Journal of the Nigerian Association of Mathematical Physics
Volume **12 (May, 2008), 285 – 296**

On the control of pure inertia plant using a hybrid of sequential variation of extremals and invariant costate imbedding algorithms.

[1]T. A. Adewale, [2]F. M. Aderibigbe
[1]Department of Industrial Mathematics,
Adekunle Ajasin University,
Akungba-Akoko, Ondo State Nigeria.
[2]Department of Mathematical Sciences,
University of Ado-Ekiti,
Ado-Ekiti, Ekiti State, Nigeria

Abstract

This paper presents the control of pure inertia plant using successive variation of extremals algorithm with invariant costate imbedding. Numerical experimentation is provided by the problem of sliding mass. The problem is solved by taking advantage of the Hamiltonian formalism via the construction of influence function matrices.

***Keywords*:** Hamiltonian functional, necessary conditions for optimality, nonlinear systems, Pontryagin's maximum principle or Weierstrass conditions, state-costate variables, invariant imbedding.

1.0 Introduction

The optimal control of a pure inertia plant has received much attention in the literature [8] especially a plant governed by the system.

$$\ddot{X} = M \tag{1.1}$$

Where M is the controlled force applied, X is the displacement. The early attention accorded this system is due, of course, to the simple nature of the plant.

The current interest in the control of spacecraft outside the earth's atmosphere now gives new and practical meaning to this type of problem. Considering the above reasons, the problem of pure inertia plant will be considered here as an example involving controls of variable inequality constraint. We stated in [1] that, although the method shows that Pontryagin's maximum principle does indeed quickly yield the form of the expected solution as often stated, the complete solution is somewhat evasive. In the previous paper [1] we employed the invariant imbedding method. The invariant imbedding method converts a given two point nonlinear boundary value problem to its equivalent initial value problems [9].

In what follows we employ the successive or sequential variation of extremals algorithm with the explicit knowledge of the initial costate vector found in [1]. We appreciate that there could be an improvement in the initial costate vector determined in equations (10) – (17) in [1] and we therefore set out to use this in the variation of externals algorithm so that if the terminal or the resulting final costate $\underline{\lambda}(t_f) = \underline{0}$ obviously our guess of the initial costate vector was correct and we have solved the problem.

We started by considering the problem of minimizing.

e-mail: timothyadewale 2005@yahoo.com
Phone: 08062472545

$$J = \int_{t_0}^{t_f} g(\underline{x}, \underline{u}, t)dt \tag{1.2}$$

Subject to the dynamical constraints

$$\dot{\underline{x}} = \underline{f}(\underline{x}, \underline{u}, t) \tag{1.3}$$

where the states and costates are not constrained by any boundaries, the final time t_f is fixed and the final state $x(t_f)$ is free. To write the necessary conditions for optimality, let us define the Hamiltonian functional as usual [5-8]:

$$H = g(\underline{x}, \underline{u}, t) + \lambda^T(t) f(\underline{x}, \underline{u}, t) \tag{1.4}$$

so that the necessary conditions become

$$\dot{\underline{x}}(t) = \frac{\partial H}{\partial \underline{\lambda}} = g(\underline{x}, \underline{u}, t) \tag{1.5}$$

$$\dot{\underline{\lambda}} = -\frac{\partial H}{\partial \underline{x}} = -\left[\frac{\partial g}{\partial \underline{x}}(\underline{x}, \underline{u}, t)\right]^T \underline{\lambda}(t) - \frac{\partial f}{\partial \underline{\lambda}}(\underline{x}, \underline{u}, t) \tag{1.6}$$

$$\frac{\partial H}{\partial \underline{u}} = 0 \ \ Or \left[\frac{\partial g}{\partial \underline{u}}(\underline{x}, \underline{u}, t)\right]^T \underline{\lambda}(t) + \frac{\partial f}{\partial \underline{u}}(\underline{x}, \underline{u}, t) = \underline{0} \tag{1.7}$$

$$\underline{x}(t_0) = \underline{x_0} \tag{1.8}$$

$$\underline{\lambda}(t_f) = \underline{0} \tag{1.9}$$

Solving equation (1.7) for $\underline{u}$ and substituting this $\underline{u}$ in terms of $\underline{x}, \underline{\lambda}$ into equations (1.5) and (1.6) yields the reduced differential equations of the form

$$\dot{\underline{x}} = \underline{\phi}_1(\underline{x}(t), \underline{\lambda}(t), t) \tag{1.10}$$

$$\dot{\underline{\lambda}} = \underline{\phi}_2(\underline{x}(t), \underline{\lambda}(t), t) \tag{1.11}$$

Therefore, to solve our optimization problem we need to solve equations (1.10) and (1.11) subject to the split boundary conditions given by equations (1.8) and (1.9). We recall that in the invariant imbedding method considered previously we determined the initial costate vector explicitly so that in any intermediated iteration the nonlinear differential equations (1.10) and (1.11) were satisfied by the initial state and the final costate vector that will make the pair $(\underline{x}(t), \underline{u}(t))$ admissible at every iteration and by some iterative technique due to Newton Raphson. The intention is to improve upon $\underline{\lambda}(t_0)$ determined previously and employed this in integrating the equations (1.5) and (1.6)/ We shall pretend that the final costate vector $\underline{\lambda}(t_f)$ is unknown and that it is a function of $\underline{\lambda}(t_0)$ and t and use $\underline{\lambda}(t_0)$ determined before, namely $\underline{\lambda}(t_0) = \left(0, \frac{1}{3}\right)^T$ as initial guess to compute the tangent t to the unknown $\underline{\lambda}(t_f)$ versus $\underline{\lambda}(t_0)$ relationship and use as the next guess for $\underline{\lambda}(t_0)$ the extrapolated point where the tangent intersects the $\underline{\lambda}(t_f)$ -axis, that is, where $\underline{\lambda}(t_f)$ is zero. The equation of the tangent for the scalar case, [by taking two arbitrary points, $\left(\lambda_0(t_0), \lambda_0(t_f)\right), \left(\lambda(t_0)\lambda(t_f)\right)$ is

$$\lambda(t_f) = \beta\lambda(t_0) + \left\lfloor \lambda_0(t_f) - \beta\lambda(t_0) \right\rfloor \tag{1.12}$$

So, for the scalar case, the equation

$$\lambda_{k+1}(t_0) = \lambda_k(t_0) - \left[\frac{\partial \lambda_k(t_f)}{\partial \lambda(t_f)}\right]^{-1} \lambda_k(t_0) \tag{1.13}$$

holds. The equation for the multivariable case (n-state and n costate) equations can be written as

$$\underline{\lambda}_{k+1}(t_0) = \lambda_k(t_0) - \left[P_\lambda\left(\underline{\lambda}_k(t_0)t_f\right)\right]^{-1} \underline{\lambda}_k(t_f) \tag{1.14}$$

where, the matrix,

$$P_\lambda(\underline{\lambda}_k(t_0),t) = \left(\frac{\partial \lambda_i(t)}{\partial \lambda_J(t_0)}\right)_{\lambda(t_0)}, \; i, j, = 1,2,\Lambda\ ,n \tag{1.15}$$

which is called the costate influence function matrix because it indicates the influence of changes in the initial costate trajectory at time *t*. In equation (1.14) we need to know P_λ only at the terminal time t_f. Equation (1.15) applies to the case where there is no termination time weighting them in the cost functional. If the termination time weighting term is included, that is, if the cost functional is of the form,

$$J = h\left(x(t_f)) + \int_{t_0}^{t_f} g(\underline{x},\underline{u},t)dt\right) \tag{1.16}$$

then the initial costate vector $\underline{\lambda}(t_0)$ can be improved using the relation

$$\underline{\lambda}_{k+1}(t_0) = \underline{\lambda}_k(t_0) - \left\{\left[\left[\frac{\partial^2 h}{\partial x^2}\left(x(t_f)\right)\right]P_x\left(\underline{\lambda}(t_0),t_f\right) - P_\lambda\left(\underline{\lambda}(t_0),t_f\right)\right]\right\}^{-1}\left[\underline{\lambda}(t_f) - \frac{\partial h}{\partial x}(x(t_f))\right]_k \tag{1.17}$$

where $[\cdot]_k$ implies that the enclosed terms are evaluated on the k-th trajectories and $P_x(\lambda_k(t_0),t_f)$ is then n x n state influence function matrix defined by:

$$P_x(\lambda_k(t_0),t) = \left(\frac{\partial x_i(t)}{\partial \lambda_j(t_0)}\right)\ i, j = 1, 2, \mathrm{K}\ , n \tag{1.18}$$

Where $\frac{\partial^2 h}{\partial x^2}(x(t_f))$ is the matrix whose i_{p-th} element is

$$\left[\frac{\partial^2 h}{\partial x^2}(x(t_f))\right]_{2p} = \frac{\partial^2 h(x(t_f))}{\partial x_p \partial x_p} \tag{1.19}$$

2.0 Determination of the influence function matrix

From the foregoing we see that the most important step in the successive variation of extremals method is the computation of the influence function matrices, P_x and P_λ. In what follows we describe an easy way of computing these on a digital computer.

2.1 Numerical Experimentation

Consider the sliding mass system shown in the figure below [6, 7, 10]

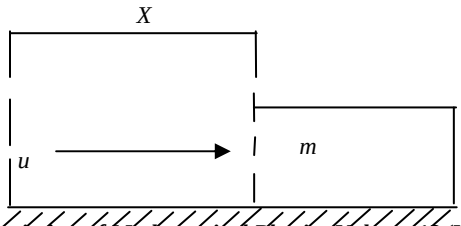


Figure 2.1: Sliding mass system

This system can be described by the second-order differential equation

$$m\ddot{x} + c\dot{x}^2 \operatorname{sgn} \dot{x} = u \tag{2.1}$$

$$\operatorname{sgn} \dot{x} = \begin{cases} 1, \dot{x} > 0 \\ -1, \dot{x} < 0 \end{cases} \tag{2.2}$$

where m is the mass of the sliding solid, c the coefficient of friction, u is the controlled force applied and x is the displacement.

We shall take $m = 1$, $c = 1$. Hence we can reduce equation (2.1) to two first-order equations and solve as in [4, 8]. The result obtained by applying the invariant imbedding algorithm is $\underline{\lambda}(t_0) = \left(0, \frac{1}{3}\right)^T$. We shall use this as the initial guess in steps of the above algorithm We shall minimize the cost functional

$$J = \int \frac{1}{2}\left(x_1^2 + x_2^2 + u^2\right) dt \tag{2.3}$$

equation (2.1) becomes

$$\ddot{x} + \dot{x}^2 \operatorname{sgn} x = u \tag{2.4}$$

Letting $x_1 = x$, then $x_2 = \dot{x}_1 = \dot{x}$ we have

$x_1 = \dot{x}$ then $x_2 = \dot{x}_1 = \ddot{x}$ we have

$$\dot{x}_1 = x_2 = f_1 \tag{2.5}$$

$$\dot{x}_2 = -x_2 + m = f_2 \tag{2.6}$$

as equivalent two first-order equations. That is, the quantity x is taken to be a system error, then x_1 and x_2 are the error and rate respectively. For the end conditions we have:

$$\begin{aligned} &t = t_1 = 0, t = t_f \text{ (unspecified)} \\ &x_1(t_1) = x_{1_0},\ x_1(t_1) = 0 \\ &x_2(t_1) = x_{2_0},\ x_2(t_1) = 0 \end{aligned} \tag{2.7}$$

we shall need to determine the control u required to derive the error and error rate to zero while minimizing the performance index or cost functional J of equation (2.3) which is obviously dependent on the amount of control u used. This example was previously solved by Bauman [3], using a hierarchical technique and subsequently by Galy as cited in [9], Singh et al by quasilinerization technique. The problem now reduces to that of minimizing the time required to transfer the system between initial and final states. The system performance index given by equation (2.3) is then minimized subject to

$$\left.\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u - u_2^2 \operatorname{sgn} x_2 \end{aligned}\right\} \tag{2.8}$$

The Hamiltonian can be written as:

$$H = \frac{1}{2}\left(x_1^2 + x_2^2 + u^2\right) + \lambda_1 x_2 + \lambda_2\left(u - x_2^2 \operatorname{sgn} x_2\right) \tag{2.9}$$

Then the conditions of optimality yield the two point boundary value problem (bvp):

$$\left.\begin{aligned} \dot{x}_1 &= x_2 \\ \dot{x}_2 &= u - x_2^2 \operatorname{sgn} x_2 \\ \dot{\lambda}_1 &= x_1 \\ \dot{\lambda}_2 &= x_2 - \lambda_1 + 2\lambda_2 x_2 \operatorname{sgn} x_2 \end{aligned}\right\} \tag{2.10}$$

Substituting $u = \lambda_2$ renders the second of equations (2.10) into

$$\dot{x}_2 = \lambda_2 - x_2^2 \operatorname{sgn} x_2 \tag{2.11}$$

and the reduced Hamiltonian H^* is given by

$$H^* = \frac{1}{2}\left(X_1^2 + \lambda_2^2\right) + \lambda_1 X_2 + \lambda_2\left(\lambda_2 - X_2^2 \operatorname{sgn} x_2\right), \mathrm{K} \tag{2.12}$$

whence we obtain,

$$\begin{aligned} \lambda_{10}(0) &= 0 \\ \lambda_{20}(0) &= \frac{1}{3} \\ \underline{\lambda}(t_0) &= (\lambda_{10}, \lambda_{20}) \end{aligned} \tag{2.13}$$

in [1] which is the initial costate vector. We shall next improve upon this using $\lambda_{10}(0) = 0$, $\lambda_{20}(0) = \frac{1}{3}$ as initial guess in the iterative scheme defined by equation (1.17) to obtain:

$$\underline{\lambda}_{k+1} = \underline{\lambda}_k(t_0) + \left\{\left[\frac{\partial^2 h(\underline{x}(t_f))}{\partial x^2}\right] P_x\left(\underline{\lambda}_k(t_0), t_f\right) - P_\lambda(\lambda_k(t_0), t_f)\right\}^{-1} \left\{\underline{\lambda}(t_f) - \frac{\partial h}{\partial x}(x(t_f))\right\} \tag{2.14}$$

or iterative scheme defined by equation (1.14) to obtain

$$\underline{\lambda}_{k+1}(t = \underline{\lambda}(t_0)\lambda(t_f) - \left[P_\lambda\left(\underline{\lambda}_k(t_0)t_f\right)\right]^{-1} \lambda\left(t_f\right) \tag{2.15}$$

where $\boldsymbol{P_\lambda}$ is as defined in (1.15). In order to integrate the resulting *bvp* using an R-K-4 type method we convert the problem into a four-dimensional problem.

3.0 Discussion of computational results

Equation (1.5) through (1.7) are nonlinear and cannot be solved in analytical closed form we therefore have to seek refuge in numerical and analytical approximation methods. From SVEICI, we obtain convergence to $\underline{\lambda}(t_0) = (4.03 \times 10^{-6}, 4.3 \times 10^{-9})^T$ in seven iterations. This is an improvement in the value of $\underline{\lambda}$ which is expected to be $(0,0)^T$ We next use this in the Evans and Sangui's algorithm [11] employed in integrating the *bvp* resulting from optimality conditions and represented by the recursive relation defined by

$$\underline{z}_{k=1} = z_k + \tfrac{h}{3}\left(\sqrt{k_1 k_2} + \sqrt{k_2 k_3} + \sqrt{k_3 k_4} + \sqrt{k_1 k_4}\right) \tag{3.1}$$

where h = 0.001 or $10^{-3}\underline{z} = (x_1, x_2, x_3, x_4)$, $\lambda_1 = x_3, \lambda_2 = x_4$ and let $\underline{z}_i = x_i, i = 1,2,3,4$, $\underline{f} = (f_1, f_2, f_3, f_4)$ and

$$\begin{aligned} k_1 = f(t_k, \underline{z}_k),\ k_2 = f\left(t_k + \frac{1}{2}h, z_k + \frac{1}{2}hk_1\right),\ k_3 = f\left(t_k + \frac{1}{2}h, z_k + \frac{h}{16}[k_1 + 9k_2]\right) \\ k_4 = f\left(t_k + h,\ \underline{z}_k + \tfrac{h}{24}[-3k + 5k_2 + 22k_3]\right) \end{aligned} \tag{3.2}$$

and for the problem under consideration we shall take the time step $\partial t = h = 10^{-3}$ and

$$\underline{z}_{ik+1} = \underline{z_{ik}} + \tfrac{1}{3}h\left(\sqrt{k_{i1}k_{i2}} + \sqrt{k_{i2}k_{i3}} + \sqrt{k_{i3}k_{i4}} + \sqrt{k_{i1}k_{i4}}\right) \tag{3.3}$$

where $k_{i1} = f_i(t_k, \underline{z}_{ik})$, $k_{i2} = f_i\left(t_k + \frac{1}{2}h, \underline{z}_k + \frac{1}{2}hk_{i1}\right)$,

$$k_{i3} = f_i\left(t_k + \frac{1}{2}h, \underline{z}_{ik} + \frac{h}{16}[-k_{i1} + 9k_{i2}]\right),$$

$$k_{i4} = f_i\left(t_k + h, \underline{z}_{ik} + \frac{h}{24}[-3k_{i1} + 5k_{i2} + 22k_{i3}]\right), \tag{3.4}$$

where $i = 1,2,3,4,\ k = 0,1,\text{K}\ ,n-1$

$$x_1 = 2,\ x_2 = -2,\ x_3 = 4.03\,\text{x}\,10^{-6},\ x_4 = 4.3\,\text{x}\,10^{-9} \tag{3.5}$$

$$f_1 = x_2,\ f_2 = x_4 - x_2^2 \operatorname{sgn} x_2,\ f_3 = x_1,\ f_4 = x_2 - x_3 + 2x_4x_2 \operatorname{sgn} x_2, \tag{3.6}$$

The results obtained are displayed in the Table 3.3. The iterations took 4 seconds to execute.

Table 3.1: The sliding mass problem: Component trajectories

T_k	$X(T)$	$Y_1(I)$	$Z(I)$	$W(I)$
.001	2.001998	-1.995676	0	0.333
.002	2.003991	-1.991369	1.999167 x 10^{-3}	0.3336696
.003	2.00598	-1.982803	4.000335 x 10^{-3}	0.3343371
.004	2.007965	1.987077	8.00865 x 10^{-2}	0.3350025
.005	2.009946	-1.978544	1.001579 x 10^{-2}	0.33698271
.006	2.011923	-1.974302	1.202491 x 10^{-2}	0.3369864
.007	2.0113895	-1.970076	1.403601 x 10^{-2}	0.3376437
.008	2.015863	-1.961671	1.604909 x 10^{-2}	0.3382991
.009	2.017827	-1.965866	1.80441 x 10^{-2}	0.3389525
.010	2.019786	-1.957493	2.008114 x 10^{-2}	0.339604
.011	2.021742	-1.949783	2.40011 x 10^{-2}	0.3402537
.012	2.023693	-1.945051	2.614392 x 10^{-2}	0.33409014
.013	2.02564	-1.940934	2.816875 x 10^{-2}	0.3415474
.014	2.027583	-1.936833	3.019553 x 10^{-2}	0.3421915
.015	2.029522	-1.932742	3.222424 x 10^{-2}	0.3428339
.016	2.031466	-1.928676	3.425489 x 10^{-2}	0.3434745
.017	2.033387	-1.924621	3.628748 x 10^{-2}	0.3441134

.018	2.035314	-1.92058	3.832199	0.3447506
.019	2.037236	-1.916554	4.035843×10^{-2}	0.3453861
.020	2.039155	-1.912543	4.239678×10^{-2}	0.3460199

Table 3.2 gives the decrease of the norm of each component trajectory between two successive iterations. The algorithm for system of non linear equations of the form

$$\dot{z}_1 = f_1(t, z_1, z_2, z_3, z_4)$$
$$\dot{z}_2 = f_2(t, z_1, z_2, z_3, z_4)$$
$$\dot{z}_3 = f_3(t, z_1, z_2, z_3, z_4)$$
$$\dot{z}_4 = f_4(t, z_1, z_2, z_3, z_4)$$

i.e. $$\dot{\underline{z}} = \underline{f}(t, \underline{z}),\ \underline{z}(t^{(0)}) = \underline{z}^{(0)},\ \underline{f} = (f_1, f_2, f_3, f_4)^T,\ \dot{\underline{z}} = (\dot{z}_1, \dot{z}_2, \dot{z}_3, \dot{z}_4)^T$$
$$\underline{z} = z = (z_1, z_2, z_3, z_4)^T$$

Table 3.2: Decrease of norm of component trajectories between successive iterates

	X(T)	***Y_1(I)***	***W(I)***	***Z(I)***
$\lvert Z_1 - Z_0 \rvert$	3.97×10^{-6}	1.86×10^{-5}	3.99×10^{-6}	4.48×10^{-7}
$\lvert Z_2 - Z_1 \rvert$	3.96×10^{-6}	1.84×10^{-5}	4.01×10^{-6}	4.46×10^{-7}
$\lvert Z_3 - Z_2 \rvert$	3.94×10^{-6}	1.83×10^{-5}	4.01×10^{-6}	4.43×10^{-7}
$\lvert Z_4 - Z_3 \rvert$	3.92×10^{-6}	1.81×10^{-5}	4.02×10^{-6}	4.40×10^{-7}
$\lvert Z_5 - Z_4 \rvert$	3.90×10^{-6}	1.40×10^{-5}	4.01×10^{-6}	4.44×10^{-7}
$\lvert Z_6 - Z_5 \rvert$	3.80×10^{-6}	1.38×10^{-5}	4.03×10^{-6}	2.5×10^{-7}
$\lvert Z_7 - Z_6 \rvert$	3.70×10^{-6}	1.36×10^{-5}	4.04×10^{-6}	4.3×10^{-8}
$\lvert Z_8 - Z_7 \rvert$	3.60×10^{-6}	1.35×10^{-5}	4.05×10^{-6}	4.3×10^{-9}
$\lvert Z_9 - Z_8 \rvert$	3.50×10^{-6}	1.34×10^{-5}	4.06×10^{-6}	4.3×10^{-9}
$\lvert Z_{10} - Z_9 \rvert$	3.40×10^{-6}	1.33×10^{-5}	4.05×10^{-6}	4.3×10^{-9}

$$Z = (X, Y, W, Z,)^T$$

Table 3.3: Minimization of the Cost Functional

Iterative Step	**Cost functional ($J \times 10^{-11}$)**
1	3.714764
2	3.713646

3	3.691509
4	3.668900
5	3.5221460
6	3.264780
7	3.186122
8	3.118502
9	1569398
10	1.496917

$$k_{11} = f_1(t^{(k)}\underline{z}^{(k)}) = f_1(t^{(k)}, z_1^{(k)}, z_2^{(k)}, z_3^{(k)}, z_4^{(k)}),$$

$$k_{12} = f_1\left(t^{(k)} + \frac{1}{2}h, z_1^{(k)} + \frac{1}{2}hk_{11}, z_2^{(k)} + \frac{1}{2}hk_{11}, z_3^{(k)} + \frac{1}{2}hk_{11}, z_4^{(k)}\frac{1}{2}hk_{11}\right)$$

$$k_{13} = f_1\left(t^{(k)} + \frac{1}{2}h, z_1^{(k)} + \frac{1}{16}(hk_{11} + 9k_{12}), z_2^{(k)} + \frac{h}{16}(k + 9k_{12}), z_3^{(k)} + \frac{h}{16}(k_{11} + 9k_{12}), z_4^{(k)} + \frac{h}{16}(k_{11} + 9k_{12})\right)$$

$$k_{14} = f_1\left(t^{(k)} + h, z_1^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}),\ z_2^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}),\right.$$

$$\left. z_3^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}),\ z_4^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13})\right)$$

For the problem under consideration, $f_1 = z_2 \Rightarrow k_{11} = f_1\left(t^{(k)}, z_1^{(k)}, z_2^{(k)}, z_3^{(k)}, z_4^{(k)}\right) = z_2^{(k)}$

$$k_{12} = f_1\left(t^{(k)} + \frac{1}{2}h,\ z_2^{(k)} + \frac{1}{2}hk_{11}\right) = z_2^{(k)} + \frac{1}{2}hk_{11} = z_2^{(k)} + \frac{1}{2}hz_2^{(k)}$$

$$k_{13} = f_1\left(t^{(k)} + \tfrac{1}{2}h, z_2^{(k)} + \tfrac{h}{16}(k_{11} + 9k_{12})\right) = z_2^{(k)} + \tfrac{h}{16}(k_{11} + 9k_{12})$$

$$= z_2^{(k)} + \frac{h}{16}[z_2^{(k)} + 9(z_2^{(k)} + \frac{1}{2}hz_2^{(k)})] = z_2^{(k)} + \frac{h}{16}[z_2^{(k)} + 9(z_2^{(k)} + \frac{9}{2}hz_2^{(k)}] = z_2^{(k)} + \frac{h}{16}[10z_2^{(k)} + 4.5hz_2^{(k)}]$$

$$= z_2^{(k)} + \frac{10}{16}hz_2^{(k)} + \frac{4.5}{16}h^2z_2^{(k)} = (1 + \frac{10}{16}h + \frac{9}{32}h^2)z_2^{(k)} = (1 + \frac{5}{8}h + \frac{9}{32}h^2)z_2^{(k)}$$

$$k_{14} = f_1(t^{(k)} + h, z_2^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13})) = z_2^{(k)} + \frac{h}{24}(-3z_2^{(k)} + 5(z_2^{(k)} + \frac{1}{2}hz_2^{(k)})$$

$$= z_2^{(k)} + \frac{h}{24}[-3z_2^{(k)} + 5(z_2^{(k)} + \frac{1}{5}hz_2^{(k)} + 22(1 + \frac{5}{8}h + \frac{9}{32}h^2)z_2^{(k)}]$$

For the original algorithm the iteration scheme is given by

$$z_1^{(k+1)} = z_1^{(k)} + \tfrac{h}{3}\left(\sqrt{k_{11}k_{12}} + \sqrt{k_{12}k_{13}} + \sqrt{k_{13}k_{14}}\right)$$

First iteration, $k = 0 \Rightarrow$

$$z_1^{(1)} = z_1^{(0)} + \frac{h}{3}\left(\sqrt{k_{11}k_{12}} + \sqrt{k_{12}k_{13}} + \sqrt{k_{13}k_{14}}\right)$$

$$\underline{z}^{(0)} = (3.40\times10^{-6},\ 1.33\times10^{-6},\ 4.05\times10^{-6}, 4\cdot3\times10^{-6})^T \ h = 10^{-3}$$

$$z_1^{(0)} = (3.40\times10^{-6} + \frac{.001}{3}\left(\sqrt{k_{11}k_{12}} + \sqrt{k_{12}k_{13}} + \sqrt{k_{13}k_{14}}\right)$$

$$k_{11} = z_2^{(0)} = 1.33\times10^{-6},\ k_{12} = z_2^{(0)} + \frac{.001}{2}(1.33\times10^{-6}) = 1.33\times10^{-6} + \frac{.001}{2}(1.33\times10^{-6})$$

$$= 1.33\times10^{-6} + 0.0005\times 1.33\times10^{-6}$$

$$= 1.33\times10^{-6} + 5\times10^{-4}\times 1.33\times10^{-6}$$

$$= 1.33\times10^{-6} + 500\ \text{x}\ 10^{-6}\times1.33\times10^{-6}$$

$$= (1.33 + .005\times\ 1.33)\ \text{x}\ 10^{-6} = 1.33\,0665\ \text{x}\ 10^{-6}$$

$$k_{13} = (1+\frac{5}{8}\ \text{x}\ 10^{-3} + \frac{9}{32}\times10^{-6})\ (1.33\times10^{-6}) = (1+.625\times10^{-3} + 2812\times10^{-6})\ (1.33\times10^{-6})$$

$$= (1.33\text{x}\ 10^{-6} + .625\times1\cdot33\times10^{-9} + .2812\ \text{x}\ 1.33\times10^{-6})\times10^{-6} = 1.330831624\times10^{-6}$$

$$k_{14} = 1.33\times10^{-6} + \frac{.001}{24}\left[-3\ \text{x}\ 1.33\div10^{-6} + 5\times1.33\times10^{-6} + \frac{.005}{2}\times1.33\times10^{-6}22\left(1+\frac{.005}{8}+\frac{.000009}{32}\right)(1.33\times10^{-6})\right]$$

$$= 1.33\times10^{-6} + \frac{.001}{24}\left[-3.99\ \text{x}\ 10^{-6} + 6.65\pm\times10^{-6} + 0.0033258\times10^{-6} + 0.018149429\times10^{-6}\right] = 1.330111728\times10^{-6}$$

$$\sqrt{k_{11}k_{12}} = \sqrt{1.33\ \text{x}\ 10^{-6}\times1.330665\times10^{-6}} = 1.330332458\times10^{-6}$$

$$\sqrt{k_{12}k_{13}} = \sqrt{1.330665\times1.330831624\times10^{-6}} = 1.330748309\times10^{-6}$$

$$\sqrt{k_{13}k_{14}} = \sqrt{1.330831624\times10^{-6}\ \times\ 1.330111728\times10^{-6}} = 1.330471671627\times10^{-6}$$

$$z_1^{(1)} = 3.40\times10^{-6} + \frac{.001}{3}(1.330332458 + 1.330748309 + 1.330471627)\times10^{-6}$$

$$= 3.40\times10^{-6} + 1.330517465\times10^{-3}\times10^{-6} = 3.401330517\times10^{-6}$$

Original algorithm

$$\sqrt{k_{11}k_{14}} = \sqrt{1.33\times10^{-6}\times1.330111728\times10^{-6}} = 1.315479374\times10^{-6}$$

$$z_1^{(1)} = 3.40\times10^{-6} + .00176901\times10^{-6} = 3.40176901\times10^{-6} \approx 3.40\times10^{-6}$$

Modified algorithm

$$f_2(t, z_1, z_2, z_3, z_4) = z_4 - z_2^2\,\text{sgn}\,z_2$$

$$z_2^{(k+1)} = z_2^{(k)} + \frac{h}{3}\left(\sqrt{k_{11}k_{12}} + \sqrt{k_{12}k_{13}} + \sqrt{k_{13}k_{14}}\right)$$

where

$$k_{11} = f_2(t^{(k)}, z_1^{(k)}, z_2^{(k)}, z_3^{(k)}, z_4^{(k)}) = z_4^{(k)} - z_2^{(k)} \text{ sgn } z_2^{(k)},$$

$$k_{12} = f_2\left(t^{(k)}, z_1^{(k)} + \frac{1}{2}hk_{11}, z_2^{(k)} + \frac{1}{2}hk_{11}, z_3^{(k)} + \frac{1}{2}hk_{11}, z_4^{(k)} + \frac{1}{2}hk_{11},\right)$$

$$= (z_4^{(k)} + \frac{1}{2}hk_{11}) - (z_2^{(k)} + \frac{1}{2}hk_{11}) \text{sgn } (z_2^{(k)} + \frac{1}{2}hk_{11})$$

$$k_{13} = f_2\left(t^{(k)} + \frac{1}{2}h, z_1^{(k)}, + \frac{h}{16}(k_{11} + 9k_{12}), z_2^{(k)} + \frac{h}{16}(k_{11} + 9k_{12}), z_4^{(k)} + \frac{h}{16}(k_{11} + 9k_{12})\right)$$

$$= z_4^{(k)} + \frac{h}{16}(k_{11} + 9k_{12}) - (z_2^{(k)} \frac{h}{16}(k_{11} + 9k_{12})) \text{ sgn } (z_2^{(k)} \frac{h}{16}(k_{11} + 9k_{12})$$

$$k_{14} = f_2(t^{(k)} + h, z_1^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}), z_2^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13})$$

$$z_3^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}), z_4^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}))$$

$$= [z_4^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13}) - \left(z_2^{(k)} + \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13})\right) \text{ sgn } z_2^{(k)}$$

$$+ \frac{h}{24}(-3k_{11} + 5k_{12} + 22k_{13})$$

$$z_2^{(1)} = z_2^{(0)} + \frac{h}{3}\left(\sqrt{k_{11}k_{12}} + \sqrt{k_{12}k_{13}} + \sqrt{k_{13}k_{14}}\right)$$

$$k_{11} = z_4^{(0)} - z_2^{(0)^2} \text{ sgn } z_2^{(0)} = 4.3 \text{ x } 10^{-9} - (1.33 \text{ x } 10^{-6}) \text{ sgn } (1.33 \text{ x } 10^{-6})$$

$$= 4.39 \text{ x } 10^{-9} - (1.33)^2 10^{-12} \text{ sgn } (1.33 \text{ x } 10^{-6})$$

$$= 4.39 \pm \text{x } 10^{-9} - 1.7689 \text{ x } 10^{-9} \text{ sgn } (.0000133) = 4.39 \text{ x } 10^{-9})^2 - 1.7689 \text{ x } 10^{-12}$$

$$= 4.39 \text{ x } 10^{-9} - .0017689 \text{ x } 10^{-9} = 4.3882311 \text{ x } 10^{-9}$$

$$k_{12} = z_4^{(0)} + \frac{1}{2}(0.001)(4.3882311 \text{x} 10^{-9}) - (1.33 \text{ x } 10^{-6}) \text{ sgn } (1.33 \text{ x } 10^{-6})$$

$$= 4.39 \text{ x } 10^{-9} + (0.5)(.001)(4.388 \text{ x } 10^{-9}) - (1.7689 \text{ x } 10^{-9})$$

$$= 4.3 \text{ x } 10^{-9} + (.0005)(4.388 \text{ x } 10^{-9}) - (1.7689 \text{ x } 10^{-9})$$

$$= [4.3 + (.0005)(4.388) - 1.7689] \text{ x } 10^{-9} - .2.533294 \text{ x } 10^{-9}$$

$$k_{13} = z_4^{(0)} + \frac{h}{16}(k_{11} + 9k_{11}) - (z_2^{(0)} + \frac{h}{16}(k_{11} + 9k_{11}))^2 \text{ sgn } (z_2^{(0)} + \frac{h}{16}(k_{11} + 9k_{11})) \textit{ e.t.c}$$

4.0 Summary of the algorithm: The successive variation of extremals with invariant costate imbedding algorithm (SVEICI)

The algorithm solves the problem:

Minimize $J = h(\underline{x}(t)) + \int_{t_0}^{t_f} g(\underline{x}(t), \underline{u}, (t), t)dt$ subject to the dynamical constraints

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

Form the Hamiltonial functional.

Step 1

For the reduced differential equations by solving the equation $\frac{\partial H}{\partial \underline{u}} = \underline{0}$ for $\underline{u}(t)$ in terms of $\underline{x}(t), \underline{\lambda}(t)$ and substituting in the state-costate equations to obtain the equivalents of equations (1.5) and (1.6) which contains only terms in $x(t)$ and t.

Step 2

Use as initial guess the value of the costate vector $\lambda_0(t_0)$ obtained from the application of the invariant imbedding algorithm and set the iteration counter $k = 0$

Step 3

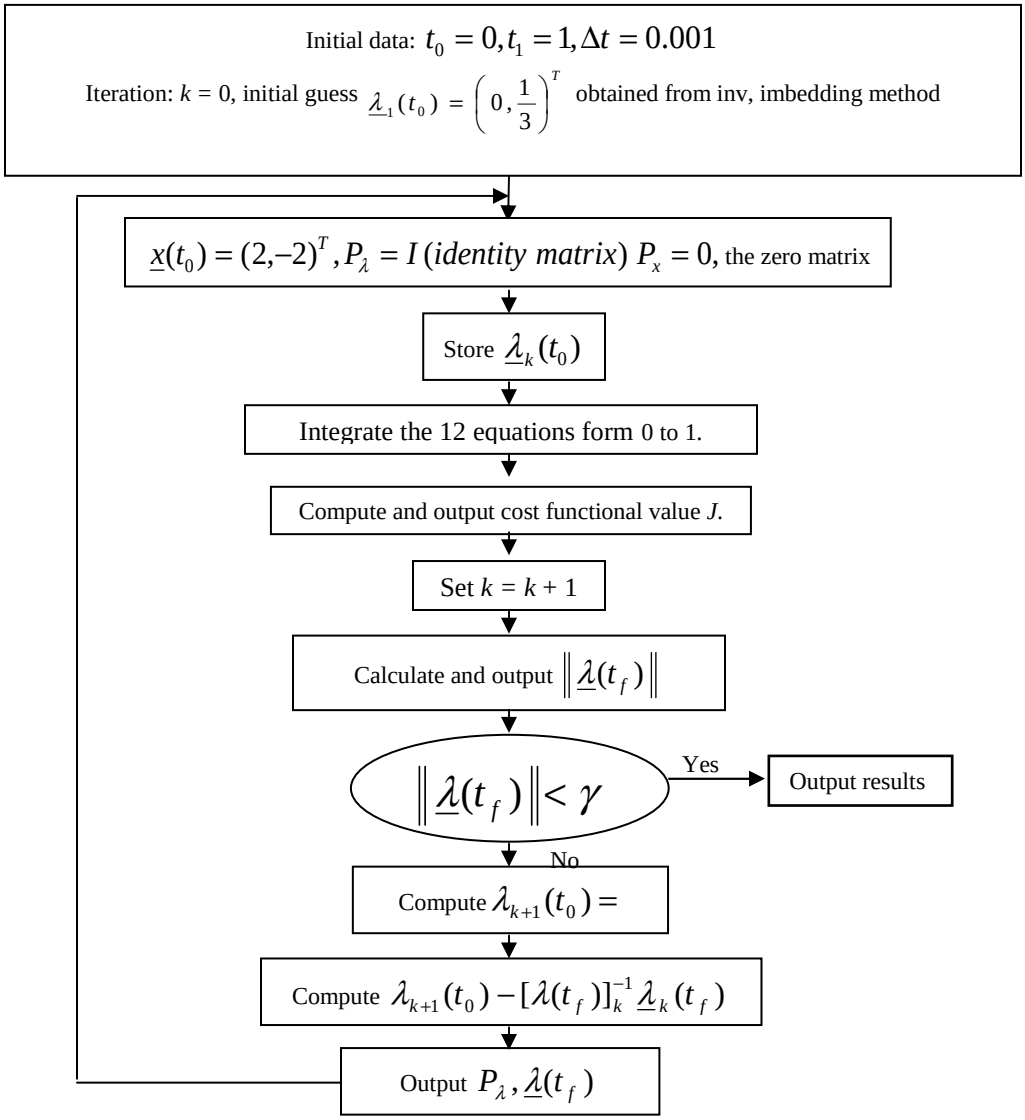
Using $\underline{\lambda}(t_0) = \underline{\lambda}_k(t_0)$ and $\underline{x}(t_0) = \underline{x}_0$ as initial conditions, integrate the reduced state-costate equations and the influence function equations (2.14) with the initial conditions given by equation (49) of [ref. 1] from t_0 to t_f. Store only the values $\underline{\lambda}_k(t_f)$, $\underline{x}_k(t_f)$ and the n x n matrices $P_\lambda(\underline{\lambda}_k(t_0),(t_f))$, $P_x(\underline{\lambda}_k(t_0),(t_f))$.

Step 4

Check to see if the termination criterion $\left\| \underline{\lambda}_k(t_f) - \frac{\partial h}{\partial \underline{x}}(x(t_f)) \right\| < \gamma$ is satisfied where γ is a small prechosen positive constant. If it is, use the final iterates $\underline{\lambda}_0^*(t_0)$ to integrate the state and costate equations and print out the optimal states and controls. Otherwise, find new values for $\underline{\lambda}_0(t_0)$, i.e. $\underline{\lambda}_{k+1}(t_0)$ using equation (1.17). Set $k = k + 1$ and return to step 3.

Now steps 1 and 2 are performed off-line by the user whilst steps 3 and 4 are performed on a digital computer.

Figure 4.1: Information flowchart for SVEICI.



5.0 Conclusion

The problem is taken to be that of minimizing the time required to transfer the system between initial and final states. The extremal control must maximize *H* for the extremal solutions to minimize the performance index. Although the improvement in the solution is appreciably small, the introduction of the fifth term in the *R-K*-4 type integrator employed leads to an error that does not violate the constraint and does not have anything to do with the boundary.

References

[1] T. A. Adewale and B.S. Aribisala (1999): On the control of the pure inertia plant with switching functions using in variant imbedding method. J. Math. Sc. And Education, Vol. I, No. 1, pp. 19 - 36

[2] T. A. Adewale (1998): A Newton-like method for the control of dynamic nonlinear systems. Systems analysis, Modelling and Simulation. (SAMS), Netherlands, Vol. 32, pp. 165-179.

[3] E. J. Bauman (1969): Multilevel optimization techniques with application to trajectory decomposition. Advances in Control systems, Vol. 6, C. T. Londes (ed), Academic Press, New York.

[4] D. J. Bell, P. A. Cook and N. Nunro (1988): Design of modern control systems, Peter Peregrinus Ltd., London, pp. 177-192.

[5] D. N. Burghus, A. Graham (1980): Introduction to Control theory including optimal control, Ellis Horwood Ltd., England, pp.247-260.

[6] A. P. Sage (1969): Optimum systems control, Prentice-Hall, New York.

[7] M. G. Singh, A. Titli (1978): Systems decomposition, optimization and control, Pergamon Press, pp 329 – 380.

[8] P. Hagedorn (1988): Nonlinear oscillations, Clarendon Press, Oxford.

[9] R. P. Agarwal (2000): Difference equations and inequalities, theory, methods, and applications, Marcel Dekker Inc., New York, USA.

[10] B. J. Driessen, N. Sadegh (2002): Minimum-time control with Coulomb friction: Near-Global and optima via integer programming, Journal of intelligent and robatic systems, Netherlands. (to appear)

[11] D.J. Evans and B. B. Sangui (1986): A new forth-order Rung Kutta method for initial value problems in computational mathematics II. Proceedings of he second international conference on numerical analysis and its applications. (S. O. Fatunla, ed.). pp. 13 – 20.