Journal of Mathematical Modeling
Vol. 10, No. 1, 2022, pp. 71–91. Research Article

JMM

Approximate solution of the Hamilton-Jacobi-Bellman equation

Atefeh Gooran Orimi[1], Sohrab Effati[1,2*], Mohammad Hadi Farahi[1,3]

[1]*Department of Applied Mathematics, Ferdowsi University of Mashhad, Mashhad, Iran*
[2]*Center of Excellence of Soft Computing and Intelligent Information Processing (SCIIP), Ferdowsi University of Mashhad, Mashhad, Iran*
[3]*The Center of Excellence on Modeling and Control Systems (CEMCS), Mashhad, Iran*
Email(s): atefehgooranorimi@yahoo.com, s-effati@um.ac.ir, farahi@math.um.ac.ir

Abstract. The Hamilton-Jacobi-Bellman (HJB) equation, as a notable approach obtained from dynamic programming, is widely used in solving optimal control problems that results in a feedback control law. In this study, the HJB equation is first transformed into the Convection-Diffusion (CD) equation by adding a viscosity coefficient. Then, a novel numerical method is presented to solve the corresponding CD equation and to obtain a viscosity solution of the HJB. The proposed approach encompasses two well-known methods of Finite Volume Method (FVM) and Algebraic Multigrid (AMG). The former as a reliable method for solving parabolic PDEs and the latter as a powerful tool for acceleration. Finally, numerical examples illustrate the practical performance of the proposed approach.

Keywords: Optimal control problems, Hamilton-Jacobi-Bellman (HJB) equation, convection-diffusion equation, finite volume method, algebraic multigrid method.
AMS Subject Classification 2010: 34A34, 65L05.

1 Introduction

Let $\Omega \subset \mathbb{R}^n$ and $\mathscr{U} \subset \mathbb{R}^m$ be the sets of all possible states and admissible controls respectively. Also for any arbitrary $x \in \Omega$ and $u \in \mathscr{U}$ consider the control system as

$$\begin{aligned} &\dot{x}(t) = \mathbf{f}(t, x(t), u(t)), \quad t_0 \le t \le t_f, \\ &x(t_0) = x_0, \end{aligned} \tag{1}$$

where $x_0 \in \Omega$ is given and $\mathbf{f}(.)$ assumed to be continuously differentiable. An optimal control problem is to find the optimal control $u^* \in \mathscr{U}$ which has the ability of moving the state from initial state x_0 to any

*Corresponding author.
Received: 10 December 2020 / Revised: 4 May 2021 / Accepted: 24 May 2021
DOI: 10.22124/jmm.2021.18386.1579

final state in a finite time interval and also optimizing a performance index given by

$$J(t_0, x(t_0), u(\cdot)) = h(t_f, x(t_f)) + \int_{t_0}^{t_f} L(\tau, x(\tau), u(\tau)) d\tau. \quad (2)$$

Many approaches have been presented to solve the optimal control problem (1)-(2) which can be divided in two main categories of (a) discretization based approaches, in which a straightforward algorithm is used to discretize the domain and transform the problem into solving a set of algebraic equations such as meshless methods [10], Wavelet methods [21, 31], Neural Networks [13] and many more; and (b) analytic based approaches which either study the Pontryagin's principle [11, 14, 38] or the dynamic programming [28]. The two analytic based approaches take advantage of the optimality conditions to derive out new sets of equations. Nevertheless, they apply different optimality conditions and thereby result in solving an ordinary or partial differential equation (the HJB equation) respectively. The main difference between these two methods is in providing a closed- or open-loop solution, however. The closed-loop solution has an utmost application in different fields of study where, for e.g., online information is required [26, 43]. Nonetheless, solving the HJB equation -as a PDE- is more challenging in comparison to the ODE provided by the Pontryagin's Principle. Therefore, exact and applicable methods are required to obtain reliable results.

In related work, Saberi Nik et al. in [28] employed the He's polynomials and presented a homotopy perturbation method for solving the HJB equation that results in an analytical-approximate solution. Other studies, e.g. [32, 33] investigate the HJB equation for a class of fractional optimal control problems. Furthermore, Richardson and Wang in [34] proposed a reliable algorithm using Finite Volume Method (FVM) for solving the viscosity approximation of the HJB equation. This viscosity approximation is obtained by adding the diffusion coefficient to the HJB equation and reformulating it as the convection-diffusion equation. This approach has been first introduced by [22] and frequently used in the literature since its introduction. More importantly, many of the discretization based algorithms, such as finite element [5] and finite volume methods [9, 25] have been developed to solve the convection-diffusion equation. Nevertheless, such approaches often result in solving a system of algebraic equations that consequently fail to deal with high-dimensional problems. This is due to the huge amount of nodes required for the accuracy of the method which increases the computations enormously. This can be seen as a common drawback of the discretization approaches for solving PDEs that suffer from the lack of memory and time consuming during the implementation because of the number of grid nodes [34]. In our contribution, the goal is to compensate for the lack of accelerated based approaches for solving the HJB equation. In this manner, we apply FVM for solving the corresponding convection-diffusion equation of the HJB and further employ a class of Algebraic Multigrid (AMG) method. FVM, as a well-known method for transforming PDEs into a set of algebraic equations, has been widely used for solving parabolic PDEs, especially in Computational Fluid Dynamics (CFDs), because of its conservation property [25]. In addition, a wide variety of AMGs have been recently developed that, based on their robustness and efficiency properties, target various problems [3, 16]. For instance, Han and Wan in [17] applied a class of multigrid method for solving the second order HJB and Hamilton-Jacobi-Bellman-Isaacs (HJBI) equations (also see [18] as an important work in this direction). Both contributions [17, 18] exploit a geometric multigrid approach combined with a finite difference method to solve the HJB equation. In contrast to the algebraic multigirds, the geometric multigrid methods require information of the grid space and are based on the assumption that the matrix A (in the linear system $Ax = b$) is symmetric. Such an assumption, however, cannot be satisfied in many of the numerical methods used for solving the HJB

equation. As a consequence, the approaches presented in [17, 18] are not applicable in the general case. In addition, their work is only intended to solve the HJB equation obtained from a set of financial problems and does not solve the optimal control problems directly. On the contrary, we here apply algebraic MG as an acceleration of the traditional iterative methods (also called smoother methods), which make the high-frequency components of the error milder. The proposed approach only requires the information of the matrix A as being M-matrix and can be directly applied when there is no grid in the background (we further discuss the properties of AMGs in the upcoming sections). We also solve the convection-diffusion equation as an approximation to the HJB equation which later provides the solution of the considered optimal control problem (1)-(2). Moreover, we emphasize that the application of the proposed approach is beyond solving the convection-diffusion equation and the method can be modified for solving many of the similar PDEs. The rest of the paper is organized as follows: we first present the preliminary concepts in section 2 and discuss the HJB equation. Section 3 is devoted to numerical study. We will first point out the salient features of the FVM and also the AMG approach and then propose the AMG-FVM algorithm for solving the considered optimal control problem (1)-(2). Convergence properties of the proposed method are then discussed accordingly. Numerical examples are further brought in section 4 to illustrate the efficacy of the presented algorithm and assess its reliability. Finally, the paper is concluded in section 5.

2 The Hamilton-Jacobi-Bellman equation

Consider the problem stated in (1)-(2) and let $t \in [t_0, t_f)$ denote the initial time (we assume t to be arbitrary). Next, let v be a continuously differentiable function given by

$$v(t, x(t)) = \inf_{u(\cdot)\in\mathscr{U}} J(t, x(t), u(\cdot)).$$

Applying the Bellman's principle [27], the HJB equation can be presented as a PDE given by

$$-v_t = \inf_{u(\cdot)\in\mathscr{U}} \{\nabla v \cdot \mathbf{f}(t, x(t), u(t)) + L(t, x(t), u(t))\}, \tag{3}$$

with the terminal condition

$$v(t_f, x(t_f)) = h(t_f, x(t_f)), \tag{4}$$

where ∇ denotes the gradient operator with respect to x. In the case of linear quadratic optimal control problems -a quadratic cost functional and a linear system- an analytic solution is proposed using the Riccati equation (see, e.g., [1]). This cannot be extended to general cases, however. In addition, no classical solution $v(t, x(t))$ -that is $v \in C^1((t_0, t_f] \times \Omega)$- exists in general for the problem (3)-(4). This is a well-known issue that occurs even for simple control problems (see Example 1 in section 4). To address this issue, it is common to use the vanishing viscosity approach [2, 6–8] which is adding a diffusion term to the HJB equation (3) and approximating it with a second order parabolic PDE as:

$$-v_{\varepsilon t} - \varepsilon \nabla^2 v_\varepsilon = \inf_{u(\cdot)\in\mathscr{U}} \{\nabla v_\varepsilon \cdot \mathbf{f}(t, x(t), u(t)) + L(t, x(t), u(t))\}, \ (t, x(t)) \in [t_0, t_f) \times \Omega, \tag{5}$$

where $0 < \varepsilon \ll 1$ is the diffusion coefficient (a.k.a. viscosity). In most practical situations, a numerical approach is then applied to solve the above equation for small values of ε and therefore approximate the

viscosity solution of the HJB equation (3). This, however, is still a challenging task since no boundary conditions exist for the problem. As a consequence, one will encounter some inefficiencies in applying a discretization method. Such problems with only one terminal condition often use explicit time stepping schemes, and therefore are subject to conditional stability [35]. In other words, too many time steps and also too many grids for the spatial solution domain in each time step are required to satisfy the stability condition for high-dimensional problems. This increases the computations in the algorithm and makes the problem intractable. In this respect, a new artificial boundary condition on a novel extended domain will be introduced to alleviate the complexity of the problem [19]. To this, let $\tilde{\Omega}$ be any extended domain of Ω (i.e. $\Omega \subset \tilde{\Omega}$), that is just assumed to be bounded, and $g(t,x(t))$ be an arbitrary artificial Dirichlet boundary condition. In practice, the function $g(t,x(t))$ for any $t \in [t_0,t_f)$ is set to be equal to $h(t,x(t))$. We then define the terminal and boundary conditions of the equation (5) as follows:

$$v_\varepsilon(t_f,x(t_f)) = h_\varepsilon(t_f,x(t_f)), \quad x(t_f) \in \tilde{\Omega}, \tag{6}$$
$$v_\varepsilon(t,x(t)) = g_\varepsilon(t,x(t)), \quad (t,x(t)) \in [t_0,t_f) \times \partial\tilde{\Omega}. \tag{7}$$

The choice of $\tilde{\Omega}$, however, is important as if the extended domain is too small, the approximate solution will be affected by the artificial boundary and yield unreliable results and if it is too large, many grids and hence excessive computations are required for convergence. The following theorem illustrates the relation between solutions of the problem (5) with conditions (6)-(7) and the HJB equation (3)-(4) which we refer the readers to [35] for its proof.

Theorem 1 ([35]). *Let v_ε be the solution of the problem (5)-(7). Then v_ε converges to the solution v of the HJB equation (3)-(4) as $\varepsilon \to 0$ in a domain $\mathscr{D}$ defined by $\mathscr{D} := \{(t,x) : t_0 < t < t_f$ and $|x-a| < r(t)\}$, where a is the center of Ω and*

$$r(t) = r_0 + \int_{t_0}^{t} C(\tau, r(\tau))d\tau,$$
$$C(t,r(t)) := \sup_{(t_0,t)\times B(a,r(t))\times \mathscr{U}} |\mathbf{f}(t,x(t),u(t))|,$$

where $B(a,r(t))$ is an open ball of radius $r(t)$ around the point a and r_0 is the radius of Ω.

As the theorem infers, solving the optimal control problem (1)-(2) is then equivalent to solving the set of equations (5)-(7) for sufficiently small values of ε. Now, substituting v_ε by v, problem (5)-(7) as a second order PDE can be restated as

$$v_t + \varepsilon\nabla^2 v + \nabla v \cdot \mathbf{f}(t,x(t),u^*(t)) + L(t,x(t),u^*(t)) = 0, \ (t,x) \in [t_0,t_f) \times \tilde{\Omega}, \tag{8}$$
$$u^*(t) = \arg\inf_{u(\cdot)\in\mathscr{U}} \{\nabla v \cdot \mathbf{f}(t,x(t),u(t)) + L(t,x(t),u(t))\}, \tag{9}$$

where $u^*(t)$ is the optimal control and the terminal and boundary conditions are

$$v(t_f,x(t_f)) = h(t_f,x(t_f)), \quad x \in \tilde{\Omega},$$
$$v(t,x(t)) = g(t,x(t)), \quad (t,x) \in [t_0,t_f) \times \partial\tilde{\Omega}.$$

Equation (8) is known as the convection-diffusion equation. Furthermore, equations (8) and (9) are required to be solved recurrently due to the fact that the two unknown functions $v(t,x(t))$ and $u^*(t)$ are

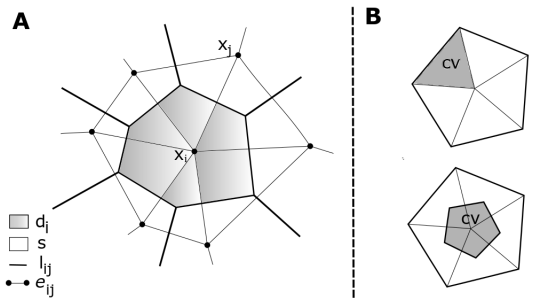

Figure 1: **A** Illustration of a considered CV and its different elements. **B** The top shows a cell-centered CV and the bottom is a node-centered CV.

coupled with each other. To do so, first, the aforementioned boundary conditions will be employed for solving (9) and the result will be then used for obtaining the next $v(t,x(t))$ through equation (8). The backward procedure will start from the terminal time t_f and continue till the initial time step which results in having the optimal control function $u^*(t)$ and the performance index value $v(t,x(t))$. However, the most important challenge is to solve the convection-diffusion equation in each time step.

3 Numerical study

In the following, we present a numerical approach for solving equation (8). To this, first the FVM is applied and then, the AMG approach is exploited as an accelerator to the process.

3.1 Finite volume method

In order to apply the FVM, first, the domain $\tilde{\Omega}$ is meshed into non-overlapping cells consisting of nodes $X = \{x_i | i = 1,\dots,N\}$ and faces $E = \{e_{i,j} | i,j = 1,\dots,M\}$ (this step is known as the primal mesh). Next, a partitioning of Control Volumes (CVs), denoting by d_i for $i = 1,2,\dots,N$, is considered as a dual mesh with a cell-centered scheme where CVs and cells coincide with each other and the variables of interest are the cell centers. This is in contrast to a node-centered partitioning where the variables of interest are the nodes themselves. Figure 1 illustrates the concepts of primal and dual meshes. Also assume $\tilde{X} = \{x_i | i = 1,\dots,\tilde{N}\}$ to be the nodes which are not on the boundary ($\partial\tilde{\Omega}$) for a value $\tilde{N} < N$. Since the FVM follows the conservation law, the neighbours of a node have important role in the full system of equations and every node is identified by its neighbours. Thus, let $I_i = \tilde{I}_i \cup \bar{I}_i$ be the index set of all x_i's neighbours consisting of interior and boundary nodes such that $\tilde{I}_i = \{j : j \in I_i, x_j \notin \partial\tilde{\Omega}\}$ and $\bar{I}_i = \{j : j \in I_i, x_j \in \partial\tilde{\Omega}\}$. Then, integrating equation (8) over an arbitrary control volume d_i for $i \in I_i$, we get

$$-\int_{d_i} \frac{\partial v}{\partial t} dx - \int_{d_i} \varepsilon \nabla^2 v dx - \int_{d_i} \nabla v \cdot \mathbf{f} dx = \int_{d_i} L dx,$$

that using the midpoint rule for integration and also the Divergence Theorem [25], one can obtain

$$-\frac{\partial v_i}{\partial t}|d_i| - \int_{\mathsf{s}} (\varepsilon \nabla v + v\mathbf{f}) \cdot \mathbf{n} \, d\mathsf{s} + v_i \int_{\mathsf{s}} \mathbf{f} \cdot \mathbf{n} \, d\mathsf{s} = L_i |d_i|,$$

where $\mathbf{n}$ denotes the outward unit normal and s is the surface of the CV (i.e. ∂d_i). Now, based on [34], using the exponential fitting for approximating the integral parts, we get

$$-\frac{\partial v_i}{\partial t}|d_i| - \sum_{j\in I_i} \varepsilon \frac{|l_{ij}|}{|e_{ij}|}\{\mathbf{B}(-\frac{f_{ij}(u^*)|e_{ij}|}{\varepsilon})v_j - \mathbf{B}(\frac{f_{ij}(u^*)|e_{ij}|}{\varepsilon})v_i\} + \sum_{j\in I_i} v_i f_{ij}(u^*)|l_{ij}| = L_i|d_i|,$$

where $\mathbf{B}(z)$ and f_{ij} are given by

$$\mathbf{B}(z) = \begin{cases} \dfrac{z}{e^z-1}, & z \neq 0, \\ 1, & z = 0, \end{cases}$$

$$f_{ij}(u^*) = \frac{1}{2}\{\mathbf{f}(t,x_i,u^*(t,x_i)) + \mathbf{f}(t,x_j,u^*(t,x_j))\}.\mathbf{e}_{ij},$$

with the unit vector $\mathbf{e}_{ij}$ from x_i to x_j. Finally, we use the first-order backward Euler method for time discretization with step size $\Delta t_k = t_k - t_{k-1} < 0$ for $k = 1,2,\ldots,t_{max}$ and t_{max} as the number of steps. This, in each time step t_k, results in solving the following system of equations with respect to $v^k = (v_1^k, v_2^k, \ldots, v_{\tilde{N}}^k)$

$$(C^k + A^k(u^*) + G^k)v^k = b^k + C^k v^{k-1}, \tag{10}$$

where G and C are positive diagonal matrices and for $i = 1,\ldots,\tilde{N}$:

$$a_{ii}{}^k = \sum_{j\in I_i} \varepsilon \frac{|l_{ij}|}{|e_{ij}|}\mathbf{B}(\frac{f_{ij}(u^*)|e_{ij}|}{\varepsilon}),$$

$$a_{ij}{}^k = \begin{cases} -\varepsilon \dfrac{|l_{ij}|}{|e_{ij}|}\mathbf{B}(-\dfrac{f_{ij}(u^*)|e_{ij}|}{\varepsilon}), & j \in \tilde{I}_i, \\ 0, & j \neq i,\ j \notin \tilde{I}_i, \end{cases}$$

$$b_i{}^k = L_i|d_i| + \sum_{j\in \tilde{I}_i} \varepsilon \frac{|l_{ij}|}{|e_{ij}|}\mathbf{B}(-\frac{f_{ij}|e_{ij}|}{\varepsilon})v_j^{k-1},$$

$$g_{ii}{}^k = \sum_{j\in I_i} f_{ij}(u^*)|l_{ij}|, \qquad c_{ii}{}^k = \frac{|d_i|}{\Delta t_k}.$$

Defining $\mathbf{A}^k := C^k + A^k(u^*) + G^k$ and $\mathbf{b}^k := b^k + C^k v^{k-1}$, system $\mathbf{A}^k v^k = \mathbf{b}^k$ in (10) with the aforementioned equations, in each step k, provides an iterative algorithm for solving the convection-diffusion equation (8). Observe the bold notation preserved for the matrix $\mathbf{A}$ and vector $\mathbf{b}$. Next, given the value of v^k and also using equation (9), the optimal control u^* is computed by:

$$u_i^* = \arg\inf_{u\in\mathscr{U}}\{\nabla v_i^k.\mathbf{f}(t_k,x_i,u_i) + L(t_k,x_i,u_i)\}, \quad i = 1,\ldots,\tilde{N}.$$

The algorithm initiates at the value $v^0 = v(t_f, x(t_f))$ and performs for each step k till achieving $v(t,x(t))$ as the optimal value of the convection-diffusion equation (8) and consequently the approximate solution of HJB equation (3) when ε is sufficiently small. The method performs well in solving the optimal control problems, however, as the complexity of the problem increases, a larger $\tilde{N}$ value is required to provide reliable results. Note that the main challenge is to solve system $\mathbf{A}^k v^k = \mathbf{b}^k$ in each iteration more effectively. We further address this issue by applying an AMG method to alleviate the effects of large

$\tilde{N}$ values. This is efficiently practicable since matrix $\mathbf{A}^k$ obtained from the FVM in each step k is an *M-matrix* that has well-behaved properties such as being *sparse* and *weakly diagonally dominant* [34, 37]. In addition, observe that each row of matrix $\mathbf{A}^k$ (denoted, for e.g., by $\mathbf{A}^k_i$) for $i = 1, \ldots, \tilde{N}$ is coupled with its corresponding variable v^k_i and node x_i. This creates a one-to-one relation between the considered mesh and matrix $\mathbf{A}^k$.

Definition 1. *Two nodes (variables) x_i (v_i) and x_j (v_j) are said to be connected if $a_{ij} \neq 0$. Also, given a threshold value $0 < \theta \leq 1$, the node x_i (variable v_i) strongly depends on (or is strongly connected to) x_j (v_j) if*

$$-a_{ij} \geq \theta \max_{a_{ik}<0} \{|a_{ik}|\}.$$

If x_i (v_i) is connected to x_j (v_j), but not strongly connected, it is said to be weakly connected.

In the following and throughout the paper, we use index i for referring to the variable v_i or its node x_i.

3.2 Algebraic multigrid method

Consider the system $\mathbf{A}^k v^k = \mathbf{b}^k$ in (10) that needs to be solved in each time step and omit the k index for the sake of simplicity. Furthermore, let S be the smoother that, in general, can be any arbitrary relaxation method such as Gauss-Seidel or Jacobi. AMG approaches use the concepts of the so-called *fine* and *coarse* levels, which are respectively the states of using all variables and a subset of variables in the computations, and vary the number of selected variables from fine to coarse to reduce the computations while maintaining the accuracy. Here, since we obtained a weakly diagonally dominant M-matrix $\mathbf{A}$ in (10), we further employ the Classical Algebraic Multigrid (CAMG) algorithm introduced by Ruge and Steuben [36]. In the case that the resultant matrix $\mathbf{A}$ is symmetric, geometric multigrid (or simply multigrid) approaches can be applied [39–41]. These approaches leverage the information of the grid space and construct a procedure in which the number of used grids reduces by moving from a fine level to a coarser one and vice versa. In this manner, the amount of computations decreases when $\mathbf{A}$ moving to the coarser levels. On the contrary, the CAMG requires no explicit knowledge of the problem geometry and only uses the information of matrix $\mathbf{A}$ to extract a submatrix of it. CAMG can be therefore seen as a more general approach since it also subsumes non-symmetric matrices and only takes the assumption of $\mathbf{A}$ being M-matrix. We preserve $\mathbf{A}^l$ and $\mathbf{A}^{l-1}$ to show respectively the states of matrix $\mathbf{A}$ in the fine and coarse levels respectively, where $\mathbf{A}^{l-1}$ is a submatrix of $\mathbf{A}^l$. In this manner, we introduce levels $l = 0, 1, \ldots, l_{max}$ with matrices $\mathbf{A}^0 \subset \mathbf{A}^1 \subset \cdots \subset \mathbf{A}^{l_{max}}$ where $\mathbf{A}^{l_{max}} = \mathbf{A}$ is the finest and $\mathbf{A}^0$ is the coarsest matrix.

In general, following steps will be applied in a two-grid approach for the $\mathbf{A}v = \mathbf{b}$. First, assume $\kappa_1, \kappa_2 > 0$ be two constants, then:

• **Pre-Smoothing:** is to apply κ_1 iterations of S on $\mathbf{A}^l v = \mathbf{b}^l$ with $v^{(0)}$ as the initial guess and computing $v^{(\kappa_1)}$.

• **Coarse-Grid Correction:** is first computing the *residual* $r^l = \mathbf{A}^l e^l$ where $e^l = v - v^{(\kappa_1)}$ and then, obtaining the *residual restriction* and the *Galerkin operator* as $r^{l-1} = \mathbf{R} r^l$ and $\mathbf{A}^{l-1} = \mathbf{R}\mathbf{A}^l\mathbf{P}$ respectively. Here $\mathbf{R}$ and $\mathbf{P}$ are the so-called *restriction* and *prolongation* operators. The restriction operator transfers the residual r^l from the fine level to the coarse level and contrariwise, the prolongation operator interpolates the residual r^l from the coarse level to the fine (in practice we set $\mathbf{R} = \mathbf{P}^T$). Figure 2 illustrates the restriction and prolongation geometry on a mesh grid scheme.

Next, $\mathbf{A}^{l-1} e^{l-1} = r^{l-1}$ is solved to find the coarse error e^{l-1} that is later transferred by the prolongation

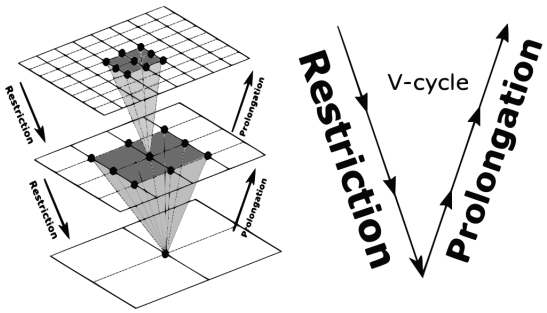

Figure 2: Geometric representation of the restriction and prolongation (interpolation) operators are illustrated in a V-cycle. In this example, in each level, 9 nodes are transferred into 1 node of the coarser level using the restriction operator. Also the procedure will be applied vice versa from a coarse level to a fine level employing the prolongation operator.

operator as $\tilde{e} = \mathbf{P}e^{l-1}$. Finally, the value $v^{(\kappa_1)}$ will be corrected by setting $v^{(\kappa_1+1)} = v^{(\kappa_1)} + \tilde{e}$. This is due to the fact that $v = v^{(\kappa_1)} + e^l$ is the exact solution and e^l is approximated by $\tilde{e}$. In short, the *coarse-grid correction operator* can be presented as

$$K := I^l - \mathbf{P}(\mathbf{A}^{l-1})^{-1}\mathbf{R}\mathbf{A}^l,$$

where I^l is the identity matrix that its dimension is equal to the number of variables in level l.
• **Post-Smoothing:** is to apply κ_2 iterations of S on $\mathbf{A}^l v = \mathbf{b}^l$ with $v^{(\kappa_1+1)}$ as the initialization.

We emphasize that the aforementioned steps represent a V-cycle of the CAMG (in analogy, one can also define a W-cycle and examine the differences between the two approaches in solving the above system of equations, see, e.g., [16] for further information). In addition, this procedure can be further generalized from a two-grid to a multi-grid method by recursively setting the current state as the fine level and the reduced one as the coarse.

Notably, the main part of the AMG approaches is the coarse-grid correction that aims to attenuate the error of the system drastically in a V-cycle and hence accelerate the convergence. Observe that after the pre-smoothing step, it is required to find the prolongation and restriction operators in order to obtain the coarse matrix $\mathbf{A}^{l-1}$. In this regards, first, the elements of matrix $\mathbf{A}^l$ should be divided into two groups of C^l and F^l, where C^l represents those variables in the coarse level (C-variables) and F^l is the complementary set (F-variables); i.e., $\mathbf{A}^l = F^l \cup C^l$. To this, consider a threshold value $0 < \theta \leq 1$ and, for each variable i, let the set of strong couplings to be $\Gamma_i^l = \{j \in \tilde{I}_i^l \,|\, v_j \text{ strongly connected to } v_i\}$. Further, we define λ_i^l as the measure of importance for each variable i by:

$$\lambda_i^l = |\Gamma_i^l \cap \tilde{\Omega}^l| + 2|\Gamma_i^l \cap F^l|, \quad i \in \Gamma_i^l,$$

where $\tilde{\Omega}^l$ is the set of all variables in the fine level. The C/F-splitting can be applied by denoting the variable i with the highest λ_i value to the set C^l and all the variables $j \in \Gamma_i^l$ to the set F^l. Observe that the variables i and j will be assigned to the sets C^l and F^l only if they are not assigned to a set before and the process will continue till all variables are included to one of the C^l or F^l sets. Given such a C/F-splitting,

Algorithm 1 Set up phase algorithm: SetUp $(\mathbf{A}^l, v)$.

$n \leftarrow$ length(v)
compute $\{\Gamma_i^l\}$ *and* $\{\lambda_i^l\}$
$F^l \leftarrow \emptyset, \quad C^l \leftarrow \emptyset.$
while $|C^l \cup F^l| < n$ **do**
  $i \leftarrow$ index max$\{\lambda_i^l\}$
  $C^l \leftarrow C^l \cup \{i\}$
  for $j = 1, \ldots, n$ **do**
    if $j \in \Gamma_i^{l^T}$ and $j \notin F^l$ and $j \notin C^l$ **then**
      $F^l \leftarrow F^l \cup \{j\}$
    end if
  end for
  update $\{\lambda_i^l\}$
end while
return P

the interpolation $e^l = \mathbf{P}e^{l-1}$ is then given by:

$$e_i^l = (\mathbf{P}e^{l-1})_i = \begin{cases} e_i^{l-1} & \text{if} \quad i \in C^l, \\ \sum_{k \in P_i^l} w_{ik} e_k^{l-1} & \text{if} \quad i \in F^l, \end{cases} \tag{11}$$

where $P_i^l \subset C^l$ is called the set of *interpolatory variables* and, here, we let

$$P_i^l := C^l \cap \Gamma_i^l, \quad w_{ik} := -\frac{\alpha_i a_{ik}}{a_{ii}}, \quad \text{and} \quad \alpha_i := \frac{\sum_j a_{ij}}{\sum_k a_{ik}}, \; j \in \tilde{I}_i^l, \; k \in P_i^l. \tag{12}$$

Finally the Galerkin operator $\mathbf{R}\mathbf{A}^l\mathbf{P}$ will be constructed using the interpolation operator $\mathbf{P}$ and restriction operator $\mathbf{R}$, where $\mathbf{R}$ is the transpose of $\mathbf{P}$ ($\mathbf{R} = \mathbf{P}^T$). This phase is known as the *set up phase* and summarized in Algorithm 1. Next, the so-called *solution phase* (presented in Algorithm 2) will be concluded using the interpolation and restriction operators and applying the correction on the error that terminates the CAMG process. At the end, we present the summary of the proposed method for solving the convection-diffusion equation (8) in Algorithm 3.

3.3 Convergence analysis

To assess the convergence of the proposed method, it is only required to examine the convergence of the CAMG algorithm for the system of $\mathbf{A}v = b$ in each iteration. Besides the convergence of the CAMG, we expect to achieve a good estimation of the function $v(t, x(t))$ for sufficiently large values of t_{max}. Now, let S be the relaxation operator employed in the CAMG, the smoothing property of S can be defined as:

Definition 2 (Smoothing Property)**.** *Let* $\mathbf{A} \in \mathbb{R}^{n \times n}$ *be a symmetric positive definite matrix and D represents its diagonal matrix. S satisfies the smoothing property w.r.t.* $\mathbf{A}$ *if*

$$||Se||_1{}^2 \leq ||e||_1{}^2 - \sigma||e||_2{}^2, \qquad (\sigma > 0), \tag{13}$$

Algorithm 2 V-cycle CAMG with γ levels in each time step: CAMG $(\mathbf{A}^l, \mathbf{b}^l, v^{(0)}, \kappa_1, \kappa_2, \gamma)$.

Require: $\mathbf{A}^l, \mathbf{b}^l, v^{(0)}, \kappa_1, \kappa_2$ and γ,
  $\mathbf{P} \leftarrow$ SetUp $(\mathbf{A}^l, v)$
  while $\gamma \geq 0$ **do**
    $v^{(\kappa_1)} \leftarrow S(\mathbf{A}^l, \mathbf{b}^l, v^{(0)}, \kappa_1)$
    $r^l \leftarrow \mathbf{b} - \mathbf{A}^l v^{(\kappa_1)}$
    $r^{l-1} \leftarrow \mathbf{R} r^l$
    $\mathbf{A}^{l-1} = \mathbf{R}\mathbf{A}^l\mathbf{P}$
    if $\gamma == 0$ **then**
      $e^{l-1} \leftarrow (\mathbf{A}^{l-1})^{-1} r^{l-1}$
    else
      $\mathbf{A}^l \leftarrow \mathbf{A}^{l-1}$
      $\mathbf{b}^l \leftarrow r^{l-1}$
      $v \leftarrow e^{l-1}$
      $\gamma \leftarrow \gamma - 1$
      CAMG $(\mathbf{A}^l, \mathbf{b}^l, 0, \kappa_1, \kappa_2, \gamma)$
    end if
    $v^{(\kappa_1+1)} \leftarrow v^{(\kappa_1)} + \mathbf{P} e^{l-1}$
    $v \leftarrow S(\mathbf{A}^l, \mathbf{b}^l, v^{(\kappa_1+1)}, \kappa_2)$
  end while
  return v

Algorithm 3 Proposed approach.

Require: $v^0 = h(t_f, x(t_f))$,
  $u^0 = \arg\inf\{\nabla v^0 \cdot \mathbf{f}(t_f, x, u) + L(t, x, u)\}$
  while $k < t_{max}$ **do**
    $\mathbf{A}^k \leftarrow \mathbf{A}(u^k)$
    $v^k =$ CAMG $(\mathbf{A}^k, \mathbf{b}^k, v^k, \kappa_1, \kappa_2, \gamma)$
    $u^k = \arg\inf\{\nabla v^k \cdot \mathbf{f}(t_k, x, u) + L(t_k, x, u)\}$
  end while
  return u, v

holds with σ *being independent of* e *and where* $||Se||_1{}^2 = \langle \mathbf{A}Se, Se\rangle$, $||e||_1{}^2 = \langle \mathbf{A}e, e\rangle$ *and* $||e||_2{}^2 = \langle D^{-1}\mathbf{A}e, \mathbf{A}e\rangle$ *with* $\langle ., .\rangle$ *denoting the inner product.*

The inequality (13) implies that S will be inefficient in reducing the error component $||e^l||_1{}^2 \gg ||e^l||_2{}^2$. Such an error component is known as *algebraically smooth error* and must be distinguished and reduced effectively in the coarse-grid correction since most iterative methods show slow convergence for these error components.

Since the obtained matrix $\mathbf{A}$ in each iteration is a weakly diagonally dominant M-matrix, there are different studies showing the convergence of CAMG where, for e.g., a two-grid approach and only one smoothing step is performed per cycle. Such investigations, however, mainly assume the M-matrix $\mathbf{A}$ to be symmetric. Although this assumption is necessary for the simplicity of the discussion, it can be shown

that the results are also valid for non-symmetric cases. This, however, requires a comprehensive study on AMGs that is far from our goal in this paper. Instead we only present the following theorem stating the convergence of CAMG for symmetric cases and further refer the readers to see [29, 30, 40, 41] for its proof and other related studies.

Theorem 2. *[40] Let* $\mathbf{A}$ *be a symmetric and weakly diagonally dominant M-matrix and S be the relaxation method satisfying the smoothing property* (13)*. Also assume a given* C/F*-splitting such that for each* $i \in F$ *there exist a set* $P_i^l \subseteq C \cap \tilde{I}_i^l$ *and a fixed* $\tau \geq 1$ *where*

$$\sum_{k \in P_i^l} |a_{ik}| \geq \frac{1}{\tau} \sum_{j \in \tilde{I}_i^l} |a_{ij}|.$$

Then, using interpolation (11) *and weights* (12) *for each* $e^l \in \mathscr{R}(K)$ *(where* $\mathscr{R}(K)$ *denotes the range of* K*), following inequality holds:*

$$\langle \mathbf{A}e^l, e^l \rangle \leq \tau \langle D^{-1}\mathbf{A}e^l, \mathbf{A}e^l \rangle.$$

In addition, the two-level approach using one relaxation step for post-smoothing converges at a rate which only depends on τ *and not on the given matrix dimensions in each level. Meaning that:*

$$\langle \mathbf{A}SK, SK \rangle^{1/2} \leq \sqrt{1 - \sigma/\tau}.$$

The theorem ensures that the properties of matrix $\mathbf{A}$ are sufficient for the convergence of the CAMG algorithm as long as the considered relaxation method is convergent. In this theorem, these properties are listed as being a symmetric M-matrix and also weakly diagonally dominant. However, as mentioned before, the theorem is in practice extendable to the non-symmetric cases. In addition, the theorem indicates that the rate of convergence depends on a scalar and *not* on the size of matrix $\mathbf{A}$ in each level l. This is of utmost importance for non-linear and high-dimensional problems since higher values of $\tilde{N}$ are required for the algorithm to yield reliable results. In practice, most of the relaxation methods require many iterations for convergence that increases both computations and running time of the algorithm. Moreover, their number of iterations scales up exponentially with the number of nodes $\tilde{N}$. The number of iterations required for convergence of the CAMG in each level, however, bounded from above to a fixed value. That is, increasing the value of $\tilde{N}$ does not affect the convergence rate of the algorithm (see Table 2.1 in [20] for more information). For our experiments here, we observed a linear increase of the iterations (required for convergence) for the proposed approach when CAMG is applied and an exponential increase when only a relaxation method is used without CAMG. Importantly, we emphasize that a careful implementation of the proposed algorithm (specially the set up phase) is required to obtain the desired acceleration. Moreover, considering the run-time of the algorithm, we mention that AMGs are often considered less effective in comparison to geometric multigrids and some other acceleration approaches such as GMRES and/or polynomial acceleration methods [42]. Nonetheless, reducing the number of computations and also the run-time of the AMGs are still an ongoing research direction and many studies have been devoted to develop new algorithms in order to further speed up the current framework of AMGs (see, e.g., [12, 15, 20, 42, 45]).

Table 1: The optimal value of $v(t,x(t))$ at different given initial points. The results correspond to $\tilde{N} = 2^{11}$.

(t_0, x_0)	Exact	Proposed method
(0.5, 0)	-0.25	-0.2502
(0.5, -0.5)	-1.002	-1.005
(0.25, 0.25)	-1.004	-1.002
(0, 0.5)	-2.253	-2.258
(0.75, 0.4)	-0.4243	-0.4256

4 Numerical examples

In this section, we investigate the reliability of the proposed method on numerical examples. For all the examples below, the diffusion coefficient ε and also the threshold value θ of the CAMG are set to be 10^{-10} and 0.25 respectively. Moreover, the two well-known relaxation operators of Jacobi and Gauss-Seidel and a 3-level CAMG approach are applied. Also, following [17, 18], we further apply two pre- and two post-smoothing steps. The stopping criteria for the algorithm is also considered to be $\|v^{\text{new}} - v^{\text{old}}\|_\infty < 10^{-6}$. Further a uniform mesh with different number of nodes is considered and the Matlab software on a 2.3 GHz Dual-Core Intel Core *i*5 laptop with 8 GB RAM is employed for the implementation. Moreover, we here follow a common procedure (similar to, e.g., [4, 23]) and report the average number of V-cycles used for the solving $\mathbf{A}v = b$ (averaging over different time steps: we here used the first 5 iterations to compute the average) alongside with the averaged solution time of the AMG-FVM for each of the examples below. These values are further compared with the cases where CAMG is not applied and only a relaxation method is used.

Example 1 ([44])**.** Consider the following optimal control problem and let $\Omega = [-1,1]$.

$$\begin{aligned} \min_{u} \quad & -x^2(1), \\ s.t. \quad & \dot{x}(t) = u(t), \quad t_0 \le t \le 1, \\ & x(t_0) = x_0. \end{aligned}$$

Note that here t_0 and x_0 are not fixed values.

The HJB equation for this problem can be presented as

$$\begin{aligned} v_t + \inf_u \{v_x u\} = 0, \\ v(1,x(1)) = -x^2(1), \end{aligned}$$

which its optimal function is given by $v(t,x(t)) = -(|x| + (1-t))^2$.
Although the problem seems to be a toy example at first glance, the optimal function of $v(t,x(t))$ is not differentiable which makes it almost impossible for the classical approaches to obtain a reliable result. On the contrary, the proposed approach can reliably approximate the optimal function. To this, we let $\tilde{\Omega} = [-3,3]$ and $\Delta t_k = -0.0025$ and applied the proposed algorithm for solving the corresponding convection-diffusion equation of this example. The obtained values of the performance index $v(t,x(t))$ for different initial points are then presented in Table 1 where we also compare the results with their

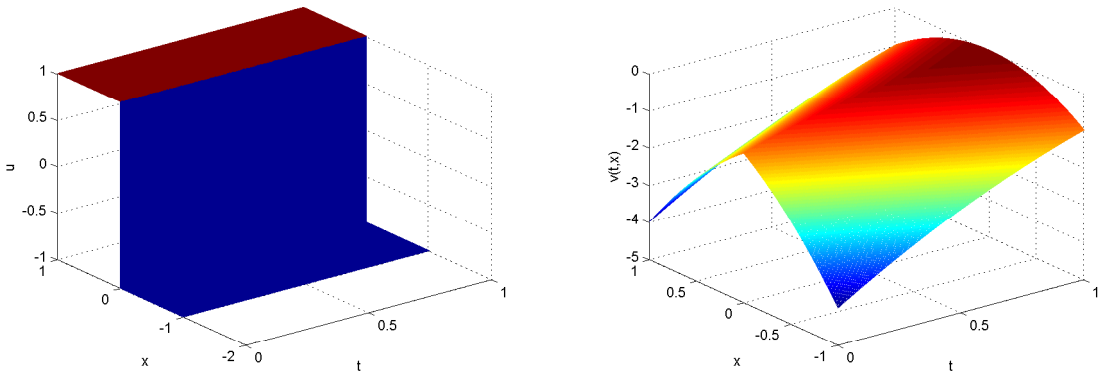

Figure 3: Surface plots of computed control (left) and optimal value of $v(t,x(t))$ (right) for different initial points. See [19, 44] for a comparison of the images.

Table 2: The average number of iterations (for AMG-FVM this corresponds to the number of V-cycles) and also the average CPU time used for solving system of $\mathbf{A}v = b$ in each time step t_k for different number of nodes $\tilde{N}$. The results reveal the efficiency of the proposed method when it is cast along with the CAMG approach.

$\tilde{N}$	AMG-FVM (Jac.)		Jacobi relax.		AMG-FVM (GS)		Gauss-Seidel relax.	
	# V-cyc	Time(s)	# Iter.	Time(s)	#V-cyc	Time(s)	# Iter.	Time(s)
2^{13}	16	0.0111	63	0.0197	15	0.0126	64	0.0212
2^{14}	29	0.055	118	0.0923	26	0.0552	119	0.0622
2^{15}	57	0.1918	228	0.2834	48	0.1828	227	0.2159

corresponding exact values. As can be seen, the proposed algorithm performs well where its closed-loop property enables us to compute the value of $v(t,x(t))$ at different initial points without any further procedure. The behaviour of the function $v(t,x(t))$ and also the computed control function $u(t)$ are further illustrated in Figure 3 for different initial points.

Next, a comparison in the number of iterations and CPU time required for solving the system of $\mathbf{A}v = b$ in each time step is presented in Table 2. We considered two different relaxation methods (Jacobi and Gauss-Seidel) and performed the proposed method using these two relaxations once with the CAMG and once without it. The presented values then correspond to different numbers of $\tilde{N}$. Importantly, the set up time (coarsening time) is excluded from the CPU time corresponding to the AMG-FVM approach as this procedure has been applied only once for solving the system of $\mathbf{A}v = b$. As it can be seen, the number of iterations drastically decreases when the CAMG is applied. In addition, we observed this number of iterations to grow linearly with the number of $\tilde{N}$ in contrast to the cases where CAMG is not applied which the growth seems to be exponentially. This further indicates that the convergence of CAMG is not dependent on the mesh size. Besides, the CPU time of the proposed AMG-FVM is also observed to be less than the other solvers for all the three different values of $\tilde{N}$ that are considered here. Finally, the performance of the algorithm is observed to improve as $\tilde{N}$ increases which shows the consistency of the method.

Example 2 ([24]). For the second example, consider the electric circuit shown in Figure 4. The optimal control problem computes the current u to minimize the energy consuming in the resistor and bring the system, initially at rest ($x_1(t_0) = x_{01}$ and $x_2(t_0) = x_{02}$), to have unitary voltage across the condenser at a given T and also zero current flowing inside the inductor. Assuming that all components are ideal, the circuit equations are

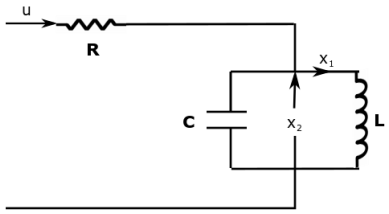

Figure 4: The electric circuit.

$$\begin{cases} \dot{x_1}(t) = \dfrac{x_2}{L}, \quad \dot{x_2}(t) = \dfrac{u - x_1}{C}, \\ x_1(t_0) = x_{01}, \quad x_2(t_0) = x_{02}, \end{cases}$$

while the performance index that should be minimized is given by

$$J = \frac{2RC^2}{T} - \frac{2RC^2}{T} x_2(T) + \int_{t_0}^{T} R\frac{u^2}{2} dt.$$

The optimal solutions for this optimal control problem in the case that $[x_1(0), x_2(0)]^T = [0,0]^T$ are observed as $u^*(t) = -2\frac{C}{T}\cos(\omega t)$, $x_1^*(t) = -\frac{1}{L\pi} t\sin(\omega t)$ and $x_2^*(t) = -\frac{1}{\pi}[\sin(\omega t) + \omega t\cos(\omega t)]$ and $v(t, x(t)) = \alpha_1(t)x_1(t) + \alpha_2(t)x_2(t) + \alpha_3(t)$, where

$$\alpha_1(t) = \frac{2RC}{\pi}\sin(\omega t),\ \alpha_2(t) = \frac{2RC^2}{T}\cos(\omega t),\ \alpha_3(t) = \frac{RC^2}{T}[1 + \frac{1}{T}(t + \frac{\sin(\omega t)\cos(\omega t)}{\omega})].$$

Here, we set $T := \pi/\omega$, $\omega := \sqrt{1/LC}$ and $L = R = C = \frac{1}{2}$ to apply the proposed method for solving the problem. Further, we aim to solve the problem on the interval $\Omega = [-1,1] \times [-1,1]$ and let $\tilde{\Omega} = [-2,2] \times [-4,4]$, $\Delta t_k = -\frac{T}{100}$ and $(h_{x_1}, h_{x_2}) = (0.05, 0.05)$. The optimal control $u^*(t)$ and also surface $v(t, x_1(t), x_2(t))$ on spatial domain $\tilde{\Omega}$ and initial time $t_0 = 0$ are presented in Figure 5. Also Figure 6 illustrates the behaviour of the optimal trajectories $x_1^*(t)$ and $x_2^*(t)$. In addition, in order to examine the performance of the proposed method, a comparison between the obtained optimal values of $v(t, x_1(t), x_2(t))$ and their exact values for different initial points is presented in Table 3. Moreover, Table 4 presents the average number of iterations and also the average CPU time used for solving the system of $\mathbf{A}v = b$ in each time step t_k. It gives a comparison of applying two different relaxation operators with or without the CAMG approach. This suggests that the CAMG approach performs well in combination with the FVM method and accelerates the performance of the proposed algorithm. Consistently, by increasing the number of nodes, the number of iterations required for solving the system of $\mathbf{A}v = b$ does not change drastically, which illustrates the reliability of the method. Regarding the CPU time, however, the implemented AMG-FVM has seen to require a bit higher amount of time for the convergence when the Jacobi relaxation is applied (note that for the Gauss-Seidel this is not the case and AMG-FVM performs faster).

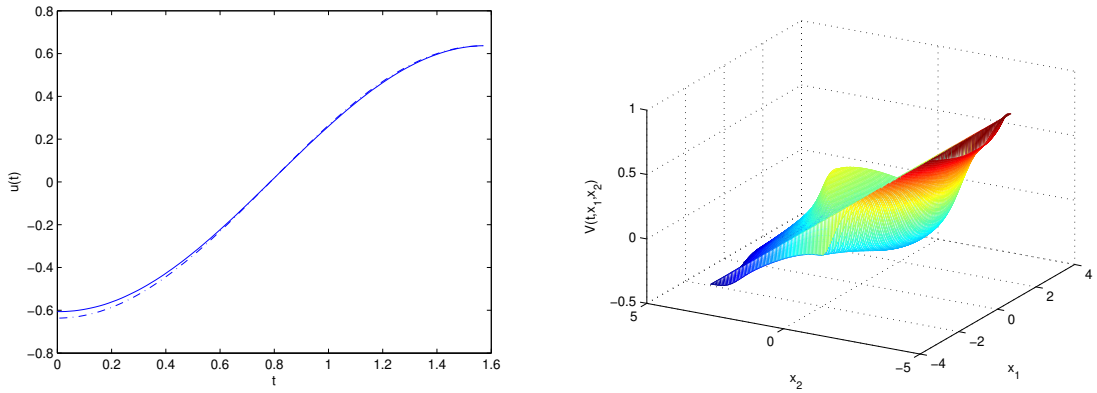

Figure 5: The left figure illustrates the behaviour of the optimal control $u^*(t)$. The line $-$ corresponds to the proposed method and the line $\cdot-$ to the exact function. The right figure presents the surface of the function $v(t,x_1(t),x_2(t))$ at initial time $t_0 = 0$.

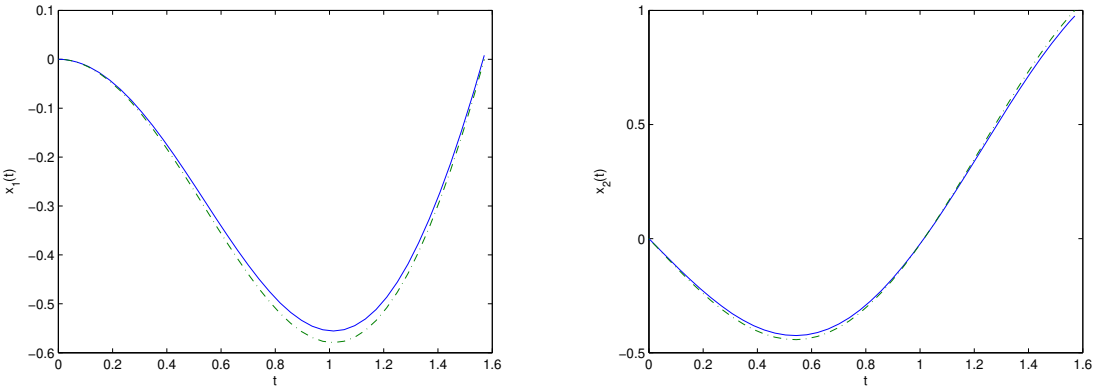

Figure 6: Behaviour of the optimal trajectories $x_1^*(t)$ (left) and $x_2^*(t)$ (right). The line $-$ corresponds to the proposed method and the line $\cdot-$ shows the exact optimal trajectory.

Table 3: The optimal value of $v(t,x_1(t),x_2(t))$ at different given initial points.

(t_0,x_{01},x_{02})	Exact	Proposed method
(0, -1, -1)	-0.0796	-0.0717
(0, 0.5, 1)	0.2387	0.2351
(0, 1, 1)	0.2387	0.236
(0, 0, 0.5)	0.1593	0.1592

Table 4: The average number of iterations (for AMG-FVM this corresponds to the number of V-cycles) and also the average CPU time used for solving system of $\mathbf{A}v = b$ in each time step t_k for different number of nodes. The results reveal the efficiency of the CAMG in reducing the number of iterations used in the algorithm. Besides, we observe that the CPU time for the solution phase of AMG-FVM is a bit higher than the classical relaxation methods.

(h_{x_1}, h_{x_2})	AMG-FVM (Jac.)		Jacobi relax.		AMG-FVM (GS)		Gauss-Seidel relax.	
	# V-cyc	Time(s)	# Iter.	Time(s)	#V-cyc	Time(s)	# Iter.	Time(s)
$(0.05, 0.05)$	10	0.079	38	0.0339	9	0.0722	36	0.5363
$(0.03, 0.03)$	14	0.3487	58	0.099	14	0.3547	56	5.5220
$(0.025, 0.025)$	17	0.6468	68	0.2278	16	0.6186	65	11.3382

Table 5: The optimal value of $v(t, x_1(t), x_2(t))$ at different initial points. We here report two different values for our method corresponding to two different discretizations. The index (1) corresponds to $(h_{x_1}, h_{x_2}) = (0.05, 0.05)$ and index (2) corresponds to $(h_{x_1}, h_{x_2}) = (0.025, 0.025)$. We also mention that the approach presented in [34] uses a number of 321×321 nodes that (approximately) corresponds to $(h_{x_1}, h_{x_2}) = (0.0065, 0.025)$.

(t_0, x_{01}, x_{02})	Wang et al. [34]	Our method$^{(1)}$	Our method$^{(2)}$
(0, 0, 1)	0.0991	0.1124	0.1037
(0, 0.5, 0.1)	0.1253	0.1356	0.1304
(0, 0.4, -0.2)	0.04683	0.0552	0.0507

Example 3 ([34]). As the third example, consider the following optimal control problem:

$$\begin{aligned} \min_{u} \quad & \int_{t_0}^{1} ((x_1(t) - 0.1\sin(10t))^2 + 0.001u^2(t))dt, \\ s.t. \quad & \dot{x}_1(t) = x_2(t), \\ & \dot{x}_2(t) = -0.1x_2(t) - x_1(t) + u(t), \\ & x_1(t_0) = x_{01}, \\ & x_2(t_0) = x_{02}, \\ & |u(t)| \leq 1, \end{aligned}$$

where $\Omega = [0, 0.5] \times [-0.3, 1]$.

We consider $\tilde{\Omega} = [-0.9, 1.2] \times [-4.6, 3.7]$ (this is the interval that has been considered in [34] and we here assume the same interval for the sake of a better comparison), and set $\Delta t_k = -0.025$. We further solve the problem for different values of discretizations, i.e. different (h_{x_1}, h_{x_2}) values. Table 5 presents the obtained optimal value of $v(t, x_1(t), x_2(t))$ for different initial points in comparison with the results stated in [34].

Moreover, behaviour of the optimal control $u^*(t)$ at point $(0, 1)$ and also the surface $v(t, x_1(t), x_2(t))$ on the spatial domain $\tilde{\Omega}$ and initial time $t_0 = 0$ are illustrated in Figure 7. We further present the behaviour of the optimal trajectories $x_1^*(t)$ and $x_2^*(t)$ in Figure 8 and the surface of control $u(t)$ at two different time steps of 0.25 and 0.75 in Figure 9.

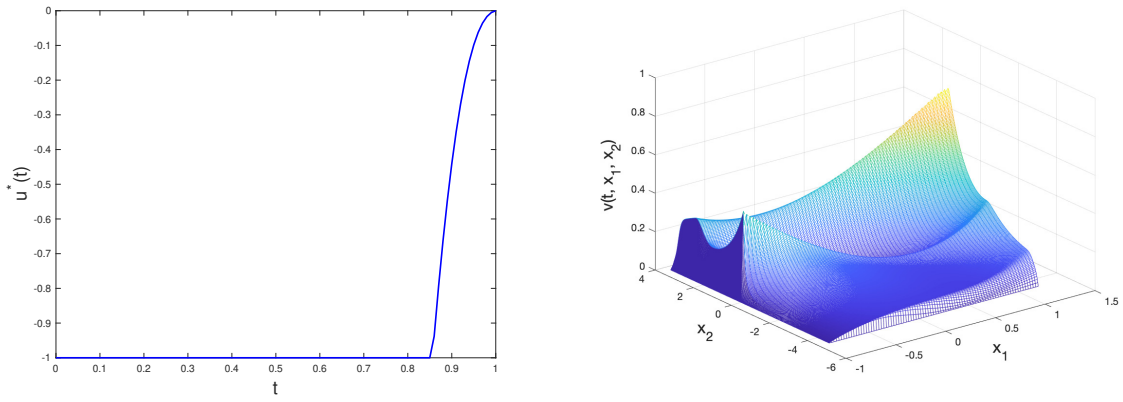

Figure 7: The left figure illustrates the behaviour of the optimal control $u^*(t)$. Also the surface of the function $v(t,x_1(t),x_2(t))$ at initial point $t_0 = 0$ is shown in the right figure.

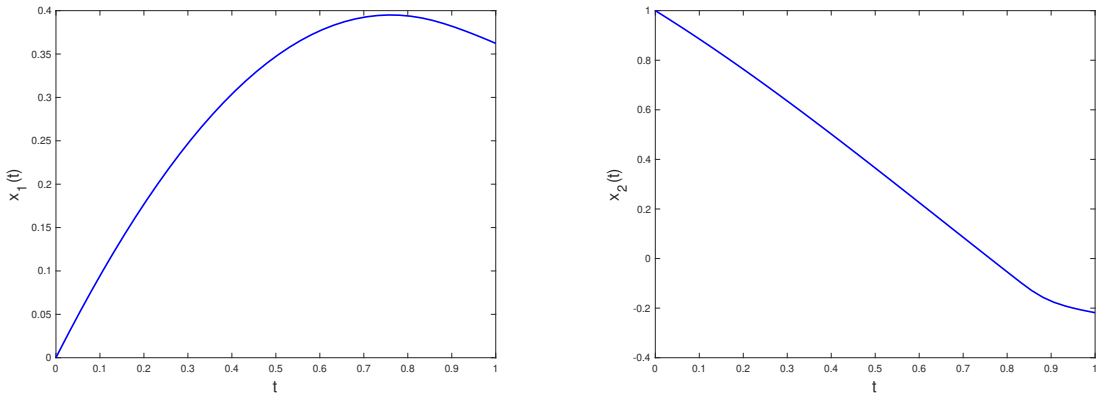

Figure 8: Behaviour of the optimal trajectories $x_1^*(t)$ (left) and $x_2^*(t)$ (right).

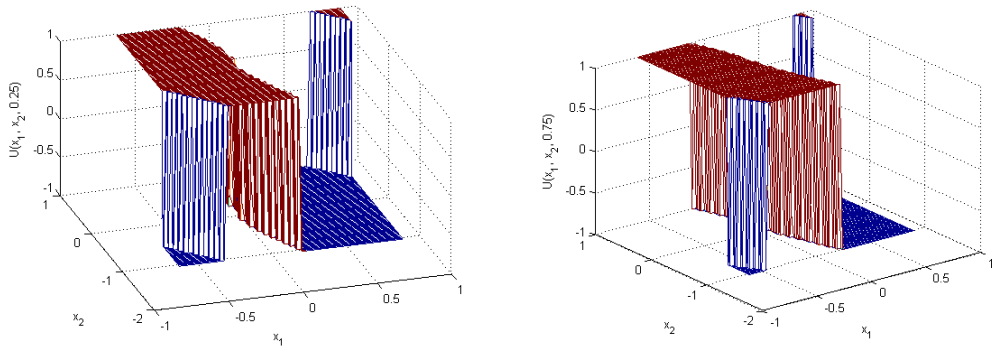

Figure 9: Surface plots of the optimal control at time points $t = 0.25$ (left) and $t = 0.75$ (right). See [34] for a comparison of the images.

Table 6: The average number of iterations (for AMG-FVM this corresponds to the number of V-cycles) and also the average CPU time used for solving system of $\mathbf{A}v = b$ in each time step t_k for different number of nodes. The results reveal the efficiency of the proposed AMG-FVM approach.

(h_{x_1}, h_{x_2})	AMG-FVM (Jac.)		Jacobi relax.		AMG-FVM (GS)		Gauss-Seidel relax.	
	# V-cyc	Time(s)	# Iter.	Time(s)	#V-cyc	Time(s)	# Iter.	Time(s)
$(0.05, 0.05)$	4	0.0311	15	0.0327	4	0.0289	15	1.4106
$(0.03, 0.03)$	6	0.1283	22	0.1336	5	0.1119	21	4.8443
$(0.025, 0.025)$	7	0.2149	27	0.1773	5	0.1885	24	10.2031

Finally, Table 6 provides the average number of iterations along with the average CPU time of the method (in solving the system of $\mathbf{A}v = b$) in each time step. The values are computed for different relaxation methods (Jacobi and Gauss-Seidel) and also for different discretization steps. As it can be seen, the CAMG approach can reliably decrease the number of iterations and also the CPU time when it is cast in the proposed method.

5 Conclusion

We investigated the HJB equation of the optimal control problems and presented a new algorithm for its solving. The algorithm is robust and performs well in combining two powerful approaches of FVM and CAMG. First, the FVM applied the viscosity approximation of the HJB equation which resulted in solving a system of equations $\mathbf{A}v = \mathbf{b}$ in each time step. Then, the CAMG was employed to accelerate the convergence of a relaxation method applied for solving $\mathbf{A}v = \mathbf{b}$. The algorithm has shown to scale linearly with the number of nodes and converge at a rate that only depends on a scaling parameter and not on the size of matrix $\mathbf{A}$. That is the number of iterations required for convergence observed to grow linearly in contrast to the case where CAMG is not applied. Besides, the CPU time of the proposed AMG-FVM has seen to be still competitive (and in most cases improved) in comparison to the classical relaxation methods even though we have considered small-scale examples with a fairly small number of nodes. Finally, numerical examples were presented to approve the theoretical results.

Acknowledgements

We thank Prof. Chin-Tien Wu from National Chiao-Tung University for assistance with programming technique and Prof. Volker Schulz from Trier University for sharing his pearls of wisdom with us. We would also like to show our gratitude to reviewers for their helpful comments and useful suggestions.

References

[1] B.D. Anderson, J.B. Moore, *Optimal Control: Linear Quadratic Methods*, Courier Corporation, 2007.

[2] M. Bardi, I. Capuzzo-Dolcetta, *Optimal Control and Viscosity Solutions of Hamilton-Jacobi-Bellman equations*, Springer Science & Business Media, 2008.

[3] A. Brandt, S. McCoruick, J. Huge, *Algebraic Multigrid (AMG) for Sparse Matrix Equations*, Cambridge Univ. Press, 1984.

[4] H. Chen, C.K. Cheng, N.C. Chou, A.B. Kahng, J.F. MacDonald, P. Suaris, B. Yao, Z. Zhu, *An algebraic multigrid solver for analytical placement with layout based clustering*, Proceedings of the 40th annual Design Automation Conference, 2003, pp. 794–799.

[5] M. Chowdhury, B.R. Kumar, *On subgrid multiscale stabilized finite element method for advection-diffusion-reaction equation with variable coefficients*, Appl. Numer. Math. **150** (2020) 576–586.

[6] M.G. Crandall, L.C. Evans, P.L. Lions, *Some properties of viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc. **282** (1984) 487–502.

[7] M.G. Crandall, P.L. Lions, *Two approximations of solutions of Hamilton-Jacobi equations*, Math. Comp. **43** (1984) 1–19.

[8] M.G. Crandall, P.L. Lions, *Viscosity solutions of Hamilton-Jacobi equations*, Trans. Amer. Math. Soc. **277** (1983) 1–42.

[9] R.C. Cruz-Rodríguez, Y.N. Skiba, D.M. Filatov, *An implicit direct unconditionally stable numerical algorithm for the solution of advection-diffusion equation on a sphere*, Appl. Numer. Math. **142** (2019) 1–15.

[10] M. Darehmiraki, M.H. Farahi, S. Effati, *A novel method to solve a class of distributed optimal control problems using Bezier curves*, J. Comput. Nonlinear Dyn. **11** (2016) 6.

[11] S. Effati, S.A. Rakhshan, S. Saqi, *Formulation of Euler–Lagrange equations for multidelay fractional optimal control problems*, J. Comput. Nonlinear Dyn. **13** (2018) 6.

[12] R.D. Falgout, P.S. Vassilevski, *On generalizing the algebraic multigrid framework*, SIAM J. Numer. Anal. **42** (2004) 1669–1693.

[13] S. Ghasemi, S. Effati, *An artificial neural network for solving distributed optimal control of the poissons equation*, Neural Process. Lett. (2018) 1–17.

[14] B. Gong, W. Liu, T. Tang, W. Zhao, T. Zhou, *An efficient gradient projection method for stochastic optimal control problems*, SIAM J. Numer. Anal. **55** (2017) 2982–3005.

[15] M. Griebel, B. Metsch, D. Oeltz, M.A. Schweitzer, *Coarse grid classification: a parallel coarsening scheme for algebraic multigrid methods*, Numer. Linear Algebra Appl. **13** (2006) 193–214.

[16] W. Hackbusch, *Multi–Grid Methods and Applications*, Vol. 4, Springer Science and Business Media, 2013.

[17] D. Han, J.W. Wan, *Multigrid methods for second order Hamilton–Jacobi–Bellman and Hamilton–Jacobi–Bellman–Isaacs equations*, SIAM J. Sci. Comput **35** (2013) S323–S344.

[18] R.H. Hoppe, *Multi-grid methods for Hamilton-Jacobi-Bellman equations*, Numer. Math. **49** (1986) 239–254.

[19] C.S. Huang, S. Wang, K. Teo, *Solving Hamilton–Jacobi–Bellman equations by a modified method of characteristics*, Nonlinear Anal. **40** (2000) 279–293.

[20] V. John, *Multigrid methods*, Tech. report, 2013.

[21] A. Kheirabadi, A.M. Vaziri, S. Effati, *Solving optimal control problem using Hermite wavelet*, Numer. Algebra Control Optim. **9** (2018) 101–112.

[22] P.L. Lions, *Generalized Solutions of Hamilton-Jacobi Equations*, Vol. 69, London Pit., 1982.

[23] H. Liu, B. Yang, Z. Chen, *Accelerating algebraic multigrid solvers on NVIDIA GPUs*, Comput. Math. Appl. **70** (2015) 1162–1181.

[24] A. Locatelli, *Optimal Control: An Introduction*, Springer Science & Business Media, 2001.

[25] F. Moukalled, L. Mangani, M. Darwish, *The finite volume method in computational fluid dynamics*, Springer, 2016.

[26] V. Naicker, K. Andriopoulos, P.G.L. Leach, *Symmetry reductions of a Hamilton–Jacobi–Bellman equation arising in financial mathematics*, J. Nonlinear Math. Phys. **12** (2005) 268–283.

[27] D.S. Naidu, *Optimal Control Systems*, CRC press, 2002.

[28] H.S. Nik, S. Effati, M. Shirazian, *An approximate-analytical solution for the Hamilton–Jacobi–Bellman equation via homotopy perturbation method*, Appl. Math. Model. **36** (2012) 5614–5623.

[29] Y. Notay, *A robust algebraic multilevel preconditioner for non–symmetric M–matrices*, Numer. Linear Algebra Appl. **7** (2000) 243–267.

[30] Y. Notay, *Algebraic analysis of two-grid methods: The nonsymmetric case*, Numer. Linear Algebra Appl. **17** (2010) 73–96.

[31] S. Radhoush, M. Samavat, M.A. Vali, *Optimal control of linear time–varying systems using the Chebyshev wavelets (a comparative approach)*, Sys. Sci. Cont. Eng. **2** (2014) 691–698.

[32] S.A. Rakhshan, S. Effati, A. Vahidian Kamyad, *Solving a class of fractional optimal control problems by the Hamilton–Jacobi–Bellman equation*, J. Vib. Control **24** (2018) 1741–1756.

[33] A. Razminia, M. Asadizadehshiraz, D.F. Torres, *Fractional order version of the Hamilton–Jacobi–Bellman equation*, J. Comput. Nonlinear Dynam. **14** (2019) 1.

[34] S. Richardson, S. Wang, *Numerical solution of Hamilton–Jacobi–Bellman equations by an exponentially fitted finite volume method*, Optimization **55** (2006) 124 – 140.

[35] S. Richardson, S. Wang, *The viscosity approximation to the Hamilton-Jacobi-Bellman equation in optimal feedback control: Upper bounds for extended domains*, J. Ind. Manag. Optim. **6** (2010) 161.

[36] J.W. Ruge, K. Stuben, *Algebraic multigrid*, Multigrid Methods, SIAM, 1987, pp. 73–130.

[37] B. Seibold, *Performance of algebraic multigrid methods for non–symmetric matrices arising in particle methods*, Numer. Linear Algebra Appl. **17** (2010) 433–451.

[38] M. Shirazian, S. Effati, *A novel successive approximation method for solving a class of optimal control problems*, CJMS **9** (2020) 124–136.

[39] K. Stuben, *Algebraic multigrid (AMG): Experiences and comparisons*, Appl. Math. Comput. **13** (1983) 419–451.

[40] K. Stuben, *Algebraic multigrid AMG: An introduction with applications*, Ph.D. Thesis, Tech. Rep., 1999.

[41] K. Stuben, *A review of algebraic multigrid*, Numer. Anal. (2001) 331–359.

[42] R. Suero, M. Pinto, C. Marchi, L. Araki, A. Alves, *Analysis of algebraic multigrid parameters for two-dimensional steady-state heat diffusion equations*, Appl. Math. Modelling **36** (2012) 2996–3006.

[43] J. Wang, P.A. Forsyth, *Numerical solution of the Hamilton–Jacobi–Bellman formulation for continuous time mean variance asset allocation*, J. Econ. Dyn. Control **34** (2010) 207–230.

[44] S. Wang, L.S. Jennings, K.L. Teo, *Numerical solution of Hamilton–Jacobi–Bellman equations by an upwind finite volume method*, J. Global Optim. **27** (2003) 177–192.

[45] P. Zaspel, *Analysis and parallelization strategies for Ruge-Stuben AMG on many-core processors*, Ph.D. Thesis, Universi Basel, 2017.