Optimal Nonlinear Control Using Hamilton-Jacobi-Bellman Viscosity Solutions on Quasi-Monte Carlo Grids

Christian M. Chilan and Bruce A. Conway
Department of Aerospace Engineering
University of Illinois at Urbana-Champaign
April 2020

Optimal Nonlinear Feedback Control

This presentation is based on the article:

Chilan, C., Conway, B.: Optimal nonlinear control using Hamilton–Jacobi–Bellman viscosity solutions on unstructured grids. J. Guid. Control Dyn. **43**, 30–38 (2020)

Feedback Control of Dynamical System

There are many choices available for feedback control including

- PID control
- Lyapunov control
- Sliding mode control
- Model predictive control

However, none of these are *optimal* feedback controllers, i.e. they do not cause the system trajectory to minimize/maximize some quantity of interest.

Feasible controllers follow references. *Proper* optimal controllers follow extremal fields.

Optimal Nonlinear Feedback Control

There are three approaches for optimal nonlinear feedback control:

I. Find the open-loop optimal trajectory and control; derive the neighboring optimal feedback controller (NOC).

ii. Kriging-based extremal field method (recent)

iii. Solve the Hamilton-Jacobi-Bellman equation for the value (cost) function. Then minimize the associated Hamiltonian w.r.t. control $\boldsymbol{u}$ to find optimal $\boldsymbol{u}^*$ (at current $\boldsymbol{x}$). Dynamic Programming.

The Optimal Control Problem

System: $\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t), \qquad \boldsymbol{x}(0)$ given

$\boldsymbol{\psi}(\boldsymbol{x}(T), T) = \boldsymbol{0}, \qquad q$ constraint eqns.

Problem: Find control $\boldsymbol{u}(t)$ to minimize $J = \phi(\boldsymbol{x}(T), T) + \int_0^T L(\boldsymbol{x}, \boldsymbol{u}, t)\, dt$

Calculus of Variations necessary conditions: (Euler-Lagrange equations),

$$H = L + \boldsymbol{\lambda}^T \boldsymbol{f} \qquad \text{Hamiltonian}$$

$$\dot{\boldsymbol{\lambda}} = -\left(\frac{\partial H}{\partial \boldsymbol{x}}\right)^T \qquad \boldsymbol{\lambda}(T) = \left[\frac{\partial \phi}{\partial \boldsymbol{x}} + \boldsymbol{\nu}^T \frac{\partial \boldsymbol{\psi}}{\partial \boldsymbol{x}}\right]_{t=T}^T \qquad \text{Co-state ODEs}$$

$$\boldsymbol{u}^* = \underset{\boldsymbol{u} \in U}{\operatorname{argmin}}\, H(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) \qquad \text{Pontryagin}$$

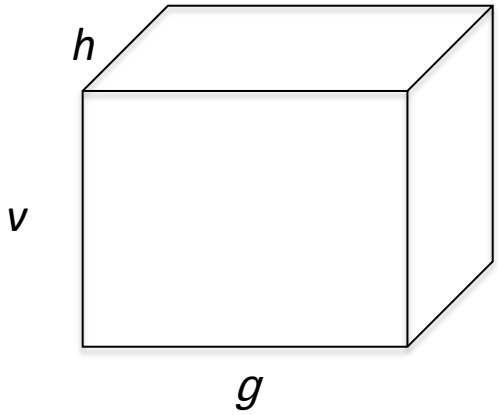
Extremal Field Method

The Extremal Field method consists on generating a set of optimal trajectories with given initial and terminal conditions. If the HJB value function is differentiable, this approach implements the *method of characteristics* (Bryson and Ho, 1975).

There is only one optimal trajectory starting at the nominal initial conditions (IC). However, it is assumed that there is an uncertainty box about the IC, which is sampled to define a number of perturbed IC to generate the extremal field.

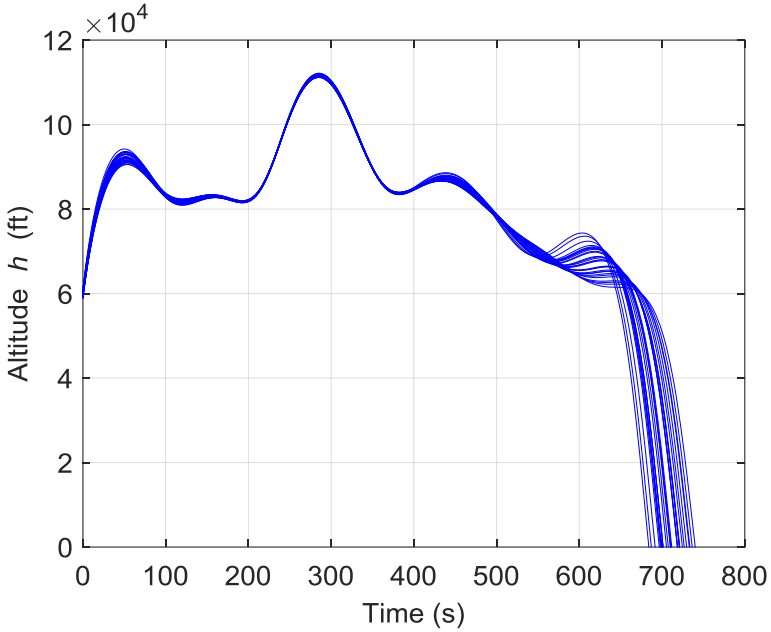
For a flight vehicle,

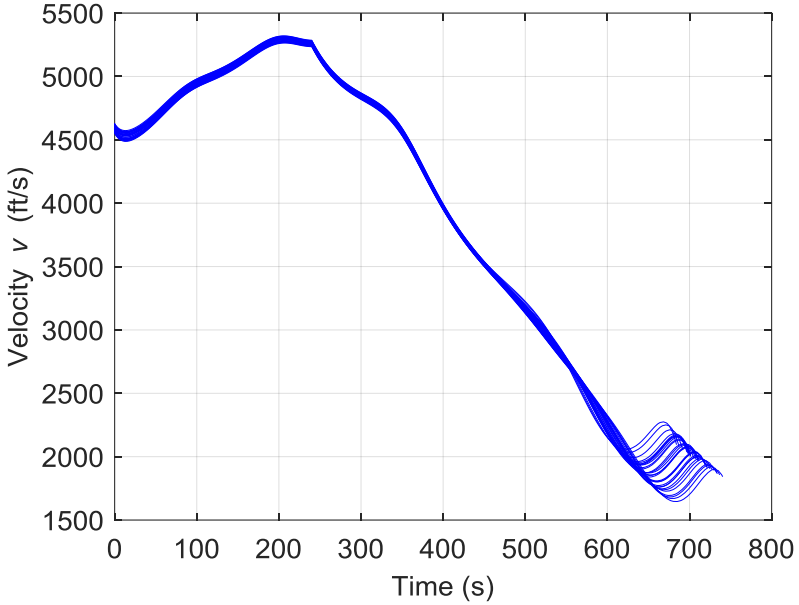
$h_0 = 60{,}000 \pm 1{,}200$ ft
$\phi_0 = 0 \pm 5\text{x}10^{-3}$ deg
$v_0 = 4{,}589.2 \pm 40$ ft/s
$\gamma_0 = 20 \pm 2$ deg
$\psi_0 = 0 \pm 2$ deg
$\theta_0 = 0 \pm 5\text{x}10^{-3}$ deg
$m_0 = 46.746 \pm 0$ slug



Extremal Field States (Input)

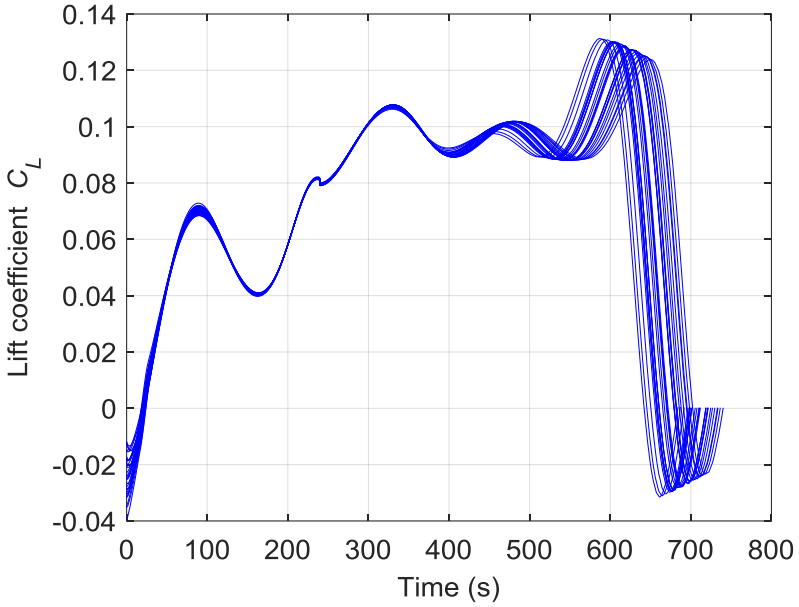
Since each extremal is computed as a point-wise trajectory, the extremal field yields a point cloud, where at each state point the optimal control is known.

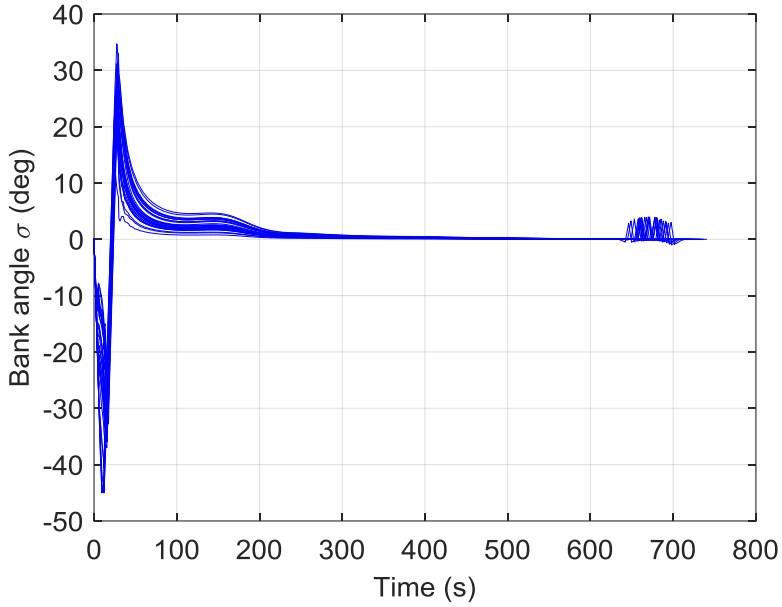




Extremal Field Controls (Output)

For an arbitrary state inside the convex hull defined by the point cloud, the optimal nonlinear feedback control can be determined via interpolation.





Extremal Field Interpolation

The optimal feedback control can be determined by interpolating the extremal field optimal control at the current state. Extremal field interpolation can be implemented using linear interpolation (Jardin and Bryson, 2012) or neural networks (Edwards and Goh, 1995).

Gosh and Conway (2013) proposed the use of kriging (stochastic process regression) for this purpose. Kriging (Krige, 1951) was developed to obtain the best prediction of gold concentrations at arbitrary locations in a field based on actual concentration data from sampled sites.

Kriging is a spatial statistical predictor comprised of a collection of linear regression techniques that produces the least mean-squared prediction error.

Extremal Field Kriging

Mathematical theory developed at *L'École des Mines*, Fontainebleau, France by Georges Mathéron in 1960s.

Traditional applications: Earth Sciences (agriculture, fisheries, hydrology, meteorology, petroleum, remote sensing, etc.).

Newer application areas: DACE (Design and Analysis of Computer Experiments, Sacks et al. 1989), Machine Learning (Rasmussen and Williams, 2006).

Prior to work of Gosh and Conway (2013), no known application in dynamical systems and control theory. In this work, kriging is implemented using the MATLAB DACE Toolbox.

Extremal Field Kriging

A kriging model is composed of two terms: the first is a (polynomial) regression function *F*(.) that captures the global trend of the unknown function. The second term *z*(.) is a multivariate white-noise process that accounts for local residuals so that the model interpolates the sampled observations,

$$y(\boldsymbol{x}) = \mathcal{F}(\boldsymbol{x}, \boldsymbol{\beta}) + z(\boldsymbol{x})$$

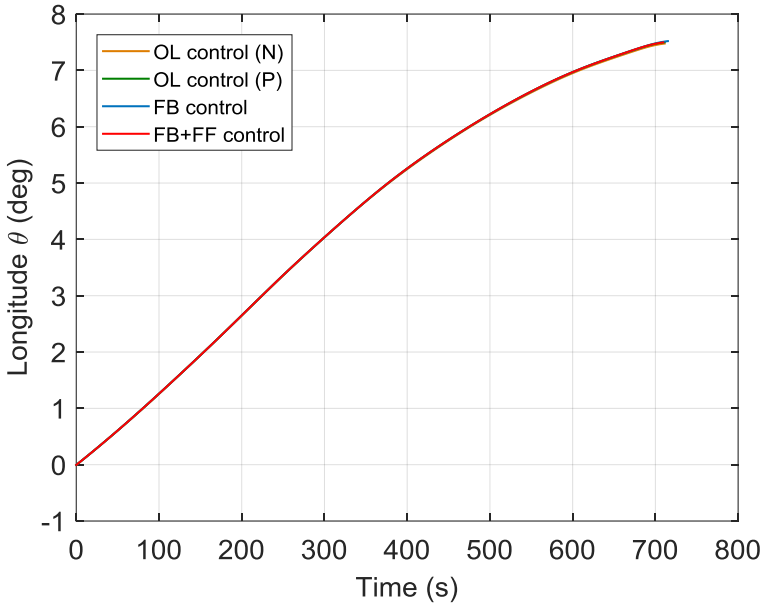
where **β** is an unknown coefficient vector for the regression functions. There are several models available for deterministic regression and stochastic correlation. In this work, we use quadratic regression polynomials for *F*(.) and generalized exponential correlation functions for *z*(.) because of their accuracy and generality.

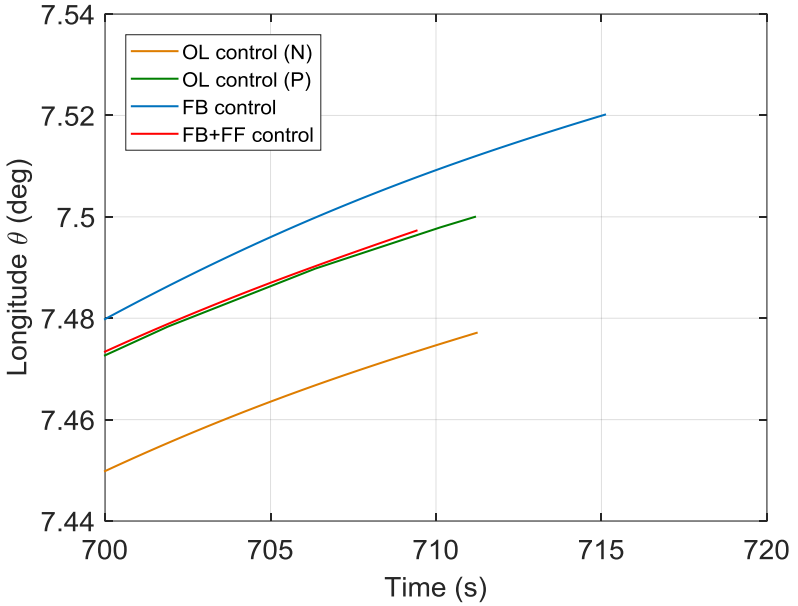
If the system dimensionality were three, the trend model *F*(.) would be the following,

$$\begin{aligned}\mathcal{F}(\boldsymbol{x}, \boldsymbol{\beta}) = {} & \beta_{000} + \beta_{100}x_1 + \beta_{010}x_2 + \beta_{001}x_3 + \beta_{110}x_1x_2 + \beta_{101}x_1x_3 \\ & + \beta_{011}x_3x_2 + \beta_{200}x_1^2 + \beta_{020}x_2^2 + \beta_{002}x_3^2\end{aligned}$$

Solution Trajectories

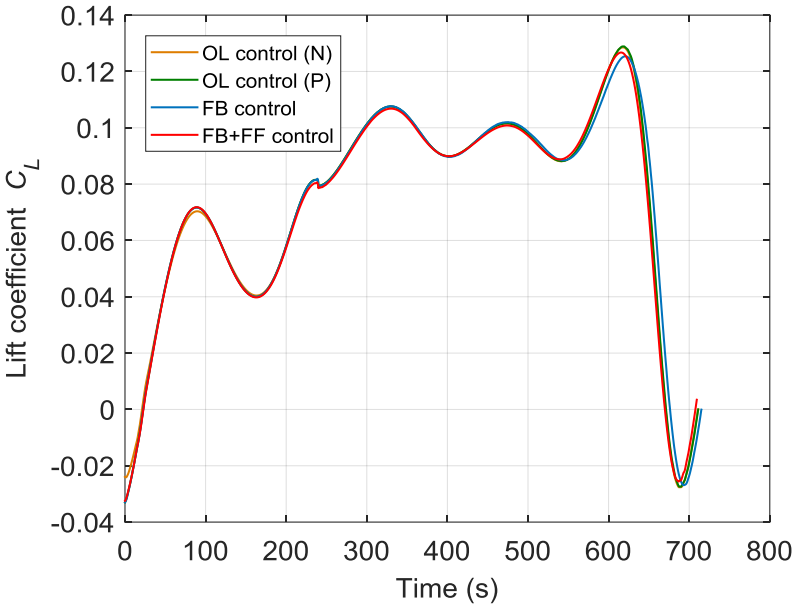
- The OL control trajectory with nominal info. deviates significantly from the terminal conditions.
- FB control does account for perturbed initial conditions. Wind disturbances are managed implicitly through feedback.
- FB+FF control takes into account the perturbed initial conditions and wind disturbances.

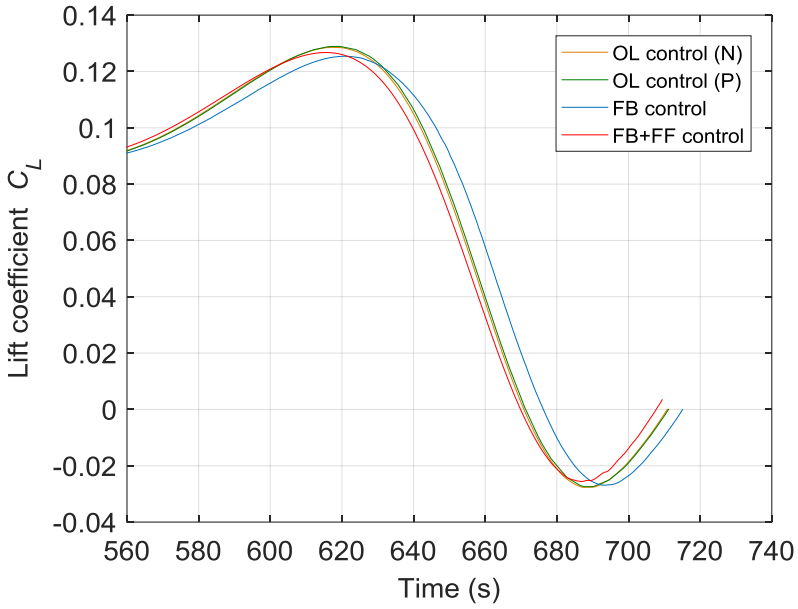




Control Histories

- Since the wind components are beneficial and the final time is free, the FB+FF controller allows approximating the terminal conditions earlier than the FB controller.





NOC and Extremal Field Method

NOC and Extremal Field are effective methods for optimal nonlinear feedback control. However, the methods have disadvantages and limitations.

NOC implementation is complex and requires significant knowledge of optimal control. The NOC controller is only valid in the state space corresponding to the linearized system about the nominal optimal trajectory.

Extremal Field method implementation is simpler. The size of the state space of application is only limited by the computing resources available. The optimal control synthesis is in the order of milliseconds. The generation of the extremal field and the kriging model can take several hours.

NOC and the Extremal Field method can only implement locally optimal feedback controllers because they are based on the Pontryagin Minimum Principle (PMP).

HJB Equation

The "value function" *J* depends only on the initial condition, i.e.

$$J = \min J(\boldsymbol{x}_0) = \min \left[\phi(\boldsymbol{x}(T), T) + \int_0^T L(\boldsymbol{x}, \boldsymbol{u}, t)\, dt \right]$$

The HJB equation (for the Bolza problem) can be written in the following form:

$$-\frac{\partial J}{\partial t} = \min_{\boldsymbol{u} \in U} \left[L(\boldsymbol{x}, \boldsymbol{u}, t) + \frac{\partial J}{\partial \boldsymbol{x}} \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}, t) \right] = H^o$$

subject to the terminal condition: $J(\boldsymbol{x}(T), T) = \phi(\boldsymbol{x}(T), T)$

For every initial condition x_0 in the state domain, the resulting value function $J(x_0)$ is the minimum cost-to-go, so x_0 can be replaced by *x*. $J(x)$ is the solution of the HJB equation.

HJB Equation

When solved over the state domain of interest, the HJB equation is a necessary and sufficient condition for an optimal trajectory. It also provides the *global* minimum proceeding forward from initial condition x_0.

The optimal control can be found using the Pontryagin Minimum Principle:

$$\boldsymbol{u}^* = \underset{\boldsymbol{u}\in U}{\operatorname{argmin}}\, H\left(\boldsymbol{x}, \frac{\partial J^0}{\partial \boldsymbol{x}}, \boldsymbol{u}\right)$$

The HJB (PDE) equation is customarily solved numerically over a grid in the *d*-dimensional state space with some type of interpolation used to evaluate *J* at off-grid points. Usually one works backward from the known value of $J(\boldsymbol{x},T)$.

"Curse of dimensionality" → only modest problems are tractable.

Challenges

As desirable as a globally optimal nonlinear feedback controller would be for many applications, the HJB equation approach has been limited to systems with $d \leq 4$ states and a correspondingly number of controls.

The reasons are principally:

i) The "curse of dimensionality", i.e. number of grid nodes increases exponentially with d.

ii) Interpolation is necessary to evaluate the solution function v_h at $\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$ but simple schemes can be inaccurate and require higher grid resolution.

iii) Conventional approaches to minimizing

$$H_h(v_h(\boldsymbol{x}),\boldsymbol{x}) = \min_{\boldsymbol{u}\in U}\{v_h(\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})) + hL(\boldsymbol{x},\boldsymbol{u})(1-v_h(\boldsymbol{x}))\}$$

can be time consuming and inaccurate.

Our Contribution

i) The "curse of dimensionality", i.e. number of nodes in a uniform rectangular grid increases exponentially with the system dimensionality.

The application of quasi-Monte Carlo grids may reduce the space complexity of the HJB integration problem from exponential to polynomial.

ii) Interpolation is necessary to evaluate the solution function v_h at $\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$ but simple schemes can yield inaccurate results.

Introduce universal kriging for accurate interpolation of the solution function

iii) Conventional approaches to minimizing

$$H_h(v_h(\boldsymbol{x}),\boldsymbol{x}) = \min_{\boldsymbol{u}\in U}\{v_h(\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})) + hL(\boldsymbol{x},\boldsymbol{u})(1-v_h(\boldsymbol{x}))\}$$

can be time consuming and inaccurate

Use nonlinear programming (NLP)

iv) Solution uses fixed point iterations on the grid. At a particular iteration, evaluation of a grid node is independent from evaluation at other nodes.

Use parallel computing for faster computation.

HJB Integration Problem

The “curse of dimensionality” has generally been understood to imply that the HJB integration problem has *exponential* space complexity with respect to the number of dimensions of the system.

However, this is only true if uniform rectangular grids are used to integrate the HJB equation.

Recent methods can solve high-dimensional HJB problems with quadrature-based sparse grids of small size. However, the methods assume that the value function is differentiable (Kang and Wilcox, 2017) or that the system is affine (Adurthi et al., 2017).

HJB Integration Problem

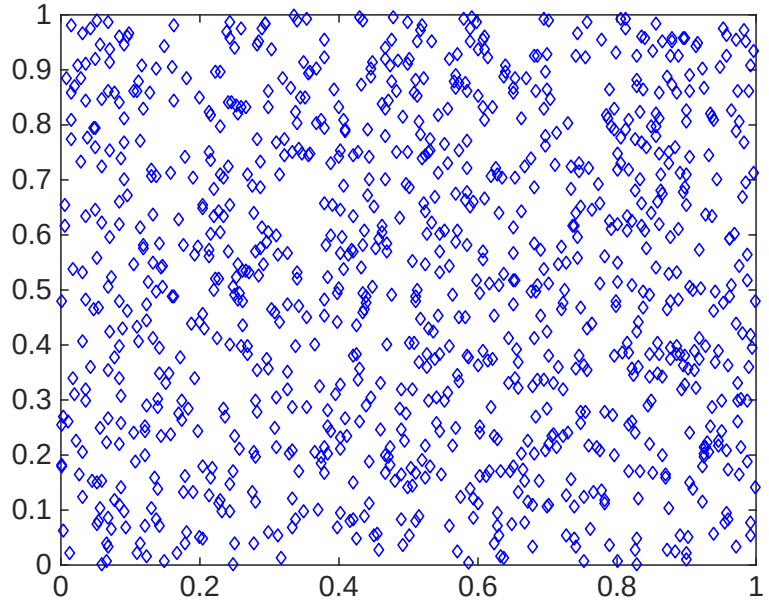
Historically, Monte Carlo (MC) grids have been used in high dimensional problems. They are simpler than quadrature-based node placement and can ameliorate the exponential complexity at the expense of accuracy and convergence rate.

Recently, quasi-Monte Carlo (QMC) sequences have been developed that are superior than MC sequences in the sense that they have low discrepancy. This allows QMC grids to achieve similar accuracy than MC with fewer grid nodes, which reduces computation time (Morokoff and Caflisch, 1995).

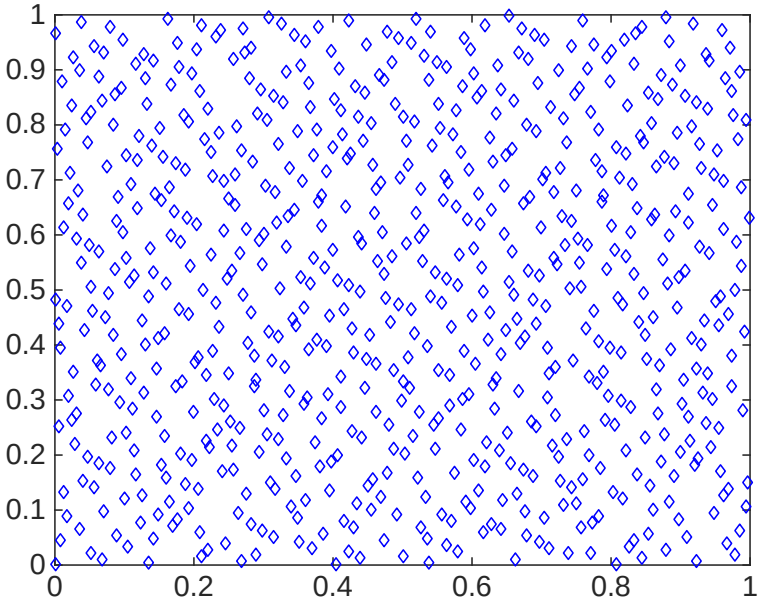
QMC grids may allow *polynomial* tractability with respect to the number of dimensions for integration problems (Wang, 2002).

Quasi-Monte Carlo Grid

Quasi-Monte Carlo sequences are based on quasi-random (deterministic) sequences with correlations between points to eliminate the clumping present in random Monte Carlo sequences, thus achieving higher uniformity. Quasi MC generated here using the MATLAB *Statistics and Machine Learning Toolbox*



MC sequence (1000 points)



Quasi MC (Halton) sequence (700 points)

Max min distance ≈ 0.048

Non-Differentiable HJB Value Function

The classical derivation of the HJB equation assumes that the value function is differentiable everywhere (Bryson and Ho, 1975). This assumption is not realistic is most cases. To address this issue, the definition of viscosity solutions of PDEs was introduced based on superdifferentials $D^{+}v(.)$ and subdifferentials $D^{-}v(.)$ (Crandall and Lions, 1983),

$$D^{+}v(\boldsymbol{x}) = \left\{ \boldsymbol{p} \in \mathbb{R}^n : \limsup_{\boldsymbol{y} \to \boldsymbol{x}} \frac{v(\boldsymbol{y}) - v(\boldsymbol{x}) - \boldsymbol{p} \cdot (\boldsymbol{y} - \boldsymbol{x})}{\|\boldsymbol{y} - \boldsymbol{x}\|} \leq 0 \right\}$$

$$D^{-}v(\boldsymbol{x}) = \left\{ \boldsymbol{q} \in \mathbb{R}^n : \liminf_{\boldsymbol{y} \to \boldsymbol{x}} \frac{v(\boldsymbol{y}) - v(\boldsymbol{x}) - \boldsymbol{q} \cdot (\boldsymbol{y} - \boldsymbol{x})}{\|\boldsymbol{y} - \boldsymbol{x}\|} \geq 0 \right\}$$

If both $D^{+}v(.)$ and $D^{-}v(.)$ are nonempty at $\boldsymbol{x}$, then the value function $v(.)$ is differentiable at $\boldsymbol{x}$,

$$D^{+}v(\boldsymbol{x}) = D^{-}v(\boldsymbol{x}) = \frac{\partial v(\boldsymbol{x})}{\partial \boldsymbol{x}}$$

Non-Differentiable HJB Value Function

Consider now the following nonlinear PDE,

$$F\left(\boldsymbol{x}, v(\boldsymbol{x}), \frac{\partial v(\boldsymbol{x})}{\partial \boldsymbol{x}}\right) = 0, \qquad \boldsymbol{x} \in \Omega \subseteq \mathbb{R}^n$$

The function *v*(.) is a viscosity subsolution of the PDE if it satisfies,

$$F(\boldsymbol{x}, v(\boldsymbol{x}), \boldsymbol{p}) \leq 0, \qquad \forall\, \boldsymbol{x} \in \Omega, \qquad \forall\, \boldsymbol{p} \in D^{+} v(\boldsymbol{x})$$

The function *v*(.) is a viscosity supersolution of the PDE if it satisfies,

$$F(\boldsymbol{x}, v(\boldsymbol{x}), \boldsymbol{q}) \geq 0, \qquad \forall\, \boldsymbol{x} \in \Omega, \qquad \forall\, \boldsymbol{q} \in D^{-} v(\boldsymbol{x})$$

The function *v*(.) that satisfies these two conditions is a viscosity solution of the PDE. The HJB equation can be derived using these definitions such that the value function is a solution in the viscosity sense (Bardi and Capuzzo-Dolcetta, 1997).

Finite Horizon Bolza Problem

The nonlinear Bolza control problem is described as follows,

$$\min\ J(\boldsymbol{x}_0) = \phi(\boldsymbol{x}(t_f)) + \int_{t_0}^{t_f} L(\boldsymbol{x}, \boldsymbol{u})\, \mathrm{d}t$$

subject to

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \qquad \boldsymbol{x}(t_f) \in C \quad t \in [t_0, t_f]$$

where $\boldsymbol{x} \in \Omega \subseteq \mathbb{R}^n$, $\boldsymbol{u} \in U \subseteq \mathbb{R}^m$, and $C \subset \Omega$. The final time t_f may be fixed in the problem definition or it may be a function of the initial state x_0 for time-free problems.

We employ the Kružkov transform $v(\boldsymbol{x})$ of the value function $J(\boldsymbol{x})$: $v(\boldsymbol{x}) = 1 - e^{-J(\boldsymbol{x})}$

Finite Horizon Bolza Problem

The Kružkov transform $v(\boldsymbol{x})$ of the value function $J(\boldsymbol{x})$ is the unique viscosity solution of the following HJB equation (Bardi and Capuzzo-Dolcetta, 1997),

$$v(\boldsymbol{x}) + \sup_{\boldsymbol{u}\in U}\left\{-\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})\frac{\partial v(\boldsymbol{x})}{\partial \boldsymbol{x}} - L(\boldsymbol{x},\boldsymbol{u}) + (L(\boldsymbol{x},\boldsymbol{u})-1)v(\boldsymbol{x})\right\} = 0, \qquad \boldsymbol{x}\in\Omega\setminus C$$

$$v(\boldsymbol{x}) = w(\boldsymbol{x}(t_f)), \qquad \boldsymbol{x}\in C$$

where

$$w(\boldsymbol{x}(t_f)) = 1 - e^{-\phi(\boldsymbol{x}(t_f))}$$

The transform allows a convenient definition of spatial boundary conditions in cases where initial conditions are unreachable via backward integration from the terminal conditions, i.e. the value function $J(.)$ becomes infinite, whereas the viscosity solution $v(.)$ becomes one.

Finite Horizon Bolza Problem

The numerical solution of this equation requires the definition of a discrete grid in the computation domain $\Omega \supset C$. The discrete HJB equation becomes (Cristiani and Martinon, 2010),

$$v_h(\boldsymbol{x}) = H_h(v_h(\boldsymbol{x}), \boldsymbol{x}), \qquad \boldsymbol{x} \in \Omega \setminus C$$

$$v_h(\boldsymbol{x}) = w(\boldsymbol{x}(t_f)), \qquad \boldsymbol{x} \in C$$

where

$$H_h(v_h(\boldsymbol{x}), \boldsymbol{x}) = \min_{\boldsymbol{u} \in U} \{ v_h(\boldsymbol{x} + h\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})) + hL(\boldsymbol{x}, \boldsymbol{u})(1 - v_h(\boldsymbol{x})) \}$$

and *h* is the discrete time step. The minimization indicated must be performed with respect to the control ***u*** at every grid node.

We use MATLAB *fmincon* for the minimization because it can solve problems where the control space *U* has high dimensionality.

The solution function $v_h(.)$ at $\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$ is determined via kriging regression because the point is normally not collocated with a grid node.

Finite Horizon Bolza Problem

The numerical viscosity solution $v_h(.)$ can be obtained using fixed point iterations on the grid,

$$v_h^{(n+1)}(\boldsymbol{x}) = H_h(v_h^{(n)}(\boldsymbol{x}), \boldsymbol{x})$$

The initial solution guess at the grid nodes can be the following,

$$v_h^{(0)}(\boldsymbol{x}) = 1 \qquad \boldsymbol{x} \in \Omega \setminus C$$

$$v_h^{(0)}(\boldsymbol{x}) = w(\boldsymbol{x}(t_f)) \qquad \boldsymbol{x} \in C$$

Once the fixed point iterations have converged, the feedback control synthesis for an arbitrary state $\boldsymbol{x}$ is implemented by finding the control $\boldsymbol{u}$ that minimizes the following equation,

$$H_h(v_h(\boldsymbol{x}), \boldsymbol{x}) = \min_{\boldsymbol{u} \in U} \{v_h(\boldsymbol{x} + h\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u})) + hL(\boldsymbol{x}, \boldsymbol{u})(1 - v_h(\boldsymbol{x}))\}$$

Example: Bolza Problem with Fixed Final Time

Consider the following problem with quadratic cost (Bryson and Ho, 1975),

$$\min J(\boldsymbol{x}) = \frac{1}{2}c(x_1(t_f))^2 + \frac{1}{2}\int_0^{t_f} u^2 \, \mathrm{d}t$$

$$\dot{x}_1 = u \quad \dot{x}_2 = -1 \quad x_2(t_f) = 0$$

where c and t_f are given. The computation domain Ω is $[0, 4]^2$.

The analytical solution is the following,

$$x_1(t) = -\frac{cx_1(0)}{1 + ct_f}t + x_1(0) \quad x_2(t) = t_f - t$$

$$u(t) = -\frac{x_1(t)}{(1/c) + t_f - t}$$

Example: Bolza Problem with Fixed Final Time

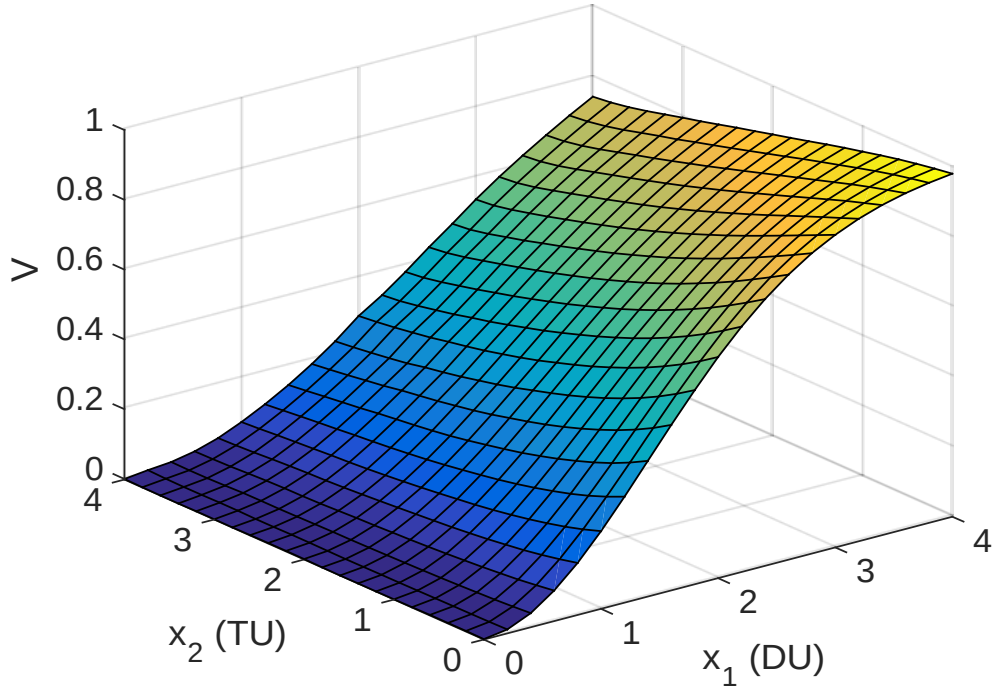
A numerical solution for the HJB equation is found for this problem with c = 0.5 using 400 grid nodes and a time step of 0.01 TU. The solution was found using 500 iterations.

Problems with Mayer terms require the cost definition in the terminal manifold, i.e. where the time-to-go is zero. Such cost definition is implemented by generating 40 grid nodes in the terminal manifold and constraining each node to be equal to the Kružkov transform of the actual terminal cost.

$$v(\boldsymbol{x}) = w(\boldsymbol{x}(t_f)), \qquad \boldsymbol{x} \in C$$

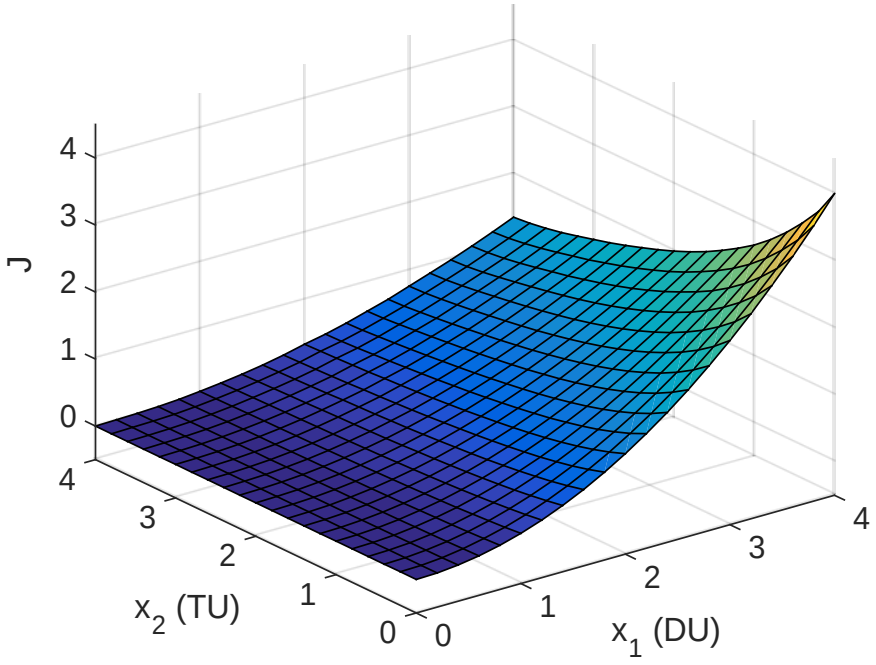
$$w(\boldsymbol{x}(t_f)) = 1 - e^{-\phi(\boldsymbol{x}(t_f))}$$

Example: Bolza Problem with Fixed Final Time



Viscosity solution $v(\boldsymbol{x})$

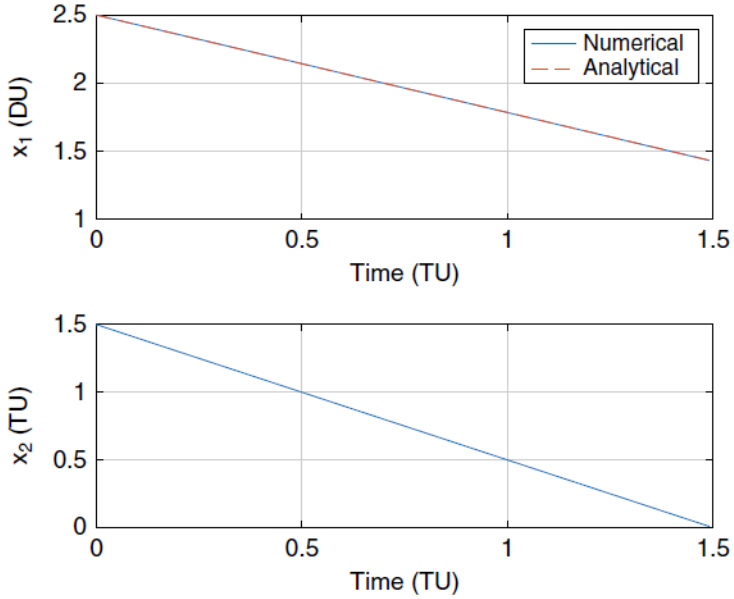
Example: Bolza Problem with Fixed Final Time

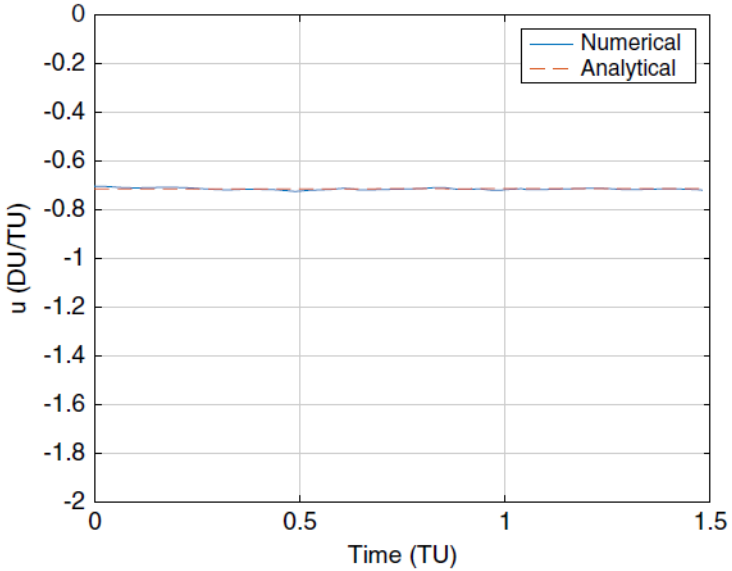


Value function $J(\boldsymbol{x})$

Example: Bolza Problem with Fixed Final Time

Optimal state and control histories for $x_1(0) = 2.5$ DU and $x_2(0) = 1.5$ TU.

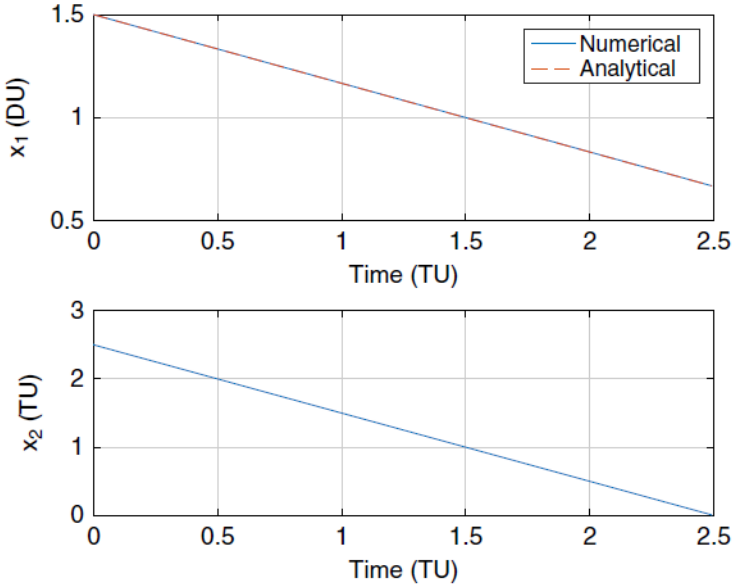


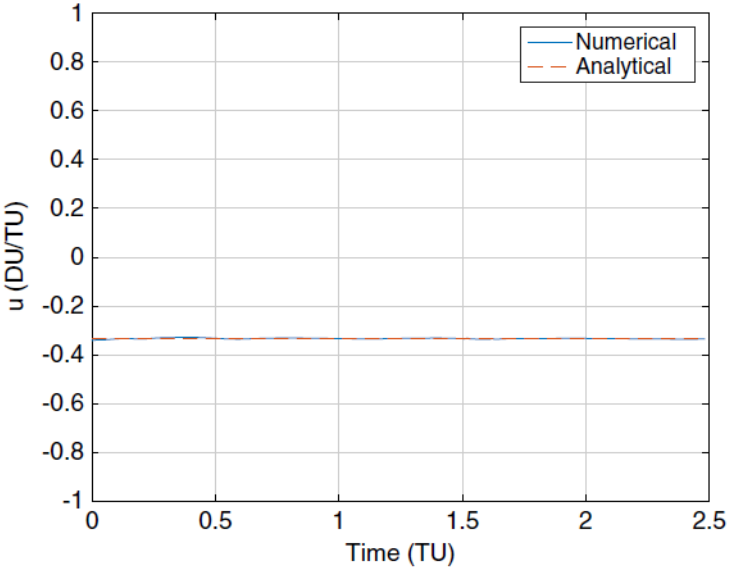


The error of the numerical cost is 1.30×10^{-5} .

Example: Bolza Problem with Fixed Final Time

Optimal state and control histories for $x_1(0) = 1.5$ DU and $x_2(0) = 2.5$ TU.





The error of the numerical cost is 5.42×10^{-6} .

Example: Free Final Time Problem with Nondifferentiable Value Function

Consider the following minimum-time problem (Cristiani and Martinon, 2010),

$$\min J = t_f$$

subject to

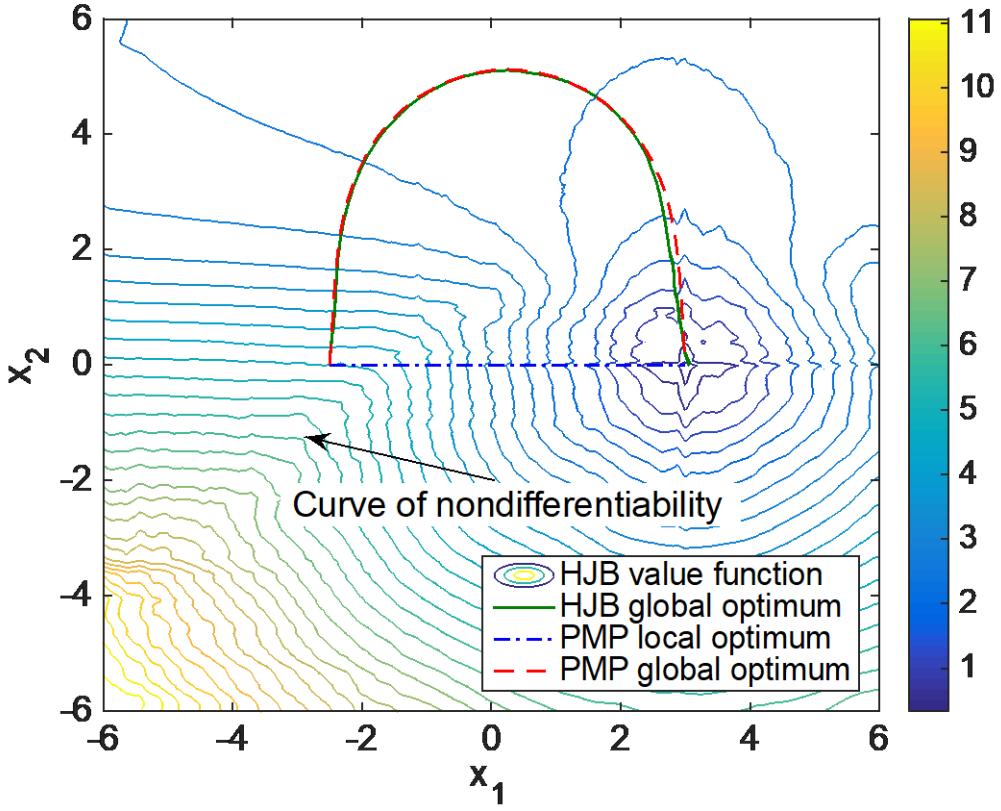
$$\dot{x}_1 = c(x_1, x_2)\cos(u)$$
$$\dot{x}_2 = c(x_1, x_2)\sin(u)$$
$$\boldsymbol{x}(t_0) = [-2.5 \quad 0]^T$$
$$\boldsymbol{x}(t_f) = [3 \quad 0]^T$$
$$u(t) \in [-\pi, \pi]$$

With

$$c(x_1, x_2) = \begin{cases} 1 & \text{if } x_2 \leq 1 \\ (x_2 - 1)^2 + 1 & \text{if } x_2 > 1 \end{cases}$$

The computation domain is $\Omega = [-6,6]^2$. The discrete HJB equation is solved using a grid of 1100 nodes and a time step of 0.02 TU for 5000 iterations (13.9 h, 12 parallel workers).

Example: Free Final Time Problem with Nondifferentiable Value Function

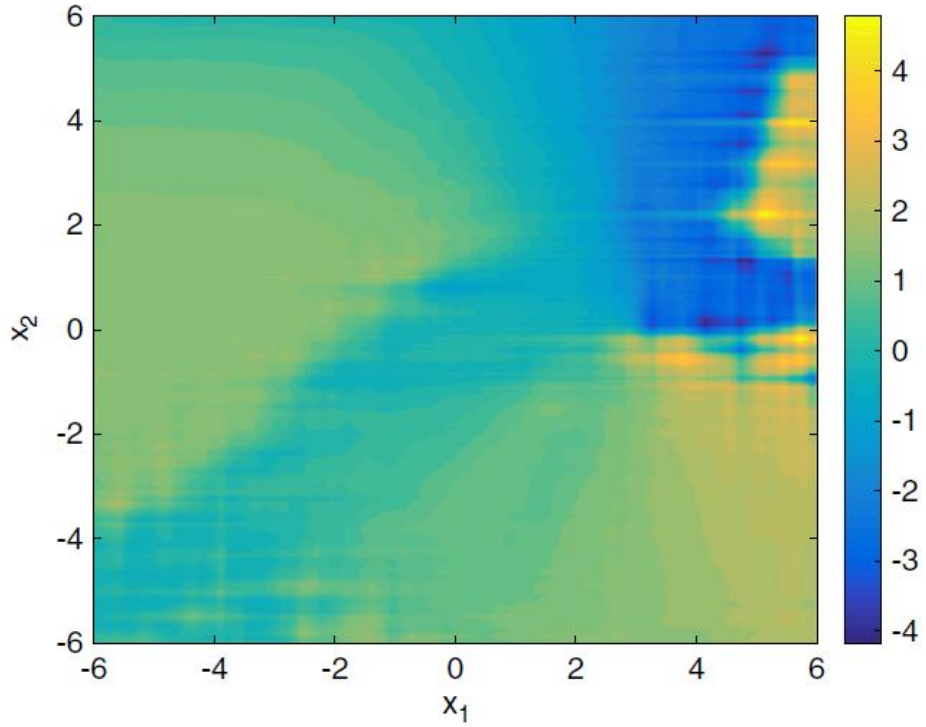


The problem has 2 locally optimal solutions for the given IC and TC (straight line J = 5.50 TU and arch J = 4.87 TU).

Either locally optimal solution could be found using the GPOPS-II depending on initial guess. Only way to verify global optimality is using HJB equation.

HJB feedback simulation is implemented with time step of 0.001 TU.

Example: Free Final Time Problem with Nondifferentiable Value Function



Optimal feedback control u^*

Infinite Horizon Discounted Regulator

A second class of optimal control problems corresponds to the infinite horizon discounted regulator. It is used in cases where a plant is to kept operating at a setpoint optimally for an indefinite time period in the presence of disturbances and uncertainty.

The nonlinear problem includes an exponentially decaying factor with a rate dependent on the parameter λ,

$$\min J(\boldsymbol{x}) = \int_{t_0}^{\infty} L(\boldsymbol{x}, \boldsymbol{u}) e^{-\lambda t} \, \mathrm{d}t$$

subject to

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, \boldsymbol{u}) \quad \boldsymbol{x}(t_0) = \boldsymbol{x}_0, \qquad \boldsymbol{u}(t) \in U$$

No terminal conditions → Kružkov transform is not necessary.

Infinite Horizon Discounted Regulator

The value function $J(.)$ is the unique viscosity solution of the following HJB equation (Bardi and Capuzzo-Dolcetta, 1997),

$$\lambda J(\boldsymbol{x}) + \sup_{\boldsymbol{u}\in U}\left\{-\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})\frac{\partial J(\boldsymbol{x})}{\partial \boldsymbol{x}} - L(\boldsymbol{x},\boldsymbol{u})\right\} = 0, \qquad \boldsymbol{x}\in\Omega\subseteq R^n$$

The discrete form of this HJB equation (Falcone, 1997),

$$J_h(\boldsymbol{x}) = \min_{\boldsymbol{u}\in U}\{(1-\lambda h)J_h(\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})) + hL(\boldsymbol{x},\boldsymbol{u})\}, \quad \boldsymbol{x}\in\Omega$$

is solved numerically on a grid in the computation domain Ω.

MATLAB *fmincon* is used for the control minimization, and universal kriging is used to determine the solution function $J_h(.)$ at $\boldsymbol{x}+h\boldsymbol{f}(\boldsymbol{x},\boldsymbol{u})$ since the point is not collocated with a grid node.

Example: Nondifferentiable Infinite Horizon Discounted Regulator

Consider the following optimal control problem (Falcone, 1997),

$$\min J(\boldsymbol{x}) = \int_{t_0}^{\infty} (|x_1| - 1)^2 e^{-\lambda t}\, \mathrm{d}t$$

subject to

$$\dot{x}_1 = x_2 u \quad \dot{x}_2 = 0 \quad u \in [-1, 1]$$

where $\Omega = [-2, 2]^2$ and is $\lambda = 1$ TU^{-1}. The HJB equation for this problem becomes,

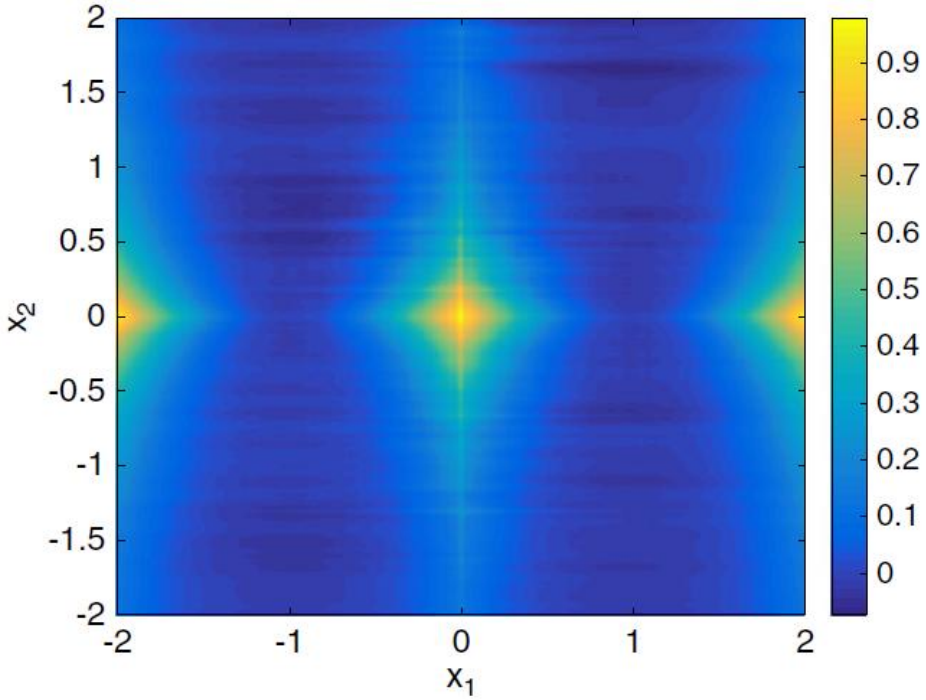
$$J(\boldsymbol{x}) + \sup_{u \in U} \left\{ -x_2 \frac{\partial J(\boldsymbol{x})}{\partial x_1} u - (|x_1| - 1)^2 \right\} = 0, \qquad \boldsymbol{x} \in \Omega$$

The bang-bang optimal control is given by $u^* = \begin{cases} -1 & \text{if } x_2 \frac{\partial J(\boldsymbol{x})}{\partial x_1} > 0 \\ +1 & \text{if } x_2 \frac{\partial J(\boldsymbol{x})}{\partial x_1} < 0 \end{cases}$

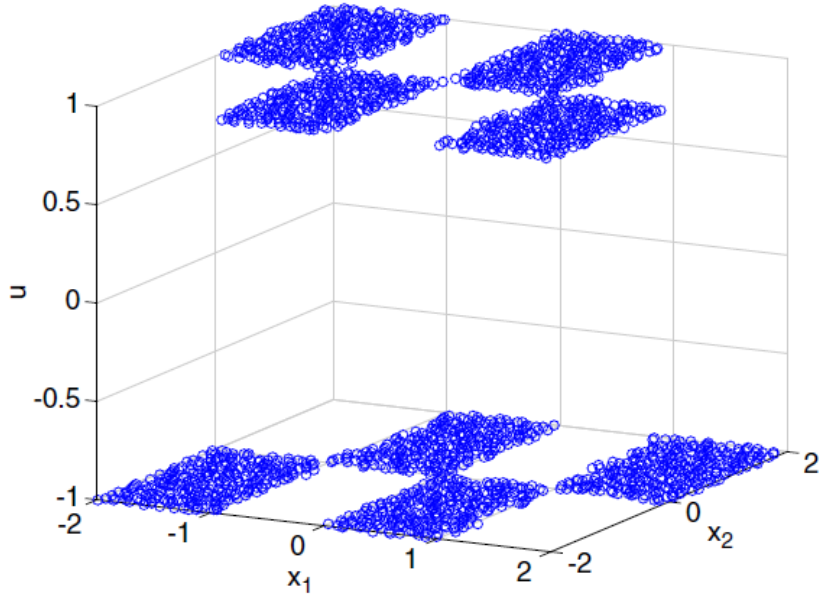
This simplifies the computation of the numerical solution.

Example: Nondifferentiable Infinite Horizon Discounted Regulator

Grid has 2000 nodes; time step = 0.01 , 2000 iterations require 14.4 hours.



Value function



Optimal feedback control

Example: High Dimensional Infinite Horizon Discounted Regulator

Consider the following spacecraft attitude regulation problem (Adurthi et al., 2017),

$$\min J(\boldsymbol{x}) = \int_{t_0}^{\infty} ([\boldsymbol{\rho}^T \quad \boldsymbol{\omega}^T]\boldsymbol{Q}[\boldsymbol{\rho}^T \quad \boldsymbol{\omega}^T]^T + \boldsymbol{u}^T\boldsymbol{R}\boldsymbol{u})e^{-\lambda t}\,\mathrm{d}t$$

subject to

$$\begin{aligned}
\dot{\rho} &= H(\boldsymbol{\rho})\boldsymbol{\omega} \\
I_1\dot{\omega}_1 + (I_3 - I_2)\omega_2\omega_3 &= u_1 \\
I_2\dot{\omega}_2 + (I_1 - I_3)\omega_1\omega_3 &= u_2 \\
I_3\dot{\omega}_3 + (I_2 - I_1)\omega_2\omega_1 &= u_3
\end{aligned}$$

where

$$H(\boldsymbol{\rho}) = \frac{1}{2}(I - S(\boldsymbol{\rho}) + \boldsymbol{\rho}\boldsymbol{\rho}^T) \quad S(\boldsymbol{\rho}) = \begin{bmatrix} 0 & \rho_3 & -\rho_2 \\ -\rho_3 & 0 & \rho_1 \\ \rho_2 & -\rho_1 & 0 \end{bmatrix}$$

Example: High Dimensional Infinite Horizon Discounted Regulator

We assume: $I_1 = 14$ kg m^2, $I_2 = 10$ kg m^2, $I_1 = 8$ kg m^2

$\boldsymbol{Q} = 40\,\boldsymbol{I}$, $\boldsymbol{R} = \boldsymbol{I}$

$\Omega = [-1, 1]^6$, $\lambda = 0.01$ s^{-1}

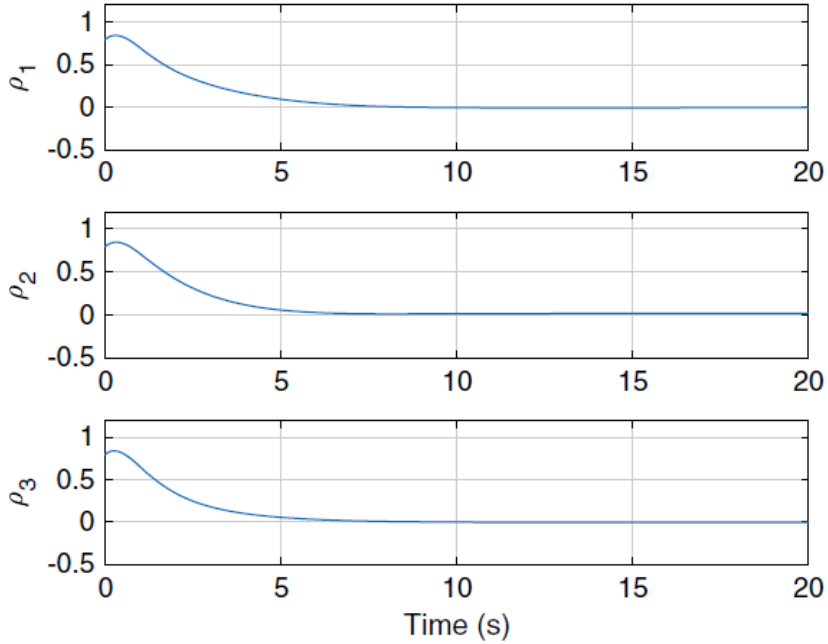
The initial conditions are $[\boldsymbol{\rho} \quad \boldsymbol{\omega}]^T = [0.8 \quad 0.8 \quad 0.8 \quad 0.2 \quad 0.2 \quad 0.2]^T$

The discrete HJB equation is solved using 5000 grid nodes and a time step of 0.01 s for 600 iterations.

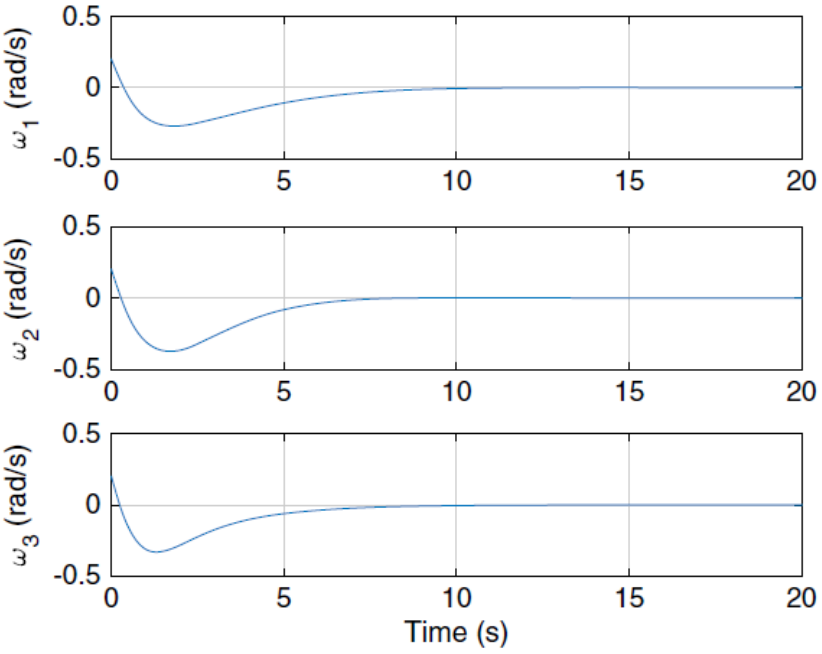
The initial guess for the value function $J(.)$ is the 2-norm of each grid node position.

The computational time was 11.8 days using MATLAB with 12 parallel workers.

Example: High Dimensional Infinite Horizon Discounted Regulator

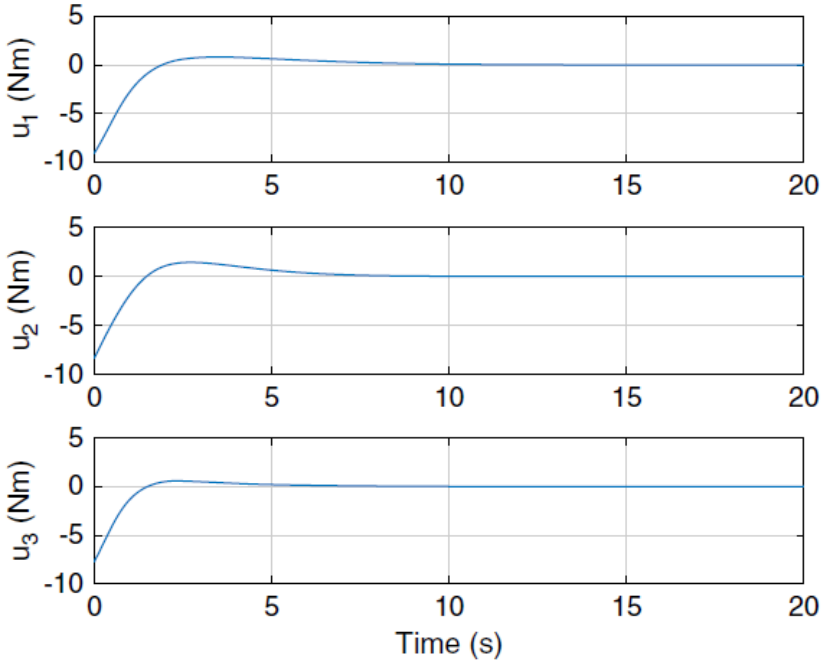


Optimal histories of the Rodrigues parameters



Optimal histories of the angular velocities

Example: High Dimensional Infinite Horizon Discounted Regulator



Optimal histories of the controls (torques)

The spacecraft is brought virtually to rest and the controls go to zero after approx. 10 s, which agrees with previous work (Adurthi et al., 2017)

The use of a global kriging model to implement the optimal feedback controller allows the control to be determined in milliseconds using MATLAB.

Conclusions

- The space complexity of the HJB problem is reduced by using quasi-Monte Carlo grids.
- Viscosity solution of the HJB PDE is used to address nondifferentiability of the value function.
- The HJB equation is solved using fixed point iterations on a discrete grid; this process has parallel scalability.
- When interpolation of the value function or viscosity solution is needed, universal kriging is used. This gives a result of greater accuracy than simpler interpolation formulas with fewer grid nodes.
- Once the HJB value function is obtained on the discrete grid, a kriging model using $\boldsymbol{x}$ as input, can yield the optimal control $\boldsymbol{u}^*$. This feedback control synthesis requires only milliseconds using MATLAB.
- The presented methodology for the solution of the HJB equation should allow more sophisticated problems to be solved.