Optimal trajectory approximation by cubic splines on fed-batch control problems

A. ISMAEL F. VAZ
Production and Systems Department
Minho University, Engineering School
aivaz@dps.uminho.pt
EUGÉNIO C. FERREIRA
Centro de Engenharia Biológica
Universidade do Minho
ecferreira@deb.uminho.pt
ALZIRA M.T. MOTA
Mathematics Department
Porto Polytechnic Institute, Porto Engineering Institute
atm@isep.ipp.pt

Abstract: Optimal control problems appear in several engineering fields and in particular on the control of fed-batch fermentation processes. These problems are often described by sets of nonlinear differential and algebraic equations, usually subject to constraints in the state and control variables.
Tradicional approaches to the optimal feed trajectory computation consists in getting a linear spline that approximates the trajectory, which optimizes a given performance of the fed-batch fermentation process. This approach leads to non-differentiable trajectories that can pose some problems to implement in practice, resulting in a possible discrepancy of the simulated and real performances.
In this paper we develop a technique to obtain a cubic spline for the approximate trajectory, leading to a smooth approximation function. We provide numerical results for a set of case studies where the AMPL modeling language, CVODE ordinary differential equations solver and a particle swarm algorithm were used.

Key–Words: Fed-batch optimal control, Ordinary differential equations, Nonlinear programming.

1 Introduction

A great number of valuable products, like biopharmaceuticals, are produced using fermentation processes and thus optimizing such processes is of great economic importance. In general highly nonlinear and complex differential equations are used to model such fermentation process. The optimization of the fermentation process results in complex optimal control problem where an analytical solution is very hard to obtain.

A special case of the optimal control problem in the fermentation context is the feed trajectory determination on a fed-batch bioreactor. A typical approach to the feed trajectory optimization is to approximate it by using linear segments (linear splines). The resulting trajectory is therefore non-differentiable, making the corresponding optimization problem also non-differentiable. The optimization problem belongs to a well known class of programming problems denoted by semi-infinite [5]. By using the semi-infinite optimality condition we can readily transforme the semi-infinite optimal control problem into a nonlinear optimal control problem (*e.g.* [11]).

While using linear splines to approximate the optimal feed trajectory leads to a simpler semi-infinite programming (SIP) problem the resulting trajectory approximation is not differentiable and therefore the simulated performance can be significantly different from the real one.

In this work we propose a new approach to the feed trajectory planning by using cubic splines instead of linear ones. The resulting approximation is now smooth and expected it to be more realistic than the one obtained with the previous approach.

A particle swarm algorithm was used to obtain the numerical results.

In section 2 we introduce the reader to optimal control and to the herein used notation. Section 3 is used to describe the common approach (the linear splines to approximate the feed trajectory). The new approach is described in section 4. In section 5 we provide some implementation details. We leave the numerical results to section 6 and we conclude in section 7. Some leading research is also presented in the last section.

2 Optimal control trajectory optimization

Many microorganisms are used for producing valuable biopharmaceuticals products. During the fermentation process the biomass and product concentrations changes considerably. The system dynamic behavior motivates the development of optimization techniques to find the optimum input feeding trajectory of substrate in order to obtain a maximum outcome from the process. The outcome can be, for example, the maximum biomass production with a fixed duration time or the minimum time with a fixed amount of substrate.

The optimal control problem is described by a set of ordinary differential equations $\dot{x} = f(x, u, t)$, $x(t_0) = x^0$, $t_0 \leq t \leq t_f$, where x are the state variables and u the input variables that are a function of time t. t_0 and t_f are the inicial and final time, respectively. The performance index J can be generally stated as

$$J(t_f) = \varphi(x(t_f), t_f) + \int_{t_0}^{t_f} \phi(x, u, t) dt,$$

where φ is the performance index of the state variables at final time t_f and ϕ is the integrated performance index during the operation.

Additional constraints on the state and input variables can be imposed that often reflet some physical limitation of the system. The general maximization problem (P) can be posed as

$$\max \ J(t_f) \tag{1}$$

$$s.t. \quad \dot{x} = f(x, u, t) \tag{2}$$

$$\underline{x} \leq x(t) \leq \overline{x}, \ \ \forall t \in [t_0, t_f] \tag{3}$$

$$\underline{u} \leq u(t) \leq \overline{u}, \ \ \forall t \in [t_0, t_f] \tag{4}$$

Problem (P) belong to a well known class of semi-infinite programming problems [5]. These problems are characterized to have a finite number of variables to be optimized subject to an infinite number of constraints.

$x(t)$ and $u(t)$ are functional vector whose components are $x_1(t)$, $x_2(t)$, $\ldots$, $u_1(t)$, $u_2(t)$, $\ldots$, respectively. Whenever x represents an n dimensional vector its components are addressed as $x_1, x_2, \ldots, x_n$ and vice-versa.

3 The common approach - Linear splines

The optimization occurs when determining the optimal input variables u or operational final time t_f. The input variables often represent feeding (or temperature, see [9]) trajectories, *i.e.*, in determining the amount of substrate to be fed into the bioreactor per time unit.

Solving problem (P) in its original formulation is not advisable and unpractical, since no of-the-shelf software for dealing with these type of problems is available. Instead problem (P) can be reformulated as a non-differentiable global optimization problem by imposing a penalty for dealing with constraint (3) and to use linear interpolation for dealing with constraints (4).

Imposing the penalty function for constraints (3) results in redefining the objective function as

$$\hat{J}(t_f) = \begin{cases} J(t_f) & \text{if } \underline{x} \leq x(t) \leq \overline{x}, \ \ \forall t \in [t_0, t_f] \\ -\infty & \text{otherwise} \end{cases}$$

An already proposed strategy to deal with constraints (4) is to interpolate the function $u(t)$ by a polynomial (e.g. [9]). A parameter that makes a great influence in the final trajectory precision is the number of discretization points (knots) of the domain $[t_0, t_f]$. A well known effect on increasing the number of knots (and consequently the polynomial degree) is the increasing poor smoothness of the resulting polynomial. Linear spline interpolation is therefore better suited for approximate function $u(t)$, where increasing the number of knots increases precision without a great affect on smoothness.

A linear interpolating function $w(t)$ (linear spline) is used to approximate the feeding trajectory function $u(t)$. Let t_i, $i = 0, \ldots, n$, denote the time instants (knots) and $h_i = t_i - t_{i-1}$, $i = 1, \ldots, n$, the time displacements. Fixed time intervals are used while the linear spline function values $u(t_i)$, $i = 0, \ldots, n$, are to be computed (in fact we may also use t_i as variables to be optimized, but keeping in mind that ill condition can occur in this case). The linear spline is composed of n linear segments. The spline segment $w^i(t)$, $i = 1, \ldots, n$, is defined as:

$$w^i(t) = u(t_{i-1}) + (t - t_{i-1}) \frac{(u(t_i) - u(t_{i-1}))}{(t_i - t_{i-1})},$$
$$\text{for } t \in [t_{i-1}, t_i], \quad i = 1, \ldots, n.$$

By using the linear spline $w(t)$ to approximate the feed trajectory $(u(t))$ we obtain a SIP problem where constraints (4) are replaced by $\underline{u} \leq w(t) \leq \overline{u}$. By careful inspecting this constraint and by using the optimality conditions for SIP we observe that candidate points to make constraints (4) active, at the solution, are the spline knots and therefore constraints (4) can be replaced by constraints imposing the limit at knots. Constraint (4) can then be replaced by $\underline{u} \leq w(t_i) \leq \overline{u}$, $i = 0, \ldots, n$.

The optimization nonlinear optimization problem (NLP) is then redefined as:

$$\begin{aligned} & \max_{w \in R^{n+1}} \hat{J}(t_f) \\ s.t. \quad & \dot{x} = f(x, w, t) \\ & \underline{u} \leq w(t_i) \leq \overline{u}, \quad i = 0, \ldots, n. \end{aligned} \tag{5}$$

If the initial dynamic system conditions (x^0) are to be considered as variable we may also impose some simple bound constraints on its attainable values. We can also consider $h \in R^n$ and t_f as variables to be optimized increasing the problem dimensional and complexity.

The major motivation for using derivative free optimization codes has to do with the fact that the objective function and the resulting $w(t)$ trajectory function are not differentiable. Recall that using $w(t)$ in the dynamic equation makes them non-differentiable. By using a stochastic algorithm we can also expect to obtain the global optimum for the NLP problem.

4 The new approach - Cubic splines

By obtaining a non-differentiable trajectory approximation one expects bioreactor feed trajectory not to be able to completely follow the optimal feed. A discrepancy between the simulated and real performance is likely to be observed.

Using a smooth approximation to the optimal feed trajectory will result in a better simulated and real performance gap. Meanwhile the trajectory optimal control problem will be differentiable if the ordinary differential equation and the performance index are also differentiable (please note that using an infinity penalty function turns the objective function into a non-differentiable objective function). Still obtaining the problem derivative would be a complex and tedious task and therefore the use of a derivative free optimization algorithm is mostly desirable.

The new approach consists therefore in approximating the optimal feed trajectory by a cubic spline.

The penalty function $\hat{J}$ is again used as the objective function and the feed trajectory $u(t)$ is approximated by a cubic spline $s(t)$. The cubic segment i, $i = 1, \ldots, n$, is defined as

$$\begin{aligned} s^i(t) = {} & \frac{M_{i-1}(t_{i-1} - t)^3}{6(t_i - t_{i-1})} + \frac{M_i(t - t_{i-1})^3}{6(t_i - t_{i-1})} + \\ & \left[\frac{u(t_{i-1})}{t_i - t_{i-1}} - \frac{M_{i-1}(t_i - t_{i-1})}{6} \right] (t_i - t) + \\ & \left[\frac{u(t_i)}{t_i - t_{i-1}} - \frac{M_i(t_i - t_{i-1})}{6} \right] (t - t_{i-1}) \end{aligned}$$

with $t \in [t_{i-1}, t_i]$, where t_i, $i = 0, \ldots, n$, are the time instants.

The semi-infinite programming problem

$$\begin{aligned} & \max_{s \in R^{n+1}} \hat{J}(t_f) \\ s.t. \quad & \dot{x} = f(x, s, t) \\ & \underline{u} \leq s(t) \leq \overline{u}, \quad \forall t \in T \equiv [t_0, t_f]. \end{aligned}$$

is now of a much harder resolution, since a reduction to a NLP is not straightforward.

Given $u(t_i)$, $i = 0, \ldots, n$, getting the global maximizer of the parametric problem,

$$\max_{t \in [t_0, t_f]} s(t)$$

in order to compute which t values make the constraint (4) active is indeed more complex.

To avoid the extra complexity a new redefinition of the objective function is proposed. The control constraint is also included in the infinity penalty objective function resulting in the new objective to be optimized, defined as

$$\bar{J}(t_f) = \begin{cases} \hat{J}(t_f) & \text{if } \underline{u} \leq s(t) \leq \overline{u}, \forall t \in [t_0, t_f] \\ -\infty & \text{otherwise} \end{cases}$$

The optimization is then redefined in the following way

$$\begin{aligned} & \max \ \bar{J}(t_f) \\ s.t. \quad & \dot{x} = f(x, s, t) \\ & \forall t \in T. \end{aligned} \tag{6}$$

In spite of having a differentiable trajectory and an ordinary differentiable system of equations the objective function of problem (6) is still a non-differentiable problem.

5 Implementation details

We briefly describe the details regarding the used environment to address problems (5) and (6).

The AMPL [4] modeling language for mathematical programming was used to code the case studies (see also [2] for another popular modeling language). AMPL provides an easy to use and powerful language, an interface that allow communication with a wide variety of solver (see `www.ampl.com` for a list of available solvers) and the possibility to load an external dynamic library.

By using AMPL as the modeling language modifications can easily be incorporated into the model. If, for example, a constraint in the total allowed glucose addition (t_G) is to be imposed in problem (5), the constraint $\sum_{i=0}^{n-1} h_{i+1}(w(t_i) + w(t_{i+1}))/2 \leq t_G$ can easily be considered in the model.

The AMPL feature to load an external dynamic library was exploited. The external library `fed-batch.dll` (available from the first author) provides five external functions to AMPL related with each test case considered. These external functions, by its turn, use the CVODE [3] package to solve the ordinary differential equations (2). The Newton iteration with the CVDiag module was selected.

The MLOCPSOA [10] solver was used to provide the numerical results shown in the next section. MLOCPSOA stands for Multi-LOCal Particle Swarm Optimization Algorithm. Multi-local optimization addresses the finding of all the local and global optima for an optimization problem. While MLOCPSOA was developed with multi-local optimization in mind by setting an option it reverses to the traditional particle swarm algorithm.

MLOCPSOA provides an interface to AMPL, allowing problems to be easily coded and solved in this modeling language. The MLOCPSOA allows a wide variety of algorithm parameters to be set. The used parameters are `size` for the population size (defaults to $\min(6^n, 1000)$), `maxiter` for the maximum allowed iterations (defaults to 2000) and `mlocal` for multi-local search (defaults to 0 – global search instead of multi-local search). All other parameters were left by default. The reader is pointed for the user manual ([10]) for further details.

6 Numerical results

Numerical results were obtained for five case studies. The parameters used are presented together with the numerical results in Table 1. 'Problem' column refers to the case study (AMPL model file); NT is the number of trajectories in the model; n is the number of time displacements (problems with $n+1$ variables) where equal displacements are considered ($h_i = t_f/n$, $i = 1, \ldots, n$). We include also the objective function value obtained in the literature for the case studies in column '$J(t_f)$'.

Problem	NT	n	t_f	$J(t_f)$
`penicillin` [1]	1	5	132.00	87.99
`ethanol` [1]	1	5	61.20	20839.00
`chemotherapy` [1]	1	4	84.00	14.48
`hprotein` [7, 8]	1	5	15.00	32.40
`rprotein` [7, 6]	2	5	10.00	0.16

Table 1: Test problems and parameters

Problem	t_f	Cubic $J(t_f)$	Linear $J(t_f)$
`penicillin`	132.00	87.70	88.29
`ethanol`	61.20	20550.70	20379.50
`chemotherapy`	84.00	15.75	16.83
`hprotein`	15.00	38.86	32.73
`rprotein`	10.00	0.13	0.12

Table 2: Numerical results

Problem `penicillin` refers to a problem of fed-batch fermentation process where the optimal feed trajectory is to be computed while the penicillin production is to be maximized. `ethanol` refers to a similar optimal control problem where the ethanol production is to be maximized. `chemotherapy` is the only optimal control problem that does not refers to a fed-batch fermentation processe. It is a problem of drug administration in chemotherapy. The optimal trajectory to be computed is the quantity of drug that must be present in order to achieve a specified tumor reduction. While `hprotein` optimal control problem is to compute a unique trajectory (substrate to be fed) problem `rprotein` includes also a trajectory for an inducer. Both problems refer to a maximization for protein production. See Table 1 for the case studies references.

Numerical results for the linear and cubic splines are shown in Table 2. Columns have the same meaning as in the previous table. 'Linear' and 'Cubic' columns reports for the $J(t_f)$ in the linear and cubic splines trajectory approximation, respectively. Recall that $J(t_f) = \hat{J}(t_f) = \bar{J}(t_f)$ for every feasible spline.

MLOCPSOA used a population size of 60 and a maximum of 1000 iterations (reaching a maximum of 60000 function evaluations). Since MLOCPSOA is a stochastic algorithm we performed 10 solver runs for each problem and the best solutions obtained are report on Table 2. $J(t_f)$ is the objective function value and t_f is the final time (t_0 is assumed 0 for all cases).

No attempt is made on comparing our results with the previously published ones, since implementation details are not reported in previous papers. The use

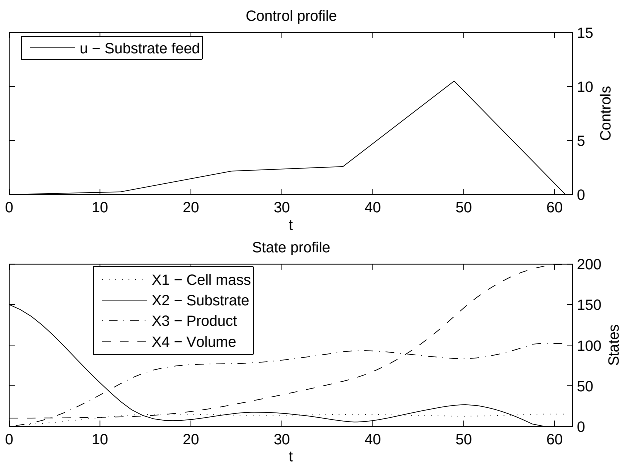

Figure 1: The ethanol case study with linear spline.

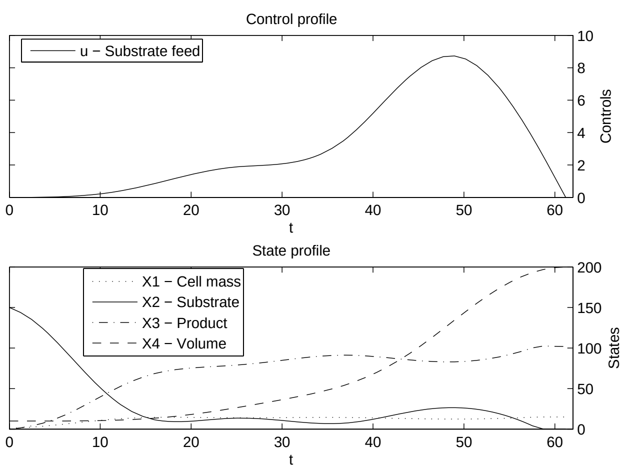

Figure 2: The ethanol case study with cubic spline. Similar solution.

of different techniques to solve the differential equations, discretization step and precision can influence the objective value at the solution found.

Meanwhile we can observe that approximating the feed trajectory by a linear or cubic spline does not greatly influence the fed-batch performance.

`ethanol` was the case study with the greatest performance difference in the obtained numerical results. This difference is due to the finding of a new best feed trajectory for this case. We provide, in figures 1 to 3, plots of the state and control profiles for the `ethanol` case study. In figure 1 we plot the previous results with the feed trajectory approximated by a linear spline. The similar result is plotted in figure 2 where the approximation is a cubic spline. Figure 3 presents the new solution obtained with the trajectory approximated by a cubic spline.

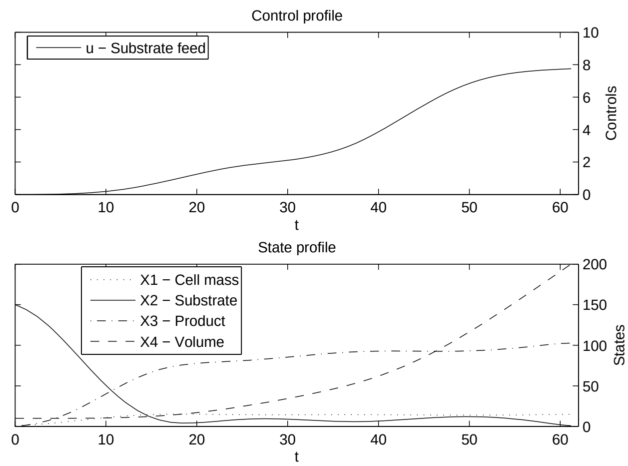

Figure 3: The ethanol case study with cubic spline. Best solution.

7 Conclusions and future work

The optimal control of fed-batch bioprocesses presents challenging nonlinear optimization problems where derivatives do not exist or are unpractical. As in a previous work we address the state constraints by combining it with the problem objective function in an infinite penalty function.

In this paper we add an extra difficulty by using cubic splines to approximate the feed trajectory. The control constraints with the trajectory approximated by cubic splines are handled in a similar way, by adding an infinite violation to the objective function. The resulting nonlinear optimization problem is characterized by possessing a non-convex non-differentiable objective function subject to bound constraints in the variables.

Particle swarm optimization belongs to a class of stochastic algorithms for global optimization and its main advantages are the easily parallelization and simplicity. In this paper we use the MLOCPSOA [10] implementation of the particle swarm paradigm to obtain numerical results with some problem formulations.

No comparison with previous work is provide, since the implementation depends on many parameters external to the problem and algorithm (ordinary differential equation solver, discretization step, etc.).

While no attempt is made to compare the previously obtained solutions with the linear spline approach, we provide a comparison among the linear and cubic spline approaches. While similar results are obtained in terms of objective function values the resulting approximation to the trajectory is smooth.

We expect that the smoothness obtained in the feed trajectory will allow better results in real experiments.

The MLOCPSOA proved to be able to find the problem solution with reasonable accuracy and the particle swarm paradigm proved to be a valuable tool in solving these optimal control problems.

The MLOCPSOA interface with AMPL allowed an easy and fast way to code the five case studies problems. Using the AMPL modeling language together with a developed external dynamic library allows a great flexibility in the problem formulation.

As a future research we plan to study the E. coli bacteria in the same framework. The availability of a lab-scale bioreactor will allow us to confirm that a cubic approximation of the feed trajectory produces better results in practice.

Acknowledgements: The research was supported by the Algoritmi Research Center and by the Portuguese FCT under grant POCI/MAT/58957/2004.

References:

[1] J.R. Banga, E.Balsa-Canto, C.G. Moles, and A.A. Alonso. Dynamic optimization of bioprocesses: Efficient and robust numerical strategies. *Journal of Biotechnology*, 117:407–419, 2005.

[2] A. Brooke, D. Kendrick, A. Meeraus, and Ramesh Raman. GAMS: A User's Guide. http://www.gams.com/, December 1998.

[3] S.D. Cohen and A.C. Hindmarsh. CVODE, a Stiff/Nonstiff ODE solver in C. *Computers in Physics*, 10(2):138–143, 1996.

[4] R. Fourer, D.M. Gay, and B.W. Kernighan. A modeling language for mathematical programming. *Management Science*, 36(5):519–554, 1990.

[5] R. Hettich and K.O. Kortanek. Semi-infinite programming: Theory, methods, and applications. *SIAM Review*, 35(3):380–429, 1993.

[6] J. Lee and W.F. Ramirez. Optimal fed-batch control of induced foreign protein-production by recombinant bacteria. *AIChE Journal*, 40(5):899–907, 1994.

[7] R. Oliveira and R. Salcedo. Benchmark testing of simulated annealing adaptative random search and genetic algorithms for global optimization of bioprocesses. In B. Ribeiro, R.F. Albrecht, A. Dobnikar, D.W. Pearson, and N.C. Steele, editors, *Adaptative and Natural Computing Algorithm*, pages 292–295, Wien, 2005. Springer-Verlag.

[8] S. Park and W.F. Ramirez. Optimal production of secreted protein in fed-batch reactors. *AIChE Journal*, 34(9):1550–1558, 1988.

[9] R. Simutis and A. Lübbert. A comparative study on random search algorithms for botechnical process optimization. *Journal of Biotecnology*, 52:245–256, 1997.

[10] A.I.F. Vaz. MultiLOCal particle swarm optimization algorithm v1.1: User's manual. http://www.norg.uminho.pt/aivaz/.

[11] A.I.F. Vaz and E.C. Ferreira. Optimal control of fed-batch process with particle swarm optimization. In *XXIX Congreso Nacional de Estadística e Investigación Operativa*, pages 972–986, Tenerife, Spain, 2006. ISBN 84-689-8552-X.