MATEMÁTICA
ESPOL – FCNM JOURNAL

2026, Vol. 24, No.1 Enero - Junio

https://www.revistas.espol.edu.ec/index.php/matematica/issue/archive

Optimal Control of the Fisher Equation: A Comparison of Steepest Descent and Quasi-Newton Methods

Cristhian Núñez Ramos * and Christian Pizanán Cárdenas

Recepción: 18/09/2025 **Aceptación:** 20/01/2026 **Publicación:** 27/01/2026

Abstract This article examines the performance of the steepest descent and quasi-Newton methods, both incorporating line search strategies based on the Armijo rule and Wolfe conditions, for solving an optimal control problem governed by the Fisher equation. To handle inequality constraints, a penalty method is employed to reformulate them as equalities. The existence of solutions is analyzed, and the optimal control is characterized. Additionally, a sensitivity analysis is conducted with respect to key algorithmic parameters to assess the robustness of the proposed approach. Numerical experiments are conducted to compare the two methods, highlighting their features, advantages, and limitations.

Keywords Steepest descent · Quasi-Newton · Line search · Fisher equation · Optimal control problem

2020 Mathematics Subject Classification 35Q93 (primary) · 49J20 · 49M37

Corresponding Author*: Cristhian Núñez Ramos
Pontificia Universidad Católica de Chile, Santiago-Chile, e-mail: aanunez6@uc.cl

Christian Pizanán Cárdenas
Pontificia Universidad Católica del Ecuador - Sede Esmeraldas. Esmeraldas-Ecuador, e-mail: chpizanan@pucese.edu.ec

1 Introduction

Optimization methods have played a central role in the advancement of knowledge and solving problems in a wide range of scientific disciplines, including population dynamics [10] and fluid dynamics [6, 12], to name a few. In population systems, Fisher's equation is a classical model in population forecasting [7, 5], which captures the complex interactions. Despite the challenges posed by this model for being non-linear, the unconstrained optimization methods [14] provide effective tools to tackle it.

Among optimization methods, the steepest descent and quasi-Newton techniques stand out [16] and are widely recognized for their effectiveness in unconstrained optimization. Both rely on gradient information to iteratively refine approximations, converging toward optimal or near-optimal solutions. To improve their performance, they are frequently combined with line search strategies [17], which determine suitable step sizes and enhance the convergence efficiency.

The usefulness of these optimization approaches goes beyond theoretical analysis, finding applications in practical and socially relevant problems such as disease control [13] and resource management [9]. For example, in epidemiology, optimization techniques have been used to design efficient vaccination strategies or to allocate limited medical resources [8]. In environmental management, they have supported the optimization of harvest quotas or the design of conservation strategies that help preserve ecological balance [15].

By integrating optimization methods with ecological modeling, researchers have enhanced their understanding of complex systems and strengthened decision-making processes aimed at achieving sustainable outcomes [11]. Optimization-based approaches facilitate informed decision-making by systematically exploring trade-offs under uncertainty. [19].

The rest of this paper is organized as follows: Section 2 introduces the statement of the optimal control problem governed by the Fisher equation. Section 3 establishes the characterization of the optimal control. Section 4 describes the optimization methods, and Section 5 presents the numerical simulations, including a sensitivity analysis of algorithmic parameters. Finally, Section 6 summarizes the main conclusions.

2 Statement of the Problem

We consider the following semi-discretized version of the problem presented in [18]. The objective is to minimize a quadratic cost functional subject to a semilinear partial differential equation, formulated as follows

$$(\mathbf{P})\begin{cases}\displaystyle\min_{(u,y)} J(u,y) := \tfrac{1}{2}(y(\mathrm{T})-y_\Omega)^T M(y(\mathrm{T})-y_\Omega) + \tfrac{1}{2}\int_0^{\mathrm{T}} u(t)^T M u(t)\,\mathrm{d}t,\\[1em] \qquad\qquad\qquad\text{subject to:}\\[1em] M y_t(t) + A y(t) + M\,\mathrm{diag}(u(t))y(t) = M y(t) - M y(t)^2,\\ \qquad\qquad 0 \le u(t) \le 1,\end{cases}$$

For all $t \in (0,T]$, where M and A denote the mass and stiffness matrices obtained from the discretization of linear finite elements, and $\mathrm{diag}(u(t))$ represents the diagonal matrix formed by the components of $u(t)$. Both M and A are assumed to be symmetric and positive definite. The objective functional J is minimized over the given set $U_{\mathrm{ad}} \times \mathbb{R}^n$, where

$$U_{ad} = \{u(t) \,|\, 0 \le u(t) \le 1 \text{ for all } t \in [0,T]\}\,.$$

The semilinear equation associated with this optimal control problem, referred to as Fisher's equation, is considered in its semi-discretized form. The resulting optimization model constitutes a constrained problem, which is generally more challenging to solve. In the following section, we derive an equivalent unconstrained formulation of problem (**P**).

2.1 Penalty method

We use the penalty method to transform the constrained problem into an unconstrained one. This approach modifies the original objective functional by adding a penalty term, known as the penalty function, which imposes a cost for violating the constraints. Specifically, the penalty term is composed of a penalty parameter multiplied by a function that quantifies the extent of constraint violation. This measure is strictly positive when the constraints are violated and reduces to zero when the constraints are satisfied.

We denote the resulting unconstrained problem associated with (**P**) as follows:

$$(\mathbf{P}_1)\begin{cases}\displaystyle\min_{(u,y)} J(u,y) := \frac{1}{2}(y(T)-y_\Omega)^T M(y(T)-y_\Omega) + \frac{1}{2}\int_0^T u(t)^T M u(t)\,\mathrm{d}t\\ \qquad\qquad +\varphi_1(u(t)) + \varphi_2(u(t))\\ \text{subject to:}\\ M y_t(t) + A y(t) + M\,\mathrm{diag}(u(t))\,y(t) = M y(t) - M y(t)^2,\end{cases}$$

for all $t \in [0,T]$, where the penalty terms φ_1 and φ_2 are defined as follows:

$$\varphi_1(u(t)) = \frac{\gamma}{2}\int_0^T \max\{-u(t),0\}^T M \max\{-u(t),0\}\,\mathrm{d}t,$$

$$\varphi_2(u(t)) = \frac{\gamma}{2}\int_0^T \max\{u(t)-1,0\}^T M \max\{u(t)-1,0\}\,\mathrm{d}t.$$

2.2 Fully discrete problem

We fully discretize the problem ($\mathbf{P}_1$) using a finite difference scheme to allow computations within finite-dimensional spaces. To this end, the time interval $[0,T]$ is partitioned into m subintervals such that $0 = t_1 < t_2 < \cdots < t_m = T$. For reasons of numerical stability, we adopt the implicit Euler method for time integration [20]. Moreover, the time integral is approximated using a Riemann sum. This leads to a fully discrete reformulation of problem ($\mathbf{P}_1$), denoted as follows:

$$(\mathbf{P}_2)\begin{cases} \displaystyle\min_{(u,y)} \; J(u,y) := \frac{1}{2}(y_T - y_\Omega)^T M (y_T - y_\Omega) + \frac{\delta t}{2} u^T M_m u + \varphi_1(u) + \varphi_2(u) \\ \text{subject to:} \\ B_m y + A_m y + M_m \operatorname{diag}(u)\, y = M_m y - M_m y^2, \end{cases}$$

where the penalty terms φ_1 and φ_2 are defined as:

$$\varphi_1(u) = \frac{\gamma\,\delta t}{2} \max\{-u,0\}^T M_m \max\{-u,0\},$$

$$\varphi_2(u) = \frac{\gamma\,\delta t}{2} \max\{u-1,0\}^T M_m \max\{u-1,0\},$$

where M and A are matrices of order $n^2 \times n^2$, with n denoting the number of spatial subdivisions per coordinate direction. The control and state variables are denoted as vectors $u = [u_1, u_2, \ldots, u_m]^T$ and $y = [y_1, y_2, \ldots, y_m]^T$, both lying in $\mathbb{R}^N$, where $N = n^2 m$, and each $u_i, y_i \in \mathbb{R}^{n^2}$ represents the snapshot at time step $i = 1, \ldots, m$. The block matrices involved are given by:

$$M_m = \begin{bmatrix} M & 0 & 0 \\ 0 & \ddots & \ddots \\ \vdots & \ddots & \ddots \end{bmatrix}, \quad A_m = \begin{bmatrix} A & 0 & 0 \\ 0 & \ddots & \ddots \\ \vdots & \ddots & \ddots \end{bmatrix}, \quad B_m = \frac{1}{\delta t}\begin{bmatrix} M & 0 & 0 \\ -M & \ddots & \ddots \\ 0 & \ddots & \ddots \end{bmatrix},$$

where M_m, A_m, and B_m are matrices of size $N \times N$.

The penalty parameter γ is chosen to be sufficiently large to enforce the box constraints on the control, while avoiding excessive ill-conditioning of the resulting optimization problem. In all experiments of next section, we set $\gamma = 1$, which was found to provide a good compromise between accuracy and numerical stability.

3 Characterization of the optimal control problem

3.1 Existence of optimal controls.

A central question is the existence of a solution to the optimal control problem. To address this, we begin by introducing the concept of a local solution to problem ($\mathbf{P}_2$).

Definition 3.1 *A function $\bar{u} \in \mathbb{R}^N$ is called a* local optimal control *for problem ($\boldsymbol{P}_2$) if*

$$\hat{J}(\bar{u}, y(\bar{u})) \leq \hat{J}(u, y(u))$$

for all u in a neighborhood of $\bar{u}$.

The residual function is defined as follows:

$$e(u, y) := B_m y + A_m y + M_m \operatorname{diag}(u)\, y - M_m y + M_m y^2.$$

Note that both the cost functional J and the residual function e are continuously differentiable. Suppose $z = (\bar{u}, \bar{y})$ is a local solution of ($\mathbf{P}_2$) and that the partial derivative $e_y(\bar{u}, \bar{y})$ is a bijective linear operator. Then, by the Implicit Function Theorem (cf. [1]), there exists a neighborhood of $\bar{u}$ in which there is a unique mapping $y(u)$ satisfying

$$e(u, y(u)) = 0, \tag{1}$$

and this mapping is continuously differentiable.

If, for every $u \in \mathbb{R}^N$, equation (1) admits a unique solution $y(u)$, we can define *control-to-state operator S*, which maps a control u to its corresponding state y as a solution of the semi-discrete Fisher equation:

$$\begin{aligned} S : \mathbb{R}^N &\to \mathbb{R}^N, \\ u &\mapsto S(u) := y. \end{aligned}$$

The operator S is continuously differentiable, but generally nonlinear due to the nonlinearity of the state equation. By substituting $y = S(u)$ into the cost functional of problem ($\mathbf{P}_2$), we obtain the corresponding reduced optimization problem:

$$\min_{u \in \mathbb{R}^N} f(u) := \frac{1}{2}\left(S(u)_T - y_\Omega\right)^T M \left(S(u)_T - y_\Omega\right) + \frac{1}{2} u^T M_m u + \varphi_1(u) + \varphi_2(u), \tag{2}$$

so that $f(u) = J(S(u), u)$.

Theorem 1. *There exists an optimal control of the reduced optimization problem* (2) *and consequently of problem ($\boldsymbol{P}_2$).*

Proof. Since $f(u) \geq 0$, the infimum

$$j := \inf_{u \in \mathbb{R}^n} f(u) \tag{3}$$

exists, and there is a sequence $\{u_n\}_{n=1}^{\infty} \subset \mathbb{R}^N$ such that $f(u_n) \to j$ as $n \to \infty$. Since S is continuous, f is also continuous, and the sequence $\{u_n\}_{n=1}^{\infty}$ is bounded; otherwise, we would have $f(u_n) \to \infty$ as $n \to \infty$. By the Bolzano–Weierstrass theorem (cf. [4]), some subsequence $\left\{u_{n_k}\right\}_{k=1}^{\infty}$ converges to some $\bar{u} \in \mathbb{R}^N$, that is,

$$u_{n_k} \to \bar{u}.$$

The continuity of f ensures that

$$f(\bar{u}) = \lim_{k\to\infty} f(u_{n_k}) = j.$$

□

3.2 *First order optimality conditions*

In this section, we derive the first-order optimality conditions that any solution to problem ($\mathbf{P}_2$) must satisfy. Since the objective functional f is differentiable, any optimal control $\bar{u}$ of ($\mathbf{P}_2$) must satisfy

$$\nabla f(\bar{u})^T h = \nabla_y J(S(\bar{u}), \bar{u})[S'(\bar{u})h] + \nabla_u J(S(\bar{u}), \bar{u})h = 0,$$

for all $h \in \mathbb{R}^N$. However, computing the derivative S' explicitly may be computationally expensive. To avoid this, we introduce the following definition.

Definition 3.2 *Let $\bar{u} \in \mathbb{R}^N$ be an optimal control for ($\mathbf{P}_2$), and let $\bar{y} = S(\bar{u})$ denote the associated state. A function $\bar{p} \in \mathbb{R}^N$ is called an adjoint state if it satisfies*

$$e_y(\bar{y}, \bar{u})^T p = \nabla_y J(\bar{y}, \bar{u}), \tag{4}$$

$$e_u(\bar{y}, \bar{u})^T p = \nabla_u J(\bar{y}, \bar{u}), \tag{5}$$

where $e(y, u) := B_m y + A_m y + M_m \mathrm{diag}(u) y - M_m y + M_m y^2$. Here, this residual is defined by bringing all terms to the left-hand side.

The introduction of the adjoint state simplifies the optimality system, avoiding the explicit computation of S'. By the Implicit Function Theorem, the control-to-state operator S is continuously differentiable in a neighborhood of $\bar{u}$, which implies that $e_y(\bar{y}, \bar{u})$ is an invertible linear operator.

We now present the first-order necessary optimality conditions for problem ($\mathbf{P}_2$).

Theorem 2. *Let $\bar{u} \in \mathbb{R}^N$ be an optimal control for ($\mathbf{P}_2$) and $\bar{y} \in \mathbb{R}^N$ the corresponding state satisfying:*

$$B_m \bar{y} + A_m \bar{y} + M_m \mathrm{diag}(\bar{u}) \bar{y} - M_m \bar{y} + M_m \bar{y}^2 = 0. \tag{6}$$

Then, there exists an adjoint state $p \in \mathbb{R}^N$ such that

$$-B_m p + A_m p + M_m \mathrm{diag}(\bar{u}) p - M_m p + 2M_m \mathrm{diag}(\bar{y}) p = 0, \qquad (7)$$
$$M_m \mathrm{diag}(p)\bar{y} - \delta t M_m \bar{u} - \delta t \gamma M_m \max\{-\bar{u}, 0\} - \delta t \gamma M_m \max\{\bar{u} - 1, 0\} = 0, \qquad (8)$$

where $\mathrm{diag}(\bar{u})$ *and* $\mathrm{diag}(p)$ *denote diagonal matrices with entries from* $\bar{u}$ *and* p*, respectively.*

Proof. By the Implicit Function Theorem [21], the control-to-state operator S is continuously differentiable near $\bar{u}$. Therefore, there exists an adjoint state $p \in \mathbb{R}^N$ that satisfies equations (4) and (5).

Computing the derivatives explicitly, we obtain:

$$\begin{aligned}
e_y(\bar{y}, \bar{u})^T h &= [B_m p + A_m p + M_m \mathrm{diag}(\bar{u}) p - M_m p + 2M_m \mathrm{diag}(\bar{y}) p]^T h, \\
e_u(\bar{y}, \bar{u})^T h &= [M_m \mathrm{diag}(p)\bar{y}]^T h, \\
\nabla_y J(\bar{y}, \bar{u})^T h &= [M y_T - y_\Omega]^T h_T, \\
\nabla_u J(\bar{y}, \bar{u})^T h &= [\delta t M_m \bar{u} - \delta t \gamma M_m \max\{-\bar{u}, 0\} + \delta t \gamma M_m \max\{\bar{u} - 1, 0\}]^T h,
\end{aligned}$$

for all $h \in \mathbb{R}^N$.

From equation (5), we immediately obtain (8). Additionally, by equating both sides of equation (4), we get:

$$[B_m p + A_m p + M_m \mathrm{diag}(\bar{u}) p - M_m p + 2M_m \mathrm{diag}(\bar{y}) p]^T h = [M y_T - y_\Omega]^T h_T.$$

This yields the terminal condition $p_T = M y_T - y_\Omega$, and the adjoint equation:

$$-B_m p + A_m p + M_m \mathrm{diag}(\bar{u}) p - M_m p + 2M_m \mathrm{diag}(\bar{y}) p = 0.$$

3.3 Numerical solution of state and adjoint equations.

To compute the state and adjoint variables associated with any given control $u \in \mathbb{R}^N$, we propose Algorithms 2 and 3. These procedures discretize the time interval $[0, T]$ using the implicit Euler method and linearize the nonlinear Fisher equation through Newton's method. Let δt denote the time step. The algorithms are detailed in **Algorithm 1** and **Algorithm 2** that can be found in the Appendix.

4 Optimization Methods

In this section, we address the problem

$$\min_{u \in \mathbb{R}^n} f(u).$$

The algorithms designed to solve this problem are iterative and start from an initial guess u_0. At each iteration k, the goal is to find a descent direction d_k that satisfies

$$f(u_k + \alpha_k d_k) < f(u_k),$$

for some step size $\alpha_k > 0$.

Once a descent direction is determined, it is necessary to select a step size $\alpha_k > 0$ that satisfies the following.

$$\alpha_k = \operatorname{argmin}_{\alpha>0} f(u_k + \alpha d_k).$$

However, since problem $(\mathbf{P}_2)$ is nonconvex, computing the exact minimizer is computationally expensive. Therefore, an inexact step size α_k is usually chosen using the Armijo rule, which proceeds as in **Algorithm 3.** An alternative step size selection method employs Wolfe conditions, described in the **Algorithm 4**. Both line-search algorithms can be found in the Appendix. See [2] for further details.

4.1 *The steepest descent method*

To compute the descent directions, we consider $d = -\nabla f(u_k)$, where

$$d = M_m \operatorname{diag}(p_k) y_k - \delta t M_m u_k + \delta t \gamma M_m \max\{-u_k, 0\} - \delta t \gamma M_m \max\{u_k - 1, 0\}.$$

By combining this direction with an inexact line search strategy of Armijo type (either using the Armijo rule or Wolfe conditions), a globally convergent behavior can be achieved.

The steepest descent algorithm with line search to solve the minimization problem is outlined in **Algorithm 5** of Appendix.

To justify the convergence of the proposed optimization methods, we assume local Lipschitz continuity of the gradient of the reduced objective function. This is stated in the following theorem:

Theorem 3. *The gradient ∇f is locally Lipschitz continuous.*

Proof. Although ∇f may not be classically differentiable, it is differentiable in the Newton sense (cf. [2]). As a consequence, the local Lipschitz continuity of ∇f follows.

□

4.2 *Quasi-Newton Methods*

Another strategy for obtaining a descent direction involves using the Hessian matrix computed by the Newton method (cf. [2]). However, calculating the exact Hessian can be computationally expensive. To address this, we adopt quasi-Newton methods, which use an approximation of the Hessian matrix, denoted by H_k, to improve numerical efficiency.

In particular, we consider the Symmetric Rank-1 (SR1) and BFGS methods [3], where the descent direction is defined as

$$d_k = -H_k^{-1}\nabla f(u_k),$$

with $s_k = u_{k+1} - u_k$, $g_k = \nabla f(u_{k+1}) - \nabla f(u_k)$, and H_k updated as follows:

- **Symmetric Rank-1 (SR1) method:**

$$H_{k+1} = H_k + \frac{(g_k - H_k s_k)(g_k - H_k s_k)^T}{(g_k - H_k s_k)^T s_k}.$$

- **BFGS method:**

$$H_{k+1} = H_k + \frac{g_k g_k^T}{g_k^T s_k} - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k}.$$

It should be noted that the SR1 update does not always guarantee a descent direction. The SR1 algorithm combined with Wolfe conditions proceeds as in **Algorithm 6** of Appendix. While the BFGS algorithm, which guarantees the positive definiteness of H_k, is presented in **Algorithm 7.**

5 Numerical tests

We study an optimal control problem in $\mathbb{R}^2$, which is solved using the Steepest Descent, SR1, and BFGS methods (the two last ones are quasi-Newton methods) combined with line search strategies (Armijo's rule and Wolfe's strategy). We analyze the convergence and performance of the sequences generated by each method. The computational domain is the unit square mesh $[0,1]\times[0,1] \subset \mathbb{R}^2$. The problem is the following

$$(\mathbf{P}_3)\begin{cases} \min\limits_{(u,y)} J(u,y) := \frac{1}{2}(y_\mathrm{T} - 0.5)^T M (y_\mathrm{T} - 0.5) + \frac{\delta t}{2} u^T M_m u + \varphi_1(u) + \varphi_2(u) \\ \text{subject to:} \\ B_m y + A_m y + M_m \operatorname{diag}(u) y = M_m y - M_m y^2, \end{cases}$$

with φ_1 and φ_2 given as follows:

$$\varphi_1(u) = \frac{\gamma\delta t}{2}\max\{-u,0\}^T M_m \max\{-u,0\},$$
$$\varphi_2(u) = \frac{\gamma\delta t}{2}\max\{u-1,0\}^T M_m \max\{u-1,0\}.$$

The following considerations are made:

- M and A are the mass and stiffness matrices associated with the linear finite element method.
- The initial value u_0 is taken as a constant vector in which each component equals 0.2.
- For the SR1 and BFGS methods, we set H_0 as the identity matrix.
- The final time $T = 1$ is considered in all experiments. This value is used in the time discretization step.
- In **Algorithm 4** (line search with Wolfe conditions), we take $\gamma_a = 10^{-4}$ and $\beta = 2 \times 10^{-1}$.
- To penalize the control variable, we set $\gamma = 1$, which ensures that the control is not costly.
- The notation for the temporal and spatial discretization parameters is m and n, respectively.
- All numerical experiments were performed on a workstation equipped with an Intel Core i7 processor (3.4 GHz), 16 GB of RAM, running Linux Ubuntu 22.04. The algorithms were implemented in python, without parallelization.
- A minimal and fully reproducible implementation of the numerical experiments presented in this section is publicly available at `https://github.com/CrisitoNunez12/fisher-optimal-control-quasi-newton`.

The spatial discretization parameter n is selected as the smallest value that accurately captures the spatial structure of the solution while keeping the computational cost moderate. To verify that the observed convergence behavior is not mesh-dependent, additional experiments with larger values of n are reported in the sensitivity study. The stopping tolerance `tol` $= 10^{-4}$ is used throughout, which is sufficient to ensure stabilization of both the objective function and the control variable.

Problem ($\mathbf{P}_3$) is solved using **Algorithm 5**. The approximated solution obtained with the steepest descent method and Armijo's rule is shown in Figure 1. The approximations of the optimal state and adjoint state are displayed in Figures 2 and 3, respectively.

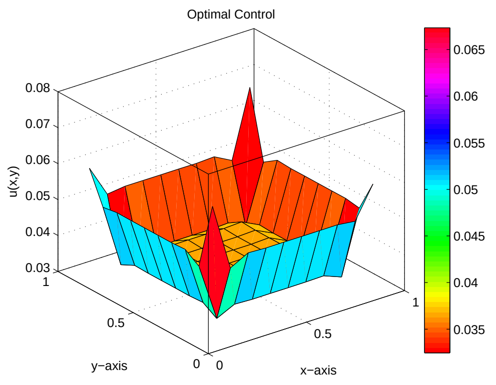

Fig. 1: Approximated optimal control at final time $T = 1$ with $\gamma = 1$, $\texttt{tol} = 10^{-4}$, $n = 10$ and $m = 10$ calculated by Steepest Descent method and Armijo rule.

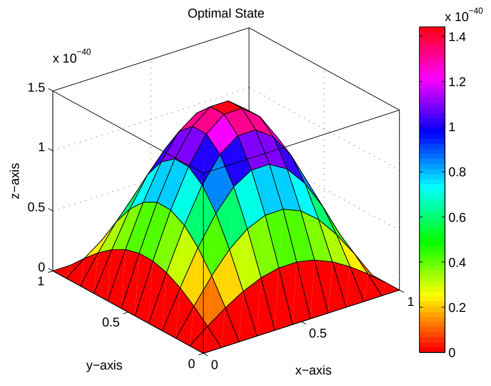

Fig. 2: Approximated optimal state at time $t = 1$ with $\gamma = 1$, $\texttt{tol} = 10^{-4}$, $n = 10$ and $m = 10$ calculated by Steepest Descent method and Armijo rule.

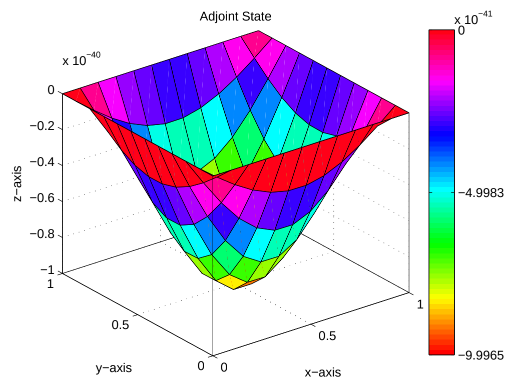

Fig. 3: Approximated adjoint state at initial time $t = 0$ with $\gamma = 1$, $\texttt{tol} = 10^{-4}$, $n = 10$ and $m = 10$ calculated by Steepest Descent method and Armijo rule.

Problem $(\mathbf{P}_3)$ is then solved using **Algorithm 6**. The approximated control obtained with the SR1 method and Armijo's rule is shown in Figure 4. The corresponding optimal and adjoint states are presented in Figures 5 and 6, respectively.

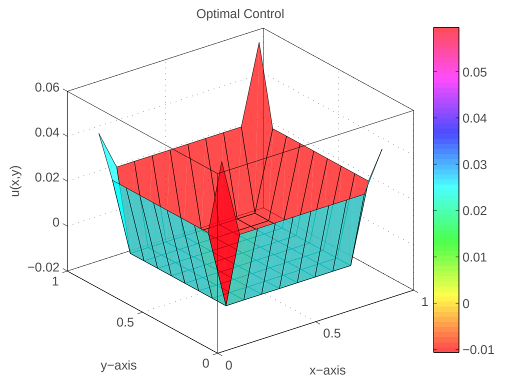

Fig. 4: Approximated optimal control at time $t = 1$ with $\gamma = 1$, $\texttt{tol} = 10^{-4}$, $n = 10$ and $m = 10$ calculated by SR1 method and Armijo rule.

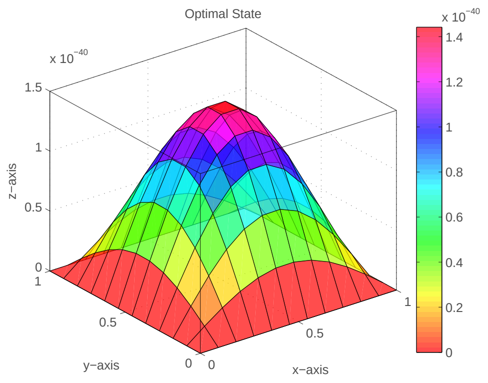

Fig. 5: Approximated optimal state at time $t = 1$ with $\gamma = 1$, `tol` $= 10^{-4}$, $n = 10$ and $m = 10$ calculated by SR1 method and Armijo rule.

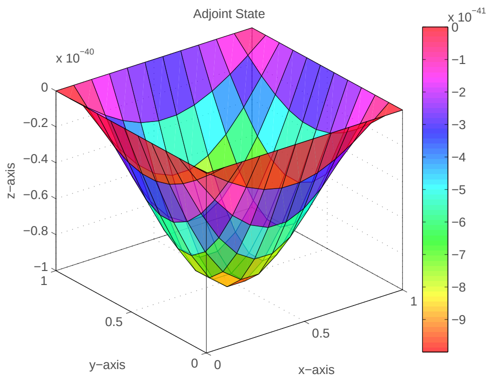

Fig. 6: Approximated adjoint state at initial time $t = 0$ with $\gamma = 1$, `tol` $= 10^{-4}$, $n = 10$ and $m = 10$ calculated by SR1 method and Armijo rule.

The approximated control obtained with the BFGS method and Armijo's rule is similar to that obtained with the steepest descent and SR1 methods.

Stopping criteria and numerical parameters.

All optimization algorithms are terminated according to the same stopping criterion. We define the error in the algorithms as

$$e_k := \|\nabla f(u_k)\|_2, \tag{9}$$

and stop the iteration when $e_k < \texttt{tol}$. Unless otherwise stated, the tolerance is set to $\texttt{tol} = 10^{-4}$. For the solution of the state equation, an implicit Euler scheme is employed in time, and the nonlinear system at each time step is solved by Newton's method with tolerance 10^{-8} in all experiments. For the line-search procedures, the Armijo parameter is fixed to $\gamma_a = 10^{-4}$, while the Wolfe curvature parameter is set to $\beta = 0.2$. The penalty parameter enforcing the box constraints is chosen as $\gamma = 1$. This value provides a good balance between constraint enforcement and numerical stability; its influence is further investigated in the sensitivity study presented at the end of the section.

The number of iterations and the execution time for the three methods using Armijo's rule are reported in Table 1.

Method	Iterations	Execution time
Steepest Descent with Armijo's rule	2048	100.330 seconds
SR1 with Armijo's rule	3	0.246 seconds
BFGS with Armijo's rule	3	0.557 seconds

Table 1: Number of iterations and execution time. Parameters: $n = 10$ and $m = 10$, $\gamma = 1$, $\texttt{tol} = 10^{-4}$.

The number of iterations and execution time for the three methods using Wolfe's conditions are listed in the forthcoming Table 2.

Method	Iterations	Execution time
Steepest descent with Wolfe conditions	-	-
SR1 with Wolfe conditions	3	1.396 seconds
BFGS with Wolfe conditions	3	1.652 seconds

Table 2: Number of iterations and execution time. Parameters: $n = 10$ and $m = 10$, $\gamma = 1$, $\texttt{tol} = 10^{-4}$.

The solution space for the optimization problem with the specified parameters is $\mathbb{R}^N$, where $N = n^2(m) = 1000$ ($n = 10$ and $m = 10$). Armijo's rule and Wolfe's conditions require iterative solutions of the state and adjoint equations. In addition, Newton's method is used to solve the Fisher equation. Notably, while the Steepest Descent method needs approximately 2000 iterations (Figure 7) to satisfy the stopping criterion, the SR1 and BFGS algorithms achieve convergence in only three iterations, see Figures 8 and 9, respectively. Moreover, the execution times of the SR1 and BFGS algorithms are significantly lower than that of the Steepest Descent method, as is shown in Tables 1 and 2.

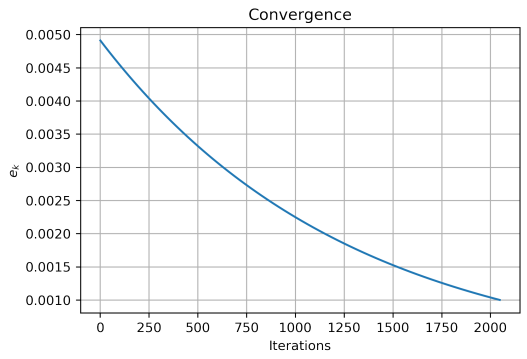

Fig. 7: Convergence of the Steepest Descent model with `tol` $= 10^{-4}$, $n = 10$ and $m = 10$. The error is $\mathrm{e}_k = \|\nabla f(u_k)\|_2$, and the stopping criterion is $\mathrm{e}_k <$ `tol`.

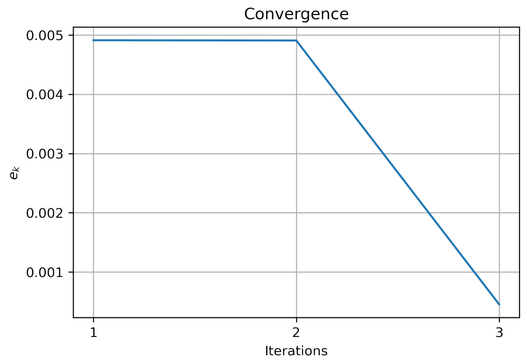

Fig. 8: Convergence of the SR1 model with `tol` $= 10^{-4}$, $n = 10$ and $m = 10$. The error is $\mathrm{e}_k = \|\nabla f(u_k)\|_2$, and the stopping criterion is $\mathrm{e}_k <$ `tol`.

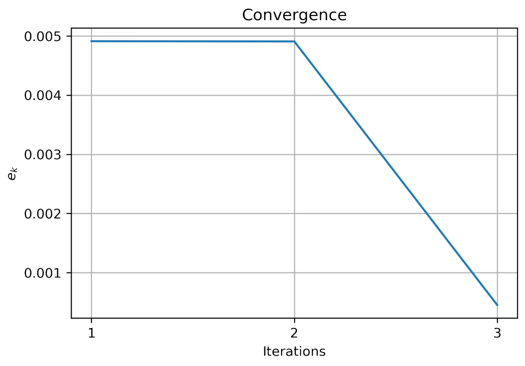

Fig. 9: Convergence of the BFGS model with `tol` $= 10^{-4}$, $n = 10$ and $m = 10$. The error is $\mathrm{e}_k = \|\nabla f(u_k)\|_2$, and the stopping criterion is $\mathrm{e}_k <$ `tol`.

Numerical results obtained with SR1 and BFGS differ only in the state and adjoint equation solves. Here, the solution space for the optimization problem is $\mathbb{R}^N$ with $N = m(n^2) = 6000$ ($n = 20$ and $m = 15$). To examine the convergence behavior of the BFGS method, the tolerance in the stopping criterion is set `tol` $= 10^{-4}$. The convergence results are shown in Figure 10.

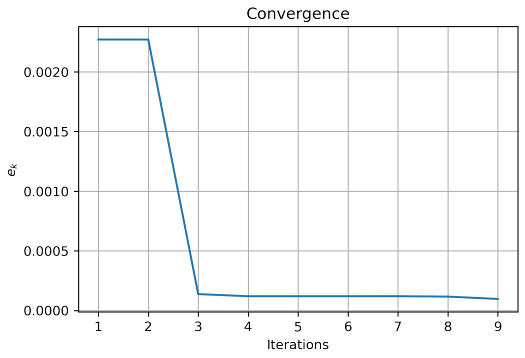

Fig. 10: Convergence of the BFGS model with $n = 20$ and $m = 15$. The error is $\mathrm{e}_k = \|\nabla f(u_k)\|_2$, and the stopping criterion is $\mathrm{e}_k <$ `tol`.

The total runtime to solve problem ($\mathbf{P}_3$) with these parameters is comparable to the previous case and requires 9 iterations.

Method	γ	Iterations	time(seconds)	$e_{\text{final}k}$	viol
BFGS with Armijo's rule	10^{-3}	3	0.617	3.30e-04	7.39e-04
Steepest Descent with Armijo's rule	10^{-3}	2048	184.529	1.00e-03	0.00e+00
SR1 with Armijo's rule	10^{-3}	3	0.227	3.30e-04	7.39e-04
BFGS with Armijo's rule	10^{-2}	3	0.466	3.38e-04	7.39e-04
Steepest Descent with Armijo's rule	10^{-2}	2048	168.202	1.00e-03	0.00e+00
SR1 with Armijo's rule	10^{-2}	3	0.230	3.38e-04	7.39e-04
BFGS with Armijo's rule	1	3	0.483	4.56e-04	7.39e-04
Steepest Descent with Armijo's rule	1	2048	183.378	1.00e-03	0.00e+00
SR1 with Armijo's rule	1	3	0.230	4.56e-04	7.39e-04
BFGS with Armijo's rule	10	4	1.117	4.55e-04	0.00e+00
Steepest Descent with Armijo's rule	10	2048	200.044	1.00e-03	0.00e+00
SR1 with Armijo's rule	10	4	0.478	4.55e-04	0.00e+00
BFGS with Armijo's rule	100	6	1.820	9.17e-04	0.00e+00
Steepest Descent with Armijo's rule	100	2048	197.207	1.00e-03	0.00e+00
SR1 with Armijo's rule	100	6	0.682	9.17e-04	0.00e+00

Table 3: Sensitivity with respect to the penalty parameter γ (fixed $n = 10$, $m = 10$ and `tol` $= 10^{-3}$).

5.1 Sensitivity with respect to algorithmic parameters

In order to assess the robustness of the methods, we conduct a sensitivity analysis with respect to key problem parameters that may influence both convergence behavior and computational cost. Specifically, we examine the impact of the penalty parameter γ, the stopping tolerance `tol`, and the spatial discretization on the performance of the considered optimization methods in Tables 3–5. This analysis allows us to evaluate the stability of the algorithms under parameter variations and to quantify how changes in problem settings affect convergence, runtime, and constraint enforcement.

Table 3 shows that the quasi-Newton methods (BFGS and SR1) are only slightly affected by variations in the penalty parameter γ. For γ ranging from 10^{-3} to 10^{2}, both methods converge in a small number of iterations, with a modest increase in runtime and final error for larger values of γ. In contrast, the steepest descent method consistently requires the maximum number of iterations and exhibits significantly larger computational times, independently of the chosen penalty parameter. Constraint violations (described by column $viol$) decrease as γ increases, confirming the expected strengthening of the penalty effect. The quantity $e_{\text{final}k}$ denotes the value of the stopping criterion at the final iteration.

The influence of the stopping tolerance is reported in Table 4. As expected, tightening the tolerance leads to an increase in the number of iterations and runtime for all methods. However, BFGS and SR1 remain highly efficient, requiring only a few additional iterations to reach stricter tolerances, while steepest descent becomes prohibitively expensive. This highlights the robustness of quasi-Newton methods with respect to the stopping criterion.

Finally, Table 5 examines the effect of spatial mesh refinement. Increasing the discretization level leads to higher computational times for all methods due to the

Method	`tol`	Iterations	time (seconds)	$e_{\text{final}k}$	viol
BFGS with Armijo's rule	10^{-5}	17	4.277	9.78e-05	1.63e-04
Steepest Descent with Armijo's rule	10^{-5}	5181	500.930	1.00e-04	1.58e-05
SR1 with Armijo's rule	10^{-5}	9	0.911	1.05e-04	2.12e-04
BFGS with Armijo's rule	10^{-4}	3	0.617	4.56e-04	7.39e-04
Steepest Descent with Armijo's rule	10^{-4}	2048	188.791	1.00e-03	0.00e+00
SR1 with Armijo's rule	10^{-4}	3	0.264	4.56e-04	7.39e-04

Table 4: Sensitivity with respect to the stopping tolerance `tol` (fixed $n = 10, m = 10$, $\gamma = 1$).

Method	Discretization	Iterations	time(seconds)	$e_{\text{final}k}$	viol
BFGS with Armijo's rule	8	3	0.366	6.11e-04	8.56e-04
Steepest Descent with Armijo's rule	8	1538	129.438	9.99e-04	0.00e+00
SR1 with Armijo's rule	8	3	0.211	6.11e-04	8.56e-04
BFGS with Armijo's rule	10	3	0.727	4.56e-04	7.39e-04
Steepest Descent with Armijo's rule	10	2048	178.405	1.00e-03	0.00e+00
SR1 with Armijo's rule	10	3	0.263	4.56e-04	7.39e-04
BFGS with Armijo's rule	12	3	1.262	3.54e-04	6.46e-04
Steepest Descent with Armijo's rule	12	2577	269.761	1.00e-03	0.00e+00
SR1 with Armijo's rule	12	3	0.242	3.54e-04	6.46e-04

Table 5: Sensitivity with respect to the spatial mesh resolution (fixed $m = 10, \gamma = 1$, `tol`$=10^{-3}$).

growth in problem size, while the number of iterations for BFGS and SR1 remains essentially unchanged. Steepest descent, on the other hand, exhibits a substantial increase in runtime as the mesh is refined. Overall, these results indicate that the quasi-Newton methods are largely insensitive to discretization changes and scale favorably with problem resolution.

6 Conclusions

In this paper, we studied an optimal control problem governed by the Fisher equation. We established the existence of solutions and derived the first-order optimality conditions that characterize the problem. To address the numerical solution, we formulated the problem using a penalty method, fully discretized it, and proposed algorithms for solving both the state and adjoint equations.

We implemented and tested three optimization methods: Steepest Descent, Symmetric Rank-1 (SR1), and BFGS. The Steepest Descent method required repeated evaluations of the cost functional during the Armijo line search, which was computationally expensive because both the state and adjoint equations had to be solved at each iteration. Although Steepest Descent converged slowly, it ensured that each search direction was a descent direction, thus guaranteeing convergence.

In contrast, the SR1 and BFGS methods, belonging to the class of quasi-Newton schemes, exhibited significantly faster convergence. The SR1 method was efficient in the numerical experiments but did not always guarantee a descent direction. The BFGS method, combined with Wolfe conditions, achieved convergence rates comparable to SR1 while guaranteeing descent directions. The numerical tests confirmed that SR1 and BFGS drastically reduced both the number of iterations and the execution time compared to the Steepest Descent method.

Regarding the sensitivity analysis, this confirms that BFGS and SR1 provide robust and efficient performance across a wide range of parameter values, whereas steepest descent is strongly affected by parameters of the problem and computational resolution.

Finally, we observed that linearizing the Fisher equation through Newton's method at each time step introduces substantial computational costs (for the state and adjoint equations). For this reason, future research should explore alternative approaches, such as primal–dual strategies, to reduce these computational demands and to extend the applicability of the proposed framework to more complex problems, including higher-dimensional domains or strongly nonlinear reaction terms, to name a few. In addition, a systematic sensitivity analysis with respect to the physical parameters of the model, such as diffusion and reaction coefficients, could be carried out to further assess the robustness of the approach.

Appendix

State and adjoint equations

Algorithm 1 State Equation

```
Input: Number of time steps $m$, initial condition $y_0$, control $u \in \mathbb{R}^n$, and Newton tolerance tol $> 0$.
Output: Approximated solution to the state equation, denoted by $w$.
1. For $k = 1, 2, \dots, m-1$, perform the following:
2. Initialize $w = y_{k+1}$, $v = y_k$.
3. Compute the residual:
       $F(w) = (M + \delta t\, A + \delta t\, M\, \mathrm{diag}(u) - \delta t\, M)w + \delta t\, M w^2 - M v,$
4. Compute the Jacobian:
       $F'(w) = M + \delta t\, A + \delta t\, M\, \mathrm{diag}(u) - \delta t\, M + 2\delta t\, M\, \mathrm{diag}(w),$
5. Set an initial guess for $w$.
6. While $\|w - v\| >$ tol, do:
7. Solve the linear system: $F'(w)\, dw = -F(w)$.
8. Update: $w \leftarrow w + dw$.
9. Set $v = w$.
10. End While.
```

Algorithm 2 Adjoint Equation

```
Input: Number of time steps $m$, final condition $\tilde{p}_T$, control $u \in \mathbb{R}^n$, and state $y \in \mathbb{R}^n$.
Output: Approximated solution to the adjoint equation, denoted by $\tilde{p}$.
1. For $k = m-1, m-2, \dots, 1$, perform the following:
2. Define the system matrix:
       $G = M + \delta t\, A + \delta t\, M\, \mathrm{diag}(u_{k+1}) - \delta t\, M + 2\delta t\, M\, \mathrm{diag}(y_{k+1}),$
3. Define the right-hand side: $b = M \tilde{p}_k$.
4. Solve the linear system: $G \tilde{p}_{k+1} = b$.
5. End For.
```

Line search

Algorithm 3 Armijo Rule

```
Input:  A descent direction $d_k$.
Output:  An approximate step size $\alpha$.
1.  Initialize $\alpha = 1$ and $k = 0$.
2.  repeat
3.      Set $\alpha = \frac{1}{2^k}$, then increment $k \leftarrow k + 1$.
4.  until
        $$f(u_k + \alpha d_k) - f(u_k) \le \gamma_a \alpha \nabla f(u_k)^T d_k,$$
    where $0 < \gamma_a < 1$.
```

Algorithm 4 Line Search with Wolfe Conditions

```
Input:  A descent direction $d_k$.
Output:  An approximate step size $\alpha$.
1.  Initialize $\alpha = 1$ and $k = 0$.
2.  repeat
3.      Set $\alpha = \frac{1}{2^k}$, then increment $k \leftarrow k + 1$.
4.  until the conditions
        $$f(u_k + \alpha d_k) - f(u_k) \le \gamma_a \alpha \nabla f(u_k)^T d_k,$$
    and
        $$\nabla f(u_k + \alpha d_k)^T d_k \ge \beta \nabla f(u_k)^T d_k,$$
    are both satisfied, where $0 < \gamma_a < \beta < 1$.
```

Optimization Algorithms

Detailed pseudo-code for the steepest descent, SR1, and BFGS algorithms is reported here for completeness.

Algorithm 5 Steepest Descent Method with Line Search

```
1.  Choose an initial guess $u_0$, tolerance tol $> 0$, and set $k = 0$.
2.  **Repeat:**
3.  Compute the state $y_k$ and adjoint state $p_k$ using **Algorithm 1** and **Algorithm 2**, respectively.
4.  Set the descent direction
        $d_k = M_m \operatorname{diag}(p_k) y_k - \delta t M_m u_k + \delta t \gamma M_m \max\{-u_k, 0\} - \delta t \gamma M_m \max\{u_k - 1, 0\}.$
5.  Determine the step size $\alpha_k$ using either **Algorithm 3** (Armijo rule) or **Algorithm 4** (Wolfe conditions).
6.  Update the control:
        $u_{k+1} = u_k + \alpha_k d_k.$
    Recompute $y_{k+1}$ and $p_{k+1}$ using **Algorithm 1** and **Algorithm 2**, and increment $k \leftarrow k + 1$.
7.  **Until**
        $\|\nabla f(u_k)\| <$ tol.
```

Algorithm 6 Symmetric Rank-1 Algorithm with Wolfe Conditions

```
1.  Choose an initial guess $u_0$, tolerance tol $> 0$, and an initial matrix $H_0$. Set $k = 0$.
2.  **Repeat:**
3.  Compute $y_k$ and $p_k$ using **Algorithm 1** and **Algorithm 2**.
4.  Compute the gradient:
        $\nabla f(u_k) = -M_m \operatorname{diag}(p_k) y_k + \delta t M_m u_k - \delta t \gamma M_m \max\{-u_k, 0\} + \delta t \gamma M_m \max\{u_k - 1, 0\}.$
5.  Compute the descent direction:
        $d_k = -H_k^{-1} \nabla f(u_k).$
6.  Determine the step size $\alpha_k$ using **Algorithm 4** (Wolfe conditions).
7.  Update the control:
        $u_{k+1} = u_k + \alpha_k d_k.$
8.  Set $s_k = u_{k+1} - u_k$, $g_k = \nabla f(u_{k+1}) - \nabla f(u_k)$, and update the Hessian approximation:
        $H_{k+1} = H_k + \frac{(g_k - H_k s_k)(g_k - H_k s_k)^T}{(g_k - H_k s_k)^T s_k}.$
9.  Increment $k \leftarrow k + 1$.
10. **Until**
        $\|\nabla f(u_k)\| <$ tol.
```

Algorithm 7 BFGS Algorithm with Wolfe Conditions

```
1.  Choose an initial guess $u_0$, tolerance tol $> 0$, and an initial matrix $H_0$. Set $k = 0$.
2.  **Repeat:**
3.  Compute $y_k$ and $p_k$ using **Algorithm 1** and **Algorithm 2**.
4.  Compute the gradient:
        $\nabla f(u_k) = -M_m \operatorname{diag}(p_k) y_k + \delta t M_m u_k - \delta t \gamma M_m \max\{-u_k, 0\} + \delta t \gamma M_m \max\{u_k - 1, 0\}.$
5.  Compute the descent direction:
        $d_k = -H_k^{-1} \nabla f(u_k).$
6.  Determine the step size $\alpha_k$ using **Algorithm 4** (Wolfe conditions).
7.  Update the control:
        $u_{k+1} = u_k + \alpha_k d_k.$
8.  Set $s_k = u_{k+1} - u_k$, $g_k = \nabla f(u_{k+1}) - \nabla f(u_k)$, and update the Hessian approximation:
        $H_{k+1} = H_k + \frac{g_k g_k^T}{g_k^T s_k} - \frac{H_k s_k s_k^T H_k}{s_k^T H_k s_k}.$
9.  Increment $k \leftarrow k + 1$.
10. **Until**
        $\|\nabla f(u_k)\| <$ tol.
```

References

[1] Haim Brezis. *Functional Analysis, Sobolev Spaces and Partial Differential Equations*. Springer, 2010.

[2] J. C. De los Reyes. *Numerical PDE-Constrained Optimization*. SpringerBriefs in Optimization. Springer International Publishing, 2015.

[3] John E. Dennis and Robert B. Schnabel. *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*. SIAM, Philadelphia, 1996.

[4] Lawrence C. Evans. *Partial Differential Equations*, volume 19. American Mathematical Society, 2022.

[5] M. Gunzburger, L. Hou, and W. Zhu. Modeling and analysis of the forced fisher equation. *Nonlinear Analysis: Theory, Methods and Applications*, 62:19–40, 2005.

[6] Innyoung Kim and Donghyun You. Fluid dynamic control and optimization using deep reinforcement learning. *JMST Advances*, 2024.

[7] John R. King and Philip M. McCabe. On the fisher–kpp equation with fast nonlinear diffusion. *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 459(2038):2529–2546, 2003.

[8] Suzanne Lenhart and J. A. Montero. Optimal control of harvesting in a parabolic system modeling two subpopulations. *Mathematical Models and Methods in Applied Sciences*, 11:1129–1141, 2001.

[9] Suzanne M. Lenhart and Mahadev G. Bhat. Application of distributed parameter control model in wildlife damage management. *Mathematical Models and Methods in Applied Sciences*, 2(04):423–439, 1992.

[10] Ming Li and Qing Xie. Optimal control for a stationary population. *Journal of Mathematics Research*, 4, 2012.

[11] Julien Martin, Matthew Richardson, Davina Passeri, Nicholas Enwright, Simeon Yurek, James Flocks, Mitchell Eaton, Sara Zeigler, Hadi Charkhgard, Bradley Udell, and Elise Irwin. Decision science as a framework for combining geomorphological and ecological modeling for the management of coastal systems. *Ecology and Society*, 28, 03 2023.

[12] Pedro Merino and Cristhian Núñez. Numerical simulation of free boundary biviscous fluids. *Bulletin of Computational Applied Mathematics*, 12(2):xx–xx, 2024. Special Issue V JEM 2023.

[13] Rachael Miller Neilan and Suzanne Lenhart. An introduction to optimal control with an application in disease modeling. *DIMACS Series in Discrete Mathematics*, 75, 10 2010.

[14] Ibrahim Mohd Asrul Hery, Mamat Mustafa, Ghazali Puspa Liza, and Salleh Zabidin. The scaling of hybrid method in solving unconstrained optimization method. *Far East Journal of Mathematical Sciences (FJMS)*, 2015.

[15] Michael Neubert. Marine reserves and optimal harvesting. *Ecology Letters*, 6:843–849, 2003.

[16] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.

[17] Jorge Nocedal and Stephen J. Wright. Line search methods. In *Numerical Optimization*, pages 30–65. Springer, 2006.

[18] Cristhian Núñez. Reducción de un modelo de control óptimo de dinámica poblacional mediante técnicas pod, 2015. Repositorio Digital EPN, Tesis Matemáticas (MAT).

[19] Tu Peng, Xu Yang, Zi Xu, and Yu Liang. Constructing an environmental friendly low-carbon-emission intelligent transportation system based on big data and machine learning methods. *Sustainability*, 12(19):8118, 2020.

[20] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical Mathematics (Texts in Applied Mathematics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[21] Walter Rudin. *Principles of Mathematical Analysis*. McGraw-Hill, New York, 3 edition, 1976.