JOURNAL OF GUIDANCE, CONTROL, AND DYNAMICS
Vol. 34, No. 1, January–February 2011

Density Functions for Mesh Refinement in Numerical Optimal Control

Yiming Zhao[*] and Panagiotis Tsiotras[†]
Georgia Institute of Technology, Atlanta, Georgia 30332

DOI: 10.2514/1.45852

This paper introduces an efficient and simple method for mesh point distribution for solving optimal control problems using direct methods. The method is based on density (or monitor) functions, which have been used extensively for mesh refinement in other areas such as partial differential equations and finite element methods. Subsequently, the problem of mesh refinement is converted to a problem of finding an appropriate density function. It is shown that an appropriate choice of density function may help increase the accuracy of the solution and improve numerical robustness.

Introduction

THE accuracy and efficiency of mesh refinement algorithms used for solving numerical optimal control problems have motivated a recent research activity in this area. Several mesh refinement methods are proposed in [1], demonstrating the advantage of such algorithms. Reference [2] introduced a mesh refinement method in which integer programming is used to minimize the maximum integration error during mesh refinement iterations. Reference [3] proposed a multiresolution trajectory optimization algorithm that refines a nonuniform mesh using local dyadic subdivisions after each iteration. A common strategy behind these mesh refinement methods is the redistribution of the mesh points based on the local integration/interpolation error.

When the solution of the optimal control problem exhibits discontinuities in the control or its higher order derivatives, a locally dense mesh is typically necessary to achieve better resolution, and obtain more accurate estimation of the location of the discontinuity. Mesh generation based on the local integration/interpolation error does not use any special treatment of the discontinuities, especially those appearing in higher order derivatives of the control or the state variables. For better accuracy, it is necessary to estimate the location of such irregularities (namely, discontinuities in the control history and/or its higher order derivatives) and subsequently incorporate this information into the mesh refinement process. A mesh refinement method following this philosophy has been proposed in [4]. This method divides the time interval at the points with maximum absolute value of the first derivative of the control, but it does not capture higher order discontinuities in the control time history.

Mesh generation and adaptation is a common topic in many areas of engineering and applied mathematics. The notion of mesh density function for mesh generation and refinement has been used in the finite element method field [5,6]. The concept of density functions is also similar to monitor functions used for the numerical solution of partial differential equations [7]. However, despite their popularity in other fields, mesh density/monitor functions have rarely been used for discretizing optimal control problems. The only exception appears to be [8], as far as the authors know. Additional studies are needed to understand how the density/monitor functions can be used in numerical optimal control and how they can influence the accuracy and robustness of numerical optimal control algorithms. Furthermore, the choice of "good" density/monitor functions for mesh discretization of optimal control problems seems to be open.

In this paper we attempt to provide a partial answer to the previous questions. We introduce a method to distribute the mesh points efficiently using density/monitor functions. Although different monitor functions can be used for mesh generation, an appropriate choice of a monitor function can generate a better quality mesh, and can improve the accuracy of the solution along with the speed of convergence. Hence, the problem of mesh generation can be treated as a problem of finding an appropriate density/monitor function. Two possible choices of density functions are used in the numerical examples, based on the discrete control/state histories from the previous iteration during the mesh refinement process. The proposed method avoids the numerical integration step and the use of ODE solvers for the system dynamics as was done in [8]. Yet, it generates a mesh with a suitable level of adaptive discretization that provides sharp resolution around the places where the control switches or the trajectory meets/leaves state constraints, thus resulting in better accuracy of the overall final solution. Numerical examples are presented to demonstrate the advantage of the proposed method, and comparisons are provided against the industry-standard sparse optimal control software (SOCS).

Problem Statement and Nonlinear Programming Formulation

We consider an optimal control problem minimizing the following Bolza cost functional:

$$\mathcal{J} = \Phi(x(t_0), t_0, x(t_f), \mathbf{p}, t_f) + \int_{t_0}^{t_f} \mathcal{L}(x(t), u(t), \mathbf{p}, t)\,\mathrm{d}t \tag{1}$$

where $t \in [t_0, t_f] \subseteq \mathbb{R}$ is the time, $x\colon [t_0, t_f] \to \mathbb{R}^n$ is the vector of state variables, $u\colon [t_0, t_f] \to \mathbb{R}^m$ is the vector of control variables, and $\mathbf{p} = [p_1, p_2, \ldots, p_l] \in \mathbb{R}^l$ the vector of additional optimization parameters. The Mayer term $\Phi\colon \mathbb{R}^n \times [t_0, t_f] \times \mathbb{R}^n \times \mathbb{R}^l \times [t_0, t_f] \to \mathbb{R}$, and the Lagrangian term $\mathcal{L}\colon \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l \times [t_0, t_f] \to \mathbb{R}$ are given functions of suitable smoothness properties. Our objective is to minimize the cost (1) subject to the dynamic constraints

$$\dot{x}(t) = f(x(t), u(t), \mathbf{p}, t), \qquad t_0 \le t \le t_f \tag{2}$$

the boundary conditions

$$\Psi(x(t_0), t_0, x(t_f), t_f, \mathbf{p}) = 0 \tag{3}$$

and the path constraints

Presented as Paper 2009-2019 at the AIAA Infotech at Aerospace Conference and AIAA Unmanned...Unlimited Conference, Seattle, WA, 6–9 April 2009; received 5 June 2009; revision received 3 October 2010; accepted for publication 8 October 2010. Copyright © 2010 by Y. Zhao and P. Tsiotras. Published by the American Institute of Aeronautics and Astronautics, Inc., with permission. Copies of this paper may be made for personal or internal use, on condition that the copier pay the $10.00 per-copy fee to the Copyright Clearance Center, Inc., 222 Rosewood Drive, Danvers, MA 01923; include the code 0731-5090/11 and $10.00 in correspondence with the CCC.

[*]Ph.D. Candidate, School of Aerospace Engineering; yzhao7@gatech.edu. Student Member AIAA.

[†]Professor, School of Aerospace Engineering; tsiotras@gatech.edu. Fellow AIAA.

$$C(x(t), u(t), \mathbf{p}, t) \leq 0, \qquad t_0 \leq t \leq t_f \tag{4}$$

where $\Psi\colon \mathbb{R}^n \times [t_0, t_f] \times \mathbb{R}^n \times [t_0, t_f] \times \mathbb{R}^l \to \mathbb{R}^{N_\Psi}$ and where $C\colon \mathbb{R}^n \times \mathbb{R}^m \times \mathbb{R}^l \times [t_0, t_f] \to \mathbb{R}^{N_C}$.

To solve this problem through nonlinear programming, the states and controls are discretized on a mesh $\{t_i\}_{i=0}^N$ for some positive integer N, with $t_N = t_f$ and $t_i < t_{i+1}$ for $0 \leq i \leq N-1$. Let X be the decision variable vector, including those determined by the discretization scheme. The corresponding discretization of the continuous time optimal control problem (1–4) can then be written as

$$\min_{\mathbf{X}} J(\mathbf{X}) \tag{5}$$

subject to

$$|F(\mathbf{X})| \leq \zeta_d \tag{6}$$

$$|\tilde{\Psi}(\mathrm{X})| \leq \zeta_{\mathbf{b}} \tag{7}$$

and

$$C(\mathbf{X}) \leq \zeta_C \tag{8}$$

where the absolute value $|\cdot|$ and the inequalities are enforced elementwise, J, F and $\tilde{C}$ are appropriate discretizations of the cost function, dynamics constraint and path constraint of the original problem, respectively, $\tilde{\Psi}$is the boundary condition, and $\zeta_d \in \mathbb{R}^{Nn}$, $\zeta_b \in \mathbb{R}^{N_\Psi}$ and $\zeta_C \in \mathbb{R}^{(N+1)\cdot N_C}$ represent defect vectors, whose elements are small positive real numbers. In particular, for the discretization of the differential constraint (2), the function J in (5) and F in (6) are obtained using a class of R-K methods ensuring consistency, such that the solution of the discrete problem converges to that of the continuous time problem [1]. For more details the reader may refer to [1,2,9,10].

Density Functions and Mesh Point Distribution

Density Function and Mesh Generation

A mesh density function, or simply a density function, is a nonnegative function $\bar{f}\colon [a, b] \to \mathbb{R}_+$, $a, b \in \mathbb{R}$ that satisfies $\int_a^b \bar{f}(t)\,\mathrm{d}t = 1$, and is zero (at most) at countably many points. Since any nonnegative function $f\colon [a, b] \to \mathbb{R}_+$ that has only countably many zeros can be normalized as

$$\bar{f}(t) = \frac{f(t)}{\int_a^b f(\tau)\,\mathrm{d}\tau} \tag{9}$$

to obtain a mesh density function, from now on we may assume, without loss of generality, that any function f applied to mesh refinement has been already normalized. The corresponding cumulative distribution function $F\colon [a, b] \to [0, 1]$ is defined by

$$F(t) \triangleq \int_a^t \bar{f}(\tau)\,\mathrm{d}\tau \tag{10}$$

The value of $F(t)$ corresponds to the area under the graph of $\bar{f}$ between a and t. Clearly, $F(a) = 0$ and $F(b) = 1$. In the sequel, and without loss of generality, we will assume that $[a, b]$ is the unit interval. Consider a mesh $\{t_i\}_{i=0}^N$ containing a total of $N+1$ points with $t_0 = 0$ and $t_N = 1$. Given a density function f, let F be the cumulative distribution function determined by f as in (10). For $i = 0, 1, \ldots, N-1$, with the ith point at t_i, the position of the $(i+1)$th point can be decided by

$$F(t_{i+1}) - F(t_i) = \frac{1}{N} \tag{11}$$

A mesh can then be generated based on the density function f, such that the distribution of grid points conforms to an equidistribution of F. Alternatively, the mesh is dense where the value of $f(t)$ is large.

The previous mesh point allocation strategy usually requires solving a nonlinear algebraic equation repeatedly $N-1$ times, which can be a quite time-consuming task when N is large. An alternative technique for achieving equidistribution requires the integration of a system of ODEs, including the transformed dynamics and the inverse of the density function [8]. The integration of dynamics requires intensive computations, especially when the dimension of the problem is large. Besides, integration is also sensitive to the accuracy of the boundary conditions (if not fixed) and the accuracy of the control history obtained from the previous iteration.

To avoid the process of repeatedly solving nonlinear equations or integrating the system dynamics, an interpolation method is used in this work to compute the points $\{t_i\}_{i=1}^{N-1}$, by taking advantage of the monotonicity of F. Specifically, given any density function f, select a grid $\{t_j\}_{j=0}^{N_j} \in [0, 1]$, which contains N_j points. During the mesh refinement iterations, $\{t_j\}_{j=0}^{N_j}$ could be chosen as the mesh used in the previous iteration. Now $y_j = F(t_j)$ can be easily calculated by $y_j = \int_0^{t_j} f(\tau)\,\mathrm{d}\tau$. For any $y \in [0, 1]$, define the inverse mapping $F^{-1}(y) = \{t \mid \int_0^t f(\tau)\,\mathrm{d}\tau = y\}$. From the properties of f, and hence F, the inverse F^{-1} is well defined and also continuous, with $t_j = F^{-1}(y_j)$. The set of pairs $\{(y_j, t_j)\}_{j=0}^{N_j}$ is then a discrete representation of the function F^{-1}. Note that the first and the last grid points are at $t_0 = 0$ and $t_N = 1$, respectively. For the allocation of the other grid points, the location t_i of the ith mesh point can be obtained by interpolating $\{(y_j, t_j)\}_{j=1}^{N_j}$ using a spline function at the position $y_i = (i-1)/(N-1)$ for $2 \leq i \leq N-1$. Using this method, as long as the selected partition is dense enough, the location of all mesh points can be calculated very fast and with high accuracy. Note that the mesh point distribution is unique once the density function is given, but the converse is not true.

Figure 1 shows the mesh point distribution obtained by two specific density functions over the unit interval. The density function in the upper left of the figure is the linear function $f(t) = t$. The resulting mesh is shown in the upper right of the figure. The lower left plot shows the density function $f(t) = e^{-50t^2+20t-2} + e^{-50t^2+80t-32}$, with its mesh shown in the lower right of the figure. In both cases, the mesh contains a total of 20 grid points.

Selection of Density Function

By definition, a mesh density function needs only to be nonnegative (and integrable). This generality provides a great deal of flexibility for achieving desired mesh point distributions and for designing different mesh refinement schemes. The particular choice of the density function can have a major impact on the numerical performance of the overall algorithm.

Certain density functions can be used to regulate the integration error. For example, if the density function is chosen as a piecewise constant function whose value on each subinterval equals the corresponding principal local truncation error function (PLTE) as in [1], then the mesh point distribution process will be the static mesh refinement Strategy 1 introduced in the same reference. This strategy tries to approximately equidistribute the PLTE, and as a result, the mesh points would be denser where the PLTE was large in the previous iteration.

Another strategy for designing a good density function is to provide better approximation to the state and/or control histories to improve the accuracy of the solution. This approach places more emphasis on the geometric properties of the *graph* of the function to be approximated. The arc length monitor function in [8], for example, equidistributes the grid points along the graph of the state. As another example, the curvature-based density function proposed in [11] provides the best piecewise linear interpolative approximation of the function of interest in the L_1 space. As it will be shown later in the paper, this density function is capable of capturing higher

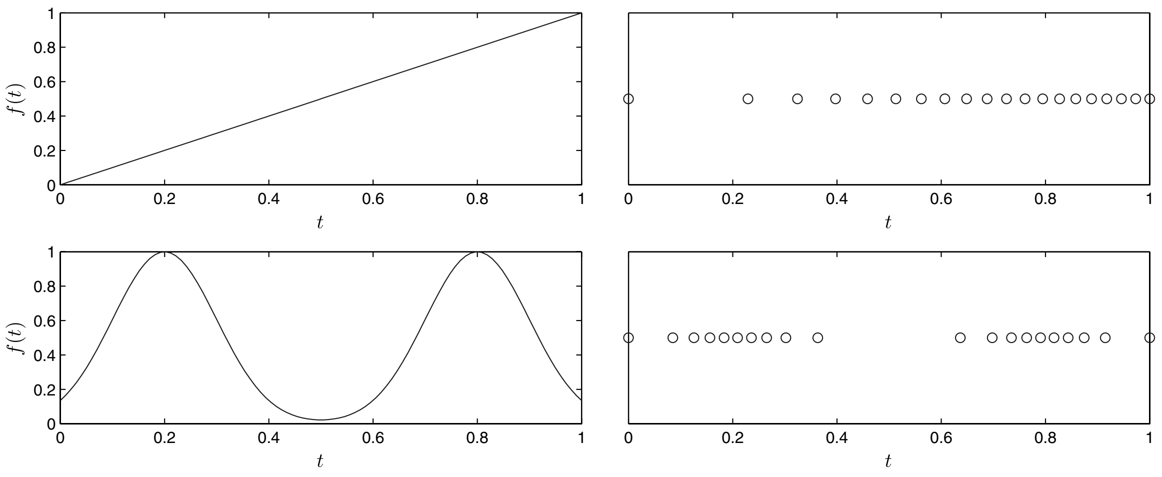

Fig. 1 Density functions and corresponding distribution of grid points.

order discontinuities of the function to be approximated with sufficient accuracy.

For more general mesh refinement schemes, it may be desirable to add new points only within certain specific time spans of the control and state histories, namely at those regions where the control or state histories exhibit discontinuities or smoothness irregularities (e.g., very fast rate of change and/or discontinuities in higher order derivatives) while keeping other points fixed. This objective can also be easily achieved by defining multiple density functions on disjoint intervals; then the number of points assigned to each interval is proportional to the integral of the corresponding density function. The points are then distributed using the preceding method. More details about this procedure are given in [11].

Although the density function uniquely determines the mesh once the total number of grid points is given, it does not provide any information what the size of the final mesh should be. In the density function-based mesh refinement algorithm (DENMRA) proposed later in the paper, the discretization error estimation method in [2] is used to determine the size of the mesh in order to ensure that the new mesh provides a better discretization compared with the one from the previous iteration.

Density Function-Based Mesh Refinement Algorithm

The use of a density function is one of the key components in DENMRA. General optimal control problems involve ordinary differential equations in terms of the state variables, which describe how the control changes the vector field of the states. For such problems, since the states are continuous, irregularities in the smoothness in the states usually correspond to fast (or discontinuous) changes in the control. Hence, typically, the control history is used in DENMRA for computing the density function to capture smoothness irregularities in both the state and the control histories, although this is not restrictive. The state histories can be used as well, if needed.

When solving a general optimal control problem that minimizes the cost function J using m control inputs, DENMRA goes through the following four major steps:

1) Set $j = 1$. Choose a positive integer N_j and generate the initial (coarse) uniform mesh $T_1 = \{t_i\}_{t_i=1}^{N_j}$, where $t_i = (i-1)/(N_j - 1)$, Generate an initial guess for the state and control variables, and solve the discretized problem that minimizes J;

2) Calculate the density function f using the discretized control $\{(t_i, \mathbf{u_i})\}_{i=1}^{N_j}$ of the previous solution, where $\mathbf{u_i} \in \mathbb{R}^m$;

3) Determine the mesh size increment ΔN_j by discretization error estimation which is introduced in [2]. Let $N_{j+1} = N_j + \Delta N_j$, and generate the new mesh $T_{j+1} = \{t_i\}_{i=1}^{N_{j+1}}$ based on f. Set $j = j + 1$;

4) Generate the initial guess based on the previous solution for mesh T_j, solve the problem, and go to step 2, unless some stopping rule is met.

The details of these steps are given next.

Initial Guess

For simplicity, DENMRA may start from a constant initial guess for all control and state variables, but (as is typical with nonlinear optimization problems) any good initial guess based on prior experience with the problem or good engineering judgment can improve convergence.

Optimization

After the cost function and the dynamic, state, control and path constraints have been discretized on the given grid, DENMRA calls a nonlinear programming solver. In this implementation, we have used the optimization software SNOPT [12] for solving the corresponding nonlinear programming problem stemming from the discretized optimal control problem.

Density Function Computation

In DENMRA, when the density function based on the local curvature as described in [11] is used, the discrete control $\{(t_i, \mathbf{u_i})\}_{i=1}^{N_j}$ from the previous iteration is used to estimate the curvature of the graph of the control history. The calculation of the density function corresponding to the control $\mathbf{u}$ is therefore computed as follows:

1) Let $u_{i,k}$ be the kth component of the discrete control value $\mathbf{u_i}$ at t_i, $\dot{u}_{i,k}$ be the first order derivative of the kth component of control at time $t_i' = (t_{i+1} + t_i)/2$, and $\ddot{u}_{i,k}$ be the second order derivative at time $t_i'' = (t_{i+1}' + t_i')/2$. Then, for $k = 1, \ldots, m$, the values $\{\dot{u}_{i,k}\}_{i=1}^{N_j-1}$ and $\{\ddot{u}_{i,k}\}_{i=1}^{N_j-2}$ can be approximated by $\dot{u}_{i,k} \approx (u_{i+1,k} - u_{i,k})/(t_{i+1} - t_i)$ and $\ddot{u}_{i,k} \approx (\dot{u}_{i+1,k} - \dot{u}_{i,k})/(t_{i+1}' - t_i')$, respectively. Interpolate $\{(t_i', \dot{u}_{i,k})\}_{i=1}^{N_j-1}$ using a spline function at t_i'' and obtain $\{(t_i'', \dot{u}_{i,k}')\}_{i=1}^{N_j-2}$.

2) Let

$$\kappa_{i,k} = \frac{|\ddot{u}_{i,k}|}{(1 + \dot{u}_{i,k}'^2)^{3/2}}, \qquad i = 1, \ldots, N_j - 2 \tag{12}$$

The curvature function $\kappa_k(t)$ for the kth control component is constructed using interpolation and/or extrapolation at the points $\{(t_i'', \kappa_{i,k})\}_{i=1}^{N_j-2}$ using a spline function. Then $\rho_k(t) = c_k \kappa_k(t)^{\frac{1}{3}}$.

3) The overall (nonnormalized) density function f is obtained by merging the density functions for all controls. For instance,

$$f(t) = \left(\sum_{k=1}^{m} \rho_k^2(t)\right)^{\frac{1}{2}} \tag{13}$$

and

$$f(t) = \max_k \rho_k(t) \tag{14}$$

are two possible methods to generate the overall density function.

Note that when the control histories are identically zero on some subinterval $\mathcal{I}_s \subset [0, 1]$, then f does not satisfy all the properties of a density function. For this reason, during implementation in DENMRA, f is modified as follows:

$$f_\varepsilon(t) = \begin{cases} f(t), & f(t) \geq \varepsilon, \\ \varepsilon, & f(t) < \varepsilon \end{cases} \tag{15}$$

where ε is a small positive real number. With this modification, f_ε is a strictly positive function such that the preceding mesh generation can be applied. In practice, this means that a few grid points are kept even on the parts of the control history that are straight lines or segments with very small curvature. This is always a good strategy since the control history on $\mathcal{I}_s$ may change in subsequent iterations, and it is thus advisable to keep some points in the interior of the interval $\mathcal{I}_s$ in order to capture possible changes of the control histories.

Mesh Generation

DENMRA typically starts with a coarse uniform mesh in order to capture the basic structure of the control history. In subsequent iterations, the user can either let DENMRA decide the mesh size based on the integration error, or adjust the final mesh size and the number of iterations according to the desired or imposed speed and accuracy requirements, depending on the problem at hand. In the former case, at each mesh refinement iteration, cubic splines are used to approximate the state and control histories, and the local discretization error of the previous mesh is estimated. After the density function is computed based on the result of the previous iteration, a temporary new mesh size $\tilde{N}_j$ is found by gradually increasing $\tilde{N}_j$ from N_j until the maximum local discretization error of the new mesh generated using the density function with $\tilde{N}_j$ points is smaller than that of the previous mesh. Let $N_{\max}$ be a limit on the final mesh size, then the actual mesh size increment after the jth iteration is determined by $\Delta N_j = \min\{\tilde{N}_j - N_j, \Delta N_{\max}\}$, where $\Delta N_{\max} = N_{\max} - N_j$. If $\Delta N_j = \tilde{N}_j - N_j$, then the last temporary mesh is used for the next iteration. Otherwise, a new mesh is generated with $N_j + \Delta N_{\max}$ points.

Stopping Rule

DENMRA stops either when the maximum number of mesh refinement iterations is reached, or when the optimality of the problem cannot be further improved and the local integration error is smaller than the specified tolerance.

Numerical Examples

In this section we report the results from two numerical examples that illustrate the good properties of the proposed mesh generation method. The first example is the double integrator minimum energy problem [13]. Since this problem has an analytical solution, it can be used to check the accuracy and optimality of the proposed method. This problem also includes a state constraint, which is used to demonstrate that the proposed methods is able to handle higher order state irregularities stemming from such state constraints. The second example deals with a "hypersensitive" optimal control problem [14] and it is used to test the robustness of the method when dealing with problems requiring highly concentrated grid points during certain phases of the solution. For comparison, the same two problems are also solved using SOCS [15], which is a widely used software for solving trajectory optimization problems. Both algorithms start with trapezoidal integration, and switch to higher order Hermite–Simpson integration later on to meet the desired accuracy/optimality. A feasibility tolerance of 10^{-10} is used for both algorithms.

Minimum Energy for Double Integrator

The double integrator problem is given by

$$\dot{v} = u, \quad v(0) = -v(1) = 1, \quad \dot{x} = v, \quad x(0) = x(1) = 0 \tag{16}$$

and the goal is to find $u(t)$, where $0 \leq t \leq 1$, to minimize

$$J = \frac{1}{2}\int_0^1 u^2 \,\mathrm{d}t \tag{17}$$

with the state constraint $x(t) \leq \ell$, where ℓ is a positive real number.

The optimal control $u^*(t)$ for this problem can be obtained as follows [13]:

$$u^*(t) = -2, \qquad 0 \leq t \leq 1, \quad \text{for } \ell \geq \frac{1}{4} \tag{18}$$

$$u^*(t) = \begin{cases} -8(1-3\ell) + 24(1-4\ell)t, & 0 \leq t \leq \frac{1}{2}, \\ -8(1-3\ell) + 24(1-4\ell)(1-t), & \frac{1}{2} < t \leq 1, \end{cases} \quad \text{for } \frac{1}{6} \leq \ell < \frac{1}{4} \tag{19}$$

$$u^*(t) = \begin{cases} -\frac{2}{3\ell}\left(1 - \frac{t}{3\ell}\right), & 0 \leq t \leq 3\ell, \\ 0, & 3\ell < t \leq 1 - 3\ell, \\ -\frac{2}{3\ell}\left(1 - \frac{1-t}{3\ell}\right), & 1 - 3\ell < t \leq 1, \end{cases} \quad \text{for } \ell < \frac{1}{6} \tag{20}$$

Comparison in Terms of Accuracy and Optimality

The curvature based-density function is used for mesh refinement in DENMRA for this problem. This density function is given by $\rho_\kappa(t) = \kappa(t)^{1/3}$, $t \in [0, 1]$, where κ is the curvature of the graph of the control function. As shown in [11], this density function provides the best piecewise linear interpolative approximation. The same problem was also solved using the commercial numerical optimal control code SOCS, which implements the mesh refinement strategy of [2]. Both algorithms were tested on the same computer, and cold-started using the same linear initial guess.

Table 1 summarizes the results from DENMRA and SOCS for the double integrator problem. In the table, N is the size of the final mesh, $|J - J^*|$ is the optimality error, and $\|u_i - u^*(t_i)\|_\infty = \max_i |u_i - u^*(t_i)|$ is the norm of the error between the discretized control $\{u_i\}_{i=1}^N$ and the exact solution u^*. Our numerical experiments showed that SOCS could not achieve highly accurate solution for this problem even if the local integration error tolerance has been set to 10^{-14}. The optimality error of the SOCS solution was around $10^{-4} \sim 10^{-6}$ with a maximum control error around $10^{-2} \sim 10^{-3}$. DENMRA exhibited an optimality error at the order $10^{-7} \sim 10^{-13}$, and a maximum control error at the order of $10^{-5} \sim 10^{-6}$.

The mesh refinement histories of the two algorithms for the case with $\ell = 0.05$ are shown in Figs. 2 and 3. In these figures, the vertical dotted lines indicate the points of discontinuities in the analytical solution (at $t = 0.15$ and $t = 0.85$). As can be seen from Fig. 3, when DENMRA is used to solve this problem, the grid points get denser around the two points with discontinuities in the control derivative

Table 1 Comparison of precision and optimality

ℓ	Algorithm	N	$\lvert J - J^*\rvert$	$\lVert u_i - u^*(t_i)\rVert_\infty$
0.04	SOCS	99	7.5×10^{-5}	4.2×10^{-3}
0.04	DENMRA-ρ_κ	40	8.9×10^{-7}	4.4×10^{-5}
0.08	SOCS	99	6.9×10^{-6}	1.4×10^{-3}
0.08	DENMRA-ρ_κ	40	1.9×10^{-8}	4.8×10^{-5}
0.12	SOCS	50	9.6×10^{-5}	3.9×10^{-3}
0.12	DENMRA-ρ_κ	40	1.2×10^{-9}	1.0×10^{-5}
0.16	SOCS	50	7.2×10^{-5}	1.8×10^{-2}
0.16	DENMRA-ρ_κ	40	2.7×10^{-13}	5.8×10^{-6}

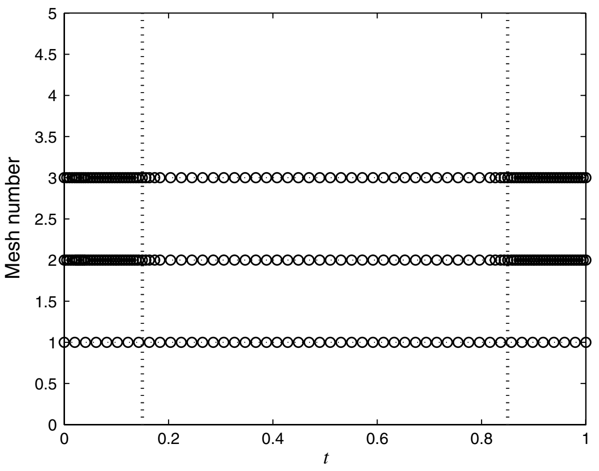

Fig. 2 Mesh refinement, SOCS, $\ell = 0.05$.

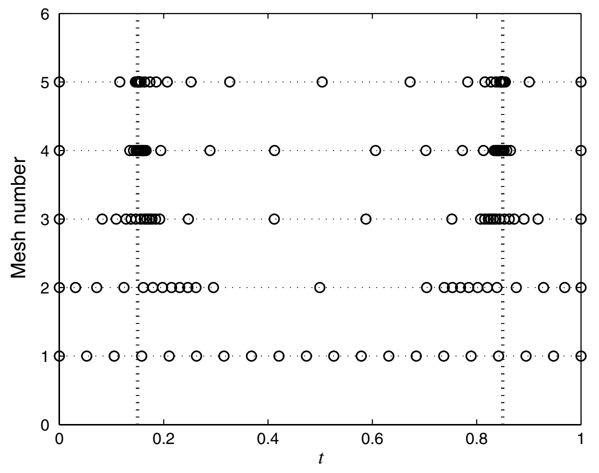

Fig. 3 Mesh refinement, DENMRA, $\ell = 0.05$.

after each iteration, thus providing a better resolution. The mesh refinement scheme in SOCS is based on the integration error, and allocates more points on the two intervals (0, 0.15) and (0.85, 1), where the absolute value of $\dot{u}^*$ is large, but beyond this, the discontinuities in control did not receive any additional special treatment. As a result of this mesh refinement procedure, SOCS always keeps the points from the previous mesh, and hence tends to generate a larger mesh size. By solving this problem with different values of ℓ, it was confirmed that, for this problem, the mesh generated by DENMRA provides better resolution around the points of discontinuities.

Comparison in Terms of Resolution

By "resolution" here we mean not only the ability of an algorithm to capture the discontinuities in the control history or its higher order derivatives using a locally denser grid, but also the ability to distinguish adjacent points of discontinuity.

1) When $\ell \geq 1/6$, the optimal control $u^*(t)$ is either constant or smooth, and both DENMRA and SOCS converge to the theoretical solution.

2) When $\ell < 1/6$, the optimal control $u^*(t)$ contains two corners. It is challenging to distinguish these corners when ℓ tends to zero or $1/6$: in the former case, the corners are very close to the endpoints of the mesh, and the fast change of control between the corner and the corresponding end point makes it difficult to obtain an accurate solution; in the second case, the two points of discontinuity tend to merge, which makes them difficult to distinguish.

The resolution test results are listed in Table 2. The resolution is denoted by Δt. Both algorithms were able to gradually decrease ℓ until $\|u_i - u^*(t_i)\|_\infty \leq 10^{-2}$ without inducing any algorithm failure. When $\ell \to 0$, $\Delta t = 3\ell$, where Δt is the distance between the discontinuities and the nearby endpoints of the mesh. When $\ell \to 1/6$, $\Delta t = 1 - 6\ell$, which is the distance between the two points of discontinuity. In both cases, a smaller Δt means a better resolution. For all test cases, DENMRA terminates with 40 points, SOCS starts from 50 points, and the final mesh sizes have 83 points when $\ell = 0.025$, and 50 points when $\ell = 0.153$. As shown in Table 2, DENMRA provides sharper resolution than SOCS while preserving the accuracy of the solution.

Hypersensitive Problem

This problem minimizes the cost function

$$J = \int_0^{t_f} (x^2(t) + u^2(t))\,\mathrm{d}t \tag{21}$$

subject to the differential constraint

$$\dot{x} = -x^3 + u \tag{22}$$

and endpoint state constraints $x(0) = 1, x(t_f) = 1.5$. For large values of t_f, the solution of this hypersensitive problem has a three-segment structure with two boundary layers [14], namely, a "takeoff, cruise and landing" structure. The "cruise" phase is determined by the cost function and the system dynamics, while the "takeoff" and "landing" phases are determined by the boundary conditions, cost function, system dynamics, and the requirement to reach the cruise phase.

As pointed out in [14], the key to solving hypersensitive problems using direct methods is to use a denser grid during the boundary layers (takeoff and landing phases) in which the state changes fast; a nonuniform mesh is imperative for the solution of this problem with large values of t_f. The hypersensitive problem with large t_f is suitable for testing the robustness of mesh refinement algorithms, because the length of the cruise phase increases with respect to t_f, which makes it more difficult to allocate enough grid points to the two boundary layers. We solved this problem for various values of t_f using both SOCS and DENMRA. Observing that the boundary layer is characterized by a large absolute value of the derivative of control, we used the density function $f(t) = |\dot{u}(t)|^{\frac{1}{2}}$ to capture these boundary layers during mesh generation in DENMRA.

SOCS was started from a mesh containing 150 points, and the maximum number of mesh refinements was set at 15. DENMRA started from a uniform mesh containing 25 points, with a maximum number of 15 mesh refinement iterations and a maximum mesh size of $N_{\max} = 100$. The problem was solved on the same computer as in the previous example. The results are summarized in Fig. 4.

In our numerical experiments when ρ_κ is used for mesh generation and refinement, DENMRA failed to allocate enough points at both ends of the mesh, and did not converge for large values of t_f. In contrast, the use of the density finction $f(t) = |\dot{u}(t)|^{\frac{1}{2}}$ captures a larger region of the two boundary layers. Figure 4 shows the result of DENMRA using the previous density function for $t_f = 1 \times 10^5$. As can be seen from this figure, the majority of the grid points are successfully allocated inside the two boundary layers.

Both SOCS and DENMRA were challenged by solving this hypersensitive problem for t_f as large as possible. To estimate the maximum solvable value of t_f, each algorithm was used to solve the

Table 2 Comparison of resolution

Algorithm	ℓ	ID	Δt	$\lvert J - J^*\rvert$	$\|u_i - u^*(t_i)\|_\infty$
SOCS	0.025	D1[a]	0.075	8.2×10^{-4}	8.5×10^{-3}
SOCS	0.153	D2[b]	0.082	2.8×10^{-5}	8.5×10^{-3}
DENMRA-ρ_κ	0.014	D1	0.042	7.3×10^{-9}	1.7×10^{-4}
DENMRA-ρ_κ	0.1662	D2	0.0028	1.9×10^{-9}	9.0×10^{-4}

[a]The smallest ℓ keeping $\|u_i - u^*(t_i)\|_\infty \leq 10^{-2}$ without algorithm failure
[b]The largest ℓ keeping $\|u_i - u^*(t_i)\|_\infty \leq 10^{-2}$ while separating the discontinuities

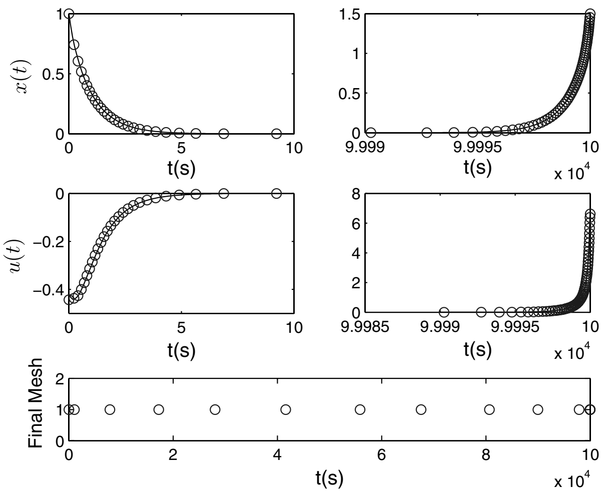

Fig. 4 DENMRA solution, $t_f = 100,000$.

hypersensitive problem for an increasing sequence of t_f values starting from $t_f = 100$. Numerical results showed that the optimal value is $J^* \approx 6.724$. If the problem was successfully solved with the final objective value $J < 7$, then t_f was updated as $t_f = t_f + \Delta t_f$, where $\Delta t_f = 10^N$ if $10^N \le t_f < 10^{N+1}$, for some positive integer N, and the problem was solved again with the new t_f. This process was repeated until $J \ge 7$. The results are shown in Table 3. As shown in the table, DENMRA exhibited good robustness by solving the hypersensitive problem for large values of t_f, which is attributed to its ability to redistribute the grid points to the boundary layers even with the presence of very long cruise phases. As a matter of fact, DENMRA was able to provide a solution up to a maximum value of $t_f = 2 \times 10^6$, whereas SOCS provided a solution up to a maximum value of $t_f = 30,000$.

The optimality of SOCS and DENMRA is shown in Table 4. It was found that the optimality of the results obtained by DENMRA deteriorates when t_f is very large, while the optimality of the SOCS solution is consistent within the range of t_f values it can solve. The mesh refinement histories of two algorithms are similar, except for the fact that the mesh generated by SOCS contains an order of magnitude more grid points.

In [8] a density (monitor) function of the form $\varphi(x,u) = (\alpha + \sum_{i=1}^{n} \beta_i g_i(x,u))^{1/2}$ where $g_i(x,u)$ is the ith component of the system dynamics, and α and β_i are constants to be adjusted, was used to initialize SOCS for solving the hypersensitive problem. This "arc length" monitor function was also tested for mesh refinement. It was found that when DENMRA uses this arc length monitor function, the maximum solvable t_f value is 10,000. A density function providing an equidistribution along the arc length of the graph of the system state is therefore not the best choice for mesh refinement for this specific problem.

Table 3 Hypersensitive problem, robustness test

Algorithm	t_f	N_{Iter}	N_f	J
SOCS	30,000	15	475	6.7241
DENMRA-f	2×10^6	15	100	6.8211

Table 4 Hypersensitive problem, optimality test

Algorithm	t_f	N_{Iter}	N_f	J
SOCS	2×10^2	11	1020	6.7241
SOCS	2×10^3	14	1201	6.7241
SOCS	2×10^4	15	1014	6.7241
DENMRA-f	2×10^2	13	100	6.7240
DENMRA-f	2×10^3	13	100	6.7240
DENMRA-f	2×10^4	15	100	6.7239

Conclusions

A new mesh refinement method is proposed, which is based on a mesh density function that determines the mesh point distribution. By using an appropriate density function, the proposed DENMRA generates a nonuniform mesh by suitably allocating the grid points over the whole time interval, putting emphasis on the points of discontinuity of the control variables or on the nonsmoothness of the state variables. The grid point allocation process is completely automatic. Two density functions are also introduced, one based on the local curvature of the graph of the intermediate solution and the other based on the first derivative of the control variable. The density function can also be chosen as the integration error, leading to the mesh refinement scheme proposed in [1]. Numerical results in the paper have shown that DENMRA automatically maintains an appropriate local level of discretization over the whole control and state time histories for different problems. The grid generation is very simple and easy to implement, while still maintaining high numerical accuracy for the overall solution. The numerical examples also demonstrated the importance of choosing an appropriate density function that captures the smoothness irregularities in the intermediate solution for best accuracy, optimality and robustness, especially when solving challenging problems.

Another attractive advantage of DENMRA is that it can be used to distribute a fixed number of grid points so as to maximize the accuracy of the final solution. In terms of real-time (or close to real-time) applications, this may be of greater interest, since the number of decision variables and constraints of the resulting nonlinear optimization problem is related to the number of grid points used. If the computational resources impose limitations on the number of constraints that can be handled during each iteration, it makes sense to limit the size of the optimization problem by keeping the number of grid points fixed. This can be easily achieved using the proposed algorithm.

Acknowledgments

This work has been supported by NASA with contract no. NNX08AB94A, monitored by NASA Ames Research Center. Contract monitor is Corey Ippolito. The first author would like to thank Sachin Jain for helpful discussions on the multiresolution trajectory optimization algorithm and for his valuable comments.

References

[1] Schwartz, A., *Theory and Implementation of Numerical Methods Based on Runge–Kutta Integration for Solving Optimal Control Problems*, Ph.D. Thesis, Univ. of California, Berkeley, CA, 1996.

[2] Betts, J. T., and Huffman, W. P., "Mesh Refinement in Direct Transcription Methods for Optimal Control," *Optimal Control Applications and Methods*, Vol. 19, No. 1, 1998, pp. 1–21. doi:10.1002/(SICI)1099-1514(199801/02)19:1<1::AID-OCA616>3.0.CO;2-Q

[3] Jain, S., *Multiresolution Strategies for the Numerical Solution of Optimal Control Problems*, Ph.D. Thesis, Georgia Inst. of Technology, Atlanta, GA, March 2008.

[4] Gong, Q., Fahroo, F., and Ross, I. M., "Spectral Algorithm for Pseudospectral Methods in Optimal Control," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 3, 2008, pp. 460–471. doi:10.2514/1.32908

[5] Babuska, I., and Gui, W., "Basic Principles of Feedback and Adaptive Approaches in the Finite Element Method," *Computer Methods in Applied Mechanics and Engineering*, Vol. 55, Nos. 1–2, April 1986, pp. 27–42. doi:10.1016/0045-7825(86)90084-8

[6] Hugger, J., "The Theory of Density Representation of Finite Element Meshes. Examples of Density Operators with Quadrilateral Elements in the Mapped Domain," *Computer Methods in Applied Mechanics and Engineering*, Vol. 109, Nos. 1–2, 1993, pp. 17–39. doi:10.1016/0045-7825(93)90223-K

[7] Beckett, G., Mackenzie, J. A., Ramage, A., and Sloan, D. M., "On the

Numerical Solution of One-Dimensional PDEs Using Adaptive Methods Based on Equidistribution," *Journal of Computational Physics*, Vol. 167, No. 2, 2001, pp. 372–392. doi:10.1006/jcph.2000.6679

[8] Betts, J. T., Campbell, S. L., and Kalla, N. N., "Initialization of Direct Transcription Optimal Control Software," *Proceedings of the 42nd IEEE Conference on Decision and Control*, Maui, HI, Dec. 2003, pp. 3802–3807.

[9] Zhao, Y. J., "Optimal Patterns of Glider Dynamic Soaring," *Optimal Control Applications and Methods*, Vol. 24, No. 2, 2004, pp. 67–89. doi:10.1002/oca.739

[10] Jain, S., and Tsiotras, P., "Trajectory Optimization Using Multiresolution Techniques," *Journal of Guidance, Control, and Dynamics*, Vol. 31, No. 5, 2008, pp. 1424–1436. doi:10.2514/1.32220

[11] Zhao, Y., and Tsiotras, P., "A Density-Function Based Mesh Refinement Algorithm for Solving Optimal Control Problems," AIAA Paper 2009-2019, Infotech at Aerospace Conference, Seattle, WA, 2009.

[12] Gill, P. E., Murray, W., and Saunders, M. A., "SNOPT: An SQP Algorithm for Large-Scale Constrained Optimization," Univ. of California, Numerical Analysis Rept. 97-2, San Diego, CA, 1997.

[13] Bryson, A. E., and Ho, Y., *Applied Optimal Control-Optimization, Estimation and Control*, Hemisphere, Washington, D.C., 1975.

[14] Rao, A. V., and Mease, K. D., "Eigenvector Approximate Dichotomic Basis Method for Solving Hyper-Sensitive Optimal Control Problems," *Optimal Control Applications and Methods*, Vol. 20, No. 2, 1999, pp. 59–77. doi:10.1002/(SICI)1099-1514(199903/04)20:2<59::AID-OCA646>3.0.CO;2-8

[15] Betts, J. T., and Huffman, W. P., "Sparse Optimal Control Software SOCS," The Boeing Company, Mathematics and Engineering Analysis Technical Document mealr-085, Seattle, WA, July 1997.