Second-Order DMOC Using Projection

Kristine L. Snyder
Department of Applied Mathematics
526 UCB
University of Colorado
Boulder, CO 80309
kristine.snyder@colorado.edu

Todd D. Murphey
Department of Mechanical Engineering
Northwestern University
2145 Sheridan Road
Evanston, IL 60208, USA
t-murphey@northwestern.edu

***Abstract*— Discrete mechanics and optimal control (DMOC) is a recent development in optimal control of mechanical systems that takes advantage of the variational structure of mechanics when discretizing the optimal control problem. Typically, the discrete Euler-Lagrange equations are used as constraints on the feasible set of solutions, and then the objective function is minimized using a constrained optimization algorithm, such as sequential quadratic programming (SQP). In contrast, this paper illustrates that by reducing dimensionality by projecting onto the feasible subspace and then performing optimization, one can obtain significant improvements in convergence, going from superlinear to quadratic convergence. Moreover, whereas numerical SQP can run into machine precision problems before terminating, the projection-based technique converges easily. Double and single pendulum examples are used to illustrate the technique.**

I. Introduction and Background

Discrete Mechanics and Optimal Control, abbreviated DMOC, is a method that uses variational integrators to find the optimal control for a given discretized physical system using ideas from Lagrangian and Hamiltonian dynamics [1], [2]. This enables DMOC to be effective for impacts and collisions as well as smooth motion in both linear and non-linear systems. Rather than discretizing a continuous equation, DMOC uses the forced discrete Euler-Lagrange (DEL) equations over multiple time steps as constraints and, within these constraints, minimize a cost functional of weighted discrete control values. It avoids many of the problems that accompany discretizing continuous equations of motion, like inaccurate loss or gain of energy over time.

The ability of DMOC to handle both non-linearity and impacts while retaining the essential characteristics of the underlying physical system has made it useful in a variety of arenas. These include undercontrolled systems, such as a group of hovercraft that have control in only two of their three degrees of freedom [3] and constrained multibody dynamcs [4]. DMOC has also been used as a way to find optimal control in hybrid systems, such as bipedal walking in two dimensions [5].

DMOC's effectiveness, however, depends on the method used to solve the ultimate constrained optimization problem. Traditionally, DMOC has been solved using Sequential Quadratic Programming (SQP) [1], [6]. SQP is a method that combines the cost function and the linearized constraints into an approximation of the Lagrangian function. At each step, it solves a quadratic programming sub-problem based on this approximation, then updates the approximation of the Hessian for the next iteration. Though this method is effective, because it is essentially a 'black box' algorithm for any arbitrary non-linear constrained optimization problem, it typically ignores some of the analytic properties of DMOC that could be used to improve convergence.

We present a method that uses the structure of the equations involved in DMOC to build a projection of arbitrary configuration and control variables onto feasible configuration and control variables. This projection can then be used to reduce the dimensionality of the problem via the constraints, allowing the optimal solution to be found via pure Newton's method without losing any information. We then present preliminary findings on how the projection method compares to both SQP with numerical derivative calculations and gradient descent with Armijo line search. The three methods are compared in terms of accuracy of results and convergence properties for a variety of example problems.

We organize the paper in the following way: Section II provides an explanation of how the continuous system can be modeled using discrete mechanics, while retaining the physical characteristics of the original problem. Section III gives a short background on the structure and use of DMOC. Section IV provides a description of our method, including the derivation of the projection and its relevant derivatives. Section V compares examples of our method to both gradient descent and numerical SQP for two non-linear systems. Lastly, Section VI gives conclusions and future work.

II. Discrete Mechanics

We begin with a continuous mechanical system with trajectory $q(t) \in Q$ (configuration space) starting at some initial point $(q(0), \dot{q}(0))$and ending at $(q(T), \dot{q}(T))$. The system is also subject to some external forcing, with the optimal control determined by minimizing the cost functional

$$J(q,u) = \int_0^T C(q(t), \dot{q}(t), u(t))dt.$$

To preserve the mechanics of the original system, the system is constrained by the Lagrange-D'Alembert principle

$$\delta \int_0^T L(q(t), \dot{q}(t))dt + \int_0^T u(t)\delta q(t)dt = 0$$

for all variations $\delta q(t)$ s.t. $\delta q(0) = \delta q(T) = 0$, where L is the Lagrangian mapping the tangent bundle of Q, TQ, to $\mathbb{R}$.

The above two equations describe a continuous constrained optimization problem; to perform DMOC, we need an approximation of this continuous problem in discrete space. A complete discussion and derivation of this discretization can be found in [2], whereas much of the following shorter derivation is taken from [1] and [3].

To define the discrete problem, we first convert the state space from TQ as $Q \times Q$ by replacing a pair $(q, \dot{(q)})$ with the triplet (q_0, q_1) and time step h (for more details see [1]). We then discretize the continuous path $q : [0, T] \rightarrow Q$ via $q_d : \{0, h, 2h, \ldots, Nh\} \rightarrow Q$, where $N \in \mathbb{N}$ and $T = Nh$, and we approximate $q(kh)$ with $q_k = q_d(kh)$. Similarly, we transform the continuous control $u : [0, T] \rightarrow T^*Q$ (the cotangent bundle) to the discretized values $u_d : \{0, h, 2h, \ldots, Nh\} \rightarrow T^*Q$, where again, $u_k = u_d(kh)$.

We next need the discretized version of the continuous cost functional and Lagrange-D'Alembert principle. We first discretize the Lagrangian L, giving

$$\begin{aligned} L_d(q_k, q_{k+1}) &\approx \int_{kh}^{(k+1)h} L(q(t), \dot{q}(t))dt \\ &\approx hL(\frac{q_{k+1} + q_k}{2}, \frac{q_{k+1} - q_k}{h}), \end{aligned}$$

with the midpoint rule used in the discretization. Similarly, we approximate a discretized version of the control variable:

$$u_k^- \cdot \delta q_k + u_k^+ \cdot \delta q_{k+1} \approx \int_{kh}^{(k+1)h} u(t) \cdot \delta q(t)dt.$$

where u_k^+ and u_k^- represent the right and left hand control for the time step k. Using the discrete Lagrangian, we can define the discrete Lagrange-D'Alembert principle, which restricts to paths $\{q_k\}_{k=0}^N$ such that all variations $\{\delta q_k\}_{k=0}^N$ with $\delta q_0 = \delta q_N = 0$ require that

$$\delta \sum_{k=0}^{n-1} L_d(q_k, q_{k+1}) + \sum_{k=0}^{n-1} u_k^- \cdot \delta q_k + u_k^+ \cdot \delta q_{k+1} = 0.$$

Rewriting, we get the forced discrete Euler-Lagrange (DEL) equations:

$$D_2 L_d(q_{k-1}, q_k) + D_1 L_d(q_k, q_{k+1}) + u_{k-1}^+ + u_k^- = 0. \quad (1)$$

where D_i represents the derivative with respect to the i^{th} argument (i.e. $D_2 L_d(q_{k-1}, q_k)$ is the derivative of L_d with respect to q_k) for $k = 1$ to N. Lastly, for each time step k, the cost functional can be rewritten as

$$C_d(q_k, q_{k+1}, u_k, u_{k+1}) \approx \int_{kh}^{(k+1)h} C(q, \dot{q}, u)dt$$

which, leads to the total cost:

$$J_d(q_d, u_d) = \sum_{k=0}^{N-1} C_d(q_k, q_{k+1}, u_k, u_{k+1}). \quad (2)$$

III. DMOC

The equations for all interior time steps can be described by (1), but we must also ensure these interior points are continuous with the fixed initial and final time steps to meet these boundary conditions. To do so, the discrete Legendre transforms $\mathbb{F}^+$ and $\mathbb{F}^-$ are used to relate the representation in the continuous (TQ) and discrete ($Q \times Q$) domains. These transforms are defined to be

$$\begin{aligned} \mathbb{F}^+ L_d &: (q_{k-1}, q_k) \rightarrow (q_k, p_k) \\ &\quad p_k = D_2 L_d(q_{k-1}, q_k) + u_{k-1}^+ \\ \mathbb{F}^- L_d &: (q_{k-1}, q_k) \rightarrow (q_{k-1}, p_{k-1}) \\ &\quad p_k = -D_1 L_d(q_{k-1}, q_k) - u_{k-1}^-. \end{aligned}$$

Further descriptions of these transforms can be found in [1]. Additionally, the standard Legendre transform can be used to map TQ to T^*Q.

$$\mathbb{F}L : (q, \dot{q}) \rightarrow (q, p) = (q, D_2 L(q, \dot{q}))$$

This gives the following additional constraint equations for the initial ($t = 0$) and final ($t = T$) time steps.

$$\begin{aligned} D_2 L(q(0), \dot{q}(0)) + D_1 L_d(q_0, q_1) + u_0^- &= 0 \quad (3) \\ -D_2 L(q(T), \dot{q}(T)) + D_1 L_d(q_{n-1}, q_n) + u_{n-1}^+ &= 0 \quad (4) \end{aligned}$$

Thus our final system consists of minimizing the discrete cost functional (2) subject to the constraints (1), (3) and (4).

In traditional DMOC, both u_k^- and u_k^+ are calculated via the midpoint rule as $\frac{h}{4}(u_k + u_{k+1})$. However, to make the projection simpler in the next section, we use a slightly different approximation, using the left hand endpoint rather than the midpoint rule. Instead of averaging the two controls, we simply denote both u_k^- and u_k^+ as u_k. Note that this does not significantly change the optimal control, and that midpoint control values can easily be recovered using the previous definition of u_k^-.

We thus have the overall DEL equations

$$\begin{aligned} D_2 L(q(0), \dot{q}(0)) + D_1 L_d(q_0, q_1) + u_0 &= 0 \quad (5) \\ D_2 L_d(q_{k-1}, q_k) + D_1(q_k, q_{k+1}) + u_{k-1} + u_k &= 0 \quad (6) \\ -D_2 L(q(T), \dot{q}(T)) + D_1 L_d(q_{n-1}, q_n) + u_{n-1} &= 0 \quad (7) \end{aligned}$$

IV. Projection-Based Optimization

DMOC is traditionally formulated as a constrained optimization problem. For this reason, the problem is generally framed as trying to find the trajectory and control that minimize the cost functional within a subset of the domain that obeys the constraints

$$\min_{(q,u) \in U \subseteq Q \times T^*Q} J(q, u).$$

In this formulation, the cost is computed while assuring the trajectory and control variables meet the constraints. Thus, though the search stays near the constraint surface, in theory, the search is being done over the entire set of trajectory and control values, not just those subject to the constraints.

It is equivalent to define the unconstrained minimization problem of finding the minimal control over configuration and control variables constrained via a projection $\mathscr{P}$,

$$\min_{x \in B_\epsilon(q,u))} J(\mathscr{P}(q,u)),$$

where x is a combination of possible configuration and control variables and $B_\epsilon(q,u)$ is a neighborhood of (q,u). In practice, this search can be done by projecting a given input of arbitrary configuration and control variables onto the subspace of admissible configurations and controls and composing that projection with the cost. Thus rather than trying to simultaneously minimize J while fulfilling the constraints, only the subspace of admissible configurations and controls is ever examined. We therefore use the projection to reduce the dimensionality of the problem via the constraints before ever attempting to find the minimum.

However, the problem now hinges on the choice of the projection $\mathscr{P}$. $\mathscr{P}$ must be defined so that it takes any arbitrary combination of configuration and control variables and maps it onto one that fulfills all of the constraint equations. Furthermore, in order to optimize over all possible projections, we will also need both first and second derivatives of $\mathscr{P}$, $D\mathscr{P}$ and $D^2\mathscr{P}$, thus requiring $\mathscr{P}$ to be at least C^2 in all its arguments.

A. Projection Definition and Differentiability

The DEL equations (5), (6) and (7) can be used to define the projection $\mathscr{P}$ and its derivatives for an arbitrary DMOC problem. Recall that the initial boundary condition is

$$D_2L(q(0),\dot{q}(0)) + D_1L_d(q_0,q_1) + u_0 = 0.$$

Defining

$$g_0(q(0),\dot{q}(0),q_0,q_1) = D_2L(q(0),\dot{q}(0)) + D_1L_d(q_0,q_1),$$

we can rewrite the DEL equation as

$$g_0(q(0),\dot{q}(0),q_0,q_1) + u_0 = 0.$$

We then solve for u_0 as a function of the boundary conditions $q(0) = q_0$ and $\dot{q}(0)$ and the configuration variable q_1, giving

$$u_0 = -g_0(q(0),\dot{q}(0),q_0,q_1). \tag{8}$$

Any DEL equation for an interior point $k = 1,\ldots,n-1$

$$D_2L_d(q_{k-1},q_k) + D_1L_d(q_k,q_{k+1}) + u_{k-1} + u_k = 0,$$

can also be rewritten as

$$g(q_{k-1},q_k,q_{k+1}) + u_{k-1} + u_k = 0,$$

where

$$g(q_{k-1},q_k,q_{k+1}) = D_2L_d(q_{k-1},q_k) + D_1L_d(q_k,q_{k+1}).$$

Solving for u_k, we have the feedback law

$$u_k = -u_{k-1} - g(q_{k-1},q_k,q_{k+1}).$$

After solving the first equation for u_0, the remaining interior equations can be solved for the control variables in terms of the configuration variables for $k = 1,\ldots n-1$, giving

$$u_k = (-1)^i g_1(q(0),\dot{q}(0),q_0,q_1) + \sum_{j=1}^{k}(-1)^j g(q_{j+1},q_j,q_{j-1}). \tag{9}$$

This defines a projection $\mathscr{P}(q,u) = (q,u(q))$, where each u_i is defined via (8) and (9). u does not appear in the projection because it can be fully defined from the configuration variables, so $u(q)$ does not depend on u.

Whether the projection becomes more complex depends on how the ending condition is met, which can be done in two different ways. One way is to impose a terminal cost by including a term in the cost function enforcing a steep cost penalty for deviating from the final condition. Because this method of prescribing the ending condition requires no further constraints, and because we have thus far used the n constraints to solve for n of the $2n$ total variables in the system, this completes the projection.

Alternatively, imposing the ending condition using the remaining DEL equation gives n constraints and $2n-1$ degrees of freedom, requiring one of the configuration time steps to be solved via the others. This can be done by again using the DEL equations (6) and (7). We first rewrite the DEL equation (7) for the ending condition as

$$g_n(q(T),\dot{q}(T),q_{n-1},q_n) + u_n = 0$$

where

$$\begin{aligned} g_n(q(T),\dot{q}(T),q_{n-1},q_n) &= -D_2L(q(T),\dot{q}(T)) + \\ &\quad D_1L_d(q_{n-1},q_n). \end{aligned}$$

If we denote (5) DEL_0 and (6) DEL_k for $k = 1\ldots n-1$ we can combine them to give

$$\begin{aligned} 0 &= \sum_{i=0}^{n}(-1)^i DEL_i \\ 0 &= g_0(q(0),\dot{q}(0),q_1) + (-1)^{n+1}g_n(q_n,\dot{q}_n,q_{n-1}) + \\ &\quad \sum_{i=1}^{n-1}(-1)^i g(q_{i+1},q_i,q_{i-1}) \end{aligned}$$

which depends only on the configuration variables. This equation can be used to, usually implicitly, solve for either the first or last time step in terms of the configuration variables. Note that because q_n is a boundary condition, our last configuration variable in this case is actually q_{n-1}.

Using these equations, we can construct an appropriate projection $\mathscr{P}(q,u)$. If we include the ending condition in the cost equation, we have a projection of our configuration variables onto themselves, and the control as a function of our configuration variables. Thus, our projection becomes

$$\mathscr{P}(q,u) = (q,u(q)).$$

If the DEL equation is used to impose the ending condition, we get the projection

$$\begin{aligned} \mathscr{P}(q,u) &= (q_1(q_2,\ldots,q_{n-1}),q_2,\ldots,q_{n-1}, \\ &\quad u_1(q_2,\ldots,q_{n-1}),\ldots,u_n(q_2,\ldots,q_{n-1})). \end{aligned}$$

Lemma 1: $\mathscr{P}$ is a projection, and $L_d, L \in C^n \rightarrow \mathscr{P} \in C^{n-1}$

Proof: For the case of terminal cost,

$$\mathscr{P}(\mathscr{P}(q,u)) = \mathscr{P}(q,u(q)) = (q,u(q)) = \mathscr{P}(q,u).$$

If the ending condition is met by including another constraint equation

$$\begin{aligned}\mathscr{P}(\mathscr{P}(q,u)) &= \mathscr{P}(q_1(q_2,\ldots,q_{n-1}),q_2,\ldots,q_{n-1},\\ &\quad u_1(q_2,\ldots,q_{n-1}),\ldots,u_n(q_2,\ldots,q_{n-1}))\\ &= (q_1(q_2,\ldots,q_{n-1}),q_2,\ldots,q_{n-1},\\ &\quad u_1(q_2,\ldots,q_{n-1}),\ldots,u_n(q_2,\ldots,q_{n-1}))\\ &= \mathscr{P}(q,u).\end{aligned}$$

Furthermore, because $\mathscr{P}$ depends solely upon the DEL equations, if $L_d, L \in C^n$, since $\mathscr{P}$ consists of elements which are linear combinations of DL_d and DL, then we must have $\mathscr{P} \in C^{n-1}$. ■

The necessary derivatives can be found using

$$DJ_c = DJ(\mathscr{P}(q)) \cdot D\mathscr{P}(q),$$

where J denotes the unconstrained cost function and J_c the cost function constrained via the projection. Similarly, the Hessian can be calculated via

$$\begin{aligned}D^2J_c &= D^2J(\mathscr{P}(q)) \cdot D\mathscr{P}(q) +\\ &\quad DJ(\mathscr{P}(q)) \cdot D^2\mathscr{P}(q)\end{aligned}$$

where the elements of $D\mathscr{P}$ and $D^2\mathscr{P}$ can be found by taking the first and second derivatives the projection elements as defined via the DEL equations.

We then use Newton's method to find the minimum cost of the projected system.

V. Examples

A. Incremental Cost

For the example of the double pendulum, the projection method was compared to two other methods, numerical SQP and gradient descent using Armijo line search. For all tests, SQP was implemented using MATLAB's fmincon function using numerical derivatives with constraint tolerance set to 10^{-14}. For this initial case, we used a cost function of

$$J(q_i,u_i) = \sum_{i=0}^{n-1} u_i^T u_i \tag{10}$$

where u_i was defined in (8) and (9).

For our simulations, θ_1 represents the angle with respect to the vertical at the base joint and θ_2 represents the angle with respect to the vertical at the outer joint (Figure 1). The Lagrangian of the double pendulum is

$$\begin{aligned}L &= \frac{1}{2}(m_1+m_2)l_1^2\dot{\theta}_1^2 + \frac{1}{2}m_2l_2^2\dot{\theta}_2^2\\ &\quad +m_2l_1l_2\dot{\theta}_1\dot{\theta}_2\cos(\theta_1-\theta_2) +\\ &\quad (m_1+m_2)gl_1\cos(\theta_1) + m_2gl_2\cos(\theta_2)\end{aligned}$$

where m_1 and m_2 are the masses for the respective joints, and l_1 and l_2 are the lengths of the segments from the base joint to the outer joint and the outer joint to the end of the pendulum, respectively (Figure 1).

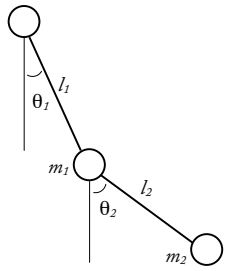

Fig. 1. A depiction of the double pendulum system, including definitions of the variables θ_1 and θ_2 and the parameters m_1, m_2, l_1 and l_2.

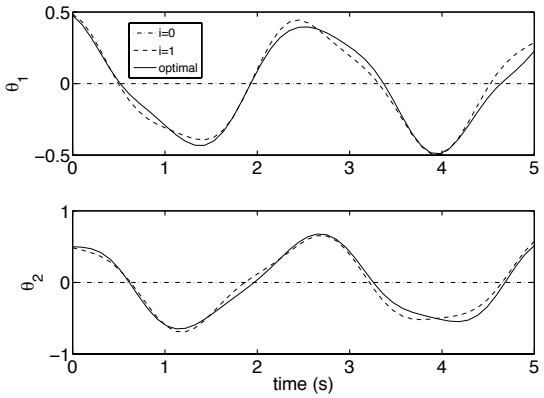

Fig. 2. A comparison of the first two iterates of the projection method with the cost function $J(q_i,u_i) = \sum_{i=0}^{n-1} u_i^T u_i$ to the unforced simulation trajectory with initial conditions $\theta_1 = \theta_2 = \frac{1}{2}$.

We used the forced DEL equations and the cost function (10) to simulate the unforced system. Because the solution was known, it allowed us to test convergence characteristics, such as the number of iterations to reach the optimum, order of convergence, and accuracy of solution. The parameter values were $m_1 = m_2 = l_1 = l_2 = 1$. For the first test, initial angle values were $\theta_{1,0} = \frac{1}{2}$ and $\theta_{2,0} = \frac{1}{2}$, with zero initial velocities. The time step h was set to 0.1 over a total of 50 time steps, or 5 seconds.

Projection with Newton's method converged to the optimal (unforced) solution within 7 iterations, whereas it took numerical SQP 58 iterations to converge. Furthermore, the projection method approximated the exact solution very closely after just one iteration (Figure 2), whereas numerical SQP took significantly longer to do so. Projection with Newton's method reduces the error in the trajectory, as measured via the 2-norm, by 85% within one iteration (Figure 2), whereas numerical SQP reduces the error by only 10% (Figure 3).

For the second test, initial angle values were $\theta_{1,0} = \frac{\pi}{2}$ and $\theta_{2,0} = \frac{\pi}{2}$, with zero initial velocities. The time step h was set to 0.1 over a total of 50 time steps, or 5 seconds. All methods were given the same input, consisting of the optimal trajectory perturbed by value of $\epsilon = .001$ with control taking on an initial value of 10 for all time steps.

As can be seen in Figure 4, numerical SQP converged to a different solution than did the projection method because

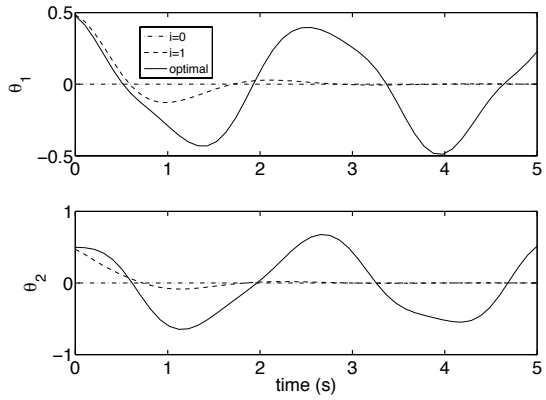

Fig. 3. A comparison the first two iterates of numerical SQP with the cost function $J(q_i, u_i) = \sum_{i=0}^{n-1} u_i^T u_i$ to the unforced simulation trajectory with initial conditions $\theta_1 = \theta_2 = \frac{1}{2}$.

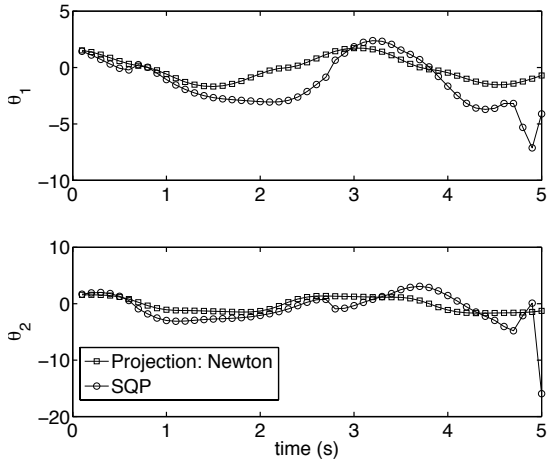

Fig. 4. Solutions using numerical SQP and Newton's method with a cost of $J(q_i, u_i) = \sum_i^n u_i^T u_i$. This DMOC problem yields a simulation of the system if it converges, but neither gradient descent (with ran for 1000 iterations without terminating) nor numerical SQP (which ran for 428 iterations to arrive at a non-optimal solution) implementation are able to converge. Newton's method converges without difficulty in 6 iterations to the actual optimal (unforced) solution.

it was compromised by the control values. Furthermore, it took only 6 iterations for the projection method to converge to the correct trajectory and 428 iterations for numerical SQP to converge to a less than optimal trajectory (Figure 5). Gradient descent was allowed to run for 10^3 iterations, at which point it had still not yet converged, and is therefore not shown. Note that if given a more accurate controls trajectory, numerical SQP will converge to the optimal solution, but that it requires both trajectory and control variables to be reasonably correct to converge to the optimal value.

Figures 4 and 5 show a comparison of the 3 methods. As can be seen in Figure 5, in addition to converging to a less than optimal solution, numerical SQP converges more slowly. Projection with Newton's method converges quadratically, whereas numerical SQP converges, at best, superlinearly, and gradient descent converges only linearly. While it is likely that SQP's convergence could be improved if it were given more derivative information, it likely would not reach the

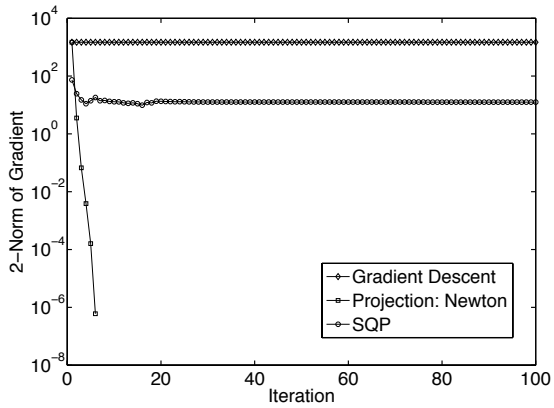

Fig. 5. Log of the 2-norm of the gradient versus iteration for gradient descent, numerical SQP, and Newton's method with a cost of $J(q_i, u_i) = \sum_i^n u_i^T u_i$ for the first hundred iterations. Gradient descent displays linear convergence, numerical SQP displays superlinear convergence, and Newton's method displays quadratic convergence, but neither gradient descent nor SQP has a high likelihood of converging to the correct optimal trajectory with a reasonable number of iterations.

convergence speed of projection with Newton due the the linearization of the constraints in SQP.

Due to its effectiveness at finding unforced trajectory with an unconstrained endpoint, it may be possible to use this method to solve the unforced DEL equations when they are degenerate. Initial results from simpler systems, such as the spring and the pendulum, show that, for short time horizons, projection with Newton's method is faster than root-finding.

B. Terminal Cost

To impose an endpoint on the system, the cost function was adjusted to include a steep penalty for deviating from the final condition. For an arbitrary system, this can be written

$$J_T = \sum_{i=0}^{n-1} u_i^T u_i + C_T(q_n - q_{end})^T(q_n - q_{end}),$$

where q_{end} indicates the required ending condition in configuration space and C_T is a coefficient representing the relative cost of not reaching the endpoint. For the double pendulum, this gives rise to the cost function

$$\begin{aligned} J_T \quad = \quad & \sum_{i=0}^{n-1} u_i^T u_i + C_T[(\theta_{1,n} - \theta_{1,end})^2 + \\ & (\theta_{2,n} - \theta_{2,end})^2 + \left(\frac{(\theta_{1,n} - \theta_{1,n-1})}{h} - \dot{\theta}_{1,end}\right)^2 + \\ & \left(\frac{(\theta_{2,n} - \theta_{2,n-1})}{h} - \dot{\theta}_{2,end}\right)^2] \end{aligned}$$

with $C_T = 1000$. C_T was set to be three orders of magnitude larger than the typical control values to assure the ending condition was met with reasonable accuracy. Increasing C_T beyond this value did not significantly affect results.

To test the effectiveness of convergence with terminal cost, we first compared gradient descent, numerical SQP, and the projection method with the initial condition $\theta_1 = \theta_2 = 0$, and the final condition $\theta_1 = \theta_2 = 0.7$. The time step value

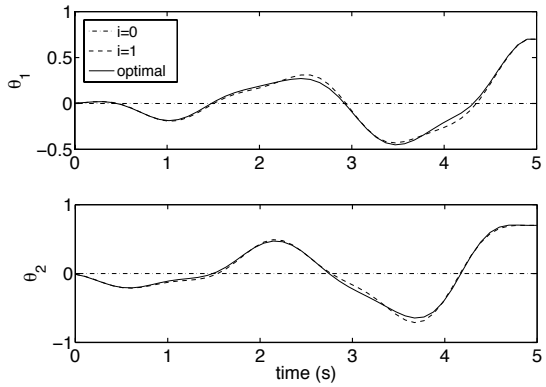

Fig. 6. A comparison of the first two iterates of the projection method via Newton with the cost function $J_T(q_i, u_i) = \sum_{i=0}^{n-1} u_i^T u_i + C_T(q_n - q_{end})^T(q_n - q_{end})$ to the ultimate optimal simulation trajectory with initial conditions $\theta_1 = \theta_2 = 0$ and ending conditions $\theta_1 = \theta_2 = 0.7$.

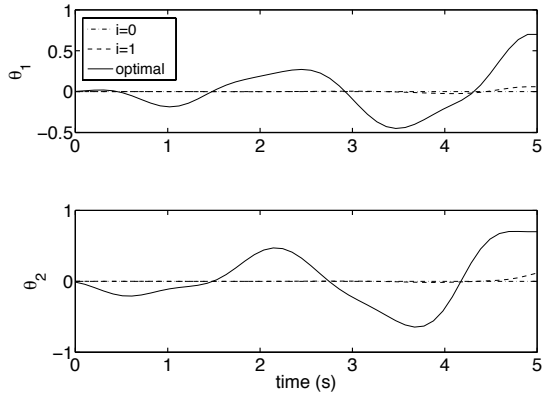

Fig. 7. A comparison of the first two iterates of numerical SQP with the cost function $J_T(q_i, u_i) = \sum_{i=0}^{n-1} u_i^T u_i + C_T(q_n - q_{end})^T(q_n - q_{end})$ to the ultimate optimal simulation trajectory with initial conditions $\theta_1 = \theta_2 = 0$ and ending conditions $\theta_1 = \theta_2 = 0.7$.

was set to $h = 0.1$ for 50 time steps, and all methods were given an input of initial trajectory and control of all zeros.

Projection with Newton's method converged to the optimal solution in 7 iterations, whereas numerical SQP took 47 iterations. However, tracking the trajectories for each iteration shows further advantages of using Newton's method. In one iteration, the difference between the optimal and estimated trajectories, measured via the 2-norm, is reduced by 92.5% (Figure 6), whereas numerical SQP reduces this value by only 7.5% (Figure 7). Past one iteration, the trajectories given by the projection method obscure the optimal trajectory.

In another test, we used the same three methods, gradient descent, numerical SQP and projection with Newton's method to invert the double pendulum, an unstable process. Both initial θ and $\dot{\theta}$ values were set to 0, with final θ values were set to π and $\dot{\theta}$ values again being 0. The time step was set to $h = 0.1$ for 40 time steps or 4 seconds.

All three methods were again given an input of an optimal trajectory perturbed by ϵ=0.001. In this case however, rather than giving zero control, the control was also perturbed by 0.001. Both SQP and Newton's method via projection converged to the same optimal trajectory (Figure 8) with

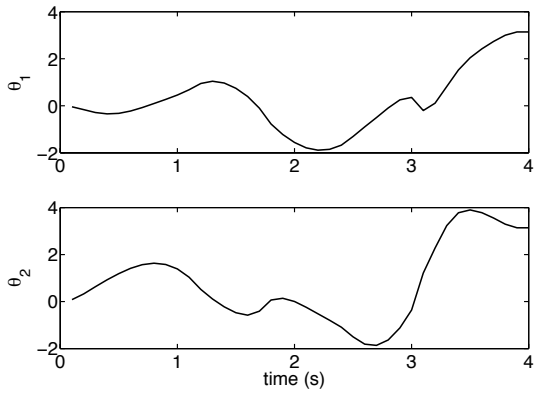

Fig. 8. Optimal trajectory given by numerical SQP and projection with Newton's Method using the cost function $J_T \sum_{i=0}^{n-1} u_i^T u_i + C_T(q_n - q_{end})^T(q_n - q_{end})$. The trajectory displays a pumping trajectory used to efficiently invert the double pendulum, taking it from initial angles of $\theta_1 = \theta_2 = 0$ to ending conditions $\theta_1 = \theta_2 = \pi$

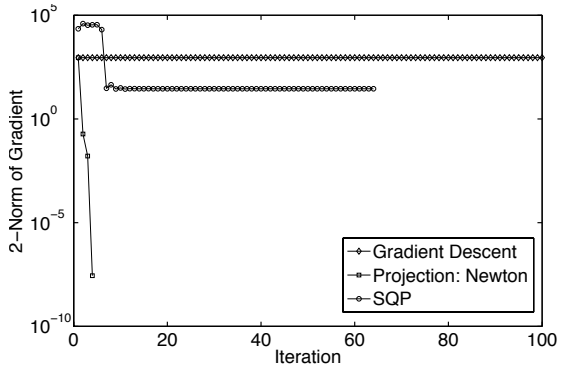

Fig. 9. Log of the 2-norm of the gradient versus iteration for gradient descent, numerical SQP, and projection with Newton's method for the first hundred iterations with a cost of $J_T \sum_{i=0}^{n-1} u_i^T u_i + C_T(q_n - q_{end})^T(q_n - q_{end})$. Both numerical SQP and projection with Newton's method converged to the optimal trajectory in 4 and 64 iterations, respectively, but gradient descent did not converge within 1000 iterations. Gradient descent exhibits linear convergence, numerical SQP superlinear convergence, and projection with Newton's method quadratic convergence.

Newton converging in 4 iterations and numerical SQP taking 64. Gradient descent again failed to converge within 1000 iterations. One issue with gradient descent for this problem is that the step size for a given gradient direction is quite small, on the order of $10^{-4} - 10^{-5}$, slowing convergence.

A performance comparison of the three methods is shown in Figures 8 and 9. Clearly, projection converges much faster than numerical SQP or gradient descent. Furthermore, because projection depends only on the trajectory values, with the controls calculated as part of the projection, it can be more robust to perturbation. Specifically, if the trajectory is perturbed by as little as .001, and the controls entered as zero, projection will converge to the optimal trajectory, but numerical SQP will converge to a non-optimal solution.

C. Constrained Endpoint

Traditional DMOC uses a constrained endpoint, which creates the need to implicitly solve for the first free time

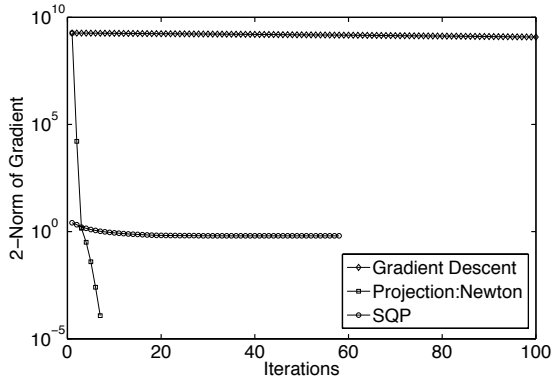

Fig. 10. Log of the 2-norm of the gradient of the cost using gradient descent, numerical SQP, and projection with Newton's method for the single pendulum with an ending condition of $\frac{\pi}{2}$ with a time step of h=0.1. Numerical SQP converges at 58 iterations, and projection with Newton converges after only 7. Gradient descent exhibits linear convergence, numerical SQP superlinear convergence, and projection with Newton's method quadratic convergence.

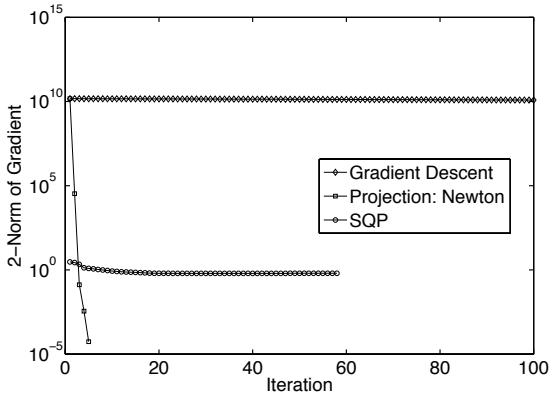

Fig. 11. Log of the 2-norm of the gradient of the cost using gradient descent, numerical SQP, and projection with Newton's Method for the single pendulum with an ending condition $\frac{\pi}{2}$ and a time step of h=0.05. Numerical SQP converges at 58 iterations, and projection with Newton converges at only 5. Gradient descent exhibits linear convergence, numerical SQP superlinear convergence, and projection with Newton's method quadratic convergence.

step of the configuration variable in terms of the remaining configuration variables. This implicit differentiation can significantly complicate calculating the Hessian and sometimes lead to ill-conditioned matrices.

For a simple system, such as the single pendulum, the first configuration variable can be explicitly solved for, due to cancellation of terms in the DEL equations, which are

$$\begin{aligned} 0 &= u_1 + l\dot{\theta}_0 - \frac{l}{h}(\theta_1 - \theta_0) - \frac{gh}{2}\sin\left(\frac{(\theta_0+\theta_1)}{2}\right) \\ 0 &= u_{i-1} + u_i - \frac{l}{h}(\theta_i - 2\theta_{i-1} + \theta_{i-2}) - \\ &\quad \frac{gh}{2}\left(\sin(\frac{(\theta_i+\theta_{i-1})}{2}) + \sin(\frac{(\theta_{i-1}+\theta_{i-2})}{2})\right) \\ 0 &= u_n + l\dot{\theta}_n + \frac{l}{h}(\theta_n - \theta_{n-1}) - \frac{gh}{2}\sin\left(\frac{(\theta_n+\theta_{n-1})}{2}\right). \end{aligned}$$

where we again have the cost functional

$$J(q_i, u_i) = \sum_{i=0}^{n-1} u_i^T u_i.$$

A simulation was performed on this system for $l = 1$. Time steps of both $h = 0.1$ and $h = 0.05$ were used to test the effectiveness of projection as compared to numerical SQP with initial condition 0, ending condition $\frac{\pi}{2}$ for 101 time steps. An initial input of zeros for both control and trajectory was given for both size time steps. Both methods converge to a gradually pumping trajectory, but as can be seen in Figures 10 and 11, projection with Newton's method significantly outperforms both numerical SQP and gradient descent.

VI. Conclusions and Future Work

This paper presents an approach to DMOC that provides local quadratic convergence that in practice is considerably faster than "black box" optimization techniques that utilize numerical differentiation approximations However, the current formulation is local, assumes full actuation and compares to SQP with numerical derivatives. Further work will focus on using implicit differentiation to differentiate the projection associated with constrained endpoints, which will allow for the more accurate simulation of an ending condition. Furthermore, all the systems discussed here have been fully actuated, whereas one of the advantages of DMOC is its ability to stably handle underactuated systems, so a next step will be adjusting the projection to be able to work with these types of systems. Because DMOC also is capable of simulating collisions and impacts within a trajectory, another goal would be to adjust this algorithm to allow it to handle systems with these characteristics. Lastly, to fully gauge the advantages of this method, we would need to input derivatives into SQP rather than calculating numerically calculating them. It is likely that initially using SQP to allow for a large basin of attraction and then projection when near the optimum could considerably improve convergence time while retaining a large basin of attraction, making a much more effective constrained optimization algorithm.

References

[1] J. Marsden and M. West, "Discrete mechanics and variational integrators," *Acta Numerica*, vol. 10, pp. 357–514, 2003.

[2] S. Ober-Bloebaum, O. Junge, and J. Marsden, "Discrete Mechanics and Optimal Control: an Analysis," *Arxiv preprint arXiv:0810.1386*, 2008.

[3] O. Junge, J. Marsden, and S. Ober-Blőbaum, "Discrete mechanics and optimal control," in *Proceedings of the 16th IFAC World Congress, Prague, Czech Republic*, 2005.

[4] S. Leyendecker, S. Ober-Blőbaum, J. Marsden, and M. Ortiz, "Discrete mechanics and optimal control for constrained multibody dynamics," in *Proceedings of the 6th International Conference on Multibody Systems, Nonlinear Dynamics, and Control, ASME International Design Engineering Technical Conferences, Las Vegas, Nevada*, 2007, pp. 4–7.

[5] D. Pekarek, A. Ames, and J. Marsden, "Discrete mechanics and optimal control applied to the compass gait biped," in *IEEE Conference on Decision and Control and European Control Conference, New Orleans, LA, USA*, 2007.

[6] D. Pekarek and J. Marsden, "Variational Collision Integrators and Optimal Control," in *Proceedings of the 18th International Symposium on Mathematical Theory of Networks and Systems, Blacksburg, VA*, 2008.