A near model-free method for solving the Hamilton-Jacobi-Bellman equation in high dimensions

Mathias Oster, Leon Sallandt, Reinhold Schneider

Technische Universität Berlin

ICODE Workshop on numerical solutions of HJB equations

10.01.2020

DAEDALUS
RTG 2433

Technische Universität Berlin

Freie Universität Berlin

Otto von Guericke Universität Magdeburg

Motivation and Ingredients

DAEDALUS
RTG 2433

Aim: Calculate optimal feedback laws (via HJB) for controlled PDEs.

Ingredients:

1. Reformulate the HJB equation as operator equation.
2. Use Monte Carlo integration for least squares approximation.
3. Use non linear, smooth Ansatz space: HT/TT – tree-based tensors.

Classical optimal control problem

DAEDALUS RTG 2433

Optimal control problem: find $u \in L^2(0, \infty)$ such that

$$\min_u J(x, u) = \min_u \int_0^\infty \frac{1}{2}\|x(s)\|^2_{\mathbb{R}^n} + \frac{\lambda}{2}|u(s)|^2 ds,$$

subject to

$$\begin{aligned} \dot{x} &= f(x, u), \quad x \in \Omega \subset \mathbb{R} \\ x(0) &= x_0 \end{aligned}$$

1. Note that the differential equation can be high-dimensional
2. linear ODE and quadratic cost $\rightarrow$ Riccati equation
3. nonlinear ODE and nonlinear cost $\rightarrow$ Hamilton-Jacobi-Bellman (HJB) equation

Feedback control problem

Define a feedback-law $\alpha(x(t)) = u(t)$. Rephrase

$$\min_{\alpha} J^{\alpha}(x) = \min_{\alpha} \int_0^{\infty} \underbrace{\frac{1}{2}\|x(s,\alpha)\|_{\mathbb{R}^n}^2 + \frac{\lambda}{2}|(\alpha(x))(s)|^2}_{=:r^{\alpha}(x)} \mathrm{d}s,$$

Our goal: find an optimal feedback law $\alpha^*(x) = u$.

Defining the *value function*

$$v(x) := \inf_{\alpha} J^{\alpha}(x) \in \mathbb{R}$$

Idea: if v is differentiable, the feedback law is given by

$$\alpha(x) = -\frac{1}{\lambda} D_x v(x) \circ D_u f(x,u) \quad \text{(easy to calculate!)}.$$

The HJB equation

The value function obeys

$$\inf_{\alpha} \left\{ f(x, \alpha(x)) \cdot \nabla v(x) + r^{\alpha}(x) \right\} = 0$$

HJB equation is highly nonlinear and potentially high-dimensional!

But: For fixed policy $\alpha(x)$ it reduces to a linear equation: Defining $L^{\alpha} := -f(x, \alpha) \cdot \nabla$ we get

$$L^{\alpha} v^{\alpha}(x) - r^{\alpha}(x) = 0.$$

Methods of characteristics

Linearized HJB:

$$L^\alpha v^\alpha(x) - r^\alpha(x) = 0.$$

Using the methods of characteristics we obtain

$$\begin{aligned} \dot{x}(t) &= f(x, \alpha), \\ v^\alpha(x(0)) &= \int_0^\tau r^\alpha(x(t))dt + v^\alpha(x(s)), \end{aligned}$$

which we call Bellman-like equation.

Reformulation as Operator Equation

Consider the Koopman operator:

$$K_\tau^\alpha : L_{loc,\infty}(\Omega) \to L_{loc,\infty}(\Omega), \quad K_\tau^\alpha[g](x) = g(x(\tau)).$$

Rewrite the Bellman-like equation: For all $x \in \Omega$:

$$v^\alpha(x(0)) = \int_0^\tau r^\alpha(x(t))dt + v^\alpha(x(s)),$$

as

$$(\mathrm{Id} - K_\tau^\alpha)[v](x) = \underbrace{\int_0^\tau K_t^\alpha r(x)dt}_{=:R_\tau^\alpha(x)}.$$

DAEDALUS
RTG 2433

Policy iteration uses a sequence of linearized HJB equations.

Algorithm (Policy iteration)

Initialize with stabilizing feedback α_0. Solve until convergence

1. *Find v_{i+1} such that $(Id - K_\tau^{\alpha_i})v_{i+1}(\cdot) - R_\tau^{\alpha_i}(\cdot) = 0$.*
2. *Update policy according to $\alpha_{i+1}(x) = -\frac{1}{\lambda} D_x v_{i+1}(x) \circ D_u f(t,x,u)$.*

Least squares ansatz

Problem: We need to solve

$$(\mathrm{Id} - K_\tau^{\alpha_i})v^{\alpha_{i+1}}(\cdot) - R_\tau^{\alpha_i}(\cdot) = 0.$$

Idea: Solve on suitable S

$$v_{\alpha_{i+1}} = \operatorname*{arg\,min}_{v \in S} \underbrace{\|(\mathrm{Id} - K_\tau^{\alpha_i})v(\cdot) - R_\tau^{\alpha_i}(\cdot)\|_{L^2(\Omega)}^2}_{=\int_\Omega |(\mathrm{Id} - K_\tau^{\alpha_i})v(x) - R_\tau^{\alpha_i}(x)|^2 dx}.$$

DAEDALUS RTG 2433

Algorithm (Projected Policy iteration)

Initialize with stabilizing feedback α_0. Solve until convergence

1. *Find*

$$v_{i+1} = \arg\min_{v \in S} \|(Id - K_\tau^{\alpha_i})v(\cdot) - R_\tau^{\alpha_i}(\cdot)\|_{L^2(\Omega)}^2.$$

2. *Update policy according to* $\alpha_{i+1}(x) = -\frac{1}{\lambda} D_x v_{i+1}(x) \circ D_u f(x,u)$.

Variational Monte-Carlo

DAEDALUS RTG 2433

Approximate by Monte-Carlo quadrature

$$\|(\mathrm{Id} - K_\tau^{\alpha_i})v(\cdot) - R_\tau^{\alpha_i}(\cdot)\|^2_{L^2(\Omega)} \approx \frac{1}{n}\sum_{j=1}^{n} |(\mathrm{Id} - K_\tau^{\alpha_i})v(x_j) - R_\tau^{\alpha_i}(x_j)|^2.$$

$$v^*_{n,s} = \arg\min_{v \in S} \frac{1}{n}\sum_{j=1}^{n} |(\mathrm{Id} - K_\tau^{\alpha_i})v(x_j) - R_\tau^{\alpha_i}(x_j)|^2$$

Proposition ([Eigel, Schneider et al, 19)

] Let $\epsilon > 0$ such that $\inf_{v_s \in S} \|v^* - v_s\|^2_{L_2(\Omega)} \leq \epsilon$. Then

$$\mathbb{P}[\|v^* - v^*_{(n,s)}\|^2_{L_2(\Omega)} > \epsilon] \leq c_1(\epsilon)e^{-c_2(\epsilon)n}$$

with $c_1, c_2 > 0$.

Exponential decay with number of samples chosen.

Solving the VMC equation

$$\arg\min \sum_{j=1}^{n} |(\mathrm{Id} - K_\tau^{\alpha_i})v(x_j) - R_\tau^{\alpha_i}(x_j)|^2.$$

1. $v(x_j) \to$ evaluate v at samples x_j.
2. $K_\tau^{\alpha_i} v(x_j) \to$ evaluate v at transported samples (with policy α_i).
3. $R_\tau^{\alpha_i}(x_j) \to$ approximate reward by trapezoidal rule

What do we need for solving the equation?

Model-free solution is possible. Only a black-box solver of the ODE is needed.

What do we need for updating the policy?

We need $D_u f(x, u)$, i.e. The derivative of the rhs w.r.t. the control.

Possible ansatz spaces

- Full linear space of polynomials
- Low-rank tensor manifolds
- Deep Neural Networks

Here used:

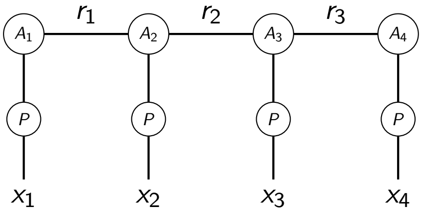
Low rank Tensor Train (TT-tensor) manifold

- Riemanian manifold structure
- Explicit representation of tangential space
- Convergence theory for optimization algorithms

Tensor Trains

- Consider $\Pi_i = (1, x_i, x_i^2, x_i^3, .., x_i^k)$ one-dimensional polynomials.
- Tensor product $\Pi = \bigotimes_{i=1}^n \Pi_i$.
- $\dim(\Pi) = (k+1)^n$, huge if $n \gg 0$.
- Reduce size of Ansatz space by considering non-linear $\mathcal{M} \subset \Pi$.

$v(x) =$



DAEDALUS RTG 2433

Modify cost-functional:

$$\mathcal{R}_N(v) = \frac{1}{n}\sum_{j=1}^{n} |(\mathrm{Id} - K_\tau^{\alpha_i})v(x_j) - R(x_j)|^2 + \underbrace{|v(0)|^2 + |\nabla v(0)|^2}_{\text{vanishes in exact case}} + \underbrace{\mu\|v\|^2_{H^1(\Omega)}}_{\text{regularizer}}$$

Example: Schloegl-like equation

DAEDALUS RTG 2433

Consider a Schlögl like system with Neumann boundary condition, c.f. [1, Dolgov, Kalise, Kunisch, 19]. Solve for $x \in \Omega = L^2(-1,1)$

$$\min_u J(x,u) = \min_u \int_0^\infty \frac{1}{2}\|x(s)\|^2 + \frac{\lambda}{2}|u(s)|^2 \mathrm{d}s,$$

subject to

$$\begin{aligned} \dot{x}(t) &= \sigma \Delta x(t) + x(t)^3 + \chi_\omega u(t) \\ x(0) &= x_0. \end{aligned}$$

χ_ω is characteristic function on $\omega = [-0.4, 0, 4]$.

After discretization in space (finite differences):

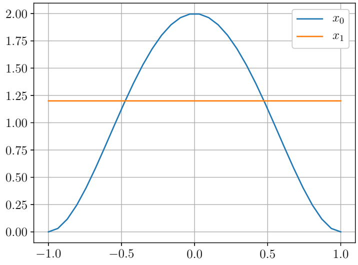
$$\begin{pmatrix} \dot{x}_1 \\ \vdots \\ \dot{x}_n \end{pmatrix} = A \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix} + \begin{pmatrix} x_1^3 \\ \vdots \\ x_n^3 \end{pmatrix} + u \begin{pmatrix} 0 \\ \vdots \\ 1 \\ \vdots \\ 0 \end{pmatrix}$$

DAEDALUS
RTG 2433

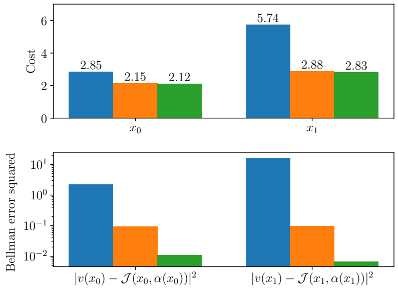
TT Degrees of Freedom

Full space: 5^{32}. Reduced to ≈ 5000.

Example: Schloegl-like equation



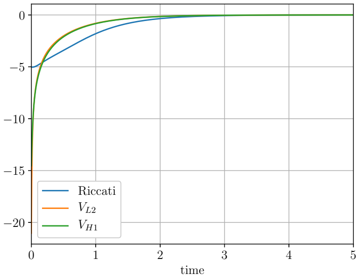
(a) Initial values.



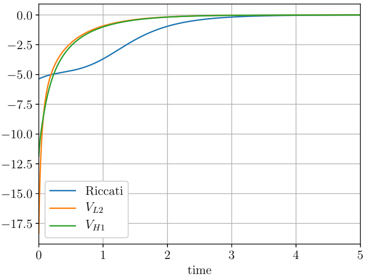
(b) Generated cost and least squares error. Blue is Riccati, orange is V_{L_2} and green is V_{H_1}.

Figure: The generated controls for different initial values.

Example: Schloegl-like equation



(a) Generated controls, initial value x_0



(b) Generated controls, initial value x_1

Figure: The generated controls for different initial values.

What do we need for optimization

We only need

- a discretization of the flow Φ (blackbox)
- the derivative of the rhs $f(x, u)$ w.r.t. the control (easy if linear)
- the cost functional

to solve the equation and generate a feedback law.

Thank you for your attention

References and related work

DAEDALUS
RTG 2433

Sergey Dolgov, Dante Kalise, and Karl Kunisch.
A Tensor Decomposition Approach for High-Dimensional Hamilton-Jacobi-Bellman Equations.
arXiv e-prints, page arXiv:1908.01533, Aug 2019.

Martin Eigel, Reinhold Schneider, Philipp Trunschke, and Sebastian Wolf.
Variational monte carlo—bridging concepts of machine learning and high-dimensional partial differential equations.
Advances in Computational Mathematics, Oct 2019.

Mathias Oster, Leon Sallandt, and Reinhold Schneider.
Approximating the stationary hamilton-jacobi-bellman equation by hierarchical tensor products, 2019.