A simple multiscale method for mean field games

Haoya Li [a],[*], Yuwei Fan [a], Lexing Ying [b]

[a] *Department of Mathematics, Stanford University, Stanford, CA 94305, United States of America*
[b] *Department of Mathematics and Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305, United States of America*

ARTICLE INFO

ABSTRACT

This paper proposes a multiscale method for solving the numerical solution of mean field games which accelerates the convergence and addresses the problem of determining the initial guess. Starting from an approximate solution at the coarsest level, the method constructs approximations on successively finer grids via alternating sweeping, which not only allows for the use of classical time marching numerical schemes, but also enables applications to both local and nonlocal problems. At each level, numerical relaxation is used to stabilize the iterative process. A second-order discretization scheme is derived for higher order convergence. Numerical examples are provided to demonstrate the efficiency of the proposed method in both local and nonlocal, 1-dimensional and 2-dimensional cases.

© 2021 Elsevier Inc. All rights reserved.

1. Introduction

Mean Field Games (MFG) theory was first introduced by Lasry and Lions in [30] and by Huang, Caines and Malhamé in [29] independently for studying the asymptotic behavior of stochastic differential games as the number of agents tends to infinity. The agents in the system are assumed to be identical, and any individual agent has little impact on the outcome of the game. Each individual agent aims to minimize a certain cost, and the strategy adopted is influenced by the average of a certain function of the states of the other agents [5]. In the limit of an infinite number of agents, a typical MFG can be described by the following system

$$\begin{cases} u_t - \nu \Delta u + H(x, \nabla u) = V[m], \\ u(t=0, x) = u_0(x) + V_0[m(t=0)](x), \end{cases} \qquad x \in \Omega \subset \mathbb{R}^d; \tag{1.1a}$$

$$\begin{cases} -m_t - \nu \Delta m - \nabla \cdot (m \nabla_p H(x, \nabla u)) = 0, \\ m(t=T, x) = m_T(x), \end{cases} \qquad x \in \Omega \subset \mathbb{R}^d, \tag{1.1b}$$

where $m(t, x)$ stands for the density of the agents, and $u(t, x)$ is the cost function (or negative utility). Among the two equations, the first one (1.1a) is a forward Hamilton-Jacobi-Bellman equation (HJB). The operators $V \geq 0$ and $V_0 \geq 0$ encode the impact of other agents (via the density function). On each point of its trajectory, each agent chooses a velocity that locally minimizes its travel cost, and this leads to the appearance of the Hamiltonian $H(x, \nabla u)$. The second equation (1.1b) is a backward Kolmogorov-Fokker-Planck equation (KFP), which results from the motion of individual agents in Ω.

---

* Corresponding author.
E-mail addresses: lihaoya@stanford.edu (H. Li), ywfan@pku.edu.cn (Y. Fan), lexing@stanford.edu (L. Ying).

The existence and regularity of the solution have been studied extensively in the literature. For the stationary case, a Bernstein integral method is used to obtain regularity bounds and existence of smooth solutions in [23]. For the evolutionary case, Gagliardo-Nirenberg type inequalities are used to develop priori bounds for the solutions when the Hamiltonian is subquadratic (see [26]). The Hopf–Cole transformation gives an explicit way to study the case where the Hamiltonian is quadratic (see [13]). However, this transformation cannot be used to study superquadratic problems. In the superquadratic case, a nonlinear adjoint method uses the adjoint equation to represent solutions by integrals with respect to the adjoint variable and leads to priori bounds for the solutions (see [25]). For a systematic probabilistic approach, we refer to the reference [20], in which the MFG system is studied in a standard stochastic control framework, and a useful analytical tool–the master equation is studied. We also refer the readers to [28] and [12] for more detailed discussions of the MFGs.

In the past decade, several numerical methods for MFG have been developed. In [3], Achdou and Capuzzo-Dolcetta proposed a first-order method for MFG with a first-order upwind scheme for the spatial discretization and an implicit discretization in the time direction. This method preserves many good properties of the MFG, for example, the convergence result proved in [2], but the resulting discrete system is a large-scale nonlinear system. The Newton-Raphson method is employed for its numerical solution but the cost is relatively high. Following the finite difference discretization in [3], [11] and [10] developed variational approaches for the stationary and time-dependent MFG with local couplings. The classical V-cycle and W-cycle multigrid preconditioner are used in [4,10,6]. In [14–19], Carlini and Silva used the optimal control formula to represent the solution of the HJB equation. Using uniform partition of the time interval, they solve the HJB equation recursively by considering a discrete optimal control problem in every single time step. Then a mass conservative scheme is used for the KFP equation in an alternating way to solve the whole MFG system. The convergence property is also studied in these references. In [27], Guéant studied a particular type of MFG system with a quadratic Hamiltonian via a change of variables given in [28]. In this way, the MFG system becomes two coupled heat equations with similar source terms. It is then possible to construct a scheme that yields monotonic sequences of approximate solutions. In [8,9], the authors studied the variational form of the MFG system with congestion penalization and investigated its duality. The variational problem is related with the solution of MFG system via the Fenchel-Rockafellar duality theorem. Since the variational problem is convex, an augmented-Lagrangian method can be applied to solve it efficiently.

For the MFG system without the randomness of behavior of agents (i.e. without the Laplacian term), various methods have been developed. The works [32,21,22] apply the generalized Hopf and Lax formulas of the HJB equation and conjure the Hamiltonian in the discretization. These algorithms have the advantage that they do not suffer from the curse of dimensionality. The work in [34] addresses a type of MFG with nonlocal interaction (for example, when the operator V is nonlocal) by using kernel-based representations of mean-field interactions and feature-space expansions. Most recently, neural network type methods, such as the APAC-net in [33] and the framework based on Lagrangian method [35], have also been applied to mean field game problems, especially in high dimensions.

Contributions   The main difficulty of solving MFG systems comes from the forward-backward nature of the system. Numerical discretization of such a system leads to a nonlinear large system of equations. To overcome this difficulty, we introduce a simple alternating sweeping algorithm to decouple the two equations and to make the application of time marching schemes possible. More specifically, the method solves the HJB equation or the KFP equation alternatingly by fixing the other quantity. This allows us to solve the MFG system by solving two time-dependent PDEs, and the overall cost is decreased remarkably when the convergence is guaranteed. In order to make this work, two critical issues need to be addressed: (1) whether the algorithm converges and (2) how to choose a good initial guess?

To address the first question, we study the condition of convergence and introduce a relaxation technique for satisfying this condition. The convergence can then be guaranteed with a proper selection of the relaxation factor. Numerical simulations show that the relaxation technique is highly effective.

To answer the second question, we introduce a multiscale method. More specifically, the problem is discretized with a sequence of successively finer grids. At the coarsest level, the system is relatively small, hence it can be solved directly by, for example, the Newton iteration in [3]. At each finer level, interpolating the solution from the previous level provides a good initial guess for the system at this level, and the solution is refined using alternating sweeping of the HJB and KFP equations. This process is repeated hierarchically until one reaches the finest grid. This multiscale method not only provides a good initial guess for each level, but also dramatically accelerates the convergence of the alternating sweeping. For the discretization of the MFG system, we also introduce a second-order scheme in order to reduce the numerical error. More specifically, a Beam-Warming scheme is applied on the spatial discretization, and a Crank-Nicolson scheme is introduced on the temporal discretization. The scheme is proved to preserve the conservation of mass.

Contents   The rest of the paper is organized as follows. Section 2 provides the main description of the numerical scheme. The general framework of the alternating sweeping procedure and the relaxation method is proposed in Section 2.2, the multiscale algorithm is detailed in Section 2.3, and the second-order finite difference scheme introduced in Section 2.4. Section 3 studies the numerical performance of the proposed algorithms. Finally Section 4 concludes with some discussion for future work.

2. Numerical algorithm

2.1. Notations

To simplify the discussion, let us consider the spatial domain $\Omega = [0, 1]^d$ with the periodic boundary condition. We introduce the notations for 1-dimensional case as it is straightforward to extend to the d-dimensional case.

We partition Ω by a hierarchical uniform Cartesian mesh with step size $h^{(\ell)} = \frac{1}{N^{(\ell)}} = 2^{-\ell}$ for each level $\ell = L_0, \ldots, L$, where $L_0 \leq L$ are given positive integers. For each ℓ, the grid points are denoted as $x_i^{(\ell)} = ih^{(\ell)}$, with $i = 1, \ldots, N^{(\ell)}$. In addition, for each ℓ, the time interval $[0, T]$ is also uniformly partitioned with time step $\tau^{(\ell)} = \frac{T}{N_t^{(\ell)}} = 2^{-\ell}T$. The grid points in time are denoted as $t_n^{(\ell)} = n\tau^{(\ell)}$. One can easily extend to the case where different stepsizes are used.

For the level ℓ mesh, denote the approximations to $u(x_i^{(\ell)}, t_n^{(\ell)})$ and $m(x_i^{(\ell)}, t_n^{(\ell)})$ by $u_i^{n,(\ell)}$ and $m_i^{n,(\ell)}$, respectively. It is often convenient to abbreviate $\left(u_i^{n,(\ell)}\right)_{i=0,\ldots,2^\ell-1}$ and $\left(m_i^{n,(\ell)}\right)_{i=0,\ldots,2^\ell-1}$ as $u^{n,(\ell)}$ and $m^{n,(\ell)}$, and further denote $(u^{n,(\ell)})_{n=0,\ldots,N_t}$ and $(m^{n,(\ell)})_{n=0,\ldots,N_t}$ by $U^{(\ell)}$ and $M^{(\ell)}$, respectively. Notice that the lower-case notations u and m represent the solution on the mesh for a given time, and the upper-case notations U and M represent the whole solution on a given level.

As an example of the notations adopted, a backward Euler discretization of the 1-dimensional MFG equations (1.1) on level ℓ is of the following form, for $i = 1, \ldots, N^{(\ell)}$,

$$
\begin{cases}
\dfrac{u_i^{n+1,(\ell)} - u_i^{n,(\ell)}}{\tau^{(\ell)}} + (\mathcal{L}u^{n+1,(\ell)})_i + H(x_i^{(\ell)}, (\mathcal{D}u^{n+1,(\ell)})_i) = V[m^{n,(\ell)}]_i, \quad n = 0, \ldots, N_t^{(\ell)} - 1, \\
u_i^{0,(\ell)} = u_0(x_i^{(\ell)}) + V_0[m^{0,(\ell)}]_i;
\end{cases}
\tag{2.1a}
$$

$$
\begin{cases}
-\dfrac{m_i^{n,(\ell)} - m_i^{n-1,(\ell)}}{\tau^{(\ell)}} + (\mathcal{L}m^{n-1,(\ell)})_i - B_i(m^{n-1,(\ell)}, u^{n,(\ell)}) = 0, \quad n = N_t^{(\ell)}, N_t^{(\ell)} - 1, \ldots, 1, \\
m_i^{N_t^{(\ell)},(\ell)} = m_T(x_i^{(\ell)}),
\end{cases}
\tag{2.1b}
$$

where $\mathcal{D}$ stands for the discretization of the spatial gradient operator ∇, and $\mathcal{L}$ stands for the discretization of $-\nu\Delta$, a scalar multiple of the Laplace operator, and B is the discrete analog of $\nabla \cdot (m\nabla_p H(x, u))$. For the sake of brevity we overload the notation $\mathcal{D}$, $\mathcal{L}$, H, V and V_0 for the corresponding operators on grid functions on each level. In the paper, if the level ℓ is clear in the context, the superscript ℓ will be omitted.

2.2. Alternating sweeping

Due to the forward-backward structure of system (2.1), one cannot directly apply a time marching scheme. A simple but key observation is that if one fixes the value of M, the system (2.1a) becomes a single forward parabolic equation with an initial condition. Similarly, if we fix the value of U, the system (2.1b) becomes a backward parabolic equation with a terminal condition. In this way, we arrive at a natural way to adopt time marching schemes.

More specifically, we start from an initial guess M_{init}, and use some time marching scheme to solve (2.1a) while fixing M, then solve (2.1b) while fixing U, and repeat this process until we reach a fixed point. We call this algorithm the *alternating sweeping*, which is depicted in Algorithm 1.

Algorithm 1 Alternating sweeping for the MFG.

Input: Initial guess of the density M_{init}, tolerance ϵ
Output: Numerical solution M and U
  function ALTERNATINGSWEEPING(M_{init})
    $M \leftarrow M_{\text{init}}$, $U \leftarrow 1$;
    do
      Fixing M and solving (2.1a) gives U_{new};
      Compute $\text{err}_U = \|U_{\text{new}} - U\|/\|U\|$;
      $U \leftarrow U_{\text{new}}$;
      Fixing U and solving (2.1b) gives M_{new};
      Compute $\text{err}_M = \|M_{\text{new}} - M\|/\|M\|$
      $M \leftarrow M_{\text{new}}$;
    while $\max\{\text{err}_M, \text{err}_U\} > \epsilon$
    return M and U;
  end function

Now, we briefly analyze the alternating sweeping procedure from the viewpoint of fixed point iteration. In the following discussion, we denote M_k and U_k as the approximate solution of M and U after the k-th iteration in Algorithm 1. For the clarity of notation, we rewrite the functions in (2.1) as:

$$F(U, M) = 0, \qquad G(U, M) = 0 \tag{2.2}$$

by moving everything to the left hand side.

For $k \geq 2$, in the k-th iteration of Alternating Sweeping, we start from an approximation of m on the grid points, i.e. M_{k-1}, and then we obtain U_k from the equation $F(U_k, M_{k-1}) = 0$, and finish this step by getting M_k from $G(U_k, M_k) = 0$. For $k = 1$ we start from M_{init}. This is a variant of the usual fixed point iteration method (see for example [24]). We give a convergence result in the following proposition.

Proposition 1. *Assume that F and G are continuously differentiable, and that $F_u(U^*, M^*)$ and $G_m(U^*, M^*)$ are invertible, where U^* and M^* are the solution to the equation* (2.2)*, and*

$$
\begin{aligned}
F_u &:= \frac{\partial F}{\partial U}(U^*, M^*), \quad F_m := \frac{\partial F}{\partial M}(U^*, M^*), \\
G_u &:= \frac{\partial G}{\partial U}(U^*, M^*), \quad G_m := \frac{\partial G}{\partial M}(U^*, M^*).
\end{aligned}
\tag{2.3}
$$

Then M_k converges to M^* locally if

$$\rho(G_m^{-1} G_u F_u^{-1} F_m) < 1, \tag{2.4}$$

where ρ denotes the spectral radius, and U^k converges to U^* locally if

$$\rho(F_u^{-1} F_m G_m^{-1} G_u) < 1. \tag{2.5}$$

Moreover, the conditions (2.4) *and* (2.5) *are equivalent, and when they hold, the convergence rate of Algorithm 1 is*

$$r = \rho(G_m^{-1} G_u F_u^{-1} F_m) = \rho(F_u^{-1} F_m G_m^{-1} G_u). \tag{2.6}$$

In order to prove this proposition, we need the following lemma on the spectral radius of matrices.

Lemma 1.

$$\forall A, B \in \mathbb{C}^{n \times n}, \quad \rho(AB) = \rho(BA). \tag{2.7}$$

The proof of this lemma is given in Appendix A. Now we can prove Proposition 1.

Proof. Take the Taylor expansion of F and G at U^* and M^* we get

$$
\begin{aligned}
F(U_k, M_{k-1}) &= F(U^*, M^*) + F_u(U_k - U^*) + F_m(M_{k-1} - M^*) + o(|U_k - U^*| + |M_{k-1} - M^*|), \\
G(U_k, M_k) &= G(U^*, M^*) + G_u(U_k - U^*) + G_m(M_k - M^*) + o(|U_k - U^*| + |M_k - M^*|).
\end{aligned}
\tag{2.8}
$$

Thus we have

$$M_k - M^* = G_m^{-1} G_u F_u^{-1} F_m(M_{k-1} - M^*) + o(|M_{k-1} - M^*|). \tag{2.9}$$

Hence M_k converges to M^* locally with rate $\rho(G_m^{-1} G_u F_u^{-1} F_m)$ if (2.4) holds (see for example [24]). Similarly, U^k converges to U^* locally with rate $\rho(F_u^{-1} F_m G_m^{-1} G_u)$ if (2.5) holds.

By Lemma 1, we have $\rho(G_m^{-1} G_u F_u^{-1} F_m) = \rho(F_u^{-1} F_m G_m^{-1} G_u)$, thus the conditions (2.4) and (2.5) are equivalent, which closes the proof. □

Remark 1. One can also first solve M with fixed U and then solve U with fixed M. Due to the upper analysis, the convergence rate of the corresponding scheme keeps the same.

The upper discussion only guarantees the convergence of the alternating sweeping method when M_{init} is sufficiently close to the real solution and the condition in Proposition 1 ($\rho(G_m^{-1} G_u F_u^{-1} F_m) < 1$) is satisfied. Thus, in order to make the alternating sweeping work, one needs to address two critical issues: (1) ensure that the spectral condition is satisfied (2) choose the initial guess M_{init} carefully. The relaxation method in Section 2.2.1 is concerned with (1) and a multiscale method in Section 2.3 addresses (2).

2.2.1. Relaxation

As a fixed point iteration, the alternating sweeping converges locally if the spectral radius of the Jacobian matrix at the fixed point is smaller than 1, as is clarified in Proposition 1. However, conditions (2.4) and (2.5) may not be satisfied. In order to address this, we propose a relaxation technique to improve the convergence of Algorithm 1. More specifically, we use

$$U \leftarrow \alpha U_{\text{new}} + (1-\alpha)U_{\text{old}}, \quad M \leftarrow \alpha M_{\text{new}} + (1-\alpha)M_{\text{old}}, \tag{2.10}$$

when updating U and M in Algorithm 1 instead of $U \leftarrow U_{\text{new}}$ and $M \leftarrow M_{\text{new}}$, where $\alpha \in (0, 1]$ is a relaxation factor.

To see how the relaxation technique helps the condition in Proposition 1 on spectral radius, we state the following proposition.

Proposition 2. *Assume that F and G are continuously differentiable, and that $F_u(U^*, M^*)$ and $G_m(U^*, M^*)$ are invertible, where U^* and M^* are the solution to the equation (2.2), and F_u and G_m are defined in Proposition 1. Then relaxation (2.10) guarantees local convergence if α satisfies*

$$0 < \alpha < 2/(1 + \rho(G_m^{-1}G_u F_u^{-1}F_m)),$$

as long as

$$\left|\lambda_{max}(G_m^{-1}G_u F_u^{-1}F_m)\right| < 1, \tag{2.11}$$

where λ_{max} denotes the eigenvalue with the largest mode.

Proof. When F and G are continuously differentiable and $F_u(U^*, M^*)$ and $G_m(U^*, M^*)$ are invertible, Algorithm 1 leads to the fixed point iteration $M_{k+1} = \phi(M_k)$ for a continuously differentiable function ϕ by the implicit function theorem, and $\phi(M^*) = M^*$. Moreover, the Jacobian matrix $J = \frac{\partial \phi}{\partial M}(M^*) = G_m^{-1}G_u F_u^{-1}F_m$, which is clear from (2.9). Now we plug in (2.10), which leads to the iteration scheme

$$M_{k+1} = \alpha\phi(M_k) + (1-\alpha)M_k. \tag{2.12}$$

By taking Taylor expansion of ϕ at M^*, we get

$$\phi(M_k) = \phi(M^*) + J(M_k - M^*) + o(\|M_k - M^*\|).$$

Plugging this equation into (2.12) leads to

$$\begin{aligned} M_{k+1} - M^* &= \alpha(\phi(M_k) - M^*) + (1-\alpha)(M_k - M^*) \\ &= (\alpha J + (1-\alpha)\mathcal{I})(M_k - M^*) + o(\|M_k - M^*\|), \end{aligned}$$

where $\mathcal{I}$ is the identity matrix. Thus the iteration converges locally if (see for example [24])

$$\rho(\alpha J + (1-\alpha)\mathcal{I}) < 1. \tag{2.13}$$

Notice that

$$\alpha J + (1-\alpha)\mathcal{I} = \mathcal{I} - \alpha(\mathcal{I} - J). \tag{2.14}$$

We see that condition (2.13) is satisfied for $\alpha < 2/(1 + \rho(G_m^{-1}G_u F_u^{-1}F_m))$ as long as $|\lambda_{max}(J)| < 1$. By Proposition 1, the local convergence of U_k holds under the same condition, which closes the proof. $\quad\square$

From Proposition 2 we see that the relaxed scheme (2.10) requires much weaker conditions than those required by the original scheme which are stated in Proposition 1. This is in agreement with our observations in numerical experiments, i.e., in many cases where the original scheme fails to converge, the relaxed scheme (2.10) still converges with a sufficiently small relaxation factor α.

In practice, we can use different relaxation factor α for different iteration steps in Algorithm 1. For example, we can start with a small α, and when M_k and U_k are close enough to the fixed point, say, $\max\{\text{err}_M, \text{err}_U\} < 5\epsilon$, we then use a larger α, or even set $\alpha = 1$. In practice, this adaptive choice of α is able to further accelerate the convergence of our method.

2.3. Multiscale algorithm

How to give a proper initial guess M_{init} is critical for the Algorithm 1. A naive way is to set

$$m_i^n = m_i^{N_t} = m_T(x_i), \quad n = 0, 1, \ldots, N_t - 1, \quad i = 1, 2, \ldots, N. \tag{2.15}$$

However, this initial guess can be far away from the real solution, which can lead to more alternating sweeping steps and longer computation time, or even failure of convergence.

We address the selection of M_{init} by using a multiscale method. In the first step, we solve the equations (2.1) on the coarsest grid with certain numerical methods (we will explain this choice more specifically later). At each finer level, an initial guess of the solution is obtained by interpolating the approximate solution from the previous level. This process is repeated until the finest grid. In particular, the initial guess on the finest grid M_{init}^L is given by interpolating M^{L-1}, which can be much better than the naive initial guess (2.15). The pseudocode of this multiscale method is summarized in Algorithm 2, where $M^{(\ell)}$, $U^{(\ell)}$ denote the approximate solution of M and U on the ℓ-th level grid, respectively.

Algorithm 2 Multiscale algorithm for MFG.

Input: Initial guess of the density $M_{\text{init}}^{(L_0)}$ on coarsest grid $\ell = L_0$
Output: Numerical solution $M^{(L)}$ and $U^{(L)}$ of MFG (2.1) on finest grid level L
  function MULTISCALE($M^{(L_0)}$)
    Solving (2.1) on the L_0 grid by a given method with initial guess M_{init} gives $M^{(L_0)}$ and $U^{(L_0)}$
    for ℓ from $L_0 + 1$ to L **do**
      Interpolating $M^{(\ell-1)}$ by an interpolation method gives $M_{\text{init}}^{(\ell)}$;
      $U^{(\ell)}$, $M^{(\ell)}$ =ALTERNATINGSWEEPING($M_{\text{init}}^{(\ell)}$);
    end for
    return $M^{(L)}$ and $U^{(L)}$;
  end function

Remark 2. This algorithm is not a multigrid type method. In each step, one moves from a coarse grid to a fine grid and then apply the Alternating Sweeping method, and never goes back to the coarse grid after that.

Numerical method on the coarsest grid   Numerical results show that, when the multiscale method is used to provide the initial guess, the alternating sweeping is significantly accelerated and the time spent on the multiscale hierarchy for constructing the initial guess is negligible compared to the time saved. Since the number of discretization points on the coarsest grid is relatively small, we are able to employ methods that are more stable and possibly more expansive. The Newton iteration method in [3] is a candidate. Another choice is the alternating sweeping method with a sufficiently small relaxation factor.

Interpolation method   For the interpolation method in Algorithm 2, there are many candidates. A simple one is the linear interpolation

$$u_{2i+1}^{2n+1,(\ell)} = \frac{1}{4}\left(u_i^{n,(\ell-1)} + u_i^{n+1,(\ell-1)} + u_{i+1}^{n,(\ell-1)} + u_{i+1}^{n+1,(\ell-1)}\right). \tag{2.16}$$

Other interpolation methods can be adopted as well, such as the cubic interpolation or spline interpolation. Higher order interpolation methods introduce less interpolation error, but at the cost of increasing computational time.

2.4. Temporal and spatial discretization

It is worth emphasizing that the alternating sweeping method in Algorithm 1 and the multiscale algorithm in Algorithm 2 are independent on the specific temporal and spatial discretization used in (2.1). Various finite difference schemes can be applied, and it is also possible to use other types of methods to solve the HJB equation and KFP equation: for example, optimal control type methods for solving the equation (1.1a) when fixing the value of M. Below we introduce a new second-order finite difference scheme for discretizing the MFG systems.

2.4.1. Properties of the MFG system

In the MFG system (1.1), the two equations are coupled through the Hamiltonian, the operator $V[m]$, and the initial and terminal condition. Among them, the coupling through Hamiltonian is critical. Precisely speaking, the Hamiltonian $H(x, \nabla u): \Omega \times \mathbb{R}^d \to \mathbb{R}$ is the nonlinear term with respect to u in (1.1a). Its gradient with respect to p, i.e. $\nabla_p H$ is the nonlinear term in (1.1b). For the well-posedness of (1.1) in continuous case, it is essential that

$$\langle \nabla \cdot (m \nabla_p H), w \rangle = -\langle m \nabla_p H, \nabla w \rangle, \quad w \in H^1(\Omega) \tag{2.17}$$

holds, where the inner product $\langle f, g \rangle = \int_\Omega f \cdot g \, dx$, both for scalar functions and vector functions. The linearized Hamiltonian with respect to p is $\nabla_p H(x, \nabla u) \cdot \nabla u$, and the operator $u \rightarrow \nabla_p H(x, \nabla u) \cdot \nabla u$ is the adjoint of the operator $m \rightarrow -\nabla \cdot (m \nabla_p H(x, \nabla u))$ because

$$\left\langle -\nabla \cdot (m \nabla_p H(x, \nabla u)), u \right\rangle = \left\langle m \nabla_p H(x, \nabla u), \nabla u \right\rangle = \left\langle m, \nabla_p H(x, \nabla u) \cdot \nabla u \right\rangle \tag{2.18}$$

This property is key to the proof of uniqueness in continuous case [30].

In the proof of uniqueness of the solution, we consider any two solutions (u_1, m_1) and (u_2, m_2), subtract the HJB equations satisfied by them, multiply it by $m_1 - m_2$, and then subtract the KFP equations satisfied by them, multiply it by $u_1 - u_2$, and then subtract these two parts. The resulting expression is the sum of 4 nonnegative terms, and is equal to 0, and it turns out that (u_1, m_1) must be the same with (u_2, m_2). In order for this argument to be valid, we must use the convexity of H with respect to ∇u, and only under the condition (2.18) can we obtain the form of the first-order condition of convexity of H. More details can be found in [30].

In [3], a first-order finite difference scheme is derived using the discrete analog of (2.17) and its well-posedness is proved. Often, first-order schemes enjoy nice properties such as non-negativity preserving of M, etc., but they can be inefficient, as they require small temporal and spatial steps even for relatively low accuracy. This problem can be serious especially in 2-dimensional or 3-dimensional cases. Below, we derive a second-order scheme, where a discrete version of (2.17) is used to handle the coupling caused by the Hamiltonian. For simplicity, the following derivation assumes the spatial dimension $d = 1$, but one can extend it to multi-dimensional cases without any difficulty.

2.4.2. Spatial discretization

In the following derivation, let W be an arbitrary grid function. The spatial discretization of the gradient operator ∇ follows the second-order upwind scheme (also called Beam-Warming scheme, see [7]). We introduce two one-sided second-order difference operators

$$(\mathcal{D}W)_i^- = \frac{3W_i - 4W_{i-1} + W_{i-2}}{2h}, \quad (\mathcal{D}W)_i^+ = \frac{-3W_i + 4W_{i+1} - W_{i+2}}{2h}, \tag{2.19}$$

denote $(\mathcal{D}W)_i = ((\mathcal{D}W)_i^-, (\mathcal{D}W)_i^+)$, and let the discrete analog of Hamiltonian H (also denoted as H for notational convenience) be

$$H(x, \nabla u)|_{x_i} \rightarrow H(x_i, (\mathcal{D}U)_i^-, (\mathcal{D}U)_i^+) = H(x_i, (\mathcal{D}U)_i). \tag{2.20}$$

The explicit expression of the discrete Hamiltonian $H(x_i, (\mathcal{D}U)_i)$ depends on the form of the original Hamiltonian. For instance, when $H(x, \nabla u) = \phi(x) + |\nabla u|^\gamma, \gamma \geq 0$,

$$\mathcal{H}(x_i, (\mathcal{D}U)_i) = \phi(x_i) + \left(\sqrt{\max(-(\mathcal{D}U)_i^-, 0)^2 + \max((\mathcal{D}U)_i^+, 0)^2} \right)^\gamma. \tag{2.21}$$

For the Laplace operator, the three-point central difference is

$$(-\Delta u)_i \rightarrow (\frac{1}{\nu} \mathcal{L}u)_i := \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}. \tag{2.22}$$

Recall that in Section 2.1 $B_i(m, u)$ is introduced as the discretization of $\nabla \cdot (m \nabla_p H(x, \nabla u))$. According to the analysis of the Hamiltonian in Section 2.4.1, we replace this term with the discrete analog of (2.17),

$$\sum_{i=1}^N B_i(m, m) W_i = -\sum_{i=1}^N m_i (\nabla_p H(x_i, (\mathcal{D}u)_i) \cdot (\mathcal{D}W)_i), \tag{2.23}$$

where $\nabla_p H(x_i, (\mathcal{D}u)_i)$ denotes the gradient of the discrete Hamiltonian H with respect to $\mathcal{D}u$. Comparing the coefficient of W_i on both sides of the equation leads to the expression of $B_i(m, u)$ as

$$\begin{aligned} B_i(m, u) = \frac{1}{2h} \left[3m_i \left(\frac{\partial H}{\partial p_1} \right)_i - 4m_{i-1} \left(\frac{\partial H}{\partial p_1} \right)_{i-1} + m_{i-2} \left(\frac{\partial H}{\partial p_1} \right)_{i-2} \right. \\ \left. - 3m_i \left(\frac{\partial H}{\partial p_2} \right)_i + 4m_{i+1} \left(\frac{\partial H}{\partial p_2} \right)_{i+1} - m_{i+2} \left(\frac{\partial H}{\partial p_2} \right)_{i+2} \right], \end{aligned} \tag{2.24}$$

where $\frac{\partial H}{\partial p_1}$ is the partial derivative of H with respect to $(\mathcal{D}u)^-$, and $\frac{\partial H}{\partial p_2}$ is the partial derivative of H with respect to $(\mathcal{D}u)^+$. Finally, a semi-discrete version of MFG with respect to x is given as follows:

$$\begin{aligned} \frac{\partial u_i}{\partial t} + (\mathcal{L}u)_i + H(x, (\mathcal{D}u)_i) &= V[M]_i, \quad i = 1, \ldots, N, \\ -\frac{\partial m_i}{\partial t} + (\mathcal{L}m)_i - B_i(m, u) &= 0, \quad i = 1, \ldots, N. \end{aligned} \tag{2.25}$$

2.4.3. Temporal discretization

The specific expression of spatial discretization is not important in the following discussion. To simplify the discussion, let us introduce

$$\mathcal{S}(u, m)_i = -(\mathcal{L}u)_i - H(x, (\mathcal{D}u)_i) + V[m]_i, \quad i = 1, \dots, N, \tag{2.26a}$$

$$\mathcal{T}(u, m)_i = (\mathcal{L}m)_i - B_i(m, u), \quad i = 1, \dots, N. \tag{2.26b}$$

Then the Crank-Nicolson scheme can be written as

$$\frac{u_i^{n+1} - u_i^n}{\tau} = \frac{1}{2} \left(\mathcal{S}(u^n, m^n)_i + \mathcal{S}(u^{n+1}, m^{n+1})_i \right), \tag{2.27a}$$

$$\frac{m_i^{n+1} - m_i^n}{\tau} = \frac{1}{2} \left(\mathcal{T}(u^n, m^n)_i + \mathcal{T}(u^{n+1}, m^{n+1})_i \right). \tag{2.27b}$$

Together with the initial condition and terminal condition

$$u_i^0 = u_0(x_i) + V_0[m^0]_i, \quad i = 1, 2, \dots, N, \tag{2.28a}$$

$$m_i^{N_t} = m_T(x_i), \quad i = 1, 2, \dots, N, \tag{2.28b}$$

the equation (1.1) is fully discretized as follows

$$\frac{u_i^{n+1} - u_i^n}{\tau} = \frac{1}{2}(-(\mathcal{L}(u^n + u^{n+1}))_i - H(x_i, (\mathcal{D}u^n)_i) - H(x_i, (\mathcal{D}u^{n+1})_i) + V[m^n]_i + V[m^{n+1}]_i),$$

$$u_i^0 = u_0(x_i) + V_0[m^0]_i, \quad i = 1, \dots, N, n = 0, 1, 2, \dots, N_t - 1, \tag{2.29a}$$

$$\frac{m_i^{n+1} - m_i^n}{\tau} = \frac{1}{2}((\mathcal{L}(m^n + m^{n+1}))_i - B_i(m^n, u^n) - B(m^{n+1}, u^{n+1})),$$

$$m_i^{N_t} = m_T(x_i), \quad i = 1, \dots, N, n = 0, 1, 2, \dots, N_t - 1. \tag{2.29b}$$

The following propositions summarize the properties satisfied by this dicretization scheme.

Proposition 3. *Assume that the real solution u and m are at least fourth-order differentiable and the operator V is defined pointwise, then the local truncation error of scheme* (2.29) *is* $O(\tau^2 + h^2)$.

Proposition 4. *The total mass* $h \sum_i m_i$ *is conserved in scheme* (2.29).

The proofs of these two propositions are given in Appendices B and C, respectively.

2.4.4. Solvers

The discretization (2.29) derived above can be used to derive the solvers of the HJB equation and KFP equation for the MFG. The discrete KFP equation (2.29b) is linear with respect to M, if U is fixed, but the HJB equation is nonlinear with respect to U due to the Hamiltonian H. However, by means of an inner iteration, the HJB equation can be reduced to a linear problem as well. In the inner iteration that solves for u^{n+1}, we use u^n as the initial guess and solve the linearized (with respect to U) version of (2.29a). Notice that the solution is usually smooth with respect to time t, so the inner iterations can be effectively carried out by a Newton type inner iteration. Hence, we have the solvers for HJB and KFP in Algorithms 3 and 4, respectively.

Remark 3. In the inner iteration of Algorithm 3, u^n is chosen as the initial value for solving u^{n+1}. If the solution u is smooth with respect to t, the difference $u^{n+1} - u^n$ is small, thus the number of inner iteration in Algorithm 3 is small. Numerical tests show that the average number of inner iterations in Algorithm 3 is approximately 2.

3. Numerical results

This section presents several numerical examples for one-dimensional and two-dimensional cases to illustrate the efficiency of the proposed algorithms.

Algorithm 3 Solving discrete HJB (2.29a).

Input: A guess of the mass density $M = (m_i^n)$
Output: Solution U of the HJB equation (2.29a)

  function SOLVEHJB(M)
    $u_i^0 \leftarrow u_0(x_i), \quad i = 1, 2, \ldots, N$
    for n from 0 to $N_t - 1$ **do**
      $Z^0 \leftarrow u^n, k \leftarrow 0$
      do
        Evaluate Z^{k+1} by solving the linear system:

$$\left(\mathcal{I} + \frac{\tau \mathcal{L}}{2} + \frac{\tau}{2} \frac{\partial H}{\partial U}(Z^k) \right) Z^{k+1} = \left(\mathcal{I} - \frac{\tau \mathcal{L}}{2} \right) u^n$$

$$+ \frac{\tau}{2} \left(V[m^n] + V[m^{n+1}] - H(x, \mathcal{D}u^n) - H(x, \mathcal{D}Z^k) + \frac{\partial H}{\partial U}(Z^k) Z^k \right)$$

        $k \leftarrow k + 1$
      while $\|Z^{k+1} - Z^k\| / \|Z^k\| < \epsilon$
      $u^{n+1} \leftarrow Z^{k+1}$
    end for
    return $U = (u_i^n)$
  end function

Algorithm 4 Solving discrete KFP (2.29b).

INPUT: A guess of the value function U
OUTPUT: Solution M of the KFP equation (2.29b) with given U

  function SOLVEKFP(U)
    $m^{N_t} \leftarrow m_T(x_i), \quad i = 1, 2, \ldots, N,$
    for n from N_t to 1 by -1 **do**
      Evaluate m^{n-1} by solving the linear system:

$$\left(\mathcal{I} + \frac{\tau \mathcal{L}}{2} + \frac{\tau}{2} \left(\frac{\partial H}{\partial U}(u^n) \right)^T \right) m^{n-1} = \left(\mathcal{I} - \frac{\tau \mathcal{L}}{2} - \frac{\tau}{2} \left(\frac{\partial H}{\partial U}(u^{n+1}) \right)^T \right) m^n$$

    end for
    return M
  end function

3.1. One-dimensional case

Following the numerical setup in [3], the following Hamiltonian is used for one-dimensional numerical tests:

$$H(x, \nabla u) = -200 \cos(2\pi x) + 10 \cos(4\pi x) + |\nabla u|^\gamma, \tag{3.1}$$

where $\gamma \geq 0$. A greater γ indicates stronger nonlinearity, and $\gamma = 2$ corresponds to the usual form of energy. In all numerical tests, we set $V_0[m(t = 0)](x) = 0$ unless specifically stated.

3.1.1. First-order scheme vs second-order scheme

Let us first compare the performance of the first-order scheme and the second-order scheme. The initial-terminal conditions are chosen as:

$$u_0(x) = \sin(4\pi x) + 0.1 \cos(10\pi x), \quad x \in [0, 1], \tag{3.2a}$$

$$m_T(x) = 1 + \frac{1}{2} \cos(2\pi x), \quad x \in [0, 1]. \tag{3.2b}$$

The potential is set as $V[m](x) = m(x)^2$, and the end time is $T = 0.01$. We set the coefficient $\nu = 0.4$, $\gamma = 2$, and the relaxation factor $\alpha = 1$.

The multiscale method starts from the level $L_0 = 4$ and stops at the level $L = 13$, i.e. the size of the finest grid is $N^{(L)} = N_t^{(L)} = 8192$. We comment here that N and N_t are not required to be equal. In practice, we can use different N and N_t, and the results are similar to what we represent here. The tolerance ϵ in Algorithm 1 is set as $\epsilon = 1 \times 10^{-6}$ while in Algorithm 3, we set $\epsilon = 1 \times 10^{-7}$. In all the algorithms, the norm is chosen as the L^2 norm. The solution of U and M on the grid $N^{(10)} = N_t^{(10)} = 1024$ is shown in Fig. 1. One can see that in this case, the value of U (indicating the negative utility) near $x = 0.88$ and $x = 0.33$ is smaller at the end of the evolution, and the density of agents gathers towards the places relatively more desirable.

The convergence order can be checked by calculating the relative error of numerical solutions on each grid in comparison to the solution on the finest grid. The numerical results are summarized in Table 1. The results show that the second-order
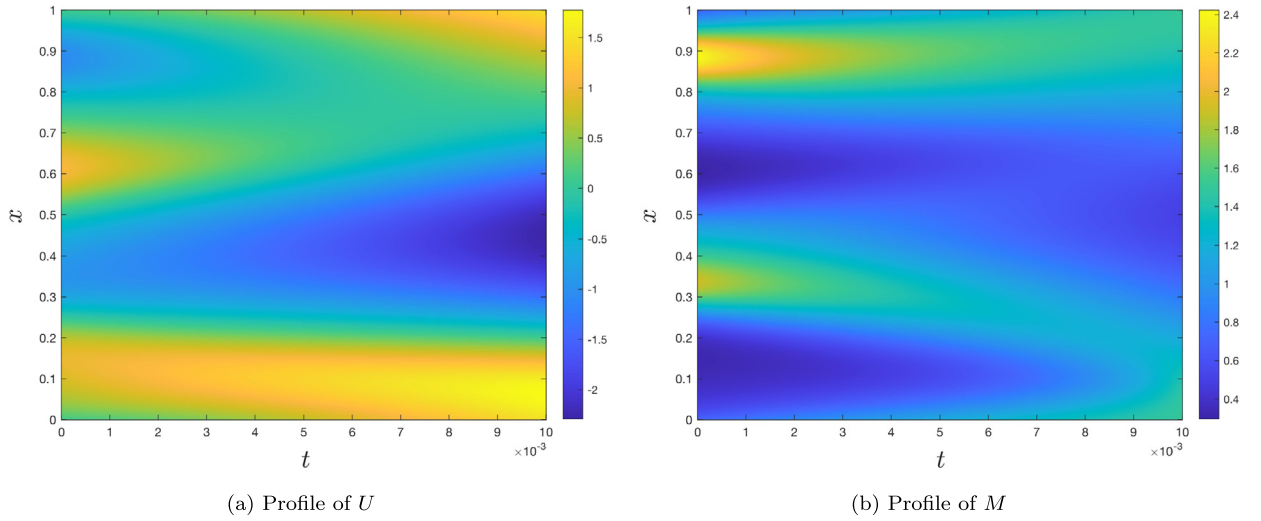
(a) Profile of U (b) Profile of M

Fig. 1. Profile of the solutions on the grid $N^{(10)} = N_t^{(10)} = 1024$.

Table 1
The convergence order of U and M.

ℓ	$\mathrm{err}_U, \mathrm{err}_M$ (1st)	Order	$\mathrm{err}_U, \mathrm{err}_M$ (2nd)	Order
4	5.2E-2, 9.1E-2	—	5.5E-2, 1.1E-1	—
5	3.0E-2, 5.4E-2	0.80, 0.77	1.9E-2, 3.9E-2	1.57, 1.54
6	1.6E-2, 3.0E-2	0.88, 0.85	5.3E-3, 1.1E-2	1.80, 1.81
7	8.5E-3, 1.5E-2	0.94, 0.92	1.4E-3, 2.9E-3	1.92, 1.92
8	4.3E-3, 8.0E-3	0.99, 0.97	3.6E-4, 7.3E-4	1.97, 2.00
9	2.1E-3, 3.9E-3	1.03, 1.02	9.0E-5, 1.8E-4	1.99, 2.00
10	9.9E-4, 1.9E-3	1.09, 1.09	2.2E-5, 4.5E-5	2.01, 2.01

Table 2
Number of inner iterations per time step in Algorithm 3.

ℓ	4	5	6	7	8	9	10	11	12	13
inner iterations / time step (1st)	3.00	3.00	2.17	2.00	2.00	2.00	2.00	2.00	2.00	2.00
inner iterations / time step (2nd)	3.00	3.00	2.20	2.00	2.00	2.00	2.00	2.00	2.00	2.00

scheme derived in this paper truly gives rise to second-order convergence: it approaches the true solution much faster than the first-order scheme, as expected.

We check next the efficiency of inner iteration applied in Algorithm 3 by counting the average number of iterations per time step. The average numbers of iterations per time step on each grid are shown in Table 2 and the result justifies Remark 3. In addition, among all time steps, the maximum of the difference between the total mass and 1 is 3.00×10^{-15} on the grid of size $N^{(10)} = N_t^{(10)} = 1024$, which verifies the conservation of total mass proved in Proposition 4.

3.1.2. Alternating sweeping vs. Newton

Let us now compare the computation time of the alternating sweeping without multiscale and the Newton method presented in [3] to demonstrate the acceleration of alternating sweeping.

The same Hamiltonian in (3.1) is used with the initial-terminal conditions

$$u_0(x) = \cos(2\pi x), \quad x \in [0, 1], \tag{3.3a}$$

$$m_T(x) = 1 + \frac{1}{2}\cos(2\pi x), \quad x \in [0, 1]. \tag{3.3b}$$

We also set $V[m](x) = m(x)^2$, $T = 1$, $\nu = 1$, $\gamma = 2$, and the relaxation factor $\alpha = 1$.

By taking appropriate stop criteria to get solutions of similar accuracy by the two methods, one can compare the running times used in the computation on the same machine, listed in Table 3. To check the correctness of our numerical solutions, the residual of (2.1) is also listed.

Clearly, the runtime of the Newton method is significantly longer than that for the alternating sweeping method (Fig. 2). The ratio of time used increases rapidly as the size of grid becomes larger, and alternating sweeping makes computation on

Table 3
The acceleration effect of the alternating sweeping and the multiscale method.

N	t_{Newton} (sec)	t_{AS} (sec)	t_{Newton}/t_{AS}	err_{Newton}	err_{AS}
32	4.4E-1	5.5E-2	8	4.3E-4	2.5E-5
64	2.0E0	1.1E-1	18	5.3E-4	1.8E-5
128	1.0E1	3.1E-1	32	6.7E-4	1.1E-5
256	7.2E1	1.0E0	72	2.2E-4	6.1E-6
512	7.6E2	3.4E0	224	5.3E-2	3.2E-6
1024	—	1.3E1	—	—	1.6E-6
2048	—	5.6E1	—	—	8.4E-7
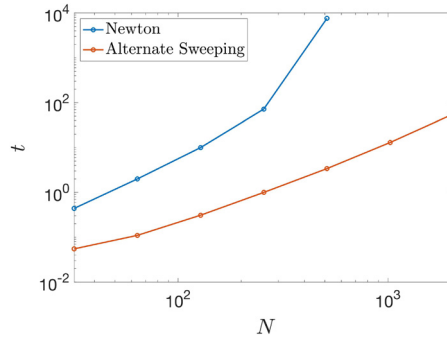
Fig. 2. Runtime of the Newton method and alternating sweeping algorithm.

large grids possible. When $N = 512$, the alternating sweeping method is 224× faster than the Newton method. Moreover, the residual of the alternating sweeping method is also relatively smaller than that of the Newton method.

3.1.3. Multiscale vs alternating sweeping

As explained earlier, alternating sweeping only ensures local convergence, and a good initial guess M_{init} in Algorithm 1 is essential. The following numerical test shows that the initial guess given by the multiscale algorithm is significantly better than the naive initial guess (2.15). Often, the algorithm fails to converge with the naive initial guess, while it converges with the initial guess given by the multiscale method. This justifies using multiscale method to improve the quality of initial guesses. The same Hamiltonian in (3.1) is used with the initial-terminal conditions

$$u_0(x) = m_0(x)^2 + \sin(2\pi x) + 0.1\cos(6\pi x), \quad x \in [0, 1], \tag{3.4a}$$

$$m_T(x) = 1 + \frac{1}{2}\cos(2\pi x), \quad x \in [0, 1], \tag{3.4b}$$

and with $V[m](x) = m(x)^2$, $T = 0.01$, $\nu = 2$, and $\gamma = 2$. Notice that here the initial condition of u is not directly given as in other examples, instead it is coupled with m_0 via $V_0[m_0](x) = m_0(x)^2$. This makes the naive initial guess particularly undesirable.

Our goal is to find the solution on the grid $N^{(10)} = N_t^{(10)} = 1024$. We compare the performance of three different ways to do this.

1. The multiscale method that starts from a coarse grid (in this example $N^{(5)} = N_t^{(5)} = 32$), and use some small α (in this example $\alpha = 0.2 * 1.1^{\ell-5}$) on the coarse grids ($5 \leq l \leq 9$), and with $\alpha = 1$ on the grid $N^{(10)} = N_t^{(10)} = 1024$.
2. The alternating sweeping that directly starts from the grid $N^{(10)} = N_t^{(10)} = 1024$ with the naive initial guess (2.15) and $\alpha = 1$.
3. The alternating sweeping that directly starts from the grid $N^{(10)} = N_t^{(10)} = 1024$ with the naive initial guess (2.15) and $\alpha = 0.2$.

These three methods correspond to the red curve, the yellow curve, and the purple curve in Fig. 3.

In Fig. 3, the red curve shows that the initial guess given by the multiscale method on each level becomes better and better and finally results in a much better initial guess than (2.15) on the finest mesh. As a result, the computation on the finest mesh $N^{(10)} = N_t^{(10)} = 1024$ converges in a few iterations. However, the yellow curve fails to converge. Comparing it with the blue curve, the reason for this failure is that the naive initial guess is much worse than the initial guess given by the multiscale method used in the blue curve. The purple curve converges but the time used is 270 seconds, which is 7.3× the time used by the multiscale method on the same machine. In conclusion, compared with the alternating sweeping without
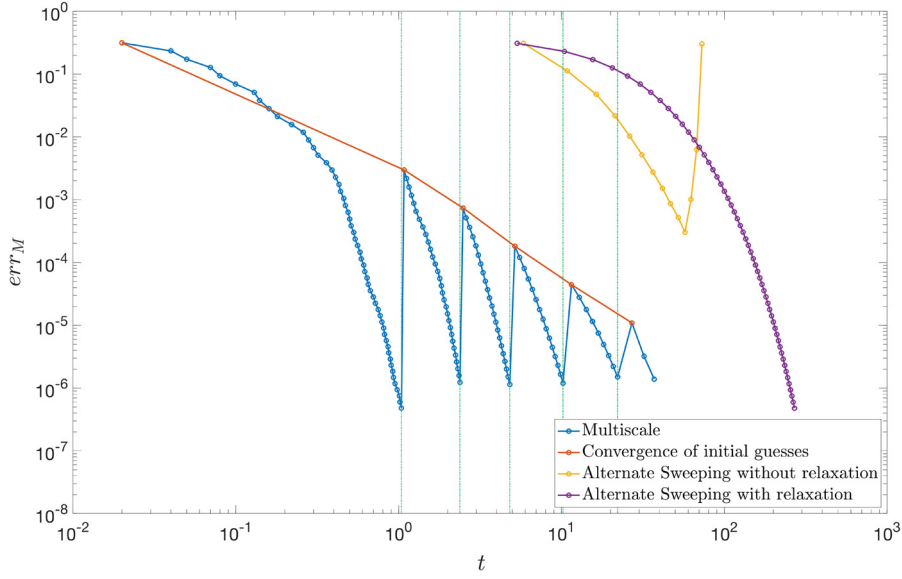
Fig. 3. Runtime of the multiscale method and the alternating sweeping algorithm. Each point in the plot denotes the end time of an alternating sweeping step, thus the plots do not start from $t = 0$. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)
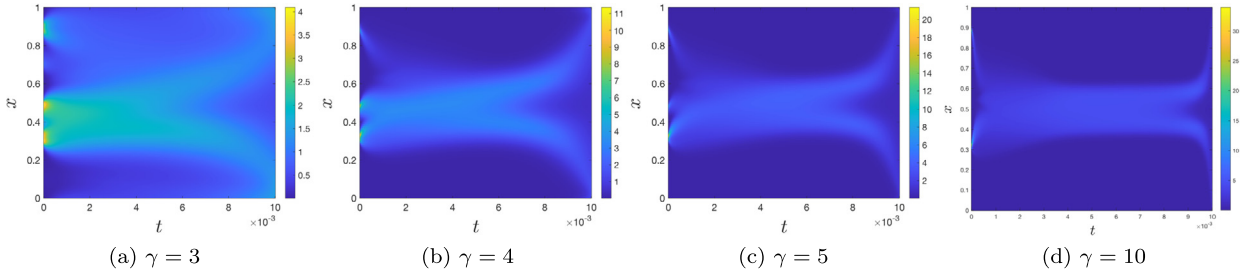

(a) $\gamma = 3$ (b) $\gamma = 4$ (c) $\gamma = 5$ (d) $\gamma = 10$

Fig. 4. Profile of the solution M for different γ.

relaxation, the multiscale method shows much better convergence; compared with the alternating sweeping algorithm with relaxation, the multiscale method is significantly faster.

3.1.4. Performance of relaxation for stronger nonlinearity case

As discussed in Section 2.2.1, one can choose sufficiently small relaxation factor α in order to improve convergence when faced with significant nonlinearity (for example, when the γ is large in the Hamiltonian in (2.21)). For strong nonlinearity cases, we carry out numerical tests with different values of γ ranges from 3 to 10. The initial-terminal conditions used are
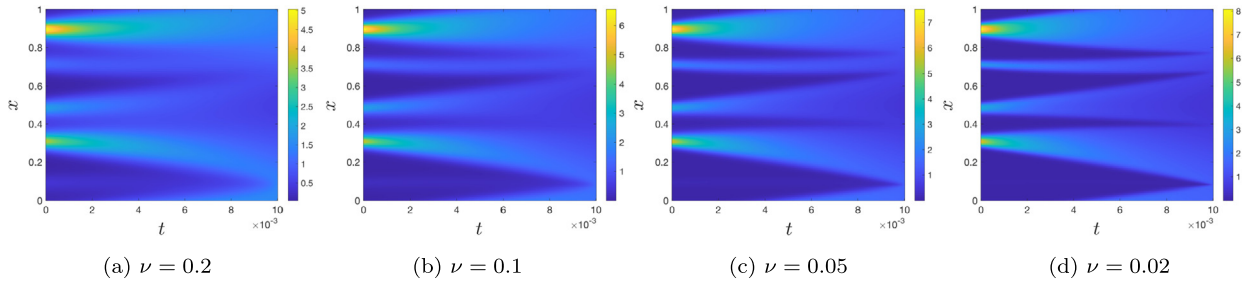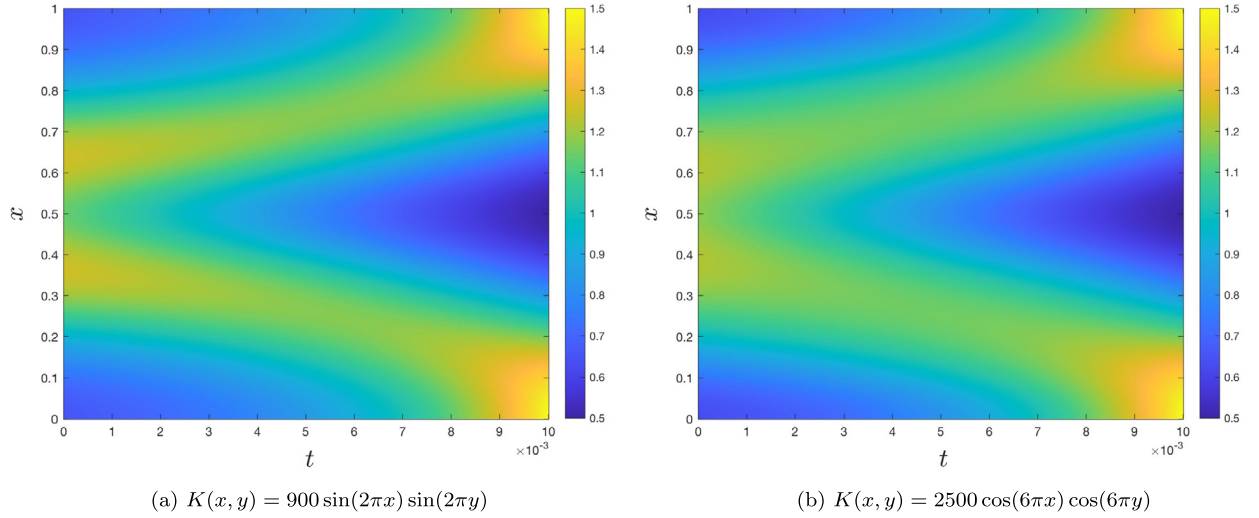
$$u_0(x) = \sin(4\pi x) + \frac{1}{2}\cos(10\pi x), \quad x \in [0, 1], \tag{3.5a}$$

$$m_T(x) = 1 + \frac{1}{2}\cos(2\pi x), \quad x \in [0, 1], \tag{3.5b}$$

and $V[m](x) = m(x)^2$, $T = 0.01$, $\nu = 2$, $\alpha = 0.1$. The results are presented in Fig. 4. As the evolution of the density of agents is more visually intuitive, we only present the solutions of M here. By checking the local truncation error, we verified that all computations converge.

3.1.5. Performance of relaxation for the weak randomness case

The Laplacian term in (1.1) is related to the stochastic behavior of agents in, for example, pedestrian dynamics. The deterministic limit is obtained by letting the coefficient ν go to 0 and thus the case with small ν is quite important. When the coefficient of the Laplace operator is small, the nonlinear term is relatively large, thus the computation usually becomes more difficult. Similar to the cases with large γ, the computation in this case typically requires using small relaxation factor α. In the following tests we use the same initial-terminal conditions as in (3.5) with $V[m](x) = m(x)^2$, $T = 0.01$, and $\gamma = 2$. Fig. 5 summarizes the results of $\nu = 0.2, 0.1, 0.05, 0.02$, with the corresponding α equal to $0.5, 0.2, 0.1, 0.05$. The result shows the process of approaching deterministic limit.

(a) $\nu = 0.2$ (b) $\nu = 0.1$ (c) $\nu = 0.05$ (d) $\nu = 0.02$

Fig. 5. Profile of the solution M for different ν.

(a) $K(x, y) = 900 \sin(2\pi x) \sin(2\pi y)$ (b) $K(x, y) = 2500 \cos(6\pi x) \cos(6\pi y)$

Fig. 6. Profile of the solution M for different nonlocal $V[m]$.

3.1.6. Nonlocal case

In this section we test the performance of the proposed method on a nonlocal MFG problem. Inspired by the examples used in [34], we consider $V[m](x) = \int_0^1 K(x, y)m(y)\mathrm{d}y$, where K is symmetric. The initial-terminal conditions used are

$$u_0(x) = \cos(2\pi x), \quad x \in [0, 1], \tag{3.6a}$$

$$m_T(x) = 1 + \frac{1}{2}\cos(2\pi x), \quad x \in [0, 1], \tag{3.6b}$$

and $T = 0.01$, $\nu = 0.4$, $\alpha = 0.1$ The results are presented in Fig. 6. By checking the local truncation error, we verified that the computation converges.

3.2. Two-dimensional case

In this subsection, we perform some tests for the two-dimensional case, and show that all the conclusions for the one-dimensional case are also valid for the two-dimensional case.

3.2.1. First-order scheme vs second-order scheme

Let us first compare the performance between the first-order scheme and the second-order scheme proposed in this paper. In two-dimensional cases, the cost of computation on the fine grid is significantly higher than that on the coarse grid. Since the first-order scheme converges only by a linear rate, one needs to compute the numerical solution on a very fine grid in order to get a solution with high accuracy. The faster convergence of the second-order scheme becomes particularly attractive. The following Hamiltonian is used in the test.

$$H(x, \nabla u) = \cos(4\pi x_1) + \sin(2\pi x_1) + \sin(2\pi x_2) + |\nabla u|^{\gamma}, \tag{3.7}$$

with initial-terminal conditions given by

Table 4
The order of convergence of U and M.

N_1	N_2	N_T	err_U, err_M (1st)	Order	err_U, err_M (2nd)	Order
16	16	16	2.6E-1, 6.5E-2	—	1.0E0, 4.4E-2	—
32	32	32	1.9E-1, 3.3E-2	0.49, 0.98	2.6E-1, 1.1E-2	1.9, 2.0
64	64	64	1.0E-1, 1.4E-2	0.93, 1.2	6.4E-2, 2.4E-3	2.0, 2.2

$$u_0(x) = \cos(2\pi x_1) + \cos(2\pi x_2), \quad x \in [0, 1], \tag{3.8a}$$

$$m_T(x) = 1 + \frac{1}{2}\cos(2\pi x_1) + \frac{1}{2}\cos(2\pi x_2), \quad x \in [0, 1]. \tag{3.8b}$$

Other parameters and functions are chosen as:

$$\nu = 1, \quad T = 1, \quad V[m](x) = m(x)^2, \quad \alpha = 1, \quad \gamma = 2, \tag{3.9}$$

where α is the relaxation factor. With the goal of computing a numerical solution on the grid of size $N_1^{(7)} = N_2^{(7)} = N_t^{(7)} = 128$, we start the computation from the grid of the size $N_1^{(4)} = N_2^{(4)} = N_t^{(4)} = 16$.

Now we check the convergence orders of the two schemes, by calculating the relative error of numerical solutions on each grid in comparison to the solution on the grid of the size $N_1^{(8)} = N_2^{(8)} = N_t^{(8)} = 256$. Table 4 shows that the second-order scheme is truly of second-order, and the first-order scheme is of first-order. The solution M is shown in Fig. 7 at different times.

3.2.2. Multiscale vs. alternating sweeping

The following example provides a clear demonstration how the multiscale method can accelerate convergence. The example used is the one in Section 3.2.1 with $\gamma = 1.5$ and $\alpha = 0.5$. The size of the finest grid is $N_1^{(L)} = N_2^{(L)} = N_t^{(L)} = 320$. We compare the performance of using multiscale method with the performance of using only alternating sweeping. The alternating sweeping without multiscale takes 9 steps to reach the accuracy desired with total time equal to 1.2×10^4 seconds. When applying the multiscale algorithm, the coarsest grid size is chosen as $N_1^{(L_0)} = N_2^{(L_0)} = N_t^{(L_0)} = 20$ and it takes 1.6×10^3 seconds in total on the same machine. The acceleration ratio is equal to $1.2 \times 10^4 / 1.6 \times 10^3 = 7.5$.

The reason of this acceleration can be seen clearly in the computation process shown in Fig. 8. The interpolation process introduces error, which is why the blue curve goes up at the first step after each dash-dot line. As we can see, the time cost on the coarse grid is negligible compared to the cost on the finest grid: although we need 14 steps on the coarsest grid, the cost of it is little, and we get a much better initial guess by the computation on coarse grids. In the multiscale algorithm, the error decreases rapidly, leading to only 1 alternating sweeping on the finest grid after applying the multiscale method. In the case without multiscale, the computation on the finest grid has to start from a much worse initial guess, leading it to take a lot more steps to converge.

4. Conclusion

We introduce an alternating sweeping method, which decouples the forward-backward MFG system into a forward HJB system and a backward KFP system and allows for the use of classical time marching numerical schemes on it. We also introduce a multiscale method along with relaxation technique in order to guarantee the convergence. A new second-order scheme is proposed for the time and spatial discretization. Numerical results show the proposed multiscale algorithm with alternating sweeping is robust and efficient in both one and two dimensions.
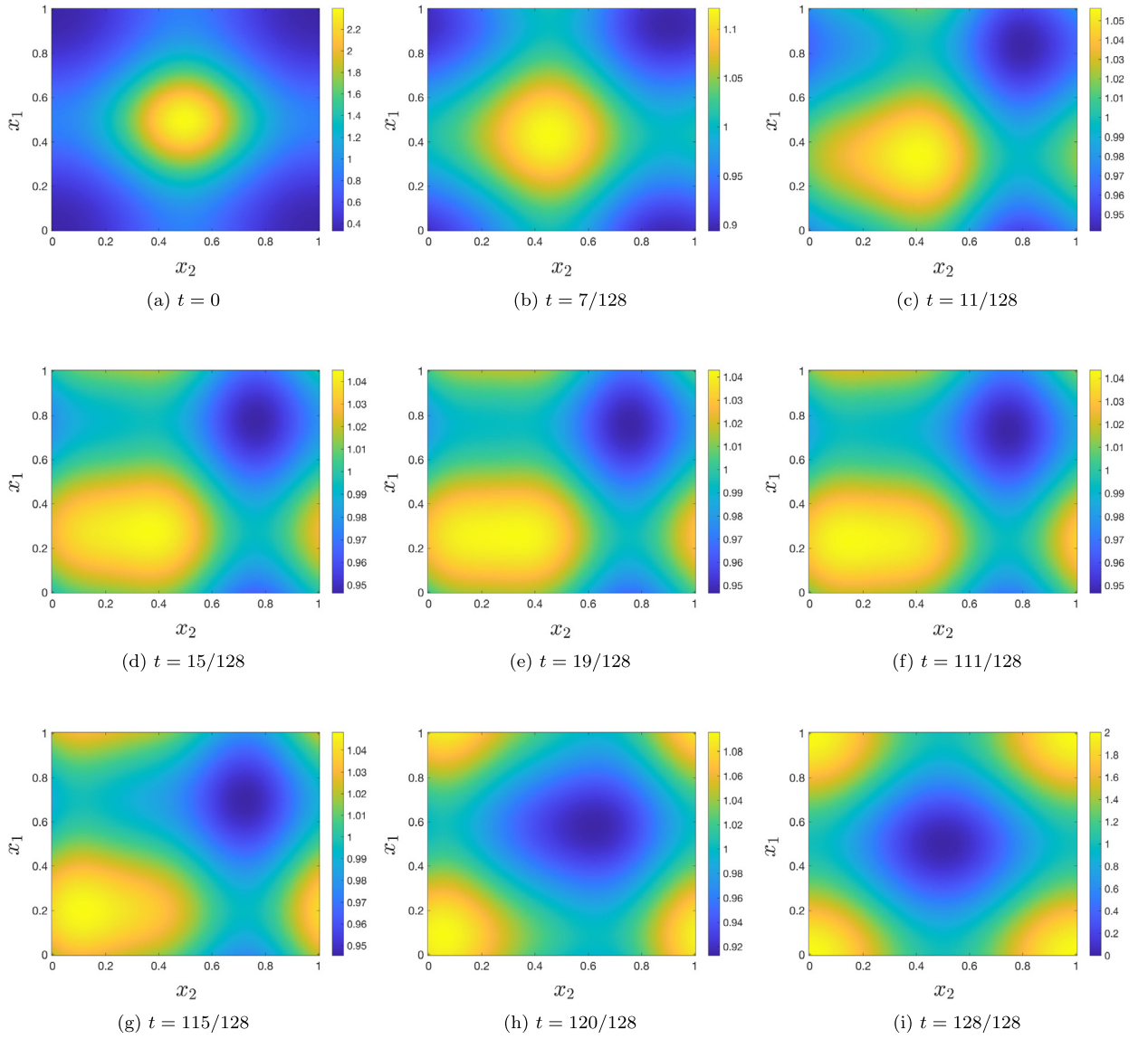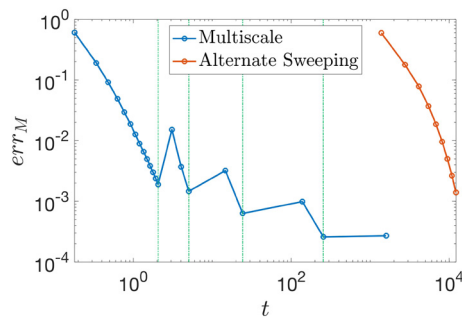
To simplify our discussion, we have only considered the periodic boundary conditions in space. An immediate future work is to incorporate other boundary conditions, for example following the work of [9]. With proper adaptation, the proposed method would also be applied to similar problems, such as the planning problems in [1].

CRediT authorship contribution statement

Haoya Li: Conceptualization, Data curation, Formal analysis, Investigation, Methodology, Resources, Software, Writing – original draft, Writing – review & editing. **Yuwei Fan:** Conceptualization, Methodology, Software, Validation, Writing – review & editing. **Lexing Ying:** Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

(a) $t = 0$

(b) $t = 7/128$

(c) $t = 11/128$

(d) $t = 15/128$

(e) $t = 19/128$

(f) $t = 111/128$

(g) $t = 115/128$

(h) $t = 120/128$

(i) $t = 128/128$

Fig. 7. Profile of the solution of M on different time t.

Fig. 8. The residual (2.2) of the multiscale method and the alternating sweeping algorithm. Each point in the plot denotes the end time of an alternating sweeping step, thus the plots do not start from $t = 0$.

Acknowledgements

Appendix A. Proof of Lemma 1

Before the proof of Lemma 1, we introduce the famous spectral radius formula (Gelfand's formula [31]) as follows.

Lemma 2. *For any element T of a Banach algebra, we have*

$$\rho(T) = \lim_{n \to \infty} \|T^k\|^{1/k}, \tag{A.1}$$

where ρ is the spectral radius. And in particular, we have

$$\rho(A) = \lim_{n \to \infty} \|A^k\|^{1/k}, \tag{A.2}$$

for any matrix $A \in \mathbb{C}^{n \times n}$ and any matrix norm $\| \cdot \|$.

Proof of Lemma 1. When $A = 0$ or $B = 0$, the conclusion holds trivially. From now on we assume that A and B are both non-zero.

For any matrix in a finite dimensional vector space, the matrix norm is always finite, and since A and B are non-zero, $0 < \|A\| < \infty$, $0 < \|B\| < \infty$. Thus $\lim\limits_{n \to \infty} \|A\|^{\frac{1}{n}} = \lim\limits_{n \to \infty} \|B\|^{\frac{1}{n}} = 1$. Notice that

$$\|(AB)^n\|^{\frac{1}{n}} \leq \|A\|^{\frac{1}{n}} \|(BA)^{n-1}\|^{\frac{1}{n}} \|B\|^{\frac{1}{n}}. \tag{A.3}$$

Since $\lim\limits_{n \to \infty} \|(BA)^{n-1}\|^{\frac{1}{n}} = \lim\limits_{n \to \infty} (\|(BA)^{n-1}\|^{\frac{1}{n-1}})^{\frac{n-1}{n}} = \rho(BA)$, $\lim\limits_{n \to \infty} \|A\|^{\frac{1}{n}} = \lim\limits_{n \to \infty} \|B\|^{\frac{1}{n}} = 1$, we have $\rho(AB) \leq \rho(BA)$ by taking the limit $n \to \infty$ in the inequality above. Put A in the place of B and B in the place of A, we get $\rho(BA) \leq \rho(AB)$. Thus $\rho(AB) = \rho(BA)$. $\quad \square$

Appendix B. Proof of Proposition 3

Proof of Proposition 3. Supposing that u and m are the true solution of the equation (1.1), we calculate the difference between the finite difference operator and the differential operator, and get

$$\frac{u(x_i, t_{n+1}) - u(x_i, t_n)}{\tau} - \frac{\partial u}{\partial t}(x_i, t_{n+\frac{1}{2}}) = -\frac{1}{12} \frac{\partial^3 u}{\partial t^3}(x_i, t_{n+\frac{1}{2}}) \tau^2 + O(\tau^4), \tag{B.1}$$

$$\frac{u(x_{i+1}, t_n) - 2u(x_i, t_n) + u(x_{i-1}, t_n)}{h^2} - \frac{\partial^2 u}{\partial x^2}(x_i, t_n) = -\frac{1}{12} \frac{\partial^4 u}{\partial x^4}(x_i, t_n) h^2 + O(h^4). \tag{B.2}$$

Furthermore, we have:

$$\frac{3u(x_i) - 4u(x_{i-1}) + u(x_{i-2})}{2h} - \frac{\partial u}{\partial x}(x_i) = \frac{h^2}{3} \frac{\partial^3 u}{\partial x^3}(\xi), \quad \xi \in [x_{i-2}, x_i],$$

$$\frac{-3u(x_i) + 4u(x_{i+1}) - u(x_{i+2})}{2h} - \frac{\partial u}{\partial x}(x_i) = \frac{h^2}{3} \frac{\partial^3 u}{\partial x^3}(\xi), \quad \xi \in [x_i, x_{i+2}],$$

thus

$$\mathcal{D}^+ u(x_i, t_n) - \frac{\partial u}{\partial x}(x_i, t_n) = O(h^2), \quad \mathcal{D}^- u(x_i, t_n) - \frac{\partial u}{\partial x}(x_i, t_n) = O(h^2).$$

Thus we have

$$\left| H(x_i, \mathcal{D}u) - H(x_i, \frac{\partial u}{\partial x}) \right| \leq \max \left(\left| \frac{\partial H}{\partial p}(x_i, \frac{\partial u}{\partial x})(\mathcal{D}^+ u - \frac{\partial u}{\partial x}) \right|, \left| \frac{\partial H}{\partial p}(x_i, \frac{\partial u}{\partial x}) \cdot (\mathcal{D}^- u - \frac{\partial u}{\partial x}) \right| \right) + O(h^6)$$

$$= O(h^2).$$

Because V is defined pointwisely, operator V has no truncation error. The analysis for the Fokker-Planck equations is the same. Combine the formula above, we get the desired conclusion. $\quad \square$

Appendix C. Proof of Proposition 4

Proof of Proposition 4. The total mass can be represented as:

$$total\ mass = h \sum_i m_i = h \langle m, 1 \rangle_h. \tag{C.1}$$

From (2.29b) we know:

$$m^n - m^{n+1} = \frac{\tau}{2}(-\mathcal{L}(m^n + m^{n+1}) + B(m^n, u^n) + B(m^{n+1}, u^{n+1})), \quad n = 0, 1, 2, \ldots, N_t - 1,$$

thus

$$\langle m^n, 1 \rangle_h - \langle m^{n+1}, 1 \rangle_h = -\frac{\tau}{2} \langle \mathcal{L}(m^n + m^{n+1}), 1 \rangle_h + \frac{\tau}{2}(\langle B(m^n, u^n), 1 \rangle_h + \langle B(m^{n+1}, u^{n+1}), 1 \rangle_h)$$

$$= -\frac{\nu\tau}{2} \langle \mathcal{D}_0(m^n + m^{n+1}), \mathcal{D}_0 1 \rangle_h - \frac{\tau}{2}(\langle m^n \nabla_p \mathcal{H}^n, \mathcal{D}1 \rangle_h + \langle m^{n+1} \nabla_p H^{n+1}, \mathcal{D}1 \rangle_h)$$

$$= 0, \quad n = 0, 1, 2, \ldots, N_t - 1,$$

where $\mathcal{D}_0 W = \left(\frac{W_{i+1} - W_i}{h}, \frac{W_i - W_{i-1}}{h} \right)$, and 1 denotes the grid function whose every coordinate equals 1, and the last equation is correct because every coordinate of $\mathcal{D}1$ and $\mathcal{D}_0 1$ is 0.

As a result,

$$h \langle m^n, 1 \rangle = h \langle m^{N_t}, 1 \rangle = 1, \tag{C.2}$$

which is exactly the conclusion we want to draw. □

References

[1] Yves Achdou, Fabio Camilli, Italo Capuzzo-Dolcetta, Mean field games: numerical methods for the planning problem, SIAM J. Control Optim. 50 (1) (2012) 77–109.
[2] Yves Achdou, Fabio Camilli, Italo Capuzzo-Dolcetta, Mean field games: convergence of a finite difference method, SIAM J. Numer. Anal. 51 (5) (2013) 2585–2612.
[3] Yves Achdou, Italo Capuzzo-Dolcetta, Mean field games: numerical methods, SIAM J. Numer. Anal. 48 (3) (2010) 1136–1162.
[4] Yves Achdou, Victor Perez, Iterative strategies for solving linearized discrete mean field games systems, Netw. Heterog. Media 7 (2) (2012) 197.
[5] Yves Achdou, Alessio Porretta, Mean field games with congestion, Annales de l'Institut Henri Poincaré C, Analyse non linéaire, vol. 35, Elsevier, 2018, pp. 443–480.
[6] Roman Andreev, Preconditioning the augmented Lagrangian method for instationary mean field games with diffusion, SIAM J. Sci. Comput. 39 (6) (2017) A2763–A2783.
[7] Richard M. Beam, R.F. Warming, An implicit factored scheme for the compressible Navier-Stokes equations, AIAA J. 16 (Suppl.) (1978) 318.
[8] Jean-David Benamou, Guillaume Carlier, Augmented Lagrangian methods for transport optimization, mean field games and degenerate elliptic equations, J. Optim. Theory Appl. 167 (1) (2015) 1–26.
[9] Jean-David Benamou, Guillaume Carlier, Filippo Santambrogio, Variational Mean Field Games, Springer International Publishing, Cham, 2017, pp. 141–171.
[10] Luis Briceño-Arias, Dante Kalise, Ziad Kobeissi, Mathieu Laurière, A. Mateos Gonzalez, Francisco José Silva, On the implementation of a primal-dual algorithm for second order time-dependent mean field games with local couplings, ESAIM Proc. Surv. 65 (2019) 330–348.
[11] Luis M. Briceno-Arias, Dante Kalise, Francisco J. Silva, Proximal methods for stationary mean field games with local couplings, SIAM J. Control Optim. 56 (2) (2018) 801–836.
[12] Pierre Cardaliaguet, Notes on mean field games, Preprint, 2011.
[13] Pierre Cardaliaguet, Jean-Michel Lasry, Pierre-Louis Lions, Alessio Porretta, Long time average of mean field games, Netw. Heterog. Media 7 (2) (2012) 279.
[14] E. Carlini, F.J. Silva, A semi-discrete approximation for a first order mean field game problem, Netw. Heterog. Media 7 (2012) 263.
[15] E. Carlini, F.J. Silva, A fully-discrete semi-Lagrangian scheme for a first order mean field game problem, SIAM J. Numer. Anal. 52 (1) (2013) 45–67.
[16] E. Carlini, F.J. Silva, Semi-Lagrangian schemes for mean field game models, in: 52nd IEEE Conference on Decision and Control, 2013, pp. 3115–3120.
[17] E. Carlini, F.J. Silva, A semi-Lagrangian scheme for a degenerate second order mean field game system, Discrete Contin. Dyn. Syst., Ser. A 35 (9) (2015) 4269–4292.
[18] Elisabetta Carlini, Francisco J. Silva, A fully-discrete scheme for systems of nonlinear Fokker-Planck-Kolmogorov equations, in: PDE Models for Multi-Agent Phenomena, Springer, 2018, pp. 195–218.
[19] Elisabetta Carlini, Francisco J. Silva, On the discretization of some nonlinear Fokker–Planck–Kolmogorov equations and applications, SIAM J. Numer. Anal. 56 (4) (2018) 2148–2177.
[20] René Carmona, Francois Delarue, Probabilistic Theory of Mean Field Games with Applications, Springer, Cham, 2018.
[21] Yat Tin Chow, Jérôme Darbon, Stanley Osher, Wotao Yin, Algorithm for overcoming the curse of dimensionality for certain non-convex Hamilton–Jacobi equations, projections and differential games, Ann. Math. Sci. Appl. 3 (2) (2018) 369–403.
[22] Yat Tin Chow, Wuchen Li, Stanley Osher, Wotao Yin, Algorithm for Hamilton–Jacobi equations in density space via a generalized Hopf formula, J. Sci. Comput. 80 (2) (2019) 1195–1239.
[23] Marco Cirant, Multi-population mean field games systems with Neumann boundary conditions, J. Math. Pures Appl. 103 (5) (2015) 1294–1315.
[24] Gautschi Walter, Numerical Analysis, Springer Science & Business Media, 1997.
[25] Diogo A. Gomes, Edgard Pimentel, Héctor Sánchez-Morgado, Time-dependent mean-field games in the superquadratic case, ESAIM Control Optim. Calc. Var. 22 (2013) 562–580.

[26] Diogo A. Gomes, Edgard A. Pimentel, Héctor Sánchez-Morgado, Time-dependent mean-field games in the subquadratic case, Commun. Partial Differ. Equ. 40 (1) (2015) 40–76.

[27] Oliver Guéant, Mean field games equations with quadratic Hamiltonian: a specific approach, Math. Models Methods Appl. Sci. 22 (09) (2012).

[28] Olivier Guéant, Jean-Michel Lasry, Pierre-Louis Lions, Mean Field Games and Applications, Springer Berlin Heidelberg, Berlin, Heidelberg, 2011, pp. 205–266.

[29] Minyi Huang, Roland P. Malhamé, Peter E. Caines, Large population stochastic dynamic games: closed-loop McKean-Vlasov systems and the Nash certainty equivalence principle, Commun. Inf. Syst. 6 (3) (2006) 221–252.

[30] Jean-Michel Lasry, Pierre-Louis Lions, Mean field games, Jpn. J. Math. 2 (1) (2007) 229–260.

[31] Peter D. Lax, Functional Analysis, Wiley, 2002.

[32] Alex Tong Lin, Yat Tin Chow, Stanley Osher, A splitting method for overcoming the curse of dimensionality in Hamilton-Jacobi equations arising from nonlinear optimal control and differential games with applications to trajectory generation, arXiv preprint, arXiv:1803.01215, 2018.

[33] Alex Tong Lin, Samy Wu Fung, Wuchen Li, Levon Nurbekyan, Stanley J. Osher, Apac-net: alternating the population and agent control via two neural networks to solve high-dimensional stochastic mean field games, arXiv preprint, arXiv:2002.10113, 2020.

[34] Siting Liu, Matthew Jacobs, Wuchen Li, Levon Nurbekyan, Stanley J. Osher, Computational methods for nonlocal mean field games with applications, arXiv preprint, arXiv:2004.12210, 2020.

[35] Lars Ruthotto, Stanley J. Osher, Wuchen Li, Levon Nurbekyan, Samy Wu Fung, A machine learning framework for solving high-dimensional mean field game and mean field control problems, Proc. Natl. Acad. Sci. USA 117 (17) (2020) 9183–9193.