Hampath 2.0: reference manual.

Jean-Baptiste Caillau, Olivier Cots and Joseph Gergaud.

Abstract

The package Hampath [6] is a an open-source software developped to solve optimal control problems via indirect method but also to study Hamiltonian flow. Hampath is developped since 2009 by members of the APO team from Institut de Recherche en Informatique de Toulouse, jointly with colleagues from the Universit de Bourgogne (see contacts tab). Hampath is distributed under the EPL license, and is free for both academic and industrial use.

It first deals with problems where one supposes that the maximization of the Hamiltonian gives the control u = u(x,p) in the state and the costate, and where one also supposes that the problem is **smooth**, *i.e.* that all the functions are smooth and that the relation u(x,p) is also smooth.

Bang-Bang problems where the control is piecewise continuous or **Singular** problems where the control can't be explicitly computed from the maximization of the Hamiltonian can be also treated.

Hampath compiles the Fortran code implementing the *maximized Hamiltonian* and the *Limits Conditions*, into a collection of Matlab, Octave or Fortran functions (depending on the chosen user interface) which allows first of all to solve the shooting equation. However, it is well known that the main difficulty to solve such problems – with indirect methods based on Newton algorithm – is to find a good initial guess. So a differential path following method has been implemented which makes Hampath the natural extension of the package Cotcot [3]. It is also possible to compute the Jacobi fields of the Hamiltonian system to check the order two conditions of optimality and seek conjugate points, as Cotcot does.

We invite the reader to refer to [4], [7], [8] to get more details on how works Hampath.

Index Terms

Geometric optimal control, Second order conditions, Cut and conjugate loci, Simple and Multiple shooting methods, Differential homotopy, Automatic Differentiation, Ordinary Differential Equation.

Contents

I Introduction 2

II Optimal Control Problems solved by Hampath 3

II-A Optimal Control Problem and Pontryagin Maximum Principle 3

II-B Smooth OCP and simple shooting method 4

II-C OCP with Bang/Singular arcs and multiple shooting method 5

II-D OCP depending on parameters and homotopic method 6

II-E User implementation and the Hampath package 7

III Examples 11

III-A Example 1: a simple problem 11

III-B Example 2 : a Bang-Bang problem 12

References 13

J-B. Caillau, IMB (UMR CNRS 5584), 9 Avenue Alain Savary, 21078 Dijon, France, jean-baptiste.caillau@u-bourgogne.fr.
O. Cots, INRIA Sophia Antipolis Méditerranée, 06902 Sophia Antipolis, France, olivier.cots@inria.fr.
J. Gergaud, IRIT (UMR CNRS 5505) and ENSEEIHT, Toulouse, France, joseph.gergaud@enseeiht.fr.

I. Introduction

Consider this simple optimal control problem $P_{\lambda,\mu}$, depending on two parameters: λ and μ.

$$(P_{\lambda,\mu}) \begin{cases} \min \int_0^1 u^2 \mathrm{d}t \\ \dot{x} = v \\ \dot{v} = -\lambda v^2 + u \\ x(0) = -1; x(1) = 0 \\ v(0) = \quad \mu; v(1) = 0, \end{cases} \tag{1}$$

where the final time is fixed: $t_f = 1$, the extremities are fixed:

$$q_0 = (x,v)(0) = (-1,\mu), \quad q_f = (x,v)(t_f) = (0,0)$$

and where u is in $\mathbb{R}$. A standard application of the maximum principle, see [1], [5], tells us that the so-called *normal* and *regular* minimizing curves are the projection of extremals $z = (q,p) = (x,v,p_x,p_v) \in \mathbb{R}^4$ such that

$$\dot{z} = \overrightarrow{H}_r(z) \tag{2}$$

where $H_r(z) = H(z,u(z)) = -u(z)^2 + p_x v + p_v(-\lambda v^2 + u(z))$ is the smooth *regular* Hamiltonian defined on $\Omega = \mathbb{R}^4$ with $u(z) = p_v/2$, and where $\overrightarrow{H}_r = (\partial H_r/\partial p, -\partial H_r/\partial q)$. Since we have boundary conditions, the extremals we are interested in are *BC-extremals*. They are zeros of the *shooting mapping* defined by

$$S_{\lambda,\mu} : p_0 \mapsto \Pi(\exp_{t_f}(q_0,p_0)) - q_f \tag{3}$$

with $\exp_t(z_0) = z(t,z_0)$ the solution of (2) with the initial condition z_0 and $\Pi : (q,p) \mapsto q$ the canonical projection. Note that the solution of the shooting function depends on the parameters λ and μ. We call the mapping in (3) an homotopic function as soon as we consider the parameters as independent variables. Moreover, the (local) optimality of such extremals is checked by a rank test on the subspaces spanned by the *Jacobi fields* along the trajectory. These fields are solutions of the variationnal equation

$$\partial\dot{z} = \mathrm{d}\overrightarrow{H}_r(z(t)) \cdot \partial z \tag{4}$$

with suitable initial conditions.

The aim of the `Hampath` code which extends the possibility of the `Cotcot` software, is to provide numerical tools

- to integrate Hamiltonian systems such as (2)
- to solve the associated shooting equation defined by (3)
- to get the zeros path of an homotopic function provided by a shooting mapping depending on parameters
- to compute the corresponding Jacobi fields (4) along the extremal
- to evaluate the resulting conjugate points, if any.

Remark 1. The `Hampath` package extends the software `Cotcot`. Moreover it permits to solve efficiently homotopic function defined from an optimal control problem (OCP) depending on parameters, and finally to solve OCP where the control solution is non-smooth. The structure of the control may be Bang-Bang or Bang-Singular.

We first define in §II all the kind of optimal control problems that can be solved by `Hampath`. After that, we present in §III two examples provided in the package.

Remark 2. Since a lot of time and effort has gone into `Hampath`'s development, please cite [6] or [7] if you are using `Hampath` for your own research.

II. Optimal Control Problems solved by Hampath

A. *Optimal Control Problem and Pontryagin Maximum Principle*

The following part is inspired from the course notes [14] of E. Trélat. In the next part, we present an overview of the Pontryagin maximum principle. See [1], [5] to learn more details on optimal control theory. We consider the following control system in $\mathbb{R}^n$

$$\dot{x}(t) = f(t, x(t), u(t)) \tag{5}$$

where $f : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}^n$ is C^1 and where the control u is an essentially bounded and measurable function on an interval $[0, t_e(u))$ of $\mathbb{R}^+$ which takes its values in $\Omega \subset \mathbb{R}^m$. Let M_0 and M_1 be two subsets of $\mathbb{R}^n$. We denote by $\mathcal{U}$ the set of admissible controls u which permit to steer the system from an initial point of M_0 to a final point of M_1 in a time $t(u) < t_e(u)$.

Moreover we define the cost of a control u on $[0, t]$

$$C(t, u) = g(t, x(t)) + \int_0^t f^0(s, x(s), u(s))ds,$$

where $f^0 : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \longrightarrow \mathbb{R}^n$ and $g : \mathbb{R} \times \mathbb{R}^n \longrightarrow \mathbb{R}$ are C^1, and $x(\cdot)$ is the trajectory solution of (5) associated to the control u.

We consider the following $\underline{\text{optimal control problem}}$: determine a trajectory from M_0 to M_1 which minimize the cost. The final time can be fixed or not.

If the control $u \in \mathcal{U}$ associated to the trajectory $x(\cdot)$ is optimal on $[0, T]$, then there exists according to the Pontryagin maximum principle, an absolutely continuous mapping $p(\cdot) : [0, T] \longrightarrow \mathbb{R}^n$ called *adjoint vector*, and a real number $p^0 \leqslant 0$, with $(p(\cdot), p^0) \neq (0, 0)$, such that for almost every $t \in [0, T]$,

$$\dot{x}(t) = \frac{\partial H}{\partial p}(t, x(t), p(t), p^0, u(t)), \quad \dot{p}(t) = -\frac{\partial H}{\partial x}(t, x(t), p(t), p^0, u(t)), \tag{6}$$

where $H(t, x(t), p(t), p^0, u(t)) = \langle p, f(t, x, u)\rangle + p^0 f^0(t, x, u)$ is the *Hamiltonian* function, and the maximization condition

$$H(t, x(t), p(t), p^0, u(t)) = \max_{v \in \Omega} H(t, x(t), p(t), p^0, v) \tag{7}$$

holds almost everywhere on $[0, T]$.

If the final time to reach the target M_1 is free, we have another condition at the final time T

$$\max_{v \in \Omega} H(T, x(T), p(T), p^0, v) = -p^0 \frac{\partial g}{\partial t}(T, x(T)). \tag{8}$$

Moreover if M_0 or M_1 (or both) is a manifold of $\mathbb{R}^n$ having tangent spaces in $x(0) \in M_0$ or $x(T) \in M_1$ (or both), then the adjoint vector must verify respectively the first or the second condition (or both) below

$$p(0) \perp T_{x(0)} M_0 \tag{9}$$

$$p(T) - p^0 \frac{\partial g}{\partial x}(T, x(T)) \perp T_{x(T)} M_1. \tag{10}$$

Remark 3. If the control u is continuous at time T, the condition (8) can be written

$$H(T, x(T), p(T), p^0, u(T)) = -p^0 \frac{\partial g}{\partial t}(T, x(T)). \tag{11}$$

Remark 4. We have for almost every $t \in [0, T]$

$$\frac{\mathrm{d}}{\mathrm{d}t} H(t, x(t), p(t), p^0, u(t)) = \frac{\partial H}{\partial t}(t, x(t), p(t), p^0, u(t)). \tag{12}$$

In particular if the augmented system is autonomous, *i.e.* if f and f^0 do not depend on t, then H does not depend on t and we have

$$\forall t \in [0, T] \ \max_{v \in \Omega} H(x(t), p(t), p^0, v) = C.$$

Remark 5. The convention $p^0 \leqslant 0$ leads to the maximum principle. The convention $p^0 \geqslant 0$ would imply that the condition (7) would be a minimum condition.

Definition 1. *An extremal of the optimal control problem is a quadruple $(x(\cdot), p(\cdot), p^0, u(\cdot))$ solution of the equations (6) and (7). If $p^0 = 0$, the extremal is said to be abnormal and if $p^0 \neq 0$ it is said to be normal.*

Definition 2. *The conditions (9) and (10) are called transversality conditions on the adjoint vector. The condition (8) is called condition of transversality on the Hamiltonian.*

Definition 3. *We call true hamiltonian the function*

$$H_r(t, x(t), p(t)) = H(t, x(t), p(t), p^0, u(t, x(t), p(t))), \tag{13}$$

when the maximum condition gives the control u in terms of (t, x, p) and with p^0 replaced by its value.

To solve this optimal control problem, `Hampath` use indirect method as simple or multiple shooting based on the famous Newton algorithm. In the next subsection we present how classic problems are solved by simple shooting. We define the two points boundary value problem (TPBVP) associated to the optimal control problem and the shooting equation. After that, we present how to deal with non-smooth control solution. Indeed in the case of Bang-Bang control or Bang-Singular control we need to use multiple shooting. We define then the multiple boundary value problem (MBVP) and the shooting equation in this case. In the final subsection of this part we explain what the user has to implement and what is automatically done. But before we introduce the core of the method *hampath* which consist to replace shooting by the mere integration of an ODE. Indeed, for optimal control problems depending on parameters, differential pathfollowing replaces Newton solver (except for the computation of the starting point where shooting is still needed).

B. Smooth OCP and simple shooting method

We consider the following optimal control problem

$$(P) \begin{cases} \min g(t_f, x(t_f)) + \int_0^{t_f} f^0(t, x(t), u(t)) \mathrm{d}t, \quad x(t) \in \mathbb{R}^n \\ \\ \dot{x}(t) = f(t, x(t), u(t)), \quad u(t) \in \mathcal{U} \subset \mathbb{R}^m \\ \\ b_0(x(0)) = 0 \in \mathbb{R}^{n_0}, \quad n_0 \leqslant n \\ b_f(t_f, x(t_f)) = 0 \in \mathbb{R}^{n_1}, \quad n_1 \leqslant n. \end{cases} \tag{14}$$

Note that there is no condition on the final time t_f, it can be fixed or free and the system may be time dependent. The Pontryagin maximum principle gives necessary conditions on the solution and asserts that any optimal trajectory is the projection of an extremal $(x(\cdot), p(\cdot), p^0, u(\cdot))$. If the maximum condition (7) gives a **smooth** control in terms of $(t, x(t), p(t))$ (*i.e.* $u(t, x(t), p(t)) = \arg\max_{v \in \Omega} H(t, x(t), p(t), p^0, v)$, p^0 fixed, then the extremal system (6) is a differential system of the form $\dot{z}(t) = \overrightarrow{H}_r(t, z(t))$, where $z(t) = (x(t), p(t))$, H_r is the true hamiltonian (Def. 3) and where p^0 is omitted because it has a known value. The PMP gives also some transversality equations (8), (9) and (10) which can be written in the form $E(z(0), z(t_f)) = 0 \in \mathbb{R}^{2n}$ if t_f is fixed and $E(z(0), t_f, z(t_f)) = 0 \in \mathbb{R}^{2n+1}$ if t_f is free.

Finally, in the case of fixed t_f, we obtain the two points boundary value problem (TPBVP)

$$(TPBVP) \begin{cases} \dot{z} = \overrightarrow{H}_r(t, z) \\ E(z(0), z(t_f)) = 0 \end{cases} \tag{15}$$

Let $z(t, z_0)$ be the solution of the Cauchy problem

$$\dot{z}(t) = \overrightarrow{H}_r(t, z(t)), \quad z(0) = z_0 \tag{16}$$

and let define the *simple shooting function* S by

$$\begin{array}{rcl} S : \mathbb{R}^{2n} & \longrightarrow & \mathbb{R}^{2n} \\ z_0 & \longmapsto & E(z_0, z(t_f, z_0)) \end{array}$$

The problem (15) is then equivalent to

$$S(z_0) = 0.$$

Therefore the *simple shooting method* consists in finding a zero of the function S. We use the fortran Newton method `Hybrj` (from the `Minpack` library) to solve this non linear equation. A simple example of smooth OCP is detailed in §III-A.

Remark 6. If the final time is free, then t_f is one more variable of the shooting function and we add the transversality condition (8) on the Hamiltonian.

C. OCP with Bang/Singular arcs and multiple shooting method

In comparison to simple shooting, multiple shooting splits the interval $[0, t_f]$ in N intervals $[t_i, t_{i+1}]$ and has the values $z(t_i)$ at the beginning of each sub-interval as unknowns. Then one has to take into account some matching conditions at each instant t_i (continuity condition). The goal is to improve the stability of the method.

In the case where there are some *switchings arcs* and where the smoothness of the control u is lost, the multiple shooting is used and the instants t_i become the switching times. The Pontryagin maximum principle gives conditions on the limits on the state and the adjoint vector and on the Hamiltonian when t_f is free. By limits, we mean at the initial time, the final time and the switching times.

Therefore we write in the case of fixed t_f this multiple boundary value problem (MBVP)

$$(MBVP) \begin{cases} \dot{z}(t) = \overrightarrow{H}_i(t, z) & \text{for} \quad i = 0, \ldots, N-1 \quad \text{and} \quad t \in [t_i, t_{i+1}] \\ \\ z(t_i) = z_i & \text{for} \quad i = 0, \ldots, N-1 \\ \\ m_i(z(t_{i+1}, t_i, z_i), z_{i+1}) = 0 & \text{for} \quad i = 0, \ldots, N-2 \\ \\ b_0(z_0) = 0 & \in \mathbb{R}^n \\ \psi_i(t_{i_k}, z_{i_k}) = 0 & \text{for} \quad i = 1, \ldots, N-1 \quad \text{and} \quad i_k \in [\![0, N]\!] \\ b_f(t_f, z(t_f, t_{N-1}, z_{N-1})) = 0 & \in \mathbb{R}^n. \end{cases} \tag{17}$$

with $\overrightarrow{H}_i(t, z)$ the true Hamiltonian vector field on the sub-interval $[t_i, t_{i+1}]$, and z_i the unknowns of the problem, for $i = 0, \ldots, N-1$. The third equation is the matching conditions where $z(t_{i+1}, t_i, z_i)$ is the solution at t_{i+1} of the Cauchy problem

$$\dot{z}(t) = \overrightarrow{H}_i(t, z(t)), \quad z(t_i) = z_i. \tag{18}$$

In general, these matching equations become

$$m_i(z(t_{i+1},t_i,z_i),z_{i+1}) = z(t_{i+1},t_i,z_i) - z_{i+1} = 0 \tag{19}$$

and are just continuity conditions. b_0 and b_f are the boundary equations. Note that the switching conditions are written $\psi_i(t_{i_k},z_{i_k}) = 0$, $i_k \in [\![0,N]\!]$. It is because you can have more than one condition at a time t_{i_k}. However, for a Bang-Bang problem, these equations become $\psi_i(t_i,z_i) = H_{i+1}(t_i,z_i) - H_i(t_i,z_i) = 0$, $i = 1,\dots,N-1$.

Let now define

- $Ti = \{t_1,\dots,t_{N-1}\}$
- $Zi = \{z_0,\dots,z_{N-1}\}$
- $expZi = \{z(t_1,t_0,z_0),\dots,z(t_f,t_{N-1},z_{N-1})\}$

Then all the conditions m_i, b_0, b_f and ψ_i can be put together in an equation function of the form $E(Zi,Ti,expZi) = 0 \in \mathbb{R}^{2nN+N-1}$ if t_f is fixed. In this case, the *multiple shooting function* S is

$$\begin{array}{rcl} S : \mathbb{R}^{2nN+N-1} & \longrightarrow & \mathbb{R}^{2nN+N-1} \\ (Ti,Zi) & \longmapsto & E(Zi,Ti,expZi) \end{array}$$

The problem (17) is then equivalent to

$$S(Ti,Zi) = 0.$$

Therefore the *multiple shooting method* consists in finding a zero of the function S. We use again `Hybrj` to solve this non linear equation. An example with Bang-Bang structure is detailed in §III-B.

Remark 7. If the final time is free, $Ti = \{t_1,\dots,t_{N-1},t_f\}$ and we add the transversality condition on the Hamiltonian (8).

D. *OCP depending on parameters and homotopic method*

The optimal control problem may have some parameters either mathematical or physical. Indeed the difficulty of shooting method is to have a good initial point because of the low convergence radius of the Newton algorithm. Continuation method may be used to transform the initial problem, which may be difficult to solve, into a family of easier problems such that the solutions of these problems converge to the initial one. A problem of contrast in imaging by NMR is presented in [4]. The problem is non-smooth and so penalized to get more regularity. The continuation gives an approximation of the solution of the original problem, and is used to catch the Bang-Singular structure. In [7] is detailed two examples. The first one presents an orbital transfer where an homotopy from the minimal time problem to the minimum fuel consumption (Bang-Bang) problem is performed. The second example studies the energy minimization problem for two-level dissipative quantum systems and uses continuation on physical parameters which characterize the kind of particles involved.

The parameters can be found in any equation or function of the optimal control problem. Finally, the optimal control problems which can be solved by `Hampath` are of the form

$$(P_\lambda)\left\{\begin{array}{l} \min g(t_f,x(t_f),\lambda) + \int_0^{t_f} f^0(t,x(t),u(t),\lambda)\mathrm{d}t \\[2ex] \dot{x}(t) = f(t,x(t),u(t),\lambda), \quad u(t) \in U \subset \mathbb{R}^m, \quad \lambda \in \mathbb{R} \\[2ex] b_0(x(0),\lambda) = 0 \in \mathbb{R}^{n_0}, \quad n_0 \leqslant n \\ b_f(t_f,x(t_f),\lambda) = 0 \in \mathbb{R}^{n_1}, \quad n_1 \leqslant n. \end{array}\right. \tag{20}$$

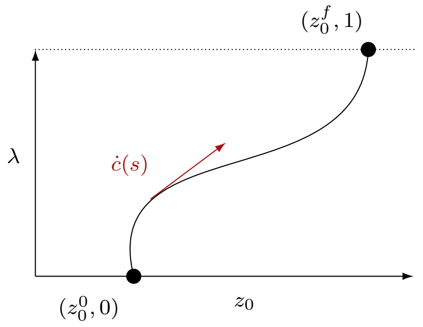
As a consequence, the shooting function either on the smooth or non-smooth case, depends on the parameter λ. Let consider the smooth case for a sake of convienence. For now, the goal is not to solve one optimal control problem but to compute the zeros path of the homotopic function

$$\begin{array}{rcl} S : \mathbb{R}^{2n} \times [0,1] & \longrightarrow & \mathbb{R}^{2n} \\ (z_0, \lambda) & \longmapsto & S(z_0, \lambda) \end{array}$$

where S is nonlinear and smooth (even with non-smooth control). Assuming that 0 is a regular value for S, the solution set of $S(c) = 0$, $c = (z_0^\lambda, \lambda)$ is made of smooth curves. Thanks to arc length parametrization and some regularity conditions on the jacobian of S, the tangent vector $\dot{c}(s)$ of the path can be computed and it is unique modulo the sign. Finally, `Hampath` uses `Dopri5` from E. Hairer and G. Wanner [10] [11], for the numerical integration of the following (IVP):

$$(IVP) \left\{ \begin{array}{l} \dot{c}(s) = T(c(s)) \\ c(0) = (z_0^0, 0) \end{array} \right.$$

until s_f such as $\lambda(s_f) = 1$ (the dense output of the integrator is used). The time s is the arc length. To compute the tangent vector, the jacobian of the homotopic function S is needed. There is still one call to the shooting method for the intial point in $\lambda = 0$. Then there is no correction like in the well known Prediction-Correction method (Allgower and Georg. [2]). The following diagram represents the zeros path of the homotopic function.



Remark 8. The OCP may depend on a vector parameter $\Lambda \in \mathbb{R}^k$, $k \geqslant 1$. As a consequence, we replace the notation P_λ by P_Λ. An homotopy on several parameters (from Λ_0 to Λ_f for instance) can be computed using a scalar reparametrization. This can be done by adding a scalar parameter λ such as $\Lambda = \Lambda_0$ when $\lambda = 0$ and $\Lambda = \Lambda_f$ when $\lambda = 1$. The affine homotopic function $\Lambda(\lambda) = (1-\lambda)\Lambda_0 + \lambda\Lambda_f$ is a simple example of such a parametrization. You can find more details on homotopy on several parameters at the end of §II-E.

Remark 9. The mathematical framework of homotopic method in `Hampath` is discussed in [7].

E. User implementation and the `Hampath` package

The user only needs to code in `Fortran` the true Hamiltonian H_r of the system (2) and the function E defined in §II-B and §II-C. Before that, he must have applied the Pontryagin maximum principle to get the control $u(x,p)$ and the transversality conditions if any. The file `hfun.f90` must contain the true Hamiltonian and `efun.f90` the function E. Then during hampath command call (try `hampath -help` in a terminal after installation, for documentation) `Hampath` generates some functions which code, among others, the shooting function and its jacobian (by automatic differenciation and using variationnal equations) and the shooting and homotopic method.

The code of the fortran subroutine `hfun` (in `hfun.f90`) is :

```
Subroutine hfun(t,n,z,nbarc,iarc,lpar,par,H)
    use utils
    implicit none
    integer,                           intent(in)   :: n,lpar,iarc,nbarc
    real(kind=8), intent(in)                        :: t
    real(kind=8), dimension(2*n),      intent(in)   :: z
    real(kind=8), dimension(lpar),     intent(in)   :: par
    real(kind=8),                      intent(out)  :: H
    IF (nbarc.NE.0) THEN
        call printandstop('Error: wrong number of arcs.')
    END IF
    IF (n.NE.0) THEN
        call printandstop('Error: wrong state dimension.')
    END IF
    IF (lpar.NE.0) THEN
        call printandstop('Error: wrong par dimension.')
    END IF
    H = ...
end subroutine hfun
```

`t`	is the current time.
`n`	is the dimension of the state x.
`z`	is a vector containing the state and the adjoint vector: $z = (x, p)$.
`nbarc`	is the number or different arcs. In the smooth case $nbarc = 1$.
`iarc`	is the index of the current arc: $\dot{z}(t) = \overrightarrow{H}_{iarc}(t, z)$.
`lpar`	is the number of additionnal parameters.
`par`	is a vector containing the $lpar$ parameters. Under the notation $\Lambda \in \mathbb{R}^k$, $par \equiv \Lambda$ and $lpar \equiv k$.
`H`	is the only output. It returns the value of the true Hamiltonian $H_{\text{iarc}}(t, z)$.

Obviously, the Hamiltonian may be time dependent. Moreover, additional parameters may be used (see remark 15). Note that, for the sake of robustness, dimensions are checked (`printandstop` calls).

The code of the fortran subroutine `efun` (in `efun.f90`) is :

```
Subroutine efun(nbarc,n,Ti,Zi,expZi,lpar,par,ne,E)
    use utils
    implicit none
    integer,                                    intent(in)  :: nbarc,n,lpar,ne
    real(kind=8),  dimension(nbarc+1),          intent(in)  :: Ti
    real(kind=8),  dimension(2*n,nbarc),        intent(in)  :: Zi
    real(kind=8),  dimension(2*n,nbarc),        intent(in)  :: expZi
    real(kind=8),  dimension(lpar),             intent(in)  :: par
    real(kind=8),  dimension(ne),               intent(out) :: E
    IF (ne.NE.0) THEN
        call printandstop('Error: wrong equations dimension.')
    END IF
    IF (nbarc.NE.0) THEN
        call printandstop('Error: wrong number of arcs.')
    END IF
    IF (n.NE.0) THEN
        call printandstop('Error: wrong state dimension.')
    END IF
    IF (lpar.NE.0) THEN
        call printandstop('Error: wrong par dimension.')
    END IF
    E = ...
end subroutine efun
```

`nbarc`	is the number or different arcs. In the smooth case $nbarc = 1$.
`n`	is the dimension of the state x.
`Ti`	is a vector containing the different instants: $Ti = (t_0, t_1, \dots, t_{\text{nbarc}-1}, t_f)$. Note that t_0 is fixed, t_f may be fixed or free and all others instants are free.
`Zi`	is a matrix containing all the states and the adjoint vectors at each instant except t_f: $Zi = (z_0, \dots, z_{\text{nbarc}-1})$.
`expZi`	is a matrix of the form: $expZi = (z(t_1, t_0, z_0), \dots, z(t_f, t_{\text{nbarc}-1}, z_{\text{nbarc}-1}))$.
`lpar`	is the number of additionnal parameters.
`par`	is a vector containing the $lpar$ parameters. Under the notation $\Lambda \in \mathbb{R}^k$, $par \equiv \Lambda$ and $lpar \equiv k$.
`ne`	is the dimension of the matrix E, *i.e.* the number of equations.
`E`	is the only output. It represents all the conditions, *i.e.* the initial, the final and the matching and the transversality if any.

Remark 10. Try `hampath -help` in a terminal after installation, for documentation on all functions provided by `Hampath` package.

Remark 11. Only the `hfun` subroutine is required. In the case where the `efun.f90` file is omitted, then `Hampath` does not produce all the functions. Only methods to study Hamiltonian flows are available.

Remark 12. Note that you can perform homotopies on several parameters using a scalar reparametrization of the problem (P_Λ). In this case, it is required to have a function $\Lambda(\lambda)$, $\lambda \in \mathbb{R}$ such that $\Lambda(0) = \Lambda_0$ and $\Lambda(1) = \Lambda_f$. The function $\Lambda(\lambda)$ can be explicitly written in `hfun` and/or `efun` but this implies to add a parameter to the variable `par`. To avoid the issue of the additionnal parameter λ, `Hampath` allows the user to implement the function `parfun` (coding $\Lambda(\cdot)$) separately from `hfun` and `efun`. The subroutine `parfun` should be written in a file `parfun.f90`. However, a default subroutine `parfun` coding $\Lambda(\lambda) = (1-\lambda)\Lambda_0 + \lambda\Lambda_f$ is integrated in `Hampath`. As a consequence, the file `parfun.f90` is not needed if the user wants to do an affine homotopy on the parameters. The code (where $\Lambda \equiv par$) in the default `parfun.f90` file is :

```
Subroutine parfun(lambda,lpar,par0,parf,par)
    implicit none
    integer                             :: lpar
    real(kind=8)                        :: lambda   !current scalar homotopic parameter
    real(kind=8), dimension(lpar) :: par0     !initial parameters
    real(kind=8), dimension(lpar) :: parf     !final parameters
    real(kind=8), dimension(lpar) :: par
    par = (1d0-lambda)*par0 + lambda*parf
end subroutine parfun
```

Remark 13. Homotopy on the final time t_f is possible. The time variable t has to be changed in s such as $t = t_0 + (t_f - t_0)s = t_f s$ and t_f becomes a new parameter of the vector `par`. The state x and the control u have a new parametrization in s and so $x(t) \equiv \tilde{x}(s)$ and $u(t) \equiv \tilde{u}(s)$ and the problem (20) becomes :

$$(P_\lambda) \begin{cases} \min g(t_f, \tilde{x}(1), \lambda) + \int_0^1 t_f \cdot f^0(s, \tilde{x}(s), \tilde{u}(s), \lambda) \mathrm{d}s \\ \\ \dot{\tilde{x}}(s) = t_f \cdot f(s, \tilde{x}(s), \tilde{u}(s), \lambda), \quad \tilde{u}(s) \in U \subset \mathbb{R}^m, \quad \lambda \in \mathbb{R} \\ \\ b_0(\tilde{x}(0), \lambda) = 0 \in \mathbb{R}^{n_0}, \quad n_0 \leqslant n \\ b_f(t_f, \tilde{x}(1), \lambda) = 0 \in \mathbb{R}^{n_1}, \quad n_1 \leqslant n \end{cases} \tag{21}$$

The following diagram, Fig. 1, resumes the main functionnalities of the `Hampath` package.

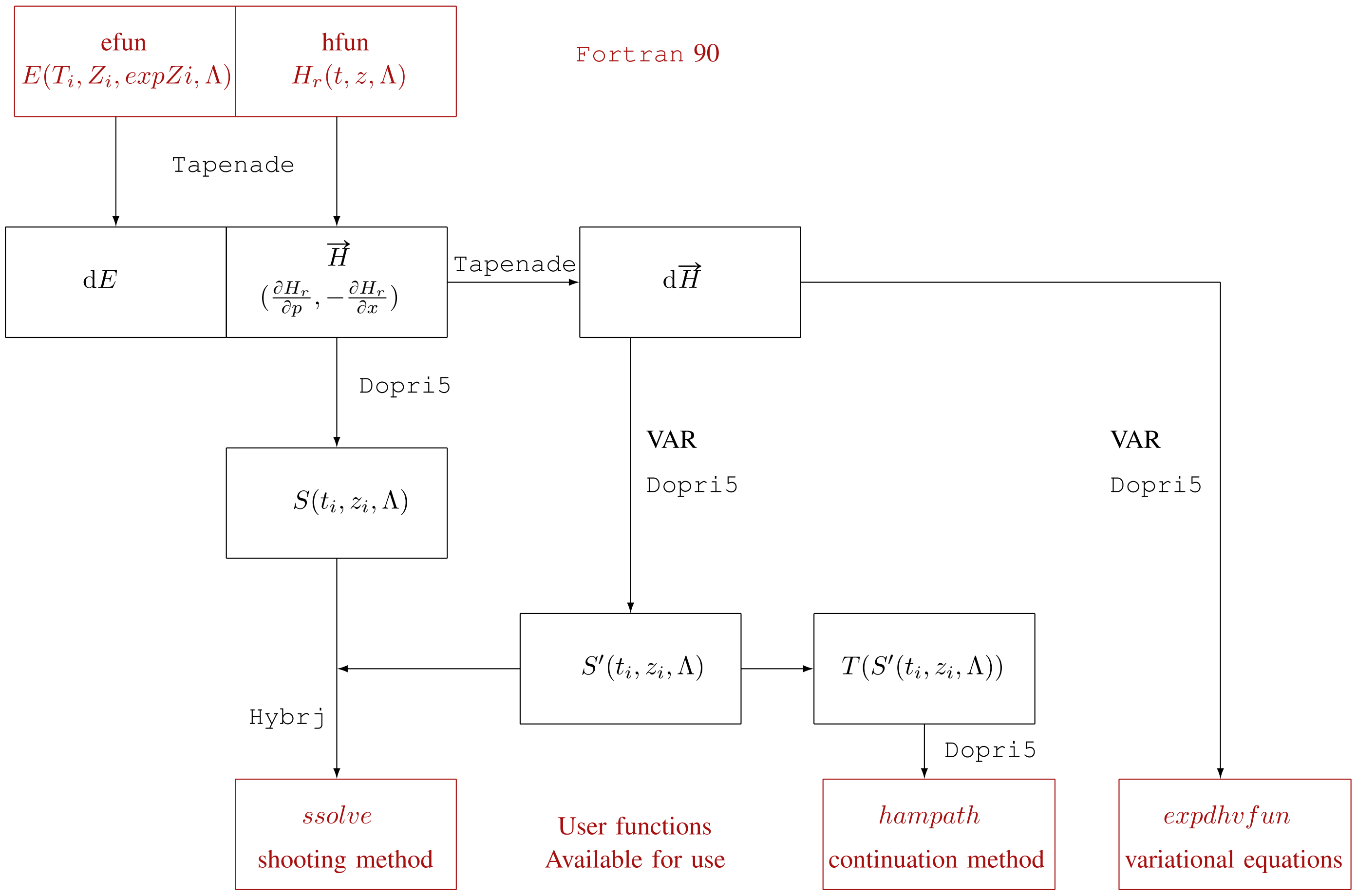


Fig. 1. Global diagram

Finally, from the true Hamiltonian and all the conditions, `Hampath`:

- produces automatically the state-costate equations (thanks `Tapenade`)
- computes the shooting function by numerical integration (thanks `Dopri5`)
- provides the VARiationnal equations (in [12] are compared variationnal equations with classical finite differences) used in the jacobian of the shooting function (thanks `Tapenade`)
- integrates the variationnal equations so that this diagram commutes

$$\begin{tikzcd}
(IVP) \arrow[r, "\text{Numerical integration}"] \arrow[d, "\text{Derivative}"'] & S(z_0, \Lambda) \arrow[d, "\text{Derivative}"] \\
(VAR) \arrow[r, "\text{Numerical integration}"] & \frac{\partial S}{\partial z_0}(z_0, \Lambda)
\end{tikzcd}$$

III. Examples

A. Example 1: a simple problem

We define the first example which has a smooth optimal control law:

$$(P_\lambda) \begin{cases} \min \int_0^1 u^2 \mathrm{d}t \\ \dot{x} = v \\ \dot{v} = -\lambda^2 v + u \\ x(0) = x_0, \; x(1) = x_f \\ v(0) = v_0, \; v(1) = v_f. \end{cases} \tag{22}$$

Remark 14. This example is provided in the `Hampath` package. Try `hampath -newSimpleShooting` to copy the files defining this problem (`hfun.f90`, `efun.f90` and a main script) and follow instructions to compile and solve the problem.

We first define the true Hamiltonian and the conditions according to the Pontryagin maximum principle, which are written in `hfun.f90` and `efun.f90`.

Define the true Hamiltonian

The user needs to provide first the `Fortran` subroutine `hfun` coding the true Hamiltonian which is in this case

$$H_r(z) = H(z, u(z)) = -u(z)^2 + p_x v + p_v(-\lambda v^2 + u(z)),$$

with $u(z) = p_v/2$ and $z = (q, p) = (x, v, p_x, p_v) \in \mathbb{R}^4$. See `hfun.f90`.

Define the function E

The second `Fortran` subroutine required from the user is `efun` which codes all the limits conditions. In this case, there is no additional transversality conditions because the state is completely fixed and the final time is not free. The function $E(z(0), z(t_f))$ is

$$E(z(0), z(t_f)) = (x(0) - x_0, v(0) - v_0, x(t_f) - x_f, v(t_f) - v_f) \in \mathbb{R}^4,$$

with $t_f = 1$. See `efun.f90`.

Remark 15. Almost all the functions provided in the package (`sfun`, `ssolve`, `exphvfun ...`) have the variable `par` which permits to pass the additional parameters to the `Fortran` subroutines `hfun` and `efun`. They can be used as homotopic parameters or constant values. The vector par is $par = (x_0, v_0, x_f, v_f, \lambda)$.

Main script

In the main script, the problem is first solved for $\lambda = 0$ and $(x_0, v_0, x_f, v_f) = (-1, -1, 0, 0)$ using the shooting method and then is computed the zeros path solution of (P_λ) for $\lambda \in [0, 1]$.

1) Matlab/Octave interface: If you have installed the `Matlab` / `Octave` version, then in the main script `main.m`, are plotted the zeros path and the extremal solution of (P_1) with the associated control. Finally we check the order two conditions of optimality by checking a rank condition on a particular matrix.

Remark 16. In `hfun.f90`, is written an auxiliary subroutine which computes the control. This subroutine has the same input variables than `hfun` subroutine and one output variable, thus `Hampath` will produce a `Matlab` / `Octave` function which interface the `control` subroutine and which can be called in the main script with the same syntax as `Matlab` / `Octave` `hfun`.

Remark 17. All auxiliary subroutines with the same input variables than `hfun` and one output variable will have an interface which can be called with the same syntax as `Matlab` / `Octave` `hfun`.

Remark 18. All auxiliary subroutines called in `hfun` must be written in `hfun.f90` and all auxiliary subroutines called in `efun` must be written in `efun.f90`.

2) Fortran stand alone version: If you have installed the `Fortran` stand alone version, then you can install `GnuPlot` to plot results. The main script is `main.f90`. There is nothing equivalent to remark 17, in the fortran stand alone version.

B. Example 2 : a Bang-Bang problem

We define the second example which has a bang-bang optimal control law:

$$(P_\lambda) \begin{cases} \min t_f \\ \dot{x} = v \\ \dot{v} = -\lambda^2 v + u \\ |u| \leqslant 1 \\ x(0) = -1, \ x(1) = 0 \\ v(0) = -1, \ v(1) = 0. \end{cases} \tag{23}$$

Remark 19. This example is provided in the `Hampath` package and has a free final time. Try `hampath -newMultipleShooting` to copy the files defining this problem (`hfun.f90`, `efun.f90` and a main script) and follow instructions to compile and solve the problem.

We first define the true Hamiltonian and the conditions according to the Pontryagin maximum principle, which are written in `hfun.f90` and `efun.f90`.

Define the true Hamiltonian

We know that the optimal structure is bang-bang with two arcs. A first arc positive, *i.e.* $u = 1$, and a second arc negative, *i.e.* $u = -1$. The user needs to provide first the `Fortran` subroutine `hfun` coding the true Hamiltonian which is in this case

$$H_r(z) = H(z, u(z)) = -1 + p_x v + p_v(-\lambda v^2 + u(z)),$$

with $z = (q, p) = (x, v, p_x, p_v) \in \mathbb{R}^4$ and $u(z) = +1$ for the first arc, and $u(z) = -1$ for the second arc. The value of the control depends on the sign of p_v which is positive and then negative. It becomes 0 at the switching time t_1 between the two bang arcs. See the role of the `iarc` and `nbarc` variables in `hfun.f90`.

Define the function E

The second `Fortran` subroutine required from the user is `efun` which codes all the limits conditions. In this case, there is no additional transversality conditions on the adjoint vector because the state is completely fixed. The function $E(z_0, z_1, t_1, t_f, z(t_1, t_0 = 0, z_0), z(t_f, t_1, z_1))$ is

$$E = (x(0) + 1, v(0) + 1, x(t_f), v(t_f), z_1 - z(t_1, t_0 = 0, z_0), p_{v,1}, H_r(z(t_f, t_1, z_1))) \in \mathbb{R}^{10},$$

where $z_1 - z(t_1, t_0 = 0, z_0) = 0$ are the matching conditions at the switching time t_1, where we note $z_1 = (x_1, v_1, p_{x,1}, p_{v,1})$, so $p_{v,1} = 0$ is the condition which makes that the control u changes its sign at the switching time, and where $H_r(z(t_f, t_1, z_1)) = 0$ is the condition of transversality on the Hamiltonian, since the final time is free, see eq. (8). See `efun.f90` file for the code of this subroutine.

Remark 20. Almost all the functions provided in the package (`sfun`, `ssolve`, `exphvfun` ...) have the variable `par` which permits to pass the additional parameters to the `Fortran` subroutines `hfun` and `efun`. They can be used as homotopic parameters or constant values. The vector par is here scalar since it is $par = \lambda$.

Main script

In the main script, the problem is first solved for $\lambda = 0$ using the shooting method and then is computed the zeros path solution of (P_λ) for $\lambda \in [0, 0.5]$.

1) Matlab/Octave interface: If you have installed the `Matlab` / `Octave` version, then in the main script `main.m`, are plotted the zeros path and the extremal solution of $(P_{0.5})$ with the associated control.

Remark 21. Same as remarks 16, 17 and 18.

2) Fortran stand alone version: If you have installed the `Fortran` stand alone version, then you can install `GnuPlot` to plot results. The main script is `main.f90`

REFERENCES

[1] A. A. Agrachev & Y. L. Sachkov, *Control theory from the geometric viewpoint*, vol **87** of *Encyclopaedia of Mathematical Sciences*, Springer-Verlag, Berlin (2004), xiv+412.

[2] E. Allgower & K. Georg, *Introduction to numerical continuation methods*, vol **45** of *Classics in Applied Mathematics*, Soc. for Industrial and Applied Math., Philadelphia, PA, USA, (2003), xxvi+388.

[3] B. Bonnard, J.-B. Caillau & E. Trélat, *Cotcot: short-reference manual. apo.enseeiht.fr/cotcot*, Rapport de recherche RT/APO/05/1, Institut National Polytechnique de Toulouse, Toulouse, France (2005).

[4] B. Bonnard & O. Cots, *Geometric numerical methods and results in the control imaging problem in nuclear magnetic resonance*, Math. Models Methods Appl. Sci., to appear, (2012).

[5] B. Bonnard & D. Sugny, *Optimal Control with Applications in Space and Quantum Dynamics*, vol 5 of *Applied Mathematics*, AIMS (2012).

[6] J.-B. Caillau, O. Cots & J. Gergaud, HAMPATH*: on solving optimal control problems by indirect and path following methods, http://cots.perso.math.cnrs.fr/hampath* (2010).

[7] J.-B. Caillau, O. Cots & J. Gergaud, *Differential continuation for regular optimal control problems*, Optimization Methods and Software, **27** (2011), no 2, 177–196.

[8] O. Cots, *Contrôle optimal géométrique : méthodes homotopiques et applications.* Phd thesis, Institut Mathématiques de Bourgogne, Dijon, France, 2012.

[9] L. Hascoët & V. Pascual, *The Tapenade Automatic Differentiation tool: principles, model, and specification*, Rapport de recherche RR-7957, INRIA (2012).

[10] E. Hairer, S. P. Nørsett & G. Wanner, *Solving Ordinary Differential Equations I, Nonstiff Problems*, vol 8 of *Springer Serie in Computational Mathematics*, Springer-Verlag, second ed. (1993).

[11] E. Hairer & G. Wanner, DOPRI5. *http://www.unige.ch/~hairer/prog/nonstiff/dopri5.f* (1996).

[12] P. Martinon & J. Gergaud, *Using switching detection and variational equations for the shooting method*, Optimal Control Appl. Methods, **28** (2007), no 2, 95–116.

[13] H. Maurer, *Numerical solution of singular control problems using multiple shooting techniques*, J. Optim. Theory Appl., (1976).

[14] E. Trélat, *Contrôle optimal*, Mathe matiques Concrètes, 2005.