ISSN 1835-9728

Environmental Economics Research Hub Research Reports

A Comparison of Parametric Approximation Techniques to Continuous-Time Stochastic Dynamic Programming Problems

Tom Kompas and Long Chu

Research Report No. 71

September 2010

About the authors

Tom Kompas is Director of the Crawford School of Economics and Government at the Australian National University and The Australian Centre for Biosecurity and Environmental Economics
Building 132, Lennox Crossing, the Australian National University
Canberra ACT 0200, Australia
tom.kompas@anu.edu.au

Long Chu is a Research and Teaching Fellow at the Crawford School of Economics and Government
Australian National University
long.chu@anu.edu.au

Environmental Economics Research Hub Research Reports are published by The Crawford School of Economics and Government, Australian National University, Canberra 0200 Australia.

These Reports present work in progress being undertaken by project teams within the Environmental Economics Research Hub (EERH). The EERH is funded by the Department of Environment and Water Heritage and the Arts under the Commonwealth Environment Research Facility.

The views and interpretations expressed in these Reports are those of the author(s) and should not be attributed to any organisation associated with the EERH.

Because these reports present the results of work in progress, they should not be reproduced in part or in whole without the authorisation of the EERH Director, Professor Jeff Bennett (jeff.bennett@anu.edu.au)

Crawford School of Economics and Government
THE AUSTRALIAN NATIONAL UNIVERSITY
http://www.crawford.anu.edu.au

Table of Contents

Abstract

We compare three parametric techniques to approximate Hamilton-Jacobi-Bellman equations via unidimensional and multidimensional problems. The linear programming technique is very efficient for unidimensional problems and offers a balance of speed and accuracy for multidimensional problems. A comparable projection technique is shown to be slow, but has stable accuracy, whereas a perturbation technique has the least accuracy although its speed suffers least from the curse of dimensionality. The linear programming technique is also shown to be suitable for problems in resource management, including applications to biosecurity and marine reserve design.

JEL Classification: C61, C63, Q22

Keywords: stochastic dynamic programming, parametric approximation, perturbation, projection, linear programming, optimal fishing, marine reserves.

I. Introduction

Continuous-time stochastic optimal control problems are used intensively in economics and finance. There has been an increasing demand for quantitative solutions to these problems, especially in the fields of resource and environmental economics. However, solving these problems, using dynamic programming approaches and Hamilton-Jacobi-Bellman (HJB) equations generates analytical solutions in only very few cases. Solutions must instead rely on numerical approximation techniques.

In general, some standard approximation techniques rigorously applicable to Bellman equations in discrete-time settings, including value function or policy function iterations, do not work well in a continuous time setting[1]. In addition, since HJB equations are partial differential equations with second derivatives and no clear boundary conditions, approximation techniques based on finite difference schemes are often complicated and inconvenient[2]. Therefore solving a HJB equation numerically often relies on parametric approximation techniques which try to produce analytical formulae in the form of a linear combination of some pre-determined basic functions. Two well-known parametric techniques which have been applied to solve HJB equations are perturbation and projection methods. Applications of these techniques to dynamic programming can be found in various studies such as Judd (1996), Gaspar and Judd (1997) and Arruda and DoVal (2006).

Parametric linear programming is another approximation approach to HJB equations. It tries to produce an approximation of the maximum value function in the form of a linear combination of some pre-determined basic functions with the coefficients being solved from a linear programming scheme. The idea of linear programming approach can be found early, for example in the work of Manne (1960) or Ross (1970), which proves why the technique works in discrete-time setting. However, parametric linear programming approaches to continuous-time HJB equations has just recently been introduced by Han and Roy (2009), showing the technique is both efficient and simple.

While there are at least two papers that develop and compare various numerical methods in discrete-time dynamic programming, namely Taylor and Uhlig (1990) and Aruoba et al. (2006)[3], no similar

[1] Theoretically one can discretize the continuous time into small intervals for the use of value or policy function iterations. However, the discount factor then will be close to unity causing the solving process to converge very slowly. See Judd (1998) for more details.

[2] Finite difference (sometimes known as discretization) which aims at producing numerical values for various points over the solution's domain is a fundamental technique to PDE with clear boundary conditions. Hedlund (2003) uses a finite difference scheme to a deterministic continuous-time dynamic programming problem. The scheme is relatively compicated even without second order derivatives.

[3] Both papers use the classical growth model as a baseline case. Taylor and Uhlig (1990)compares ten then available techniques to highlight significant difference in the results. Aruoba et al. (2006)compare value function iterations to projection and perturbation and encourages the use of perturbation technique as a compromise of speed and accuracy.

work has been done for the continuous-time setting. This is perhaps because the parametric linear programming technique is relatively new. In this paper, we compare this new technique to the projection and perturbation techniques.

However, our comparison differs from the two previous papers. First, the comparison of the techniques' performance is done through two numerical case studies in fishery economics. The first model is uni-dimensional, with one control and one state variable. The second model is multidimensional with one control and two state variables. Each problem will be solved with three techniques with the same workstation and coding platform, Matlab 7.0[4]. The evaluation of their performance will be based on the approximation errors and computation time. This helps highlight how relative superiority changes with respect to the dimension of the problem being solved.

Second, we are not solving a classical growth model where its qualitative properties are already known. Instead, both numerical case studies in this paper have highly generalized non-linear structures where return functions are dependent on both state and control variables and uncertainty components are state-dependent. These models are increasingly used in applications to biosecurity and fisheries economics where not only qualitative properties but numerical results are also important to researchers.

Third, as a natural question arising from any comparisons that 'pick a winner', we compare taking into account not only the approximation quality and computation time, but other considerations that influence the choice of a technique in practice. For example, all three techniques have different approaches to HJB equations so their software package requirements vary. Another consideration is the fact that each technique may have variants that are more or less efficient in a particular problem. Although it is impossible to report all variants, we nevertheless hope to provide a general guide to the relative advantages and disadvantages of these three techniques.

The remainder of the paper is organized as follows. In Section II, we formulate a generalized dynamic optimization problem and specify the corresponding HJB equation. The uncertainty components in our generalized formulation are not only driven by Brownian diffusions but also Poisson diffusions, a standard instrument to model randomly discontinuous jumps[5]. Poisson diffusions have been used to model events that generate key results in many studies, for example technological progress in Aghion

[4] We use a Maple toolbox for Matlab for symbolic calculation.

[5] Though involving discontinuous jumps, Poisson processes are considered as continuous diffusions with respect to time as the probability of the discontinuous jump occurring in a time interval converges to zero when the time interval approaches zero

and Walde (1999), interest movements in Das (2002) and Piazzesi (2005), and negative shocks to fish stocks in Grafton et al. (2006). In this paper, both numerical case studies used to compare the performance of three techniques involve Poisson diffusions.

In Section III, we briefly describe the three techniques to solve the HJB equation for readers' convenience. The description of projection and perturbation techniques are especially brief. Further details for these techniques can be found in Judd (1998). Parametric linear programming will be described, on the other hand, with more detail as it is relatively new. Here, we provide a theorem, extended from Han and Roy (2009), to include models with Poisson diffusions, which is the theoretical base for this new technique.

Section IV and Section V are devoted to the two numerical case studies. Each problem is introduced and solved, with reports on approximation errors and computation time. With each technique, we solve the problems in the most plain manner, putting aside variants that can be applied to a particular situation. Section VI addresses the question 'which technique wins in practice' and Section VII concludes.

II. A generalized stochastic optimal control problem in continuous-time setting

We begin with a general optimal control problem in a continuous-time setting. The problem is to identify the maximum value function $V(s)$ and/or the optimal profile of the control variables $c(t)$ such that:

$$V(s) = max_{<c(t)\in\Phi(k(t))>} E_0 \int_0^\infty \; u(k,c)dt \tag{1}$$

subject to:

$$k(0) = s \tag{2}$$

$$dk = g(k,c)dt + \varphi(k,c)dw + \mu(k,c)dq \tag{3}$$

$$\lim_{t\to\infty} Prob(|k(t)| = \infty) = 0 \tag{4}$$

where w is a standard Brownian diffusion and q is a Poisson jump diffusion process with an arrival rate $\lambda(k,c)$.

To simplify the notation, we define a functional operator:

$$\begin{aligned} H_c(V(k)) &= -\rho V(k) + u(k,c) + \left(\tfrac{\partial V}{\partial k}\right)' g(k,c) \\ &+ \tfrac{1}{2} tr\left(\tfrac{\partial^2 V}{(\partial k)(\partial k)'}\varphi(k,c)'\varphi(k,c)\right) + \lambda(k,c)\left[V(k+\mu(k,c)) - V(k)\right] \end{aligned} \tag{5}$$

Then the HJB equation for Problem (1) is:

$$0 = \max_{<c>} \{H_c(V(k))\} \tag{6}$$

This HJB equation can be derived by applying the Principle of Optimality introduced by Bellman (1957) in a heuristic manner as in Appendix A1. The equation is confirmed in the following theorem:

Theorem 1: Suppose (i) $k(t)$ evolves in accordance with equations (2), (3) and (4); (ii) $V(.)$ is a twice differentiable function which satisfies equation (6), then $V(s)$ is the maximum value function for Problem (1).

A proof for Theorem 1 is provided in Appendix A2.

III. Parametric approximation approaches to HJB equations

1. Projection technique

Projection is a natural technique for analytically approximating differential equations and hence applicable to dynamic programming problems (Judd, 1998). In fact, it solves the system of first order conditions and envelope results, and not directly the HJB equation. Specifically, if an interior solution is assumed, we can differentiate the HJB equation (6) and apply the Envelope theorem to obtain a system of differential equations.

$$\begin{cases} 0 = \frac{\partial H_c(V(k))}{\partial c} \\ 0 = \frac{\partial H_c(V(k))}{\partial k} \end{cases} \tag{7}$$

To approximate the system of differential equations (7), the projection technique needs to assume an approximation forms for the maximum value function $V(.)$ and the policy function $C(.)$. These are pre-determined combinations of selected basic functions with undetermined coefficients. The method then tries to determine the coefficients with which the right hand sides (RHS) have smallest distance to zero. Different definitions of "smallest distance" lead to different variants of the technique. However, the most common variant is to find the smallest sum square of the RHSs at pre-determined collocation points.

2. Perturbation technique

Perturbation theory has a long history in numerical approximation but its application to dynamic programming is newer than the projection technique. The perturbation approach to HJB equations was first introduced by Judd and Guu (1993) with a unidimensional problem and more formally in Gaspar and Judd (1997). The technique starts with adding an auxiliary variable (say ϵ) into the HJB equation such that if $\epsilon = 0$, the problem is deterministic and if $\epsilon \neq 0$, it is stochastic. Specifically, the HJB equation is converted to:

$$0 = \max \left\{ \begin{array}{c} -\rho V(k) + u(k,c) + \frac{\partial V}{\partial k}' g(k,c) + \epsilon^2 \frac{1}{2} tr \left(\frac{\partial^2 V}{(\partial k)(\partial k)'} \phi(k,c)' \phi(k,c) \right) \\ + \epsilon \lambda(k,c)[V(k + \mu(k,c)) - V(k)] \end{array} \right\} \tag{8}$$

The perturbation idea will be applied in two rounds. The first round is to solve for the deterministic version by setting $\epsilon = 0$. After determining the steady state, it successively differentiates and evaluates the first order condition and the envelope results of the HJB equation at the steady state to solve for the derivatives of the maximum value and the policy functions with respect to the state variables. The second round is to differentiate first order condition and the envelope results with respect to the auxiliary variable ε and solve for the corresponding derivatives. Once all the necessary derivatives with respect to the state variables and ε are identified, functions of state variables and ε can be constructed using Taylor expansions. The final step is to substitute $\epsilon = 1$ to obtain the approximation for the stochastic problem.

3. Parametric linear programming technique

3.1. Theoretical basis of the technique

The linear programming technique approaches the HJB equation (6) as a system of (weak) inequalities. The max operator in the equation implies two points. First, at any state and any feasible levels of the control variables, the term $H_c(V(k))$ is non-positive. Thus, the maximum value function can be identified from a class of functions which satisfy the system of the weak inequalities imposed by the HJB equation (6).

Second, at a state there exists at least a feasible control such that the term $H_c(V(k))$ is zero. This is the optimal control for at the state in question. Feasible control levels other than optimal lead to a strictly negative value. This suggests a way to pin down the optimal policy function. More importantly, it guarantees that the maximum value function is the smallest among those satisfying the system of the weak inequalities imposed by the HJB equation (5). This is an important property we can confirm in the following theorem:

Theorem 2 : Suppose (i) $k(t)$ evolves in accordance with equations (2), (3) and (4); (ii) $V(s)$ is the maximum value function for Problem (1); (iii) $w(s) > 0$ for all s, then $V(s)$ uniquely solves the optimization problem:

$$\min_{J(s)\in C^2} \int w(s)J(s)ds \qquad (9)$$

subject to: $0 \geq H_c(J(s))$ *for all s and with each s for all $c \in \Phi(s)$.*

A proof for Theorem 2 is provided in Appendix A3. The idea of the theorem is simple. As the maximum value function is the smallest, the sum of its values over a domain with any positive weights is also the smallest. This is a non-linear minimization problem, but can be approximated via a linearization process. The technical properties of this approach can be found in Farias and Roy (2003).

3.2. *Technical procedure for parametric linear programming technique*

First the maximum value function is conjectured to be a linear combination of some pre-selected basic functions with undetermined coefficients. Suppose the vector of the basic functions is $\Phi(s) = (\Phi_1(s), \Phi_2(s), ..\Phi_K(s))$ and the vector of the undetermined coefficients is $r = (r_1, r_2, ...r_K)$ where K is the number of coefficients to be estimated, then the maximum value function can be approximated in the form $V(s) = r'\Phi(s)$.

In the second step, the domain of interest is discretized with a set (denoted as G_s hereafter) of state collocation points. Then with each state collocation point ($s \in G_s$), the associated action correspondence $\Phi(s)$ is discretized with a set, denoted as $G_c(s)$, of action collocation points. These state and action collocation points are used as representatives for the whole state and correspondence spaces. Given the choice of the collocation points, the nonlinear minimization in Problem (9) can be approximated by the following linear programming scheme:

$$\min_{<r>} \sum_{s\in G_s} w(s)r'\Phi(s) \qquad (10)$$

subject to: $0 \geq H_c(r'\Phi(s))$ for all $s \in G_s$ and with each s for all $c \in G_c(s)$.

The size of this linear programming scheme depends on the dimension of the dynamic programming problem and the choice of the collocation points. Denote m as the number of state variables and η as the number of control variables, or in other words the dimension of the state space is m and the dimension of the action correspondence is η. If we discretize each state variable with κ and each action correspondence with ψ collocation points, the set of state collocation points G_s will contain κ^m

points. Associated with each point $s \in G_s$, there will be a set $G_c(s)$ containing ψ^η action collocation points. Hence the constraint system of Problem (10) has $\kappa^m \psi^\eta$ weak inequalities. As the number of coefficients to be determined is K, the size of the linear programming scheme is $\kappa^m \psi^\eta \times K$.

Problem (10) can be solved by any linear programming solver for the undetermined coefficients. Once the maximum value function is approximated, the optimal policy function can be calculated from the HJB equation (6). In many cases, the structure of the HJB equation allows one to directly solve for the optimal policy function analytically through the FOC. In other cases where this is not possible, a grid search for the maximizers over the action correspondence works well.

3.3. Approximation errors and accuracy improvement

Given the approximation of the maximum value and optimal policy functions, we can calculate the approximation errors following the procedure in Judd (1998). The two functions are substituted into the HJB equation (6). The errors are then calculated as the discrepancies between two sides of the equation as a percentage of the maximum value function.

If the errors are not satisfactory, there are a number of ways to improve the approximation quality. First, the number of state collocation points can be increased. This enhances the representativeness of the state collocation set (G_s) over the domain of interest. However, the linear programming scheme becomes larger due to the increased number of constraints. Hence, this solution is dependent on the capacity of the solver package.

The second measure is to increase the effectiveness of the action collocation sets $G_c(s)$. A natural but costly way is to increase the number of action collocation points. Similar to the state collocation, this makes the linear programming scheme larger and more costly to be solved.

A more efficient way to increase the effectiveness of the action collocation set is to reduce the size of the discretized correspondence. At any states, the constraints in Problem (9) bind only at the optimal policy and are slack at all other levels. Thus if the optimal policy function was known, evaluating the constraints at the optimal level would suffice. Each state would need only one constraint and the number of constraints could be dramatically reduced to κ^m. However the optimal policy is not known before the approximation process begins. It can even never be perfectly approximated. But if we have some information about the optimal policy levels at any particular states and narrow the relevant correspondence, we can make the discretization finer, given the same size of action collocation sets. This type of information can come from experience, insights of the problem in question or even a draft solution.

The third measure is to use different weight functions, $w(s)$, as suggested by the authors of the technique, Han and Roy (2009). A higher relative weight attached to one state will lead to a more accurate approximation at that state. Hence this measure is useful if we need to increase the quality in a specific part of the domain. The trade off is that this may reduce the approximation quality in the rest of the domain. Farias and Roy (2004) provide a comprehensive analysis on the constraint sampling property of the parametric linear programming approach.

Finally, we can enlarge or change the set of basic functions. If more functions are added, there will be more coefficients to be estimated, which results in a higher column dimension of the linear programming scheme. In some situations, choosing another set of basic functions may be helpful. The choice of basic functions can be flexible. While polynomials are the most common due to their simplicity and convenience, other functions can be chosen if they are believed to work better. This indeed is a useful direction for future research efforts.

IV. Case study 1: Unidimensional standard fishery problem

1. The model

In this section, we present a numerical example in fisheries economics. The transition law of a fish population has deterministic and stochastic components. The deterministic component is the difference between a logistic fish growth function with an intrinsic rate r and harvest h. The stochastic component consists of two types of diffusions: Brownian motion and Poisson diffusion with a negative magnitude. The Brownian diffusion represents neutral natural shocks while the Poisson diffusion represents negative shocks caused by harvest activities. The magnitudes of both shocks are stock dependent.

Denote s as the fish population, MCC as 'maximum carrying capacity', r as the intrinsic biological parameter, h as harvest, w as a standard Brownian diffusion and q as a Poisson diffusion with an arrival rate $\lambda > 0$, then the transition of the fish population is described by the following stochastic differential equation:

$$ds = \left[rs\left(1 - \tfrac{s}{MCC}\right) - h\right]dt + \varphi(s)dw + \mu(s)dq \tag{11}$$

for $\phi(s) > 0$ and $\mu(s) < 0$ the magnitudes of the Brownian and Poisson diffusions.

The profit function for fishing activities is standard. Fishing revenue is $\frac{P}{h^\alpha} \times h$ with $0 < \alpha < 1$ where $\frac{P}{h^\alpha}$ is the sale price with price elasticity α. Fishing unit cost is proportional to the fish density with a cost parameter c. The return is the fishing profit:

$$u(s,h) = Ph^{1-\alpha} - \tfrac{c}{s}h \tag{12}$$

The problem of the regulator is to maximize the aggregate return (discounted at a rate ρ) defined in equation (12) subject to the transition law defined in equation (11) given an initial fish stock and the sustainability condition. Specifically, the problem is to approximate the maximum value function $V(s(0))$ such that:

$$V(s(0)) = \max_{<h(t) \le s(t)>} \int_0^\infty e^{-\rho t} u(s,h)dt \tag{13}$$

subject to equation (11), given $s(0)$ and $\lim_{t\to\infty} E_0 s(t) > 0$.

2. HJB equation and numerical values for the parameters

Given the problem specification, the HJB equation for Problem (13) is:

$$0 = \max_{<h>} \left\{ \begin{array}{c} -\rho V(s) + \left(Ph^{1-\alpha} - \frac{c}{s}h\right) + \frac{\partial V}{\partial s}\left[rs\left(1 - \frac{s}{MCC}\right) - h\right] + \\ \frac{1}{2}\frac{\partial^2 V}{\partial s^2}\varphi^2(s) + \lambda\left[V(s+\mu(s)) - V(s)\right] \end{array} \right\} \tag{14}$$

The numerical values for the biological, economic and uncertainty parameters are taken from Grafton et al. (2006) with the price and cost parameters scaled up for a graphical convenience. The standard error of the natural shock is assumed to be 5 per cent of the current fish population. Negative shocks which reduce the fish population by 13 per cent are assumed to occur every 10 years. All parameters are reported in Table 1.

3. Technical choice for the approximation process

Given $MCC = 1$, we approximate the maximum value function in a relatively wide domain $[0.2..1]$. The function is conjectured to be a polynomial of order n, $V(s) = \sum_{i=0}^{n} r_i s^i$. There are $n+1$ parameters to be approximated. We solve the problem in two cases: $n = 6$ and $n = 10$.

The implementation of perturbation technique does not offer much flexibility during its approximation process. However, we may have different choices for the state collocation points with projection and linear programming techniques. To make the comparison fair, we discretize the domain into 100 evenly distributed intervals with a set G_s of 101 collocation points and use this for both techniques.

For the projection technique, we choose the coefficients to minimize the sum square of the RHSs in the system of differential equations (7). The minimization algorithm is the quasi-Newton method. For the linear programming technique, we need action collocation points as well. Hence for each state s in the collocation set G_s, the action correspondence $[0, s]$ is discretized with a set $G_h(s)$ of 201 nodes. The weight is simply chosen to be the unit vector, $w(s) = 1$ for all s. Given the choice of the state and action collocation points, the constraint system of the linear programming scheme includes $|G_s| = 101$ states. Each state is associated with the $G_h(s) = 201$ constraints corresponding to 201 possible actions in the action collocation set. Thus the total number of constraints is $|G_s| \times |G_h(s)| = 101 \times 201 = 20,302$.

Implementing the algorithms of the three techniques, we have three different analytical approximations of the maximum value functions. Based on these we can calculate the approximation errors.

4. Calculating the approximation errors

To calculate the approximation errors, we have to calculate the optimal harvest function first. In this case, the structure of the HJB equation (14) allows us to calculate the optimal harvest function analytically, or from the first order condition:

$$h = \left[\frac{1}{(1-\alpha)P}\left(\frac{\partial V}{\partial s} + \frac{c}{s}\right)\right]^{-\frac{1}{\alpha}} \qquad (15)$$

After the optimal harvest function is calculated from equation (15), we substitute it into the HJB equation (14) and divide the RHS by the maximum value function to obtain the approximation errors. The maximum value function, optimal harvest function and approximation errors produced by the linear programming approach are plotted in Figure 1, Figure 2 and Figure 3.

5. Performance of the three techniques in unidimensional case study

To compare the performance of the three techniques, we calculate the average and the maximum of the absolute values of the approximation errors in 101 state collocation points and report them with the computation time measured in seconds in Table 2 and Table 3.

Though the computation time will vary across work stations, the tables convey three clear messages about the performance of the techniques. First, the projection technique is accurate but slow in comparison to its competitors. Second, the perturbation technique is the least accurate. This is understandable because the domain of the approximation is wide and perturbation accuracy decays quickly away from the steady state. The third message is the superiority of linear programming technique. It is not as accurate as the projection technique when $n = 6$ but improves substantially when $n = 10$. The most striking feature is the small extra cost of improving accuracy with respect to computation time, that makes it faster than the perturbation technique when $n = 10$.

The perturbation technique has been believed to be fast so its speed championship when $n = 6$ is not a surprise. However, the superior speed of the linear programming technique where $n = 10$ is a surprise, needing an explanation. The reason is behind the algorithms of the two techniques. In spite of the uni-dimension, the perturbation technique has to approximate a 2-dimensional polynomial (with respect to the state and the auxiliary variables). This 2-dimensional polynomial has 28 coefficients when $n = 6$ and 66 coefficients when $n = 10$. Therefore, an increase in the polynomial order from $n = 6$ to $n = 10$ requires an additional 38 coefficients[6]. With the linear programming technique, the increase in polynomial order involves solving a linear programming scheme with size 11×20,302 instead of 7×20,302, an only additional four columns. In other words, the increase in the number of coefficients is quadratic with the perturbation technique while it is linear with the linear programming technique, which makes the latter faster when $n = 10$.

V. Case study 2: Multidimensional marine reserve problem

1. The model

In this section, we solve the marine reserve problem introduced in Grafton et al. (2006). In this model, the authority sets a certain proportion (denoted by R) of a fish population as a reserve. This protected area is closed from harvesting activities. Firms can only catch fish (denoted as h) in the exploitable

[6] If we count the coefficients in the optimal policy function approximated at the same time, the number will be higher.

area with size $1 - R$. The number of fish which transfer from the protected reserve to the exploitable area (T) is proportional to the size of each part and the differential in fish densities. The fish stocks in both areas are also subject to a standard Brownian motion ω and Poisson diffusion q with an arrival rate λ, so the transition laws are:

$$\begin{aligned} ds_E &= \left[rs_E\left(1-\tfrac{s_E}{(1-R)\times MCC}\right)+T(s_E,s_R)-h\right]dt+\varphi_E(s_E)dw+\mu_E(s_E)dq \\ ds_R &= \left[rs_R\left(1-\tfrac{s_R}{R\times MCC}\right)-T(s_E,s_R)\right]dt+\varphi_R(s_R)dw+\mu_R(s_R)dq \end{aligned} \tag{16}$$

where $T(s_E, s_R) = \phi \frac{R(1-R)}{MCC}\left(\frac{s_R}{R} - \frac{s_E}{1-R}\right)$.

Fishing profit is similar to the standard fishery model. Revenue is identical and unit cost is proportional to the inverse of the fish density in the exploitable area:

$$u(s_E, h) = Ph^{1-\alpha} - \frac{c}{\frac{s_E}{(1-R)MCC}}h \tag{17}$$

The problem of the regulator is to maximize the (discounted) aggregate return defined in equation (17) subject to the transition laws defined in equation (16) given initial fish stocks and the sustainability condition. In particular, the problem is to approximate the maximum value function $V(s_E(0), s_R(0))$ such that:

$$V(s_E(0), s_R(0)) = \max_{<h(t)\leq s_E(t)>} \int_0^\infty e^{-\rho t}u(s_E, h)dt \tag{18}$$

subject to the transition laws in equation (16), given $s_E(0)$ and $s_R(0)$ and $\lim_{t\to\infty} E_0(s_E(t) + s_R(t)) > 0$.

2. Multidimensional HJB equation and numerical values for the parameters

Given the problem specification, the HJB equation for Problem (18) becomes:

$$0 = \max_{<h>} \left\{ \begin{array}{l} -\rho V(s_E, s_R) + \left(Ph^{1-\alpha} - \frac{c}{\frac{s_E}{(1-R)MCC}}h\right) \\ +\frac{\partial V(s_E,s_R)}{\partial s_E}\left[rs_E\left(1-\frac{s_E}{(1-R)*MCC}\right)+T(s_E,s_R)-h\right] \\ +\frac{\partial V(s_E,s_R)}{\partial s_R}\left[rs_R\left(1-\frac{s_R}{R*MCC}\right)-T(s_E,s_R)\right] \\ +\frac{1}{2}\frac{\partial^2 V(s_E,s_R)}{\partial s_E^2}\varphi_E^2(s_E)+\frac{1}{2}\frac{\partial^2 V(s_E,s_R)}{\partial s_R^2}\varphi_R^2(s_R) \\ +\frac{\partial^2 V(s_E,s_R)}{\partial s_E \partial s_R}\varphi_E(s_E)\varphi_R(s_R) \\ +\lambda\left[V(s_E+\mu_E(s_E), s_R+\mu_R(s_R)) - V(s_E,s_R)\right] \end{array} \right\} \tag{19}$$

Similar to the unidimensional case study, the numerical values for the biological and economic parameters are taken from Grafton et al. (2006) as reported in Table 1. The reserve size is assumed to be $R = 30$, around the optimal value reported in Grafton et al. (2006).

3. Technical choice for the approximation process

As the reserve size $R = 30\%$, the maximum carrying capacity in the protected area is $R \times MCC = 0.3$ and in the exploitable area is$(1 - R) \times MCC = 0.7$. We approximate the maximum value function in the domain $(s_E, s_R) \in [0.2..0.7, 0..0.3]$. The maximum value function is conjectured to be a polynomial of order four, $V(s_E, s_R) = \sum_{i=0}^{4} \sum_{j=0}^{4-i} r_{ij} (s_E)^i (s_R)^j$. There are 15 coefficients to be approximated.

We discretize the domain and action correspondence into evenly distributed grids. The two-dimensional domain is discretized with a set G_s consisting of 51×51=2601 state collocation points and use this for both the projection and linear programming techniques. Similar to the unidimensional case, the linear programming technique needs action collocation points as well. Hence with each state collocation point $(s_E, s_R) \in G_s$, the action correspondence $[0, s_E]$ is discretized with a set $G_h(s_E, s_R)$ of 51 nodes. The weight is simply chosen to be the unit vector, $w(s_E, s_R) = 1$ for all (s_E, s_R). Given the choice of the state and action collocation points, the constraint system of the linear programming scheme includes $|G_s| = 2601$ states. Each state is associated with $|G_h(s_E, s_R)| = 51$ constraints corresponding to 51 possible actions in the action collocation set. Thus the total number of constraints is $|G_s| \times |G_h(s_E, s_R)| = 51 \times 51 \times 51 = 132,651$.

4. Calculating approximation errors

The structure of the multidimensional HJB equation (19) again allows us to calculate the optimal harvest function analytically. The first order condition implies:

$$h = \left[\frac{1}{(1-\alpha)P} \left(\frac{\partial V(s_E, s_R)}{\partial s_E} + \frac{c(1-R)}{s_E} \right) \right]^{-\frac{1}{\alpha}} \tag{20}$$

The optimal harvest function is calculated using equation (20). Both maximum value and optimal harvest functions are substituted into the HJB equation (19) to calculate the approximation errors. The maximum value, optimal harvest functions and approximation errors produced by the linear programming approach are plotted in Figure 4, Figure 5 and Figure 6.

5. Performance of the three techniques in the multidimensional case study

Similar to what have been done with the unidimensional problem, we calculate the average and the maximum of the absolute values of the approximation errors in the 2601 state collocation points and report them together with the computation time in Table 4. Two points can be concluded, similar to the unidimensional case. First, both the projection and linear programming techniques are good in terms of quality, while the perturbation is still the least accurate. Second, projection technique is again relatively slow.

However, the difference from the unidimensional case study is that perturbation technique is now fastest. The linear programming computation time is still quite good but relatively slower. What explains this is the curse of dimensionality, which influences the techniques in different ways. A higher dimensional problem requires more coefficients to be approximated in all techniques, implying more derivatives in perturbation and more column dimension in the linear programming method. However, the row dimension of the linear programming scheme has risen significantly. In the unidimensional problem, where the state contains 101 collocation points and the correspondence contains 201 points, the row dimension is only 101×201=20,302. In the multidimensional problem with an additional state variable, though each state and the action correspondence contains only 51 points, the row dimension is 51× 51×51=132,651, more than six times larger. If we have another state or control variable and discretize it with 51 collocation points, the size of the linear programming will be 51 times larger and the computation time will increase very quickly.

VI. Practical choice over on the three techniques

In this section, we discuss some practical considerations on the choice of the three techniques. Obviously, their performance (approximation quality and computation time) is an important indicator. However, the choice of the techniques in practice also depends on its flexibility, applicability, software requirement and problem-specific factors. Instead of naming the best technique in general, we try to address the question 'which win in what situation' or equivalently 'what a technique can do best and what it does worst'.

1. Perturbation technique

The numerical case studies provide evidence that the perturbation technique is not efficient in unidimensional problems where it loses its speed advantage to the linear programming technique, and remains the least accurate technique. For the purpose of accuracy improvement, the perturbation technique also has a very limited variant. The only technique available so far is to use rational functional form (Pade approximation) instead of polynomials but this can only be applied to deterministic unidimensional HJB equations (see Judd and Guu, 1993). In addition, perturbation is the least flexible technique in controlling the distribution of approximation errors, which always takes the shape of an increasing deterioration away from the steady state. It is hardly possible to increase quality in a certain part of the domain even if we are willing to scarify the quality in others.

With respect to the scope of application, the perturbation technique is also the most restrictive. As its algorithm relies on differentiating and evaluating the derivatives at a steady state, it works better in problems with a unique steady state. In problems without a steady state (or one that is impossible to calculate), the use of perturbation technique is ruled out. The perturbation technique cannot work where there exists a corner solution or a non-differentiable HJB equation. This also requires a strong symbolic toolbox for successive symbolic differentiation and evaluation, which is relatively expensive in comparison to the software required by the other two techniques.

However, the perturbation technique still retains some attractions. First, this is only technique that produces the steady state during the approximation process. With the projection and linear programming methods, the calculation of the steady state (if necessary) often needs be done with another simulation step after the approximation of value and optimal policy functions. The perturbation technique is thus convenient in economic problems where the steady state and the dynamic behavior around that are the main focus of the analysis.

Second, the perturbation technique is still superior for comparative static analysis on the effects of various parameters. For example, it produces the optimal harvest as a function of both fish stock and fish price and make the analysis of the fish price effect on the optimal path to MEY very tractable. This is impossible or extremely difficult with the projection and linear programming techniques.

Third, the perturbation technique suffers least from the curse of dimensionality, hence it is more appropriate in multidimensional problems. In addition, it is the only technique that allows to increase the approximation quality by adding more terms to existing ones. For example in the unidimensional problem, in order to improve the accuracy by increasing from $n = 6$ to $n = 10$, the perturbation technique does not need to recalculate the coefficients for the first to the sixth order terms. It only has to further calculate the seventh to tenth order terms to generate a longer Taylor expansion. The projection and linear programming techniques do not have this feature as all terms need recalculating when the polynomial order is changed. Therefore, improving approximation quality in the perturbation method involves a trivial programming burden compared to the other two techniques.

2. Projection technique

Projection technique is slow but its accuracy is stable. Practical experience shows that this technique offers the clearest opportunity for a trade off between speed and accuracy in most problems, i.e. the approximation quality can improve significantly in most situations when more time is devoted to computation. A higher number of state collocation points or basic functions will lead to significantly more accurate results. In some situations with the other two techniques, a large attempt leads to only an insignificant improvement.

In addition, this technique has a wide range of variants that can be applied flexibly to a particular problem. For example, one may use Chebychev polynomials or other functional forms to improve the quality. The variants of the projection technique allow a highest degree of flexibility to control the distribution of the approximation errors. For example, a higher weight attached to a particular collocation point in defining the distance function leads to a smaller error in that state. Even if a researcher needs the errors at some particular points to be zero, the task of finding coefficients, which minimizes the distance function can be converted to solving a system of equations with which the distance is zero in the desired collocation points.

However, due to its very slow speed, the projection technique is not appropriate in many practical situations. Economic parameters are usually estimated from econometrics with a confidence interval, therefore sensitivity analyses with different scenarios are often required. Projection technique may be too slow to be used in such a sensitivity analysis. Suppose we calculate the maximum value function of the multidimensional case study with different possible levels of five parameters, namely the price coefficient (P), the cost coefficient (c), the intrinsic rate (r), the reserve size (R) and the likelihood of a negative shock (λ). If the size of the sensitivity analysis β (each parameter has β possible alternative values), then the total number of parameter combinations is β^5. Based on the computation time for one set of parameter values reported in Table 5, we calculate the computation time required by the three techniques to implement this sensitivity analysis in Table 6. If the analysis size is $\beta = 5$, the perturbation technique can finish in less than an hour, the linear programming technique needs around four hours while the projection technique needs about 38 days. If the size $\beta = 6$, the perturbation and linear programming techniques can be completed within hours while the projection technique needs three months.

In addition, as projection technique uses non-linear optimization, the computation time will increase quickly with respect to the number of coefficients. This is especially true when the distance function is very complicated as it is the sum of a large number of nonlinear terms, taking time to be evaluated numerically. Perhaps, increasing the number of coefficients to be approximated in projection technique is most costly though a significant quality improvement is assured.

The application of projection technique is wider than perturbation technique as it does not need a steady state. However it is still restricted to solving the FOC and envelope results of HJB equations. Thus it cannot solve for a corner solution or approximate non-differentiable HJB equations.

3. Linear programming technique

The performance of the linear programming technique in the two case studies provides evidence that this is a combination of being fast and accurate. It strongly competes with the perturbation technique in speed and with the projection technique in approximation quality. It requires the most available software with a linear programming package.

Though the linear programming technique does not have as many variants to improve the accuracy as projection technique, the choice of collocation points and functional form can be still be very flexible as discussed in Section III. The distribution of approximation errors can also be controlled somewhat by choosing different weight vectors. More importantly, this technique allows for the use of prior information to improve approximation quality. For example in the unidimensional case study, an experienced researcher in fisheries economics may guess that the optimal harvest level is certainly less than half of the fish stock. Then it is not necessary to discretize the whole action correspondence $[0, s]$ into 201 collocation points. Instead, the researcher can discretize the suspected action correspondence $[0, \frac{1}{2}s]$ into 101 points and obtains the same result with the row dimensional of the linear programming scheme reduced by 50%.

The applicability of linear programming technique is widest. It relies on neither the existence of a steady state nor differentiating HJB equations. Therefore, this is the only technique that can be used in case of a corner solution or non-differentiable HJB equations.

Beside these strengths, the parametric linear programming technique has some weaknesses. First, it is the only technique which approximates the maximum value function only, the optimal policy function may not be available in analytical form as with the two other methods. When the optimal policy function cannot be solved from the FOC of the HJB equation, it has to be approximated numerically from a grid search. Then it is usually costly to obtain the optimal policy function if one needs it, say to simulate the steady state.

Second, the linear programming problem is most affected by the curse of dimensionality. Adding one variable into the problem, regardless of whether it is a state or control variable, will enlarge the size of the linear programming scheme by several dozen times. Assuming each state or control variable has only 51 collocation points and the value function is conjectured to be a fourth order polynomial as in the multidimensional case studies, we calculate the sizes of the linear programming schemes with different dimensions and report in Table 6 where m is the dimension of the state space and η is the dimension of the action correspondence. It is clear that the column size depends on the number of the state variables but the row size increases exponentially with respect to both variables. For the case $m = 3$ and $\eta = 3$, we need at least 4720 GB memory to store the matrix, let alone solving the linear programming scheme. Therefore if there are more than three state or control variables, the linear programming technique is not practical unless we accept an inaccurate approximation by reducing the number of collocation points to a trivial level.

Finally, the linear programming technique needs to have a bounded action correspondence. Without boundedness, the action correspondence cannot be discretized into action collocation points. Fortunately in economics where a decision maker always faces scarcity constraints, most optimal control problems satisfy or can be converted to satisfy the boundedness requirement. Hence this is a theoretical rather than a practical issue.

VII. Concluding Remarks

In this paper, we extend the parametric linear programming approach to include problems with Poisson jump diffusions and compare it to the projection and perturbation techniques. The performances of the three techniques in two case studies show that the linear programming technique is a combination of speed and accuracy. It is a strong competitor with the perturbation technique in speed and with projection technique in approximation quality. This new technique also has a widest applicability when it works even if HJB equations are not differentiable and requires most simple software package.

However, the linear programming technique is most affected by the curse of dimensionality hence it may not be suitable for large dimension problems. In this aspect, the perturbation technique is still the most attractive as it can significantly soften the curse and produce approximations with a reasonable computation time. The perturbation technique is also ideal for analyzing the effect of exogenous parameters in a comparative static analysis. The weakness of perturbation technique is the accuracy which decays very quickly away from the steady state of the dynamic system.

The projection technique is too slow and hence it is not appropriate to solve practical problems in economics. However, it has a wide range of variants that may be applied to a particular problem to meet specific demands of researcher with stable approximation qualities. It is also fairly intuitive, easy to code and requires only a non-linear optimizer which is widely available with many coding platforms.

Appendices

A.1 Heuristic derivation of the HJB equation (6)

Denote $U^{\{c\}}(t)$ the aggregate return from time t onward contingent on the choice of a feasible policy plan $\{c\}$ or $U^{\{c\}}(t) = \int_t^{\infty} e^{-\rho(\tau-t)}u(k,c)d\tau$. The expectation of this return depends on the value of the state variables at the initial time *t*, so we can write:

$$E_t U^{\{c\}}(t) = V^{\{c\}}(k(t)) \tag{21}$$

where $V^{\{c\}}(.)$ is the value function associated with the policy plan $\{c\}$.

To keep in line with the Optimality Principle, we add a small dt to define 'the next period'. Consider an infinitesimal time interval $[t, t+dt]$ where the state variables are fixed without uncertainty at $k(t)$ and the control variables are fixed at $c(t)$. Then the expected aggregate return from time t onward can be decomposed into the return obtained in $[t, t+dt]$ and the (discounted) aggregate return from time $t+dt$ onward.

$$\begin{aligned} &V^{\{c\}}(k(t)) = E_t U^{\{c\}}(t) \\ &= \int_t^{t+dt} e^{-\rho(\tau-t)}u(k,c)d\tau + E_t \int_{t+dt}^{\infty} e^{-\rho\tau}u(k,c)d\tau \\ &= \int_t^{t+dt} e^{-\rho(\tau-t)}u(k,c)d\tau + e^{-\rho dt}E_t U^{\{c\}}(t+dt) \\ &= u(k,c)\tfrac{1-e^{-\rho dt}}{\rho} + e^{-\rho dt}E_t U^{\{c\}}(t+dt) \end{aligned} \tag{22}$$

where we use $\int_0^{dt} e^{-\rho\tau}d\tau = \frac{1-e^{-\rho dt}}{\rho}$.

Applying Maclaurin's expansion $e^{-\rho dt} = \sum_{i=0}^{\infty}(-1)^i$ and using the fact that $(dt)^i \simeq 0$ for all $i \geq 2$, we have $e^{-\rho dt} = 1 - \rho dt = \frac{1}{1+\rho dt}$. Substituting these into equation (22) and using the definition of the value function in equation (21) gives:

$$V^{\{c\}}(k(t)) = u(k(t), c(t))dt + \tfrac{1}{1+\rho dt}E_t V^{\{c\}}(k(t+dt)) \tag{23}$$

At this stage, we can apply the Optimality Principle to derive the dynamic programming equation for the maximum value function $V(k(t))$:

$$V(k(t)) = \max_{c(t)\in\Phi(k(t))} \left\{u(k(t), c(t))dt + \tfrac{1}{1+\rho dt}E_t V(k(t+dt))\right\} \tag{24}$$

Denote $dV(k(t)) \equiv V(k(t+dt)) - V(k(t))$. Rearranging equation (24), and taking the limit as $dt \to 0$, give:

$$0 = \max_{c(t)\in\Phi(k(t))} \left\{ u(k(t), c(t)) + E_t \frac{dV(k(t))}{dt} - \rho V(k(t)) \right\} \tag{25}$$

Invoking the Ito lemma to expand the term $E_t dV(k(t))$, we have:

$$\begin{aligned} E_t dV(k) &= E_t \left\{ \begin{array}{c} \left[\left(\frac{\partial V}{\partial k}\right)' g(k,c) + \frac{1}{2} tr\left(\frac{\partial^2 V}{(\partial k)(\partial k)'} \varphi(k,c)'\varphi(k,c) \right) \right] dt \\ + \left(\frac{\partial V}{\partial k}\right)' \varphi(k,c) dw + \left[V(k+\mu) - V(k) \right] dq \end{array} \right\} \\ &= \left[\left(\frac{\partial V}{\partial k}\right)' g(k,c) + \frac{1}{2} tr\left(\frac{\partial^2 V}{(\partial k)(\partial k)'} \varphi(k,c)'\varphi(k,c) \right) \right] dt \\ &\quad + \left[V(k+\mu(k,c)) - V(k) \right] E_t dq \\ &= \left[\left(\frac{\partial V}{\partial k}\right)' g(k,c) + \frac{1}{2} tr\left(\frac{\partial^2 V}{(\partial k)(\partial k)'} \varphi(k,c)'\varphi(k,c) \right) \right] dt \\ &\quad + \lambda(k,c) \left[V(k+\mu(k,c)) - V(k) \right] dt \end{aligned} \tag{26}$$

Substituting equation (26) into equation (25) and simplifying the terms give the HJB equation (6).

A.2 A proof of Theorem 1

In the first part of the proof, we prove that if the function $V(.)$ satisfies the HJB equation (6), then it will not be smaller than any aggregate return obtained by any feasible policy profile. In the second part, we pin down the optimal policy function which results in the aggregate return equal $V(s)$.

First, consider an arbitrary policy $\{c(t)_{t=0}^{\infty}\}$, by the Ito lemma:

$$E_0 d(e^{-\rho t}V(k)) = e^{-\rho t}\left\{ \begin{array}{c} \left[-\rho V(k) + \left(\frac{\partial V}{\partial k}\right)' g(k,c) + \frac{1}{2}tr\left(\frac{\partial^2 V}{(\partial k)(\partial k)'}\varphi(k,c)'\varphi(k,c)\right)\right] \\ +\lambda\left[V(k+\mu(k,c)) - V(k)\right] \end{array} \right\} dt \tag{27}$$

Integrating both sides of equation (27), evaluating the stochastic integration in the time interval $[0,+\infty)$ and taking the expectation given the information set at time zero, give:

$$E_0 \lim_{\tau\to\infty} e^{-\rho\tau}V(k(\tau)) - E_0 V(k(0)) + E_0 \int_0^\infty e^{-\rho t}u(k,c)dt = E_0 \int_0^\infty e^{-\rho t}\left[H_c V(k)\right]dt \tag{28}$$

Invoking the no-Ponzi condition in expression (4) which implies that $\lim_{\tau\to\infty} e^{-\rho\tau}V(k(\tau)) = 0$ and the HJB equation (6) which implies that the RHS of equation (28) is non-positive, all yield:

$$E_0 \int_0^\infty e^{-\rho t}u(k,c)dt \leq V(k(0)) \tag{29}$$

Second, we define the optimal policy function:

$$C(k) = \arg\max_{<c>} \left\{ \begin{array}{c} u(k,c) + \left(\frac{\partial V}{\partial k}\right)' g(k,c) + \\ \frac{1}{2}tr\left(\frac{\partial^2 V}{(\partial k)(\partial k)'}\varphi(k,c)'\varphi(k,c)\right) + \lambda\left[V(k+\mu) - V(k)\right] \end{array} \right\} \tag{30}$$

Comparing equations (6) and (30) implies that if the policy applied is $C(k)$, the RHS of equation (28) is zero. Put it in another way, the equal sign in expression (29) is feasible. Combining the results of two parts, we have:

$$V(k(0)) = \max E_0 \int_0^\infty e^{-\rho t}u(k,c)dt \tag{31}$$

A.3 A proof of Theorem 2

We prove that the maximum value function $V(s)$ is the smallest among those satisfying the constraints in Problem (9) In other words, any function $J(s)$ satisfying the constraints will not be smaller than $V(s)$.

Consider an arbitrary function $J(s)$ that satisfies the constraints in Problem (9) and an arbitrary policy $\{c(t)_{t=0}^{\infty}\}$. By the Ito lemma:

$$E_0 d(e^{-\rho t}J(k)) = e^{-\rho t}\left\{ \begin{array}{c} \left[-\rho J(k) + \left(\frac{\partial J}{\partial k}\right)' g(k,c) + \frac{1}{2}tr\left(\frac{\partial^2 J}{(\partial k)(\partial k)'}\varphi(k,c)'\varphi(k,c)\right)\right] \\ +\lambda\left[J(k+\mu(k,c)) - J(k)\right] \end{array} \right\} dt \tag{32}$$

Integrating both sides of equation (32), evaluating the stochastic integration in the time interval $[0,\infty)$ and taking the expectation given the information set at time zero, give:

$$E_0 \lim_{\tau\to\infty} e^{-\rho\tau}J(k(\tau)) - E_0 J(k(0)) + E_0\int_0^\infty e^{-\rho t}u(k,c)dt = E_0\int_0^\infty e^{-\rho t}\left[H_c J(k)\right]dt \tag{33}$$

Invoking the no-Ponzi condition in expression (4) which implies that $\lim_{\tau\to\infty} e^{-\rho\tau}J(k(\tau)) = 0$ and the constraints in Problem (9) which implies that the RHS of equation (33) is non-positive, give:

$$J(k(0)) \geq E_0\int_0^\infty e^{-\rho t}u(k,c)dt \tag{34}$$

Since expression (34) holds with all policy plans, it holds with the optimal policy profile. Evaluating the RHS of expression (34) at the optimal policy profile which leads to the maximum value function, we have:

$$J(s) \geq V(s) \text{ with all } s \tag{35}$$

The weak inequality in expression (35) simply implies that the maximum value function is the smallest among those satisfying the constraints in Problem (9) and uniquely solves the minimization problem with any positive weights $w(s) > 0$.

References

Aghion, P. and P. Howitt (1992). A model of growth through creative destruction. Econo metrica 60(2), 323-51.

Arruda, E. and J. DoVal (2006). Approximate dynamic programming based on expansive projections. In Proceedings on 45th IEEE Conference on Decision and Control, pp. 5537-42.

Aruoba, S., F. Jesus, and F. Juan (2006). Comparing solution methods for dynamic equilibrium economies. Journal of Economic Dynamics and Control 30(12), 2477-08.

Bellman, R. (1957). Dynamic Programming. New Jersey: Princeton University Press.

Das, S. (2002). The surprise element: Jumps in interest rates. Journal of Econometrics 106(1), 27-65.

Farias, D. and B. Roy (2004). On constraint sampling for the linear programming approach to approximate dynamic programming. Mathematics of Operations Research 29 (3), 462-478.

Gaspar, J. and K. Judd (1997). Solving large-scale rational-expectations models. Macroeconomic Dynamics 1(1), 45-75.

Grafton, Q., T. Kompas, and H. Pham (2006). The economic payoffs from marine reserves: Resource rents in a stochastic environment. Economic Record 82 , 469-80.

Han, J. and B. Roy (2009). Control of diffusions via linear programming. In G. Infanger (Ed.), Stochastic dynamic programming, the state of the art. In honor of George B. Dantzig. Upcoming.

Hedlund, S. (2003). Computational Methods for Optimal Control of Hybrid Systems. Ph. D. thesis, Department of Automatic Control, Lund Institute of Technology, Sweden.

Judd, K. (1996). Approximation, perturbation, and projection methods in economic analysis. In H. Amman, D. A. Kendrick, and J. Rust (Eds.), Handbook of Computational Economics, Volume 1, Chapter 12, pp. 509-85. Elsevier.

Judd, K. (1998). Numerical methods in economics. Cambridge: The MIT Press.

Judd, K. and S. Guu (1993). Perturbation solution methods for economic growth model. In H. Varian (Ed.), Economic and financial modeling with mathematica. New York: Springer Verlag.

Manne, A. (1960). Linear programming and sequential decisions. Management Sci ence 6(3), 259-67.

Piazzesi, M. (2005). Bond yields and the federal reserve. Journal of Political Economy 113(2), 311-44.

Ross, S. (1970). Applied Probability Models with Optimization Applications. San Francisco: Holden-Day.

Taylor, J. and H. Uhlig (1990). Solving nonlinear stochastic growth models: A comparison of alternative solution methods. Journal of Business and Economic Statistics 8(1), 1-17.

Walde, K. (1999). Optimal saving under poisson uncertainty. Journal of Economic Theory 87(2), 194-17.

Tables

Paramters	**Notation**	**Values**
Fish price coefficient	P	700
Fishing cost coefficient	c	17
Price elasticity	α	0.81
Biological intrinsic rate	r	0.2985
Discount rate	ρ	0.05
Maximum Carrying Capacity	MCC	1 (million tons)
Standard error of the natural shock	$\varphi(s)$	0.05s
Likelihood of the negative shock	λ	0.1
Magnitude of the negative shock	$\mu(s)$	-0.13s

Table 1. Numerical values for the paramters in the unidimensional problem

	Linear programming	Perturbation	Projection
Maximum error	8.6323e-4	3.3214e-3	1.36e-4
Average error	4.6487e-5	3.4357e-4	2.0467e-5
Computation time (s)	1	0.8	140

Table 2. Performance of the parametric techniques in the unidimensional case with polynomial order $n = 6$

	Linear programming	Perturbation	Projection
Maximum error	6.5833e-5	1.1836e-3	8.905e-5
Average error	9.2673e-6	8.4807e-5	1.5723e-5
Computation time (s)	1.2	3	380

Table 3. Performance of the parametric techniques in the unidimensional case with polynomial order $n = 10$

	Linear programming	Perturbation	Projection
Maximum error	2.6869e-3	5.10e-3	1.2769e-3
Average error	1.1195e-4	7.1e-4	1.3096e-4
Computation time	5	1	1050

Table 4. Performance of the parametric techniques in the multidimensional problem

Size of the sensitivity (β)	Number of parameter combinations (β^5)	Linear programming technique	Perturbation	Projection
3	243	0:21 hour	0:04 hour	3.0 days
4	1,024	1:26 hour	0:17 hour	12.5 days
5	3,125	4:20 hour	0:52 hour	40.0 days
6	7,776	10:48 hour	2:09 hour	94.5 days

Table 5. Computation time and the size of the sensivity analysis

	$\eta = 1$	$\eta = 2$	$\eta = 3$
$m = 1$	2601×5	132,651×5	6,765,201×5
$m = 2$	132,651×15	6,765,201×15	345,025,251×15
$m = 3$	6,765,201×35	345,025,251×35	17,596,287,801×35

Table 6. Sizes of the linear programming scheme with various dimensions

Figures

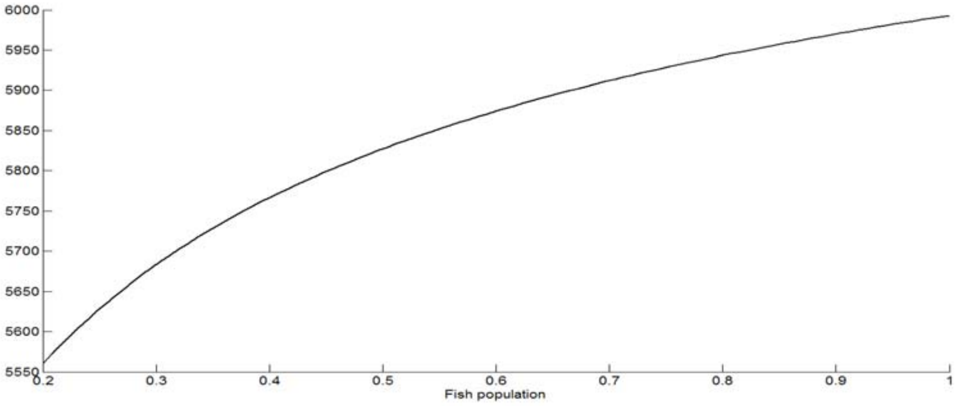

Figure 1: Unidimensional maximum value function

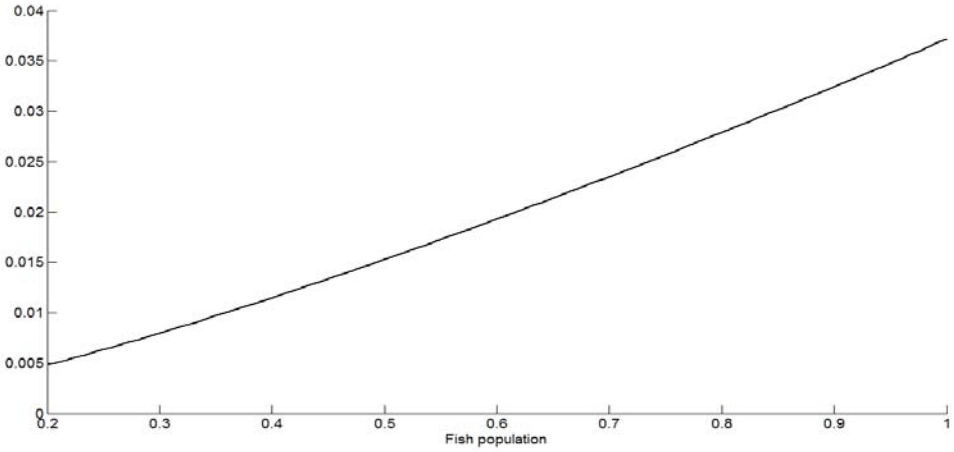

Figure 2: Unidimensional optimal harvest function

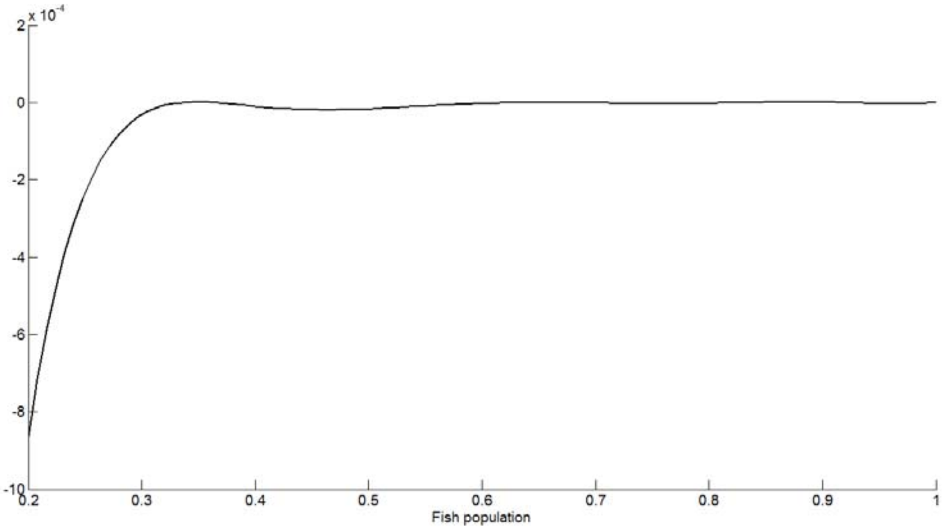

Figure 3: Undimensional approximation errors

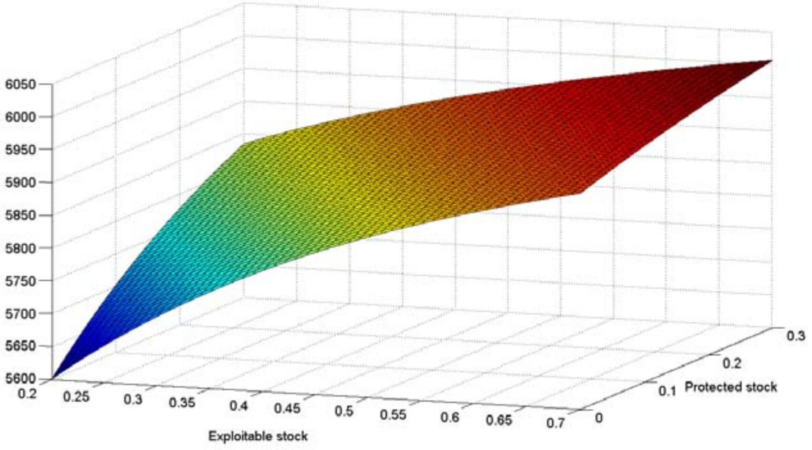

Figure 4: Multidimensional maximum value function

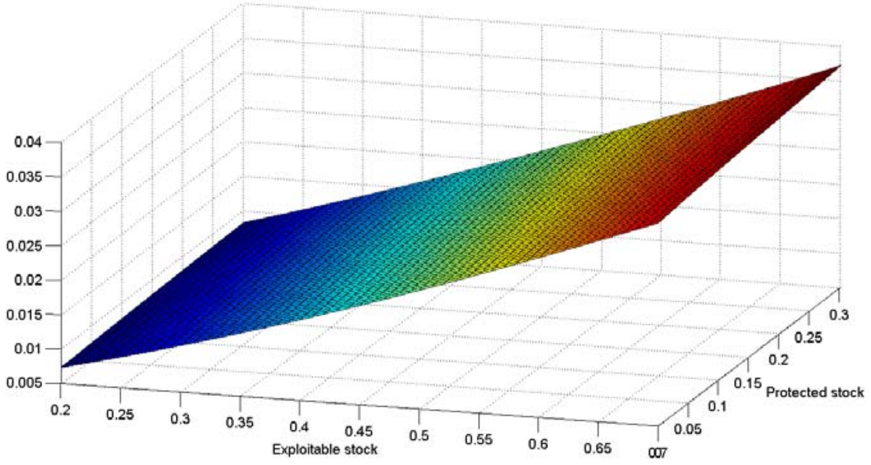

Figure 5: Multidimensional optimal harvest function

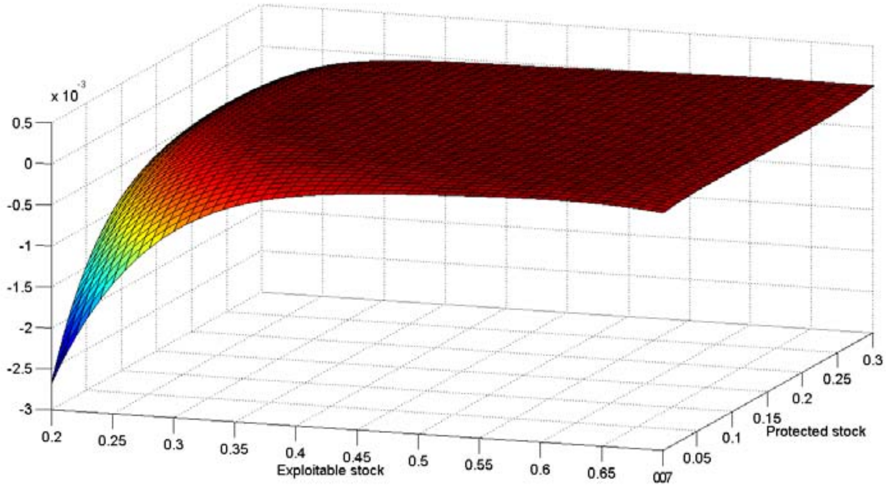

Figure 6: Multidimensional approximation errors