PERGAMON

Computers & Operations Research 27 (2000) 399–408

computers & operations research

www.elsevier.com/locate/orms

# Solving a savings allocation problem by numerical dynamic programming with shape-preserving interpolation

Sheng-Pen Wang[a,*], Kenneth L. Judd[b]

[a]*Department of Business Administration, Chang Gung University, Taoyuan 33302, Taiwan*
[b]*Hoover Institution, Stanford University, Stanford, CA 94305, USA*

Received 1 October 1998; received in revised form 1 January 1999

**Abstract**

This article introduces a bivariate shape-preserving interpolation algorithm to approximate the value function of a dynamic program. First, we present a savings allocation problem between a pension account and another non-pension one. With the objective of maximizing the present value of utility over a life cycle, the investor can distribute his or her savings, in each account, between stocks and cash funds. Formally, this complex problem involved with various tax rules is in dynamic programming formulation and can only be solved numerically. It is known that the value function of the associated two-dimensional dynamic program inherits monotonicity and convexity of the investor's risk-averse utility function. To preserve these shape characteristics, we apply a bivariate shape-preserving interpolation algorithm in the successive approximation of the value function. Finally, we have computational results for this savings allocation problem, showing that the proposed shape-preserving interpolation method is superior to other dynamic programming methods with less sophisticated interpolation techniques.

**Scope and purpose**

The savings allocation problems with several dimensions of continuous states are too complicated and thus can only be solved by numerical dynamic programming. Theory of dynamic programming has shown that the associated value function inherits the shape characteristics – monotonicity and concavity – of a risk-averse investor's utility function. However, there are no numerical methods which guarantee to preserve these shape features in the course of approximation of the value function. In this article, we model a savings allocation problem as a two-dimensional dynamic program and we present a bivariate shape-preserving interpolation method to solve it. 

*Keywords:* Dynamic programming; Savings allocation; Interpolation

* Corresponding author. Tel.: + 886-3-238-3016-x5413; fax: + 886-3-327-1304.
*E-mail address:* wangsp@mail.cgu.edu.tw (S.-P. Wang)

0305-0548/00/$ - see front matter 
PII: S0305-0548(99)00044-1

## 1. Introduction

Tax policy discussions often concern how taxes affect savings and what policies can encourage investment. Two such examples are the debates about pension savings accounts and the taxation of capital gains from investment; see, for example, Shoven and MaCurdy [1]. Any analysis of these issues is seriously hindered by the complexity that tax rules generate. For example, pension savings are tax-favored and illiquid and thus individuals may manipulate the realization of capital gains to either minimize tax payments or maximize the present value of utility. These complexities imply that behavior is not just a function of market value of one's portfolio but also depends on the history of investment returns, rates of consumption, and the distribution of wealth between tax-favored pension accounts and ordinary savings accounts.

In this paper we aim to investigate an individual optimal policy of consumption stream and allocation of investment between a tax-favored pension savings account and a ordinary, taxable savings account. In each of these two accounts, the individual decides his portfolios between stocks and cash funds. The optimization criterion is to maximize the present value of utility over consumption across the investor's life cycle. We use an increasing and concave utility function to represent the individual risk-averse preference. Formally, this problem is a dynamic programming problem and can only be solved numerically. Indeed, according to Kendrick [2], the only way to rigorously analyze investor behavior faced with various tax rules and different consumption patterns is numerically. We model this savings allocation problem as a two-dimensional dynamic program and iteratively solve this finite-horizon problem, starting at the termination time (the last period of individual life) and working backwards to the beginning. The best approximate investment policy for this savings allocation problem can be derived from the optimal control variables while computing the associated value function.

It is important, however, to note that the risk-averse utility function has particular shape characteristics – monotonicity and concavity – over the consumption domain. Under fairly standard conditions as indicated in any dynamic-programming text such as Stokey and Lucas [3], the value function inherits these shape features. As the state variables in this investment allocation problem – savings in the pension account and that in the non-pension account – are naturally continuous, we have to apply approximation methods in computing the discretized value function. To our best knowledge, however, no approximation methods guarantee to preserve the shape characteristics of value function data; the monotonicity and concavity of the value function are deformed in the course of approximation, errors propagate and hence the approximate solution is questionably acceptable. To preserve these shape features of the associated value function of this savings allocation problem, we propose to apply the Costantini [4] bivariate shape-preserving interpolation algorithm in the backward computation of the value iterations.

The paper is structured as follows. In Section 2, we describe the savings allocation problem and its associated two-dimensional dynamic programming formulation. Section 3 outlines the procedures of Costantini's bivariate interpolation and its special properties of shape preservation. Section 4 reports the numerical results of the savings allocation problem solved by shape-preserving approximation. Some concluding remarks follow in the final section.

## 2. The savings allocation problem

Suppose a worker earns before-tax wage $w_t$ in period $t$ for $t = 1, \ldots, T$ and is retired for $t = T+1, \ldots, D$ (could be life expectancy). His utility function over consumption, $u(c)$, is age independent. Suppose that in each period the worker decides to allocate his savings after consumption in a pension account and a conventional taxable account. The worker cannot borrow from the pension account, but he can do so from the non-pension account. In each period he also has to choose the investment portfolio of each account between stocks and cash funds. His objective is to choose the stream of optimal consumption $\{c_t\}$, contributions to the pension account $\{x_t\}$, savings to the non-pension account $\{y_t\}$, and portfolios between stocks and cash funds to maximize the present value of his life-cycle utility with a time-preference discount factor $0 < \beta < 1$.

To make this problem less complicated, we first specify the notation as follows:

$x_t$: contributions to pension savings in period $t$, with restriction $0 \leqslant x_t \leqslant p \cdot w_t$, when $1 \leqslant t \leqslant T$, where $p$ is the maximal fraction of income allowed to contribute to the pension account; here we assume there is no early withdrawal before retirement.

$y_t$: contributions to the non-pension account in period $t$, might be negative to represent borrowing.

$X_t$: wealth accumulation in the pension account at the end of period $t$.

$Y_t$: wealth accumulation in the non-pension account at the end of period $t$.

$\theta_t$: fraction of pension savings invested in stocks carried over the period $t$.

$\varphi_t$: fraction of non-pension savings in stocks carried over the period $t$.

$\tilde{z}_t$: identically and independently distributed rate of return of stocks.

$r_t$: nominal rate of return on cash funds.

$\tau_t^w$: combined state and federal marginal tax rate on wage in period $t$.

$\tau_t^r$: combined marginal tax rate on cash fund yield in period $t$.

$\tau_t^z$: marginal tax rate on return (such as dividends and capital gains) of stock investments in period $t$. Here we focus only on the special case that dividend tax rate is the same as capital gains tax rate.

Using the above notation and $E$, the expectation with respect to the stochastic process $\{\tilde{z}_t\}$, we can formulate the savings allocation problem in the form of non-linear programming with the objective

$$\max_{\{c_t, x_t, y_t, \theta_t, \varphi_t\}} E\left\{ \sum_{t=1}^{D} \beta^{t-1} u(c_t) \right\}$$

and four constraints:

(1) $c_t = w_t - x_t - y_t - \tau_t^w(w_t - x_t) \geqslant 0$, $t = 1, \ldots, T, \ldots, D$. (This is consumption–savings dynamics; here we assume the interest rates of return and of borrowing from the non-pension account are equal.)

(2) $X_t = X_{t-1} \cdot (\theta_t(1 + \tilde{z}_t) + (1 - \theta_t)(1 + r_t)) + x_t \geqslant 0$, $t = 1, \ldots, T, \ldots, D$. (This represents the dynamics of wealth accumulation in pension savings. Note that investment profit in the pension account is tax-free. Let $X_0 = 0$ indicate zero initial pension savings.)

(3) $Y_t = Y_{t-1} \cdot \{\varphi_t(1 + \tilde{z}_t(1 - \tau_t^z)) + (1 - \varphi_t)(1 + r_t(1 - \tau_t^r))\} + y_t \geqslant 0$, $t = 1, \ldots, D$. (This is the dynamics of wealth accumulation in the non-pension account. Taxes are levied on investment returns of non-pension savings. Also let $Y_0 = 0$ represent no initial wealth therein.)

(4) $0 \leqslant \theta_t \leqslant 1$, and $-1 \leqslant \varphi_t \leqslant 2$. (In the non-pension account, leverage in investment between stocks and cash funds is allowed up to the double of wealth accumulation at the beginning of period $t$.)

Most optimization software packages can handle the above non-linear program, but they will fail once the time horizon of the planning problem increases and the number of control variables grows. On the other hand, due to its strong recursive property, this problem can be easily treated by the dynamic programming technique. Dynamic programming is a powerful optimization procedure since it exploits the sequential character of the operation of dynamic models and allows non-linearities and stochastic inputs to be represented. It uses the value function to decentralize a complicated stochastic/multiperiod optimization problem into a sequence of simpler deterministic/static optimization problems.

The Bellman [5] equation provides the necessary and sufficient condition for optimality in the dynamic programming formulation. To start with it, we have to specify the state variables. The wealth accumulation levels at time period $t$, $(X_t,\ Y_t)$, are the natural choices. If we define the value iteration at the end of period $t$ as

$$V^t(X_t,\ Y_t) = \max E\{u(c_{t+1}) + \cdots + \beta^{D-t-1}u(c_D)\} = \max_{\{c_s, x_s, y_s, \theta_s, \varphi_s\}} E\left\{\sum_{s=1}^{D-t} \beta^{s-1}u(c_{t+s})\right\}, \tag{1}$$

then Bellman's equation for this finite-horizon dynamic program has the form

$$V^t(X_t,\ Y_t) = \max_{c_{t+1}}\ u(c_{t+1}) + \beta\{V^{t+1}(X_{t+1},\ Y_{t+1}) \,|\, X_t,\ Y_t\}, \quad t = 0,\ 1, \ldots,\ D-1, \tag{2}$$

with the terminal condition

$$V^D(X_D,\ Y_D) \equiv 0. \tag{3}$$

Here we impose the constraint (3) to force the worker to spend all his wealth at the last period. Following the definition (1), we can rewrite the objective of the savings allocation problem as the value function at time period 0; that is

$$\max_{\{c_t, x_t, y_t, \theta_t, \varphi_t\}} E\left\{\sum_{t=1}^{D} \beta^{t-1}u(c_t)\right\} = V^0(X_0,\ Y_0). \tag{4}$$

Starting with terminal condition (3), we shall solve backwards the value iterations – those recursive equations in (2) – to find value function (4) defined on the two-dimensional space $X \times Y$ and its corresponding optimal control variables. As we typically compute the value function at a finite collection of points in $X \times Y$ and make a guess about its value elsewhere, we need a smooth interpolation method beforehand to make it work effectively the optimization step performed in (2).

## 3. The shape-preserving interpolation

An interpolation method is shape-preserving means that the interpolation preserves the shape characteristics, particularly, the monotonicity and concavity, of interpolating data. Since monotonicity/concavity for bivariate functions is somewhat less clear than for univariate functions, to

apply a bivariate shape-preserving interpolation method in solving value iterations (2) and (3), we have to clarify the definition of the shape feature in two-dimensional spaces. Let $f(x, y)$ be a bivariate function defined on $D = [a, b] \times [c, d]$ and suppose that both points $z_1 = (x_1, y_1)$ and $z_2 = (x_2, y_2)$ belong to $D$. We call $f$ a monotone increasing function on $D$ if

$$f(x_2, y_2) \geqslant f(x_1, y_1),$$

whenever $x_2 \geqslant x_1$ and $y_2 \geqslant y_1$; and we call $f$ a concave function on $D$ if

$$f(\alpha z_1 + (1 - \alpha)z_2) \geqslant \alpha f(z_1) + (1 - \alpha)f(z_2)$$

for any $\alpha \in (0, 1)$.

In the last two decades, many papers dealing with surface fitting and with shape-preserving interpolation have appeared in the literature, but few have been concerned with the intersection of these two fields, that is, with bivariate shape-preserving interpolation. Let

$$\{(x_i, y_j, f_{ij}) : x_i < x_{i+1}, y_j < y_{j+1}, i = 0, 1, \ldots, N, j = 0, 1, \ldots, M\} \tag{5}$$

be a given set of data, where $f_{ij} = f(x_i, y_j)$. Dodd et al. [6] used one-dimensional shape-preserving curves along the grid lines and constructed rectangular patches. Although this method produces visually pleasing plots, it does not guarantee that the surface will have the same shape of the data inside the rectangles $R_{i,j} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$. On the other hand, Costantini and Fontanella [7] gave the first method for constructing shape-preserving surfaces which interpolate arbitrary sets of data on rectangular grids using additional information

$$\{((f_x)_{i,j}, (f_y)_{i,j}, (f_{xy})_{i,j}) : i = 0, 1, \ldots, N, j = 0, 1, \ldots, M\}. \tag{6}$$

Note that in the numerical implementation most two-dimensional dynamic programs are expected to have continuously differentiable value functions as in theory. By using the envelope theorem, which basically states the relationship between the marginal value and marginal utility in the Bellman equation (see, e.g., Stokey and Lucus), we can derive the extra data of the first partial derivatives and mixed partial derivatives of $x$ and $y$. With the availability of slope data in (6) and level information in (5), we can employ in solving dynamic programs more sophisticated approximation methods by which better computational results can be achieved.

Let the following set of two-dimensional data be given:

$$\{(x_i, y_j, f_{i,j}^{(0,0)}, f_{i,j}^{(1,0)}, f_{i,j}^{(0,1)}, f_{i,j}^{(1,1)}) : i = 0, \ldots, N, j = 0, \ldots, M; x_i < x_{i+1}, y_j < y_{j+1}\},$$

where $f_{i,j}^{(0,0)} = f(x_i, y_j)$, $f_{i,j}^{(1,0)} = f_x(x_i, y_j)$, $f_{i,j}^{(0,1)} = f_y(x_i, y_j)$, and $f_{i,j}^{(1,1)} = f_{xy}(x_i, y_j)$. For $i, j$ arbitrary but fixed, we denote by $h_i^x = x_{i+1} - x_i$, and by $h_j^y = y_{j+1} - y_j$ the length of edges of $R_{i,j} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$ and, for the integers $n_i, m_j \geqslant 3$, we set the quantities

$$t_0 = x_i,\ t_1 = x_i + h_i^x/n_i, \quad t_2 = x_{i+1} - h_i^x/n_i \quad \text{and} \quad t_3 = x_{i+1},$$
$$u_0 = y_j,\ u_1 = y_j + h_j^y/m_j, \quad u_2 = y_{j+1} - h_j^y/m_j \quad \text{and} \quad u_3 = y_{j+1}.$$

The restriction that $n_i$ is greater than three ensures the quantities $t_1$ and $t_2$ are distinct. The same explanation applies to $m_j$. We say that the data are (coordinatewise) increasing and concave with

respect to $x$ along the edge $[x_i, x_{i+1}] \times \{y_s\}$, $s = j, j+1$, if, for the corresponding $x$-sections of the data,

$$0 \leqslant f_{i+1,s}^{(1,0)} < \Delta_{i,s}^{x} < f_{i,s}^{(1,0)}$$

holds and where $\Delta_{i,s}^{x} = (f_{i+1,s}^{(0,0)} - f_{i,s}^{(0,0)})/h_i^x$. This definition can be easily extended to the case where the data are increasing and concave with respect to $y$. Moreover, for the fixed $y = y_j$, we can easily construct a one-dimensional piecewise interpolation of the given data $\{f_{r,j}^{(0,0)}, f_{r,j}^{(1,0)}: r = i, i+1\}$ along $[x_i, x_{i+1}] \times \{y_j\}$ and the function is linear on each interval $[t_i, t_{i+1}]$, $i = 0, 1, 2$; another piecewise linear function in $y \in [u_0, u_3]$ can be constructed by the same way to interpolate $\{f_{i,s}^{(0,0)}, f_{i,s}^{(0,1)}: s = j, j+1\}$ along $\{x_i\} \times [y_j, y_{j+1}]$. By taking the product of these two univariate piecewise linear functions (in $x$ and $y$, respectively), we have a bivariate function $l(x, y)$ on $R_{i,j}$ such that for $\bar{y} \in [y_j, y_{j+1}]$, and $\bar{x} \in [x_i, x_{i+1}]$ arbitrary but fixed,

$$l(x, \bar{y}) \in C[t_0, t_3] \text{ (continuous on } [t_0, t_3]) \quad \text{and} \quad l(x, \bar{y}) \in P_1 \text{(piecewise linear)}$$

for $x \in [t_i, t_{i+1}]$, $i = 0, 1, 2$; similarly,

$$l(\bar{x}, y) \in C[u_0, u_3] \quad \text{and} \quad l(\bar{x}, y) \in P_1,$$

for $y \in [u_j, u_{j+1}]$, $j = 0, 1, 2$.

We outline as follows the construction of the $C^1$(continuously differentiable) Costantini bivariate shape-preserving interpolation on the rectangle $R_{i,j} = [x_i, x_{i+1}] \times [y_j, y_{j+1}]$:

*Step* 1: Find $n_i \geqslant 3$, such that

$$n_i \geqslant \max\left\{ \left\lceil \frac{f_{i,j}^{(1,0)} + f_{i+1,j}^{(1,0)}}{\Delta_{i,j}^{x}} \right\rceil, \left\lceil \frac{f_{i,j}^{(1,0)} - f_{i+1,j}^{(1,0)}}{\Delta_{i,j}^{x} - f_{i+1,j}^{(1,0)}} \right\rceil, \left\lceil \frac{f_{i,j}^{(1,0)} - f_{i+1,j}^{(1,0)}}{f_{i,j}^{(1,0)} - \Delta_{i,j}^{x}} \right\rceil \right\}.$$

Actually, if (5) and (6) indicate the given data are increasing and strictly concave, finite $n_i$ always exists. The choice of $m_j \geqslant 3$ has the same corresponding form with respect to $y$.

*Step* 2: Construct one univariate piecewise linear function fitting $\{f_{r,j}^{(0,0)}, f_{r,j}^{(1,0)}: r = i, i+1\}$ along $[x_i, x_{i+1}] \times \{y_j\}$ and the other one interpolating $\{f_{i,s}^{(0,0)}, f_{i,s}^{(0,1)}: s = j, j+1\}$ on $\{x_i\} \times [y_j, y_{j+1}]$.

*Step* 3: Take the product of these two univariate piecewise linear functions to form a smooth, bivariate function $l(x, y)$ on $R_{i,j}$. It can be easily verified that $l(x,y)$ satisfies

$$l(x, \bar{y}) \in P_1 \quad \text{for a fixed } \bar{y} \in [y_j, y_{j+1}] \quad \text{and} \quad x \in [t_i, t_{i+1}],\ i = 0, 1, 2,$$

$$l(\bar{x}, y) \in P_1 \quad \text{for a fixed } \bar{x} \in [x_i, x_{i+1}] \quad \text{and} \quad y \in [u_j, u_{j+1}],\ j = 0, 1, 2,$$

and

$$l(x_r, y_s) = f_{r,s}^{(0,0)},\ l_x(x_r, y_s) = f_{r,s}^{(1,0)},\ l_y(x_r, y_s) = f_{r,s}^{(0,1)} \quad \text{and} \quad l_{xy}(x_r, y_s) = f_{r,s}^{(1,1)}$$

for $r = i, i+1$, and $s = j, j+1$.

*Step* 4: Evaluate the shape-preserving surface $Bl(x, y)$ on $R_{i,j}$ defined as

$$\frac{1}{(h_i^x)^{n_i}(h_j^y)^{m_j}} \sum_{p,q=0}^{n_i,m_j} \binom{n_i}{p}\binom{m_j}{q} l\left(x_i + \frac{ph_i^x}{n_i}, y_j + \frac{qh_j^y}{m_j}\right)(x - x_i)^p (x_{i+1} - x)^{n_i - p}(y - y_j)^q (y_{j+1} - y)^{m_j - q}.$$

The mathematical expression in Step 4 is the bivariate Bernstein polynomial of $l(x, y)$ on the rectangle $R_{i,j}$. The variation diminishing property of Bernstein polynomials will ensure visually

pleasing and smooth surfaces. Indeed, Costantini and Fontanella [7] proved that $(Bl)(x, y)$ interpolates the given data (5) and (6) up to the first-order degree, namely,

$$(Bl)(x_r, y_s) = f_{r,s}^{(0,0)}, \qquad (Bl)_x(x_r, y_s) = f_{r,s}^{(1,0)},$$
$$(Bl)_y(x_r, y_s) = f_{r,s}^{(0,1)}, \qquad (Bl)_{xy}(x_r, y_s) = f_{r,s}^{(1,1)},$$

for $r = 0, \ldots, N$, $s = 0, \ldots, M$, and $(Bl)(x, y)$ also preserves the monotonicity and concavity of the interpolating suface over $R_{i,j}$.

Finally, we sum up the application of shape preservation in numerical dynamic programming as the following proposition.

**Proposition 1.** *Suppose the value function $V^t(\cdot, \cdot)$ is defined on a convex, compact domain $X \times Y$. For any strictly increasing and concave utility $u(\cdot)$, the value function $V^t(x, y)$ as calculated by Costantini bivariate shape-preserving interpolation is always increasing and concave.*

## 4. A numerical experiment

In this section we apply the bivariate shape-preserving interpolation in computing the value iterations (2). For the purpose of illustration by a tractable planning, we take a flat tax system in the pension savings allocation problem. We assume $u(c) = -e^{-c}$, and $\beta = 0.9$; $w_t = (1.05)^{t-1}$, $t = 1, \ldots, T$ (wage rises with time,) and $w_5 = w_6 = 0$ (no wage during retirement,) so correspondingly, $T = 4$, and $D = 6$ (here one time period may represent, say, 10 years) the upper limit of fraction of wage contribution to pension accounts is $p = 20\%$; the tax rates are $\tau_t^w = 20\%$, $\tau_t^r = 10\%$, and $\tau_t^z = 30\%$ for all $t = 1, \ldots, D$; the cash fund is assumed to have steady flows of interests with $r_t = 7\%$, and the return rates of stocks are $\tilde{z}_t \in \{-5\%, 5\%, 15\%, 25\%\}$ with equal probabilities to model the nature of high volatility yet higher average returns. Note that the utility function $u(c) = -e^{-c}$ is strictly increasing and concave, the value iterations $V^t(\cdot, \cdot)$ is expected to be monotonically increasing and concave in the variables $(X_t, Y_t)$ for $t = 0, 1, \ldots, 5$.

We incorporate NPSOL in the computation procedure. NPSOL is a collection of optimization subroutines designed to solve the non-linear programming problem: the minimization of a smooth non-linear function subject to a set of constraints on the variables; see Gill et al. [8] The code is written in standard Fortran 77, which in a double-precision floating-point arithmetic allows for a 16-digit accuracy. We take the grid size as 0.5 unit in each coordinate direction of the state space $[0, 5] \times [0, 5]$. Computing backwards the value iterations by the $C^1$ Costantini bivariate interpolation method, we first report in Table 1 the coordinatewise shape preservation – monotonicity and concavity – at grid points $(X_t, Y_t) \in [0, 1.5] \times [0, 1.0]$. Note that application of other methods with less sophisticated interpolation techniques such as the state increment dynamic programming in Larson and Casti [9] does not assure the interpolating surface over $[0, 5] \times [0, 5]$ is shape preserving, approximation errors hence propagate through the iterations (Table 1).

We then compute the corresponding optimal solutions of all decision variables such as $\{c_t, x_t, y_t, \theta_t, \varphi_t\}$ at all grid points. We go forward from the current period 0, starting with $X_0 = Y_0 = 0$, and use bilinear interpolation in each period successively to get the approximate solutions of those decision variables. We report in Table 2 these optimal streams of consumption

Table 1
Value iterations preserve the monotonicity and concavity coordinatewise

| $X$ | $Y$ | $V^5(X, Y)$ | $V^5_X(X, Y)$ | $V^5_Y(X, Y)$ | $V^0(X, Y)$ | $V^0_X(X, Y)$ | $V^0_Y(X, Y)$ |
|---|---|---|---|---|---|---|---|
| 0.0 | 0.0 | − 1.00000 | 0.88000 | 1.07000 | − 2.53552 | 0.46405 | 0.56420 |
| 0.5 | 0.0 | − 0.67032 | 0.58988 | 0.71724 | − 2.31273 | 0.41931 | 0.50984 |
| 1.0 | 0.0 | − 0.44933 | 0.39541 | 0.48078 | − 2.11544 | 0.39541 | 0.48078 |
| 1.5 | 0.0 | − 0.30119 | 0.26505 | 0.32228 | − 1.95047 | 0.39540 | 0.48076 |
| 0.0 | 0.5 | − 0.60653 | 0.53375 | 0.64899 | − 2.28362 | 0.39078 | 0.47516 |
| 0.5 | 0.5 | − 0.40657 | 0.35778 | 0.43503 | − 2.07832 | 0.35670 | 0.43371 |
| 1.0 | 0.5 | − 0.27253 | 0.23983 | 0.29161 | − 1.90012 | 0.33372 | 0.40577 |
| 1.5 | 0.5 | − 0.18268 | 0.16076 | 0.19547 | − 1.74266 | 0.30917 | 0.37593 |
| 0.0 | 1.0 | − 0.36788 | 0.32373 | 0.39363 | − 2.05834 | 0.34792 | 0.42304 |
| 0.5 | 1.0 | − 0.24660 | 0.21701 | 0.26386 | − 1.87345 | 0.31987 | 0.38894 |
| 1.0 | 1.0 | − 0.16530 | 0.14546 | 0.17687 | − 1.70975 | 0.29513 | 0.35885 |
| 1.5 | 1.0 | − 0.11080 | 0.09751 | 0.11856 | − 1.56677 | 0.27306 | 0.33202 |

Table 2
Optimal streams of consumption, and optimal patterns of contributions, fractions of account balance in stock investment, and accumulative savings in both accounts for the savings allocation problem

| | | | | | | |
|---|---|---|---|---|---|---|
| | $c_1$ | $c_2$ | $c_3$ | $c_4$ | $c_5$ | $c_6$ |
| | 0.64509 | 0.65469 | 0.64297 | 0.64510 | 0.61600 | 0.63114 |
| | (0.6451) | (0.6511) | (0.6387) | (0.6302) | (0.6103) | (0.6553) |
| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
| | 0.17342 | 0.21000 | 0.22050 | 0.23153 | − 0.45759 | − 0.78893 |
| | (0.1736) | (0.2146) | (0.2261) | (0.2298) | ( − 0.4483) | ( − 0.7942) |
| | $\theta_1$ | $\theta_2$ | $\theta_3$ | $\theta_4$ | $\theta_5$ | $\theta_6$ |
| | * | 57.14% | 57.14% | 57.14% | 57.14% | 57.14% |
| | (*) | (57.19%) | (57.19%) | (57.19%) | (57.19%) | (57.19%) |
| $X_0$ | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ | $X_6$ |
| 0 | 0.17342 | 0.40756 | 0.68365 | 1.00888 | 0.691135 | 0.00000 |
| (0) | (0.1736) | (0.4128) | (0.7013) | (1.0102) | (0.7024) | (0.0000) |
| | $y_1$ | $y_2$ | $y_3$ | $y_4$ | $y_5$ | $y_6$ |
| | 0.01617 | 0.01731 | 0.06263 | 0.11139 | − 0.24993 | 0.00000 |
| | (0.0160) | (0.0172) | (0.0624) | (0.1119) | ( − 0.2507) | (0.0000) |
| | $\varphi_1$ | $\varphi_2$ | $\varphi_3$ | $\varphi_4$ | $\varphi_5$ | $\varphi_6$ |
| | * | 39.42% | 48.53% | 57.14% | 57.14% | 46.01% |
| | (*) | (39.38%) | (48.46%) | (57.19%) | (57.19%) | (46.02%) |
| $Y_0$ | $Y_1$ | $Y_2$ | $Y_3$ | $Y_4$ | $Y_5$ | $Y_6$ |
| 0 | 0.01617 | 0.03613 | 0.10439 | 0.22776 | 0.00000 | 0.00000 |
| (0) | (0.0160) | (0.0358) | (0.1039) | (0.2281) | (0.0000) | (0.0000) |

$\{c_t\}$, contributions to pension and non-pension accounts $\{x_t, y_t\}$, and the fractions of each account savings invested on stocks $\{\theta_t, \varphi_t\}$ at each time period $t$. Note that negative entries of $x$ and $y$ indicate withdrawal. Also $\theta_1$ and $\varphi_1$ can be anything, denoted by $*$, since we are starting with $X_0 = Y_0 = 0$, no any initial wealth in both accounts.

To compare the numerical performance and verify the accuracy of the results solved by dynamic programming with bivariate shape-preserving interpolation, we redo this stochastic optimization problem by using only the NPSOL subroutines to solve the corresponding non-linear program. We report the decision variable outputs in the corresponding parentheses in Table 2. Note that the minor differences indicate the dynamic programming scheme with shape-preserving smooth approximation methods can do as well as NPSOL in solving the pension savings allocation problem. However, if we increase the time periods, $T$ and $D$, to model, say, every single day's process, then the number of decision variables will far exceed the capacity that NPSOL can handle; instead, dynamic programming with smooth, shape-preserving interpolation methods can still solve the subproblems iteratively and successfully to find solutions of the original large-scale problem within the tolerance of approximation errors.

## 5. Conclusions and further research

This paper has demonstrated how a bivariate shape-preserving interpolation method can be used to preserve the shape feature of the value function of a two-dimensional dynamic program. The numerical outputs indicate that bivariate shape-preserving interpolation can not only improve the approximation accuracy, but also ensure the stability of value iterations. A much simpler version of univariate shape-preserving approximation method is discussed in Judd and Solnick [10]. Wang [11] has numerical analysis for various shape-preserving approximation methods applied to optimal economic growth problems. To the best of our knowledge, our treatment provides the first study of shape preservation for two-dimensional economic dynamics. We incorporate the Costantini bivariate shape-preserving interpolation procedure and a dynamic programming formulation to solve numerically a pension savings allocation problem whose closed-form solution is not available. The numerical solutions also display economically important properties. At early ages the worker saves, building up his assets, and the assets in both accounts are then dissaved during retirement. Quantitative outputs, even approximate, can usually be of great help to us in reaching qualitative conclusions for a complex economic problem.

Possible directions for future research in the part of smooth approximation to solving multi-dimensional continuous-state dynamic economic problems include carrying out a systematical numerical comparisons of various high-dimensional methods for numerical dynamic programming and extending the definition of shape preservation to higher dimensional spaces.

## References

[1] Shoven JB, MaCurdy TE. Stocks, bond and pension wealth. In: Wise D, editor. Topics in the economics of aging, National Bureau of Economic Research. Chicago: University of Chicago Press; 1992:92–122.

[2] Kendrick D. Research opportunities in computational economics. Computational Economics 1993;6:257–314.
[3] Stokey NL, Lucas RE. Recursive methods in economic dynamics. MA: Harvard University Press, 1989.
[4] Costantini P. Boundary-valued shape-preserving interpolating splines. ACM Transactions on Mathematical Software 1997;23:229–51.
[5] Bellman R. Dynamic programming. Prinecton, NJ: Princeton University Press, 1957.
[6] Dodd SL, McAllister DF, Roulier JA. Shape-preserving spline interpolation for specifying bivariate functions on grids. IEEE Computer Graphics and Applications 1983;3:70–9.
[7] Costantini P, Fontanella F. Shape-preserving bivariate interpolation. SIAM Journal of Numerical Analysis 1990;27:488–506.
[8] Gill PE, Murray W, Saunders MA, Wright MH. User's guide for NPSOL: a Fortran package for nonlinear programming. Systems Optimization Laboratory, Stanford University, USA, 1986.
[9] Larson RE, Casti JL. Principles of dynamic programming, part II. New York: Marrcel Dekker, 1982.
[10] Judd KL, Solnick A. Numerical dynamic programming with shape-preserving splines. Mimeo, Hoover Institution, Stanford University, USA, 1996.
[11] Wang SP. Approximation of discrete dynamic programs with economic applications. Dissertation, Stanford University, USA, 1998.

**Sheng-Pen Wang** is currently an assistant professor in the Department of Business Administration, Chang Gung University, Taoyuan, Taiwan. His current research focuses on dynamic programming applications, as well as developing computational methods for economic and finance problems. He received his B.S. degree in mathematics, from National Taiwan Normal University, Taiwan, his M.A. in economics from Stanford University, California, and a Ph.D. in operations research from Stanford University.

**Kenneth L. Judd** is a senior fellow at the Hoover Institution. He is an associate editor of the *Journal of Computational Economics* (1993) and the *Journal of Economic Dynamics and Control* (1997). He has published articles in several academic journals including *Econometrica*, *Journal of Political Economy*, and *Journal of Economic Theory*. His book *Numerical Methods in Economics* has been published by MIT Press. Judd received a M.A. in mathematics and a Ph.D. in economics from the University of Wisconsin.