NBER WORKING PAPER SERIES

NONLINEAR PROGRAMMING METHOD FOR DYNAMIC PROGRAMMING

Yongyang Cai
Kenneth L. Judd
Thomas S. Lontzek
Valentina Michelangeli
Che-Lin Su

Working Paper 19034
http://www.nber.org/papers/w19034

NATIONAL BUREAU OF ECONOMIC RESEARCH
1050 Massachusetts Avenue
Cambridge, MA 02138
May 2013

Cai and Judd gratefully acknowledge NSF support (SES-0951576). Michelangeli acknowledges the funding of the Bank of Italy research fellowship. The views expressed herein are those of the authors and do not necessarily reflect the views of the National Bureau of Economic Research or the Bank of Italy.

Nonlinear Programming Method for Dynamic Programming

Yongyang Cai, Kenneth L. Judd, Thomas S. Lontzek, Valentina Michelangeli, and Che-Lin Su

NBER Working Paper No. 19034
May 2013
JEL No. C61,C63

ABSTRACT

A nonlinear programming formulation is introduced to solve infinite horizon dynamic programming problems. This extends the linear approach to dynamic programming by using ideas from approximation theory to avoid inefficient discretization. Our numerical results show that this nonlinear programming method is efficient and accurate.

Yongyang Cai
Hoover Institution
Stanford University
Stanford, CA 94305
yycai@stanford.edu

Kenneth L. Judd
Hoover Institution
Stanford University
Stanford, CA 94305-6010
and NBER
kennethjudd@mac.com

Thomas S. Lontzek
University of Zurich
Moussonstrasse 15, 8044 Zurich
Thomas.Lontzek@Business.uzh.ch

Valentina Michelangeli
Banca d'Italia
Via Nazionale 91, 00184 Roma
valentina.michelangeli@gmail.com

Che-Lin Su
University of Chicago
che-lin.su@ChicagoBooth.edu

1 Introduction

Dynamic programming (DP) is the essential tool in solving problems of dynamic and stochastic controls in economic analysis. The nonlinearities of dynamic economic problems make them numerically challenging. To avoid the nonlinearity, linear programming (LP) approaches have been studied in the literature; see De Farias and Van Roy (2003), and Trick and Zin (1997). However, the LP approach has limited value for problems with continuous actions and/or states since an LP approach would have to discretize the states and controls. The most common discretization technique is the Cartesian grid, which leads to a curse-of-dimensionality in both the state and action spaces: if each of action or state variable is discretized by m equally spaced nodes, then the number of points is up to m^d, where d is the number of both action and state variables. Moreover, in many economic problems, especially those involving policy evaluations or welfare analysis, it is necessary to obtain accurate approximations of the decision rules, a task that is much more difficult than approximating the value function. Therefore, if we want to have 5-digit accuracy for a problem defined on one state variable and two controls, all of which are mapped to a unit cube, the LP approach would require 100,000 points in each dimension for a total of 10^{15} points, a problem size that is currently infeasible.

This paper presents a nonlinear programming (NLP) method, called DPNLP, to solve the infinite horizon DP problems with or without stochasticity. The method uses shape-preserving approximation methods to approximate the optimal value function by adding some extra degree of freedom. DPNLP solves the deterministic or stochastic DP problem with one or two continuous state variables and several continuous control variables without the curse-of-dimensionality of the action space. Moreover, in our numerical examples, DPNLP uses only 19 nodes of the continuous state and their corresponding 19 two-dimensional actions and takes only seconds or about one minute to achieve the 5-digit or higher accuracy for both deterministic and stochastic DP problems with one continuous state (and one discrete state for the stochastic DP) and two continuous control variables. Since DPNLP has no curse-of-dimensionality of the action space, it can also solve DP problems with many continuous control variables easily and quickly. In our two-country optimal growth examples, the problems have two continuous state variables and six continuous control variables. This makes the LP approach infeasible even within only 2-digit accuracy, but DPNLP solves them in minutes with up to 5-digit accuracy.

In this paper, the DPNLP method is described as an adequate tool to solve a DP problem with one or multiple continuous state variables (and discrete state variables) and multiple continuous control variables. Many economic problems involve models with one or several variables that is "by definition" continuous (such as wealth or capital). Discretizing such a state

would require many grid points, with the computational costs associated to it. However, other state variables, even though continuous, often follow processes that make them suitable for discretization, without significant loss in terms of accuracy of the solution.

Our DPNLP method can also be a crucial component of empirical estimation methods. Michelangeli (2009) used DPNLP inside an MPEC approach (see Su and Judd, 2012) to estimating a model of the demand for reverse mortgages.

The paper is constructed as follows. Section 2 describes the kind of dynamic problem commonly used in economics and the subject of this paper. Section 3 briefly reviews approximation methods such as Chebyshev polynomial approximation. Section 4 defines the DPNLP method for solving infinite horizon DP problems. Section 5, 6 and 7 apply DPNLP to optimal accumulation problems similar to many economics problems. Section 8 concludes.

2 Dynamic Programming

An infinite horizon stochastic optimal decision-making problem has the following general form:

$$V(x_0,\theta_0) = \max_{a_t\in\mathcal{D}(x_t,\theta_t)} \mathbb{E}\left\{\sum_{t=0}^{\infty}\beta^t u(x_t,a_t)\right\}, \tag{1}$$
$$\text{s.t.} \quad x_{t+1} = g(x_t,\theta_t,a_t),$$
$$\theta_{t+1} = h(\theta_t,\epsilon_t),$$

where x_t is the discrete time continuous state vector process with initial state x_0, θ_t is the discrete state vector process with initial state θ_0, ϵ_t is a serially uncorrelated random vector process, g is a continuous function representing the change in the state x_t as a function of the state and action, a_t, and h represents the transition process for θ_t respectively, $\mathcal{D}(x_t,\theta_t)$ is a feasible set of a_t dependent on (x_t,θ_t), β is the discount factor with $0<\beta<1$, u is a concave utility function, and $\mathbb{E}\{\cdot\}$ is the expectation operator. While this description does not apply to many applications of dynamic programming, it does apply to most models in dynamic economics. Examples include economic growth, portfolio decisions, and investment decisions by firms.

The DP model for the general infinite horizon problem is the following Bellman equation (Bellman, 1957):

$$V(x,\theta) = \max_{a\in\mathcal{D}(x,\theta)} u(x,a) + \beta\mathbb{E}\left\{V(x^+,\theta^+)\mid x,\theta,a\right\}, \tag{2}$$
$$\text{s.t.} \quad x^+ = g(x,\theta,a),$$
$$\theta^+ = h(\theta,\epsilon),$$

where (x^+, θ^+) is the next-stage state conditional on the current-stage state (x, θ) and the action a, ϵ is a random variable, and $V(x, \theta)$ is the value function.

In the simpler case where there is no uncertainty, there is no stochastic state θ_t, the problem (1) becomes

$$V(x_0) = \max_{a_t \in \mathcal{D}(x_t)} \sum_{t=0}^{\infty} \beta^t u(x_t, a_t),$$
$$\text{s.t.} \quad x_{t+1} = g(x_t, a_t),$$

and its Bellman equation is:

$$V(x) = \max_{a \in \mathcal{D}(x)} u(x, a) + \beta V(x^+), \tag{3}$$
$$\text{s.t.} \quad x^+ = g(x, a).$$

3 Approximation

In the DP problem (3), we want to solve for the optimal value function. Even though the method allows to compute both the value functions and the policy functions, the implementation of the steps require to solve for the optimal value functions. But when state and control variables are continuous such that value functions are also continuous, we have to use some approximation for the value functions, since computers cannot model the entire space of continuous functions.

An approximation scheme consists of two parts: basis functions and approximation nodes. Approximation nodes can be chosen as uniformly spaced nodes, Chebyshev nodes, or some other specified nodes. From the viewpoint of basis functions, approximation methods can be classified as either spectral methods or finite element methods. A spectral method uses globally nonzero basis functions $\{\phi_j(x)\}$ and coefficients $\mathbf{b} = \{b_j\}$ such that $\hat{V}(x; \mathbf{b}) = \sum_{j=0}^{n} b_j \phi_j(x)$ is a degree n approximation. Examples of spectral methods include ordinary polynomial approximation, Chebyshev polynomial approximation, and shape-preserving Chebyshev polynomial approximation (Cai and Judd, 2012b). In contrast, a finite element method uses locally basis functions $\{\phi_j(x)\}$ that are nonzero over sub-domains of the approximation domain. Examples of finite element methods include piecewise linear interpolation, Schumaker interpolation, shape-preserving rational function spline Hermite interpolation (Cai and Judd, 2012a), cubic splines, and B-splines. See Cai (2010), Cai and Judd (2010), and Judd (1998) for more details.

3.1 Chebyshev Polynomial Approximation

Chebyshev polynomials on $[-1, 1]$ are defined as $\mathcal{T}_j(z) = \cos(j \cos^{-1}(z))$. Economics problems typically live on an interval $[x_{\min}, x_{\max}]$; if we let

$$Z(x) = \frac{2x - x_{\min} - x_{\max}}{x_{\max} - x_{\min}},$$

then $\mathcal{T}_j(Z(x))$ are Chebyshev polynomials adapted to $[x_{\min}, x_{\max}]$ for $j = 0, 1, 2, \ldots$. These polynomials are orthogonal under the weighted inner product: $\langle f, g \rangle = \int_{x_{\min}}^{x_{\max}} f(x)g(x)w(x)dx$ with the weighting function $w(x) = \left(1 - Z(x)^2\right)^{-1/2}$. A degree n Chebyshev polynomial approximation for $V(x)$ on $[x_{\min}, x_{\max}]$ is

$$V(x; \mathbf{b}) = \sum_{j=0}^{n} b_j \mathcal{T}_j(Z(x)), \tag{4}$$

where $\mathbf{b} = \{b_j\}$ are the Chebyshev coefficients. It is often more stable to use the expanded Chebyshev polynomial interpolation (Cai, 2010), as the above standard Chebyshev polynomial interpolation gives poor approximation in the neighborhood of end points.

In this section we describe the Chebyshev polynomial approximation because it is the approximation scheme used in our examples. While also other approximation schemes may be adequate and with good performances, the Chebyshev polynomial approximation presents advantages in terms of coding simplicity and reliability, and easy extension to multidimensional approximation.

3.2 Multidimensional Complete Chebyshev Approximation

In a d-dimensional approximation problem, let the domain of the approximation function be

$$\left\{x = (x_1, \ldots, x_d) : x_i^{\min} \leq x_i \leq x_i^{\max},\ i = 1, \ldots d\right\},$$

for some real numbers $x_i^{\min}$ and $x_i^{\max}$ with $x_i^{\max} > x_i^{\min}$ for $i = 1, \ldots, d$. Let $x^{\min} = (x_1^{\min}, \ldots, x_d^{\min})$ and $x^{\max} = (x_1^{\max}, \ldots, x_d^{\max})$. Then we denote $\lfloor x^{\min}, x^{\max} \rfloor$ as the domain. Let $\alpha = (\alpha_1, \ldots, \alpha_d)$ be a vector of nonnegative integers. Let $\mathcal{T}_\alpha(z)$ denote the product $\mathcal{T}_{\alpha_1}(z_1) \cdots \mathcal{T}_{\alpha_d}(z_d)$ for $z = (z_1, \ldots, z_d) \in [-1, 1]^d$. Let

$$Z(x) = \left(\frac{2x_1 - x_1^{\min} - x_1^{\max}}{x_1^{\max} - x_1^{\min}}, \ldots, \frac{2x_d - x_d^{\min} - x_d^{\max}}{x_d^{\max} - x_d^{\min}} \right)$$

for any $x = (x_1, \ldots, x_d) \in \lfloor x^{\min}, x^{\max} \rfloor$.

Using these notations, the degree-n complete Chebyshev approximation for $V(x)$ is

$$\hat{V}_n(x;\mathbf{b}) = \sum_{0\le|\alpha|\le n} b_\alpha T_\alpha(Z(x)), \tag{5}$$

where $|\alpha| = \sum_{i=1}^{d} \alpha_i$ for the nonnegative integer vector $\alpha = (\alpha_1, \ldots, \alpha_d)$. So the number of terms with $0 \le |\alpha| = \sum_{i=1}^{d} \alpha_i \le n$ is $\binom{n+d}{d}$ for the degree-n complete Chebyshev approximation in $\mathbb{R}^d$.

4 Nonlinear Programming Method to Solve Bellman Equations

There are many approaches to solve Bellman equations, such as value function iteration and policy iteration methods, or LP approaches. This section describes the general nonlinear programming method (DPNLP) to solve the Bellman equations (3) or (2).

4.1 Basic DPNLP

To solve the problem (3), we discretize the nonlinear approximation of the value function instead of the state and action spaces. The following nonlinear programming problem expresses one possible formulation of this method:

$$\begin{aligned}
\max_{a_i\in\mathcal{D}(x_i),x_i^+,v_i,\mathbf{b}} \quad & \sum_{i=1}^{m} v_i, \\
\text{s.t.} \quad & v_i \le u(x_i,a_i) + \beta\hat{V}(x_i^+;\mathbf{b}), \quad i=1,\ldots,m, \\
& x_i^+ = g(x_i,a_i), \quad i=1,\ldots,m, \\
& v_i = \hat{V}(x_i;\mathbf{b}), \quad i=1,\ldots,m,
\end{aligned}$$

where m is the number of the approximation nodes. In this method, the choice variables are the actions a, the next-stage states x^+, the value functions v, and the coefficients $\mathbf{b}$.

Unfortunately the solutions of the above model often has no shape properties, i.e., the value function is not increasing or concave. One approach to improve it is to add shape-preservation in the model. See Cai and Judd (2010, 2012a, 2012b) for the discussion of importance of shape-preservation

in DP. Now we have the basic DPNLP model:

$$\max_{a_i \in \mathcal{D}(x_i), x_i^+, v_i, \mathbf{b}} \quad \sum_{i=1}^{m} v_i, \tag{6}$$

$$\begin{aligned}
\text{s.t.} \quad & v_i \le u(x_i, a_i) + \beta \hat{V}(x_i^+; \mathbf{b}), \quad i = 1, \ldots, m, \\
& x_i^+ = g(x_i, a_i), \quad i = 1, \ldots, m, \\
& v_i = \hat{V}(x_i; \mathbf{b}), \quad i = 1, \ldots, m, \\
& \hat{V}'(y_{i'}; \mathbf{b}) \ge 0, \quad i' = 1, \ldots, m', \\
& \hat{V}''(y_{i'}; \mathbf{b}) \le 0, \quad i' = 1, \ldots, m',
\end{aligned}$$

where $\{y_{i'} : i' = 1, \ldots, m'\}$ are the set of shape nodes for shape preservation constraints. Usually the number of shape nodes, m', is more than the number of approximation nodes, m.

To solve the stochastic Bellman equation (2) where $\theta \in \Theta = \{\vartheta_j : j = 1, ..., J\}$, the basic DPNLP model becomes

$$\min_{a_{i,j} \in \mathcal{D}(x_i, \vartheta_j), x_{i,j}^+, v_i, \mathbf{b}} \quad \sum_{j=1}^{J} \sum_{i=1}^{m} v_{i,j},$$

$$\begin{aligned}
\text{s.t.} \quad & v_{i,j} \le u(x_i, a_{i,j}) + \beta \sum_{j'=1}^{J} P_{j,j'} \hat{V}(x_{i,j}^+, \vartheta_{j'}; \mathbf{b}), \\
& x_{i,j}^+ = g(x_i, \vartheta_j, a_{i,j}), \\
& v_{i,j} = \hat{V}(x_i, \vartheta_j; \mathbf{b}), \\
& \hat{V}'(y_{i'}, \vartheta_j; \mathbf{b}) \ge 0, \quad i' = 1, \ldots, m', \\
& \hat{V}''(y_{i'}, \vartheta_j; \mathbf{b}) \le 0, \quad i' = 1, \ldots, m', \\
& i = 1, \ldots, m, \ j = 1, ..., J,
\end{aligned}$$

where $P_{j,j'}$ is the conditional probability of $\theta^+ = \theta_{j'}$ given $\theta = \theta_j$, i.e., $P_{j,j'} = \Pr\left(\theta^+ = \theta_{j'} \mid \theta = \theta_j\right)$, for any $j, j' = 1, \ldots, J$.

4.2 Iterative DPNLP

One problem of the basic DPNLP model (6) is that an optimization solver often gives a solution where the equality in (7) does not hold while $\hat{V}'(y_{i'}; \mathbf{b}) \ge 0$ or $\hat{V}''(y_{i'}; \mathbf{b}) \le 0$ are binding at some shape nodes instead. However, the true solution of the basic DPNLP model (6) should let the inequality constraints

$$v_i \le u(x_i, a_i) + \beta \hat{V}(x_i^+; \mathbf{b}), \tag{7}$$

be binding for all $i = 1, \ldots, m$, and $\hat{V}(y_{i'}; \mathbf{b})$ should be strictly increasing and concave at all the shape nodes. We introduce an iterative DPNLP method to solve these problems.

In this paper, we use the Chebyshev polynomial approximation in $\hat{V}$. For a smooth function, we know that the Chebyshev polynomial approximation usually have a smaller coefficients in magnitude for higher-degree terms. This tells us that a small-degree Chebyshev polynomial approximation in $\hat{V}$ is a good initial guess for a higher-degree Chebyshev polynomial approximation. Another issue is that a quadratic Chebyshev polynomial approximation in $\hat{V}$ will be a good shape-preserving approximation with increasing and concave properties. Therefore, we have the following iterative DPNLP method to solve the infinite horizon deterministic optimal decision-making problems.

Algorithm 1 *Iterative DPNLP Method for Infinite Horizon Deterministic Optimal Decision-Making Problems*

Initialization. *Choose m expanded Chebyshev nodes $\{x_i : 1 \leq i \leq m\}$ on the range $[x_{\min}, x_{\max}]$ as the approximation nodes (with an odd number m), choose m' expanded Chebyshev nodes $\{y_i : 1 \leq i \leq m'\}$ on the range $[x_{\min}, x_{\max}]$ as the shape nodes, and choose the Chebyshev polynomial approximation for $\hat{V}(x; \mathbf{b})$ with degree n. Then solve the Basic DPNLP model (6) with degree-2 Chebyshev polynomial approximation. For a degree $n = 3, \ldots, m-1$, iterate through steps 1 and 2.*

Step *1. Use the solutions of the Basic DPNLP model (6) with degree $n-1$ Chebyshev polynomial approximation as the initial start point of the Basic DPNLP model (6) with degree n.*

Step *2. Use a reliable optimizer to solve the Basic DPNLP model (6) with degree n.*

It is easy to extend the algorithm to solve the infinite horizon stochastic and/or multidimensional optimal decision-making problems.

5 Applications to Deterministic Optimal Growth Problems

An infinite-horizon economic problem is the discrete-time optimal growth model with one good and one capital stock, which is a deterministic model[1]. The aim is to find the optimal consumption function and the optimal labor supply function such that the total utility over the infinite-horizon time is maximal, i.e.,

$$V(k_0) = \max_{c,l} \sum_{t=0}^{\infty} \beta^t u(c_t, l_t), \tag{8}$$
$$\text{s.t.} \quad k_{t+1} = F(k_t, l_t) - c_t,$$

[1] Please see Judd (1998) for a detailed description of this.

where k_t is the capital stock at time t with k_0 given in $[0.3, 2]$, c_t is the consumption, l_t is the labor supply, β is the discount factor, $F(k, l)$ is the aggregate production function, and $u(c_t, l_t)$ is the utility function.

In the examples, the aggregate production function is $F(k, l) = k + Ak^{\psi} l^{1-\psi}$ with $\psi = 0.25$ and $A = (1 - \beta)/(\psi\beta)$. The utility function is

$$u(c, l) = \frac{((c/A)^{1-\gamma} - 1}{1 - \gamma} - (1 - \psi)\frac{l^{1+\eta} - 1}{1 + \eta}. \tag{9}$$

The functional forms for utility and production imply that the steady state of the infinite horizon deterministic optimal growth problems is $k_{ss} = 1$, and the optimal consumption and the optimal labor supply at k_{ss} are respectively $c_{ss} = A$ and $l_{ss} = 1$. The code for DPNLP is written in GAMS (McCarl, 2011), and the optimization solver is CONOPT (in the GAMS environment).

5.1 True Solution

In order to estimate the accuracy of solution given by DPNLP, we compute the "true" optimal solution on a large set of test points for initial capital $k_0 \in [0.3, 2]$, and then compare those results with the computed optimal solution from DPNLP. To get the "true" optimal solution, we discretize the range of capital, $[0.3, 2]$, with one million equally-spaced capital nodes, and also discretize the range of labor supply, $[0.4, 2.5]$, with another one million equally-spaced labor supply nodes. for a discrete capital node k among the one million capital nodes and a discrete labor supply node l among the one million labor supply nodes, we choose consumption $c = F(k, l) - k^+$ such that k^+ is also one node among the one million capital nodes. Then using the one million capital nodes as discrete states, we apply the alternating sweep Gauss-Seidel algorithm (Judd, 1998) to compute the optimal value function until it converges under the stopping criterion 10^{-7}.

5.2 DPNLP Solution

We use the iterative DPNLP method (Algorithm 1) to solve the deterministic optimal growth problem. The basic DPNLP model is

$$\max_{c,l,k^+,v,\mathbf{b}} \sum_{i=1}^{m} v_i, \tag{10}$$

$$\begin{aligned}
\text{s.t.} \quad & v_i \le u(c_i, l_i) + \beta \hat{V}(k_i^+; \mathbf{b}), \quad i = 1, \ldots, m, \\
& k_i^+ \le F(k_i, l_i) - c_i, \quad i = 1, \ldots, m, \\
& v_i = \hat{V}(k_i; \mathbf{b}), \quad i = 1, \ldots, m, \\
& \hat{V}'(y_{i'}; \mathbf{b}) \ge 0, \quad i' = 1, \ldots, m', \\
& \hat{V}''(y_{i'}; \mathbf{b}) \le 0, \quad i' = 1, \ldots, m'.
\end{aligned}$$

For our examples in this section, we always choose $m = 19$ expanded Chebyshev nodes, k_i, in $[0.3, 2]$, as the approximation nodes, and the approximation method, $\hat{V}$, is the expanded Chebyshev polynomial up to the maximal degree 18, and we choose $m' = 100$ expanded Chebyshev nodes, $y_{i'}$, in $[0.3, 2]$, as the shape nodes. In fact, in some cases among our examples, we could use less numbers to save computational time but with almost the same accuracy.

5.3 Error Analysis of DPNLP Solution

We next use some basic examples of the deterministic optimal growth problem to test DPNLP. We tries $\beta = 0.9, 0.95, 0.99$, $\gamma = 0.5, 2, 8$, and $\eta = 0.2, 1, 5$, all these examples give us good solutions.

Table 1 lists relative errors of optimal solutions computed by DPNLP for these cases in comparison with the "true" solution given by the high-precision discretization method. The errors for optimal consumptions are computed by

$$\max_{k\in[0.3,2]} \frac{|c^*_{\text{DPNLP}}(k) - c^*(k)|}{|c^*(k)|},$$

where $c^*_{\text{DPNLP}}(k)$ is the optimal consumption computed by DPNLP, and $c^*(k)$ is the "true" optimal consumption, for $k \in [0.3, 2]$. The errors for optimal labor supply, l^*_{DPNLP}, have the similar computation formula. The last column of Table 1 lists the running time of the iterative DPNLP method for various cases in the GAMS environment, on a single core of a Mac laptop with a 2.5 GHz processor.

Table 1 shows that DPNLP solves the examples with accuracy up to 5 digits or higher for optimal control policy functions in all cases. Moreover, the DPNLP method is fast and takes only several seconds for each case For example, row one in Table 1 assumes $\beta = 0.9$, $\gamma = 0.5$, and $\eta = 0.2$. For this case, the error in consumption is 1.5×10^{-6}, the error in labor supply is 1.8×10^{-6}, and the running time is only 6.8 seconds.

6 Applications to Stochastic Optimal Growth Problems

When the capital stock is dependent on a random economic shock θ_t, the optimal growth problem (8) becomes a stochastic dynamic optimization problem. Assume that the random economic shock θ_t is a stochastic process following $\theta_{t+1} = h(\theta_t, \varepsilon_t)$, where ϵ_t is a serially uncorrelated random process. Let $f(k, l, \theta)$ denote net production function, and $F(k, l, \theta) = k + f(k, l, \theta)$. Then the infinite-horizon discrete-time stochastic optimization problem be-

Table 1: Relative Errors of DPNLP for Deterministic Optimal Growth Problems

β	γ	η	Error of c^*_{DPNLP}	Error of l^*_{DPNLP}	Time (seconds)
0.9	0.5	0.2	1.5(−6)	1.8(−6)	6.8
		1	3.1(−6)	1.5(−6)	3.8
		5	3.0(−6)	1.1(−6)	4.4
	2	0.2	1.1(−6)	3.6(−6)	4.3
		1	1.4(−6)	2.3(−6)	7.0
		5	2.2(−6)	1.2(−6)	4.5
	8	0.2	9.7(−6)	3.7(−6)	5.7
		1	1.0(−6)	2.6(−6)	3.9
		5	1.5(−6)	3.5(−6)	3.8
0.95	0.5	0.2	3.1(−6)	3.7(−6)	3.8
		1	4.7(−6)	1.9(−6)	3.7
		5	4.8(−6)	1.2(−6)	3.4
	2	0.2	1.6(−6)	5.8(−6)	4.2
		1	2.2(−6)	3.4(−6)	4.3
		5	3.5(−6)	1.9(−6)	3.7
	8	0.2	1.2(−6)	6.7(−6)	4.6
		1	1.2(−6)	5.2(−6)	4.3
		5	2.8(−6)	4.8(−6)	4.3
0.99	0.5	0.2	1.2(−5)	1.3(−5)	4.8
		1	3.0(−5)	1.1(−5)	4.8
		5	4.2(−5)	4.3(−6)	3.9
	2	0.2	6.1(−6)	2.4(−5)	5.7
		1	1.0(−5)	1.6(−5)	5.3
		5	1.8(−5)	7.7(−6)	5.6
	8	0.2	2.0(−6)	3.2(−5)	7.4
		1	3.9(−6)	2.2(−5)	6.3
		5	1.1(−5)	1.6(−5)	6.9

Note: $a(k)$ means $a \times 10^k$.

comes

$$V(k_0,\theta_0)=\max_{k,c,l}\ \mathbb{E}\left\{\sum_{t=0}^{\infty}\beta^t u(c_t,l_t)\right\}, \tag{11}$$
$$\text{s.t.}\quad k_{t+1}=F(k_t,l_t,\theta_t)-c_t,$$
$$\theta_{t+1}=h(\theta_t,\varepsilon_t),$$

where $k_0 \in [0.3, 2]$ and θ_0 are given. The parameter θ has many economic interpretations. In the life-cycle interpretation, θ is a state variable that may affect either asset income, labor income, or both. In the monopolist interpretation, θ may reflect shocks to costs, demand, or both.

We use the same utility function (9), but the production function is changed to

$$F(k,l,\theta)=k+\theta A k^{\psi} l^{1-\psi}$$

where θ is the stochastic state, $\psi = 0.25$ and $A = (1-\beta)/(\psi\beta)$. In the examples, θ_t is assumed to be a Markov chain with 3 possible values:

$$\vartheta_1 = 0.95,\ \vartheta_2 = 1.0,\ \vartheta_3 = 1.05,$$

and the probability transition matrix from θ_t to θ_{t+1} is

$$P = \begin{bmatrix} 0.75 & 0.25 & 0 \\ 0.25 & 0.5 & 0.25 \\ 0 & 0.25 & 0.75 \end{bmatrix}.$$

The code for DPNLP is written in GAMS (McCarl, 2011), and the optimization solver is CONOPT (in the GAMS environment).

6.1 True Solution

For the deterministic optimal growth problem (8), we use the discretized method and the alternating sweep Gauss-Seidel algorithm to get the "true" solution. But the DP method with high-precision discretization will be too time-consuming for solving the stochastic optimal growth problem (11). However, Cai and Judd (2012a) introduces a value function iteration method using a shape-preserving rational spline interpolation and shows that it is very accurate for solving multi-period portfolio optimization problems. For the deterministic optimal growth problem, since the value function is smooth, increasing and concave over the continuous state, capital k, we can also apply this shape-preserving DP algorithm to solve the deterministic optimal growth problem and realize that it is also very accurate (by comparing its solution with those given by the alternating sweep Gauss-Seidel algorithm).

For the stochastic optimal growth problem, the value function for each discrete state is also smooth, increasing and concave over the continuous

state, capital k. Therefore, we can again choose the shape-preserving value function iteration method to solve the stochastic optimal growth problem and iterates until it converges under the stopping criterion 10^{-7}. We use 1000 equally-spaced interpolation nodes on the range of the continuous state, $[0.3, 2]$, for each discrete state θ.

6.2 DPNLP Solution

We use the iterative DPNLP method (stochastic version of Algorithm 1) to solve the stochastic optimal growth problem. The basic DPNLP model is

$$\max_{c,l,k^+,v,\mathbf{b}} \quad \sum_{j=1}^{J}\sum_{i=1}^{m} v_{i,j}, \tag{12}$$

$$\begin{aligned}
\text{s.t.} \quad & v_{i,j} \le u(c_{i,j}, l_{i,j}) + \beta \sum_{j'=1}^{J} P_{j,j'} \hat{V}(k^+_{i,j}, \theta_{j'}; \mathbf{b}), \\
& k^+_{i,j} \le F(k_i, l_{i,j}, \theta_j) - c_{i,j}, \\
& v_{i,j} = \hat{V}(k_i, \theta_j; \mathbf{b}), \\
& \hat{V}'(y_{i'}, \theta_j; \mathbf{b}) \ge 0, \\
& \hat{V}''(y_{i'}, \theta_j; \mathbf{b}) \le 0, \\
& i = 1, \ldots, m, \ j = 1, \ldots, J, \ i' = 1, \ldots, m',
\end{aligned}$$

where $J = 3$, $m = 19$, $m' = 100$, k_i are expanded Chebyshev nodes in $[0.3, 2]$, $\hat{V}$ is the expanded Chebyshev polynomial up to the maximal degree 18, and $y_{i'}$ are expanded Chebyshev nodes in $[0.3, 2]$ as the shape nodes.

6.3 Error Analysis of DPNLP Solution

We examine the errors for the stochastic model in the similar manner we did for the deterministic optimal growth problems: We apply the high-precision value function iteration to get the "true" optimal solution for every test point of initial capital k_0 and every possible initial discrete state θ_0, and then use them to check the accuracy of the computed optimal solution from the DPNLP model (12).

Table 2 lists relative errors of optimal solutions computed by DPNLP for the stochastic optimal growth problem with the following cases: $\beta = 0.9, 0.95, 0.99$, $\gamma = 0.5, 2, 8$, and $\eta = 0.2, 1, 5$. The errors for optimal consumptions at time 0 are computed by

$$\max_{k\in[0.3,2],\theta\in\{0.95,1.0,1.05\}} \frac{\|c^*_{\text{DPNLP}}(k,\theta) - c^*(k,\theta)\|}{\|c^*(k,\theta)\|},$$

where c^*_{DPNLP} is the optimal consumption computed by DPNLP on the model (12), and c^* is the "true" optimal consumption computed by the high-precision value function iteration method. The similar formula applies to

Table 2: Relative Errors of DPNLP for Stochastic Optimal Growth Problems

β	γ	η	Error of c^*_{DPNLP}	Error of l^*_{DPNLP}	Time (seconds)
0.9	0.5	0.2	1.9(−7)	5.2(−7)	11
		1	2.5(−7)	4.5(−7)	9
		5	2.5(−7)	4.7(−7)	9
	2	0.2	1.4(−7)	5.0(−7)	16
		1	2.0(−7)	5.9(−7)	12
		5	3.0(−7)	4.4(−7)	12
	8	0.2	1.1(−7)	8.4(−7)	22
		1	1.6(−7)	8.8(−7)	18
		5	8.5(−7)	1.2(−6)	15
0.95	0.5	0.2	3.7(−7)	4.8(−7)	15
		1	3.9(−7)	4.2(−7)	11
		5	4.4(−7)	4.4(−7)	10
	2	0.2	2.9(−7)	6.6(−7)	22
		1	3.2(−7)	5.9(−7)	17
		5	4.4(−7)	4.4(−7)	13
	8	0.2	2.3(−7)	9.6(−7)	25
		1	3.0(−7)	8.7(−7)	21
		5	9.7(−7)	1.3(−6)	23
0.99	0.5	0.2	4.1(−7)	6.1(−7)	30
		1	4.5(−7)	4.6(−7)	22
		5	4.1(−7)	4.6(−7)	17
	2	0.2	3.0(−7)	1.1(−6)	50
		1	3.4(−7)	7.4(−7)	40
		5	5.9(−7)	5.4(−7)	40
	8	0.2	1.5(−7)	1.5(−6)	55
		1	1.8(−7)	1.3(−6)	58
		5	2.2(−6)	3.1(−6)	56

Note: $a(k)$ means $a \times 10^k$.

compute errors for optimal labor supply. The last column of Table 2 lists the running time of the iterative DPNLP method for various cases in the GAMS environment, on a single core of a Mac laptop with a 2.5 GHz processor.

From Table 2, we can also see the similar pattern shown in Table 1. That is, DPNLP solves the examples with accuracy up to 6 or higher digits for optimal control policy functions in all cases. Moreover, for these stochastic examples, DPNLP is also fast, and takes less than one minute to solve any one case. For example, row one in Table 2 assumes $\beta = 0.9$, $\gamma = 0.5$, and $\eta = 0.2$. For this case, the error in consumption is 1.9×10^{-7}, the error in labor supply is 5.2×10^{-7}, and the running time is only 11 seconds.

7 Applications to Two-Dimensional Optimal Growth Problem

The key DPNLP idea is clearly applicable to multidimensional problems. Of course, multidimensional problems are more demanding. Our next example illustrates DPNLP applied to a two-dimensional extension of our earlier models. The results indicate that DPNLP is a reasonable method for low-dimensional problems.

We assume that there are two countries, and let $k_t = (k_{t,1}, k_{t,2})$ denote the capital stocks of two countries which is a two-dimensional continuous state vector at time t. Let $l_t = (l_{t,1}, l_{t,2})$ denote elastic labor supply levels of the countries which is a two-dimensional continuous control vector variable at time t. Assume that the net production of country i at time t is

$$f_i(k_{t,i}, l_{t,i}) = A k_{t,i}^{\psi} l_{t,i}^{1-\psi},$$

with $A = (1-\beta)/(\psi\beta)$, for $i = 1, 2$. Let $c_t = (c_{t,1}, c_{t,2})$ denote consumption of the countries which is another two-dimensional continuous control vector variable at time t. The utility function is

$$u(c, l) = \sum_{i=1}^{2} \left[\frac{(c_i/A)^{1-\gamma} - 1}{1-\gamma} - (1-\psi) \frac{l_i^{1+\eta} - 1}{1+\eta} \right].$$

We want to find an optimal consumption and labor supply decisions such that expected total utility over the infinite-horizon time is maximized. That

is,

$$
\begin{aligned}
V(k_0) \;=\; \max_{k_t, I_t, c_t, l_t} \; & \sum_{t=0}^{\infty} \beta^t u(c_t, l_t) \qquad (13)\\
\text{s.t.} \quad & k_{t+1,i} = (1-\delta)k_{t,i} + I_{t,i},\\
& \Gamma_{t,i} \equiv \frac{\zeta}{2} k_{t,i} \left(\frac{I_{t,i}}{k_{t,i}} - \delta \right)^2,\\
& \sum_{i=1}^{2} (c_{t,i} + I_{t,i} - \delta k_{t,i}) \equiv \sum_{i=1}^{2} (f_i(k_{t,i}, l_{t,i}) - \Gamma_{t,i}),
\end{aligned}
$$

where δ is the depreciation rate of capital, $I_{t,i}$ is the investment of country i, $\Gamma_{t,i}$ is the investment adjustment cost of country i, and ζ governs the intensity of the friction. Detailed discussion of multi-country growth models with infinite horizon can be seen in Den Haan et al (2011) and Juillard and Villemot (2011). For the multi-country growth models with finite horizon, they can be solved efficiently using dynamic programming with Hermite approximation, see Cai and Judd (2012c). In our examples, we let $\psi = 0.36$, $\delta = 0.025$, and $\zeta = 0.5$.

The functional forms for utility and production imply that the steady state of the infinite horizon deterministic optimal growth problems is $k_{ss,1} = k_{ss,2} = 1$, and the optimal consumption, labor supply and investment at the steady state are respectively $c_{ss,1} = c_{ss,2} = A$, $l_{ss,1} = l_{ss,2} = 1$, and $I_{ss,1} = I_{ss,2} = \delta$. The code for DPNLP is written in GAMS (McCarl, 2011), and the optimization solver is CONOPT (in the GAMS environment).

7.1 True Solution

Discretization method will be too time-consuming to solve the two-country optimal growth problem with two continuous state variables $(k_{t,1}, k_{t,2})$ and six continuous control variables $(c_{t,1}, c_{t,2}, l_{t,1}, l_{t,2}, I_{t,1}, I_{t,2})$. In order to get the "true" solution, we use the value function iteration with high-degree complete Chebyshev polynomials and iterates until it converges under the stopping criterion 10^{-7} (i.e., the difference between two consecutive value functions is less than 10^{-7}). We use 51^2 tensor Chebyshev nodes on the state space $[0.5, 1.5]^2$, and the degree of the complete Chebyshev polynomials is 30.

7.2 DPNLP Solution

We use the iterative DPNLP method (multidimensional version of Algorithm 1) to solve the two-dimensional optimal growth problem. The basic DPNLP

model is the multidimensional extension of the model (10). That is,

$$\max_{c,l,I,k^+,v,\mathbf{b}} \sum_{i=1}^{m} v_i, \tag{14}$$

$$\text{s.t.} \quad v_i \le u(c_i, l_i) + \beta \hat{V}(k_i^+; \mathbf{b}), \quad i = 1, \ldots, m,$$

$$k_{i,j}^+ \le (1-\delta) k_{i,j} + I_{i,j}, \quad i = 1, \ldots, m, \quad j = 1, 2,$$

$$\Gamma_{i,j} \equiv \frac{\gamma}{2} k_{i,j} \left(\frac{I_{i,j}}{k_{i,j}} - \delta \right)^2, \quad i = 1, \ldots, m, \quad j = 1, 2,$$

$$\sum_{j=1}^{2} (c_{i,j} + I_{i,j} - \delta k_{i,j}) = \sum_{j=1}^{2} (f_j(k_{i,j}, l_{i,j}) - \Gamma_{i,j}), \quad i = 1, \ldots, m,$$

$$v_i = \hat{V}(k_i; \mathbf{b}), \quad i = 1, \ldots, m,$$

$$\hat{V}'(y_{i'}; \mathbf{b}) \ge 0, \quad i' = 1, \ldots, m',$$

$$\hat{V}''(y_{i'}; \mathbf{b}) \le 0, \quad i' = 1, \ldots, m',$$

where $k_i = (k_{i,1}, k_{i,2})$, $c_i = (c_{i,1}, c_{i,2})$, $l_i = (l_{i,1}, l_{i,2})$, $k_i^+ = (k_{i,1}^+, k_{i,2}^+)$, $y_{i'} = (y_{i',1}, y_{i',2})$, and $\hat{V}'$ is the 2-dimensional gradient of $\hat{V}$, and $\hat{V}''$ is the 2-dimensional second order derivatives of $\hat{V}$.

For our examples in this section, we choose $m = 11^2$ tensor Chebyshev nodes in the state space $[0.5, 1.5]^2$, as the approximation nodes. The approximation method, $\hat{V}$, is the complete Chebyshev polynomial up to the maximal degree 10. And we choose $m' = 100$ tensor Chebyshev nodes, $y_{i'}$, in the state space $[0.5, 1.5]^2$, as the shape nodes.

7.3 Error Analysis of DPNLP Solution

We examine the errors for the multidimensional model in the similar manner we did for the unidimensional optimal growth problems: We apply the high-precision value function iteration to get the "true" optimal solution for every test point of initial capitals, and then use them to check the accuracy of the computed optimal solution from the DPNLP model (12).

Table 3 lists relative errors of optimal solutions computed by DPNLP for the two-dimensional optimal growth problem with the following cases: $\beta = 0.95$, $\gamma = 0.5, 2, 8$, and $\eta = 0.2, 1, 5$. The last column of Table 3 lists the running time of the iterative DPNLP method for various cases in the GAMS environment, on a single core of a Mac laptop with a 2.5 GHz processor.

Table 3 shows that DPNLP solves the examples with accuracy up to 4 digits or higher for optimal control policy functions in all the cases besides one case having 3 digits. Moreover, the DPNLP method is not slow and takes only several minutes for each case. For example, row one in Table 3 assumes $\gamma = 0.5$, and $\eta = 0.2$. For this case, the error in consumption is 6.4×10^{-5}, the error in labor supply is 1.5×10^{-4}, and the running time is 3.9 minutes.

Table 3: Relative Errors of DPNLP for Two-Country Optimal Growth Problems

γ	η	Error of c^*_{DPNLP}	Error of l^*_{DPNLP}	Time (minutes)
0.5	0.2	6.4(−5)	1.5(−4)	3.9
	1	9.0(−6)	3.0(−5)	2.5
	5	8.0(−6)	8.0(−7)	1.4
2	0.2	6.2(−5)	2.3(−4)	3.6
	1	4.5(−5)	6.5(−5)	3.1
	5	8.5(−5)	3.2(−5)	2.2
8	0.2	8.4(−5)	1.2(−3)	5.0
	1	2.1(−5)	1.1(−4)	6.6
	5	1.3(−4)	2.0(−4)	5.6

Note: $a(k)$ means $a \times 10^k$.

8 Conclusion

This paper presents a nonlinear programming formulation of dynamic programming problems common in economic decision making. We have applied it to a variety of optimal accumulation problems, showing that our DPNLP method performs very well, with high accuracy, reliability and efficiency for those problems. The variety of example problems indicate that our DPNLP method could be applied to many problems.

References

[1] Bellman, R. (1957). Dynamic Programming. Princeton University Press.

[2] Cai, Y. (2010). Dynamic Programming and Its Application in Economics and Finance. PhD thesis, Stanford University.

[3] Cai, Y., and K.L. Judd (2010). Stable and efficient computational methods for dynamic programming. Journal of the European Economic Association, Vol. 8, No. 2-3, 626–634.

[4] Cai, Y., and K.L. Judd (2012a). Dynamic programming with shape-preserving rational spline Hermite interpolation. Economics Letters, Vol. 117, No. 1, 161–164.

[5] Cai, Y., and K.L. Judd (2012b). Shape-preserving dynamic programming. Mathematical Methods of Operations Research, DOI: 10.1007/s00186-012-0406-5.

[6] Cai, Y., and K.L. Judd (2012c). Dynamic programming with Hermite approximation. NBER working paper No. w18540.

[7] De Farias, D.P., and B. Van Roy (2003). The linear programming approach to approximate dynamic programming. Operations Research, 51(6), 850–865.

[8] Den Haan, W.J., K.L. Judd and M. Juillard (2011). Computational suite of models with heterogeneous agents II: Multi-country real business cycle models. Journal of Economic Dynamics & Control, 35, 175–177.

[9] Judd, K.L. (1998). Numerical Methods in Economics. The MIT Press.

[10] Juillard, M., and S. Villemot (2011). Multi-country real business cycle models: Accuracy tests and test bench. Journal of Economic Dynamics & Control, 35, 178–185.

[11] McCarl, B., et al. (2011). McCarl GAMS User Guide. GAMS Development Corporation.

[12] Michelangeli, V. (2009). Economics of the Life-Cycle: Reverse Mortgage, Mortgage and Marriage. PhD thesis, Boston University.

[13] Rust, J. (2008). Dynamic Programming. In: Durlauf, S.N., Blume L.E. (Eds.), New Palgrave Dictionary of Economics. Palgrave Macmillan, second edition.

[14] Stokey, N.L., R.E. Lucas, and E.C. Prescott (1989). Recursive Methods in Economic Dynamics. Harvard University Press.

[15] Su, C.H, and K.L. Judd (2012). Constrained Optimization Approaches to Estimation of Structural Models. Econometrica, Vol. 80 (5), 2213–2230.

[16] Trick, M.A., and S.E. Zin (1997). Spline approximations to value functions — linear programming approach. Macroeconomic Dynamics, 1, 255–277.