International Journal of Control

ISSN: 0020-7179 (Print) 1366-5820 (Online) Journal homepage: https://www.tandfonline.com/loi/tcon20

A Second-order Gradient Method for Determining Optimal Trajectories of Non-linear Discrete-time Systems

By DAVID MAYNE

To cite this article: By DAVID MAYNE (1966) A Second-order Gradient Method for Determining Optimal Trajectories of Non-linear Discrete-time Systems, International Journal of Control, 3:1, 85-95, DOI: 10.1080/00207176608921369

To link to this article: https://doi.org/10.1080/00207176608921369

Published online: 24 Feb 2007.

INT. J. CONTROL, 1966, VOL. 3, NO. 1, 85–95

A Second-order Gradient Method for Determining Optimal Trajectories of Non-linear Discrete-time Systems

By DAVID MAYNE
Electrical Engineering Department,
Imperial College of Science and Technology,
Exhibition Road, London, S.W.7.

[Received December 16, 1965]

ABSTRACT

A second-order method of successively improving a control sequence for a non-linear discrete-time system is derived. One step convergence is obtained for linear systems with quadratic performance functions. Although the results are of interest in their own right, a second-order method for continuous-time systems is obtained by formally allowing the sampling interval to approach zero. The equations so obtained differ slightly, because of the method of derivation, from results already obtained using the calculus of variations approach. The difference, which is an advantage of the method described in this paper, is that one vector differential equation less has to be integrated. The approach used in the derivation is motivated by dynamic programming and facilitates the application of gradient methods to stochastic problems which will be the subject of a future paper.

§ 1. INTRODUCTION

FIRST and second-order methods for optimizing continuous-time systems have been proposed (Bryson and Denham 1962, Merriam 1964, Mitter 1965). In this paper equivalent methods for discrete-time systems are derived. A non-optimal trajectory is generated using a nominal control sequence. The effect on the return (cost) function of small variations of the control sequence is determined. This enables an improved control sequence to be chosen. The procedure is then repeated until the optimal control sequence is obtained. The derivation, which is formal, uses the dynamic programming technique (Bellman and Dreyfus 1962); however, in place of the optimal return function V^0, which is used in conventional dynamic programming optimizations and which requires immense storage facilities, the variable ΔV^0, which is the optimal variation in the non-optimal return function due to a small variation in the state variable, is used. ΔV^0 can be expanded in a power series of the variation of the state variable, and difference equations, analogous to those for linear systems, derived for the coefficients of the power series.

Consider a non-linear system described by the difference equations:

$$x_k = f_{k-1}(x_{k-1}, u_{k-1}). \tag{1}$$

† Communicated by Dr. A. T. Fuller.

The cost (return) function is:

$$V_r = \sum_{k=r}^{N-1} L_k(x_k, u_k) + L_N(x_N). \tag{2}$$

The optimization problem is: determine the sequence of control actions $\{u_k, k = 1 \ldots N-1\}$ which optimizes (minimizes) V_1, given the initial condition x_1.

§ 2. Second-order Method

A nominal control sequence $\{u_k, k = 1 \ldots N-1\}$ is chosen and the resultant state variable sequence $\{x_k, k = 1 \ldots N\}$ calculated using eqn. (1) and the given initial condition x_1. These sequences are stored. We now proceed, in reverse time from $k = N$ to $k = 1$, to chose optimal incremental control laws:

$$\delta u_k = g_k(\delta x_k), \tag{3}$$

valid for a small region about the initial trajectory, and which enable improved control and state variable sequences to be calculated.

Since from (1), δx_{k+r} for all $r \geqslant 1$ is a function of $\delta x_k, \delta u_k \ldots \delta u_{k+r-1}$, we can express ΔV_k as a function of $\delta x_k, \delta u_k \ldots \delta u_{N-1}$. If ΔV_k is optimized with respect to $\delta u_k \ldots \delta u_{N-1}$ (it is shown later how the magnitude of the variables $\delta u_k \ldots \delta u_{N-1}$ may be constrained if required) the resultant $\Delta V_k{}^0$ is a function of δx_k only:

$$\Delta V_k{}^0(\delta x_k) = \min_{\delta u_k \ldots \delta u_{N-1}} \Delta V_k(\delta x_k, \delta u_k \ldots \delta u_{N-1}). \tag{4}$$

$\Delta V_k{}^0$ is, of course, a function of x_1 and the initial control sequence $\{u_k\}$ as well. We assume that δx_k is sufficiently small and that $\Delta V_k{}^0$ is sufficiently smooth to justify a power series expansion to second-order terms only:

$$\Delta V_k{}^0 = \tfrac{1}{2}\langle \delta x_k, V_{xx_k} \delta x_k \rangle + \langle V_{x_k}, \delta x_k \rangle + a_k, \tag{5}$$

where $\langle \;\; , \;\; \rangle$ denotes the inner product of two vectors.

Using the dynamic programming technique we write:

$$\Delta V_k{}^0(\delta x_k) = \min_{\delta u_k \ldots \delta u_{N-1}} [\Delta L_k(\delta x_k, \delta u_k) + \Delta V_{k+1}(\delta x_{k+1}, \delta u_{k+1} \ldots \delta u_{N-1})] \tag{6}$$

where δx_{k+1} is, via eqn. (1), a function of δx_k and δu_k:

$$\delta x_{k+1} = \Delta f_k(\delta x_k, \delta u_k). \tag{7}$$

Minimizing with respect to $\delta u_{k+1} \ldots \delta u_{N-1}$ yields:

$$\Delta V_k{}^0(\delta x_k) = \min_{\delta u_k} [\Delta L_k(\delta x_k, \delta u_k) + \Delta V^0{}_{k+1}(\delta x_{k+1})], \tag{8}$$

where δx_{k+1} is given by eqn. (7)

For simplicity we assume initially that x_k and u_k are scalar variables and expand ΔL_k and Δf_k up to terms of second order:

$$\Delta L_k(\delta x_k, \delta u_k) = \delta L(\delta x_k, \delta u_k) + \tfrac{1}{2}\delta^2 L(\delta x_k, \delta u_k),$$

where

$$\delta L(\delta x_k, \delta u_k) = L_{x_k}\delta x_k + L_{u_k}\delta u_k,$$

$$\delta^2 L(\delta x_k, \delta u_k) = L_{xx_k}\delta x_k{}^2 + 2L_{xu_k}\delta x_k \delta u_k + L_{uu_k}\delta u_k{}^2$$

and

$$L_{x_k} = \frac{\partial L_k(x_k, u_k)}{\partial x_k},$$

$$L_{xu_k} = \frac{\partial^2 L_k(x_k u_k)}{\partial x_k \partial u_k}, \text{etc.}$$

Similarly:

$$\Delta f_k(\delta x_k, \delta u_k) = \delta f_k(\delta x_k, \delta u_k) + \tfrac{1}{2}\delta^2 f_k(\delta x_k, \delta u_k),$$

where

$$\delta f_k(\delta x_k, \delta u_k) = f_{x_k}\delta x_k + f_{u_k}\delta u_k,$$

$$\delta^2 f_k(\delta x_k, \delta u_k) = f_{xx_k}\delta x_k{}^2 + 2f_{xu_k}\delta x_k \delta u_k + f_{uu_k}\delta u_k{}^2.$$

Hence, on substituting these expansions into eqn. (8) where we put:

$$\Delta V_{k+1}{}^0(\delta x_{k+1}) = \tfrac{1}{2}V_{xx_{k+1}}\delta x_{k+1}^2 + V_{x_{k+1}}\delta x_{k+1} + a_{k+1},$$

we obtain, neglecting terms of order higher than two:

$$\begin{aligned}\Delta V_k{}^0(\delta x_k) = \min_{\delta u_k} [&(L_{x_k} + V_{x_{k+1}} f_{x_k})\delta x_k + (L_{u_k} + V_{x_{k+1}} f_{u_k})\delta u_k \\ &+ \tfrac{1}{2}(L_{xx_k} + V_{x_{k+1}} f_{xx_k})\delta x_k{}^2 + (L_{xu_k} + V_{x_{k+1}} f_{xu_k})\delta x_k \delta u_k \\ &+ \tfrac{1}{2}(L_{uu_k} + V_{x_{k+1}} f_{uu_k})\delta u_k{}^2 \\ &+ \tfrac{1}{2}V_{xx_{k+1}} f_{x_k}^2 \delta x_k{}^2 + V_{xx_{k+1}} f_{x_k} f_{u_k} \delta x_k \delta u_k \\ &+ \tfrac{1}{2}V_{xx_{k+1}} f_{u_k}^2 \delta u_k{}^2] + a_{k+1}.\end{aligned}$$

If we now introduce the Hamiltonian H_k:

$$H_k(x_k, u_k, V_{x_{k+1}}) = L_k(x_k, u_k) + V_{x_{k+1}} f_k(x_k, u_k),$$

$$\begin{aligned}\Delta V_k{}^0(\delta x_k) = \min_{\delta u_k} [&H_{x_k}\delta x_k + H_{u_k}\delta u_k \\ &+ \tfrac{1}{2}H_{xx_k}\delta x_k{}^2 + H_{xu_k}\delta x_k \delta u_k + \tfrac{1}{2}H_{uu_k}\delta u_k{}^2 \\ &+ \tfrac{1}{2}V_{xx_{k+1}} f_{x_k}^2 \delta x_k{}^2 + V_{xx_{k+1}} f_{x_k} f_{u_k} \delta x_k \delta u_k \\ &+ \tfrac{1}{2}V_{xx_{k+1}} f_{u_k}^2 \delta u_k{}^2] + a_{k+1}\end{aligned}$$

in the scalar case. When x_k and u_k are vectors it can be shown that a similar result is obtained. First we define H_k for the vector case as:

$$H_k(x_k, u_k, V_{x_{k+1}}) = L_k(x_k, u_k) + \langle V_{x_{k+1}}, f_k(x_k, u_k) \rangle. \tag{9}$$

Then

$$\Delta V_k^0(\delta x_k) = \min_{\delta u_k} [\langle H_{x_k}, \delta x_k \rangle + \langle H_{u_k}, \delta u_k \rangle + \tfrac{1}{2}\langle \delta x_k, A_k \delta x_k \rangle$$

$$+ \tfrac{1}{2}\langle \delta u_k, B_k \delta x_k \rangle + \tfrac{1}{2}\langle \delta x_k, B_k^T \delta u_k \rangle + \tfrac{1}{2}\langle \delta u_k, C_k \delta u_k \rangle] + a_{k+1}, \tag{10}$$

where

$$A_k = H_{xx_k} + f_{x_k}^T V_{xx_{k+1}} f_{x_k}, \tag{11}$$

$$B_k = H_{ux_k} + f_{u_k}^T V_{xx_{k+1}} f_{x_k}, \tag{12}$$

$$C_k = H_{uu_k} + f_{u_k}^T V_{xx_{k+1}} f_{u_k}, \tag{13}$$

where, for example, H_{x_k} is a vector:

$$H_{x_k} = \begin{Bmatrix} \dfrac{\partial H_k}{(\partial x_k)_1} \\ \cdot \\ \cdot \\ \cdot \\ \dfrac{\partial H_k}{(\partial x_k)_n} \end{Bmatrix}$$

and

$$f_{x_k} = \begin{Bmatrix} \dfrac{\partial (f_k)_1}{(\partial x_k)_1} & \cdots & \dfrac{\partial (f_k)_1}{(\partial x_k)_n} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \dfrac{\partial (f_k)_n}{(\partial x_k)_1} & \cdots & \dfrac{(\partial f_k)_n}{(\partial x_k)_n} \end{Bmatrix},$$

where $(f_k)_r$ and $(x_k)_r$ are the rth components of the vectors f_k and x_k respectively. Also

$$H_{ux_k} = \begin{Bmatrix} \dfrac{\partial^2 H_k}{(\partial u_k)_1 (\partial x_k)_1} & \cdots & \dfrac{\partial^2 H_k}{\partial (u_k)_1 (\partial x_k)_n} \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \cdot & & \cdot \\ \dfrac{\partial^2 H_k}{\partial (u_k)_m (\partial x_k)_1} & \cdots & \dfrac{\partial^2 H_k}{\partial (u_k)_m (\partial x_k)_n} \end{Bmatrix}.$$

H_{xx_k}, H_{uu_k}, f_{u_k}, H_{u_k} are similarly defined.

If the right-hand side of eqn. (10) is now minimized with respect to δu_k we obtain:

$$\delta u_k = \alpha_k + \beta_k \delta x_k, \tag{14}$$

where

$$\alpha_k = -C_k^{-1} H_{u_k}, \tag{15}$$

$$\beta_k = -C_k^{-1} B_k. \tag{16}$$

Equation (14) is the optimal incremental control law for time k. Note that δu_k as given by (14) may be too large given that terms up to second order only have been used; we will discuss this point later. If the control law (14), (15), (16) is substituted in (10) we get:

$$\Delta V_k^0(\delta x_k) = \langle (H_{x_k} + \beta_k^T H_{u_k}), \delta x_k \rangle \; + \tfrac{1}{2}\langle \delta x_k, (A_k - B_k^T C_k^{-1} B_k)\delta x_k \rangle$$
$$- \tfrac{1}{2}\langle H_{u_k}, C_k^{-1} H_{u_k}\rangle + a_{k+1}, \tag{17}$$

whence

$$V_{xx_k} = A_k - B_k^T C_k^{-1} B_k, \tag{18}$$

$$V_{x_k} = H_{x_k} + \beta_k^T H_{u_k}, \tag{19}$$

$$a_k = a_{k+1} - \tfrac{1}{2}\langle H_{u_k}, C_k^{-1} H_{u_k}\rangle. \tag{20}$$

Since

$$\Delta V_N^0(\delta x_N) = \langle L_{x_N}, \delta_{x_N}\rangle + \tfrac{1}{2}\langle \delta x_N, L_{xx_N}\delta x_N\rangle$$

the boundary conditions for (18), (19) and (20) are:

$$V_{xx_N} = L_{xx_N}, \tag{21}$$

$$V_{x_N} = L_{x_N}, \tag{22}$$

$$a_N = 0. \tag{23}$$

The procedure for optimization is as follows:

(i) A nominal control sequence $\{u_k\}$ is chosen and the state variable sequence $\{x_k\}$ calculated using eqn. (1). These sequences are stored.

(ii) The sequences $\{V_{xx_k}\}$, $\{V_{x_k}\}$ and $\{a_k\}$ are calculated in reverse time using eqns. (18), (19) and (20) together with the boundary conditions (21), (22) and (23). The incremental control law variables $\{\alpha_k\}$ and $\{\beta_k\}$ are calculated and stored. It is not necessary to store the sequences $\{V_{xx_k}\}$, $\{V_{x_k}\}$ and $\{a_k\}$.

(iii) Improved $\{x_k\}$ and $\{u_k\}$ sequences are calculated and stored using (1) and:

$$(u_k)_{\text{new}} = (u_k)_{\text{old}} + \delta u_k,$$

$$\delta u_k = \alpha_k + \beta_k[(x_k)_{\text{new}} - (x_k)_{\text{old}}].$$

(iv) Repeat (ii), etc.

The improvement obtained during one cycle of this procedure (if $\delta u_k, \delta x_k$ for all k are sufficiently small) is:

$$\Delta V_1^0(0) = a_1, \tag{24}$$

since

$$\delta x_1 = 0.$$

§ 3. Examples

The procedure can be illustrated by means of a very simple example which shows that one step convergence can be obtained if the system is linear and L_k is quadratic function. The system equations are:

$$x_{k+1} = x_k + u_k, x_1 = 1.$$

The cost function is:

$$V_1 = \left(\frac{x_1^2}{2} + \frac{u_1^2}{2}\right) + \left(\frac{x_2^2}{2} + \frac{u_2^2}{2}\right) + \frac{x_3^2}{2}\ .$$

Thus

$$f_{x_k} = f_{u_k} = 1, f_{xx_k} = f_{uu_k} = f_{ux_k} = 0,$$

$$H_{x_k} = x_k + V_{x_{k+1}}, H_{u_k} = u_k + V_{x_{k+1}}, H_{xx_k} = 1,$$

$$H_{uu_k} = 1, H_{ux_k} = 0.$$

(i) The initial control sequence is $u_1 = 0$, $u_2 = 0$.
The initial state sequence is, therefore, $x_1 = 1$, $x_2 = 1$, $x_3 = 1$.
The initial cost is $1\frac{1}{2}$.
(ii) Since

$$V_3 = \frac{x_3^2}{2}\ ; V_{x_3} = 1\ ; V_{xx_3} = 1,$$

$$A_2 = 1 + 1 = 2\ ; B_2 = 0 + 1 = 1\ ; C_2 = 1 + 1 = 2,$$

$$\alpha_2 = -\tfrac{1}{2}\ ; \beta_2 = -\tfrac{1}{2},$$

$$V_{xx_2} = 1\tfrac{1}{2}, V_{x_2} = 1\tfrac{1}{2}\ ; a_2 = -\tfrac{1}{4},$$

$$A_1 = 2\tfrac{1}{2}, B_1 = 1\tfrac{1}{2}\ ; C_1 = 2\tfrac{1}{2},$$

$$\alpha_1 = -\tfrac{3}{5}\ ; \beta_1 = -\tfrac{3}{5},$$

$$a_1 = -\tfrac{1}{4} - \tfrac{9}{20} = -\tfrac{7}{10}.$$

(iii) The improved control laws are:

$$u_1 = 0 - \tfrac{3}{5} - \tfrac{3}{5}(x_1 - 1) = -\tfrac{3}{5}x_1,$$

$$u_2 = 0 - \tfrac{1}{2} - \tfrac{1}{2}(x_2 - 1) = -\tfrac{1}{2}x_2,$$

whence

$$u_1 = -\tfrac{3}{5}\ ; x_2 = \tfrac{2}{5}\ ; u_2 = -\tfrac{1}{5}\ ; x_3 = \tfrac{1}{5}.$$

The new cost is $V_1 = \frac{4}{5} = 1\frac{1}{2} - \frac{7}{10}$.

This is the optimal cost and $u_1 = -\frac{3}{5}$ and $u_2 = -\frac{1}{5}$ are the optimal controls.

A more complicated example is a discrete-time version of a problem considered by Fuller (1964). The original problem does not cost control, but the control is constrained in magnitude. To make the application of the method described in this paper possible the problem is restated as follows.

The system equations are:

$$x_{k+1} = x_k + hs(u_k),$$

$$y_{k+1} = y_k + h \, . \, x_k + \frac{h^2}{2} s(u_k),$$

where

$$s(u_k) = u_k \text{ if } |u_k| \leqslant B,$$

$$s(u_k) = 1 - (1-B) \exp\left\{\frac{-u_k + B}{1-B}\right\} \text{ if } u_k > B,$$

$$s(u_k) = -1 + (1-B) \exp\left\{\frac{u_k + B}{1-B}\right\} \text{ if } u_k < -B.$$

h is the sampling interval. As h tends to zero and B tends to unity the above equations tend to:

$$\dot{x} = \text{sat}\,(u),$$

$$\dot{y} = x,$$

where

$$\text{sat}\,(u) = u \text{ if } |u| \leqslant 1,$$

$$\text{sat}\,(u) = 1 \text{ if } u > 1,$$

$$\text{sat}\,(u) = -1 \text{ if } u < -1.$$

The cost function is:

$$V = \sum_{k=1}^{J} \left[\frac{h}{2} (T \, . \, u_k{}^2 + y_{k+1}{}^2)\right].$$

As h and T tend to zero V tends to:

$$V = \int_0^{t_f} \tfrac{1}{2} y^2 \, dt,$$

where

$$t_f = J \, . \, h.$$

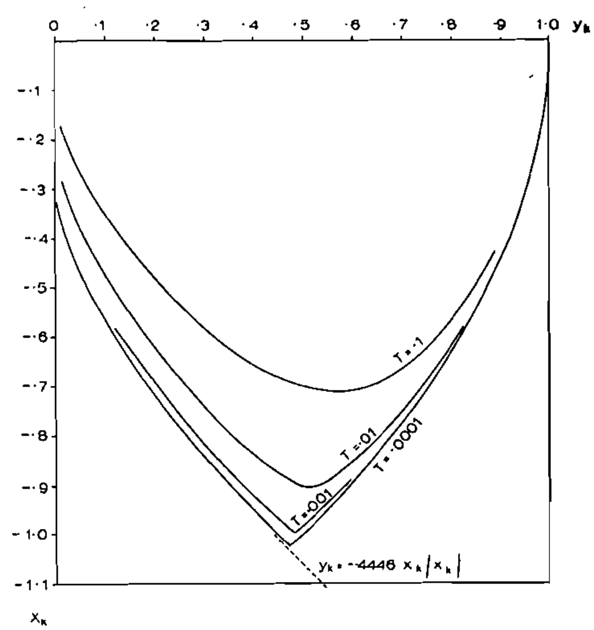
Some calculations were performed with $J = 1000, h = 0{\cdot}01$ (i.e. $t_f = 10$) and $B = 0{\cdot}5$ for the following values of T: 0·1, 0·01, 0·001, 0·0001. The results are shown in the figure. Note as $T \to 0$ the trajectory tends to the continuous-time solution for which the switching curve, as has been shown by Fuller, is:

$$y = -0{\cdot}4446 x |x|$$

for the case $t_f = \infty$.

The calculation was also performed for the case $J=1000$, $h=0\cdot01$, $B=0\cdot9$ and $T=0\cdot0001$. The trajectory for this case approaches the switching curve even more closely, the coordinates at $k=104$ being $x_k=0\cdot4696, y_k=-1\cdot027$ the corresponding point on the switching curve being $x=0\cdot4689, y=-1\cdot027$. However, although the method has successfully converged for a problem of the switching type, it must be appreciated that the method is more suited for problems where the cost is a smooth function of the control and state sequences and therefore locally approximated more closely by quadratic functions. In the switching problem the optimum cost function is such that local approximation at the switching surface is difficult. This difficulty is reflected in the number of iterations required to obtain the optimal solution:

(*a*) $B=0\cdot5, T=0\cdot1$: 5 iterations,
(*b*) $B=0\cdot5, T=0\cdot01$: 6 iterations,
(*c*) $B=0\cdot5, T=0\cdot001$: 10 iterations,
(*d*) $B=0\cdot5, T=0\cdot0001$: 20 iterations,
(*e*) $B=0\cdot9, T=0\cdot0001$: 43 iterations.



The choice of ϵ (see § 4) proved very critical in case (*e*), and it was found necessary to have a few trial evaluations of the cost with different ϵ for each iteration. In some iterations it was also found that $C_k(H_{uu}$ in the continuous-time problem) was negative. Rather than resort to first order methods the simple stratagem of using the absolute value of C_k was employed. With ϵ small enough this guarantees an improvement with each iteration.

§ 4. Convergence of the Algorithm

In the linear example δu_k and δx_k are large. This is possible because of the global properties of linear systems. In a non-linear problem it is necessary to limit the magnitude of these variations as otherwise the expansions will be inaccurate. One way of doing so is to modify eqn. (14) to:

$$\delta u_k = \epsilon \alpha_k + \beta_k \delta x_k, \tag{25}$$

where

$$0 < \epsilon \leqslant 1.$$

If $\epsilon = 0$ then, since $\delta x_1 = 0$, $\delta u_k = \delta x_k = 0$ for all k. If eqn. (25) is substituted into (10) eqn. (17) is modified to:

$$\begin{aligned} \Delta V_k^0(\delta x_k) = \langle (H_{x_k} + \beta_k^T H_{u_k}), \delta x_k \rangle \\ + \tfrac{1}{2} \langle \delta x_k, (A_k - B_k^T C_k^{-1} B_k) \delta x_k \rangle \\ - \left(\epsilon - \frac{\epsilon^2}{2} \right) \langle H_{u_k}, C_k^{-1} H_{u_k} \rangle, \end{aligned} \tag{17 a}$$

so that the difference eqns. (18) and (19) for V_{xx_k} and V_{x_k} are not altered, but eqn. (20) becomes:

$$a_k = a_{k+1} - \left(\epsilon - \frac{\epsilon^2}{2} \right) \langle H_{u_k}, C_k^{-1} H_{u_k} \rangle. \tag{20 a}$$

The total improvement during one cycle is:

$$a_1 = -\epsilon \left(1 - \frac{\epsilon}{2} \right) \sum_{k=1}^{N-1} \langle H_{u_k}, C_k^{-1} H_{u_k} \rangle. \tag{26}$$

Thus provided ϵ is chosen so that δu_k and δx_k are small enough for all k to justify the expansions, C_k is positive definite, and solutions to the V_{x_k} and V_{xx_k} difference equations exist, the algorithm improves the trajectory with each cycle as a_1 is then negative.

§ 5. First-order Method

Expansions up to first-order terms only are made. Equation (10) becomes:

$$\Delta V_k^0(\delta x_k) = \min_{\delta u_k} [\langle H_{x_k}, \delta x_k \rangle + \langle H_{u_k}, \delta u_k \rangle] + a_{k+1}.$$

Since δu_k must be limited we put:

$$\delta u_k = -\epsilon H_{u_k},$$

where ϵ is a suitably chosen small positive number. Hence

$$V_{x_k} = H_{x_k}, \tag{27}$$

$$a_k = a_{k+1} - \epsilon \langle H_{u_k}, H_{u_k} \rangle, \tag{28}$$

whence

$$a_1 = -\epsilon \sum_{k=1}^{N-1} \langle H_{u_k}, H_{u_k} \rangle \tag{29}$$

is the improvement per cycle.

§ 6. Continuous-time Systems

Consider the system:

$$\dot{x} = f(x, u, t).$$

The cost function is:

$$V(t_0) = \int_{t_0}^{t_f} L(x, u, t)\, dt + L_f(x(t_f)).$$

Divide the interval t_0, t_f into N intervals of duration Δ. Let u be piecewise during each interval. Provided Δ is sufficiently small:

$$x_{k+1} = x_k + f_k \Delta,$$

where $x_k = x(k\Delta), f_k = f(x_k, u_k, k\Delta)$. Thus in place of f_k in the discrete-time equations we substitute $x_k + f_k\Delta$. We write this substitution: $f_k \to x_k + f_k\Delta$. Similarly the following substitutions are made:

$$f_k \to x_k + f_k\Delta; \quad f_{x_k} \to I + f_{x_k}\Delta; \quad f_{u_k} \to f_{u_k}\Delta;$$

$$L_k \to L_k\Delta; \quad V_{x_k} \to V_{x_k}; \quad V_{xx_k} \to V_{xx_k}; \quad a_k \to a_k;$$

$$H_k \to L_k\Delta + \langle V_{x_{k+1}}, x_k \rangle + \langle V_{x_{k+1}}, f_k \rangle \Delta = H_k\Delta + \langle V_{x_{k+1}}, x_k \rangle,$$

where, for continuous-time systems:

$$H(x, u, V_x, t) = L(x, u, t) + \langle V_x, f(x, u, t) \rangle$$

$$H_{u_k} \to H_{u_k}\Delta; H_{xx_k} \to H_{xx_k}\Delta; H_{uu_k} \to H_{uu_k}\Delta; H_{ux_k} \to H_{ux_k}\Delta; C_k \to H_{uu_k}\Delta;$$

$$\alpha_k \to \alpha; \ \beta_k \to \beta, \ H_{x_k} \to H_{x_k}\Delta + V_{x_{k+1}};$$

where, for continuous-time system:

$$\alpha = -[H_{uu}^{-1}]H_u, \quad \beta = -[H_{uu}^{-1}][H_{ux} + f_u^T V_{xx}].$$

$$\therefore \quad V_{x_k} = H_{x_k}\Delta + V_{x_{k+1}} + \beta^T H_u \Delta.$$

Allowing $\Delta \to 0$ gives:

$$-\frac{d}{dt} V_x = H_x + \beta^T H_u. \tag{30}$$

Similarly:

$$V_{xx_k} = H_{xx_k}\Delta + V_{xx_{k+1}} + f_{x_k}{}^T V_{xx_{k+1}}\Delta + V_{xx_{k+1}} f_{x_k} \cdot \Delta - B_k{}^T [H_{uu_k}{}^{-1}] B_k \Delta.$$

Allowing $\Delta \to 0$ gives:

$$-\frac{d}{dt} V_{xx} = H_{xx} + f_x{}^T V_{xx} + V_{xx} f_x - [H_{ux} + f_u{}^T V_{xx}]^T [H_{uu}{}^{-1}][H_{ux} + f_u{}^T V_{xx}]. \tag{31}$$

Also

$$a_k = a_{k+1} - \epsilon \left(1 - \frac{\epsilon}{2}\right) \langle H_{u_k}, C_k{}^{-1} H_{u_k} \rangle \Delta,$$

which tends to:

$$-\frac{da}{dt} = -\epsilon \left(1 - \frac{\epsilon}{2}\right) \langle H_u, H_{uu}{}^{-1} H_u \rangle. \tag{32}$$

§ 7. Conclusion

An interesting feature of the derivation is the fact that the equation for V_{x_k} differs from that used in the first order methods due to the inclusion of the term $\beta_k{}^T H_{uk}$. Derivations based on the calculus of variations approach use for first-order equation for V_{x_k} and therefore have to introduce an extra set of vector difference equations (of the same dimensions as x). In the method described in the paper the differential equation for x (an n-vector) has to be integrated in forward time, and differential equations for V_x (an n-vector) and V_{xx} (an $n \times n$ symmetric matrix) in reverse time. The method described by Mitter requires the reverse time integration of λ (an n-vector), l (an n-vector) and K (an $n \times n$ symmetric matrix). The method due to Merriam is similar and requires the integration of the same number of equations. Hence the algorithm proposed here requires fewer equations. However the main reason for using this approach is that it is more direct than the calculus of variations approach and facilitates the study of stochastic systems, which will be discussed in a future paper. The algorithm possesses second order convergence near the optimum.

References

Bryson, A. E., and Denham, W., 1962, *J. appl. Mech.*, **29**, 247.

Merriam, III, C. W., 1964, *Optimisation Theory and the Design of Feedback Control Systems* (McGraw-Hill).

Mitter, S. K., 1965, Ph.D. Thesis, University of London.

Bellman, R., and Dreyfus, S. E., 1962, *Applied Dynamic Programming* (Princeton University Press).

Fuller, A. T., 1964, *J. Electr. Contr.*, **17**, 301.