Leapfrog for Optimal Control*

C. Yalçın Kaya† J. Lyle Noakes‡

January 27, 2008

Abstract

The leapfrog algorithm, so-called because of its geometric nature, for solving a class of optimal control problems is proposed. Initially a feasible trajectory is given and subdivided into smaller pieces. In each subdivision, with the assumption that local optimal controls can easily be calculated, a piecewise-optimal trajectory is obtained. Then the junctions of these smaller pieces of optimal control trajectories are updated through a scheme of midpoint maps. Under some broad assumptions the sequence of trajectories is shown to converge to a trajectory that satisfies the Maximum Principle. The main advantages of the leapfrog algorithm are that (i) it does not need an initial guess for the costates, (ii) the piecewise-optimal trajectory generated in each iteration is feasible. These are illustrated through a numerical implementation of leapfrog on a problem involving the van der Pol system.

Key words: Optimal control, two-point boundary-value problem, multiple shooting, geodesics, numerical methods, van der Pol system.

1 Introduction

Computational techniques for finding an optimal control solution of a nonlinear system can be classified into two major categories, namely *direct* and *indirect* methods [37]. In a direct method, the problem is discretized and the resulting (usually large scale) finite-dimensional optimization problem is solved using nonlinear programming techniques [4, 36, 6, 3, 19]. The solution obtained this way is an approximation of the true solution, unless the control function is of a special class, for example of bang-bang type [18, 14, 28]. On the other hand, in an indirect method, one tackles the two-point boundary-value problem (TPBVP) arising from the Pontryagin Maximum Principle (PMP). Accurate solutions can be obtained through indirect methods; however, available solution techniques for TPBVPs often face serious convergence difficulties, mainly because of the lack of a good initial guess. If the set of controls is bounded, the problem is even more difficult.

The leapfrog algorithm for optimal control, so-called because of its geometric nature, is an extension of a similar iterative algorithm given by Noakes [30] for finding a geodesic

*Authors offer their warm thanks to the anonymous reviewer whose comments and suggestions improved the manuscript.

†School of Mathematics and Statistics, University of South Australia, Mawson Lakes, S.A. 5095 Australia. The author gratefully acknowledges support by a fellowship from CAPES, Ministry of Education, Brazil (Grant No. 0138-11/04), for his visit to Department of Systems and Computing at the Federal University of Rio de Janeiro, where a part of this research was carried out. E-mail: yalcin.kaya@unisa.edu.au

‡School of Mathematics and Statistics, The University of Western Australia, Nedlands, W.A. 6009 Australia. E-mail: lyle@maths.uwa.edu.au

joining two given points on a Riemannian manifold. The main advantages of leapfrog are that a good initial guess is not needed and the trajectories generated in each iteration are feasible. Under fairly general conditions, the algorithm is convergent to a trajectory that satisfies the PMP. The algorithm presented here, as well as that given in [30], can be viewed as a continuous-time analogue of the so-called two-stage approximation technique described by Zuo [39] for discrete-time optimal control.

Methods that are used for the solution of TPBVPs arising in optimal control usually incorporate some combination of (a) multiple shooting, (b) collocation methods, (c) continuation (homotopy) methods. Various expositions of multiple shooting can be found in [20, 32, 2, 35, 31, 11, 10, 26, 25, 23, 22]. Multiple shooting is a very accurate method but does still require good initial guesses in the subintervals taken. As the number of subdivisions is increased in order to exploit simple shooting more efficiently in each subinterval, the number of variables to be determined is substantially increased. With these problems in mind, multiple shooting is sometimes used in conjunction with some other method, for example, the method of collocation [38]. The method of collocation is used to obtain an approximate solution, in terms of cubic splines for example, which is then fed into multiple shooting to achieve the required solution accurately. This two-fold procedure may eventually need long computational times. The continuation method is also a method used in conjunction with multiple shooting [9, 7]. Usually, the nonlinear system is partitioned into its linear and nonlinear parts, and using the initial linear system solution and continuation techniques, the nonlinear system's solution is obtained. This method can also take long computational times depending on the nonlinearity of the system.

The leapfrog algorithm resembles multiple shooting in that the time interval is subdivided and local optimal controls are found separately over each subinterval. A significant difference is that leapfrog uses overlapping subintervals. Because of the use of subdivisions the algorithm is not expected to be troubled by the nonlinear dynamics. Three apparent advantages of the leapfrog algorithm over multiple shooting can be listed:

- Once a feasible trajectory is constructed, affine approximation of the subproblem in a subdivision (between nearby points) provides a good initial guess needed for simple shooting in that subdivision. So leapfrog does not depend critically on the supply of good initial guesses along the feasible trajectory.
- State trajectories generated by leapfrog are feasible at each step of the iteration, which is not in general the case in multiple shooting. Continuing improvements are obtained on the feasible trajectories, resulting in suboptimal trajectories. This may in particular be useful in real-time (on-line) applications, where time allowed to compute a solution is restricted.
- In each leapfrog iteration the number of subdivisions can be adjusted along the feasible trajectory. This allows one to eventually reduce the number of subdivisions as the solution is approached.

A drawback of leapfrog is that, as the solution is approached, convergence of leapfrog slows down; therefore the number of subdivisions, q, has to be progressively reduced to 2 (which is the case of simple shooting) so as to obtain a solution in a reasonable number

of iterations. The effort to see, in each iteration, whether a reduction in q is possible or not, incurs an additional computational cost.

Although leapfrog needs an initial feasible trajectory, in an implementation one can as well start with a set of nearby subdivision points through which concatenated local optimal solutions are obtained. Generally speaking, feasible trajectories can be established much more easily than optimal ones.

We list below some additional features of the leapfrog algorithm.

- Parallel computing is one of the efficiency concerns in the TPBVP literature (see [1, 29, 33]). Multiple shooting is particularly suited to doing computations in each subdivision independently, and for this reason it is sometimes referred to as *parallel shooting*. Using a slight variation of a leapfrog iteration, those subproblems whose time intervals are not overlapping can be solved independently. Therefore leapfrog is also suited to parallel computations.

- Suppose that an optimal control problem is given with n state equations, where the initial and terminal states are prescribed and the terminal time is fixed. For finding a local optimal control in a subdivision, the leapfrog algorithm updates only $2n$ real variables (states and costates) at a time. This amounts to a total of $[(2q-4)n]$ updates in one leapfrog iteration. With q subdivisions, multiple shooting updates $[(2q-1)n]$ variables at a time in one iteration.

- After a subproblem is solved, if two or more subdivision points (i.e. some of the junctions of the piecewise-optimal trajectories) become the same, they are referred to as *multiple points*. Multiple points occur rarely in practice. However, if they occur, leapfrog can eliminate them, yielding fewer subdivisions to deal with along a (piecewise-optimal) feasible trajectory. Elimination of multiple points does not have a clear advantage with multiple shooting.

In this paper, we show that the leapfrog algorithm converges under some general assumptions, such as the uniqueness of the local optimal control and costate variables in a given subdivision. In [15, 16], we gave some background and a survey of ideas leading to the development of leapfrog for solving the geodesic problem. We also reported an implementation of leapfrog (with fixed terminal time) for optimal control, without elaboration. In the present paper, we provide a careful and detailed account of a numerical implementation of leapfrog. We also gave some preliminaries of the theory in an earlier publication [17], which did not have the detailed setting and approach of the present article.

The present paper is organised as follows. In Section 2 we state the optimal control problem, give definitions and assumptions, verifications of which are provided through example problems. In Section 3 we describe the leapfrog algorithm. In Section 4 we address multiple points and provide further definitions and facts. We prove convergence of the leapfrog algorithm in Section 5. A computational illustration of the algorithm is given in Section 6 by solving a problem involving the van der Pol system.

2 Preliminaries

Consider the control system

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

where the state $x(t) \in \mathbb{R}^n$, the control $u : \mathbb{R} \longrightarrow \mathbb{R}^m$ is an arbitrary piecewise-continuous function, which, as described in [34], has discontinuity at finitely many points and has finite right- and left-hand limits at the points of discontinuity. The function $f : \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}^n$ is C^1 in x and u.

The following optimal control problem is considered.

$$\mathrm{P} : \begin{cases} \text{minimize} & \displaystyle\int_{t_0}^{t_f} f_0(x(t), u(t))\, dt \\ \text{subject to} & \dot{x}(t) = f(x(t), u(t)), \\ & x(t_0) = x_0 \quad \text{and} \quad x(t_f) = x_f \ . \end{cases}$$

where the function $f_0 : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is C^1 and nonnegative, and the final time $t_f > t_0$ is free. A *state trajectory* corresponding to a control $u(\cdot)$ is a continuous curve $x(\cdot)$ solving $\dot{x}(t) = f(x(t), u(t))$ for almost all t. An *optimal state trajectory* $x(\cdot)$ and a corresponding *optimal control* $u(\cdot)$ are those which solve Problem (P).

The Pontryagin Maximum Principle (PMP) gives necessary conditions of optimality [34] for large classes of problems including Problem (P). In these problems the so-called Hamiltonian $H : \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}$ is defined as

$$H(x, \psi, u) = \psi_0\, f_0(x, u) + \langle \psi, f(x, u) \rangle$$

where $\psi_0(t) \in \mathbb{R}$ and $\psi(t) \in \mathbb{R}^n$ are the adjoint or costate variables, and $\langle \cdot, \cdot \rangle$ is the Euclidean inner product. The PMP states that if a state trajectory $x(\cdot)$ and a corresponding control $u(\cdot)$ are optimal, then there exists a nontrivial pair $(\psi_0, \psi) : [t_0, t_f] \longrightarrow \mathbb{R}^{n+1}$ such that, for almost every $t \in [t_0, t_f]$,

$$\dot{x} = \partial H / \partial \psi = f(x, u) \tag{2}$$

$$\dot{\psi} = -\partial H / \partial x \tag{3}$$

$$H(x, \psi, u) = \min_{v \in \mathcal{U}} H(x, \psi, v) \tag{4}$$

$$H(x, \psi, u) \equiv 0 \tag{5}$$

with $x(t_0) = x_0$ and $x(t_f) = x_f$. Furthermore PMP requires that $\psi_0(t) \equiv c$, where c is some nonnegative constant. The costate variable ψ_0 accounts for the case when cost is treated as an additional state coordinate. Problems that yield $c = 0$ are referred to as *abnormal* in the literature, for which the necessary conditions in (2)-(5) are independent of the cost and therefore insufficiently informative. Discussion of necessary conditions of optimality for abnormal trajectories can be found in [24] and the references therein.

Assumption 1 *Problem (P) is normal. Without loss we set* $\psi_0 = 1$.

We refer to the conditions given in (2)-(5) as the *optimality system*. A trajectory of the optimality system corresponding to a control $u(\cdot)$ is a continuous pair of curves $(x(\cdot), \psi(\cdot))$

which solve (2)-(5). If the trajectory pair $(x(\cdot), \psi(\cdot))$ corresponding to control $u(\cdot)$ satisfies the optimality system, then it is said to be a *critical trajectory* (*pair*). We will refer to the trajectory $(x(\cdot), \psi(\cdot))$ as optimal if the state trajectory $x(\cdot)$ along with a corresponding control $u(\cdot)$ is optimal. Note that a critical trajectory is not necessarily optimal.

Remark 2.1 *Condition* (5) *stands because the terminal time t_f is free – t_f is an unknown of the problem. In the case when t_f is prescribed however, Condition* (5) *is not part of the optimality system anymore. In this case one has $H(x, \psi, u) \equiv a$, where a is any real constant, but this condition is usually not needed for computations, resulting in a simpler optimality system.*

Remark 2.2 *Although leapfrog is described for free terminal time problems, we can also tackle problems with fixed terminal time. Given a problem with fixed t_f, say $t_f = 1$, we can define a new state variable $x_{n+1}(t) = t$, such that $\dot{x}_{n+1}(t) = 1$, with the end points $x_{n+1}(0) = 0$ and $x_{n+1}(t_f) = 1$, where we now let t_f be free. The equation for x_{n+1} is trivial to solve. Therefore, in a practical implementation, there would be no need to augment the state equations in this way.*

Next we describe the subproblems employed within leapfrog. Let $x(\cdot)$ be a continuous piecewise-C^1 trajectory such that $x(t_0) = x_0$ and $x(t_f) = x_f$. The point x_f is not necessarily close to x_0. The trajectory $x(\cdot)$ is simply a curve between x_0 and x_f. The aim is to find a critical trajectory between x_0 and x_f. We initially partition this feasible trajectory so as to obtain q pieces with roughly comparable costs. Let the corresponding partition times be $t_0 < t_1 < \ldots < t_{q-1} < t_q = t_f$. The number q of partitions is chosen so that the partition points $z_{i-1} := x(t_{i-1})$ and $z_i := x(t_i)$, $i = 1, \ldots, q$, are sufficiently close to each other that there exists a local optimal solution from z_{i-1} to z_{i+1}, $i = 1, \ldots, q-1$, which is easily computable. Namely we assume that local solutions to the following subproblems can be easily obtained:

$$\mathrm{P}_i : \begin{cases} \text{minimize} & \displaystyle\int_{t_{i-1}}^{t_{i+1}} f_0(x(t), u(t))\, dt \\ \text{subject to} & \dot{x}(t) = f(x(t), u(t)), \\ & x(t_{i-1}) = z_{i-1} \quad \text{and} \quad x(t_{i+1}) = z_{i+1} \end{cases}$$

where t_{i-1} is fixed by the solution of Problem (P_{i-1}), and t_{i+1} is free. Note that $z_0 = x_0$ and $z_q = x_f$. Through the PMP, the optimality system associated with Problem (P_i) can be written as in (2)-(5), for $t \in [t_{i-1}, t_{i+1}]$, with endpoints $x(t_{i-1}) = z_{i-1}$ and $x(t_{i+1}) = z_{i+1}$. Solution of the optimality system provides an initial value of the costate variable, namely $\psi(t_{i-1}) =: \lambda_{i-1}$, for Problem ($\mathrm{P}_i$).

Assumption 2 *There exists a.e. a (locally) unique optimal control $u : [t_{i-1}, t_{i+1}] \longrightarrow \mathbb{R}^m$ taking the system from z_{i-1} to z_{i+1}.*

Assumption 3 *The normalized costate vector $\psi(t)$ associated with Problem (P_i) is determined uniquely by the locally optimal control $u(t)$ and the initial point z_{i-1} of the optimal trajectory.*

In what follows we discuss two example problems, albeit relatively simple, to illustrate Assumptions 1-3. In these problems the terminal time is fixed: as pointed earlier they can easily be converted into an equivalent free terminal time form used for leapfrog.

Example 1 (Geodesics) Let $\{g_{ij} : 1 \leq i, j \leq n\}$ define a Riemannian metric on an open subset U of $\mathbb{R}^n$, namely the matrix $[g_{ij}(x)]$ is symmetric positive definite, and C^∞ in x. The problem of finding geodesics between two given points can be formulated as an optimal control problem as follows.

$$\mathrm{P}_g : \begin{cases} \text{minimize} & \displaystyle\int_0^1 \sum_{i,j} g_{ij}(x) u_i u_j \, dt \\ \text{subject to} & \dot{x} = u, \\ & x(0) = x_0 \quad \text{and} \quad x(1) = x_f \ , \end{cases}$$

where the initial and terminal points x_0 and x_f are in $\mathbb{R}^n$. The control $u(t)$ is also in $\mathbb{R}^n$. The Hamiltonian for the problem is

$$H(x, u, \psi) = \psi_0 \sum_{i,j} g_{ij}(x)\, u_i u_j + \langle \psi, u \rangle$$

Condition (4) in this case implies $\partial H / \partial u_i = 0$, $i = 1, \ldots, n$, which gives

$$\psi_i = -2\, \psi_0 \sum_j g_{ij}(x)\, u_j \ . \tag{6}$$

Suppose $\psi_0 = 0$. Then $\psi_i = 0$, $i = 1, \ldots, n$, i.e. (ψ_0, ψ) is trivial, which is a contradiction. This verifies Assumption 1. Set $\psi_0 = 1$ from this point onwards.

Given u_i, $i = 1, \ldots, n$, ψ_i are determined uniquely from (6). Differentiating ψ_i one gets

$$\dot{\psi}_i = -2 \left(\sum_{j,k} \frac{\partial g_{ij}}{\partial x_k} u_j u_k + \sum_j g_{ij}(x)\, \dot{u}_j \right) \ .$$

On the other hand, another condition (3) of PMP yields

$$\dot{\psi}_i = - \sum_{j,k} \frac{\partial g_{jk}}{\partial x_i} u_j u_k \ .$$

Equating the two expressions for $\dot{\psi}_i$ and rearranging,

$$\dot{u}_i = - \sum_{j,k} \Gamma^i_{jk}(x)\, u_j u_k \tag{7}$$

where

$$\Gamma^i_{jk}(x) = \frac{1}{2} \sum_s g^{is}(x) \left(2 \frac{\partial g_{sj}}{\partial x_k} - \frac{\partial g_{jk}}{\partial x_s} \right) \ .$$

In the above equations Γ^i_{jk} are the so-called Christoffel symbols [5], and g^{is} is the is-th entry of the inverse of the matrix $[g_{is}]$. A direct substitution from the state equations give the so-called geodesic equations which has a unique solution for x_f sufficiently close to x_0:

$$\ddot{x}_i = - \sum_{j,k} \Gamma^i_{jk}(x)\, \dot{x}_j \dot{x}_k \ . \tag{8}$$

Then locally the control u has a unique solution, which verifies Assumption 2; and in turn, Assumption 3 is verified by Equation (6). □

Example 2 (LTI systems) Consider the minimum energy control of a linear time-invariant (LTI) system:

$$\mathrm{P}_{LTI}: \begin{cases} \text{minimize} & \dfrac{1}{2}\displaystyle\int_0^1 u^T R u \, dt \\ \text{subject to} & \dot{x}(t) = Ax(t) + Bu(t) \\ & x(0) = x_0 \quad \text{and} \quad x(1) = x_f \ , \end{cases}$$

where $x(t) \in \mathbb{R}^n$, $u : \mathbb{R} \longrightarrow \mathbb{R}^m$, and the constant matrices A and B have appropriate dimensions. The constant matrix R is symmetric and positive definite. The system is assumed to be controllable. The Hamiltonian is

$$H(x, \psi, u) = \frac{1}{2}\, \psi_0\, u^T R u + \psi^T (Ax + Bu) \ .$$

The condition $\partial H / \partial u = 0$ gives

$$\psi_0\, Ru = -B^T \psi \tag{9}$$

Furthermore (3) of PMP yields

$$\dot{\psi}^T = -\psi^T A \ . \tag{10}$$

Differentiating (9) up to $(n-1)$ times and using (10) one gets

$$\begin{aligned} \psi^T B &= \psi_0\, Ru \\ \psi^T AB &= \psi_0\, R\dot{u} \\ &\cdots \\ \psi^T A^{n-1} B &= \psi_0\, Ru^{(n-1)} \ . \end{aligned} \tag{11}$$

where $u^{(i)} = d^i u / dt^i$, $i = 1, \ldots, n-1$, with $\dot{u} = u^{(1)}$.

Suppose $\psi_0 = 0$. Since the system is controllable, the controllability matrix $[B|AB|\cdots|A^{n-1}B]$ is of rank n, so $\psi = \mathbf{0}$, which contradicts the nontriviality of (ψ_0, ψ) in PMP. This verifies Assumption 1.

We set $\psi_0 = 1$. Again because the system is controllable, ψ can be expressed uniquely from (11) in terms of u and its derivatives. This verifies Assumption 3.

It can be shown (using $R = R^{1/2} R^{1/2}$ and by considering a generalization of the controllability Gramian in Exercise 9.2-11 of [13]) that the optimal control $u(t)$ is given by the expression

$$u(t) = -R^{-1} B^T e^{-A^T t} M^{-1} \left(x(0) - e^{-A} x(1) \right) \tag{12}$$

where

$$M = \int_0^1 e^{-sA} B R^{-1} B^T e^{-sA^T} ds \ ,$$

and nonsingularity of M is furnished by the controllability of the system. As can be seen, the optimal control $u(t)$ is unique by the choice of $x(0)$ and $x(1)$, which verifies Assumption 2. □

A point $\widetilde{x}$ is said to be *reachable* from a point x_0 with cost less than $\Delta > 0$ if and only if, for some admissible control, there exists a trajectory $x(\cdot)$ defined on an interval $[t_0, t]$, $t > t_0$, such that $x(t_0) = x_0$, $x(t) = \widetilde{x}$, and $\int_{t_0}^{t_f} f_0(x(t), u(t))\, dt < \Delta$. The trajectory $x(\cdot)$ of concern here does not necessarily satisfy PMP, namely it is not necessarily critical. The set of all such points $\widetilde{x}$ is denoted by $\mathcal{R}(x_0, < \Delta)$. The reachable set $\mathcal{R}(x_0, < \Delta)$ is said to be a *small-cost reachable set* when Δ is sufficiently small.

Let D denote the set of all ordered pairs (z_{i-1}, z_{i+1}) such that z_{i+1} can be reached from z_{i-1} with a cost less than 2δ, namely

$$D = \{(z_{i-1}, z_{i+1}) \in \mathbb{R}^n \times \mathbb{R}^n : z_{i+1} \in \mathcal{R}(z_{i-1}, < 2\delta)\}\ , \tag{13}$$

where $\delta > 0$ is so chosen that there exists a local optimal control from z_{i-1} to z_{i+1}, which is easily computable. Namely, the choice of δ ensures that the points z_{i-1} and z_{i+1} are nearby so that affine approximations and the simple shooting method can be used to get a local solution for Problem (P_i) efficiently.

Let $\tau : D \longrightarrow \mathbb{R}$ denote the *local cost function* such that, given $(z_{i-1},\ z_{i+1})$ in D, $\tau(z_{i-1}, z_{i+1})$ is the infimum of the cost to get from z_{i-1} to z_{i+1}. Note that $\tau(z_{i-1}, z_{i+1}) < 2\delta$.

Clearly both D and τ depend on δ; however, for brevity, we will not show this dependence explicitly. The associated value of δ should be clear from the context.

3 Leapfrog Algorithm

In this section we give a description of the leapfrog algorithm. Suppose that a feasible trajectory $\mu_z : [t_0, t_f] \longrightarrow \mathbb{R}^n$ is given between the initial and terminal states $z_0 = \mu_z(t_0)$ and $z_q = \mu_z(t_f)$, which is divided into q trajectory segments, such that $z_i = \mu_z(t_i)$, $i = 1, \ldots, q-1$. The junction points of μ_z are represented by the $(q+1)$-tuple $z = (z_0, z_1, z_2, \ldots, z_q)$. Through the leapfrog scheme, while the initial and terminal states z_0 and z_q, and t_0, are kept fixed, each z_i, $i = 1, \ldots, q-1$, and t_i, $i = 1, \ldots, q$, are adjusted in the following fashion.

For $(z_{i-1}, z_{i+1}) \in D$, let $\gamma_{z_{i-1}, z_{i+1}} : [t_{i-1}, t_{i+1}] \longrightarrow \mathbb{R}^n$ be the unique *local optimal trajectory* (by Assumption 2) such that $\gamma_{z_{i-1}, z_{i+1}}(t_{i-1}) = z_{i-1}$ and $\gamma_{z_{i-1}, z_{i+1}}(t_{i+1}) = z_{i+1}$. The *midpoint* $\widehat{z}_i := \gamma_{z_{i-1}, z_{i+1}}(\widehat{t}_i)$, $t_{i-1} \le \widehat{t}_i \le t_{i+1}$, between z_{i-1} and z_{i+1} is defined by $\tau(z_{i-1}, \widehat{z}_i) = \tau(\widehat{z}_i, z_{i+1}) = \tau(z_{i-1}, z_{i+1})/2$. For convenience, we also set $\widehat{t}_0 := t_0$ and $\widehat{t}_q := t_q$. The point z_i is moved to the midpoint $\widehat{z}_i$, while the time t_i is updated as $\widehat{t}_i$.

Moving z_i onto an optimal trajectory from z_{i-1} to z_{i+1} (obtained by solving Problem (P_i)) achieves locally the largest possible decrease in cost while keeping other junction points fixed. The point where z_i goes onto in the optimal trajectory segment is taken to be the point such that the cost of getting from z_{i-1} to that point is half the cost of getting from z_{i-1} to z_{i+1}. However, somewhat different positioning of z_i is also permissible.

The leapfrog algorithm consists of updating z_i and t_i in this way infinitely often where i cycles through $1, \ldots, q-1$. This generates a sequence

$$\mathcal{M}_z = \{\mu_z^{(k)} : [t_0, t_f^{(k)}] \longrightarrow \mathbb{R}^n : k > 0\}$$

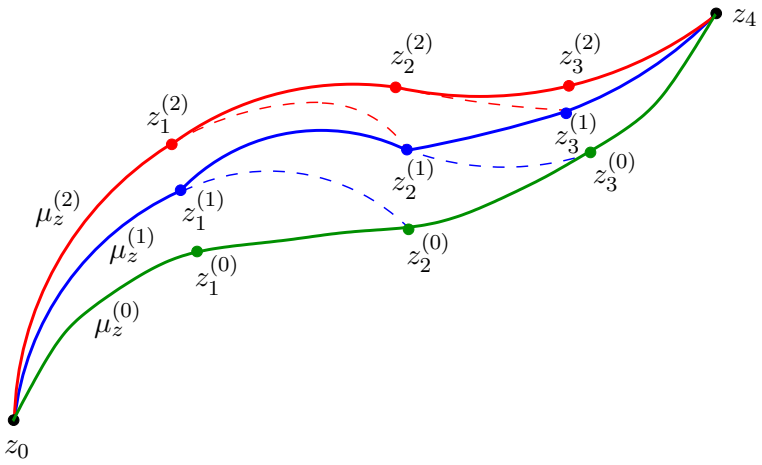


Figure 1: An illustration of the leapfrog algorithm in state-space.

of piecewise-optimal (state) trajectories. Along each consecutive trajectory $\mu_z^{(k)}$ the cost decreases. This decrease precludes convergence to a global maximum. A schematic illustration of the leapfrog algorithm is given in state-space in Figure 1. The initial set of points $z_i^{(0)}$, $i = 1, 2, 3$, are the junctions chosen on an initial feasible trajectory. Two leapfrog iterations are depicted. The piecewise-optimal trajectory $\mu_z^{(1)}$ is obtained after the first iteration from the initial feasible trajectory. The second iteration yields the piecewise-optimal trajectory, $\mu_z^{(2)}$.

Let us outline the leapfrog algorithm using the definitions we have just given.

Step 0 Choose the number of subdivisions $q > 0$ and final time $t_f > t_0$.

Take an initial feasible trajectory $\mu_z^{(0)}$ given between $z_0 = x_0$ and $z_q = x_f$. Subdivide $\mu_z^{(0)}$ into q segments with subdivision points

$$z_0^{(0)},\ z_1^{(0)}, \ldots, z_{q-1}^{(0)},\ z_q^{(0)}\ .$$

Set $t_0^{(0)} = t_0$ and $k = 0$.

Step k Perform Steps $k.1$ and $k.2$ below for $i = 1, 2, \ldots, q-1$.

Step $k.1$ Given $z_{i-1}^{(k)}$, $z_{i+1}^{(k)}$ and $t_{i-1}^{(k)}$:

Solve Problem (P_i) with $x(t_{i-1}) = z_{i-1}^{(k)}$, $x(t_{i+1}) = z_{i+1}^{(k)}$ and $t_{i-1} = t_{i-1}^{(k)}$.

Step $k.2$ Let $z_i^{k+1} = x(t_i)$ and $t_i^{k+1} = t_i$ such that

$$\tau(x(t_{i-1}), x(t_i)) = \frac{1}{2}\,\tau(x(t_{i-1}), x(t_{i+1}))\ .$$

Set $k = k + 1$ and repeat Step k.

For proving convergence to a critical trajectory, it is necessary to consider the local critical trajectory pairs $(x(\cdot), \psi(\cdot))$ obtained by solving the subproblems (P_i). For each

update of z_i, an update is also obtained for $\lambda_{i-1} := \psi(t_{i-1})$ and t_i, $i = 1, \ldots, q-1$, through the solution of (P_i) and the associated optimality system. The updates for λ_{q-1}, and λ_q and $t_q = t_f$ are given by the trajectories of the optimality system associated with Problem (P_{q-1}).

Now let $y_i = (z_i, \lambda_i) = (x(t_i), \psi(t_i))$, consider the $(q+1)$-tuple $y = (y_0, y_1, y_2, \ldots, y_q)$ and the sequence

$$\mathcal{M} = \{\mu^{(k)} : [t_0, t_f^{(k)}] \longrightarrow \mathbb{R}^{2n} : k > 0\} \ ,$$

where $\mu^{(k)}(\cdot) = (x^{(k)}(\cdot), \psi^{(k)}(\cdot))$, with $\mu^{(k)}(t_0) = (z_0, \lambda_0^{(k)})$ and $\mu^{(k)}(t_q) = (z_q, \lambda_q^{(k)})$. In each iteration of leapfrog the states $x(t_0) = z_0$ and $x(t_f) = z_q$ at the end points of $\mu^{(k)}$ are fixed but the costates $\psi(t_0) = \lambda_0$ and $\psi(t_f) = \lambda_q$, as well as the terminal time t_f, are updated when the new local optimal controls are found and concatenated. Under the assumptions we pose, notably under the uniqueness of the local optimal control and costates, the sequence $\{\mu^{(k)} : k > 0\}$ will be shown to converge to a critical trajectory.

4 Further Definitions and Facts

4.1 Elimination of multiple points

If two or more of z_i become the same point, then they are referred to as *multiple points*. Clearly there are no multiple points along the initial trajectory, because the trajectory is not self-intersecting and along each trajectory segment the cost is positive. Multiple points may occur, albeit rarely, in leapfrog, but they can be eliminated through a re-labelling procedure described below.

Suppose that there are $\tilde{q}$ subdivisions. Fix i, $0 < i < \tilde{q}$. Let $z_i^{(k+1)}$ be the midpoint along the local optimal trajectory from $z_{i-1}^{(k+1)}$ to $z_{i+1}^{(k)}$.

(M1) If $z_i^{(k+1)} = z_\ell^{(k+1)}$ for some ℓ where $0 \leq \ell < i$, then re-label $z_r^{(k)}$, $r = i+1, \ldots, \tilde{q}$, such that

$$z_{i+s}^{(k)} = z_{\ell+s}^{(k)}, \quad s = 1, \ldots, \tilde{q} - i.$$

(M2) If $z_i^{(k+1)} = z_\ell^{(k)}$ for some ℓ where $i < \ell \leq \tilde{q}$, then re-label $z_r^{(k)}$, $r = \ell, \ldots, \tilde{q}$, such that

$$z_{\ell+s}^{(k)} = z_{i+s}^{(k)}, \quad s = 0, 1, \ldots, \tilde{q} - \ell.$$

Note that if (M1) applies, the earlier piecewise-optimal trajectory from $z_\ell^{(k+1)}$ to $z_{i+1}^{(k)}$ is replaced by the new local optimal trajectory from $z_i^{(k+1)}$ to $z_{i+1}^{(k)}$. If (M2) applies, the earlier piecewise-optimal trajectory from $z_{i-1}^{(k+1)}$ to $z_\ell^{(k)}$ is replaced by the new local optimal trajectory from $z_{i-1}^{(k+1)}$ to $z_i^{(k)}$. In either case the number of subdivisions $\tilde{q}$ is reduced by $|\ell - i|$. The removal of trajectory segments also reduces total cost.

Either (M1) or (M2) is repeated as often as there are multiple points. Because $\tilde{q}$ is finite, the elimination procedure is applied only a finite number of times, until no more

multiple points occur in the remaining leapfrog iterations, yielding a constant number of subdivisions q such that $1 \leq q \leq \tilde{q}$.

It can be shown as a special case that in the steps (M1) and (M2), one cannot have $\ell = i-1$ or $i+1$. We state this observation as a lemma below.

Lemma 4.1 *$z_i^{(k+1)} \neq z_{i-1}^{(k+1)}$ and $z_i^{(k+1)} \neq z_{i+1}^{(k)}$.*

Proof. Suppose $z_i^{(k+1)} = z_{i-1}^{(k+1)}$. Then there exist two local optimal trajectories from $z_{i-1}^{(k+1)}$ to $z_{i+1}^{(k)}$: the local optimal trajectory emanating from $z_{i-1}^{(k+1)}$, revisiting $z_{i-1}^{(k+1)} = z_i^{(k+1)}$ and terminating at $z_{i+1}^{(k)}$, and the segment of the same trajectory after revisiting $z_{i-1}^{(k+1)}$. This contradicts the uniqueness assumption of the local optimal trajectories. Therefore $z_i^{(k+1)} \neq z_{i-1}^{(k+1)}$. Suppose $z_i^{(k+1)} = z_{i+1}^{(k)}$. A similar argument leads to a similar contradiction. Hence $z_i^{(k+1)} \neq z_{i+1}^{(k)}$. □

It should be noted that, in practice, sufficiently nearby z_i may be treated as multiple points. This is highly desirable because it would result in a reduction in both $\tilde{q}$ and cost along the current leapfrog trajectory.

In the sections to follow we give more tools and assumptions for the convergence analysis of leapfrog with constant q after the elimination process described above is completed.

4.2 Continuity of the local cost function

The following assumption will be used in proving the continuity of τ stated in Lemma 4.2, which will in turn be useful in the convergence analysis.

Assumption 4 *Given $\epsilon > 0$, there exists $\kappa > 0$ such that, for any $z_i, \widetilde{z}_i \in \mathbb{R}^n$,*

$$\|z_i - \widetilde{z}_i\| < \kappa \implies \tau(z_i, \widetilde{z}_i) < \epsilon \ .$$

Example 3 Recall the optimal control problem with the LTI system in Example 2. With $x_0 = z_i$ and $x(1) = \widetilde{z}_i$, it can be shown that

$$\tau(z_i, \widetilde{z}_i) = \frac{1}{2} \left(z_i - e^{-A}\widetilde{z}_i\right)^T M^{-1} \left(z_i - e^{-A}\widetilde{z}_i\right) \ ,$$

where

$$M = \int_0^1 e^{-sA} B R^{-1} B^T e^{-sA^T} \, ds \ .$$

Because the LTI system is controllable, the Gramian M is positive definite. Therefore $\tau(z_i, \widetilde{z}_i) < \epsilon$ defines a region enclosed by an ellipsoid containing z_i, and thus some $\kappa > 0$ can be found to enclose an open ball of radius $\kappa > 0$ centred at z_i within the ellipsoid. This justifies Assumption 4. In fact, one can conclude from the above expression that the local cost function τ is continuous.

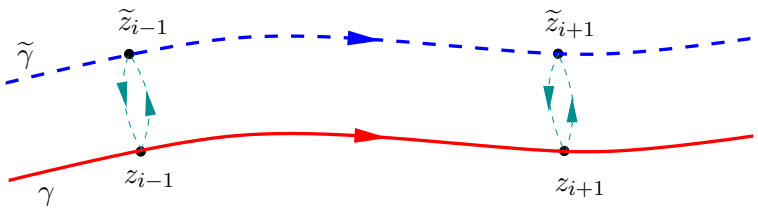


Figure 2: Two neighbouring optimal trajectories γ and $\widetilde{\gamma}$.

Lemma 4.2 *The local cost function τ is continuous.*

Proof. Consider the set D given in (13) with some $\delta > 0$ and the open ball $B_\kappa(z_{i-1}, z_{i+1})$ with radius $\kappa > 0$ centred at $(z_{i-1}, z_{i+1}) \in D$, namely

$$B_\kappa(z_{i-1}, z_{i+1}) = \{(\widetilde{z}_{i-1}, \widetilde{z}_{i+1}) \in D : \|(z_{i-1}, z_{i+1}) - (\widetilde{z}_{i-1}, \widetilde{z}_{i+1})\| < \kappa\} \tag{14}$$

where $\|\cdot\|$ is the Euclidean norm. The inequality in (14) implies that

$$\|z_{i-1} - \widetilde{z}_{i-1}\| < \kappa \qquad \text{and} \qquad \|z_{i+1} - \widetilde{z}_{i+1}\| < \kappa \ .$$

Because $(z_{i-1}, z_{i+1}) \in D$ there is a local optimal trajectory, γ, passing through z_{i-1} and z_{i+1}. Similarly because $(\widetilde{z}_{i-1}, \widetilde{z}_{i+1}) \in D$, one has a local optimal trajectory, $\widetilde{\gamma}$, passing through $\widetilde{z}_{i-1}$ and $\widetilde{z}_{i+1}$. The trajectories γ and $\widetilde{\gamma}$ are illustrated in Figure 2.

By Assumption 4, for any $\epsilon > 0$, there exist $\kappa_1 > 0$ and $\kappa_2 > 0$ such that

$$\|z_i - \widetilde{z}_i\| < \kappa_1 \implies \tau(z_i, \widetilde{z}_i) < \epsilon/2 \ ,$$

and

$$\|\widetilde{z}_i - z_i\| < \kappa_2 \implies \tau(\widetilde{z}_i, z_i) < \epsilon/2 \ .$$

Let $\kappa = \min\{\kappa_1, \kappa_2\}$. Then, for any $\epsilon > 0$,

$$\|z_i - \widetilde{z}_i\| < \kappa \implies \tau(z_i, \widetilde{z}_i) < \epsilon/2 \quad \text{and} \quad \tau(\widetilde{z}_i, z_i) < \epsilon/2 \ .$$

This statement is depicted in Figure 2 for the pairs $(z_{i-1}, \widetilde{z}_{i-1})$ and $(z_{i+1}, \widetilde{z}_{i+1})$. Because γ is optimal,

$$\begin{aligned} \tau(z_{i-1}, z_{i+1}) &\leq \tau(z_{i-1}, \widetilde{z}_{i-1}) + \tau(\widetilde{z}_{i-1}, \widetilde{z}_{i+1}) + \tau(\widetilde{z}_{i+1}, z_{i+1}) \\ &\leq \tau(\widetilde{z}_{i-1}, \widetilde{z}_{i+1}) + \epsilon \ . \end{aligned} \tag{15}$$

Similarly, because $\widetilde{\gamma}$ is optimal,

$$\begin{aligned} \tau(\widetilde{z}_{i-1}, \widetilde{z}_{i+1}) &\leq \tau(\widetilde{z}_{i-1}, z_{i-1}) + \tau(z_{i-1}, z_{i+1}) + \tau(z_{i+1}, \widetilde{z}_{i+1}) \\ &\leq \tau(z_{i-1}, z_{i+1}) + \epsilon \ . \end{aligned} \tag{16}$$

Inequalities (15)-(16) furnish continuity of τ. □

4.3 Midpoint maps and total cost

Consider $(z_{i-1}, z_{i+1}) \in D$, where D is as in (13). We define a *midpoint map* to be any $M : \mathbb{R}^n \times \mathbb{R}^n \longrightarrow \mathbb{R}^n$ such that $M(z_{i-1}, z_{i+1}) = \widehat{z}_i$, where $\widehat{z}_i$ is the midpoint between z_{i-1} and z_{i+1}.

Let $X \subseteq \mathbb{R}^{2n}$, and Y be the set of all $(q+1)$-tuples $y = (y_0, y_1, \ldots, y_q) \in X^{q+1}$ such that $y_i = (z_i, \lambda_i)$ and $\tau(z_i, z_{i+1}) \leq \delta$ for all $i = 0, 1, \ldots, q-1$.

For $1 \leq p < q$ define $G_p : Y \longrightarrow X^{q+1}$ by

$$G_p(y) = (y_0, \ldots, y_{p-1}, \widehat{y}_p, y_{p+1}, \ldots, y_q)$$

where $\widehat{y}_p = (\widehat{z}_p, \widehat{\lambda}_p)$, $\widehat{z}_p = M(z_{p-1}, z_{p+1})$, and $\widehat{\lambda}_p$ is given by one of the two cases below.

(i) For $p = 0, 1, \ldots, (q-3)$, $\widehat{\lambda}_p = \psi(\widehat{t}_p)$, the initial costate associated with Problem (P_{p+1}),

(ii) For $p = (q-2), (q-1), q$, $\widehat{\lambda}_p = \psi(\widehat{t}_p)$, costates associated with Problem (P_{q-1}).

It should be noted that the update $\widehat{y}_p$, $p = 0, 1, \ldots, (q-3)$ is obtained by solving two subproblems; namely Problem (P_p) is solved to get $\widehat{z}_p$, and Problem (P_{p+1}) is solved to get $\widehat{\lambda}_p$.

Lemma 4.3 $G_p(y) \in Y$.

Proof. $\tau(z_{p-1}, \widehat{z}_p) = \tau(z_{p-1}, z_{p+1})/2 \leq (\tau(z_{p-1}, z_p) + \tau(z_p, z_{p+1}))/2 \leq (2\delta)/2$. □

So $G_p : Y \longrightarrow Y$ where $1 \leq p < q$. Define $F : Y \longrightarrow Y$ as the composite

$$F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1 \ .$$

The $(q+1)$-tuple $\widehat{y} = (\widehat{y}_0, \widehat{y}_1, \ldots, \widehat{y}_q) = F(y) \in Y$ can be defined alternatively by

(i) $\widehat{y}_0 = (\widehat{z}_0, \widehat{\lambda}_0)$ where $\widehat{z}_0 = z_0$, and $\widehat{\lambda}_0 = \psi(t_0)$ is given by the optimal control from z_0 to z_2;

(ii) $\widehat{y}_i = (\widehat{z}_i, \widehat{\lambda}_i)$ where $\widehat{z}_i = M(\widehat{z}_{i-1}, z_{i+1})$, and $\widehat{\lambda}_i = \psi(\widehat{t}_i)$ is given by the optimal control from $\widehat{z}_i$ to z_{i+2}, for $1 \leq i \leq q-2$;

(iii) $\widehat{y}_{q-1} = (\widehat{z}_{q-1}, \widehat{\lambda}_{q-1})$ where $\widehat{z}_{q-1} = M(\widehat{z}_{q-2}, z_q)$, and $\widehat{\lambda}_{q-1} = \psi(\widehat{t}_{q-1})$ is given by the optimal control from $\widehat{z}_{q-2}$ to z_q;

(iv) $\widehat{y}_q = (\widehat{z}_q, \widehat{\psi}_q)$ where $\widehat{z}_q = z_q$, and $\widehat{\psi}_q = \psi(\widehat{t}_q)$ is given by the optimal control from $\widehat{z}_{q-2}$ to z_q.

Let Z be the set all $(q+1)$-tuples $z = (z_0, z_1, \ldots, z_q)$. The $(q+1)$-tuple z is said to have cost $\alpha(z)$ where $\alpha : Z \longrightarrow \mathbb{R}$ is defined by

$$\alpha(z) = \sum_{i=1}^{q} \tau(z_{i-1}, z_i) \ .$$

Note that $\alpha(z) = \int_{t_0}^{t_f} f_0(x,u)\,dt$ is the cost incurred along the piecewise-optimal trajectory defined by the $(q+1)$-tuple z. The function α is continuous, because τ is continuous.

Let the local cost function $\tau : D \longrightarrow \mathbb{R}$ be extended as $\widetilde{\tau} : \widetilde{D} \longrightarrow \mathbb{R}$ where

$$\widetilde{D} = \{(x, \widetilde{x}) \in \mathbb{R}^n \times \mathbb{R}^n : \widetilde{x} \in \mathcal{R}(x, < \Delta)\}$$

and Δ is not necessarily small, such that $\widetilde{\tau}(x, \widetilde{x})$ is the infimum of the cost to get from x to $\widetilde{x}$. Then

$$\widetilde{\tau}(z_0, z_q) \leq \alpha(y) \ .$$

Define the state-space projection map $\pi_1 : Y \longrightarrow Z$ such that

$$\pi_1 \circ G_p(y) = (z_0, z_1, \ldots, \widehat{z}_p, \ldots, z_q)$$

and that $\pi_1 \circ F(y) = (\widehat{z}_0, \widehat{z}_1, \ldots, \widehat{z}_q) =: \widehat{z}$.

Lemma 4.4 *For $1 \leq p < q$ and all $y \in Y$*

$$\alpha(\pi_1 \circ G_p(y)) \leq \alpha(z) \ .$$

Proof.

$$\begin{aligned} \alpha(z) - \alpha(\pi_1 \circ G_p(y)) &= \tau(z_{p-1}, z_p) + \tau(z_p, z_{p+1}) - \tau(z_{p-1}, \widehat{z}_p) - \tau(\widehat{z}_p, z_{p+1}) \\ &= \tau(z_{p-1}, z_p) + \tau(z_p, z_{p+1}) - \tau(z_{p-1}, z_{p+1}) \\ &\geq 0 \ . \end{aligned}$$

□

If one considers the composite $F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1$ in Lemma 4.4 instead of G_p, then the following lemma is obtained.

Lemma 4.5 *For all $y \in Y$*

$$\alpha(\pi_1 \circ F(y)) \leq \alpha(z) \ .$$

5 Convergence Analysis

5.1 Compactness of the state-costate space

Define the set $C(x_0, r)$ of all final states $x_f = x(t_f)$ and costates $\psi_f = \psi(t_f)$, such that the states x_f can be reached from $x_0 = x(t_0)$ over $[t_0, t_f]$ with a minimum cost less than or equal to $r \geq 0$; namely

$$C(x_0, r) = \{(x_f, \psi_f) \in \mathbb{R}^{2n} : \widetilde{\tau}(x_0, x_f) \leq r\} \ .$$

For each (x_0, x_f, ψ_f) above, the optimal trajectory pair $(x(t), \psi(t))$ and a corresponding control trajectory $u(t)$ satisfy Conditions (2)-(5) with $x(t_0) = x_0$, $x(t_f) = x_f$ and $\psi(t_f) = \psi_f$.

Let α_0 be the cost incurred along the initial feasible trajectory.

Assumption 5 *The set $C(x_0, \alpha_0)$ is bounded.*

Assumption 6 *The optimal terminal time t_f is finite.*

The following two examples verify Assumptions 5 and 6.

Example 4 On a Riemannian manifold, N, minimal geodesics between two sufficiently nearby given points in a compact subset S of N are contained within a bounded set $W \supset S$. The costates associated with the geodesic problem are also bounded by virtue of Equation (6) because the controls $u_i = \dot{x}_i$ are bounded. Furthermore $t_f = 1$ is fixed. □

Example 5 In the case of linear time-varying control systems the right-hand side of (1) is not bounded. However suppose that the optimal control can be expressed as a linear state feedback, and that it is substituted into the system equation to get a system dependent only on the state. It is a well-known fact that there is no *finite-escape time* for linear time-varying systems [21]. So the states are bounded. The associated costates are bounded likewise. The simple case of a fixed time-horizon is an example to Assumption 6. □

Let $y \in Y$ and define $y^{(k)} = F^k(y)$ for $k \geq 1$.

Lemma 5.1 *Y is compact.*

Proof. Since $C(x_0, \alpha_0)$ is bounded (by Assumption 5), the set Y is also bounded. Now one needs to show that Y is closed as well. Consider $y = (y_0, y_1, \ldots, y_q) \in Y$. Note that $z_i^{(k)} \in \mathcal{R}(z_{i-1}^{(k)}, \leq \delta)$ for the kth iterate $y^{(k)} = F(y^{(k-1)})$. Since the closure $\overline{C}(x_0, \alpha_0)$ is compact, one can take a convergent subsequence $\{y^{(k_j)} : j \geq 1\}$ of the sequence $\{y^{(k)} : k \geq 0\} \subset Y$. Write

$$\lim_{j \longrightarrow \infty} y^{(k_j)} = y^{(\infty)} \ , \tag{17}$$

and so

$$\lim_{j \longrightarrow \infty} z^{(k_j)} = z^{(\infty)} \ ,$$

where

$$y^{(\infty)} = \left(y_0^{(\infty)}, y_1^{(\infty)}, \ldots, y_q^{(\infty)} \right)$$

and

$$z^{(\infty)} = \left(z_0^{(\infty)}, z_1^{(\infty)}, \ldots, z_q^{(\infty)} \right) \ .$$

Recall that for $1 \leq i \leq q$

$$\tau(z_{i-1}^{(k)}, z_i^{(k)}) \leq \delta \ .$$

By taking limits and using the continuity of τ, we get

$$\tau(z_{i-1}^{(\infty)}, z_i^{(\infty)}) \leq \delta$$

which implies that $y^{(\infty)} \in Y$. □

Lemma 5.2 *Each sequence* $\{t_i^{(k)} : k \geq 0\}$, $i = 1, \ldots, q$, *has a convergent subsequence with the same index set as that of* $\{y^{(k_j)} : j \geq 1\}$ *such that* $\lim_{j \longrightarrow \infty} y^{(k_j)} = y^{(\infty)}$.

Proof. The conclusion follows from Assumption 6 and the limit expression in (17). □

Lemma 5.3 *The sequence* $\{\alpha(z^{(k)}) : k \geq 1\}$ *converges to its infimum* $\alpha^{(\infty)}$. *In particular*

$$0 < \alpha(z^{(\infty)}) = \alpha^{(\infty)} \in [\widetilde{\tau}(z_0, z_q), q\delta] \ .$$

Proof. The result follows from the continuity of α, and Lemmas 5.1 and 4.5. □

5.2 Extreme points and leapfrog splicing

Recall that by Assumption 2, the curve $\gamma_{z_{i-1},z_{i+1}} : [t_{i-1}, t_{i+1}] \longrightarrow \mathbb{R}^n$ is the unique optimal trajectory from z_{i-1} to z_{i+1}. The point $y_i = (z_i, \lambda_i)$ is said to be *between* y_{i-1}, y_{i+1}, if z_i lies in the image of $\gamma_{z_{i-1},z_{i+1}}$. Note that, trivially, y_{i-1} and y_{i+1} are between y_{i-1} and y_{i+1}.

The $(q+1)$-tuple $y \in Y$ is said to be *extreme* when y_i is between y_{i-1}, y_{i+1} for all $1 < i < q$.

Let $\mu_i : [t_i, t_{i+1}] \longrightarrow \mathbb{R}^{2n}$ be the optimal trajectory pair $(\gamma_{z_{i-1},z_i}(\cdot), \psi(\cdot))$ solving the optimality system associated with Problem (P_i) from z_{i-1} to z_i, $i = 1, \ldots, q$. Then $\mu_y : [t_0, t_f] \longrightarrow \mathbb{R}^{2n}$ is defined as the concatenation (or track-sum) of $\mu_1, \mu_2, \ldots, \mu_q$ in the given order such that $\mu_y(t_0) = \mu_1(t_0)$, $\mu_y(t_i) = \mu_i(t_i) = \mu_{i+1}(t_i)$, $i = 1, \ldots, q-1$, and $\mu_y(t_f) = \mu_q(t_q)$. It is convenient to use a special symbol, for example $\smile$, for this kind of concatenation. The trajectory μ_y can then be expressed by

$$\mu_y := \mu_1 \smile \mu_2 \smile \cdots \smile \mu_q \ .$$

Lemma 5.4 *If* y *is extreme, then* μ_y *is a critical trajectory.*

Proof. Suppose $y \in Y$ is extreme. We will use induction on, what we call, the *leapfrog splicing* of trajectories, which is illustrated in Figure 3, to prove the lemma. We define

$$\mu_y^{(p)} := \mu_1 \smile \mu_2 \smile \cdots \smile \mu_p$$

and aim to prove the following: $\mu_y^{(p)}$ is critical with the fixed end points $x(t_0) = z_0$ and $x(t_p) = z_p$, for $2 \leq p \leq q$, where q is provided.

Suppose $p = 2$. Then $\mu_y^{(2)} = \mu_1 \smile \mu_2$. Because y_1 is between y_0 and y_2, z_1 is in the image of the optimal trajectory γ_{z_0,z_2}. Therefore there exists some nontrivial $\psi(\cdot)$ such that $\mu_y^{(2)} = (\gamma_{z_0,z_2}(\cdot), \psi(\cdot))$ solves the associated optimality system. So $\mu_y^{(2)}$ is critical.

Assume that $\mu_y^{(k)} = \mu_1 \smile \mu_2 \smile \cdots \smile \mu_k$ is critical. The next leapfrog step from z_{k-1} to z_{k+1} gives the local optimal trajectory segment $\gamma_{z_{k-1},z_{k+1}}(\cdot)$ and the associated costate trajectory $\widehat{\psi}(\cdot)$. So $(\widehat{\gamma}_{z_{k-1},z_{k+1}}(\cdot), \widehat{\psi}(\cdot)) = \widehat{\mu}_k \smile \mu_{k+1}$. By the induction hypothesis, $\mu_k =$

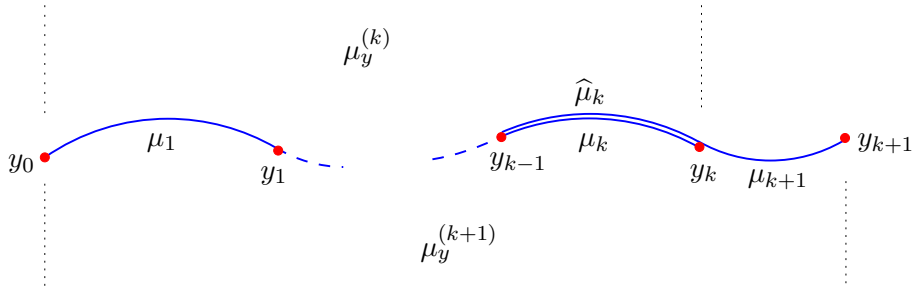


Figure 3: Concatenation of the local optimal trajectories μ_i, and the leapfrog splicing.

$(\gamma_{z_{k-1},z_k}(\cdot), \psi(\cdot))$ is optimal. Because y_k is between y_{k-1} and y_{k+1}, $\widehat{\mu}_k = (\widehat{\gamma}_{z_{k-1},z_k}(\cdot), \widehat{\psi}(\cdot))$ is also optimal. By the local uniqueness of the controls and costates posed in Assumptions 2 and 3, $\gamma_{z_{k-1},z_k}(\cdot) = \widehat{\gamma}_{z_{k-1},z_k}(\cdot)$ and $\psi(\cdot) = \widehat{\psi}(\cdot)$. Therefore $\mu_k = \widehat{\mu}_k$. By "splicing" the trajectory segments $\mu_1 \smile \mu_2 \smile \cdots \smile \mu_k$ and $\widehat{\mu}_k \smile \mu_{k+1}$, by means of the overlap $\widehat{\mu}_k = \mu_k$ between y_{k-1} and y_k (see Figure 3), we obtain the trajectory

$$\mu_y^{(k+1)} = \mu_1 \smile \mu_2 \smile \cdots \smile \mu_k \smile \mu_{k+1}$$

which satisfies the optimality system associated with the optimal control problem with the end conditions $x(t_0) = z_0$ and $x(t_{k+1}) = z_{k+1}$. So $\mu_y^{(k+1)}$ is critical. □

Remark 5.1 *The converse of Lemma* 5.4 *holds if μ_y is not only critical but also (locally) optimal.*

Lemma 5.5

(a) If y is extreme and all z_i, $i = 1, \ldots, q-1$, are midpoints, then $F(y) = y$.

(b) If $F(y) = y$ then y is extreme.

Proof. **(a)** Suppose that y is extreme and all z_p, $p = 1, \ldots, q-1$, are midpoints, that is $z_p = \widehat{z}_p = M(z_{p-1}, z_{p+1})$. One further has $\lambda_p = \widehat{\lambda}_p$, because, using the notation of the proof of Lemma 5.4, $\widehat{\mu}_{p-1} \smile \widehat{\mu}_p = \mu_{p-1} \smile \mu_p$ by the local uniqueness of optimal trajectories. Therefore $y_p = \widehat{y}_p = G_p(y)$, $p = 1, \ldots, q-1$. The composition $F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1$ then gives $F(y) = \widehat{y} = y$.
(b) Suppose $F(y) = y$. Recall that $\widehat{z}_0 = y_0$, $\widehat{z}_p = M(\widehat{z}_{p-1}, z_{p+1})$ for $p = 1, \ldots, q-1$, and $\widehat{z}_q = z_q$, where M is the midpoint map. Now since $F(y) = y$,

$$z_p = \widehat{z}_p = M(z_{p-1}, z_{p+1}) \ .$$

So y_p is between y_{p-1}, y_{p+1}, $p = 1, \ldots, q-1$. □

Lemma 5.6 *y is extreme if and only if $\mu_{F(y)} = \mu_y$.*

Proof. Suppose that y is extreme. Because y_p is between y_{p-1} and y_{p+1}, $\widehat{y}_p = G_p(y)$ is also between y_{p-1} and y_{p+1}. So $\mu_{G_p(y)} = \mu_y$, by the uniqueness of $\gamma_{z_{i-1},z_{i+1}}$. The composition $F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1$ gives $\mu_{F(y)} = \mu_y$.

Suppose $\mu_{F(y)} = \mu_y$. Then $\mu_{G_1(y)} = \mu_y$. Using the notation of the proof of Lemma 5.4, $\widehat{\mu}_1 \smile \widehat{\mu}_2 = \mu_1 \smile \mu_2$. So $\widehat{y}_1$ is between y_0 and y_1. The composition $F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1$ yields that y is extreme. □

Lemma 5.7 *y is extreme if and only if $\alpha(\pi_1 \circ F(y)) = \alpha(z)$.*

Proof. Suppose that y is extreme. Then $\alpha(\pi_1 \circ F(y)) = \alpha(z)$ by Lemma 5.6.
Suppose that $\alpha(\pi_1 \circ F(y)) = \alpha(z)$. Recall

$$\alpha(z) = \tau(z_0, z_1) + \cdots + \tau(z_{p-1}, z_p) + \tau(z_p, z_{p+1}) + \cdots + \tau(z_{q-1}, z_q) \ .$$

For $0 < p < q$,

$$\alpha(\pi_1 \circ G_p(y)) = \tau(z_0, z_1) + \cdots + \tau(z_{p-1}, \widehat{z}_p) + \tau(\widehat{z}_p, z_{p+1}) + \cdots + \tau(z_{q-1}, z_q) \ .$$

Then

$$\tau(z_{p-1}, \widehat{z}_p) + \tau(\widehat{z}_p, z_{p+1}) = \tau(z_{p-1}, z_p) + \tau(z_p, z_{p+1}) \ . \tag{18}$$

By the definition of the midpoint map, $\widehat{z}_p$ lies in the image of $\gamma_{z_{p-1}, z_{p+1}}$. Equation (18) implies that z_p lies in a trajectory of equal cost, and this trajectory can only be $\gamma_{z_{p-1}, z_{p+1}}(\cdot)$, by uniqueness. Therefore y_p is between y_{p-1} and y_{p+1}. The composition $F = G_{q-1} \circ G_{q-2} \circ \cdots \circ G_1$ then provides that y is extreme. □

5.3 Convergence to a critical trajectory

Lemma 5.8 *$y^{(\infty)}$ is extreme.*

Proof. By Lemmas 4.5 and 5.3, $\alpha(\pi_1 \circ F(y^{(\infty)})) = \alpha^{(\infty)} = \alpha(\pi_1(y^{(\infty)}))$. Then the result follows by Lemma 5.7. □

Lemma 5.9 *$\mu_{y^{(\infty)}}$ is a critical trajectory.*

Proof. Lemmas 5.8 and 5.4 furnish the lemma. □

Theorem 5.1 *The leapfrog iterations converge to a critical trajectory $\mu_{y^{(\infty)}}$.*

Proof. Lemmas 5.8 and 5.6 imply that $\mu_{F(y^{(\infty)})} = \mu_{y^{(\infty)}}$, which together with Lemma 5.9 prove the theorem. □

Although Theorem 5.1 guarantees convergence of the trajectory iterates $\mu_{y^{(k)}}$, no assertion has yet been made about convergence of the iterates $y^{(k)}$. Convergence of $y^{(k)}$ is particularly important in a numerical implementation. If it is known that $y^{(k)}$ is convergent too, then instead of using an infinite-dimensional norm (for example the L^∞ norm) to measure the change from the trajectory iterate $\mu_{y^{(k)}}$ to $\mu_{y^{(k+1)}}$, a finite-dimensional norm (for example the Euclidean norm) can be used to measure the change from the iterate $y^{(k)}$ to $y^{(k+1)}$, which is much less demanding computationally. In what follows, we will give a lemma, and Theorem 5.2, which guarantees convergence of the iterates $y^{(k)}$.

Lemma 5.10 *If y is extreme, then all $z_i^{(\infty)}$, $i = 1, \ldots, q-1$, are uniformly distributed with respect to cost, namely*

$$\tau(z_{i-1}^{(\infty)}, z_i^{(\infty)}) = \tau(z_i^{(\infty)}, z_{i+1}^{(\infty)}) \ ,$$

$i = 1, \ldots, q-1$.

Proof. Let y be extreme. Then by Lemma 5.4, μ_y is critical. Suppose that μ_y is parameterized proportionally to the cost so that $\mu_y : [0,1] \longrightarrow \mathbb{R}^{2n}$, with $\mu_y(0) = y_0$ and $\mu_y(1) = y_q$. Note that, by Lemma 5.6, $\mu_{F(y)} = \mu_y$. Then it suffices to show that the sequence $\{F^k(y) : k \geq 1\}$ converges to the uniformly distributed $(q+1)$-tuple

$$(y_0, \mu_y(1/q), \mu_y(2/q), \ldots, \mu_y(i/q), \ldots, y_q) \ .$$

The rest of the proof follows similar lines to those given by [30, Lemma 3.2], which we adapt here using our notation and definitions.

If $q = 2$ the result is clear since $F(y)$ is independent of y_1 and the limit is achieved immediately as $F(y)$. For $q > 2$ we give some definitions first, and then prove the convergence to the required $(q+1)$-tuple.

Let $y^{(0)} = y$, and $y^{(k)} = F^k(y)$ for $k = 1, 2, \ldots$. Since $\mu_{y^{(k)}} = \mu_{y^{(0)}}$ by Lemma 5.6,

$$y^{(k)} = (\mu(0), \mu(t_1^{(k)}), \mu(t_2^{(k)}), \ldots, \mu(t_{q-1}^{(k)}), \mu(t_q^{(k)})) \, ,$$

for $k = 0, 1, 2, \ldots$, where $t_i^{(k)}$, $i = 1, \ldots, q-1$, are the junction times. Since μ_y is parameterized proportionally to the cost, application of F gives the recursion

$$t_i^{(k)} = (t_{i-1}^{(k)} + t_{i+1}^{(k-1)})/2 \, ,$$

for $i = 1, \ldots, q-1$, where $t_0^{(k)} = 0$, $t_q^{(k)} = 1$. Write

$$t^{(k)} = [t_2^{(k)}, t_3^{(k)}, \ldots, t_{q-1}^{(k)}]^T \in \mathbb{R}^{q-2} \ .$$

Then $t^{(k)} = A\, t^{(k-1)} + b$, where

$$A = \begin{bmatrix} 1/2^2 & 1/2 & 0 & \cdots & 0 \\ 1/2^3 & 1/2^2 & 1/2 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ 1/2^{q-1} & 1/2^{q-2} & 1/2^{q-3} & \cdots & 1/2^2 \end{bmatrix}$$

and $b = [0, 0, \ldots, 0, 1/2]^T \in \mathbb{R}^{q-2}$. It is straightforward to obtain

$$t^{(k)} = b + Ab + A^2 b + \ldots + A^{k-1} b + A^k t^0 \ ,$$

which converges since $\|A\| < 1$. So the limit $t^{(\infty)}$ satisfies $t^{(\infty)} = A\, t^{(\infty)} + b$. It is seen that

$$t^{(\infty)} = [2, 3, ..., q-1]^T / q$$

is the unique solution, which completes the proof. □

Theorem 5.2 $\{F(y^{(k)}) : k \geq 0\}$ *has a unique accumulation point.*

Proof. By Lemmas 5.8, 5.10 and 5.5(a), $F(y^{(\infty)}) = y^{(\infty)}$. □

6 Numerical Implementation

We will illustrate a numerical implementation of leapfrog on an optimal control problem with fixed terminal time. Recall Remark 2.2. For a fixed terminal time problem, one solves the subproblems (P_i) with both t_{i-1} and t_{i+1} fixed. At the start of the algorithm, the values $t_{i-1}^{(0)}$ and $t_{i+1}^{(0)}$ are given by a partition between $x_{n+1}(0) = 0$ and $x_{n+1}(t_f) = 1$: one typically takes a regular partition of the time interval $[0, 1]$. Then, given $t_{i-1}^{(k+1)}$ and $t_{i+1}^{(k)}$, the time iterate $t_i^{(k+1)}$ is determined through the solution of (P_i) as the time at which the midpoint $z_i^{(k+1)}$ is placed.

6.1 The van der Pol system

The van der Pol system has been used in earlier control applications, including time-optimal control problems where the control regime is bang–bang [18, 27]. Here we consider the van der Pol system with unbounded control,

$$\dot{x}_1(t) = x_2(t) \ , \tag{19}$$
$$\dot{x}_2(t) = -x_1(t) - (x_1^2(t) - 1)\, x_2(t) + u(t) \ , \tag{20}$$

where the problem is to get from $x(0) = (-2, -3)$ to $x(1) = (5, 0)$, minimizing the quadratic cost

$$\frac{1}{2}\int_0^1 (x_1^2(t) + x_2^2(t) + u^2(t))dt \ .$$

The Hamiltonian is

$$H(x, \psi, u) = (x_1^2 + x_2^2 + u^2)/2 + \psi_1 x_2 + \psi_2[-x_1 - (x_1^2 - 1)x_2 + u] \ .$$

The Maximum Principle yields, in addition to the system equations (19)-(20), the following costate equations

$$\dot{\psi}_1(t) = -x_1(t) + (1 + 2x_1(t)x_2(t))\, \psi_2(t) \ , \tag{21}$$
$$\dot{\psi}_2(t) = -x_2(t) - \psi_1(t) + (x_1^2(t) - 1)\, \psi_2(t) \ . \tag{22}$$

Furthermore optimal control has to satisfy

$$u(t) = -\psi_2(t) \ . \tag{23}$$

Substitution of (23) gives rise to the following TPBVP.

$$\dot{x}_1 = x_2 \ , \tag{24}$$
$$\dot{x}_2 = -x_1 - (x_1^2 - 1)\, x_2 - \psi_2 \ , \tag{25}$$
$$\dot{\psi}_1 = -x_1 + (1 + 2x_1 x_2)\, \psi_2 \ , \tag{26}$$
$$\dot{\psi}_2 = -x_2 - \psi_1 + (x_1^2 - 1)\, \psi_2 \ , \tag{27}$$
$$\text{with} \quad x(0) = (-2, -3) \quad \text{and} \quad x(1) = (5, 0) \ .$$

6.1.1 Solution with leapfrog

Leapfrog algorithm owes its working largely to the solutions of Subproblems (P_i). These subproblems for the van der Pol problem have been reduced to Equations (24)-(27) with the accompanying boundary conditions. A solution to these equations can be found by using simple shooting, because the leapfrog partition points are initially chosen close enough to one another so that affine approximations provide good initial guesses for simple shooting.

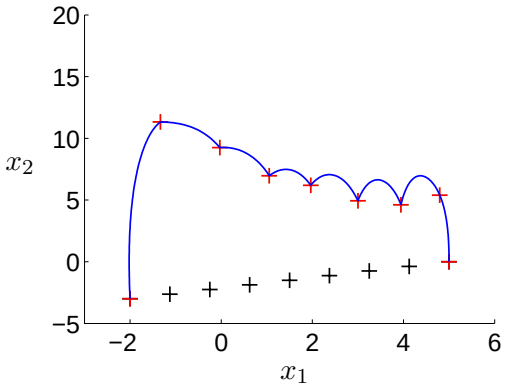
A MATLAB code has been written to implement the leapfrog algorithm. For solving the ODEs (24)-(27), the MATLAB function `ode45` is utilised with the absolute and relative tolerances of 10^{-8}. The simple shooting technique we have programmed incorporates Newton's method with backtracking [8]. This increases the chances of simple shooting to converge; however if the number of iterations within simple shooting exceeds 10, then we regard simple shooting to have failed. In the final iteration of leapfrog the relative and absolute tolerances for `ode45` are sharpened to 10^{-10}.

Iterations of the leapfrog algorithm in the state space are shown in Figure 4. Figure 4(a) displays both the initial partition we start with and the concatenation of the locally optimal trajectories, $\mu_z^{(1)}$, obtained as a result of the first leapfrog iteration. The initial points $z_i^{(0)}$ have been chosen to be equally spaced along the straight line between x_0 and x_T. For ease of bookkeeping in the code we consider powers of 2 for the number of subdivisions, i.e. we use $q = 2, 4, 8$, etc. In each iteration we use the smallest possible q with which each Subproblem (P_i) is solved successfully. This eventually leads to a final iteration where $q = 2$ and simple shooting is used to achieve the solution.

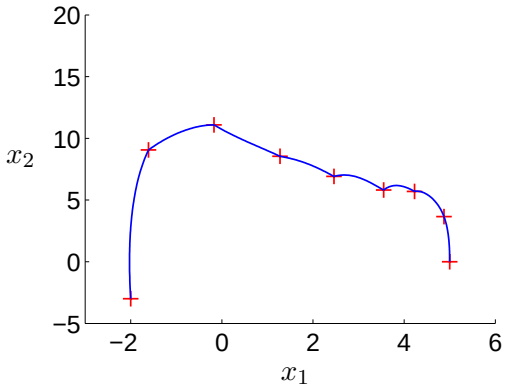
With leapfrog, different positioning of the midpoints is permissible, as long as the subintervals of the consequent subproblems overlap. So, in the implementation, the midpoint $z_i^{(k+1)}$ is chosen as the point which is reached in roughly half the time along the optimal trajectory from $z_{i-1}^{(k+1)}$ to $z_{i+1}^{(k)}$. This choice of midpoints is computationally more economical than the choice of midpoint where *exactly* half the cost is incurred.

First to fifth iterations could all be successfully completed by taking at least eight subdivisions. Sixth to ninth iterations could be carried out with minimum four subdivisions, where the new partition is formed in the sixth iteration by taking the points $z_2^{(5)}$, $z_4^{(5)}$ and $z_6^{(5)}$ from iteration 5. The trajectory $\mu_z^{(3)}$ and the updated partition points $z_i^{(3)}$, $i = 1, 2, 3$, are illustrated in Figure 4(c). The tenth iteration, after reducing the number of partitions to two, or simply applying simple shooting between x_0 and x_T with the initial guess value $\psi(0) = \lambda_0^{(9)}$, yields the solution trajectory for the problem, which is depicted in Figure 4(d). The costs along each of the leapfrog trajectory iterates $\mu_z^{(k)}$, $k = 1, 3, 6, 10$, are reported in Figures 4(a)-(d), respectively. We also report the CPU times, cumulatively at each iteration, as measured within MATLAB, running on (single user) Windows XP Professional (Version 5.1) operating system with a 2.00 GHz Intel Pentium M processor and 1 GB of RAM.

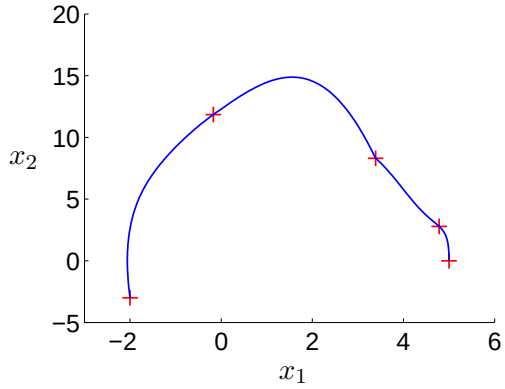
The first leapfrog iteration can also be viewed as one where a trajectory is constructed in a simple way and verified to be feasible. Because of the Newton backtracking, this iteration takes more than twice longer than each of the rest of the iterations except the last one. The last iteration, where we reduce the whole process to simple shooting, also takes longer because of the Newton backtracking and because we require a higher accuracy.



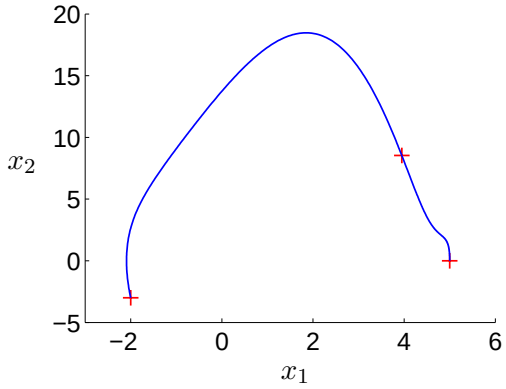
(a) $\mu_z^{(1)}$ with $q = 8$ and cost 3043.74. (CPU time: 2.6 s)



(b) $\mu_z^{(3)}$ with $q = 8$ and cost 2189.33. (CPU time: 4.7 s)



(c) $\mu_z^{(6)}$ with $q = 4$ and cost 1151.34. (CPU time: 8.0 s)



(d) $\mu_z^{(10)}$ with $q = 2$ and cost 993.67. (CPU time: 14.2 s)

Figure 4: Locally optimal state trajectories $\mu_z^{(k)}$ during leapfrog iterations for the van der Pol system. Crosses in iteration k represent partition points $z_i^{(k)}$, $i = 0, 1, \ldots, q$.

The solution trajectories of the state, costate and control variables as functions of time can be seen in Figures 5(a)-(b) and 6. In Figure 6, simply $u(t) = -\psi_2(t)$. The initial costate vector $\psi(0) = (-113.817881, -40.950820)$ generates an accurate solution.

Some of the locally optimal costate trajectories obtained in the leapfrog iterations are shown in Figure 7.

Overall it appears that there are pros and cons with using leapfrog. Numerical evidence arising from our application of leapfrog to the van der Pol problem point to certain advantages and disadvantages of leapfrog.

Advantage 1 *With leapfrog, one only has to choose the starting values of the states. These values can usually be chosen in a practical manner (for example, along a straight line joining x_0 and x_T), which could then generate a feasible trajectory. In the leapfrog scheme, it is not necessary to guess the starting values of costates. For solving the subproblems in leapfrog, affine approximations usually give reasonable starting values for the costates.*

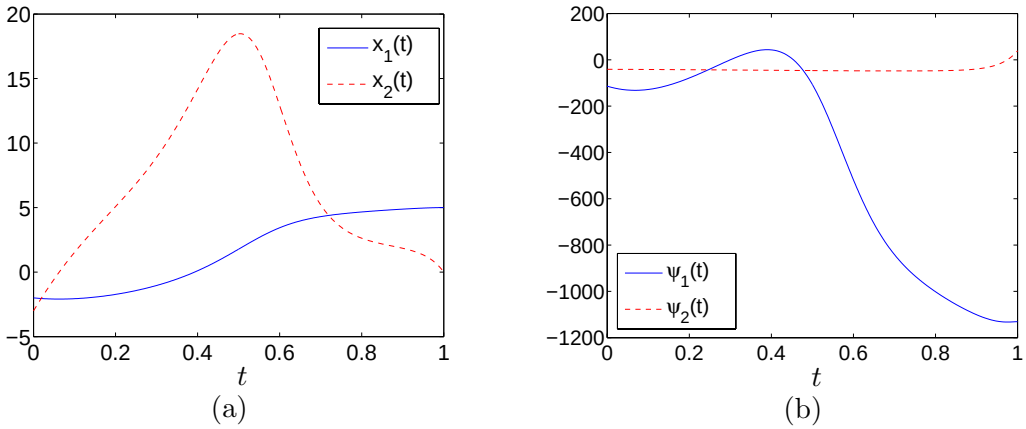


Figure 5: The solution state and costate trajectories for the van der Pol system.

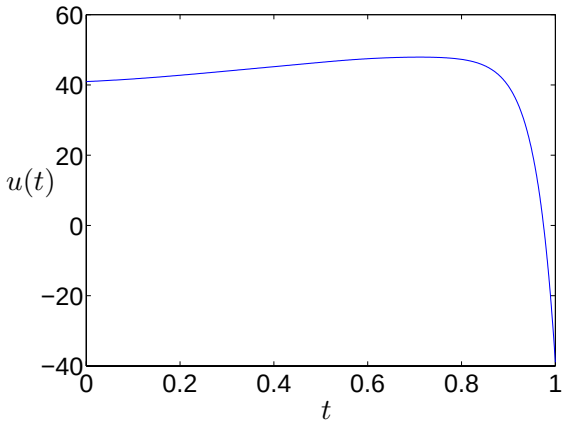
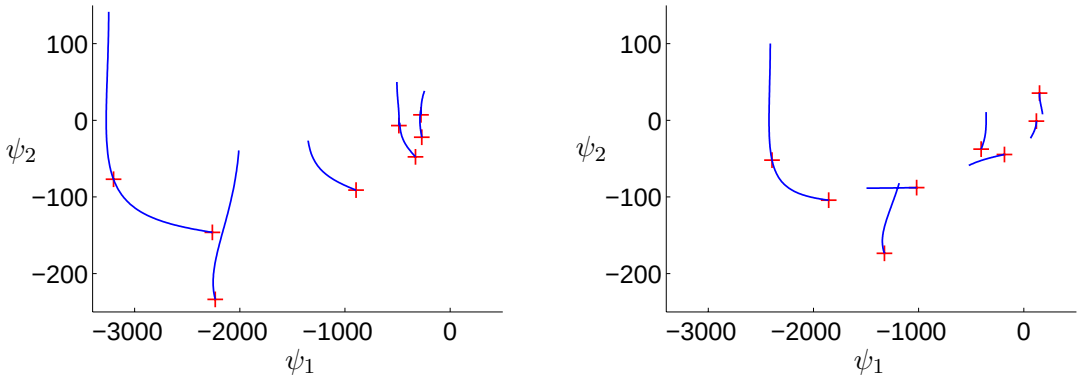
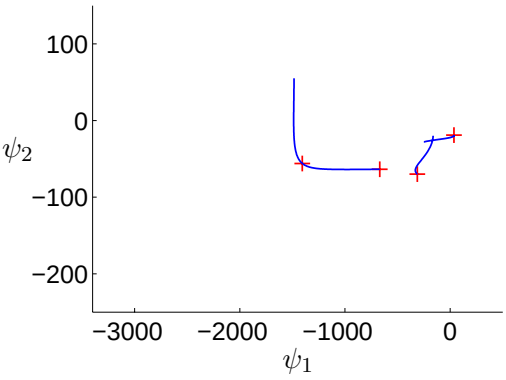


Figure 6: The solution control trajectory for the van der Pol system.

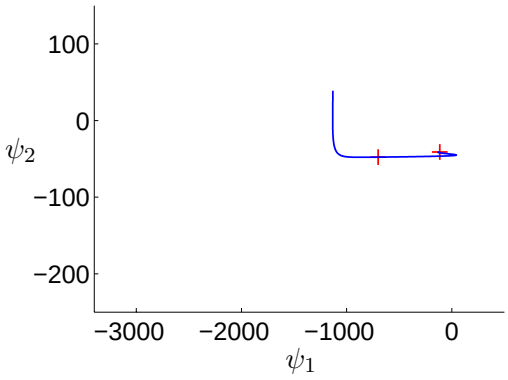


(a) $\mu_\lambda^{(1)}$ with $q = 8$.

(b) $\mu_\lambda^{(3)}$ with $q = 8$.



(c) $\mu_\lambda^{(6)}$ with $q = 4$.



(d) $\mu_\lambda^{(10)}$ with $q = 2$.

Figure 7: Locally optimal costate trajectories during leapfrog iterations for the van der Pol system. Crosses in iteration k represent partition points $\lambda_i^{(k)}$, $i = 0, 1, \ldots, q-1$.

Advantage 2 *At each iteration, leapfrog generates a locally optimal trajectory $\mu_z^{(k)}$, which is feasible. Suppose that the time allowed to design an optimal control for the system is not long enough to compute a solution. Then leapfrog may provide at least a suboptimal control. For example, if it is necessary to obtain a solution in less than 10 seconds, then, within the computational means we have in this article, we may offer the suboptimal solution depicted in Figure 4(c), which is far better than the feasible solution given in Figure 4(a). One restriction on the applicability of the suboptimal solution found by leapfrog is that the control $u(t) = -\psi(t)$ is discontinuous, because the piecewise optimal $\psi(t)$ obtained in the third iteration is discontinuous, as can be seen in Figure 7(c). However, many dynamical systems allow discontinuous controls; so the suboptimal control generated by leapfrog can possibly be implemented.*

An apparent drawback of leapfrog is that, as the solution is approached, it is observed from the numerical experiments that, with a fixed number of partitions $q > 2$, the leapfrog updates get sluggish. Therefore, q needs to be reduced progressively (eventually down to $q = 2$) for convergence in a reasonable number of iterations. This effort may in general incur an additional computational cost.

6.2 Further comments on implementation of leapfrog

We observe that the solution for the problem in the previous subsection is obtained in ten iterations. It is worthwhile to note that the same problem with different initial and target states may require different number of subdivisions at the start, and reducing the number of subdivisions may take a lot longer than what is presented here. The performance of the algorithm would depend not only on the difficulty of the problem, but also on the way the algorithm is implemented. Two particular issues seem to be of main concern in any implementation of leapfrog, namely (i) the efficiency of simple shooting technique employed in Subproblems (P_i), and (ii) a dynamic way of doing the partitioning along a leapfrog trajectory iterate.

Because we were only concerned with providing an illustration of the leapfrog algorithm in this particular study, we have coded a rather straightforward simple shooting scheme. A more sophisticated simple shooting scheme may be necessary for more difficult problems. In particular, one should note that `ode45` is not necessarily the method of choice for the integration of the state-costate equations; it merely serves to illustrate an implementation of leapfrog. Otherwise, in a more elaborate implementation, a certain approximation order for both the states and costates can only be achieved if certain conditions are satisfied [12].

In the case when $q > 2$ is fixed, then convergence can usually not be obtained in a finite number of iterations. Furthermore if q is large, then the numerical experiments show that convergence becomes rather slow. So it is important to employ some kind of *dynamic* partitioning. A simple dynamic partitioning scheme was prescribed in the example application in Section 6.1. However more sophisticated schemes can also be employed. One can increase or decrease the number of partitions *locally* in a smaller segment of the leapfrog trajectory, as the need arises. Given a trajectory, it may be appropriate to increase the number of subdivisions in one segment of the trajectory, while it is appropriate to decrease the number in some other segment. The ultimate aim would

still remain as to reduce the number of iterations to just two, in which case simple shooting, if successful, would deliver the required solution.

References

[1] Ascher, U. M., & Chan, S. Y. P. (1991). On parallel methods for boundary value ODEs. *Computing*, **46**, pp. 1-17.

[2] Ascher, U. M., Mattheij, R. M. M., & Russell, R. D. (1995). *Numerical Solution of Boundary Value Problems for Ordinary Differential Equations*, SIAM Publications, Philadelphia.

[3] Betts, J. T. (2001). *Practical Methods for Optimal Control Using Nonlinear Programming*. SIAM Publications, Philadelphia, U. S. A..

[4] Bock, H. G., & Plitt, K. J. (1984). A multiple shooting algorithm for direct solution of optimal control problems. Proceedings of the 9th IFAC World Congress, Budapest, Hungary.

[5] Boothby, W. M. (1986). *An Introduction to Differentiable Manifolds and Riemannian Geometry*. Academic Press, London.

[6] Büskens, C. (1998). *Optimierungsmethoden und Sensitivitätsanalyse für optimale Steuerprozesse mit Steuer- und Zustands-Beschränkungen*. Dissertation, Institut für Numerische Mathematik, Universität Münster, 1998.

[7] Chen, Y., & Desrochers, A. A. (1993). Minimum-time control laws for robotic manipulators. *International Journal of Control*, **57**(1), 1-27.

[8] Dennis, J. E. Jr., & Schnabel, R. B. (1996). *Numerical Methods For Unconstrained Optimization and Nonlinear Equations*, SIAM Publications, Philadelphia.

[9] Deuflhardt, P. (1979). A stepsize control for continuation methods and its special application to multiple shooting techniques. Numerische Mathematik, **33**, pp. 115-146.

[10] Fraser-Andrews, G. (1999). A multiple shooting technique for optimal control. *Journal of Optimization Theory and Applications*, **102**(2), pp. 299-313.

[11] Grimm, W., & Markl, A. (1997). Adjoint estimation from a direct multiple shooting method. *Journal of Optimization Theory and Applications*, **92**(2), pp. 263-283.

[12] Hager, W. W. (2000). Runge-Kutta methods in optimal control and the transformed adjoint system. *Numerische Mathematik*, **87**, pp. 247-282.

[13] Kailath, T. (1980). *Linear Systems*. Prentice-Hall, Inc., Englewood Cliffs, N. J., U. S. A..

[14] Kaya, C. Y., Lucas, S. K., & Simakov, S. T. (2004). Computations for bang–bang constrained optimal control using a mathematical programming formulation. *Optimal Control Applications and Methods*, **25**(6), pp. 295-308.

[15] Kaya, C. Y., & Noakes, J. L. (1997). Geodesics and an optimal control algorithm. *Proceedings of the 36th IEEE Conference on Decision and Control (CDC)*, San Diego, California, U.S.A., December 1997, pp. 4918-4919.

[16] Kaya, C. Y., & Noakes, J. L. (1998a). The Leap-Frog Algorithm and optimal control: background and demonstration. *Proceedings of International Conference on Optimization and Applications (ICOTA '98)*, Perth, Australia, pp. 835-842.

[17] Kaya, C. Y., & Noakes, J. L. (1998b). The Leap-Frog Algorithm and optimal control: theoretical aspects. *Proceedings of International Conference on Optimization and Applications (ICOTA '98)*, Perth, Australia, pp. 843-850.

[18] Kaya, C. Y., & Noakes, J. L. (2003). Computational method for time-optimal switching control. *Journal of Optimization Theory and Applications*, **117**(1), pp. 69-92.

[19] Kaya, C. Y., & Martínez, J. M. (2007). Euler discretization and inexact restoration for optimal control. *Journal of Optimization Theory and Applications*, **134**(2), pp. 191-206.

[20] Keller, H. B. (1968). Numerical Methods for Two-Point Boundary-Value Problems, Blaisdell, London.

[21] Khalil, H. K. (1992). *Nonlinear Systems*, Maxwell Macmillan International, New York.

[22] Kramer, M. E., & Mattheij, M. M. (1993). Application of global methods in parallel shooting. *SIAM Journal on Numerical Analysis*, **30**(6), pp. 1723-1739.

[23] Kunkel, P., Mehrmann, V., & Stöver, R. (2005). Multiple shooting for unstructured nonlinear differential-algebraic equations of arbitrary index. *SIAM Journal on Numerical Analysis*, **42**(6), pp. 2277-2297.

[24] Ledzewicz, U., & Schättler, H. (2000). A high-order generalized local maximum principle. *SIAM Journal on Control and Optimization*, **38**(3), pp. 823-854.

[25] Leineweber, D. B., Bauer, I., Bock, H. G., & Schlöder, J. P. (2003). An efficient multiple shooting based reduced SQP strategy for large-scale dynamic process optimization. Part 1: theoretical aspects. *Computers and Chemical Engineering*, **27**, pp. 157-166.

[26] Maurer, H. (1976). Numerical solution of singular control problems using multiple shooting techniques. *Journal of Optimization Theory and Applications*, **18**(2), pp. 235-257.

[27] Maurer H., & Osmolovskii N. P. (2004). Second order sufficient conditions for time-optimal bang–bang control problems. *SIAM Journal on Control and Optimization*, **42**, pp. 2239-2263.

[28] Maurer H., Büskens, C., Kim, J.-H. R., & Kaya, C. Y. (2005). Optimization methods for the verification of second-order sufficient conditions for bang–bang controls, *Optimal Control Applications and Methods*, **26**(3), pp. 129-156.

[29] Miele, A., & Wang, T. (1993). Parallel computation of two-point boundary-value problems via particular solutions, *Journal of Optimization Theory and Applications*, **79**, pp. 5-29.

[30] Noakes, J. L. (1998). A global algorithm for geodesics. *Journal of the Australian Mathematical Society, Series A*, **65**, pp. 37-50.

[31] Oberle, H. J., & Grimm, W. (1989). BNDSCO - A Program for the Numerical Solution of Optimal Control Problems, Report No. 515, Institute for Flight System Dynamics, Oberpfaffenhofen, German Aerospace Research Establishment (DLR).

[32] Osborne, M. R. (1969). On shooting methods for boundary value problems. *Journal of Mathematical Analysis and Applications*. **29**, pp. 417-433.

[33] Pasic, H. (1999). Multi-point boundary-value solution of two-point boundary-value problems. *Journal of Optimization Theory and Applications*, **100**(2), pp. 397-416.

[34] Pontryagin, L. S., Boltyanskii V. G., Gamkrelidze, R. V., & Mishchenko, E. F. (1962). *The Mathematical Theory of Optimal Processes*. Interscience, New York.

[35] Stoer, J., & Bulirsch, R. (1993). *Introduction to Numerical Analysis*, 2nd edition, Springer-Verlag, New York.

[36] Teo, K. L., Goh, C. J., and Wong, K. H. (1991). *A Unified Computational Approach to Optimal Control Problems*, Longman Scientific and Technical, New York.

[37] von Stryk, O., & Bulirsch, R. (1992). Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, **37**, pp. 357-373.

[38] von Stryk, O., & Schlemmer, M. (1994). Optimal control of the industrial robot Manutec r3. In: *Computational Optimal Control, Eds: Bulirsch, R., & Kraft, D., Birkhäuser Verlag, Basel*.

[39] Zuo, Z.-Q. (1991). Two new techniques for optimal control. *IEEE Transactions on Automatic Control*, **36**(11), pp. 1307-1310.