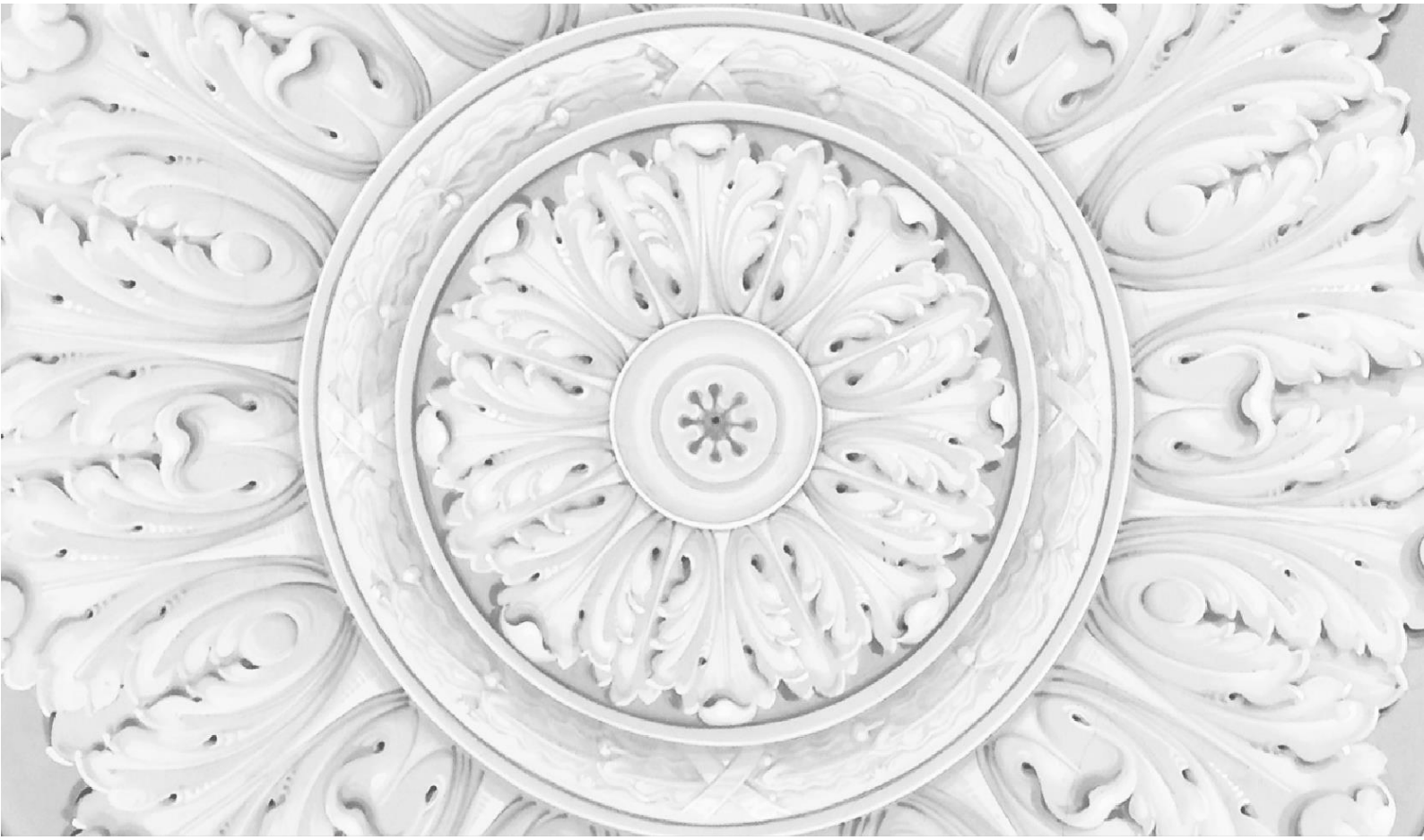
Банк России
Центральный банк Российской Федерации



СЕРИЯ ДОКЛАДОВ ОБ ЭКОНОМИЧЕСКИХ ИССЛЕДОВАНИЯХ

Сергей Селезнев

Решение DSGE-моделей со стохастическими трендами

№ 15 / Сентябрь 2016 г.

Сергей Селезнев
E-mail: SeleznevSM@cbr.ru

Автор выражает благодарность Гафарову Б., Иващенко С., Крепцеву Д., Кулагину Н., Молякову П., Полбину А., Пономаренко А. и Слободяну С. за помощь в проведении исследования и полезные комментарии. Все ошибки, которые могут содержаться в данной работе, являются сферой ответственности автора.

Адрес 107016, Москва, ул. Неглинная, 12
Телефоны +7 495 771-91-00, +7 495 621-64-65 (факс)
Сайт www.cbr.ru

Резюме

В данной работе мы предлагаем алгоритм аппроксимации DSGE модели около стохастических трендов. Ряд реализаций помогают относительно быстро решать модели с небольшим количеством стохастических трендов при отсутствии траектории сбалансированного роста, а также позволяют контролировать точность аппроксимации в некотором диапазоне. С учетом того, что многие из имплементаций могут быть легко реализованы с использованием параллельных вычислений, данный алгоритм дает возможность оценки моделей без траектории сбалансированного роста. Мы также предлагаем некоторые из возможных методов оценки. Предложенный алгоритм позволяет решать и оценивать множество моделей, оценка которых стандартными пертурбационными методами занимает огромное количество времени или обладает плохой точностью ввиду неаккуратной аппроксимации решения. В контексте российской экономики данный алгоритм может быть использован для моделирования изменения долгосрочного уровня реальной цены на нефть.

Ключевые слова: нестационарные DSGE, стохастические тренды, алгоритм Смоляка.
JEL классификация: C61, C63

Оглавление

Введение

В последние десятилетия DSGE-модели приобрели огромную популярность среди макроэкономистов. При наличии должной теоретической интерпретации они могут быть соотнесены с данными и применены, например, для получения прогнозов или анализа оптимальных правил политики центрального банка. Для работы с моделью и многих методов оценки необходимо сначала решить модель, и существует обширный пласт литературы по решению и оценке DSGE-моделей (см., например обзор *Fernandez-Villaverde et al. (2016)*). Однако методам решения нестационарных моделей посвящено не так много работ, в то время как часть наблюдаемых данных нестационарные по своей природе (ВВП, инвестиции, потребление, …). Стандартными являются модели, которые некоторым преобразованием переменных могут быть сведены к стационарным, как например, *Fernandez-Villaverde and Rubio-Ramirez (2007)*, но такой класс упускает множество интересных моделей. Развитие также получили модели, где динамика трендов определяется заранее, и формируются некие ожидания о динамике этих трендов. Такие модели обычно решаются с использованием обратной рекурсии, как например, в *Kulish and Pagan (2014).*

Иногда для соотнесения данных и модели используются временные ряды, предварительно очищенные от трендов[1] в других моделях или с помощью различных фильтров. Но такой подход может приводить к смещенным оценкам параметров[2]. Также используется гибридный подход, описанный в *Canova (2014),* при котором тренд и цикл моделируются отдельно, но оцениваются одновременно. Такой метод может приводить к потере теоретической интерпретации трендов и их несогласованности с микрообоснованными моделями.

Для стохастических трендов (unit root process) в ряде работ были предложены алгоритмы аппроксимации решения. В *Kim et al. (2008)* и *Lan and Meyer-Gohde (2014)* используется пертурбационный алгоритм для аппроксимации решения, основанной на разложении в ряд Тейлора относительно какой-либо единственной точки. В *Evans and Phillips (2015)* предложен алгоритм линеаризации около текущего состояния. Этот алгоритм применим в большей степени для симуляций и достаточно сложен вычислительно для оценки в том виде, в котором он представлен в исходной статье.

[1] Часто DSGE-модели используются для анализа циклических колебаний, а сырые данные включают в себя еще и тренды.
[2] См. *Gorodnichenko and Ng (2010).*

В данной работе мы предлагаем алгоритм для решения моделей со стохастическими трендами, который, как и алгоритмы из *Kim et al. (2008)*, *Lan and Meyer-Gohde (2014)* и *Evans and Phillips (2015)*, не предполагает наличие сбалансированной траектории роста. При этом мы обобщаем алгоритмы из *Kim et al. (2008)* и *Lan and Meyer-Gohde (2014)* и добавляем элементы линеаризации около текущего состояния. В отличие от используемых ранее, наш алгоритм является более гибким (для большинства реализаций) и позволяет контролировать точность в некоторых пределах. Также мы обращаем большое внимание на время работы алгоритма ввиду того, что при оценке приходится решать модель десятки или даже сотни тысяч раз.

Предлагаемый алгоритм состоит из двух шагов. На первом шаге аппроксимируется зависимость стационарного состояния модели от значений стохастических трендов. На втором шаге аппроксимируется отклонение от стационарного состояния. Это отклонение зависит от точки, в которой производится аппроксимация.

Для каждого из двух шагов мы приводим несколько возможных имплементаций, однако все они не зависят от какой-либо одной реализации шоков и дадут одинаковые решения независимо от того, по какой траектории система пришла в текущую точку. Трендовая часть аппроксимируется с помощью наивного перебора по сетке, решения в конечном числе точек при необходимости, линейной аппроксимации, алгоритма Смоляка и перебора по сетке с использованием дифференциальных уравнений. Для аппроксимации циклической части мы используем те же самые алгоритмы, за исключением перебора по сетке с использованием дифференциальных уравнений, и добавляем константную аппроксимацию.

Несмотря на то, что данный алгоритм в ряде имплементаций оказывается вычислительно сложнее, чем алгоритмы для линейной части из *Kim et al. (2008) и Lan and Meyer-Gohde (2014)*, он позволяет моделировать нелинейные зависимости, являющиеся следствием изменения стохастических трендов. Время работы большинства из этих имплементаций приемлемо для оценки моделей с небольшим количеством трендов и предложенных алгоритмов оценки. Также большинство предложенных здесь процедур может быть легко распараллелено, что значительно сократит время работы этих алгоритмов.

Данный алгоритм может быть применен, например, для моделей, в которых TFP является процессом с единичным корнем, а функция полезности не является логарифмической (как альтернатива для добавления тренда напрямую в функцию полезности, см., например, *Aruoba et al. (2016)*), для моделей стран экспортеров нефти, где

реальная цена на нефть следует случайному блужданию или моделей открытой экономики, где тренды роста отечественной и зарубежной экономик следуют различным стохастическим трендам.

Дальнейшее изложение построено следующим образом: в разделе 2 будет описан алгоритм решения, раздел 3 посвящен алгоритмам оценки, в разделе 4 приведены различные алгоритмы имплементации и результаты расчетов, в разделе 5 заключение.

Решение модели

В данной работе мы рассматриваем системы вида:

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

при этом последние n_A уравнений:

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

где $A_t -$ стохастические тренды, $y_t -$ эндогенные переменные, $e_t -$ трендовые шоки, $\varepsilon_t -$ не трендовые шоки. Шоки имеют нулевое математическое ожидание и постоянную диагональную ковариационную матрицу.

Также мы предполагаем непрерывную дифференцируемость f по всем аргументам, полный ранг для матрицы $f_1' + f_2' + f_3'$ и наличие единственного ограниченного состояния равновесия (*steady state*) $y_{ss}(A_t)$ для каждого A_t в некоторой (интересующей) области[3]

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

Предлагаемый далее метод аппроксимации нестационарных моделей является адаптацией предложенных пертурбационных методов в *Kim et al. (2008)* и *Lan and Meyer-Gohde (2014)*. Также в этих работах обсуждается (*accuracy in probability for finite time span*) мотивация для аппроксимации нестационарный моделей и соотнесения их с данными. Далее мы ищем решение в виде[4]:

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

[3] Т.е. если мы знаем значение всех стохастических трендов, мы можем однозначно определить состояние равновесия.

[4] Для простоты мы полагаем, что для каждого y_{ss} существует единственное невзрывное решение в интересующей области и будем работать только с этим случаем. Подробнее о множественных решениях см. *Lubik and Schorfheide (2003)* и *Ascari et al.(2016).*

Алгоритм аппроксимации можно разделить на два этапа. На первом решается система (3) для различных значений A_t. На втором этапе производится аппроксимация около полученных трендовых значений и делается корректировка на их изменение.

Аппроксимация цикла

Решение системы (3) для фиксированного значения A_t является стандартной процедурой при аппроксимации решения DGSE-моделей. При наличии траектории сбалансированного роста замена переменных вида похожего на:

$$\hat{y}_t = \frac{y_t}{F(A_t)}$$

(где $F(A_t)$ – некая функция от значений трендов в момент времени t) обычно переводит систему вида (3) в систему, не зависящую от A_t. При наличии такой замены для некоторого подмножества A_t (т.е. для некоторых из стохастических трендов) мы предполагаем предварительную замену переменных модели и уменьшение количества стохастических трендов для ускорения процедуры.

После этого для некоторых фиксированных значений A_t решается система (3) или решение в зависимости от A_t аппроксимируется одним из приведенных в разделе 3 способов. Таким образом, после этого шага имеется решение для некоторых значений трендов. Далее мы покажем, что для предлагаемых в данной работе методов оценки необходимо решение лишь в пропорциональном количеству периодов наблюдений числе точек.

Аппроксимация тренда

На шаге 2 для некоторых значений A_t аппроксимируются уравнения из системы (1), за исключением уравнений (2) относительно $y_{ss}(A_t)$. Аппроксимация записывается следующим образом[5]:

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

где $g_1'(A_t)$ – коэффициенты, зависящие от A_t.

Далее система корректируется на изменение $y_{ss}(A_t)$ и $y_t - y_{ss}(A_t)$ обозначается как $\hat{y}_t$:

[5] См. Приложение А.

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

При этом $y_{ss}(A_{t-1}) - y_{ss}(A_t)$ можно получить из результатов первого шага.

Некоторые алгоритмы оценки

Добавляя уравнения измерений в виде:

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

где y_t^{obs} – наблюдаемые переменные, $c_0(A_t, A_{t-1}, \dots)$ и $c_1(A_t, A_{t-1}, \dots)$ – коэффициенты, зависящие от текущих и прошлых значений трендов, e_t^{obs} – ошибки измерений.

Полученная система может быть оценена с использованием фильтра частиц (*particle filter*). Несложно заметить, что при заданных значениях трендов система является линейной. Предположив, например, нормально распределенные ошибки, систему можно оценить с использованием условного фильтра частиц[6]. Фактически при использовании фильтра частиц необходимо решение в NT-точках (где N – количество частиц, а T – количество точек) для каждого вычисления функции правдоподобия. В случае байесовского оценивания, используя оценки функции правдоподобия и техники сэмплирования, такие как MH (*Metropolis-Hastings*) и SMC (*Sequential Monte Carlo*), можно получить апостериорное распределения для параметров модели[7,8,9]. При использовании оценок с применением метода максимального правдоподобия возникают проблемы с зависимостью аппроксимации от случайных величин, которые сэмплируются при реализации данного алгоритма. Во

[6] Более подробно о фильтре частиц и условном фильтре частиц написано, например, в *Doucet and Johansen (2011)* или *Herbst and Schorfheide (2015).*

[7] Стоит отметить, что при использовании оценок функции правдоподобия вместо настоящих значений из-за использования аппроксимации фильтром частиц алгоритмы сэмплирования могут давать точки не из апостериорного распределения. Валидность MH и SMC алгоритмов доказана в *Andrieu et al. (2010)* и *Chopin et al. (2012)* соответственно.

[8] Другой возможный алгоритм – IS с аккуратным построением вспомогательной плотности. Например, апостериорное распределение в линейной моли может быть использовано для построения вспомогательного распределения. Вычисления, относящиеся к нелинейной модели, могут быть проведены с использованием распараллеливания.

[9] Ряд улучшений стандартных алгоритмов были использованы для алгоритмов оценки и вычисления функции правдоподобия. В работе *Tran et al. (2016)* был использован IS^2-алгоритм для оценки модели *state space* с несмещенной оценкой функции правдоподобия и предложен алгоритм выбора количества частиц для сокращения времени работы. Одной из работ, посвященных оптимальному выбору количества частиц в контексте PMMH-алгоритмов может служить, например, работа *Doucet et al. (2015)*. Добавление коррелированных состояний также может улучшать свойства алгоритмов (см. *Dahlin et al.(2015)* и *Deligiannidis et al.(2016)*). Такие алгоритмы, как *auxiliary particle filter* (*Doucet and Johansen (2011)*), *tempered particle filter* (*Herbst and Schorfheide (2016)*) при условии несмещенности оценок функции правдоподобия, *particle EIS* (*Scharth and Kohn (2016)*) могут быть применены для сокращения дисперсии оценок функции правдоподобия. Для улучшения свойств вспомогательных плотностей существует ряд техник (см. *Giordani and Kohn (2010), Hoogerheide et al. (2012), Herbst and Schorfheide (2015)*). Для стандартных SMC-алгоритмов (см. *Herbst and Schorfheide (2015)*) тоже существует ряд расширений, например SQMC (*Gerber and Chopin (2014)*).

избежание этой проблемы можно использовать алгоритм из *Malik and Pitt (2011)* с непрерывным ресэмплингом.

Данная работа в большей мере посвящена решению моделей, поэтому мы не будем производить их оценку, однако приведем еще один алгоритм сэмплирования в дополнение к описанным выше:

1. Выбрать начальную точку для параметров модели θ_0
2. Для $n = 1, \dots, N$:

   2a. Для $t = 1, \dots, T$:

   Сэмплируем A_t^n из $p(A_t^n | A_{-t}, \theta_{n-1}, y_{1:T})$ используя MH алгоритм[10]

   2b. Сэмплируем θ_n из $p(\theta_n | A_{1:\mathrm{T}}^n, y_{1:T})$ используя MH алгоритм.

Дальнейшая часть работы посвящена сравнению различных алгоритмов решения. Ключевыми параметрами будут являться скорость решения и точность аппроксимации.

Перед тем как перейти к оценке свойств алгоритмов, отметим, что в случае наблюдаемых трендов функция правдоподобия может быть найдена точно[11]. Функция правдоподобия может быть записана следующим образом:

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

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

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

Первый множитель отражает правдоподобие трендов и может быть вычислен при учете того, что тренды являются наблюдаемыми; второй множитель может быть посчитан с помощью фильтра Калмана (при условии нормально распределенных ошибок).

Имплементация алгоритма

Для сравнения алгоритмов мы будем использовать модель, описанную в приложении Б. Это модифицированная модель без траектории сбалансированного роста из *Evans and*

[10] Возможна модификация с дополнительным сэмплированием состояний.

[11] Примером такой модели может быть модель, в которой цены на нефть следуют случайному блужданию.

Phillips (2015). Мы добавляем в нее товар, который не требует труда и капитала для производства (*endowment good*). Производство этого товара следует случайному блужданию. Калибровка модели приведена в таблице 1.

Алгоритмы решения

Реализация алгоритма делится на две части: аппроксимация трендов и циклической части. Для аппроксимации трендов будут использованы следующие алгоритмы: наивный перебор по сетке, решение в конечном числе точек при необходимости, линейная аппроксимация, алгоритм Смоляка и перебор по сетке с использованием дифференциальных уравнений[12].

Наивный перебор по сетке является простым в реализации алгоритмом, однако вычислительная сложность этого алгоритма растет экспоненциально с увеличением количества трендов. Алгоритм Смоляка работает на разреженной сетке и частично решает эту проблему, однако его сложность увеличивается при увеличении количества трендов. В меньшей степени этой проблеме подвержены алгоритм линейной аппроксимации и решение в конечном числе точек при необходимости. Первый из них, однако, может быть недостаточно точным при наличии сильных нелинейностей, а сложность второго растет при увеличении числа точек аппроксимации (обычно число точек аппроксимации пропорционально количеству периодов при оценке). Алгоритм с использованием дифференциальных уравнений в том виде, в котором он применяется в данной статье, тоже страдает от «проклятия размерности», но его использование позволяет значительно сократить необходимое количество численных решений системы уравнений для стационарного состояния.

Для аппроксимации циклической части мы используем те же самые алгоритмы, за исключением перебора по сетке с использованием дифференциальных уравнений, и добавляем константную аппроксимацию.

Аппроксимация тренда

SMC-и MH-алгоритмы требуют нахождения решения в NT-точках для вычисления правдоподобия, а предложенный двухшаговый алгоритм для одной итерации – в $2T$ точках,

[12] См. приложение В.

поэтому мы сравниваем алгоритмы решения в конечном числе точек с остальными для *NT*-и *2T*-точек.

Мы используем MATLAB при реализации алгоритмов[13], при этом мы не делаем параллельные вычисления для алгоритмов решения, которые могут быть применены в том или ином виде к большинству из них. Для решения в определенной точке используется функция *fsolve*[14]. При этом начальные условия берутся из решения в предыдущей точке. Более подробное описание алгоритмов может быть найдено в приложении В.

При использовании наивного перебора по сетке для каждого из трендов мы перебираем по 49 точек в обе стороны, используя равномерный шаг, равный стандартному отклонению для изменения их логарифмов, т.е. в итоге мы получаем решение в области $[logA_0 - 49\sigma_1; logA_0 + 49\sigma_1] \times [logd_0 - 49\sigma_2; logd_0 + 49\sigma_2]$. Мы используем ту же сетку (99×99) при аппроксимации трендов дифференциальными уравнениями, а также сетку 9×9 с теми же границами[15]. Далее при аппроксимации мы используем эту область и для алгоритма Смоляка. При сравнении скорости алгоритма в *2T*-точках используется сгенерированной ряд из *T=100* точек, и модель два раза решается в этих точках.

Мы решаем модели, следуя уравнениям (Б.7.ss)-(Б.13.ss). При этом численно мы решаем только уравнение (Б.10.ss), поэтому сравниваем точность аппроксимации этого уравнения. Берутся две меры точности: разность логарифмов численного решения и аппроксимации; безразмерная величина, равная остатку при подстановке, деленной на сумму модулей всех слагаемых[16]. Для сравнения остатков вычисляются значения на сетке 999×999.

Результаты представлены в таблице 2. Перебор по сетке занимает порядка 30 секунд и является самым медленным (за исключением аппроксимации в конечном числе точек для *NT*-точек) алгоритмом ввиду того, что приходится много раз численно решать уравнение (Б.10.ss). Все остальные алгоритмы требуют гораздо меньшего времени и считаются менее двух секунд. Алгоритм с линейной аппроксимацией решает уравнение (Б.10.ss) три раза, используя численную производную, однако может быть легко модифицирован к алгоритму с использованием дифференциальных уравнений с использованием касательной плоскости в начальной точке, т.е. находить производную аналитически. Алгоритм линейной

[13] При симуляциях использовался компьютер со следующими характеристиками: Intel(R) Core(TM) i5-4210U CPU @ 1.70 GHz 2.40 GHz, RAM 4 GB.
[14] С точностью 10^{-6}, что во многом определяет в дальнейшем порядок точности алгоритмов, использующих эту функцию.
[15] Вне узлов сетки используется билинейная аппроксимация.
[16] Далее мы покажем, что точность аппроксимации уравнения (Б.3.ss) зависит от этих ошибок.

аппроксимации в представленном здесь виде занимает от 2 до 11 сотых секунды[17] в зависимости от количества точек, в которых делается аппроксимация. При этом время работы алгоритма с использованием дифференциальных уравнений для 200 точек составляет 1 сотую секунды для сетки 9 × 9 и 27 сотых на сетке 99 × 99, для 100000 точек – 137 и 162 сотые секунды. Аппроксимация в конечном числе точек при необходимости также требует множество раз решать уравнение (Б.10.ss). Для 200 точек решение занимает 61 сотую секунды, а для 100000 – 305 секунд. Алгоритм Смоляка с 29 узловыми точками работает 11 и 129 сотых секунды для 200 и 100000 точек соответственно. Аналогичные результаты для алгоритма Смоляка с использованием 13 точек – 6 и 120 сотых, 5 точек – 2 и 115 сотых.

Естественно, что в большинстве случаев платой за скорость алгоритма является его точность. Алгоритм аппроксимации в конечном числе точек является наиболее точным ввиду того, что он решает уравнение (Б.10.ss) в необходимых при аппроксимации точках и отличается от точного решения лишь из-за ошибок при численном решении, поэтому мы не приводим показатели точности данного алгоритма. В случае, когда размер сетки превышает количество точек аппроксимации, алгоритм перебора по сетке является менее точным, но занимает больше времени (как в примере с 200 точками). Наивный перебор по сетке является вторым из представленных по точности алгоритмов. Максимальная ошибка в отклонении логарифма труда от стационарного значения порядка 0,0001. Для алгоритма с использованием дифференциальных уравнений на той же сетке ошибка порядка 0,025, а для алгоритма Смоляка с 29 точками – 0,04. Максимальные ошибки в остальных алгоритмах достаточно велики. При этом средние ошибки для представленных алгоритмов более чем на полпорядка меньше.

Аппроксимация цикла

Мы используем несколько модификаций стандартной линеаризации вокруг состояния равновесия, которые описаны в приложении В. Для линеаризации мы используем стандартный алгоритм *gensys*, описанный в работе *Sims (2002)*. Используется точно такая же сетка, как и для аппроксимации тренда. Мы не проводим оценку точности для цикла, так как сложно разделить цикл и тренд. Вместо этого мы приводим ниже оценку точности полного решения для различных комбинаций аппроксимаций тренда и цикла.

[17] 1 сотая в случае аналитической производной.

В таблице 3 представлена продолжительность работы различных аппроксимаций цикла[18]. Все времена приведены при предварительном просчете стационарных состояний, которые входят в систему (Б.1.lin)-(Б.7.lin).

Как и для алгоритмов аппроксимации тренда, перебор по сетке занимает наибольшее время (9 и 12 секунд для 200 и 100000 точек). Больше времени занимает аппроксимация в конечном числе точек для 100000 точек, т.к. количество решений системы (Б.1.lin)-(Б.7.lin) значительно больше, чем для алгоритма перебора по сетке. Для 200 точек время решения составляет 19 сотых секунды. Алгоритм Смоляка с использование 5, 13 и 29 узлов для аппроксимации в 200 точках занимает 0,6, 2 и 3 сотые секунды соответственно, а для 100000 – 50, 56 и 60 сотых секунды. Время работы линейной аппроксимации 0,4 и 4 сотые секунды.

Полное решение

Здесь мы комбинируем одинаковые способы аппроксимации циклической и трендовой части, чтобы оценить точность полного решения. Любые другие комбинации также возможны (например, линейная аппроксимация тренда и алгоритм Смоляка для цикла), однако мы не рассматриваем их в данной работе. Для оценки точности полного решения были сгенерированы 1000 случайных точек внутри используемой ранее области. Также при $A_{ss} = 1$ и $d_{ss} = 1$ был сгенерирован ряд из 100 наблюдений. В этих 100000 узлах мы оцениваем точность алгоритмов, вычисляя для каждого из уравнений (Б.1)-(Б.6) десятичный логарифм безразмерной ошибки[19]. Для численного интегрирования применяется метод, изложенный в *Tauchen (1986)*.

Мы проводим данную процедуру для двух наборов параметров: с $\sigma_\varepsilon = 0.01$ и $\sigma_\varepsilon = 1$. В первом случае влияние циклического шока мало по сравнению с трендовыми, во втором – превосходит его.

В таблице 4 представлены результаты. Несложно заметить, что уравнения (Б.4)-(Б.6) хорошо аппроксимируются всеми алгоритмами. Эти уравнения выполняются как для аппроксимации трендов ((Б.9.ss), (Б.11.ss) и (Б.13.ss) задают эквивалентную систему уравнений), так и для аппроксимации цикла ((Б.4.lin)-(Б.6.lin)). Прологарифмировав уравнение (Б.3), мы получим разделение на слагаемые, подобное разделению в уравнениях (Б.4)-(Б.6), однако точность аппроксимации этого уравнения гораздо меньше. Такая ситуация возникает из-за предложенного способа аппроксимации решения системы (Б.1.ss)-

[18] Описание алгоритмов можно найти в приложении В.
[19] Мы делим остаток уравнения на сумму модулей его слагаемых.

(Б.6.ss). Обозначим аппроксимированное решение для труда l_t^{*ss}. Тогда перепишем уравнение (Б.10.ss) в виде:

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

где err – ошибка аппроксимации. Подставляя уравнения (Б.8.ss) и (Б.12.ss), получим:

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

или

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

Таким образом, ошибка в уравнении (Б.3) зависит от ошибки аппроксимации решения уравнения (Б.10.ss). Можно также увидеть, что эта ошибка не сильно зависит от дисперсии циклического шока. Для алгоритма Смоляка и линейной аппроксимации ошибка в данном уравнении достаточно велика, однако, для алгоритмов с использованием сетки точность увеличивается при добавлении новых узлов, что дает возможность увеличивать точность, добавляя узлы[20]. Уравнения (Б.1)-(Б.2) после логлинеаризации не оказываются эквивалентными изначальным. Ошибки в них в большей степени зависят от дисперсии циклического шока.

Как и для аппроксимации трендов, аппроксимация полного решения показывает, что алгоритм решения в конечном числе точек является наиболее точным. Следующим по точности оказывается алгоритм наивного перебора по сетке. Менее точен алгоритм Смоляка. Алгоритм с линейной аппроксимацией достаточно плохо приближает решение и не может быть использован для решения предложенной модели.

В заключениие мы протестируем алгоритм линеаризации вокруг одной точки, как в *Kim et al. (2008)*. Для этого мы решаем систему для трендов в точке $A_{ss} = 1$ и $d_{ss} = 1$. Затем используя систему (Б.1.dif)-(Б.6.dif), проводим касательную плоскость (в терминах логарифмов) в этой точке. Добавляя константную аппроксимацию цикла, получаем полное решение. В отличие от обычной линейной аппроксимации трендов, приближается не только уравнение для труда, но и все остальные уравнения. Вследствие этого не возникает проблемы, связанной с уравнением (Б.3), как было описано выше. Наибольшая ошибка для этого метода появляется в уравнении (Б.1), которое зависит от значений стационарных состояний. Десятичный логарифм максимальной ошибки равен -0,34 и -0,33 для $\sigma_\varepsilon = 0{,}01$ и $\sigma_\varepsilon = 1$ соответственно. Средняя ошибка равна -1,93 и -1,77. Остальные ошибки не большего

[20] См. ниже обсуждение возможности распараллеливания таких алгоритмов.

порядка, чем в других методах, что позволяет при небольших колебаниях трендовых переменных в интересующей области использовать этот метод ввиду его простоты и скорости работы.

Обсуждение алгоритмов

Ограничения и возникающие проблемы

Как и большинство пертурбационных методов, предложенный в данной работе алгоритм будет работать хорошо лишь для моделей «без изломов», каковой и является описанная здесь модель. При этом если возможно для увеличения скорости лучше избавиться от трендов, которые могут быть исключены из системы той или иной заменой переменных. В моделях со слабыми нелинейностями в трендах большинство из имплементаций предложенного алгоритма не будет работать существенно лучше, чем обыкновенная линеаризация около одной точки. Прежде чем переходить к алгоритмам, которые занимают больше времени, вероятно, стоит попробовать оценить модель с помощью метода из *Kim et al. (2008)* и посмотреть на возникающие нелинейности.

Существует также ряд проблем, связанных с идентификацией, которые могут возникать при оценке. Поясним это на примере описанной в данной работе модели. Предположим, что мы наблюдаем y_t, c_t и r_t. Возьмем $\chi' = 2\chi$, тогда $\frac{w_t'}{l_t'^{\theta}} = 2\frac{w_t}{l_t^{\theta}}$. Учитывая при этом, что $w_t'\, l_t' = w_t l_t$ получим выражения для труда и зарплат. Также скорректировав A_t из уравнения $A_t' l_t'^{1-\alpha} = A_t l_t^{1-\alpha}$, имеем полную эквивалентность параметров χ' и χ. В моделях со сбалансированной траекторией роста такая неидентифицируемость может исчезать вследствие нормировки и использования логлинеаризации. Существует ряд стандартных методов борьбы с проблемой идентификации, на которых мы не будем подробно останавливаться здесь.

Увеличение скорости работы и точности алгоритмов

Реализация предложенной двухшаговой процедуры может быть улучшена в нескольких направлениях, некоторые из них мы обсудим в этом разделе.

Для ускорения большинства из предложенных имплементаций могут быть применены параллельные вычисления[21] (с использованием нескольких процессоров или видеокарты). Алгоритмы, в которых прибегают к вычислениям на сетке (перебор по сетке, алгоритм Смоляка), легко могут быть распараллелены для вычисления нужных функций в узлах сетки. Таким образом, точность алгоритмов может быть улучшена за счет увеличения количества узлов аппроксимации с меньшей потерей времени. При этом необходимо также соотносить время, которое мы можем выиграть от распараллеливания, и время, которое мы потеряем при использовании параллельных вычислений. Сложнее обстоит дело с алгоритмом решения в конечном числе точек. При применении фильтра частиц мы не знаем заранее точек, в которых нам нужно делать аппроксимацию. Мы можем использовать распараллеливание по частицам. При алгоритме с поочередным сэмплированием трендов и параметров существует возможность предварительного расчета трендовых значений для МН-алгоритма[22] (см. *Strid (2009)*) на шаге 2а. Шаг 2б в этом алгоритме может быть распараллелен, т.к. мы заранее знаем значения трендов. Также может быть использован МН с сэмплированием трендов, которое не зависит от значений трендов на текущей итерации на шаге 2а, что также позволяет заранее находить решения[23].

Большей точности можно добиться увеличением порядка аппроксимации. Так, вместо линейной аппроксимации мы можем использовать квадратичную, которая легко вычисляется при наличии линейной[24]. Точность алгоритмов с использованием сетки может быть улучшена при увеличении числа узлов. Вероятно, другой способ выбора точек на разреженной сетке может помочь увеличить точность. Поясним это на примере алгоритма Смоляка с 5 точками. На рисунке B1.a показано расположение узловых точек. Однако при таком расположении наибольшие ошибки возникают в углах области. При помещении узловых точек в углы ошибка уменьшается.

[21] См. *Brumm and Scheidegger (2015)* как пример использования параллельных вычислений для решения DSGE-моделей.
[22] Также может быть применен для МН-шага в алгоритмах с использованием фильтра частиц.
[23] Использование простых техник может сильно снизить эффективность алгоритма, а использование адаптивных требует отдельного доказательства их валидности (см. *Roberts and Rosenthal (2007, 2009)*). См. также *Giordani and Kohn (2010)* и *Hoogerheide et al. (2012)* для построения вспомогательной плотности (proposal density).
[24] См., например, *Shmitt-Grohe and Uribe (2004)* или *Kim et al. (2008).*

Заключение

В данной работе был представлен двухшаговый алгоритм для решения DSGE-моделей, в которых отсутствует траектория сбалансированного роста. Ряд предложенных имплементаций позволяет в некотором диапазоне контролировать точность аппроксимации, и они могут быть легко распараллелены. Все это дает возможность для решения и оценки большого класса моделей. Мы надеемся, что данный алгоритм будет полезен в большом количестве прикладных исследований, включая исследования, посвященные развивающимся экономикам, где многие соотношения, которые являются постоянными для развитых экономик, меняются достаточно сильно, что делает невозможным их анализ с использованием стандартных техник оценивания DSGE-моделей.

Алгоритм может быть полезен для моделирования российской экономики. После резкого падения цен на нефть в 2014–2015 годах стандартные DSGE-модели, где реальная цена на нефть возвращается к среднему, не соответствовали ожиданиям того, что цена останется на таком уровне достаточно долго. При этом наш алгоритм позволяет моделировать часть динамики нефтяных цен как случайное блуждание, что задает долгосрочное падение и не подразумевает возвращение к среднему.

Список литературы

1. Andrieu C., Doucet A., Holenstein R. Particle Markov Chain Monte Carlo Methods // Journal of the Royal Statistical Society Series. 2010. 72(3). P. 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. P. 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). P. 295–313.
11. Gerber M., Chopin N.. Sequential Quasi-Monte Carlo // 2014. arXiv:1402.4039v5.
12. Gorodnichenko Y., Ng S. Estimation of DSGE Models When the Data are Persistent // Journal of Monetary Economics. 2010. Vol. 57(3). P. 325-340.
13. 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.
14. Fernandez-Villaverde, J., Rubio-Ramirez, J. Estimating Macroeconomic Models: A Likelihood Approach // Review of Economic Studies.2007. Vol. 74(4). P. 1059–1087.
15. Fernández-Villaverde, J., Rubio-Ramirez, J., Schorfheide, F. Solution and Estimation Methods for DSGE Models // Handbook of Macroeconomics. 2016. Vol. 2. Preliminary draft.

16. Giordani, P., Kohn, R. Adaptive Independent Metropolis-Hastings by Fast Estimation of Mixtures of Normals // Journal of Computational and Graphical Statistics. 2010. Vol.16. P. 243-259.
17. Herbst, Ed., Schorfheide, F. Bayesian Estimation of DSGE Models // Princeton University Press. 2015.
18. Herbst, Ed., Schorfheide, F. Tempered Particle Filtering // Manuscript. 2016.
19. Hoogerheide, L., Opschoor, A., van Dijk, H. A class of adaptive importance sampling weighted EM algorithms for efficient and robust posterior and predictive simulation // Journal of Econometrics. Vol. 171(2). P. 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. P. 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, P. 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. P. 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). P. 273–285.
25. Malik, S., Pitt, M. Particle Filters for Continuous Likelihood Evaluation and Maximization // Journal of Econometrics. 2011. Vol. 165. P. 190–209.
26. Roberts, G. and Rosenthal, J. Coupling and ergodicity of adaptive MCMC // Journal of Applied Probability. 2007. Vol. 44. P. 458–475.
27. Roberts, G. and Rosenthal, J. Examples of adaptive MCMC // Journal of Computational and Graphical Statistics. 2009. Vol. 18. P. 349–367.
28. 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. P. 755-775.
29. Scharth M. and Kohn R. Particle efficient importance sampling // Journal of Econometrics. 2016. Vol. 190(1). P. 133 – 147.

30. Sims C. Solving Linear Rational Expectations Models // Computational Economics. Society for Computational Economics, 2002. Vol. 20 (1–2). P. 1-20.
31. Smolyak S. Quadrature and Interpolation Formulas for Tensor Products of Certain Classes of Functions // Soviet Mathematics. 1963. Vol. 4. 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, P. 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

Приложение А

Подставляя (4) в (1) получаем:

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

Или в аппроксимации первого порядка:

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

Собирая коэффициенты при одинаковых переменных:

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

Решив систему квадратичных уравнений относительно g_1' (см., например, *Sims (2002)*), далее подставляем это решение в последующие уравнения и получаем g_3' и g_4'. Несложно также заметить, что $g_5' = 0$.

Приложение Б

Домохозяйства максимизируют полезность:

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

при бюджетном ограничении:

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

Условия первого порядка:

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

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

Производственная функция и условия первого порядка для фирмы:

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

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

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

Имея 6 уравнений и 6 переменных $[c_t, k_t, y_t, w_t, r_t, l_t]$ и дополняя систему уравнениями для экзогенных процессов:

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

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

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

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

Система для стационарных состояний

Система для стационарных состояний может быть записана следующим образом:

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

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

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

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

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

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

Решение записывается следующим образом:

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

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

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

Далее решаем относительно l_t^{ss} следующее уравнение:

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

Далее находим

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

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

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

Система дифференциальных уравнений

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

$$r = 0 \quad \text{(Б.2.dif)}$$

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

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

$$r = y - k \quad \text{(Б.5.dif)}$$

$$w = y - l \quad \text{(Б.6.dif).}$$

Линеаризованная система

Линеаризованная относительно стационарного состояния в момент времени *t* система выглядит следующим образом:

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

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

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

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

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

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

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

Приложение В

Алгоритмы решения для трендовой части

Наивный перебор по сетке

Обычный перебор при разбиении интересующего пространства (после некоторого преобразования в куб $[-1; 1]^n$), нахождение решение в узлах и последующая аппроксимация внутри при необходимости. Так для одного тренда может быть использована линейная функция, для двух – билинейная и т.д.

Алгоритм Смоляка

Алгоритм Смоляка (*Smolyak (1963)*) является стандартным алгоритмом аппроксимации и представляет один из алгоритмов перебора по разреженной сетке[25]. Ряд расширений был предложен для алгоритма Смоляка (см., например, *Judd et al. (2014)*), однако для простоты мы будем использовать стандартный алгоритм.

Вместо обычного перебора по сетке используется тензорное произведение по каждой координате, однако с рядом ограничений. Множество вершин порядка *i* по координате *k* (после линейной трансформации к отрезку [-1; 1]) будем определять следующей формулой:

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

где $m_1 = 1,$ а для $i > 1 \;\; m_i = 2^{i-1} + 1.$

Вместо обычного перебора по сетке, где множество точек определяется тензорным произведением:

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

множество точек определяется тензорным произведение с ограничением:

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

[25] Алгоритм Смоляка был использован как один из представителей алгоритмов на разреженной сетке. Другие алгоритмы на разреженной сетке могут быть использованы для аппроксимации как трендовой части, так и для циклической.

В основном тексте для модели с двумя координатами мы будем использовать следующие сетки с $q = 3$, $q = 4$ и $q = 5$ (см. рисунок B1).

Рисунок B1а. Сетка с q = 3

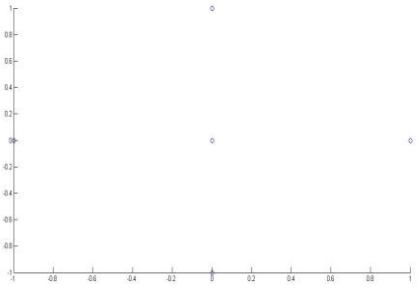


Рисунок B1б. Сетка с q = 4

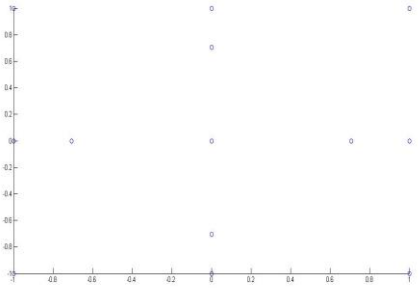
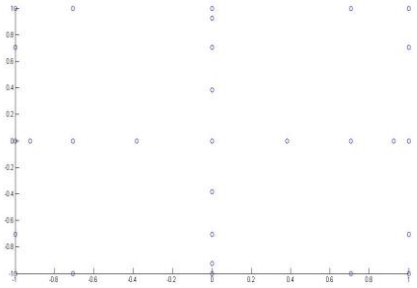


Рисунок B1в. Сетка с q = 5



Функция $f(x_1, \dots, x_n)$ аппроксимируется функцией:

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

где $\varphi^{\sum i_n}(x, a) = \sum_{j_1=1}^{m_{i_1}} \dots \sum_{j_n=1}^{m_{i_n}} a_{j_1,\dots,j_n} T_{j_1-1}(x_1) \dots T_{j_n-1}(x_n)$, а в качестве базисных функций $T_j(x_i)$ используются полиномы Чебышева.

Линейная аппроксимация

Под линейной аппроксимацией здесь понимается аппроксимация функции $f(x_1, \dots, x_n)$ в точке x^0

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

Фактически функция аппроксимируется касательной плоскостью в точке x^0.

Аппроксимация в конечном числе точек при необходимости

Как видно из уравнений (6)-(7) и алгоритмов оценки, необходимо решить модель лишь в конечном числе точек. Так при алгоритмах, основанных на фильтре частиц, необходимо решить модель в *NT*-точках, а в алгоритме с поочередным сэмплированием трендов и параметров – в *2T*-точках.

Перебор по сетке с использованием дифференциальных уравнений

Система (3) для стационарных состояний при условии непрерывной дифференцируемости по всем переменным может быть записана следующим образом:

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

Также если определитель матрицы $f_1' + f_2' + f_3'$ не равен нулю, то можем записать:

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

Найдя решение в одной точке и продляя его с помощью полученной системы, можно получить решение в необходимой области. В данной работе в примере для двух трендов мы будем находить сначала решение для фиксированной одной точки по второй координате (в точке точного решения), т.е. мы фиксируем $A_{ss}^{2,0}$ и находим $y_{ss}(A_{ss}^{1,i}, A_{ss}^{2,0})$ для $i = -n, \dots, 0, \dots, n$, двигаясь с постоянным шагом от нуля в обе стороны. Затем для $j = -n, \dots, 0, \dots, n$ вычисляем $y_{ss}(A_{ss}^{1,i}, A_{ss}^{2,j})$ аналогичным образом для каждого i. Результаты метода зависят от порядка трендов в векторе A_{ss}. Этого можно избежать, перебрав все возможные перестановки, однако это приводит к увеличению времени работы алгоритма. Из основного текста видно, что в предложенном в данной работе примере результаты не меняются сильно при выборе направления.

Алгоритмы решения для циклической части

Наивный перебор по сетке, алгоритм Смоляка, линейная аппроксимация

Эти алгоритмы по своей сути ничем не отличаются от аналогичных, представленных для трендов. Находится линейное решение вида (6). Далее поэлементно для матриц $g_1'(A_t)$, $g_3'(A_t)$, $g_4'(A_t)$ находится аппроксимация.

Аппроксимация в конечном числе точек при необходимости

Алгоритм аналогичен алгоритму для трендов.

Константная аппроксимация

Матрицы $g_1'(A_t)$, $g_3'(A_t)$, $g_4'(A_t)$ полагаются не зависящими от A_t и равными линейной аппроксимации в одной точке.

Приложение Г

Таблица 1. Параметры модели

Параметр	*Значение*
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

Таблица 2.а. Десятичный логарифм ошибки аппроксимации в уравнении Б.10.ss

	Максимальное	*Среднее*
Перебор по сетке	-4,2	-4,7
Аппроксимация в конечном числе точек	-	-
Линейная аппроксимация	-0,2	-1,0
Алгоритм Смоляка (29 точек)	-1,8	-2,7
Алгоритм Смоляка (13 точек)	-1,2	-2,0
Алгоритм Смоляка (5 точек)	-0,4	-1,2
ДУ 99x99	-1,7	-2,4
ДУ 9x9	-0,1	-0,8

Таблица 2.б. Ошибка стационарного состояния для труда, десятичный логарифм процентного отклонения

	Максимальное	*Среднее*
Перебор по сетке	-3,9	-Inf
Аппроксимация в конечном числе точек	-	-
Линейная аппроксимация	0,2	-0,9
Алгоритм Смоляка (29 точек)	-1,4	-2,5
Алгоритм Смоляка (13 точек)	-0,9	-1,7
Алгоритм Смоляка (5 точек)	-0,1	-1,0
ДУ 99x99	-1,6	-2,2
ДУ 9x9	-0,5	-1,1

Таблица 2.в. Время работы алгоритма аппроксимации тренда для 2T = 200 и NT = 100000 точек, сек

	2T	NT
Перебор по сетке	30,22	30,56
Аппроксимация в конечном числе точек	0,61	305,48
Линейная аппроксимация	0,02	0,11
Алгоритм Смоляка (29 точек)	0,11	1,29
Алгоритм Смоляка (13 точек)	0,06	1,20
Алгоритм Смоляка (5 точек)	0,02	1,15
ДУ 99x99	0,27	1,62
ДУ 9x9	0,01	1,37

Таблица 3. Время работы алгоритма аппроксимации цикла для 2T = 200 и NT = 100000 точек, сек

	2T	NT
Перебор по сетке	9,38	12,15
Аппроксимация в конечном числе точек	0,19	97,70
Линейная аппроксимация	0,004	0,04
Константная аппроксимация	0,001	0,001
Алгоритм Смоляка (29 точек)	0,03	0,60
Алгоритм Смоляка (13 точек)	0,02	0,56
Алгоритм Смоляка (5 точек)	0,006	0,50

Таблица 4а. Десятичный логарифм ошибки полного решения, $\sigma_\varepsilon = 0,01$

		(Б.1)	(Б.2)	(Б.3)	(Б.4)	(Б.5)	(Б.6)
Перебор по сетке	Максимальное	-3,21	-3,97	-3,75	-15,44	-15,29	-15,37
	Среднее	-5,39	-5,57	-4,31	-Inf	-Inf	-Inf
Аппроксимация в конечном числе точек	Максимальное	-3,21	-3,97	-6,32	-15,44	-15,29	-15,34
	Среднее	-5,4	-5,59	-Inf	-Inf	-Inf	-Inf
Линейная аппроксимация	Максимальное	-0,35	0	-0,02	-14,44	-13,88	-14,57
	Среднее	-1,75	-1,06	-0,79	-Inf	-Inf	-Inf
Алгоритм Смоляка (29 точек)	Максимальное	-3,18	-3,68	-1,41	-15,46	-15,23	-15,36
	Среднее	-5,27	-5,34	-2,3	-Inf	-Inf	-Inf
Алгоритм Смоляка (13 точек)	Максимальное	-2,85	-2,92	-0,8	-15,49	-15,29	-15,36
	Среднее	-4,63	-4,65	-1,57	-Inf	-Inf	-Inf
Алгоритм Смоляка (5 точек)	Максимальное	-2,25	-2,43	-0,11	-15,46	-15,29	-15,36
	Среднее	-4,23	-4,31	-0,85	-Inf	-Inf	-Inf

Таблица 4б. Десятичный логарифм ошибки полного решения, $\sigma_\varepsilon = 1$

		(Б.1)	*(Б.2)*	*(Б.3)*	*(Б.4)*	*(Б.5)*	*(Б.6)*
Перебор по сетке	***Максимальное***	-1,34	-1,76	-3,75	·15,26	·15,03	·15,18
	Среднее	-3,22	-3,16	-4,3	-Inf	-Inf	-Inf
Аппроксимация в конечном числе точек	***Максимальное***	-1,34	-1,77	-6,34	·15,14	·15,03	-15,1
	Среднее	-3,22	-3,16	-Inf	-Inf	-Inf	-Inf
Линейная аппроксимация	***Максимальное***	0	0	-0,02	·12,82	·12,52	·12,87
	Среднее	-0,13	-0,07	-0,8	-Inf	-Inf	-Inf
Алгоритм Смоляка (29 точек)	***Максимальное***	-1,33	-1,73	-1,4	·14,94	·14,82	·14,88
	Среднее	-3,22	-3,18	-2,33	-Inf	-Inf	-Inf
Алгоритм Смоляка (13 точек)	***Максимальное***	-1,19	-1,5	-0,8	·15,19	-15	·14,97
	Среднее	-3,08	-3,16	-1,57	-Inf	-Inf	-Inf
Алгоритм Смоляка (5 точек)	***Максимальное***	-1,28	-1,53	-0,11	·15,22	·15,15	·15,12
	Среднее	-2,98	-3,18	-0,85	-Inf	-Inf	-Inf