Applied Stochastic Processes and Control for Jump-Diffusions: Modeling, Analysis and Computation

Floyd B. Hanson
University of Illinois
Chicago, Illinois, USA

Chapter 9 Computational Stochastic Control Methods

May 24, 2006

Chapter 9

Computational Stochastic Control Methods

God does not care about our mathematical difficulties.
He integrates empirically.
—Albert Einstein (1879-1955).

An idea which can be used once is a trick.
If it can be used more than once it becomes a method.
—George Polya and Gabor Szego.

"That's when I realized that research was my true calling,
not software," he says. Developing software so other people
could answer the big questions wasn't for him. He wanted to
get back to answering them himself.
—Ajay Royyuruat , IBM Genographer, *Dream Jobs, IEEE*
Spectrum, vol. 43, no. 2, February 2006, pp. 40-41.

The use of stochastic models, on the other hand, can result
in gigantic increases in the complexity of data volume, stor-
age, manipulation, and retrieval requirements.
—Simulation-Based Engineering Science, Report of the Na-
tional Science Foundation Blue Ribbon Panel on Simulation-
Based Engineering Science, *J. T. Oden, Chair, February*
2006, 85 pages.

Stochastic dynamic programming is not easy since the **PDE of stochastic dynamic programming** or the **Hamilton-Jacobi equation** given in (7.15-7.18) of Chapter 7 is not a standard PDE (partial differential equation). In fact, it is a functional PDE with just diffusion owing to the presence of a maximum with respect to the control. Also, for the more general jump-diffusion, the additional jump integrals make the PDE of stochastic dynamic programming a functional

333

partial integral differential equation or functional PIDE (partial integral differential equation). The analytic complexity of this functional PIDE means that for the usual finite difference or finite element methods, numerical convergence conditions are unknown or not easily ascertainable.

This chapter discusses PDE-oriented finite difference methods developed by the author and coworkers [106, 107, 108, 272, 110] for solving the PDE of **stochastic dynamic programming (SDP)** (7.15-7.18), with special emphasis on techniques and convergence conditions. The numerical foundations and complexity of computational stochastic control are discussed in [110].

An alternative method relies on using Markov chain probabilities to construct convergent finite difference approximations that are rigorously convergent in the weak sense and is called the **Markov chain approximation (MCA)** developed by Kushner and coworkers [170, 171, 175].

Some methods use a canonical model formulation whose solution algorithm results in significant reduction in the dimensional complexity, e.g., the **linear-quadratic (LQ) model** for the optimal control of jump-diffusions (LQJD or LQGP) [269] and the **constant relative risk aversion (CRRA) utility model** for the optimal portfolios in finance [122, 123, 129, 286]. In addition, special integration methods for jump integrals and a least squares approximation for forming simpler LQJD problems are also discussed [272]. The LQJD canonical model dimensional reduction algorithm is covered in Section 7.4 on page 294 in Chapter 7 while the deterministic LQ and variants are covered in Section 6.3 on page 262 in Chapter 6.

Another canonical model dimensional reduction algorithm is treated in Sections 11.4 on page 436 and 11.5 on page 447 in Chapter 11 for two different optimal portfolio and consumption applications.

For a more historical introduction to computational methods in control, see Larson [178], Polak [223] and Dyer and McReynolds [76].

9.1 Finite Difference PDE Methods of SDP

A decade ago, the author contributed an invited chapter on *Computational Stochastic Dynamic Programming* [108] in a *Control and Dynamic Systems* volume discussing the use of finite difference methods of solution. This section is based on his past experience with large scale stochastic control applications using many of the largest vector and parallel computers available academically from national centers such as Argonne National Laboratory, Los Alamos National Laboratory, National Center for Supercomputing Applications, San Diego Supercomputing Center and the Pittsburgh Supercomputing Center. An updated version of the techniques involved is given but simplified to one state dimension initially for convenience.

Consider the jump-diffusion SDE for state $X(t)$ and control $U(t)$,

$$dX(t) \stackrel{\text{sym}}{=} f(X(t),U(t),t)dt + g(X(t),U(t),t)dW(t) + \sum_{k=1}^{dP(t)} h(X(T_k^-),U(T_k^-),T_k^-,Q_k),$$

(9.1)

where $dP(t)$ and $dW(t)$ are the stochastic differentials of the jump-diffusion process

including the compound Poisson mark Q_k and pre-jump-time T_k^- realizations with jump-rate $\lambda(t; x, u, t)$. The SDE coefficients, $(f(x, u, t), g(x, u, t), h(x, u, t, q))$, are assumed to be bounded or at least integrable in their arguments, so as not to over-restrict the problem. Let the objective be the minimum of the expected cumulative running costs $C(x, u, t)$ and terminal cost $S(x_f, t_f)$,

$$v^*(x, t) \equiv \min_{U[t, t_f)} \left[\mathop{\mathrm{E}}_{(dW, dP)[t, t_f)} \left[\int_t^{t_f} C(X(s), U(s), s) ds + S(X(t_f), t_f) \middle| X(t) = x, U(t) = u \right] \right], \tag{9.2}$$

for $t_0 \leq t < t_f$.

The application of **Bellman's Principle of Optimality** and the stochastic chain rule along with the infinitesimal moments $\mathrm{E}[dW(t)] = 0$, $\mathrm{Var}[dW(t)] = dt$ and $\mathrm{E}[dP(t; X(t), U(t), t) | X(t) = x, U(t) = u] = \lambda(t; X(t), U(t)) dt$ leads to the stochastic dynamic programming PIDE using only order dt terms,

$$\begin{aligned}
0 = {} & v_t^*(x, t) + \min_u \left[\mathcal{H}(x, u, t) \right] \\
\equiv {} & v_t^*(x, t) + \min_u \left[C(x, u, t) + f(x, u, t) v_x^*(x, t) + \tfrac{1}{2} g^2(x, u, t) v_{xx}^*(x, t) \right. \\
& \left. + \lambda(t; x, u, t) \int_{\mathcal{Q}} \left(v^*(x + h(x, u, t, q), t) - v^*(x, t) \right) \phi_Q(q; x, u, t) \right] \\
= {} & v_t^*(x, t) + \mathcal{H}^*(x, t) .
\end{aligned} \tag{9.3}$$

If the regular or unconstrained optimal control exists and is unique, then

$$u^{(\mathrm{reg})}(x, t) = \mathop{\mathrm{argmin}}_u \left[\mathcal{H}(x, u, t) \right] , \tag{9.4}$$

but, in general, the optimal control, $u^*(x, t)$, is subject to any control constraints. The final condition from the minimal conditional expected cost objective (9.2) is

$$v^*(x, t) = S(x, t_f). \tag{9.5}$$

However, the boundary conditions in general are model and domain dependent.

9.1.1 Linear Control Dynamics and Quadratic Control Costs

In order, to keep the focus on basic computations, it will be assumed that the drift of the state dynamics is linear in the control and that the running costs are quadratic in the control, i.e, the **LQJD problem in control only (LQJD/U)** discussed in Subsection 7.4.1. These assumptions are more general than the LQJD problem, but are sufficient to determine optimal control clearly in terms of (x, t). Hence, let

$$\begin{aligned}
f(x, u, t) &= f_0(x, t) + f_1(x, t) u, \\
g(x, u, t) &= g_0(x, t), \quad h(x, u, t, q) = h_0(x, t, q), \\
\lambda(x, u, t) &= \lambda(x, t), \quad \phi_Q(q; x, u, t) = \phi_Q(q), \\
C(x, u, t) &= c_0(x, t) + c_1(x, t) u + c_2(x, t) u^2, \\
\mathcal{H}(x, u, t) &= \mathcal{H}_0(x, t) + \mathcal{H}_1(x, t) u + \tfrac{1}{2} \mathcal{H}_2(x, t) u^2 .
\end{aligned} \tag{9.6}$$

Thus, the PDE of stochastic dynamic programming in Hamilton-Jacobi form using (7.20) with the current assumptions,

$$
\begin{aligned}
0 = \ & v_t^*(x,t) + \mathcal{H}^*(x,t) \\
= \ & v_t^*(x,t) + C_0(x,t) + C_1(x,t)u^* + \tfrac{1}{2}C_2(x,t)(u^*)^2 \\
& + (f_0(x,t) + f_1(x,t)u^*)v_x^*(x,t) + \tfrac{1}{2}g_0^2(x,t)v_{xx}^*(x,t) \\
& + \lambda(t)\int_{\mathcal{Q}} \left(v^*(x + h_0(x,t,q),t) - v^*(x,t)\right)\phi_Q(q)dq \ ,
\end{aligned}
\tag{9.7}
$$

and the regular control is from (7.32) after simplifications for the current one state dimension form,

$$
u^{(\mathrm{reg})}(x,t) = -\left(c_1(x,t) + f_1(x,t)v_x^*(x,t)\right)/c_2(x,t) \ ,
\tag{9.8}
$$

provided $c_2(x,t) > 0$, i.e., positive definite, for a minimum. Since real problems have contraints, let $U^{(\mathrm{min})} \le u(x,t) \le U^{(\mathrm{max})}$. Then the optimal control law can be written

$$
\begin{aligned}
u^*(x,t) = \ & \min(U^{(\mathrm{max})}, \max(U^{(\mathrm{min})}, u^{(\mathrm{reg})}(x,t))) \\[2mm]
= \ & \left\{
\begin{array}{ll}
U^{(\mathrm{min})}, & u^{(\mathrm{reg})}(x,t) \le U^{(\mathrm{min})} \\
u^{(\mathrm{reg})}(x,t), & U^{(\mathrm{min})} \le u^{(\mathrm{reg})}(x,t) \le U^{(\mathrm{max})} \\
U^{(\mathrm{max})}, & U^{(\mathrm{max})} \le u^{(\mathrm{reg})}(x,t)
\end{array}
\right\} \ .
\end{aligned}
\tag{9.9}
$$

For multidimensional state space problems see the stochastic dynamic programming Chapter 7 here or Hanson's computational stochastic dynamic programming chapter in [108].

9.1.2   Crank-Nicolson, Extrapolation-Predictor-Corrector Finite Difference Algorithm for SDP

The numerical algorithm used here is basically a modification of the work of Douglas and Dupont [72, 73] on nonlinear parabolic equations modified for stochastic dynamic programming and the PIDE for jump-diffusions.

First the problem is discretized in backward time since stochastic dynamic programming is a backward problem but the state space is discretized in a regular grid, with N_t nodes in t on $[t_0, t_f]$ and N_x nodes in x on $[x_0, x_{\max}]$,

$$
\begin{aligned}
t \to T_k &= t_f - (k-1)\cdot\Delta t, \text{ for } k = 1\!:\!N_t, \ \Delta t = (t_f - t_0)/(N_t - 1) \ , \\
x \to X_j &= x_0 + (j-1)\cdot\Delta X, \text{ for } j = 1\!:\!N_x, \ \Delta X = (x_{\max} - x_0)/(N_x - 1) \ .
\end{aligned}
\tag{9.10}
$$

This grid leads to a corresponding discretization of the dependent variables that follow using a **second order central finite difference (CFD)** for the time derivative, evaluating at the mid-time point, and second order CFDs for the state derivatives when $j = 1\!:\!N_x$ for each $k = 1\!:\!N_t$ corresponding to the backward time count

with $T_1 = t_f$:

$$v^*(X_j, T_k) \rightarrow V_{j,k} ,$$
$$v_t^*(X_j, T_{k+0.5}) \rightarrow (V_{j,k+1} - V_{j,k})/(-\Delta t) ,$$
$$v_x^*(X_j, T_k) \rightarrow DV_{j,k} = 0.5(V_{j+1,k} - V_{j-1,k})/\Delta X ,$$
$$v_{xx}^*(X_j, T_k) \rightarrow DDV_{j,k} = (V_{j+1,k} - 2V_{j,k} + V_{j-1,k})/(\Delta X)^2 , \quad (9.11)$$
$$u^{(\text{reg})}(X_j, T_k) \rightarrow UR_{j,k} = -(C_{1,j,k} + F_{1,j,k}DV_{j,k})/C_{2,j,k} ,$$
$$u^*(X_j, T_k) \rightarrow US_{j,k} = \min(\text{UMAX}, \max(\text{UMIN}, UR_{j,k})) ,$$
$$v^*(X_j + h_0(X_j, T_k, q), T_k) \rightarrow VH_{j,k}(q),$$

where $F_{i,j,k} = f_i(X_j, T_k)$ for $i = 0 : 1$, $C_{i,j,k} = c_i(X_j, T_k)$ for $i = 0 : 2$, UMIN $= U^{(\min)}$ and UMAX $= U^{(\max)}$.

The **Crank-Nicolson Implicit (CNI)** method provides central differencing in state and time, so is second order accurate in both independent variables, i.e., $O^2(\Delta X) + O^2(\Delta t)$, and the implicitness provides stability over all positive steps in time, Δt. However, for general problems, such as those that are multi-dimensional or are nonlinear, the implicit and tridiagonal properties are no longer valid, unless CNI can be extended by **alternating directions implicit (ADI)** through known splittings of the spatial operators. However, for nonlinear problems, recalling from Chapter 7 that the PDE of stochastic dynamic programming is nonlinear, ADI is not useful and predictor-corrector methods can be used to preserve the second order accuracy in several dimensions and for nonlinear problems. For these more general applications, the basic structure of the CNI method upon dissection consists of a midpoint integral approximation and an averaging to convert the time-midpoint to integral grid point values. Thus, symbolically using the PDE of stochastic dynamic programming in Hamilton-Jacobi form, $0 = v_t^*(x, t) + \mathcal{H}^*(x, t)$, using (9.7), the **midpoint rule approximation** is then

$$V_{j,k+1} - V_{j,k} = \int_{T_k}^{T_k - \Delta t} v_t^*(X_j, t)dt = -\int_{T_k}^{T_k - \Delta t} \mathcal{H}(X_j, t)dt$$
$$\simeq +\Delta t \cdot \mathcal{H}(X_j, T_{k+0.5}) = +\Delta t \cdot \mathcal{H}_{j,k+0.5} , \quad (9.12)$$

which is finally followed by a **second order accuracy preserving averaging step**,

$$V_{j,k+1} \simeq V_{j,k} + 0.5 \cdot \Delta t \cdot (\mathcal{H}_{j,k} + \mathcal{H}_{j,k+1}) , \quad (9.13)$$

where the midpoint (mid-time) value of the objective has been replaced by targeted values at given time nodes. While this last step may look like a linear assumption, in most cases this can be extended by quasi-linearization, e.g., the average for a power can be approximated by $(V_{j,k+0.5})^{n+1} \simeq 0.5(V_{j,k})^n(V_{j,k} + V_{j,k+1})$ in the zeroth correction with further refinement in subsequent corrections, always keeping the newest update of $V_{j,k+1}$ as a linear term. The reader can show that under second order differentiability the averaging step is second order accurate in time $(O^2(\Delta t))$ at the midpoint, it being well-known that the midpoint rule used here is second order accurate in time. It is the midpoint rule evaluation that makes the seemingly

first order approximation for $v_t^*(x,t)$ in (9.11) accurate to $O^2(\Delta t)$ rather than to $O(\Delta t)$.

Integration and Interpolation for Jump Integrals

Another modification is needed for handling the jump integrals. One procedure is the use of **Gauss-statistics rules** introduced by Westman and Hanson in [272] as a generalization of the Gaussian quadrature rules, but customized for the given mark density $\phi_Q(q)$ in the application. These rules use N_q points Q_i and N_q weights w_i and have a polynomial precision of degree $n_q = N_q - 1$. The weights and nodes satisfy the $2 \cdot N_q$ nonlinear equations,

$$\sum_{i=1}^{N_q} w_i \cdot Q_i^j = \mathrm{E}_Q[Q^j] = \int_{\mathcal{Q}} q^j \phi_Q(q) dq \,, \tag{9.14}$$

for $j = 0:2N_q-1$. This leads to the Gauss-statistics approximation for the jump integral:

$$\begin{aligned} \mathrm{IVH}_{j,k} &\equiv \int_{\mathcal{Q}} \mathrm{VH}_{j,k}(q)\phi_Q(q)dq \simeq \sum_{i=1}^{N_q} w_i \mathrm{VH}_{j,k}(Q_i) \\ &= \sum_{i=1}^{N_q} w_i v^*(X_j + h_0(X_j, T_k, Q_i), T_k) \,. \end{aligned} \tag{9.15}$$

In general, the $\mathrm{VH}_{j,k}(Q_i)$ will be implicit values that are not necessarily at specified state nodes $j' = 1:N_t$ in $V_{j',k}$. Just as in Crank-Nicolson averaging, $O^2(\Delta X)$ interpolation is needed relative to the nearest neighbor state nodes. Let the ith state argument be

$$X_j + h_0(X_j, T_k, Q_i) = X_{j+\ell_i} + \epsilon_i \Delta X \,,$$

where the floor integer is

$$\ell_i = \ell_{i,j,k} = \lfloor h_0(X_j, T_k, Q_i)/\Delta X \rfloor$$

and fraction

$$\epsilon_i = \epsilon_{i,j,k} = h_0(X_j, T_k, Q_i)/\Delta X - \ell_i.$$

Thus, the $O^2(\Delta X)$ interpolation is

$$\mathrm{VH}_{j,k}(Q_i) \simeq (1-\epsilon_i) \cdot V_{j+\ell_i,k} + \epsilon_i \cdot V_{j+\ell_i+1,k} \,, \tag{9.16}$$

assuming the jumps are not out of range of the state space or are handled by proper boundary conditions. Thus,

$$\mathrm{IVH}_{j,k} \simeq \sum_{i=1}^{N_q} w_i \left((1-\epsilon_i) \cdot V_{j+\ell_i,k} + \epsilon_i \cdot V_{j+\ell_i+1,k}\right) \,. \tag{9.17}$$

Example 9.1. *Gauss-Statistics Quadrature for Log-Uniform Jump-Amplitudes:*

For example, in the case that $\phi_Q(q)$ is the density of the uniform distribution on $[a,b]$, then

$$\text{for } N_q = 1, \ n_q = 1, \ w_1 = 1, \ Q_1 = 0.5(a+b) \ ;$$

or

$$\text{for } N_q = 2, \ n_q = 3, \ w_1 = 0.5, \ w_2 = 0.5,$$
$$Q_1 = 0.5(a+b) - 0.5(b-a)/\sqrt{3}, \ Q_2 = 0.5(a+b) + 0.5(b-a)/\sqrt{3} \ .$$

For higher precision on finite mark domains $[a,b]$, piecewise applications of these rules can be made on subdivisions $[q_i, q_{i+1}]$ where $q_i = a + (i-1)\Delta q$ for $i = 1 : M_q$ nodes with $\Delta q = (b-a)/(M_q - 1)$. See Westman and Hanson [272] for more information.

In the case that there is a special q-dependence of the jump-amplitude coefficient $h_0(x,t,q)$ for which the moments can be easily or conveniently calculated, then it may be possible to use just the interpolation of $\text{VH}_{j,k}(q)$ without Gauss-statistics quadrature in q.

Example 9.2. *Geometric Jump-Diffusion with Log-Uniform Jump-Amplitudes Jump-Integral Approximation:*

In the financial geometric jump-diffusion with log-uniform jump-amplitude distribution (11.119), the distribution of q is uniform with respect to the log-return $\ln(x)$, but in the original return values the jump in the return is $h(x,t,q) = x \cdot (e^q - 1)$ by Itô's chain rule. For the financial market q is very small, then so is $e^q - 1$, while a is small and negative with b small and positive. Provided $|\epsilon| \leq 1$ where $\epsilon = X_j(e^q - 1)/\Delta X$, then the appropriate piece-wise linear interpolation using the explicit node set $\{V_{j-1,k}, V_{j,k}, V_{j+1,k}\}$ is

$$\text{VH}_{j,k}(q) \simeq \left\{ \begin{array}{ll} (1-\epsilon)V_{j,k} + \epsilon V_{j+1,k}, & q \geq 0, \epsilon \geq 0 \\ -\epsilon V_{j-1,k} + (1+\epsilon)V_{j,k}, & q \leq 0, \epsilon \leq 0 \end{array} \right\} \ . \tag{9.18}$$

Since the factor $(e^q - 1)$ is now explicit, it can be integrated directly without Gaussian quadrature to produce,

$$\begin{aligned} \int_a^b \text{VH}_{j,k}(q)\phi_Q(q)dq &\simeq V_{j,k} + \tfrac{X_j}{\Delta X}(V_{j,k} - V_{j-1,k})\tfrac{1+a-e^a}{b-a} \\ &+ \tfrac{X_j}{\Delta X}(V_{j+1,k} - V_{j-,k})\tfrac{e^b-1-b}{b-a} \ . \end{aligned} \tag{9.19}$$

Extrapolation, Prediction and Correction

Summarizing the above CNI discretizations, the PIDE of stochastic dynamic programming of (9.7) can be put in the preliminary form

$$\begin{aligned} V_{j,k+1} &= V_{j,k} + \Delta t \cdot \mathcal{H}_{j,k+0.5} \\ &= V_{j,k} + \Delta t \left(C_{j,k+0.5} + F_{j,k+0.5} \cdot \text{DV}_{j,k+0.5} \right. \\ &\quad \left. +0.5 \cdot G_{0,j,k+0.5}^2 \cdot \text{DDV}_{j,k+0.5} + \Lambda_k \cdot (\text{IVH}_{j,k+0.5} - V_{j,k+0.5}) \right) \ , \end{aligned} \tag{9.20}$$

where $C_{j,k} = C_{0,j,k} + C_{1,j,k}\mathrm{US}_{j,k} + 0.5 \cdot C_{2,j,k} \cdot \mathrm{US}_{j,k}^2$, $F_{j,k} = F_{0,j,k} + F_{1,j,k}\mathrm{US}_{j,k}$, $G_{0,j,k} = g_0(X_j, T_k)$, $\Lambda_k = \lambda(T_k)$, $\mathrm{US}_{j,k} = \min(\mathrm{UMAX}, \max(\mathrm{UMIN}, \mathrm{UR}_{j,k}))$ and $\mathrm{UR}_{j,k} = -\left(C_{1,j,k} + F_{1,j,k} \cdot \mathrm{DV}_{j,k}\right)/C_{2,j,k}$, using (9.11).

Once there are two prior values $V_{j,k-1}$ and $V_{j,k}$ which happens when $k \geq 2$, linear extrapolation (ex) can be used to accelerate the SDP corrections. The first step from the final condition at $k = 1$ to $k = 2$ takes the most corrections since no trend is available, only $V_{j,1}$. Otherwise the extrapolation (ex) step for the time-midpoint is used for $k \geq 2$ rather than the initial prediction at $k = 1$,

$$V_{j,k+0.5}^{(ex)} = \left\{ \begin{array}{ll} V_{j,k}, & k = 1 \\ 0.5(3V_{j,k} - V_{j,k-1}), & k \geq 2 \end{array} \right\} , \tag{9.21}$$

which is used to update the derivative $\mathrm{DV}_{j,k+0.5}$, 2nd derivative $\mathrm{DDV}_{j,k+0.5}$, regular control $\mathrm{UR}_{j,k+0.5}$, optimal control $\mathrm{UR}_{j,k+0.5}$ and jump functions $\mathrm{VH}_{j,k+0.5}(q)$ in the list (9.11) for the pseudo-Hamiltonian $\Delta t \cdot \mathcal{H}_{j,k+0.5}^{(ex)}$ in (9.12, 9.20) using quasi-linearization for nonlinear terms. The resulting update of the value is called the predictor or 1st correction step $(c, 1)$,

$$V_{j,k+1}^{(c,1)} = V_{j,k} + \Delta t \cdot \mathcal{H}_{j,k+0.5}^{(ex)} , \tag{9.22}$$

for all j, as long as $k \geq 2$. Otherwise the predicted step uses the current value or $V_{j,k+1}^{(c,1)} = V_{j,k} + \Delta t \cdot \mathcal{H}_{j,k}$ using (9.20). The evaluation step uses the updated average,

$$V_{j,k+0.5}^{(c,1)} = 0.5(V_{j,k+1}^{(c,1)} + V_{j,k}) , \tag{9.23}$$

which is used to update all the needed values in (9.11) and finally in all the next correction $(c, 2)$,

$$V_{j,k+1}^{(c,2)} = V_{j,k} + \Delta t \cdot \mathcal{H}_{j,k+0.5}^{(c,1)} . \tag{9.24}$$

The γth correction loop given $V_{j,k+1}^{(c,\gamma)}$ will contain

$$V_{j,k+0.5}^{(c,\gamma)} = 0.5(V_{j,k+1}^{(c,\gamma)} + V_{j,k}) , \tag{9.25}$$

plus the corresponding evaluations of $\mathrm{DV}_{j,k+0.5}^{(c,\gamma)}$, $\mathrm{DDV}_{j,k+0.5}^{(c,\gamma)}$, $\mathrm{UR}_{j,k+0.5}^{(c,\gamma)}$, $\mathrm{UR}_{j,k+0.5}^{(c,\gamma)}$, $\mathrm{VH}_{j,k+0.5}^{(c,\gamma)}(q)$ including integration, and $\mathcal{H}_{j,k+0.5}^{(c,\gamma)}$. Then

$$V_{j,k+1}^{(c,\gamma+1)} = V_{j,k} + \Delta t \cdot \mathcal{H}_{j,k+0.5}^{(c,\gamma)} . \tag{9.26}$$

The corrections continue until the stopping criterion is reached, for instance, the relative criteria given tolerance tol_v,

$$\left\| V_{j,k+1}^{(c,\gamma+1)} - V_{j,k+1}^{(c,\gamma)} \right\|_1 < \mathrm{tol}_v \left\| V_{j,k+1}^{(c,\gamma)} \right\|_1 , \tag{9.27}$$

for each k, continuing corrections if not satisfied, otherwise stopping the corrections setting $\gamma_{\max} = \gamma + 1$ and setting the final $(k+1)$st value at

$$V_{j,k+1} = V_{j,k+1}^{(c,\gamma_{\max})} . \tag{9.28}$$

In (9.27), $\| * \|_1$ denotes the one-norm with respect to the state index j for current time index k, but other norms could be used with the one-norm being less computationally costly.

Stability criteria is another matter due to the complexity of the PIDE of SDP in terms of multi-state systems, jump integrals, nonlinear terms and optimization terms. A rough criterion focuses on the diffusion term $G_{0,j,k+0.5}^2 \mathrm{DDV}_{j,k+0.5}$ in (9.20), which can be expanded by substituting the CFD form (9.11) for $\mathrm{DV}_{j,k+0.5}$ and $\mathrm{DDV}_{j,k+0.5}$ into (9.20) and produces

$$
\begin{aligned}
V_{j,k+1} = \quad & \left(1 - \frac{\Delta t}{\Delta X^2}G_{0,j,k+0.5}^2\right)V_{j,k+0.5} \\
& +0.5\frac{\Delta t}{\Delta X^2}\left(G_{0,j,k+0.5}^2 + F_{j,k+0.5}\Delta X\right)V_{j+1,k+0.5} \\
& +0.5\frac{\Delta t}{\Delta X^2}\left(G_{0,j,k+0.5}^2 - F_{j,k+0.5}\Delta X\right)V_{j-1,k+0.5} \\
& +\Delta t C_{j,k+0.5} + \Lambda_k \Delta t \cdot (\mathrm{IVH}_{j,k+0.5} - V_{j,k+0.5}) ,
\end{aligned}
\tag{9.29}
$$

where $C_{j,k} = C_{0,j,k} + C_{1,j,k}\mathrm{US}_{j,k} + 0.5 \cdot C_{2,j,k} \cdot \mathrm{US}_{j,k}^2$ and $F_{j,k} = F_{0,j,k} + F_{1,j,k}\mathrm{US}_{j,k}$.

Following Kushner and Dupuis [175] and ignoring the jump and cost terms, the positivity of the diffusion with drift terms leads to a **parabolic mesh ratio**

$$
\max_{j,k}\left(G_{0,j,k+0.5}^2\right)\frac{\Delta t}{(\Delta X)^2} < 1,
\tag{9.30}
$$

or so, but certainly should be less than one. This assumes that the PIDE is **diffusion-dominated** and accounts for the drift as well as other terms in (9.3). The discrete HJB equation is said to be **diffusion-dominated**, modified for current form from a relation in [175], if

$$
\min_{j,k}\left(G_{0,j,k}^2 - |F_{j,k}|\,\Delta X\right) \ge 0,
\tag{9.31}
$$

where $F_{j,k} = F_{0,j,k} + F_{1,j,k}\mathrm{US}_{j,k}$, so that the coefficients of the non-diagonal terms, $V_{j+1,k+0.5}$ and $V_{j-1,k+0.5}$ are also positive. Otherwise the discrete problem is either mixed domination or **drift-dominated**, ignoring the jump cost terms. The technique is to decrease Δt and/or increase ΔX if spurious oscillations appear. Not that the diffusion-dominated condition (9.31) is satisfied for sufficiently small state stepsize ΔX as long as the diffusion coefficient $G_{0,j,k+0.5}^2$ is not also sufficiently small. For more information on linear and multi-state models, see Hanson [108], [212] and [111] or see Kushner and Dupuis [175].

The central finite differences for state derivatives work quite well in the diffusion-dominated regime, but are not useful for specified derivative boundary conditions, such as the convection boundary condition and the no-flux or reflecting boundary condition (8.43), e.g., $v_x^*(x_0,t) = 0$ on the left boundary or $v_x^*(x_{\max},t) = 0$ on the right boundary, respectively, assuming the diffusion coefficient $g_0^2(x,t)/2 > 0$ for a well defined flux and nonsingular boundary condition. Using second order forward and backward finite differences, respectively, to maintain consistency in numerical accuracy with the central differences in the interior of $[x_0, x_{\max}]$, the derivatives at

the boundaries are

$$v_x^*(x_0, T_k) \simeq \text{DV}_{1,k} = -0.5(V_{3,k} - 4V_{2,k} + 3V_{1,k})/\Delta x,$$

$$v_x^*(x_{\max}, T_k) \simeq \text{DV}_{N_x,k} = +0.5(V_{N_x-2,k} - 4V_{N_x-1,k} + 3V_{N_x,k})/\Delta x. \tag{9.32}$$

Now, these signs of these terms are not a problem for stability since these conditions are used as eliminants for $V_{1,k}$ for left boundary values and $V_{N_x,k}$ for right boundary values rather than a replacements for the discrete HJB equations (9.29). An alternate derivative boundary condition implementation is to add artificial boundary to the domain, but this author has found better performance using only the domain with the derivative boundary values like (9.32).

For **finite element** versions see Chung, Hanson and Xu [54] or Hanson[108]. Although not on SDP, the work of Chakrabarty and Hanson [49] uses the CNI-predictor-corrector methods discussed here with finite elements for a large scale distributed parameter or PDE-driven system. Finite element methods are better for presenting multidimensional systems and systems on irregular domains.

9.1.3   Upwinding Finite Differences If Not Diffusion-Dominated

When the diffusion-dominated condition (9.31) is no longer valid then the drift term becomes important or the system (9.3) becomes drift dominant and the coefficients of the non-diagonal terms, $V_{j+1,k}$ and $V_{j-1,k}$ are no longer guaranteed to be positive. In this case the system takes upon more hyperbolic PDE characteristics since the drift terms are of hyperbolic as are first order PDEs. In the case of drift-dominance or near-drift-dominance, following Kushner [175] and others, the finite difference to the first state partial of the optimal value function $v_x^*(X_j, T_k)$ in (9.11) should be changed from second-order CFD to first-order **upwinded finite differences (UFD)** which uses forward or backward finite differences (FFDs or BFDs) to coincide with the sign of the drift coefficient, respectively, i.e.,

$$\text{DV}_{j,k} = \left\{ \begin{array}{ll} (V_{j+1,k} - V_{j,k})/\Delta x, & F_{j,k} \geq 0 \\ (V_{j,k} - V_{j-1,k})/\Delta x, & F_{j,k} < 0 \end{array} \right\}, \tag{9.33}$$

where again $F_{j,k} = F_{0,j,k} + F_{1,j,k}\text{US}_{j,k}$. Thus, upwind is in the direction of the drift. However, upwinding requires a sacrifice of numerical accuracy consistency, going from $\text{O}(\Delta X^2)$ CFD to $\text{O}(\Delta X)$ UFD for the first state partial, in favor of more stable numerical calculations. Substituting the UFD form (9.33) for $\text{DV}_{j,k}$ in (9.20) produces

$$\begin{aligned} V_{j,k+1} = \quad & \left(1 - \frac{\Delta t}{\Delta X^2}\left(G_{0,j,k+0.5}^2 + 0.5|F_{j,k+0.5}|\Delta X\right)\right)V_{j,k+0.5} \\ & +0.5\frac{\Delta t}{\Delta X^2}\left(G_{0,j,k+0.5}^2 + [F_{j,k+0.5}]_+\Delta X\right)V_{j+1,k+0.5} \\ & +0.5\frac{\Delta t}{\Delta X^2}\left(G_{0,j,k+0.5}^2 + [F_{j,k+0.5}]_-\Delta X\right)V_{j-1,k+0.5} \\ & +\Delta t C_{j,k+0.5} + \Lambda_k \Delta t \cdot (\text{IVH}_{j,k+0.5} - V_{j,k+0.5})\,, \end{aligned} \tag{9.34}$$

where $[f]_{\pm} \equiv \max[\pm f] \geq 0$, such that $[f]_+ + [f]_- = |f|$ and $[f]_+ - [f]_- = f$. Hence, for the diffusion terms, all coefficients are positive provided the **drift-adjusted parabolic mesh ratio** condition is satisfied,

$$\max_{j,k} \left(G^2_{0,j,k+0.5} + 0.5|F_{j,k+0.5}| \right) \frac{\Delta t}{(\Delta X)^2} < 1, \qquad (9.35)$$

without the extra diffusion-dominated condition in (9.31) being needed.

9.1.4   Multi-state Systems and Bellman's Curse of Dimensionality

Generalization to multi-dimensional state spaces can lead to very large scale computational problems, since the size of the computational problem grows with the number of dimensions multiplied the number of nodes per dimension.

Starting with a version of the PDE of SDP in (7.20) modified for the LQJD/U form in (7.22-7.26) and no diffusion process correlations ($R' = I_{nw \times n_w}$),

$$\begin{aligned}
0 = {}& v^*_t(\mathbf{x}, t) + C_0(\mathbf{x}, t) + \mathbf{C}_1^\top(\mathbf{x}, t)\mathbf{u}^* + \frac{1}{2}(\mathbf{u}^*)^\top C_2(\mathbf{x}, t)\mathbf{u}^* \\
& + \nabla_\mathbf{x}^\top[v^*](\mathbf{x}, t) \cdot (\mathbf{f}_0(\mathbf{x}, t) + f_1(\mathbf{x}, t)\mathbf{u}^*) \\
& + \frac{1}{2}\left(g_0 g_0^\top\right)(\mathbf{x}, t) : \nabla_\mathbf{x}\left[\nabla_\mathbf{x}^\top[v^*]\right](\mathbf{x}, t) \\
& + \sum_{\ell=1}^{n_p} \lambda_\ell(t) \int_{\mathcal{Q}_\ell} \left(v^*\left(\mathbf{x} + \widehat{\boldsymbol{h}}_{0,\ell}(\mathbf{x}, t, q_\ell), t\right) - v^*(\mathbf{x}, t) \right) \phi_{Q_\ell}(q_\ell) dq_\ell,
\end{aligned} \qquad (9.36)$$

where the double-dot product (:) is defined as a trace in (5.89) and the ℓth jump-amplitude vector is $\widehat{\boldsymbol{h}}_{0,\ell}(\mathbf{x}, t, q_\ell) \equiv [h_{0,i,\ell}(\mathbf{x}, t, q_\ell)]_{n_x \times 1}$ for $\ell = 1 : n_p$.

Let the state dimension be n_x and realized state vector be given by $\mathbf{x} = [x_i]_{n_x \times 1}$. In discrete form, the state vector with a common N_x nodes per dimension becomes $\mathbf{x} = [x_i]_{n_x \times 1} \to \mathbf{X_j} = [X_{i,j_i}]_{n_x \times 1}$, representing a single point in state space, given one j_i for each state i from the range $j_i = 1 : N_x$ for $i = 1 : n_x$ with $X_{i,j_i} = x_{i,0} + (j_i - 1)\Delta X_i$ and $\Delta X_i = (x_{i,\max} - x_{i,0})/(N_x - 1)$. The entire set of points in state space can be represented by $\boldsymbol{\mathcal{X}} = [X_{i,j}]_{n_x \times N_x}$ with corresponding vector index $J = [J_{i,j}]_{nx \times N_x}$. This representation leads to a large scale expansion of the independent variables of SDP from that in (9.37) for each current $k = 1 : N_t$,

using CFD for each state component of state partial derivatives:

$$
\begin{aligned}
v^*(\mathbf{X_j}, T_k) &\to \; V_{J,k} \equiv [V_{j_1,j_2,\ldots,j_{n_x},k}]_{N_x \times N_x \times \cdots \times N_x}, \\
v_t^*(\mathbf{X_j}, T_k) &\to \; (V_{J,k+1} - V_{J,k})/(-\Delta t), \\
\nabla_x[v^*](\mathbf{X_j}, T_k) &\to \; \mathbf{DV}_{J,k} \equiv [\mathrm{DV}_{i,j_1,\ldots,j_{n_x},k}]_{n_x \times N_x \times \cdots \times N_x} \\
&= \big[\big(V_{j_1+\delta_{i,1},\ldots,j_{n_x}+\delta_{i,n_x},k} \\
&\quad - V_{j_1-\delta_{i,1},\ldots,j_{n_x}-\delta_{i,n_x},k} \big) / \Delta X_i \big]_{n_x \times N_x \times \cdots \times N_x}, \\
\nabla_\mathbf{x}\left[\nabla_\mathbf{x}^\top[v^*]\right](\mathbf{X_j}, T_k) &\to \; \mathrm{DDV}_{J,k} \equiv [\mathrm{DDV}_{i,j,j_1,\ldots,j_{n_x},k}]_{n_x \times n_x \times N_x \times \cdots \times N_x}, \quad (9.37) \\
u^{(\mathrm{reg})}(\mathbf{X_j}, T_k) &\to \; \mathbf{UR}_{J,k} \equiv [\mathrm{UR}_{i,j_1,\ldots,j_{n_x},k}]_{n_x \times N_x \times \cdots \times N_x} \\
&= -\left(C_{1,J,k} + F_{1,J,k}\mathrm{DV}_{j,k}\right)./C_{2,J,k}, \\
u^*(\mathbf{X_j}, T_k) &\to \; \mathbf{US}_{J,k} \equiv [\mathrm{US}_{i,j_1,\ldots,j_{n_x},k}]_{n_x \times N_x \times \cdots \times N_x} \\
&= [\min(\mathrm{UMAX}_i, \max(\mathrm{UMIN}_i \\
&\quad , \mathrm{UR}_{i,j_1,\ldots,j_{n_x},k}))]_{n_x \times N_x \times \cdots \times N_x}, \\
v^*(\mathbf{X_j} + \widehat{\boldsymbol{h}}_{0,\ell}(\mathbf{X_j}, T_k, q_\ell), T_k) &\to \; \mathrm{VH}_{J,k}(q_\ell).
\end{aligned}
$$

where $\delta_{i,j}$ is the Kronecker delta, $F_{i,J,k} = f_i(X_J, T_k)$ for $i = 0:1$, $C_{i,J,k} = c_{i-}(X_J, T_k)$ for $i = 0:2$, the symbol "./" denotes element-wise division, $\mathrm{UMIN}_i = U_i^{(\min)}$ for $i = 1:n_x$ and $\mathrm{UMAX}_i = U_i^{(\max)}$ for $i = 1:n_x$. The hypercube form of the control constraints is used here only for a concrete example, and can be replaced for what is appropriate in the application of interest.

The Hessian matrix is not necessarily diagonal and is only so if the diffusion coefficient $0.5(g_0 g_0^\top)(\mathbf{x}, t)$ is diagonal, so the full, asymmetric Hessian is given here:

$$
\begin{aligned}
\mathrm{DDV}_{J,k} \equiv \; & \left[\mathrm{DDV}_{i,j,j_1,\ldots,j_{n_x},k}\right]_{n_x \times n_x \times N_x \times \cdots \times N_x} \\
= \; & \Big[\big(V_{j_1+\delta_{i,1},\ldots,j_{n_x}+\delta_{i,n_x},k} - 2V_{j_1,\ldots,j_{n_x},k} + V_{j_1-\delta_{i,1},\ldots,j_{n_x}-\delta_{i,n_x},k} \big) \, \delta_{i,j} \, / \Delta X_i^2 \\
& +0.25 \big(V_{j_1+\delta_{i,1}+\delta_{j,1},\ldots,j_{n_x}+\delta_{i,n_x}+\delta_{j,n_x},k} \\
& - V_{j_1-\delta_{i,1}+\delta_{j,1},\ldots,j_{n_x}-\delta_{i,n_x}1+\delta_{j,n_x},k} - V_{j_1+\delta_{i,1}-\delta_{j,1},\ldots,j_{n_x}+\delta_{i,n_x}-\delta_{j,n_x},k} \\
& + V_{j_1-\delta_{i,1}-\delta_{j,1},\ldots,j_{n_x}-\delta_{i,n_x}-\delta_{j,n_x},k} \big) \\
& \cdot (1-\delta_{i,j}) \, / (\Delta X_i \Delta X_j) \Big]_{n_x \times n_x \times N_x \times \cdots \times N_x},
\end{aligned} \quad (9.38)
$$

in the second order accuracy, central finite difference form. If the Hessian is diagonal, then only the second line of (9.38) is needed. The off-diagonal terms, i.e., when $i \neq j$, are conveniently calculated as the operator product of two central finite differences for the two independent partials. In the case where the off-diagonal terms are significant enough that they can affect stability and convergence, Kushner and Dupuis [175] recommend a better form form than that given in (9.38) for the cross term in $\mathrm{DDV}_{J,k}$.

These are the basic numerical ingredients for converting the one-state problem **Crank-Nicolson Extrapolator-Predictor-Corrector method** in Subsection 9.1.2 to the multi-state problem.

Curse of Dimensionality

In the full Hessian case, the Hessian is the largest array that will be needed in the computation and will basically determine the order of both computing and memory

demands for the solution of the PDE of SDP. In this full case the demands per time-step k will then be roughly proportional to the order of the $\text{DDV}_{\mathbf{j},k}$ count or

$$\text{O}\left(N_{\text{DDV}}\right) = \text{O}\left(n_x^2 \cdot \prod_{i=1}^{n_x} N_x\right) = \text{O}\left(n_x^2 \cdot N_x^{n_x}\right) = \text{O}\left(n_x^2 \cdot e^{n_x \ln(N_x)}\right) , \ (9.39)$$

which is n_x times the size of the vector functions like $\text{DV}_{\mathbf{j},k}$ and will grow exponentially with state dimension times the logarithm of the common number of nodes per dimension. If the number of nodes per dimension varies, i.e., N_i nodes in dimension i, then the geometric mean $N_x = (\sum_{i=1}^{n_x} N_i)^{1/n_x}$ can be used in place of the common value N_x in the above exponential estimate. This exponential growth in demands quantifies the exponential complexity in solving the PIDE of SDP and is called **Bellman's curse of dimensionality**. However, the very same exponential complexity (9.39) is found in high dimensional, second order PDEs. If there are $n_x = 6$ states and there are $N_x = 64$ nodes per state using 8-byte (8B) or double words, then the order of the amount of storage required is $N_{\text{DDV}} = 8 \cdot 6^2 \cdot 64^6\text{B} = 18\text{GB}$, where 1GB is a gigabyte or a computer billion bytes or 1024^4 bytes.

If the discrete Hessian is diagonal, then the amount of storage needed is reduced to some multiple of

$$N_{\text{DV}} = 8 \cdot n_x \cdot N_x^{n_x}\text{B},$$

using 8 byte (8B) words, DDV that has the same size as DV, so in the example with $n_x = 6$ and $N_x = 64$, $N_{\text{DV}} = 8 \cdot 6 \cdot 64^6\text{B} = 3\text{GB}$, a more reasonable size for a large scale problem capable computer. If the number of nodes per dimension is reduced to 32 instead of 64, then the amount of storage needed is some multiple of $8 \cdot 6 \cdot 32^6\text{B} = 49,152\text{MB} = 0.0469\text{GB}$, approaching PC desktop capability (1MB being a megabyte or or 1024^2 bytes). The growth of the curse of dimensionality in the logarithm to the base 2 scale is illustrated in Fig. 9.1 for the diagonal Hessian size case N_{DV}. Note the top scale in the figure is about $60 \log(\text{B})$ and $2^{60}\text{B} = 1024^6\text{B}$ is 1 terabytes (1TB) or $1024^2 GB$ (1GB $= 2^{40}\text{B}$, while 1MB $= 2^{20}\text{B}$) and that is well within the capabilities of our current largest scale computers.

For parallel processing techniques in computational stochastic programming refer to Hanson's 1996 chapter [108]. See also [109] for more general supercomputing techniques that were developed originally solving computational control application problems.

9.2 Markov Chain Approximation for SDP

Another method for numerically solving stochastic dynamic programming problems in continuous time is Kushner's **Markov chain approximation** (MCA) [170, 171] that implicitly provides good convergence properties by normalizing the corresponding finite differences as proper Markov chains. In addition, MCA facilitates the proof of weak convergence using probabilistic arguments. Kushner and Dupuis's [175] method of using an auxiliary stochastic process, so that the composite stochastic process properly satisfies boundary conditions, is also treated. The summary here is in the spirit of this applied text to make the Markov chain approximation method
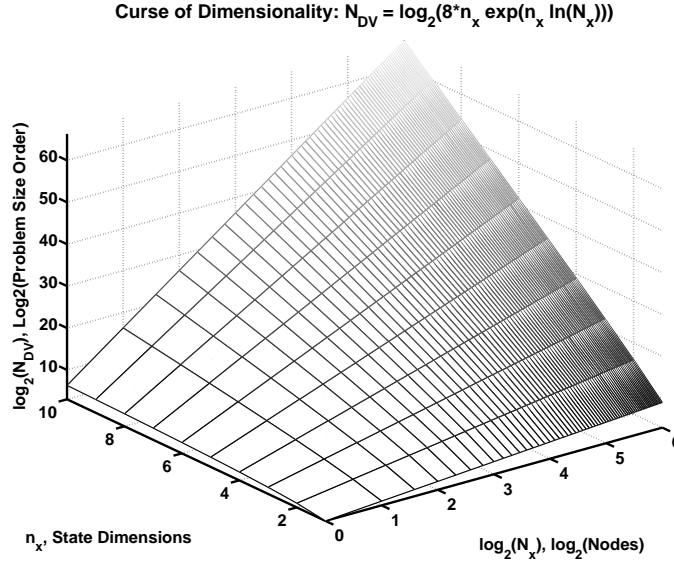
Figure 9.1. *Estimate of the logarithm to the base 2 of the order of the growth of memory and computing demands using 8 byte words to illustrate the curse of dimensionality in the diagonal Hessian case for $n_x = 1:10$ dimensions and $N_x = 1:64 = 1:2^6$ nodes per dimension. Note that 1KB or one kilobyte has a base 2 exponent of $10 = log_2(2^{10})$, while the base 2 exponent is 20 for 1MB, 40 for 1GB and is 60 for 1TB.*

more accessible, concentrating on the techniques, rather than the problems and formal definitions.

9.2.1 The MCA Formulation for Stochastic Diffusions

Although MCA is valid for jump-diffusions, only diffusions will be considered here to keep the complexity manageable and the reader can consult [175] for a more complete treatment of MCA. Let the diffusion satisfy the SDE,

$$dX(t) \stackrel{\text{sym}}{=} f(X(t), U(t), t)dt + g(X(t), t)dW(t) , \qquad (9.40)$$

where the notation otherwise is the same as in (9.1) of the prior section, with f and g being bounded, continuous and Lipshitz continuous in X, while f has the same properties in U, but uniformly. For later reference, the following conditional infinitesimal moments are

$$\begin{aligned}
E[dX(t)|X(t) = x, U(t) = u] &= f(x, u, t)dt, \\
\text{Var}[dX(t)|X(t) = x, U(t) = u] &= g^2(x, t)dt.
\end{aligned} \qquad (9.41)$$

Let the minimal, expected costs be defined as

$$v^*(x,t) \equiv \min_{U[t,t_f)} \left[\mathop{\mathrm{E}}_{(dW,dP)[t,tf)} \left[\int_t^{t_f} C(X(s),U(s),s)ds + S(X(t_f),t_f) \right. \right.$$
$$\left. \left. \Big| X(t) = x, U(t) = u \right] \right], \tag{9.42}$$

for $t_0 \leq t < t_f$. The corresponding PDE of stochastic dynamic programming is

$$0 = v_t^*(x,t) + \min_u [\mathcal{H}(x,u,t)]$$

$$\equiv v_t^*(x,t) + \min_u \left[C(x,u,t) + f(x,u,t)v_x^*(x,t) + \tfrac{1}{2}g^2(x,t)v_{xx}^*(x,t) \right] \tag{9.43}$$

$$= v_t^*(x,t) + \mathcal{H}^*(x,t) .$$

The first step of the numerical part of the MCA procedure is to approximate the backward PDE (9.43) by a backward Euler method in time for simplicity. Then using the kth time step at t_k with optimal value $v_k(x) \simeq v^*(x,t_k)$, the next value is

$$v_{k-1}(x) = v_k(x) + \Delta t_{k-1} \min_u \left[C_k(x,u) + f_k(x,u)v_k'(x) + \tfrac{1}{2}g_k^2(x)v_k''(x) \right] , \tag{9.44}$$

for forward index $k = 1 : N_t$, $t_k \equiv t_{k-1} + \Delta t_{k-1}$, $t_{N_t} = t_f$, $C_k(x,u) = C(x,u,t_k)$, $f_k(x,u,t_k)$ and $g_k(x) = g(x,t_k)$. The final condition is $v_{N_t}(x) = S(x,t_f)$. The time step Δt_{k-1} is called the MCA **interpolation time increment** and is selected to help form a proper Markov chain for convergence, so the increments are not necessarily constant. Though motivated by an approximation in time, time has been removed from the problem, i.e., the current problem is actually time-independent. Finite differences in the state come after specifying diffusion consistency conditions.

9.2.2 MCA Local Diffusion Consistency Conditions

Let ξ_k for $k \geq 0$ denote a Markov chain of discrete stages, intended as a discrete model for the state x, whose spacing is the order of some state mesh measure ΔX, i.e., $|\Delta \xi_k| = \mathrm{O}(\Delta X)$ where $\Delta \xi_k \equiv \xi_{k+1} - \xi_k$. Let the Markov chain **transition probability** for diffusions (D) be defined by

$$p^{(D)}(x,y|u) \equiv \mathrm{Prob}[\xi_{k+1} = y | \xi_j, u_j, j < k, \xi_k = x, u_k = u] \tag{9.45}$$

for transitions from current stage $\xi_k = x$ to the next stage $\xi_{k+1} = y$ using control policy $u_k = u$. (The term stage is used to denote a discrete state.) These transitions must satisfy the probability rules of non-negativity $p^{(D)}(x,y|u) \geq 0$ and probability conservation for transitions, $\sum_\ell p^{(D)}(x,X_\ell|u) = 1$, under current control u and over probable state transitions $y = X_\ell$. The increments $\Delta \xi_k$ must satisfy the MCA **local diffusion consistency** conditions:

$$\mathrm{E}[\Delta \xi_k | x, u] \equiv \sum_\ell (X_\ell - x) \cdot p^{(D)}(x,X_\ell|u) = \Delta t_{k-1} \cdot (f_k(x,u) + \mathrm{o}(1));$$

$$\mathrm{Var}[\Delta \xi_k | x, u] \equiv \sum_\ell (X_\ell - x - \mathrm{E}[\Delta \xi_k | x, u])^2 \cdot p^{(D)}(x,X_\ell|u) \tag{9.46}$$
$$= \Delta t_{k-1} \cdot (g_k^2(x) + \mathrm{o}(1)),$$

with $\Delta\xi_k \to 0^+$ as $\Delta X \to 0^+$, for $k = 0 : N_t-1$. The conditions are consistent with the first two conditional infinitesimal moments (9.41) of a stochastic diffusion approximation corresponding to the SDE (9.40), so they are neccesary preconditions for convergence of the Markov chain to the diffusion SDE (9.40).

See Sect. 8.8 on p. 326 or Feller, vol. II [84]) for more information. Also, see Kloeden and Platen [161] for stricter definitions of diffusion consistency conditions. The generalization of these diffusion consistency conditions to jump-diffusions is much more complicated, but is treated in Subsect. 9.2.4.

The discrete process can be used to construct a **piece-wise constant** $(pw\phi)$ **interpolation** of the state and control processes in continuous time, i.e.,

$$(X^{(\text{pw}\phi)}(t), U^{(\text{pw}\phi)}(t) = \{(\xi_k, u_k),\ \ t_{k-1} \le t < t_{k-1} + \Delta t_{k-1} = t_k, \atop \text{for } k \ge 1\}, \quad (9.47)$$

with the relationship between the interpolation times t_k and interpolation time increments Δt_{k-1} being $t_{k+1} = \sum_{j=0}^{k} \Delta t_j$. In general, the time increments will depend on ξ_k and u_k, which also depends on the order of state mesh ΔX, so $\Delta t_{k-1} = \Delta t_{k-1}(\xi_k, u_k; \Delta X)$. As the state mesh goes ot zero, it is required that the maximal state mesh go to zero, i.e., $\max_{u,x}[\Delta t_{k-1}(x, u; \Delta X)] \to 0^+$.

9.2.3   MCA Numerical Finite Differences for State Derivatives and Construction of Transition Probabilities

Construction of the Markov chain transition probabilities is found by finite differencing the state derivative. The state derivative is upwinded by first order forward or backward differences (UFD) for greater stability depending on the sign of the drift coefficient $f_k(x, u, t)$ as in (9.33),

$$v_k'(x) \simeq \left\{ \begin{array}{ll} (\dfrac{v_k(x + \Delta X) - v_k(x)}{\Delta X}, & f_k(x, u) \ge 0 \\ (\dfrac{v_k(x) - v_k(x - \Delta X)}{\Delta X}, & f_k(x, u) < 0 \end{array} \right\} \quad (9.48)$$

and central finite differences (CFDs) of second order accuracy are used for the second state partial

$$v_k''(x) \simeq \frac{v_k(x + \Delta X) - 2v_k(x) + v_k(x - \Delta X)}{\Delta X^2}. \quad (9.49)$$

Alternately, second order upwinding can be used for the state first derivative so that the accuracy is consistent with $O(\Delta X^2)$ accuracy of the second derivative used above, but this leads to a double jump in the state by $2 \pm \Delta X$ so this complication will not be introduced here although the larger $O(\Delta X)$ error **numerically pollutes** the smaller $O(\Delta X^2)$ error for small ΔX. Using the $O(\Delta X^2)$ forward and backward finite differences of the form used for derivative boundary conditions in (9.32) would not be useful since the alternating signs would lead to improper, negative transition probabilities for a least one double step transition.

Substituting into Eq. (9.44) for $v_{k-1}(x)$ and then collecting the coefficients in terms of transition probabilities,

$$
\begin{aligned}
v_{k-1}(x) = \quad & \min_{u_{k-1}} \Big[\Delta t_{k-1} \cdot C_k(x, u_{k-1}) + p_k^{(D)}(x, x|u_{k-1}) \cdot v_k(x) \\
& + p_k^{(D)}(x, x + \Delta X|u_{k-1}) \cdot v_k(x + \Delta X) \\
& + p_k^{(D)}(x, x - \Delta X|u_{k-1}) \cdot v_k(x - \Delta X) \Big],
\end{aligned}
\tag{9.50}
$$

the transition probabilities are found to be

$$
p_k^{(D)}(x, x|u_{k-1}) = 1 - \frac{\Delta t_{k-1}}{\Delta X^2} \cdot \left(g_k^2(x) + \Delta X |f_k(x, u_{k-1})| \right),
\tag{9.51}
$$

$$
p_k^{(D)}(x, x + \Delta X|u_{k-1}) = \frac{\Delta t_{k-1}}{\Delta X^2} \cdot \left(0.5 g_k^2(x) + \Delta X [f_k(x, u_{k-1})]_+ \right),
\tag{9.52}
$$

$$
p_k^{(D)}(x, x - \Delta X|u_{k-1}) = \frac{\Delta t_{k-1}}{\Delta X^2} \cdot \left(0.5 g_k^2(x) + \Delta X [f_k(x, u_{k-1})]_- \right),
\tag{9.53}
$$

where $[f]_\pm \equiv \max[\pm f] \geq 0$. Upwinding ensures that all terms in the coefficients of Δt_{k-1} are non-negative, so that the up and down transition probabilities, $p_k^{(D)}(x, x + \Delta X|u_{k-1})$ and $p_k^{(D)}(x, x - \Delta X|u_{k-1})$ are nonnegative. Note that on the right-hand-side of the conservation law (9.50) for the transition probabilities to get the value function for the past time t_{k-1}, the value function is evaluated at the current time t_k, but the control is for the past time t_{k-1} which makes it seem like the control is implicit. However, u_{k-1} is thought to be the control to get the state x from time t_{k-1} to time t_k and the optimization over u_{k-1} will determine u_{k-1} in terms of values at t_k anyway, so is not really an implicit term. Genuine implicit methods are discussed in Kushner and Dupuis [175].

It is clear that Δt_{k-1} must be sufficiently small so that the state self-transition probability $p_k^{(D)}(x, x|u_{k-1})$ is non-negative, i.e., is a proper probability. This implies the following convergence criteria

$$
\frac{\Delta t_{k-1}}{\Delta X^2} \leq \frac{1}{\gamma_k^2(x) + \Delta X |f_k(x, u_{k-1})|}
\tag{9.54}
$$

or in terms of a generalization of the parabolic mesh ratio condition

$$
\left(g_k^2(x) + \Delta X |f_k(x, u_{k-1})| \right) \cdot \frac{\Delta t_{k-1}}{(\Delta X)^2} \leq 1,
\tag{9.55}
$$

including both the diffusion coefficient and the upwinded drift term in the scaling of $\Delta t_{k-1}/(\Delta X)^2$. Since (9.54) should hold for all discrete time steps k, then we should have

$$
\max_{x,u,k} \left[\left(g_k^2(x) + \Delta X |f_k(x, u)| \right) \frac{\Delta t_{k-1}}{\Delta X^2} \right] \leq 1.
\tag{9.56}
$$

The diffusion consistency conditions (9.46) can be confirmed in this three local state case directly,

$$
\begin{aligned}
\mathrm{E}[\Delta\xi_k|x,u_{k-1}] = \; & p_k^{(D)}(x,x|u_{k-1})\cdot 0 + p_k^{(D)}(x,x+\Delta X|u_{k-1})\cdot(+\Delta X) \\
& +p_k^{(D)}(x,x-\Delta X|u_{k-1})\cdot(-\Delta X) \\
= \; & \Delta t_{k-1}\cdot([f_k(x,u_{k-1})]_+ - [f_k(x,u_{k-1})]_-) \\
\equiv \; & \Delta t_{k-1}\cdot f_k(x,u_{k-1}),
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{Var}[\Delta\xi_k|x,u_{k-1}] = \; & p_k^{(D)}(x,x|u_{k-1})\cdot(\Delta t_{k-1}f_k(x,u_{k-1}))^2 \\
& +p_k^{(D)}(x,x+\Delta X|u_{k-1})\cdot(\Delta X-\Delta t_{k-1}f_k(x,u_{k-1}))^2 \\
& +p_k^{(D)}(x,x-\Delta X|u_{k-1})\cdot(-\Delta X-\Delta t_{k-1}f_k(x,u_{k-1}))^2 \\
= \; & \Delta t_{k-1}\cdot\left(g_k^2 + |f_k(x,u_{k-1})|\Delta X - 2\Delta t_{k-1}f_k^2(x,u_{k-1})\right) \\
= \; & \Delta t_{k-1}\cdot\left(g_k^2 + \mathrm{o}(1)\right)
\end{aligned}
$$

as $\Delta X \to 0^+$ and consequently $\Delta t_{k-1} \to 0^+$.

Upon proper choice of the time and state grids satisfying (9.56), for example in the case of regular grids as used in the previous section in (9.10) with N_t nodes in t on $[t_0,t_f]$ and N_x nodes in x on $[x_0,x_{\max}]$, $T_k = t_f - (k-1)\Delta t$ for $k=1:N_t$, $\Delta t_{k-1} = \Delta t = (t_f-t_0)/(N_t-1)$ and $X_j = x_0 + (j-1)\Delta X$ for $j = 1:N_x$, $\Delta X = (x_{\max}-x_0)/(N_x-1)$, then

$$
\begin{aligned}
V_{j,k-1} \equiv \; & v_{k-1}(X_j) \\
= \; & \Delta t\cdot C_k(X_j,U_{j,k-1}) + p_k^{(D)}(X_j,X_j|U_{j,k-1})\cdot V_{j,k} \\
& +p_k^{(D)}(X_j,X_{j+1}|U_{j,k-1})\cdot V_{j+1,k} \\
& +p_k^{(D)}(X_j,X_{j-1}|U_{j,k-1})\cdot V_{j-1,k},
\end{aligned}
\tag{9.57}
$$

when the optimal control is

$$
\begin{aligned}
U_{j,k-1} = \; & \mathrm{argmin}_{u_{k-1}}\Big[\Delta t_{k-1}\cdot C_k(X_j,u_{k-1}) + p_k^{(D)}(X_j,X_j|u_{k-1})\cdot V_{j,k} \\
& +p_k^{(D)}(X_j,X_{j+1}|u_{k-1})\cdot V_{j+1,k} \\
& +p_k^{(D)}(X_j,X_{j-1}|u_{k-1})\cdot V_{j-1,k}\Big],
\end{aligned}
\tag{9.58}
$$

for $j = 1:N_x$ for each stage $k = N_t : -1 : 2$ in backward order. Note that in [175], Kushner and Dupuis suggest a preference for selecting the interpolation time-step Δt_{k-1} so that the self-transition probability $p^{(D)}(x,x|u)$ vanishes leading to a renormalization of the non-self-transition probabilities like $p^{(D)}(x,x\pm\Delta X|u)$.

In this section, the **Markov chain approximation** has only been summarized to convey the main ideas, but for those interested in the weak convergence proofs and related theory they should consult [172, 175] and additional references therein.

2006/5/24
page 351

9.2.4 MCA Extensions to Include Jump Processes

In [175, Sect. 5.6, pp. 127-133], Kushner and Dupuis briefly present the extensions of the **Markov chain approximation** for diffusions to that for jump-diffusions. Earlier Kushner and DiMasi [174] made contributions to the jump-diffusion optimal control problem, while Kushner [173] more recently gave further results on existence and numerical methods for the problem.

The main idea is based upon the facts that the Poisson process is instantaneous compared to the continuity of the diffusion process and that the Poisson process during short time intervals Δt can be asymptotically treated as a zero-one Bernoulli process as mentioned in prior chapters. Starting with the jump-diffusion SDE extension of (9.40),

$$
\begin{aligned}
dX(t) \overset{\text{sym}}{=} \quad & f(X(t), U(t), t)dt + g(X(t), t)dW(t)\,, \\
& + \sum_{k=1}^{dP(t)} h(X(T_k), T_k, Q_k)\,,
\end{aligned}
\tag{9.59}
$$

where $dP(t)$ is the differential Poisson process with rate λ, $h(x, t, q)$ is the state jump-amplitude, T_k is the kth jump-time and Q_k is the kth jump-amplitude mark with generalized probability density $\phi_Q(q)$. The conditional infinitesimal moments are given by

$$
\begin{aligned}
\mathrm{E}[dX(t)|X(t) = x, U(t) = u] = \quad & f(x, u, t)dt + \mathrm{E}_Q[h(x, t, Q)]\lambda dt, \\
\mathrm{Var}[dX(t)|X(t) = x, U(t) = u] = \quad & g^2(x, t)dt + \mathrm{E}_Q[h^2(x, t, Q)]\lambda dt.
\end{aligned}
\tag{9.60}
$$

By separability of the diffusion and the jumps for sufficiently small time-steps Δt_{k-1}, the diffusion transition probabilities are unchanged, $p_k^{(D)}(x, y|u)$ for stage k. The probability of zero or one Poisson jump in time-steps of Δt_{k-1} can be written

$$
p_{j,k}^{(J)} = \left\{
\begin{array}{ll}
1 - \lambda\Delta t_{k-1} + \mathrm{o}(\Delta t_{k-1}), & j = 0 \text{ jumps} \\
\lambda\Delta t_{k-1} + \mathrm{o}(\Delta t_{k-1}), & j = 1 \text{ jump} \\
\mathrm{o}(\Delta t_{k-1}), & j \geq 2 \text{ jumps}
\end{array}
\right\},
\tag{9.61}
$$

as $\Delta t_{k-1} \to 0^+$.

For the discretization jump-amplitude function $h(x, t, q)$ of the corresponding compound Poisson process, a concrete rather than the abstract formulation of Kushner and Dupuis [175] will be given so that the transition of a piece-wise-constant pre-jump stage $x = X_j$ for some j to a piece-wise-constant post-jump stage $y = X_\ell$ for some ℓ, where $X_{j+1} = X_j + \Delta X_j$ for $j = 1 : N_x - 1$, $X_1 = x_0$, $X_{N_x} = x_{\max}$ and the mesh is given by $\Delta X = \max_j (\Delta X_j) \to 0^+$. However, the treatment of jumps is much more complicated than that for diffusion whose dependence is only local, depending on only nearest neighbor or similarly close nodes, but jump behavior is globally dependent on nodes that may be remote from the current node X_j. Also, the connection of the jump-amplitude function to the jump-amplitude random mark variable q will be clarified. The jump-amplitude may be continuously distributed due to a continuous mark density $\phi_Q(q)$. It is assumed that post-jump stage $y = x + h(x, t, q)$ is uniquely invertible with q as a function of y given x, but

it is necessary to have a set target $\mathcal{S}(X_\ell)$ rather than a point target $y = X_\ell$ so a corresponding set $\mathcal{Q}_{j,\ell}(t)$ of positive probability measure can be found. Let $\mathcal{S}(X_\ell)$ be a partition of the state domain $[X_1, X_{N_x}]$ such that

$$\sum_{\ell=1}^{N_x} \mathcal{S}(X_\ell) = [X_1, X_{N_x}].$$

The $\mathcal{S}(X_\ell)$ will usually depend on the application due particular boundary conditions, singular points or related zero points, which could lead to forward or backward shifted intervals or intervals centered about X_ℓ as with rounding. The discretized, here piece-wise-continuous (pwc), instead of the prior piece-wise-constant ($pw\not{c}$) step functions, jump-amplitude $H_{j,\ell}^{(\mathrm{pwc})}(t)$ given the stage set $\mathcal{S}(X_\ell)$ is

$$H_{j,\ell}^{(\mathrm{pwc})}(t) = h(X_j, t, \mathcal{Q}_{j,\ell}(t)) = \mathcal{S}(X_\ell) - X_j, \tag{9.62}$$

implicitly defining the mark set $\mathcal{Q}_{j,\ell}(t)$ for $1 \le j < \infty$ and $1 \le \ell < \infty$. This ensures that a jump takes a proper (pwc) stage X_j to a proper (pwc) stage X_ℓ defined by the set $\mathcal{S}(X_\ell)$. Given a jump it is also necessary to know the corresponding probability of the transition referenced by (9.62) , i.e.,

$$\begin{aligned}
\mathrm{Prob}\,[y = x + h(x,t,q) \in \mathcal{S}(X_\ell) \mid x = X_j, \ y \in \mathcal{S}(X_\ell)] \\
= \overline{\Phi}(X_j, X_\ell, t) \equiv \int_{\mathcal{Q}_{j,\ell}(t)} \phi_Q(q) dq,
\end{aligned} \tag{9.63}$$

where $\phi_Q(q)$ is the generalized mark density with corresponding distribution $\Phi_Q(q)$, except that when $h(X_{\hat{j}}, t, q) = 0$ for some $\hat{j}$, i.e., there is a **zero jump** and $y \in \mathcal{S}(X_\ell)$ is not achievable for general ℓ, then $\overline{\Phi}_{\hat{j},\ell}(t) \equiv 0$. In the case that $\overline{\Phi}(X_j, X_\ell, t)$ leads to a probabilistically deficient distribution, in general the renormalized form is

$$\widehat{\Phi}(X_j, X_\ell, t) = \overline{\Phi}(X_j, X_\ell, t)\Big/ \overline{\overline{\Phi}}(X_j, t) , \tag{9.64}$$

where

$$\overline{\overline{\Phi}}(X_j, t) \equiv \sum_{\ell=1}^{N_x} \overline{\Phi}(X_j, X_\ell, t) = \sum_{\ell=1}^{N_x} \int_{\mathcal{Q}_{j,\ell}(t)} \phi_Q(q) dq.$$

Example 9.3. Geometric Jump-Diffusion Target Mark Set Calculations:
For the geometric jump-diffusion used in finance, with linear jump-amplitude

$$h(x, t, q) = xJ(q, t),$$

it is convenient to choose the log-return jump as the mark, i.e.,

$$q = [\ln(X)](t) = \ln\left(\left(X(t^-) + X(t^-)J(q,t^-)\right)/X(t^-)\right) = \ln(1 + J(q, t^-)),$$

*so $h(x, t, q) = x(\exp(q) - 1)$. Hence, $X_1 = x_0 = 0$ is a **zero point** needing special treatment since there can be no target stage except for $[X_1, X_1] = \{0\}$, so that a proper partition of $[X_1, X_{N_x}]$ would be $\mathcal{S}(X_1) = \{0\}$ and $\mathcal{S}(X_\ell) = (X_{\ell-1}, X_{\ell-2}]$ for*

$\ell = 2\!:\!N_x$. The discrete jump-amplitude $H_{1,\ell}^{(\mathrm{pwc})}(t) \equiv 0$ for definiteness when $X_1 = 0$ and

$$H_{j,\ell}^{(\mathrm{pwc})}(t) \equiv X_\ell - X_j$$

for $\ell = 2\!:\!N_x$. The target mark set is

$$\mathcal{Q}_{j,\ell}(t) = (\ln(X_{\ell-1}/X_j), \ln(X_\ell/X_j)]$$

for $\ell = 2 : N_x$ when $j > 1$. Given a mark density, then a renormalized target distribution $\widehat{\Phi}(X_j, X_\ell, t)$ can be calculated.

The Markov chain approximation $\xi_k(\Delta X)$ is **locally jump-diffusion consistent** if there is an **interpolation time interval** $\Delta t_{k-1} = \Delta t(x, u; \Delta X) \to 0^+$ uniformly in $(x, u, \Delta X)$ as the mesh gauge $\Delta X \to 0^+$ and so that

1. Along with $\Delta t(x, u; \Delta X)$, there is a locally diffusion consistent transition probability $p^{(D)}(x, y \mid u; \Delta X)$ satisfying the conditions in (9.46);

2. The **jump-diffusion transition probabilities** $p^{(JD)}(x, y \mid u; \lambda, \Delta X)$ must conserve probability over the post-jump values $y = X_\ell$ from any given pre-jump value $x = X_j$, i.e.,

$$\sum_\ell p^{(JD)}(X_j, X_\ell \mid u; \lambda, \Delta X) = 1.$$

3. Markov chain increments $\Delta \xi_k$ satisfy the MCA **jump-diffusion local consistency** conditions consistent with the jump-diffusion conditional infinitesimal moments (9.60), with replacements $f(x, u, t) \to f_{k,}(x, u)$, $g(x, t) \to g_k(x)$, $h(x, t, q) \to h_k(x, q)$, $H_{j,\ell}^{(\mathrm{pwc})}(t) \to H_{j,\ell,k}^{(\mathrm{pwc})}$ $\widehat{\Phi}(X_j, X_\ell, t) \to \widehat{\Phi}_k(X_j, X_\ell)$, under current control u and over probable state transitions

$$
\begin{aligned}
\mathrm{E}[\Delta\xi_k \mid X_j, u_{k-1}] &\equiv \sum_\ell (X_\ell - X_j) \cdot p^{(JD)}(X_j, X_\ell \mid u_{k-1}; \lambda, \Delta X) \\
&= \Delta t_{k-1} \cdot \left(f_k(X_j, u_{k-1}) + \lambda \mathrm{E}_Q[h_k(X_j, Q)] + \mathrm{o}(1) \right);
\end{aligned}
$$

$$
\begin{aligned}
\mathrm{Var}[\Delta\xi_k \mid X_j, u_{k-1}] &\equiv \sum_\ell (X_\ell - X_j - \mathrm{E}[\Delta\xi_k \mid X_j, u_{k-1}])^2 \qquad (9.65)\\
&\quad \cdot p^{(JD)}(x, X_\ell \mid u_{k-1}; \lambda, \Delta X) \\
&= \Delta t_{k-1} \cdot \left(g_k^2(x) + \lambda \mathrm{E}_Q[h_k^2(X_j, Q)] + \mathrm{o}(1) \right),
\end{aligned}
$$

with $\Delta\xi_k \to 0^+$ as $\Delta X \to 0^+$, for $k = 0\!:\!N_t-1$.

4. There is a small error factor $\varepsilon(s, u; \Delta X) = \mathrm{o}(\Delta t(x, u; \Delta X))$ that can be used to construct (Kushner and Dupuis [175], modified for clarification here) the **jump-diffusion transition probability** $p^{(JD)}(x, y \mid u; \lambda, \Delta X)$ and is of the form

$$p^{(JD)}(X_j, X_\ell \mid u; \lambda, \Delta X)$$

$$= (1 - \lambda\Delta t(X_j, u; \Delta X) - \varepsilon(X_j, u; \Delta X)) \cdot p^{(D)}(X_j, X_\ell \mid u; \Delta X) \qquad (9.66)$$

$$+ (\lambda\Delta t(X_j, u; \Delta X) + \varepsilon(X_j, u; \Delta X)) \cdot \widehat{\Phi}_k(X_j, X_\ell)\mathbf{1}_{X_\ell \in X_j + H_{j,\ell,k}^{(\mathrm{pwc})}},$$

for $1 \leq j < \infty$ and $1 \leq \ell < \infty$, where $\mathbf{1}_{\mathcal{S}}$ is the indicator function for set $\mathcal{S} = \{X_\ell \in X_j + H_{j,\ell,k}^{(\mathrm{pwc})}\}$ and is used so the term it multiplies is only used for a jump.

By using the conservation laws

$$\sum_{\ell=1}^{N_x} p^{(D)}(X_j, X_\ell \mid u; \Delta X) = 1$$

and

$$\sum_{\ell=1}^{N_x} \widehat{\Phi}(X_j, X_\ell, t) = 1,$$

it is easy to show the constructed jump-diffusion transition probability in (9.66) is conserved, i.e.,

$$\sum_{\ell=1}^{N_x} p^{(JD)}(X_j, X_\ell \mid u; \lambda, \Delta X) = 1.$$

The error factor $\varepsilon(s, u; \Delta X)$ reflects the the asymptotically small error terms $\mathrm{o}(\Delta t_{k-1})$ in the Poisson counting process definition (9.61), but is selected so the conservation is exact.

Using the first moment diffusion local consistency condition in (9.46) and a mark density weighted rectangular integration rule,

$$\mathrm{E}_Q[h_k(x, Q)] \simeq \sum_{\ell=1}^{N_x} H_{j,\ell,k}^{(\mathrm{pwc})} \widehat{\Phi}_k(X_j, X_\ell).$$

Then,

$$
\begin{aligned}
\mathrm{E}[\Delta \xi_k \mid X_j, u] &\simeq \; \Delta t_{k-1}(X_j, u; \Delta X) \cdot (f_k(X_j, u) + \mathrm{E}_Q[h_k(x, Q)] + \mathrm{o}(1)) \\
&= \; \overline{X}^{(D)} + \overline{X}^{(J)},
\end{aligned}
$$

splitting the diffusion and jump parts. Similarly, for the second moment jump-diffusion consistency condition, except with more algebra with the above splitting and more small time asymptotics in absorbing all quadratic and smaller time increments into $\Delta t_{k-1} \cdot \mathrm{o}(1)$, it can be demonstrated that

$$\mathrm{Var}[\Delta \xi_k \mid X_j, u] \simeq \Delta t_{k-1}(X_j, u; \Delta X) \cdot \left(g_k^2(X_j) + \mathrm{E}_Q[h_k^2(x, Q)] + \mathrm{o}(1) \right).$$

Further evaluations require knowledge of the mark density, the jump-diffusion coefficients (f, g, h) and the boundary condition on the state domain. Due to the global nature of the compound jump process with jump beyond the local nodes needed by the diffusion component process, the diffusion mesh ratio criteria (9.56) (or (9.30) in case the central finite differences are usable) will have to suffice for practical reasons. See Kushner and Dupius [175] for information on reflected boundary conditions and other techniques for handling boundary conditions when there are jumps.

Suggested References for Further Reading

- Chung, Hanson and Xu, 1992 [54].

- Douglas and Dupont, 1970 [72].

- Douglas, 1979 [73] .

- Dyer and McReynolds, 1979 [76].

- Gunzburger, 2003 [101].

- Hanson, 1989 [106], 1991 [107], 1996 [108] and 2003 [109, 110].

- Hanson and Naimipour, 1993 [111].

- Kushner, 1976 [170] and 1990 [171], 2000a [172] and 2000b [173].

- Kushner amd DiMasi, 1978 [174].

- Kushner and Dupuis, 2001 [175].

- Kushner and Yin, 1997 [177].

- Larson, 1967 [178].

- Naimipour and Hanson, 1993 [212].

- Polak, 1973 [223].

- Press et al., 2002 [226].

- Westman and Hanson, 1997 [269] and 2000 [272].

- Zhu and Hanson, 2006 [286].