

NC STATE UNIVERSITY

Center for Environmental and Resource Economic Policy
College of Agriculture and Life Sciences
<https://cenrep.ncsu.edu>

Solving Optimal Switching Models

Paul L. Fackler

Center for Environmental and Resource Economic Policy
Working Paper Series: No. 18-019
September 2018

Suggested citation: Fackler, Paul L. (2018). Solving Optimal Switching Models. (CEnREP Working Paper No. 18-019). Raleigh, NC: Center for Environmental and Resource Economic Policy.



Solving Optimal Switching Models

Paul L. Fackler*

September 26, 2018

Abstract

Numerous control problems in economics can be characterized as optimal switching problems, including optimal stopping and entry/exit problems. With certain notable exceptions, solutions to such problems must be computed numerically. This paper discusses an approach to obtaining a numerical solution in a fairly general setting and describes a user-friendly MATLAB implementation that simplifies that process.

Keywords: stochastic control, optimal switching, real options, computational methods

JEL classification codes: C61, C63, C88

*The author is an Associate Professor at North Carolina State University.

Mail: Department of Agricultural and Resource Economics

NCSU, Box 8109

Raleigh NC, 27695, USA

e-mail: paul.fackler@ncsu.edu

Web-site: <http://www4.ncsu.edu/~pfackler/>

© 2004, Paul L. Fackler

Solving Optimal Switching Models

Abstract

Numerous control problems in economics can be characterized as optimal switching problems, including optimal stopping and entry/exit problems. With certain notable exceptions, solutions to such problems must be computed numerically. This paper discusses an approach to obtaining a numerical solution in a fairly general setting and describes a user-friendly MATLAB implementation that simplifies that process.

Keywords: stochastic control, optimal switching, real options, computational methods

JEL classification codes: C61, C63, C88

Many problems in stochastic control involve situations in which an agent can choose among a discrete set of states or regimes. The choice of a control can be thought of as the choice of the regime. In addition, a diffusion process drives one or more state variables that affect the returns associated with each of the regimes. The total returns to the agent are also affected by the existence of costs associated with switching among the regimes.

Optimal switching models often arise in pricing so-called real options. The recent collections of Brennan and Trigeorgis and Schwartz and Trigeorgis provide an introduction to this area and numerous examples. The widely read text by Dixit and Pindyck also contains numerous examples of optimal switching models. The real options literature recognizes and attempts to value such things as the flexibility to defer action, to change from one activity to another, to abandon an investment or to default on a project.

The simplest optimal switching models are optimal stopping problems, where one of the

33 regimes represents continuation and other represents a permanent “stopped” state. The Amer-
34 ican option pricing problem is a well known example. More complicated optimal switch-
35 ing models allow for movement back and forth among regimes and may have more than two
36 regimes. A well known example introduced in McDonald and Siegel, McDonald and Siegel,
37 and Brennan and Schwartz, is that of firm entry/exit. In such problems, a firm can make an
38 investment in an asset that produces an output with a stochastic price. When the price is high
39 enough, it is worth activation though investment in the asset. When the price is low enough,
40 however, the firm may abandon its investment and become inactive. In the basic entry/exit
41 model there are two regimes, active and inactive. An extension of this model adds a third
42 regime that allows for temporary suspension of production.

43 Brekke and Oksendal have provided general solution conditions for optimal switching mod-
44 els that can be expressed as a set of functional complementarity conditions. In most cases
45 explicit solutions to these conditions cannot be obtained. This paper discusses how numer-
46 ical solutions can be obtained using function approximation and collocation. The approach
47 results in a type of problem known as an Extended Vertical Linear Complementarity Prob-
48 lem (EVLCP) for which a number of solution algorithms exist. A user friendly interface for
49 defining and solving optimal switching models using MATLAB is documented and illustrated.

50 **Problem Statement and Optimality Conditions**

51 The general optimal switching model applies to problems in which an agent must choose, at
52 each moment, among m regimes, $R \in \{1, \dots, m\}$. The agent can move from regime i to j at a

53 cost of C_{ij} (which may be arbitrarily high, thereby ruling out such a movement). C represents
54 a lump-sum switching cost with $C_{ii} = 0$, i.e., there is no lump sum cost to remaining in the
55 current regime. The agent also receives a flow of payments per unit time of $f(S, R)$, which
56 depends on both the active regime and on a continuous d -dimensional state process S that is
57 described by

$$58 \quad dS = \mu(S, R)dt + \sigma(S, R)dW. \quad (1)$$

59 The agent desires to maximize over an infinite time horizon the discounted value, discounted
60 at rate $\rho(S)$, of the flow of payments received less any switching costs incurred.

61 The solution of the control problem consists of a value function $V(S, R)$ and a control
62 function $A(S, R)$. By Ito's Lemma the expected rate of appreciation of the value function

$$63 \quad \frac{dE[V(S, R)]}{dt} = \mathcal{L}V(S, R) = \sum_i \mu_i \frac{\partial V(S, R)}{\partial S_i} + \frac{1}{2} \sum_i \sum_j \Sigma_{ij} \frac{\partial^2 V(S, R)}{\partial S_i \partial S_j} \quad (2)$$

64 where $\Sigma_{ij} = \sum_k \sigma_{ik} \sigma_{jk}$. Brekke and Oksendal (Theorem 3.4) have shown that the optimal
65 value function $V(S, R)$ satisfies

$$66 \quad \rho(S)V(S, R) \geq f(S, R) + \mathcal{L}V(S, R) \quad (3)$$

67 and the $m - 1$ conditions

$$68 \quad V(S, R) \geq V(S, x) - C_{Rx}, \quad \forall x \neq R \quad (4)$$

69 Furthermore, one of these m conditions must be satisfied with equality at each (S, R) . Which
70 one is satisfied with equality determines the optimal policy $A(S, R)$. Thus, if $V(S, R) =$
71 $V(S, x) - C_{Rx}$, for some x , it is optimal at S to switch from R to x . Otherwise it is optimal to
72 remain in regime R and the first condition is satisfied with equality.

73 There is a simple economic intuition behind these conditions. The value function can be
74 thought of as the value of the assets that generate the payment flows. The total rate of return
75 when regime R is active equals the current return flow $f(S, R)$ plus the expected rate of capital
76 appreciation $dE[V(S, R)]/dt$. Thus, the first condition states that the rate of return obtainable
77 by investing V dollars must be at least as great as the total rate of return generated by the assets
78 if one remains in regime R . The second condition states that the value function must be at least
79 as great as the value that could be obtained by switching regimes.

80 Anticipating the numerical approach discussed in the next section, the optimality conditions
81 can be written in the following equivalent form

$$82 \quad 0 = \min \left(\beta(S, R) - f(S, R), \min_{x \neq R} V(S, R) - V(S, x) + C_{Rx} \right) \quad (5)$$

83 where

$$84 \quad \beta(S, R) = \rho(S)V(S, R) - \mathcal{L}V(S, R) \quad (6)$$

85 It should also be pointed out that this condition does not fully characterize the solution. In
86 particular, additional regularity conditions are needed to uniquely define V . Essentially these
87 amount to conditions that rule out explosive growth in the value function.

88 To illustrate the framework, an example from Brekke and Oksendal is reviewed. Consider
89 a mine currently containing Q units of ore. The mine is either idle ($R = 1$) or ore is extracted
90 at rate hQ ($R = 2$) with a fixed cost of k incurred. The transition equation for Q is thus

$$91 \quad dQ = \begin{cases} 0 & \text{if } R = 1 \\ -hQdt & \text{if } R = 2 \end{cases} \quad (7)$$

92 The current price at which ore can be sold evolves according to a geometric Brownian motion

$$93 \quad dP = \mu P dt + \sigma P dW \quad (8)$$

94 The flow of returns to the mine is

$$95 \quad f(Q, P, R) = \begin{cases} 0 & \text{if } R = 1 \\ hQP - k & \text{if } R = 2 \end{cases} \quad (9)$$

96 The firm incurs fixed startup and shutdown costs of C_{12} and C_{21} and uses a fixed discount rate
97 of ρ .

98 The solution conditions are

$$99 \quad 0 = \min \left(\rho V(Q, P, 1) - \mu P V_p(Q, P, 1) - \frac{1}{2} \sigma^2 P^2 V_{PP}(Q, P, 1), \right. \\ \left. V(Q, P, 1) - V(Q, P, 2) + C_{12} \right) \quad (10)$$

100 and

$$101 \quad 0 = \min \left(\rho V(Q, P, 2) - \mu P V_p(Q, P, 2) - \frac{1}{2} \sigma^2 P^2 V_{PP}(Q, P, 2) \right. \\ \left. + hQV_Q(Q, P, 2) - (hQP - k), V(Q, P, 2) - V(Q, P, 1) + C_{21} \right) \quad (11)$$

102 As Brekke and Oksendal point out, the problem can be simplified by defining $y = QP$ and

103 noting that

$$104 \quad dy = \begin{cases} \mu y dt + \sigma y dW & \text{if } R = 1 \\ (\mu - h)y dt + \sigma y dW & \text{if } R = 2 \end{cases} \quad (12)$$

105 and

$$106 \quad f(Q, P, R) = f(y, R) = \begin{cases} 0 & \text{if } R = 1 \\ hy - k & \text{if } R = 2 \end{cases} \quad (13)$$

107 Expressing the value function in terms of y and R , the optimality conditions are

$$108 \quad 0 = \min \left(\rho V(y, 1) - \mu y V_y(y, 1) - \frac{1}{2} \sigma^2 y^2 V_{yy}(y, 1), V(y, 1) - V(y, 2) + C_{12} \right) \quad (14)$$

109 and

$$110 \quad 0 = \min \left(\rho V(y, 2) - (\mu - h)y V_y(y, 2) - \frac{1}{2} \sigma^2 y^2 V_{yy}(y, 2) - (hy - k), \right. \\ \left. V(y, 2) - V(y, 1) + C_{21} \right) \quad (15)$$

111 $\forall y \in [0, \infty)$.

112 Numerical Solution Methods

113 Optimal switching models generally require numerical approximations. In simple models the
 114 functional form of the solution may be known and numerical methods are required only to
 115 compute a limited set of parameter values. In particular, problems with a single geometric
 116 Brownian motion state, such as the one-dimensional formulation of the example given in the
 117 previous section, often can be solved in this way (see Dixit and Pindyck for more examples
 118 and nearly explicit solutions).

119 For general problems, however, the functional form of the solution may not be known.
 120 Projection methods (Judd, Miranda and Fackler) using complete families of approximating
 121 functions represent a natural way to find approximate solutions to such models. Suppose that
 122 $V(S, i)$ is approximated by $\phi(S)\theta_i$, where ϕ represents a set of n basis functions for a family of
 123 approximating functions and θ_i is an n -vector of coefficients for the value function associated
 124 with the i th regime. Define the approximate differential operator

$$125 \quad \beta(S, i) = \rho(S)\phi(S) - \mu(S, i)\phi'(S) + \frac{\sigma^2(S, i)}{2}\phi''(S) \quad (16)$$

126 The inequality conditions can now be written as

$$127 \quad \beta(S, i)\theta_i - f(S, i) \geq 0 \quad (17)$$

128 and

$$129 \quad \phi(S)\theta_i - \phi(S)\theta_j + C_{ij}(S) \geq 0, \quad \forall j \neq i \quad (18)$$

130 Values of the θ_i can be obtained by collocation, which solves the optimality conditions at a
131 set of n nodal state values $\{s_k\}$. Define the $n \times n$ matrices Φ and B_i as the functions $\phi(S)$ and
132 $\beta(S, i)$ evaluated at the nodal values. Similarly, define f_i to be the n -vector of values of $f(S, i)$
133 evaluated at the n nodal state values.

134 The problem can now be stated as an extended vertical linear complementarity problem
135 (EVLCP),¹ which seeks a solution z to

$$136 \quad 0 = \min(M_1z + q_1, M_2z + q_2, \dots, M_mz + q_m). \quad (19)$$

137 where the M_i are each $N \times N$ and the q_i are each $N \times 1$ and where the min operator is applied
138 element-wise. The EVLCP could also be written as $w_i = M_i z + q_i \geq 0$ for $i = 1, \dots, m$ and

$$139 \quad \prod_{i=1}^m w_i = 0 \quad (20)$$

140 where the product is taken element-wise. The solution to an EVLCP problem thus requires
141 that each of the w_i be nonnegative and that for each of the N elements, at least one of the m w_i
142 values is exactly zero. Unlike more common complementarity conditions (such as the Karush-
143 Kuhn-Tucker conditions) the complementarity here extends over m variables rather than only
144 over two and hence cannot be written as a vector orthogonality condition.

¹This generalization of the standard linear complementarity problem has a number of names in the literature including the extended generalized order LCP (Gowda and Sznajder).

145 In the general form z represents the m column vectors θ_i stacked vertically. The M_i are
 146 given by

$$147 \quad M_i = e_i e_i^\top \otimes B_i + (I_m - \mathbf{1}_m e_i^\top) \otimes \Phi \quad (21)$$

148 and the q_i are given by

$$149 \quad q_i = \begin{bmatrix} C_{1i} \mathbf{1}_n \\ \cdots \\ C_{mi} \mathbf{1}_n \end{bmatrix} - [e_i \otimes f_i] \quad (22)$$

150 where $\mathbf{1}_m$ is a column vector composed of m ones and e_i is the i th column of an $m \times m$ identity
 151 matrix. Thus

$$152 \quad M_1 = \begin{bmatrix} B_1 & 0 & \cdots & 0 \\ -\Phi & \Phi & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ -\Phi & 0 & \cdots & \Phi \end{bmatrix}, \quad q_1 = \begin{bmatrix} -f_1 \\ C_{21} \mathbf{1}_n \\ \cdots \\ C_{m1} \mathbf{1}_n \end{bmatrix} \quad (23)$$

$$153 \quad M_2 = \begin{bmatrix} \Phi & -\Phi & \cdots & 0 \\ 0 & B_2 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots \\ 0 & -\Phi & \cdots & \Phi \end{bmatrix}, \quad q_2 = \begin{bmatrix} C_{12} \mathbf{1}_n \\ -f_2 \\ \cdots \\ C_{m2} \mathbf{1}_n \end{bmatrix} \quad (24)$$

154 etc.

155 A number of solution approaches for solving EVLCPs have been proposed, each based on
 156 solution approaches that have proven useful for related problems. Cottle and Dantzig proposed
 157 a pivoting strategy for a related problem that is based on Lemke's algorithm for solving or-
 158 dinary LCPs (Lemke).² Details on modifications to the Lemke-based algorithm for EVLCPs
 159 are available from the author. In the first discussion of the application of EVLCPs to con-
 160 trol theory, Sun proposed an iterative algorithm similar in spirit to the Projected Successive

²Cottle and Dantzig proposed this algorithm for what they termed the generalized LCP, which has since be-
 come known as the vertical LCP. This is a special case of the EVLCP in which one of the M_i is an identity matrix
 and the associated q_i is a vector of zeros.

161 Over-Relaxation (PSOR) method for solving LCPs. This method, however, can be unstable for
 162 solving the kind of problems considered here. Recently, a smoothing Newton method using a
 163 so-called aggregation function (also known as an entropy function) has been proposed by Qi
 164 and Liao. In numerical trials for the kind of problems considered here, the smoothing Newton
 165 method has the best performance characteristics.

166 Another approach is to reformulate the problem using alternative semi-smooth functions.
 167 The Fischer-Burmeister function

$$168 \quad \Phi^-(x, y) = x + y - \sqrt{x^2 + y^2} \quad (25)$$

169 has the same roots as $\min(x, y)$ and has been found in practice to work well with Newton type
 170 methods for solving root-finding problems. Thus, the problem of solving

$$171 \quad 0 = \min(M_1z + q_1, M_2z + q_2) \quad (26)$$

172 is equivalent to that of solving

$$173 \quad 0 = \Phi^-(M_1z + q_1, M_2z + q_2) \quad (27)$$

174 This function can be extended recursively by defining

$$175 \quad F_i(z) = \Phi^-(M_i z + q_i, F_{i+1}(z)) \quad (28)$$

176 for $i = 1, \dots, m - 1$, with $F_m(z) = M_m z + q_m$.³ The problem of solving

$$177 \quad 0 = \min(M_1 z + q_1, \dots, M_m z + q_m) \quad (31)$$

178 is equivalent to solving $0 = F(z) = F_1(z)$.

179 The derivative of F can be found recursively

$$180 \quad \frac{dF_i(z)}{dz} = \text{diag} \left(\frac{d\Phi^-(M_i z + q_i, F_{i+1}(z))}{dx} \right) M_i + \text{diag} \left(d \frac{\Phi^-(M_i z + q_i, F_{i+1}(z))}{dy} \right) \frac{dF_{i+1}}{dz} \quad (32)$$

181 Letting $c_i = d\Phi^-(M_i z + q_i, F_{i+1}(z))/dx$ and $d_i = d\Phi^-(M_i z + q_i, F_{i+1}(z))/dy$ the deriva-

182 tive of F can be written as

$$183 \quad \frac{dF(z)}{dz} = \sum_{i=1}^{m-1} \text{diag} \left(c_i \prod_{j=1}^{i-1} d_j \right) M_i + \text{diag} \left(\prod_{j=1}^{m-1} d_j \right) M_m \quad (33)$$

184 This is convenient because it maintains the sparsity structure of the M_i .

185 An alternative to the Fischer-Burmeister function is a function proposed by Qi, referred to

186 here as Qi's Ψ function:⁴

$$187 \quad \begin{aligned} \Psi(x, y) &= x + y - |y| \left(1 + \frac{1}{3} \left(\frac{x}{y} \right)^2 \right) && \text{if } |y| \geq |x| \\ &= x + y - |x| \left(1 + \frac{1}{3} \left(\frac{y}{x} \right)^2 \right) && \text{if } |x| > |y| \end{aligned} \quad (35)$$

³This version could be called a backwards version. One can also define a forwards version:

$$G_i(z) = \Phi^-(G_{i-1}(z), M_i z + q_i) \quad (29)$$

for $i = 2, \dots, m$, with $G_1(z) = M_1 z + q_1$. One then solves $0 = G(z) = G_m(z)$. The derivative is

$$\frac{dG(z)}{dz} = \text{diag} \left(\prod_{j=2}^m c_j \right) M_1 + \sum_{i=2}^m \text{diag} \left(d_i \prod_{j=i+1}^m c_j \right) M_i \quad (30)$$

⁴The Qi Ψ function and its derivatives can be evaluated efficiently using

	$\Psi(x, y)$	$\Psi_x(x, y)$	$\Psi_y(x, y)$
$ x > y $ and $x > 0$	$y \left(1 - \frac{1}{3} \frac{y}{x} \right)$	$\frac{1}{3} \left(\frac{y}{x} \right)^2$	$1 - \frac{2}{3} \frac{y}{x}$
$ x > y $ and $x < 0$	$2x + y \left(1 + \frac{1}{3} \frac{y}{x} \right)$	$2 - \frac{1}{3} \left(\frac{y}{x} \right)^2$	$1 + \frac{2}{3} \frac{y}{x}$
$ y \geq x $ and $y > 0$	$x \left(1 - \frac{1}{3} \frac{x}{y} \right)$	$1 - \frac{2}{3} \frac{x}{y}$	$\frac{1}{3} \left(\frac{x}{y} \right)^2$
$ y \geq x $ and $y < 0$	$2y + x \left(1 + \frac{1}{3} \frac{x}{y} \right)$	$1 + \frac{2}{3} \frac{x}{y}$	$2 - \frac{1}{3} \left(\frac{x}{y} \right)^2$

(34)

188 To date theoretical results on the solvability of EVLCPs are mostly limited to the case in
189 which all row-representative matrices associated with the M_i are nonsingular with determinants
190 of the same sign, the so-called W property (see Gowda and Sznajder). Unfortunately, in the
191 present case, singularity of a row representative matrix can result, and hence these results do
192 not apply. It also means that it is possible for the Lemke-based and smoothing Newton based
193 methods to fail. In practice, however, this does not seem to be a problem.

194 The quality of the approximate solution depends on both the choice of the family of approx-
195 imating functions and the set of nodal values used to obtain the collocation solution. In general
196 $V(S, R)$ is not smooth, but exhibits discontinuity in the second derivative at points for which
197 it is optimal to switch from R to another regime.⁵ For this reason polynomial approximations
198 generally perform poorly. A better alternative is to use either piecewise linear functions (using
199 finite differences to approximate ϕ' and ϕ'') or cubic spline functions because these functions
200 are not as adversely affected by the derivative discontinuities.

⁵If $C_{ij} = C_{ji} = 0$ at a point S for which is it optimal to switch between i and j , then the value function will exhibit discontinuity in the third derivative (see the related discussion of super-contact conditions in Dumas)

201 **A MATLAB Implementation**

202 To specify an optimal switching model, an analyst must define the functions $f(S, R)$, $\rho(S)$,
203 $\mu(S, R)$ and $\sigma(S, R)$ and assign values to any parameters that these function use as well as to
204 the cost parameter matrix C . To solve the model using the function approximation approach
205 described in the previous section also requires that the analyst specify the family of approxi-
206 mating functions to be used and the nodal values of S at which to evaluate the complementarity
207 conditions.

208 This section describes an implementation in MATLAB that makes the solution process rela-
209 tively simple (the code described here can be downloaded from the author's website). The first
210 step is to code a model function file according to the following template:

```
211     function out=func(flag,S,R,additional parameters)
212     switch flag
213     case 'f'
214         out = reward function f(S,R);
215     case 'mu'
216         out = drift function mu(S,R);
217     case 'sigma'
218         out = diffusion function sigma(S,R);
219     case 'rho'
220         out = discount rate rho(S);
221     end
```

222 The procedure will be passed a string variable `flag`, an $nm \times d$ matrix of values of the
223 continuous state S , a scalar value of the regime R and any additional parameters needed to
224 evaluate the model functions. When the `flag` variable is `f` the procedure should return an
225 $nm \times 1$ vector, when `flag` is `mu` it should return an $nm \times d$ matrix, when `flag` is `sigma` it
226 should return an $nm \times d \times d$ array and when `flag` is `rho` it should return an $nm \times 1$ vector

227 (or a scalar if ρ is a constant).

228 The model specification is completed by defining a structure variable `model` with the fol-
229 lowing three fields:⁶

230	<code>func</code>	the name of the model function file
	<code>params</code>	a cell array with the parameters to pass to the model function file
	<code>cost</code>	the $m \times m$ switching cost matrix C

231 The family of approximating functions is specified by using the procedures in the Comp-
232 Econ Toolbox described in Miranda and Fackler which is available from Fackler's website.
233 The toolbox procedures can be used to create a structure variable called `fspace` containing
234 all the information needed to define the necessary basis matrices.

235 The main solution procedure is a procedure called `ossolve` which has the following syn-
236 tax:

```
237 [cv, snodes, x, xindex]=ossolve(model, fspace, snodes, cv);
```

238 The first two inputs have already been described and are the only ones needed. The third input,
239 if passed, is a set of nodal values of S . If S is one-dimensional, this should be a simple column
240 vector. If S is d -dimensional, `snodes` should be passed as a cell array of column vectors.
241 These will be expanded by `ossolve` into a grid of values, the size of which, n , will equal the
242 product of the lengths of the vectors. The fourth input, if passed, is an $n \times m$ matrix of initial
243 values for the coefficients of the value function approximations.

244 The first output returned by the procedure is an $n \times m$ matrix of coefficients of the value
245 function approximations. The value function can now be evaluated at arbitrary values of (S, R)
246 using the CompEcon Toolbox function call

⁶Structure variables in MATLAB are user defined data type with named fields. The data in a field is accessed using the syntax `variablename.fieldname`. Cell arrays are data types with fields accessed by index number, e.g., `variablename{i, j}`.

247 `V=funeval (cv (:, R) , fspace, S) ;`

248 The marginal value (shadow price) function can be obtained in a similar fashion using

249 `dV=funeval (cv (:, R) , fspace, S, 1) ;`

250 The second and third outputs provide the nodal values used in finding the solution ($nm \times d$)
251 and the optimal regime choice at the nodal values ($nm \times m$). This provides a representation of
252 optimal decision rule. This representation, of course, is only as accurate the mesh size of the
253 grid of nodal points.

254 The fourth output `xindex` ($m \times 6$) is useful for characterizing on the optimal decision
255 rule in one dimensional problems. For each of the m regimes it contains information about the
256 lower and upper boundaries of the no-switch region. The first column contains the approximate
257 location of the lower bound, the second column contains the regime number that it is optimal to
258 switch to at that point and the third column contains the difference in the derivatives of the value
259 functions for the two regimes at the approximate switch point (this difference should be zero
260 and thus provides a check on the solution). Columns 4 through 6 provides similar information
261 at the upper bound of the no-switch region. `xindex` is returned as an empty matrix when
262 $d > 1$.

263 A separate utility is also provided to evaluate the optimal decision rule and the value func-
264 tion at arbitrary points and works for any value of d . It is called with the following syntax:

265 `[regime, V, dV]=osoptimum (S, R, cv, model, fspace) ;`

266 The first input is a $k \times d$ matrix of arbitrary values of S . The second input R is either scalar or
267 a $k \times 1$ vector of values of the regime number. The remaining inputs are the coefficient matrix
268 returned from `ossolve` and the model definition and function definition structures passed to
269 `ossolve`.

270 The utility returns a best guess of the optimal decision rule, the value function and the
 271 marginal value (shadow price) function at the points (S, R) (the sizes of the outputs are $k \times 1$,
 272 $k \times 1$ and $k \times d$, respectively). The optimal decision rule is determined by calculating the
 273 value function at (S, R) as well as the value function at S for the other regimes less the cost of
 274 switching to them from R . The best switch is computed and then compared to the no-switch
 275 strategy.

276 It is, however, impossible to determine with complete satisfaction the location of no-switch
 277 boundary due to the approximations inherent in the solution approach. The value function for
 278 any given regime should be exactly equal to the value of the best switch strategy outside the no-
 279 switch region and should be strictly greater everywhere within the region. As a compromise,
 280 the utility assumes that a switch should occur unless

$$281 \quad V(S, R) - \max_{x \neq R} (V(S, x) - C_{rx}) > \epsilon |V(S, R)| \quad (36)$$

282 where ϵ is set to the default value of 10^{-5} (this value can be set by the user with the call
 283 `optset('osoptimum','tol',epsilon)`, where `epsilon` is any desired value). A
 284 useful check on the accuracy of this utility is to compare the output of the call

```
285     regime=osoptimum(snodes,i,cv,model,fspace);
```

286 to the values of `x(:,i)` returned by `ossolve` (they should be identical).

287 **Practicalities and Extensions**

288 There are three choices an analyst must make in using `ossolve`: the family of approximating
 289 functions, the nodal values at which to solve the complementarity conditions and the initial

290 coefficient values to use. As already mentioned, the use of smooth families of functions, like
291 polynomials, is not recommended for switching models because of the discontinuities that
292 occur in the second derivatives of the value function at the optimal switching points. Instead
293 either cubic splines or piecewise linear functions with finite difference derivatives are better
294 choices. These families can be defined using the CompEcon toolbox calls

```
295     fspace=fundefn('spli',n,a,b);
```

296 or

```
297     fspace=fundefn('lin',n,a,b);
```

298 In either case n is the number of basis functions needed to define the family and a and b are
299 the lower and upper bounds of the approximation interval. When S is d -dimensional, n , a and
300 b should all be $1 \times d$ vectors and tensor product basis functions will be formed from the basis
301 functions for each dimension (see Miranda and Fackler, chapter 6, for more details).

302 The syntax above defines cubic spline or piecewise linear functions with evenly spaced
303 breakpoints on the interval $[a, b]$. The choice of the approximation interval is very important
304 for the accuracy of the solution. If the interval is too wide, a large number of nodal points
305 will be needed to obtain accurate solutions. If the interval is too small, endpoint problems may
306 corrupt the solution. A general rule of thumb is that the interval should wide enough to include
307 values to which the ergodic distribution of S (if it exists) assigns non-trivial probability. If no
308 ergodic distribution exists (as with the widely used geometric Brownian motion), the interval
309 should be large enough relative to the discount rate. Essentially this means that if the process
310 starts near a switch point, the probability that it reaches the approximation boundary within
311 short period of time is small. How short a period of time is appropriate in this calculation

312 is determined by how fast the future is discounted. From a practical point of view, one can
313 experiment with alternative values of a and b . These should be made extreme enough that the
314 results of interest (generally the values of the switching points and the value function near these
315 points) are relatively unaffected by the choice.

316 The CompEcon toolbox also allows one to specify spline or piecewise linear functions with
317 unevenly spaced breakpoints. This may be useful if more accuracy is required. Putting more
318 breakpoints in regions of non-smooth behavior, especially around the switch points, can result
319 in much greater accuracy for the same order of approximation.

320 Some choice of nodal values must also be made. It is probably best to use the default
321 values provided by the CompEcon Toolbox unless one has strong reasons for another choice.
322 For splines and piecewise linear functions, the default nodal values are the breakpoints (with
323 an extra point added near each end for cubic splines). Evaluation at the breakpoints ensures
324 that the resulting basis matrices are well conditioned. The default nodal values used can be
325 obtained prior to calling `ossolve` with the call⁷

```
326     snodes=funnode ( fspace ) ;
```

327 Starting values for the function coefficients can also be passed to the complementarity
328 solver. If initial values are not passed, `ossolve` will compute default values by solving for
329 the function coefficients that approximate a suboptimal value function with a decision rule that
330 never switches the currently active regime. This may be a reasonable choice for starting values
331 if nothing is known about the solution. If comparative static exercises are being performed
332 by solving a model multiple times at a set of alternative model parameters, a good choice of

⁷For multidimensional state variables ($d > 1$), `snodes` is returned as a $1 \times d$ cell array of column vectors. This can be transformed into a d -column matrix of points using `S=gridmanke (snodes)`.

333 starting values will generally be the values obtained from a previous call to `ossolve`.

334 Several extensions are of the basic model increase the flexibility with which it can be ap-
335 plied. First, suppose that, in addition to the decision maker choosing which regime is active, it
336 is also possible that random exogenous shifts of regime occur. This possibility is described in
337 greater detail in another paper, but suffice is to say here that a model of this type can be spec-
338 ified by defining two additional $m \times m$ matrices. The first of these, Λ , contains the Poisson
339 jump intensities associated with an exogenous switch from regime i to regime j . The second
340 matrix, Q , contains the costs imposed if such a jump occurs. Both matrices should have zeros
341 on the diagonal.

342 To specify a model of this type, the `model` structure variable should contain two additional
343 fields, `L` and `Q`, which contain the two matrices (if either `L` or `Q` is missing or empty, it is
344 assumed to imply an $m \times m$ matrix of zeros). The only change in the algorithm is that the
345 definitions of the M_i and q_i need to be appropriately modified.

346 Another extension that is straightforward is to allow the switching costs C , the Poisson
347 intensities Λ and the costs due to exogenous switching Q to all be functions of S . Again, the
348 solution approach is essentially unchanged except that the specific values of the M_i and q_i need
349 to be adjusted appropriately. If the cost field is a string containing the word 'variable' the
350 solver calls the model function file with the flag `C`. The model function file should return an
351 $n \times m$ matrix containing the cost of switching from regime x to the other regimes (this should
352 contain a vector of zeros in column x). A similar syntax applies to Λ and Q , with the flag
353 variable set to `L` and `Q`, respectively.

354 A third extension handles the situation in which the value function is known at some speci-

355 fixed point or points. In such a case the complementary conditions (17) and (18) associated with
356 the point (S, R) could be replaced by

$$357 \quad \phi(S)\theta_R - V(S, R) \geq 0 \quad (37)$$

358 To implement this feature, the model variable should include a field named `values` con-
359 taining a $d + 2$ column matrix. Columns 1 through d are the values of S , column $d + 1$ contains
360 the regime number R and the last column contains the value of $V(S, R)$. For example suppose
361 S is two-dimensional, with $V([S_1 \ S_2], R)$. Setting

```
362 model.values=[0 0 1 5;0 0 2 5]
```

363 will force $V([0 \ 0], 1) = V([0 \ 0], 2) = 5$. If all values in a specific dimension are involved, a
364 NaN can be used in the associated column. For example, setting

```
365 model.values=[0 nan 0 1;nan 0 1 2]
```

366 will set $V([0 \ S_2], 1) = 0, \forall S_2$ and $V([S_1 \ 0], 2) = 1, \forall S_1$. A value of NaN can also be used for
367 the regime number, so

```
368 model.values=[0 0 nan 5]
```

369 produces the same result as

```
370 model.values=[0 0 1 5;0 0 2 5]
```

371 To force the numerical procedure to produce an approximate solution with known values, the
372 optimality conditions (17) and (18) are replaced by condition (37) at the nodal points closest
373 to each of the (S, R) values.

374 Finally, it may be desirable to allow for resetting of S when a regime switch occurs. For
375 example, one of the states might measure the time spent in the current regime since the last

376 regime change (so $dS = dt$). This state would be reset to 0 every time the regime changes.

377 In general, if switching from regime i to regime j causes S to be reset to S_{ij} , condition (4) is

378 modified to

$$379 \quad V(S, i) \geq V(S_{ij}, j) - C_{ij} \quad (38)$$

380 and condition (18) to

$$381 \quad \phi(S, i)\theta_i - \phi(S_{ij})\theta_j + C_{ij} \geq 0 \quad (39)$$

382 To implement this feature, the model variable should include a field named `reset` con-

383 taining a $2 + d$ column matrix. The first d columns contain the target value of the state after

384 resetting. Column $d + 1$ contains the regime before the switch and columns $d + 2$ is the regime

385 after the switch. For example, if, on switching from regime $i = 2$ to regime $j = 3$, the state is

386 reset to $S_{ij} = [0 \ 1]$, use

387 `model.reset=[0 1 2 3]`

388 If some of the variables are not reset upon switching, set the value of the state for these di-

389 mensions to NaN. For example, if only the first state variable is reset to 0 when switching from

390 regime 1 to 2, use

391 `model.reset=[0 nan 1 2]`

392 **A Worked Example**

393 This section demonstrates the application of the MATLAB procedures to the mine operation
394 example discussed earlier (the demonstration files are included with the solver). The example
395 first solves the model with a two dimensional state space (Q, P) and then solves the same
396 model with the one-dimensional state space $y = QP$.

397 The first requirement is the model function file:

```
398 function out=minemodel2(flag,s,R,mu,sigma,rho,h,k)
399     switch flag
400         case 'f'
401             out=(h*s(:,1).*s(:,2)-k).*(R==2);
402         case 'mu'
403             out=[-h*s(:,1).*(R==2) mu*s(:,2)];
404         case 'sigma'
405             out=[zeros(size(s,1),3) sigma*s(:,2)];
406         case 'rho'
407             out=rho;
408     end
```

409 In addition to the required first three inputs, this function is defined in terms of five of the model
410 parameters μ , σ , ρ , h and k (there are two additional model parameters C_{12} and C_{21}).

411 To solve the model, we also write a MATLAB script file that is called from the MATLAB
412 command line. This file begins by defining the model parameters:

```
413 mu      = 0.01;
414 sigma   = 0.02;
415 rho     = 0.04;
416 h       = 1;
417 k       = 2;
418 C12     = 5;
419 C21     = 2;
```

420 (the specific values are for demonstration purposes only). Next the model structure variable is
421 defined:

```

422     clear model
423     model.func    = 'minemodel2';
424     model.params  = {mu, sigma, rho, h, k};
425     model.cost    = [0 C12; C21 0];

```

426 Then the family of approximating functions is defined:

```

427     fspace=fundefn('lin',[51 51],[0 0],[100 10]);

```

428 Here a piecewise linear function with 51 evenly spaced breakpoints for Q on $[0,100]$ and 51
429 evenly spaced breakpoints for P on $[0,10]$ are used (this family of approximating functions
430 uses finite difference derivatives). The solver is now called:

```

431     [cv, snodes, x]=ossolve(model, fspace);

```

432 Using the output, a plot of the optimal switch boundaries can be computed, as is shown in
433 the solid lines in Figure 1. The lower line represents the points for which it is optimal to switch
434 from active to inactive, the upper line represents the points for which it is optimal to switch
435 from inactive to active. The jaggedness is an inevitable consequence of the discreteness of the
436 nodal values. Although it might be useful to smooth these curves, no attempt has been made to
437 do so here.

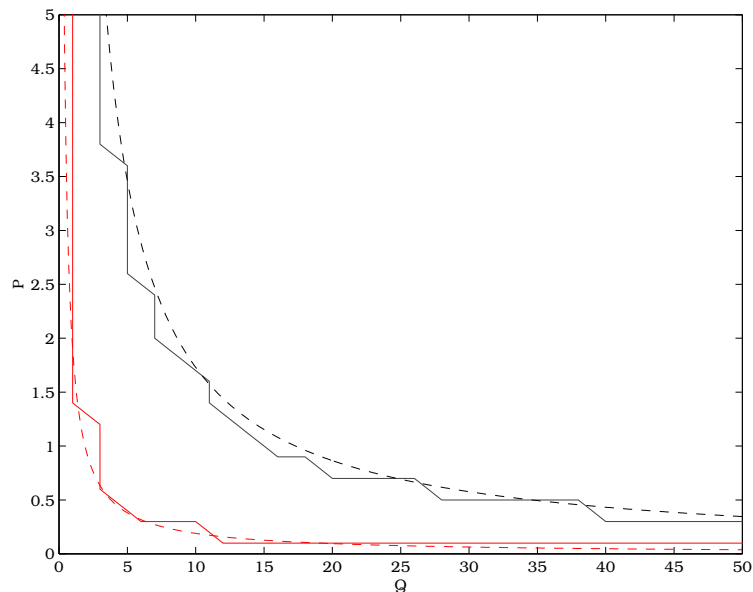
438 The model can also be solved using the one dimensional reformulation. In this case the
439 model function file would be written

```

440     function out=minemodel1(flag,s,R,mu,sigma,rho,h,k)
441         switch flag
442             case 'f'
443                 out=(h*s-k).*(R==2);
444             case 'mu'
445                 out=(mu-h*(R==2))*s;
446             case 'sigma'
447                 out=sigma*s;
448             case 'rho'
449                 out=rho;
450         end

```


Figure 1: Optimal Switch Boundaries for the Mine Example



451 The model variable `func` field would be changed to the name of this procedure

```
452 model.func = 'minemodell';
```

453 Also the `fspace` variable would be altered

```
454 fspace=fundefn('lin', 501, 0, 50);
```

455 This defines a family of piecewise linear functions with 501 breakpoints on [0 50]. When
456 calling the solver, it will be useful now to obtain the fourth output `xindex`:

```
457 [cv, snodes, x, xindex]=ossolve(model, fspace);
```

458 Elements (1,4) and (2,1) of `xindex` contain the approximate locations of the switch points
459 for the inactive and active regimes, respectively. For the parameter values given above, these
460 points are computed to be 17.3 and 1.9. The optimal switchpoints are therefore approximately
461 points satisfying $QP = 17.3$ and $QP = 1.9$, which are shown in the dashed lines in Figure 1.

462 The one-dimensional mine example has an almost explicit solution, which can be used to
463 obtain highly accurate optimal switching boundaries and value functions (see, e.g., Dixit and
464 Pindyck for discussion of this approach). To four decimal places, the switching boundaries are
465 17.2522 and 1.9233. For practical purposes, this is indistinguishable from the one-dimensional
466 numerical solution. Furthermore, the value function approximation was accurate to approxi-
467 mately four significant digits.

468 **Summary**

469 This paper describes a general numerical approach to solving optimal switching problems and
470 documents a MATLAB based implementation of the approach. The basic framework consists
471 of a model for a stochastic process S that characterized by its drift and diffusion functions
472 μ and σ , by a stream of rewards described by the function f , by a discount rate ρ (possibly
473 state contingent) and by a switching cost matrix C . The solution approach requires that these
474 parameters be specified along with a family of approximating functions. The solution algorithm
475 can then set up and solve a set of complementarity conditions that are satisfied at the problem
476 solution.

477 The solution approach described and implemented here has a number of important advan-
478 tages over previously described solution approaches. First, it is generic and hence the code
479 required to solve specific models is mainly limited to specifying the functions and parameter
480 values that define the model, along with code to call the solver and interpret the solver output.
481 Second, it can solve models with general multidimensional diffusion processes. This makes it
482 easy to solve models without the restriction to one-dimensional geometric Brownian motion

483 found in much of the existing literature. Third, unlike the generic one-dimensional solver de-
484 scribed in Miranda and Fackler (chap. 11), the solution approach used here does not require
485 that the analyst guess at the qualitative nature of the optimal solution. In particular, it is not
486 necessary to know to which regime it is optimal to switch at the boundaries of the no-switch
487 regions.

References

- 488
489 Brekke, Kjell Arne and Bernt Oksendal. “Optimal Switching in an Economic Activity Under
490 Uncertainty.” *SIAM Journal on Control and Optimization* 32(1994):1021–1036.
- 491 Brennan, Michael J. and Eduardo S. Schwartz. “Evaluating Natural Resource Investments.”
492 *Journal of Business* 58(1985):135–157.
- 493 Brennan, Michael J. and Lenos Trigeorgis, editors. *Project Flexibility, Agency and Competi-*
494 *tion*. Oxford: Oxford University Press, 2000.
- 495 Cottle, R.W. and G.B. Dantzig. “A Generalization of the Linear Complementarity Problem.”
496 *Journal of Combinatorial Theory* 8(1970):79–90.
- 497 Dixit, Avinash K. and Robert S. Pindyck. *Investment Under Uncertainty*. Princeton, NJ:
498 Princeton University Press, 1994.
- 499 Dumas, Bernard. “Super Contact and Related Optimality Conditions.” *Journal of Economic*
500 *Dynamics and Control* 15(1991):675–685.
- 501 Gowda, M. Seetharama and Roman Sznajder. “The Generalized Order Linear Complementar-
- 502 ity Problem.” *SIAM Journal of Matrix Analysis and Applications* 15(1994):779–795.
- 503 Judd, Kenneth L. *Numerical Methods in Economics*. Cambridge, MA: MIT Press, 1998.
- 504 Lemke, C.E. “Bimatrix Equilibrium Points and Mathematical Programming.” *Management*
505 *Science* 11(1965):681–689.
- 506 McDonald, Robert L. and Daniel R. Siegel. “Investment and the Valuation of Firms When
507 There is an Option to Shut Down.” *International Economic Review* 26(2)(1985):331–349.
- 508 McDonald, Robert L. and Daniel R. Siegel. “The Value of Waiting to Invest.” *Quarterly*
509 *Journal of Economics* 101(1986):707–727.
- 510 Miranda, Mario J. and Paul L. Fackler. *Applied Computational Economics and Finance*. Cam-
511 bridge MA: MIT Press, 2002.
- 512 Qi, Hou-Duo and Li-Zhi Liao. “A Smoothing Newton Method for Extended Vertical Linear
513 Complementarity Problems.” *Journal of Matrix Analysis and Applications* 21(1999):45–66.
- 514 Schwartz, Eduardo S. and Lenos Trigeorgis, editors. *Real Options and Investment Under Un-*
515 *certainty*. Cambridge, MA: MIT Press, 2001.
- 516 Sun, Min. “Monotonicity of Mangasarian’s Iterative Algorithm for Generalized Linear Comple-
- 517 mentarity Problems.” *Journal of Mathematical Analysis and Applications* 144(1989):474–
518 485.