procedure for two sets of parameters: $\sigma_\varepsilon = 0.01$ and $\sigma_\varepsilon = 1$. In the first case, the influence of the cyclic shock is small relative to the trend shocks; in the second, it is dominating.

The results are presented in table 4. It is simple to notice that equations (B.4)–(B.6) are well approximated by all the algorithms. These equations hold for the approximation of trends ((B.9.ss), (B.11.ss) and (B.13.ss) set the equivalent system of equations) and for the approximation of a cycle ((B.4.lin)–(B.6.lin)). (B.3) has the same form, but its approximation is less precise. Such a situation arises due to the offered approximation method of system (B.1.ss)–(B.6.ss). Let us denote the approximated solution for labour as l_t^{*ss} and rewrite equation (B.10.ss):

$$\left(\chi \frac{(l_t^{*ss})^\theta}{w_t^{*ss}}\right)^{-\frac{1}{\gamma}} + l_t^{*ss}(\delta - r_t^{*ss})\left(\frac{\alpha A_t}{r_t^{*ss}}\right)^{\frac{1}{1-\alpha}} = w_t^{*ss} l_t^{*ss} + d_t + err$$

[16] The description can be found in appendix C.
[17] We divide the error by the sum of absolute values of all the terms.

where err is an approximation error. Substitute (B. 8.ss) and (B. 12.ss):

$$\left(\chi\frac{(l_t^{*ss})^{\theta}}{w_t^{*ss}}\right)^{-\frac{1}{\gamma}} = c_t^{*ss} + err$$

or

$$\chi(l_t^{*ss})^{\theta} = w_t^{*ss}(c_t^{*ss} + err)^{-\gamma}$$

Thus, the error of (B.3) depends on the error of (B.10.ss). It is easy to note that this error is weakly dependent on the standard deviation of the cyclical shock. For Smolyak's algorithm and the linear approximation, the error in this equation is large enough; however, for algorithms using a grid, the accuracy increases with the addition of new nodes. That gives the ability to increase the accuracy by adding nodes.[18] Equations (B.1)–(B.2) after log-linearization are not equivalent to the initial. The errors in (B.1)–(B.2) depend more on the dispersion of the cyclical shock.

As well as for trends, the approximation of the full solution shows that the solution in a finite number of points is the most precise algorithm. The grid search is the second most accurate algorithm. Smolyak's algorithm is less precise. The linear algorithm approximates poorly and cannot be used for the solution of the presented model.

Finally, we test the algorithm of linearization about one point as in Kim et al. (2008). For this purpose we solve the system for trends in a point $A_{ss} = 1$, $d_{ss} = 1$. Then, using system (B.1.dif)–(B.6.dif), we find the tangent plane (in terms of logarithms) in this point. Adding the constant approximation of a cycle, we obtain the full solution. Unlike linear approximation, not only the labour equation but also all the other equations are approximated. Consequently, there is no problem in (B.3) described above. The largest error for this method appears in equation (B.1), which depends on steady-state values. The logarithm of the maximal error is equal to -0.34 and -0.33 for $\sigma_\varepsilon = 0.01$ and $\sigma_\varepsilon = 1$, respectively. The average errors are equal to -1.93 and -1.77. The errors in the other equations are compatible, allowing the use of this method, in view of its simplicity and speed, when the fluctuations are small.

4.2. Algorithm discussion

4.2.1. Problems and limitations

[18] See the discussion of parallelization below.

As well as the majority of perturbation methods, the algorithm offered in this paper works well only for models without kinks. At the same time, if it is possible to speed up, it is better to remove trends that can be excluded from the system by a change of variables. In models with weak nonlinearities in trends, most of the implementations do not work much better than simple linearization. Before proceeding to the algorithms that take more time, it is probably worth trying to solve the model using the method of Kim et al. (2008) and looking at the arising nonlinearities.

There are also a number of problems related to the identification, which may arise in the estimation step. Let us explain this with the model described in this paper. Suppose that we observe y_t, c_t and r_t. Let $\chi' = 2\chi$, then $\frac{w_t'}{l_t'^{\theta}} = 2\frac{w_t}{l_t^{\theta}}$. Bearing in mind that $w_t'\, l_t' = w_t l_t$, we obtain expressions for labour and wages. Adjusting A_t from equation $A_t' l_t'^{1-\alpha} = A_t l_t^{1-\alpha}$, we have full equivalence of parameters χ' and χ. In models with a balanced growth path, such non-identifiability may disappear as a result of scaling and log-linearization. A number of standard methods are available to avoid the problem of identification, which we do not discuss here.

4.2.2. Increasing the speed and accuracy

The implementation of the proposed two-step procedure can be improved in several ways, some of which we discuss in this section.

Parallel computation (using CPU or GPU) can be applied to accelerate the majority of the offered implementations.[19] Grid algorithms (grid search or sparse grid search) can easily be parallelized in nodes for calculation of the required function. Thus, the accuracy can be improved by increasing the number of nodes with a smaller loss of time. It is also necessary to take into account the time that we can benefit from parallelization and the time that we lose by using parallel computing. Parallelizing the algorithm of the solution in a finite number of points is more difficult. Using particle filters, we do not know in advance the points at which we need to make an approximation. We can use parallelization on particles. Applying the algorithm with alternate sampling trends and parameters, it is possible to use the prefetching MH algorithm[20] (see. Strid (2009)) to step 2.a. Step 2.b. in this algorithm can be parallelized, since we know the values of the

[19] See Brumm and Scheidegger (2015) as an example of the use of parallel computing for solving DSGE models.
[20] It can also be applied to MH step in algorithms with particle filters.

trends. In addition, MH with sampling of trends that does not depend on their values of trends in the current iteration in step 2.a can be used.[21] It allows the solution to be found in advance.

An increase in accuracy can be achieved by a higher approximation order. Quadratic approximation can be calculated easily in the presence of the linear[22]. The accuracy of the grid search can be improved by increasing the number of nodes. Furthermore, potentially, a different method of selecting grid points can help to increase the accuracy. Let us explain this using Smolyak's algorithm with five points. Figure V1.a shows the nodal points. With such collocation, the greatest errors arise in the corners. When placing nodal points in corners, the errors decrease.

5. CONCLUSION

In this paper a two-step algorithm for solving DSGE models without a balanced growth path was presented. A number of proposed implementations allow the controlling of the accuracy of approximation in a certain range and can easily be parallelized. This makes it possible to solve and estimate a large class of models. We hope that this algorithm will be useful in a large number of applied research areas, including those devoted to developing economies, where many ratios that are stable for developed economies change significantly, making their analysis impossible with the use of standard techniques.

[21] The use of simple techniques can strongly reduce the effectiveness of the algorithm, and the use of adaptive algorithms requires the proof of their validity (see Roberts and Rosenthal (2007, 2009)). See also Giordani and Kohn (2010) and Hoogerheide et al. (2012) for algorithms of proposal distribution construction.

[22] See, for instance, Shmitt-Grohe and Uribe (2004) or Kim et al. (2008).

REFERENCES

1. Andrieu, C., Doucet, A., Holenstein, R. Particle Markov Chain Monte Carlo Methods // Journal of the Royal Statistical Society Series. 2010. Vol. 72(3). Pp. 269–342.
2. Aruoba, B., Cuba-Borda, P., Schorfheide, F. Macroeconomic Dynamics Near the ZLB: A Tale of Two Countries // Manuscript. 2016.
3. Ascari, G., Bonomolo, P., Lopes, H. Rational Sunspots // Manuscript. 2016.
4. Brumm, J., Scheidegger, S. Using Adaptive Sparse Grids to Solve High Dimensional Dynamic Models // University of Zurich. 2015.
5. Canova, F. Bridging Cyclical DSGE Models and the Raw Data // Journal of Monetary Economics. 2014. Vol. 67. Pp. 1–15.
6. Chopin, N., Jacob, P., Papaspiliopoulos, O. SMC^2: An Efficient Algorithm for Sequential Analysis of State-Space Models // 2012. arXiv:1101.1528.
7. Dahlin, J., Lindsten, F., Kronander, J., Schön, T. Accelerating Pseudo-Marginal Metropolis–Hastings by Correlating Auxiliary Variables // 2015. arXiv:1511.05483.
8. Deligiannidis, G., Doucet, A., Pitt, M. The Correlated Pseudo-Marginal Method // 2016. arXiv:1511.04992v3.
9. Doucet, A., Johansen, A. A Tutorial on Particle Filtering and Smoothing: Fifteen Years Later // Handbook of Nonlinear Filtering. Oxford University Press. 2011.
10. Doucet, A., Pitt, M., Deligiannidis, G., Kohn, R. Efficient Implementation of Markov Chain Monte Carlo When Using an Unbiased Likelihood Estimator // Biometrika. 2015. Vol. 102(2). Pp. 295–313.
11. Evans, R., Phillips, K. Linearization about the Current State: A Computational Method for Approximating Nonlinear Policy Functions during Simulation // BYU Macroeconomics and Computational Laboratory Working Paper Series. 2015.
12. Fernandez-Villaverde, J., Rubio-Ramirez, J. Estimating Macroeconomic Models: A Likelihood Approach // Review of Economic Studies. 2007. Vol. 74(4). Pp. 1059–1087.
13. Fernández-Villaverde, J., Rubio-Ramirez, J., Schorfheide, F. Solution and Estimation Methods for DSGE Models // Handbook of Macroeconomics. 2016. Vol. 2. Preliminary draft.
14. Gerber, M., Chopin, N.. Sequential Quasi-Monte Carlo // 2014. arXiv:1402.4039v5.
15. Giordani, P., Kohn, R. Adaptive Independent Metropolis–Hastings by Fast Estimation of Mixtures of Normals // Journal of Computational and Graphical Statistics. 2010. Vol. 16. Pp. 243–259.

16. Gorodnichenko, Y., Ng, S. Estimation of DSGE Models When the Data are Persistent // Journal of Monetary Economics. 2010. Vol. 57(3). Pp. 325–340.
17. Herbst, E., Schorfheide, F. Bayesian Estimation of DSGE Models // Princeton University Press. 2015.
18. Herbst, E., Schorfheide, F. Tempered Particle Filtering // Manuscript. 2016.
19. Hoogerheide, L., Opschoor, A., van Dijk, H. A Cclass of Adaptive Importance Sampling Weighted EM Algorithms for Efficient and Robust Posterior and Predictive Simulation // Journal of Econometrics. 2012. Vol. 171(2). Pp. 101–120.
20. Judd, K., Maliar, L., Maliar, S., Valero, R. Smolyak Method for Solving Dynamic Economic Models: Lagrange Interpolation, Anisotropic Grid and Adaptive Domain // Journal of Economic Dynamics and Control. 2014. Vol. 44. Pp. 92–123.
21. Kim, J., Kim, H., Schaumburg, E., Sims, C. Calculating and Using Second-Order Accurate Solutions of Discrete Time Dynamic Equilibrium Models // Journal of Economic Dynamics and Control. 2008. Vol. 32, Pp. 3397–3414.
22. Kulish, M., Pagan, A. Estimation and Solution of Models with Expectations and Structural Changes // Dynare Working Papers. 2014, No. 34.
23. Lan, H., Meyer-Gohde, A. Solvability of Perturbation Solutions in DSGE Models // Journal of Economic Dynamics and Control. 2014. Vol. 45. Pp. 366–388.
24. Lubik, T., Schorfheide, F. Computing Sunspot Equilibria in Linear Rational Expectations Models // Journal of Economic Dynamics and Control. 2003. Vol. 28(2). Pp. 273–285.
25. Malik, S., Pitt, M. Particle Filters for Continuous Likelihood Evaluation and Maximization // Journal of Econometrics. 2011. Vol. 165. Pp. 190–209.
26. Roberts, G., Rosenthal, J. Coupling and Ergodicity of Adaptive MCMC // Journal of Applied Probability. 2007. Vol. 44. Pp. 458–475.
27. Roberts, G., Rosenthal, J. Examples of Adaptive MCMC // Journal of Computational and Graphical Statistics. 2009. Vol. 18. Pp. 349–367.
28. Scharth, M., Kohn, R. Particle Efficient Importance Sampling // Journal of Econometrics. 2016. Vol. 190(1). Pp. 133–147.
29. Schmitt-Grohe, S., Uribe, M. Solving Dynamic General Equilibrium Models Using a Second-Order Approximation to the Policy Function // Journal of Economic Dynamics and Control. 2004. Vol. 28. Pp. 755–775.
30. Sims, C. Solving Linear Rational Expectations Models // Computational Economics. Society for Computational Economics, 2002. Vol. 20 (1–2). Pp. 1–20.

31. Smolyak, S. Quadrature and Interpolation Formulas for Tensor Products of Certain Classes of Functions // Soviet Mathematics. 1963. Vol. 4. Pp. 240–243.
32. Strid, I. Efficient Parallelization of Metropolis–Hastings Algorithms Using a Prefetching Approach // Computational Statistics and Data Analysis. 2009.
33. Tauchen, G. Finite State Markov-Chain Approximations to Univariate and Vector Autoregressions // Economics Letters. 1986. Vol. 20, Pp. 177–181.
34. Tran, M., Scharth, M., Pitt M., Kohn, R. Importance Sampling Squared for Bayesian Inference and Model Choice with Estimated Likelihoods // 2016. arXiv:1309.3339v4

APPENDIX A

Substituting (4) in (1), we receive

$$\int f(g(g(y_{t-1}, A_t, \varepsilon_t, e_t, \chi), A_t + \chi e_{t+1}, \chi\varepsilon_{t+1}, \chi e_{t+1}, \chi), g(y_{t-1}, A_t, \varepsilon_t, e_t, \chi), y_{t-1}, A_t, A_t - e_t, \chi\varepsilon_{t+1}, \varepsilon_t, \chi e_{t+1}, e_t)\, \mu(d\varepsilon_{t+1}, de_{t+1}) = 0_{n_y \times 1}$$

or first-order approximation

$$\int (f_1'(g_1'(g_1' y_{t-1} + g_3'\varepsilon_t + g_4' e_t + g_5'\chi) + g_2'\chi e_{t+1} + g_3'\chi\varepsilon_{t+1} + g_4'\chi e_{t+1} + g_5'\chi) + f_2'(g_1' y_{t-1} + g_3'\varepsilon_t + g_4' e_t + g_5'\chi) + f_3' y_{t-1} - f_5' e_t + f_6'\chi\varepsilon_{t+1} + f_7'\varepsilon_t + f_8'\chi e_{t+1} + f_9' e_t)\, \mu(d\varepsilon_{t+1}, de_{t+1}) = 0_{n_y \times 1}$$

We collect coefficients under the same terms:

y_{t-1}: $f_1'(g_1')^2 + f_2' g_1' + f_3' = 0$

ε_t: $f_1' g_1' g_3' + f_2' g_3' + f_7' = 0$

e_t: $f_1' g_1' g_4' + f_2' g_4' - f_5' + f_9' = 0$

χ: $f_1' g_1' g_5' + f_1' g_5' + f_2' g_5' = 0$

Having solved the system of equations relative to g_1' (see for example, Sims (2002)), we substitute this solution in the subsequent equations and receive g_3' and g_4'. It is also easy to see that $g_5' = 0$.

APPENDIX B

A household maximizes its utility function given by:

$$U_t = \sum_{i=0}^{\infty} \beta^i \left(\frac{c_{t+i}^{1-\gamma} - 1}{1-\gamma} - \chi_t \frac{l_{t+i}^{1+\theta}}{1+\theta} \right)$$

subject to an intertemporal budget constraint:

$$c_t + k_t(1 - r_t) = w_t l_t + (1-\delta)k_{t-1} + d_t \quad \text{(B.1)}$$

First-order conditions:

$$\beta \frac{1-\delta}{1-r_t} E_t \left(\frac{c_{t+1}}{c_t} \right)^{-\gamma} = 1 \quad \text{(B.2)}$$

$$w_t c_t^{-\gamma} = \chi_t l_t^{\theta} \quad \text{(B.3)}$$

The production function has the following form:

$$y_t = A_t k_t^{\alpha} l_t^{1-\alpha} \quad \text{(B.4)}$$

First-order conditions:

$$r_t = \alpha \frac{y_t}{k_t} \quad \text{(B.5)}$$

$$w_t = (1-\alpha) \frac{y_t}{l_t} \quad \text{(B.6)}$$

Exogenous processes:

$$logA_t = logA_{t-1} + e_t^A \quad \text{(B.7)}$$

$$logd_t = logd_{t-1} + e_t^d \quad \text{(B.8)}$$

$$log\chi_t = log\chi + z_t^{\chi} \quad \text{(B.9)}$$

$$z_t^{\chi} = \rho * z_{t-1}^{\chi} + \varepsilon_t \quad \text{(B.10)}$$

B.1. Steady-state system

The steady-state system can be written in the following form:

$$c_t^{ss} + k_t^{ss}(\delta - r_t^{ss}) = w_t^{ss} l_t^{ss} + d_t \quad \text{(B.1.ss)}$$

$$\beta \frac{1-\delta}{1-r_t^{ss}} = 1 \quad \text{(B.2.ss)}$$

$$w_t^{ss} (c_t^{ss})^{-\gamma} = \chi (l_t^{ss})^{\theta} \quad \text{(B.3.ss)}$$

$$y_t^{ss} = A_t (k_t^{ss})^{\alpha} (l_t^{ss})^{1-\alpha} \quad \text{(B.4.ss)}$$

$$r_t^{ss} = \alpha \frac{y_t^{ss}}{k_t^{ss}} \quad \text{(B.5.ss)}$$

$$w_t^{ss} = (1-\alpha) \frac{y_t^{ss}}{l_t^{ss}} \quad \text{(B.6.ss)}$$

The solution is given by:

$$r_t^{ss} = 1 - \beta(1-\delta) \quad \text{(B.7.ss)}$$

$$\frac{k_t^{ss}}{l_t^{ss}} = \left(\frac{\alpha A_t}{r_t^{ss}}\right)^{\frac{1}{1-\alpha}} \quad \text{(B.8.ss)}$$

$$w_t^{ss} = (1-\alpha)A_t\left(\frac{k_t^{ss}}{l_t^{ss}}\right)^{\alpha} \quad \text{(B.9.ss)}$$

Next, solve the following equation with respect to l_t^{ss}:

$$\left(\chi\frac{(l_t^{ss})^{\theta}}{w_t^{ss}}\right)^{-\frac{1}{\gamma}} + l_t^{ss}(\delta - r_t^{ss})\left(\frac{\alpha A_t}{r_t^{ss}}\right)^{\frac{1}{1-\alpha}} = w_t^{ss}l_t^{ss} + d_t \quad \text{(B.10.ss)}$$

Finally, find k_t^{ss}, c_t^{ss} and y_t^{ss}:

$$k_t^{ss} = \left(\frac{\alpha A_t}{r_t^{ss}}\right)^{\frac{1}{1-\alpha}} l_t^{ss} \quad \text{(B.11.ss)}$$

$$c_t^{ss} = w_t^{ss}l_t^{ss} + d_t - k_t^{ss}(\delta - r_t^{ss}) \quad \text{(B.12.ss)}$$

$$y_t^{ss} = A_t(k_t^{ss})^{\alpha}(l_t^{ss})^{1-\alpha} \quad \text{(B.13.ss)}$$

System of differential equations:

$$c^{ss}c + k^{ss}(\delta - r^{ss})k - k^{ss}r^{ss}r = w^{ss}l^{ss}(w+l) + d^{ss}d \quad \text{(B.1.dif)}$$

$$r = 0 \quad \text{(B.2.dif)}$$

$$w - \gamma c = \theta l \quad \text{(B.3.dif)}$$

$$y = A + \alpha k + (1-\alpha)l \quad \text{(B.4.dif)}$$

$$r = y - k \quad \text{(B.5.dif)}$$

$$w = y - l \quad \text{(B.6.dif)}$$

B.2. Log-linearized system

The log-linearization of the steady-state system has the following form:

$$c_t^{ss}c_t + k_t^{ss}k_t - k_t^{ss}r_t^{ss}(k_t + r_t) = w_t^{ss}l_t^{ss}(w_t + l_t) + (1-\delta)k_t^{ss}k_{t-1} \quad \text{(B.1.lin)}$$

$$\frac{r_t^{ss}}{1-r_t^{ss}}r_t + \gamma(c_t - E_t c_{t+1}) = 0 \quad \text{(B.2.lin)}$$

$$w_t - \gamma c_t = z_t^{\chi} + \theta l_t \quad \text{(B.3.lin)}$$

$$y_t = \alpha k_t + (1-\alpha)l_t \quad \text{(B.4.lin)}$$

$$r_t = y_t - k_t \quad \text{(B.5.lin)}$$

$$w_t = y_t - l_t \quad \text{(B.6.lin)}$$

$$z_t^{\chi} = \rho z_{t-1}^{\chi} + \varepsilon_t \quad \text{(B.7.lin)}$$

APPENDIX C

C.1. Algorithms for the trend part

C.1.1. Grid search

A simple search is conducted when splitting the interesting space (a cube $[-1; 1]^n$ after some transformation), finding the solution in nodes and the approximation inside if it is necessary. For one trend the linear function can be used, for two the bilinear one and so on.

C.1.2. Smolyak's algorithm

Smolyak's algorithm (Smolyak (1963)) is the standard algorithm in a sparse grid.[23] A number of extensions have been offered for Smolyak's algorithm (see, for instance, Judd et al. (2014)); however, for simplicity we use the standard algorithm.

Instead of a simple grid search, the tensor product with a number of restrictions is used. We will determine a set of nodes of order *i* on the *k-th* coordinate (after the linear transformation to interval [-1; 1]) using the following formula:

$$x_j^k = -\cos\left(\frac{j-1}{m_i-1}\pi\right), j = 1, \dots, m_i$$

where $m_1 = 1; m_i = 2^{i-1} + 1$ for $i > 1$.

Instead of the simple grid search, in which the set of nodes is determined by:

$$X_i^1 \times \dots \times X_i^n, X_i^k = \{x_1^k, \dots, x_{m_i}^k\}$$

Smolyak's algorithm determines this set as:

$$\bigcup_{q-n+1 \le \sum i_n \le q} X_{i_1}^1 \times \dots \times X_{i_n}^n$$

In the paper we use grids with $q = 3$, $q = 4$ and $q = 5$ (see figure C1).

[23] Smolyak's algorithm was used as a representative of sparse grid algorithms. Other algorithms in a sparse grid can be used as an approximation for the trend and cycle.

Figure C1a. Smolyak's grid, q = 3

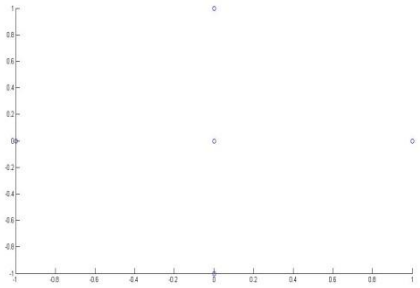


Figure C1b. Smolyak's grid, q = 4

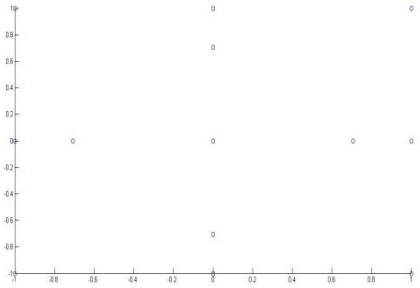
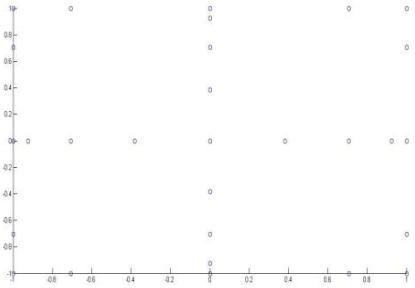


Figure C1c. Smolyak's grid, q = 5



Function $f(x_1, \dots, x_n)$ is approximated by:

$$\tilde{f}(x_1, \dots, x_n, a) = \sum_{\max(n, q-n+1) \le \sum i_n \le q} (-1)^{q-\sum i_n} \binom{n-1}{q-\sum i_n} \varphi^{\sum i_n}(x_1, \dots, x_n, a)$$

where $\varphi^{\sum i_n}(x, a) = \sum_{j_1=1}^{m_{i_1}} \dots \sum_{j_n=1}^{m_{i_n}} a_{j_1,\dots,j_n} T_{j_1-1}(x_1) \dots T_{j_n-1}(x_n)$ and $T_j(x_i)$ are Chebyshev polynomials.

C.1.3. Linear approximation

Here the linear approximation is an approximation of function $f(x_1, \dots, x_n)$ around x^0

$$\tilde{f}(x_1, \dots, x_n, a) = f(x_1^0, \dots, x_n^0) + \sum_{i=1}^{n} a_i (x_i - x_i^0)$$

In other words, the linear approximation is the tangent plane at x^0.

C.1.4. Solution in a finite number of points

As can be seen from equations (6)–(7) and the estimation algorithms, it is necessary to solve the model only in a finite number of points. Applying algorithms based on the particle filter or

alternate sampling of trends and parameters, it is necessary to solve the model for the NT and 2T points.

C.1.5. Grid search using differential equations

.

Under continuous differentiability assumption, system (3) can be written as follows:

$$f_1' dy_{ss} + f_2' dy_{ss} + f_3' dy_{ss} + f_4' dA_{ss} + f_5' dA_{ss} = 0$$

If also matrix $f_1' + f_2' + f_3'$ is not singular:

$$dy_{ss} = (f_1' + f_2' + f_3')^{-1}(f_4' + f_5')dA_{ss}$$

Having the solution in one point and extending it by the received system of differential equations, it is possible to find the solution in the interesting area. In this paper, in the example with two trends, firstly we find the solution for one fixed point on the second coordinate (at the point of the exact solution). We fix $A_{ss}^{2,0}$ and find $y_{ss}(A_{ss}^{1,i}, A_{ss}^{2,0})$ for $i = -n, \dots, 0, \dots, n$, moving with a constant step from zero in both directions. Then for $j = -n, \dots, 0, \dots, n$ we calculate $y_{ss}(A_{ss}^{1,i}, A_{ss}^{2,j})$ in the same way for each i. The results of this method depend on an order of trends in vector A_{ss}. This can be avoided by an exhausting search (using all the possible permutations), but it leads to an increase in the running time of the algorithm.

C.2. Algorithms for the trend part

C.2.1. Grid search, Smolyak's algorithm and linear approximation

These algorithms are essentially no different from those presented for trends. The linear solution of type (6) is found. After that, we provide pointwise approximation for $g_1'(A_t)$, $g_3'(A_t)$, $g_4'(A_t)$.

C.2.2. Solution in a finite number of points

The algorithm is similar to the algorithm for trends.

C.2.3. Constant approximation

Matrices $g_1'(A_t)$, $g_3'(A_t)$ and $g_4'(A_t)$ are assumed to be independent of A_t and equal to the linear approximation in one point.

APPENDIX D

Table 1. Model parameters

Parameter	*Value*
δ	0.025
β	0.99
χ	5
α	0.33
θ	2
γ	2.5
σ_ε	0.01
σ_A	0.02
σ_d	0.04
ρ	0.7

Table 2.a. Error in B.10.ss, $\log_{10}$(abs)

	Max	*Mean*
Grid search	-4.2	-4.7
Approximation in a finite number of points	-	-
Linear approximation	-0.2	-1.0
Smolyak's algorithm (29 nodes)	-1.8	-2.7
Smolyak's algorithm (13 nodes)	-1.2	-2.0
Smolyak's algorithm (5 nodes)	-0.4	-1.2
DE 99x99	-1.7	-2.4
DE 9x9	-0.1	-0.8

Table 2.b. Steady-state error for labour, $\log_{10}$(abs)

	Max	*Mean*
Grid search	-3.9	-Inf
Approximation in a finite number of points	-	-
Linear approximation	0.2	-0.9
Smolyak's algorithm (29 nodes)	-1.4	-2.5
Smolyak's algorithm (13 nodes)	-0.9	-1.7
Smolyak's algorithm (5 nodes)	-0.1	-1.0
DE 99x99	-1.6	-2.2
DE 9x9	-0.5	-1.1

Table 2.c. Running time for trend approximation, 2T = 200, NT = 100,000, sec

	2T	*NT*
Grid search	30.22	30.56
Approximation in a finite number of points	0.61	305.48
Linear approximation	0.02	0.11
Smolyak's algorithm (29 nodes)	0.11	1.29
Smolyak's algorithm (13 nodes)	0.06	1.20
Smolyak's algorithm (5 nodes)	0.02	1.15

DE 99x99	0.27	1.62
DE 9x9	0.01	1.37

Table 3. Running time for cycle approximation, 2T = 200, NT = 100,000, sec

	2T	***NT***
Grid search	9.38	12.15
Approximation in a finite number of points	0.19	97.70
Linear approximation	0.004	0.04
Constant approximation	0.001	0.001
Smolyak's algorithm (29 nodes)	0.03	0.60
Smolyak's algorithm (13 nodes)	0.02	0.56
Smolyak's algorithm (5 nodes)	0.006	0.50

Table 4.a. Error of the full solution, $\sigma_\varepsilon = 0.01$, $\log_{10}$(abs)

		(B.1)	***(B.2)***	***(B.3)***	***(B.4)***	***(B.5)***	***(B.6)***
Grid search	***Max***	-3.21	-3.97	-3.75	-15.44	-15.29	-15.37
	Mean	-5.39	-5.57	-4.31	-Inf	-Inf	-Inf
Approximation in a finite number of points	***Max***	-3.21	-3.97	-6.32	-15.44	-15.29	-15.34
	Mean	-5.4	-5.59	-Inf	-Inf	-Inf	-Inf
Linear approximation	***Max***	-0.35	0	-0.02	-14.44	-13.88	-14.57
	Mean	-1.75	-1.06	-0.79	-Inf	-Inf	-Inf
Smolyak's algorithm (29 nodes)	***Max***	-3.18	-3.68	-1.41	-15.46	-15.23	-15.36
	Mean	-5.27	-5.34	-2.3	-Inf	-Inf	-Inf
Smolyak's algorithm (13 nodes)	***Max***	-2.85	-2.92	-0.8	-15.49	-15.29	-15.36
	Mean	-4.63	-4.65	-1.57	-Inf	-Inf	-Inf
Smolyak's algorithm (5 nodes)	***Max***	-2.25	-2.43	-0.11	-15.46	-15.29	-15.36
	Mean	-4.23	-4.31	-0.85	-Inf	-Inf	-Inf

Table 4.b. Error of the full solution, $\sigma_\varepsilon = 1$, $\log_{10}$(abs)

		(B.1)	***(B.2)***	***(B.3)***	***(B.4)***	***(B.5)***	***(B.6)***
Grid search	***Max***	-1.34	-1.76	-3.75	-15.26	-15.03	-15.18
	Mean	-3.22	-3.16	-4.3	-Inf	-Inf	-Inf
Approximation in a finite number of points	***Max***	-1.34	-1.77	-6.34	-15.14	-15.03	-15.1
	Mean	-3.22	-3.16	-Inf	-Inf	-Inf	-Inf
Linear approximation	***Max***	0	0	-0.02	-12.82	-12.52	-12.87
	Mean	-0.13	-0.07	-0.8	-Inf	-Inf	-Inf
Smolyak's algorithm (29 nodes)	***Max***	-1.33	-1.73	-1.4	-14.94	-14.82	-14.88
	Mean	-3.22	-3.18	-2.33	-Inf	-Inf	-Inf
Smolyak's algorithm (13 nodes)	***Max***	-1.19	-1.5	-0.8	-15.19	-15	-14.97
	Mean	-3.08	-3.16	-1.57	-Inf	-Inf	-Inf
Smolyak's algorithm (5 nodes)	***Max***	-1.28	-1.53	-0.11	-15.22	-15.15	-15.12
	Mean	-2.98	-3.18	-0.85	-Inf	-Inf	-Inf