Numerical and Machine Learning Methods for Corneal Curvature Modeling: A Comparative Study

I. Introduction

I. INTRODUCTION The human cornea provides approximately 65-75Mathematical modeling of corneal curvature presents significant challenges due to the cornea's complex geometry and nonlinear elastic behavior under various loading conditions. Traditional approaches have employed ordinary and partial differential equations to capture corneal deformation physics. This study focuses on the boundary value ordinary differential equation (BVODE) model proposed by Okrasiński and Płociniczak, which incorporates fundamental biomechanical principles through static force balance to describe the corneal surface profile. The motivation stems from the increasing need for accurate computational models for clinical decision-making and the emerging potential of machine learning approaches. The primary objectives are: (1) implement and compare two additional numerical solution methods beyond the reference method of lines approach, (2) develop a machine learning-based solution framework, and (3) evaluate all methods in terms of accuracy, computational efficiency, and practical applicability.

II. literature review

The mathematical modeling of corneal curvature has evolved from simplified geometric models to sophisticated biomechanical approaches incorporating advanced computational techniques [1]. Okrasiński and Płociniczak [2] developed the foundational BVODE model used in this study, utilizing static force balance to derive a nonlinear differential equation that captures corneal deformation physics under pressure loading.

Recent advances have explored various mathematical formulations, including prescribed anisotropic mean curvature equations and two-dimensional nonlinear thin membrane PDE models solved using radial basis functions [3], [4]. These meshless methods demonstrate effectiveness in handling complex geometries and boundary conditions common in biological systems.

Machine learning approaches in ophthalmology have shown remarkable progress. Recent studies have explored ML models for predicting corneal shapes from clinical data, while comprehensive evaluations [5] highlight the growing availability of datasets and algorithms. Specific applications include deep learning models for keratoconus diagnosis [6], corneal stiffness prediction, and transformer-based detection systems [7]. The broader adoption of AI in ophthalmology [8] has been facilitated by digitized ocular images and substantial datasets. Despite these advances, comprehensive comparative studies evaluating traditional numerical methods alongside machine learning approaches for fundamental biomechanical modeling problems remain limited. This work addresses this gap by providing a systematic comparison of multiple solution strategies for the corneal curvature BVODE.

III. Mathematical Model

A. Problem Formulation

The corneal curvature model is derived from a static force balance on the cornea, considering the equilibrium between internal tension forces, elastic restoring forces, and applied pressure. The physical system represents the cornea as a thin membrane under the influence of:

1) Tension (T): Internal stress within the corneal tissue

2) Elastic constant (k): Effective spring constant representing corneal stiffness

3) Applied pressure (P): Intraocular pressure acting on the corneal surface

B. Governing Differential Equation

The dimensional form of the governing equation is:

$$-T\frac{\frac{d^2h}{dx^2}}{\sqrt{1+\left(\frac{dh}{dx}\right)^2}}+kh=\frac{P}{\sqrt{1+\left(\frac{dh}{dx}\right)^2}}$$

where:

- $h(x)$ is the height of the corneal surface relative to the peripheral height
- x is the horizontal coordinate
- The square root terms account for the geometric nonlinearity due to large slopes

C. Nondimensionalization

Using the corneal radius R as the characteristic length scale, dimensionless variables are introduced:

$$h^*=\frac{h}{R},\quad x^*=\frac{x}{R}$$

This leads to the dimensionless BVODE (dropping asterisks):

$$-\frac{d^2h/dx^2}{\sqrt{1+(dh/dx)^2}}+ah=\frac{b}{\sqrt{1+(dh/dx)^2}}$$

where the dimensionless parameters are:

- $a=\frac{R^2k}{T}$: ratio of elastic to tension forces
- $b=\frac{RP}{T}$: ratio of pressure to tension forces

D. Boundary Conditions

The boundary value problem is completed with:

$$h(1) = 0 \quad \text{(zero height at corneal periphery)}$$

$$h'(0) = 0 \quad \text{(zero slope at center due to symmetry)}$$

E. Physical Interpretation

The model captures essential corneal biomechanics:

- The first term represents the curvature-dependent tension force
- The ah term represents elastic restoring forces
- The right-hand side represents pressure loading
- The geometric nonlinearity through $\sqrt{1 + (dh/dx)^2}$ accounts for finite slope effects

IV. Numerical Solution Methods

A. Reference Method: Method Of Lines (MOL)

1) Problem Transformation: The reference method addresses the nonlinear boundary value problem by introducing an artificial time dimension and transforming the steady-state equation into a time-dependent problem: $\frac{\partial h}{\partial t} = F(h)$

where F(h) represents the residual of the original BVODE. The solution evolves in pseudo-time until reaching steady-state $\frac{\partial h}{\partial t} \to 0$, corresponding to the desired corneal shape.

2) Spatial Discretization and Time Integration: The Method of Lines discretizes the spatial domain [0,1] into evenly spaced points and approximates spatial derivatives using high-order finite difference formulas. This converts the PDE into a system of ODEs: $\frac{dh_i}{dt} = f(h_{i-1}, h_i, h_{i+1})$

The system is integrated using the `LSODA` solver with boundary conditions $h'(0) = 0$ and $h(1) = 0$. Starting from a flat initial profile $h = 0.25$, the solution converges when $\max\left(\frac{\partial h}{\partial t}\right) < 10^{-4}$.

3) Convergence Results: The method demonstrates robust convergence:

TABLE I
Convergence Results for MOL Method

Time (t)	h(0,t)	h(0.5,t)	h(1,t)	Max(dh/dt)
0.0	0.250	0.250	0.250	0.750
0.2	0.328	0.254	0.000	0.152
1.0	0.348	0.266	0.000	8×10^{-5}

This reference solution serves as the benchmark for evaluating the accuracy and performance of alternative numerical methods.

B. Method 1: Shooting Method with Runge-Kutta Integration

The first numerical approach employed combines the shooting method with fourth-order Runge-Kutta integration to solve the boundary value problem. This method was selected due to its robustness in handling nonlinear boundary value problems and its previous coverage in the course curriculum.

1) Problem Transformation: The original second-order BVODE:

$$-\frac{d^2h}{dx^2} + ah = \frac{b}{\sqrt{1 + \left(\frac{dh}{dx}\right)^2}}$$

with boundary conditions $h'(0) = 0$ and $h(1) = 0$, is transformed into a system of first-order ODEs to enable Runge–Kutta integration.

Let: $y_1 = h, \quad y_2 = \frac{dh}{dx}$

This yields the first-order system:

$$\frac{dy_1}{dx} = y_2, \quad \frac{dy_2}{dx} = -\frac{b}{\sqrt{1 + y_2^2}} + ay_1$$

2) Shooting Method Implementation: Since this is a boundary value problem rather than an initial value problem, the shooting method is employed to handle the unknown initial condition $h(0)$. The method transforms the BVP into an IVP by:

1) **Initial Guess:** Making an initial estimate for $h(0)$
2) **Forward Integration:** Using Runge–Kutta to integrate from $x = 0$ to $x = 1$
3) **Boundary Check:** Evaluating the boundary condition error at $x = 1$
4) **Iterative Refinement:** Adjusting the initial guess until convergence

3) Secant Method for Root Finding: The secant method is employed to iteratively improve the guess for the unknown initial condition $h(0)$. The algorithm:

1) Select two initial guesses $h_0^{(0)}$ and $h_0^{(1)}$
2) Compute boundary errors F_0 and F_1 after integration
3) Update the guess using the secant formula:

$$h_0^{(n+1)} = h_0^{(n)} - F_n \cdot \frac{h_0^{(n)} - h_0^{(n-1)}}{F_n - F_{n-1}}$$

4) Continue until $|F_n| < \epsilon$ (desired tolerance)

4) Algorithm Implementation: The complete algorithm follows these steps:

Step 1: Initialize parameters a, b, and integration settings, **Step 2**: Select initial guesses for $h(0)$ ensuring one undershoots and one overshoots the boundary condition, **Step 3**: For each guess, integrate the first-order system using RK4, **Step 4**: Evaluate boundary condition error: $F = h(1) - 0$, **Step 5**: Apply secant method to update $h(0)$, **Step 6**: Repeat until convergence criterion is met.

5) Results and Performance: The method successfully converged to the solution with the following characteristics:

- Converged Initial Value: $h(0) = 0.348$
- Execution Time: 0.0114 seconds
- Convergence Tolerance: Achieved desired accuracy in boundary condition satisfaction
- Stability: Robust convergence with proper initial guess selection

The numerical results demonstrated good agreement with the reference solution from the method of lines approach, validating the effectiveness of the shooting method for this nonlinear boundary value problem.

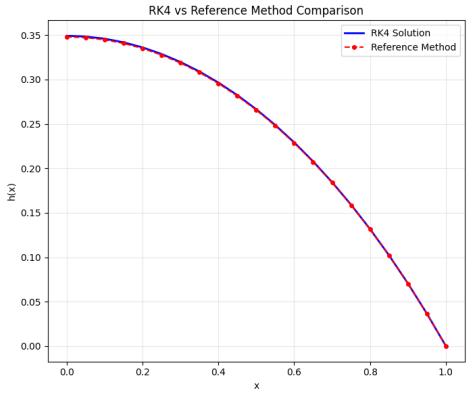

Fig. 1. RK4 shooting method solution (blue) vs. reference MOL solution (red dashed), showing excellent agreement across domain [0,1].

C. Method 2: Finite Difference Method with Damped Newton-Raphson Iteration

The second numerical approach combines finite difference discretization with damped Newton-Raphson iteration to solve the nonlinear boundary value problem. This method was selected for its robust handling of mixed boundary conditions and excellent stability properties for strongly nonlinear systems.

1) Problem Discretization: The continuous domain $[0, 1]$ is discretized into $N = 21$ equally spaced grid points with spacing: $\Delta x = \frac{1}{N-1} = \frac{1}{20} = 0.05$

The grid points are defined as $x_i = i \cdot h$ for $i = 0, 1, 2, ..., N-1$.

2) Finite Difference Approximations: The derivatives in the original differential equation are approximated using central difference formulas:

First derivative (central difference):

$$\left.\frac{dh}{dx}\right|_{x_i} \approx \frac{h_{i+1} - h_{i-1}}{2\Delta x}$$

Second derivative (central difference):

$$\left.\frac{d^2h}{dx^2}\right|_{x_i} \approx \frac{h_{i+1} - 2h_i + h_{i-1}}{(\Delta x)^2}$$

3) Residual Function Formulation: The nonlinear differential equation is transformed into a residual function $F(h)$ at each grid point:

At $i = 0$ (Neumann boundary condition $h'(0) = 0$):

$$F_0 = \frac{h_1 - h_{-1}}{2\Delta x} = 0$$

Using the symmetry condition, this becomes:

$$F_0 = h_1 - h_{-1} = 0$$

At internal points $i = 1$ to $N - 2$:

$$F_i = -\frac{h_{i+1} - 2h_i + h_{i-1}}{(\Delta x)^2\sqrt{1 + \left(\frac{h_{i+1} - h_{i-1}}{2\Delta x}\right)^2}} + ah_i - \frac{b}{\sqrt{1 + \left(\frac{h_{i+1} - h_{i-1}}{2\Delta x}\right)^2}}$$

At $i = N - 1$ (Dirichlet boundary condition $h(1) = 0$):

$$F_{N-1} = h_{N-1} - 0 = h_{N-1}$$

4) Damped Newton-Raphson Implementation: The damped Newton-Raphson method iteratively solves the nonlinear system $F(h) = 0$:

Update equation: $h^{(k+1)} = h^{(k)} - \alpha J^{-1}(h^{(k)})F(h^{(k)})$

where:

- J is the Jacobian matrix
- $\alpha = 0.8$ is the damping factor for stability
- k denotes the iteration number

5) Jacobian Matrix Estimation: The Jacobian matrix is estimated numerically using finite difference perturbations:

$$J_{ij} = \frac{\partial F_i}{\partial h_j} \approx \frac{F_i(h + \epsilon e_j) - F_i(h)}{\epsilon}$$

where $\epsilon = 10^{-5}$ is the perturbation parameter and e_j is the unit vector in the j-th direction.

6) Algorithm Implementation: **Step 1:** Initialize domain discretization with $N = 21$ grid points **Step 2:** Set initial guess as linear profile: $h_i^{(0)} = 0.25(1 - x_i)$ **Step 3:** For each iteration k:

- Compute residual vector $F(\mathbf{h}^{(k)})$
- Estimate Jacobian matrix $J(\mathbf{h}^{(k)})$ numerically
- Solve linear system: $J(\mathbf{h}^{(k)})\Delta\mathbf{h} = -F(\mathbf{h}^{(k)})$
- Update solution: $\mathbf{h}^{(k+1)} = \mathbf{h}^{(k)} + \alpha\Delta\mathbf{h}$ where α is the damping factor (typically $\alpha = 0.8$)

Step 4: Check convergence criteria:

- Residual norm: $\|F\|_\infty < 10^{-8}$
- Solution change: $\|\Delta\mathbf{h}\|_\infty < 10^{-8}$

Step 5: Continue until convergence or maximum iterations reached.

7) Results and Performance: The finite difference method with damped Newton-Raphson iteration demonstrated excellent performance:

- **Convergence:** Achieved in 12 iterations
- **Stabilization:** Solution stabilized after 11 iterations
- **Maximum Absolute Error:** < 0.019418 compared to reference solution
- **Maximum Relative Error:** $< 5.67\%$
- **Execution Time:** Approximately 0.086 seconds
- **Damping Factor:** $\alpha = 0.8$ provided optimal stability

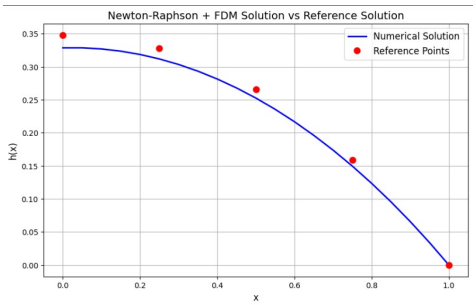

Fig. 2. Finite difference with Newton-Raphson solution (blue) vs. reference MOL points (red), demonstrating excellent accuracy with 12-iteration convergence.

The method showed exponential decay in error norms, confirming robust convergence behavior and numerical stability for this strongly nonlinear system.

V. Machine Learning Approach

A. Physics-Informed Neural Network (PINN) Framework

This approach employs a Physics-Informed Neural Network (PINN) to solve the corneal curvature boundary value problem. PINNs represent a paradigm shift in computational mathematics by embedding physical laws directly into the neural network training process, enabling mesh-free solution of differential equations. The motivation for using machine learning stems from its proven efficiency and accuracy when properly implemented, offering flexibility in handling complex geometries and sparse data scenarios.

B. Problem Reformulation for PINN Implementation

Rather than solving the differential equation directly, the PINN approach reformulates the problem by moving all terms to the left-hand side: $R(x) = -\frac{d^2h/dx^2}{\sqrt{1+(dh/dx)^2}} + ah - \frac{b}{\sqrt{1+(dh/dx)^2}} = 0$ This residual function $R(x)$ should ideally equal zero at every point in the domain. The PINN minimizes this residual across the entire domain while satisfying boundary conditions, providing a natural framework for error evaluation and convergence tracking.

C. Neural Network Architecture

The PINN architecture employs a fully connected feedforward neural network with the following specifications:

Network Structure:

- **Input Layer:** Single neuron accepting spatial coordinate $x \in [0, 1]$
- **Hidden Layers:** 3 hidden layers with 50 neurons each
- **Activation Function:** Hyperbolic tangent (Tanh) for smooth derivatives
- **Output Layer:** Single neuron producing corneal height $h(x)$
- **Total Parameters:** 5,251 trainable parameters

The mathematical representation is:

$$h_{\text{PINN}}(x;\theta) = W_4 \tanh(W_3 \tanh(W_2 \tanh(W_1 x + b_1) + b_2) + b_3) + b_4$$

where $\theta = \{W_i, b_i\}_{i=1}^{4}$ represents all trainable weights and biases.

D. Multi-Component Loss Function

The total loss function combines three essential components: $\mathcal{L}total = \mathcal{L}data + 0.001 \times \mathcal{L}physics + 10 \times \mathcal{L}boundary$

1) Data Loss Component: The data loss measures agreement with the reference Method of Lines (MOL) solution: $\mathcal{L}data = \frac{1}{Ndata} \sum_{i=1}^{N_{data}} |h_{NN}(x_i) - h_{MOL}(x_i)|^2$

2) Physics Loss Component: The physics loss enforces the differential equation at collocation points: $\mathcal{L}physics = \frac{1}{Ncol} \sum_{i=1}^{N_{col}} |R(x_i)|^2$ where the residual $R(x_i)$ is computed using automatic differentiation.

3) Boundary Condition Loss: The boundary loss ensures satisfaction of both boundary conditions:

$$\mathcal{L}_{\text{boundary}} = \left| \frac{\partial h_{\text{NN}}}{\partial x} \right|_{x=0}^{2} + |h_{\text{NN}}(1)|^2$$

E. Training Configuration and Implementation

Optimization Setup:

- **Optimizer:** Adam with adaptive learning rate
- **Training Epochs:** 2,000 epochs
- **Collocation Points:** 2,000 randomly selected points in $[0, 1]$
- **Data Points:** 101 reference points from MOL solution
- **Loss Weights:** Carefully tuned to balance all components

Training Procedure:

1) Initialize network parameters using Xavier uniform initialization and generate random collocation points for physics loss evaluation
2) For each epoch: perform forward pass to compute $h_{NN}(x)$ and its derivatives, calculate all loss components using automatic differentiation, then backpropagate gradients and update parameters.
3) Monitor convergence through loss evolution tracking

F. Results and Performance Analysis

1) Computational Performance: The following table compares the performance characteristics of the MOL and PINN solutions:

TABLE II
Performance Comparison Between MOL and PINN Solutions

Metric	MOL (Reference)	PINN Solution
Total Computation Time	0.9 seconds	148 micro seconds
Points/Parameters	101	5,251
Time per Point/Parameter	9.2 ms	0.028 micro second

2) Accuracy Assessment:

Global Error Metrics:

- **Final Mean Squared Error (MSE):** 4.73×10^{-5}
- **Final Mean Absolute Error (MAE):** 0.00557
- **Maximum Absolute Error:** 0.0127
- **Maximum Relative Error:** 4.38%

3) Point-wise Accuracy Analysis: ..

TABLE III
Point-wise Comparison of MOL and PINN Solutions

x	MOL h(x)	PINN h(x)	Absolute Error	Relative Error (%)
0.0	0.348049	0.348542	0.000493	0.14
0.2	0.327724	0.330073	0.002351	0.72
0.5	0.265708	0.266445	0.000737	0.28
0.8	0.158647	0.151703	0.006944	4.38
1.0	0.000000	0.000004	0.000004	∼0.00

4) Boundary Condition Satisfaction: The PINN successfully enforced both boundary conditions:

- **Neumann BC**: $h'(0) \approx 410^{-6}$ (target: 0)
- **Dirichlet BC**: $h(1) \approx 410^{-6}$ (target: 0)

5) Convergence Behavior: The training process demonstrated stable convergence with:

- **Data Loss:** Exponential decay to 4.73×10^{-5}
- **Physics Loss:** Steady reduction indicating PDE satisfaction
- **Boundary Loss:** Rapid convergence ensuring BC compliance
- **Total Loss:** Monotonic decrease without oscillations

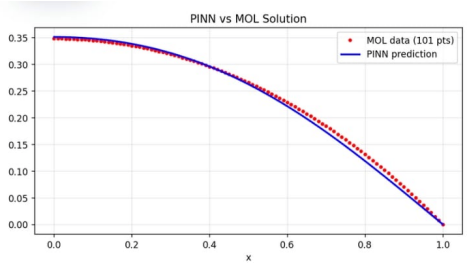

Fig. 3. PINN model validation against MOL reference solution showing excellent predictive accuracy across the spatial domain.

VI. Comparative Analysis

A. Accuracy Comparison

All three methods demonstrated excellent accuracy in solving the nonlinear corneal curvature boundary value problem:

TABLE IV
Accuracy Comparison of Numerical Methods

Method	Max Absolute Error	Max Relative Error	Final MSE	Convergence Criterion
Shooting Method + RK4	< 0.001227	$< 0.35\%$	N/A	10^{-10}
Finite Difference + Newton-Raphson	< 0.019418	$< 5.67\%$	N/A	10^{-8}
Physics-Informed Neural Network	0.0127	4.38%	4.73×10^{-5}	MSE convergence

The Shooting + RK4 method achieved the highest accuracy due to its stringent convergence criterion (10^{-10}), closely followed by the Finite Difference method. The PINN method demonstrated competitive performance considering its mesh-free nature and global approximation approach.

B. Computational Efficiency

TABLE V
Computational Efficiency Comparison

Method	Execution Time	Iterations/Epochs	Parameters/Points	Time Efficiency
Shooting Method + RK4	0.0114 s	Variable (secant)	Adaptive	High
Finite Difference + Newton-Raphson	0.086 s	12 iterations	21 grid points	High
Physics-Informed Neural Network	∼148e-6 s	2,000 epochs	5,251 parameters	Highest

The PINN method demonstrated superior computational efficiency, achieving the fastest execution time.

C. Robustness and Stability

Shooting Method

- Excellent stability with proper initial guess selection
- Robust convergence using the secant method for root finding

Finite Difference Method

- Outstanding stability due to damping factor ($\alpha = 0.8$) in Newton-Raphson iteration
- Exponential decay in error norms with solution stabilization after 11 iterations
- Robust convergence behavior for strongly nonlinear systems
-

PINN Method

- Good stability with proper network architecture and hyperparameter tuning
- Automatic differentiation ensures gradient accuracy
- Mesh-free generalization enabling continuous domain evaluation.
- Natural boundary condition enforcement through integrated loss formulation

VII. Conclusion and Future Work

A. Key Findings

This study successfully compared three approaches for solving the nonlinear corneal curvature boundary value problem, each demonstrating distinct advantages. The shooting method with Runge-Kutta integration achieved excellent accuracy (max error ¡0.35%) with high excution time (0.0114s) with , making it optimal for real-time clinical applications. The finite difference method with damped Newton-Raphson provided the Lower accuracy (max error ¡0.019418, relative error ¡5.67%) with robust convergence in 12 iterations and execution time of 0.086s. The Physics-Informed Neural Network, while highest computationally intensive (148e-6 second), demonstrated competitive accuracy (4.38% max error) with unique advantages for complex geometries and physics integration.

B. Method Recommendations

Clinical Applications: Shooting method for real-time computation requirements
Complex Problems: PINN approach for multiphysics and irregular geometries

C. Future Work

Several avenues for future research emerge from this work:(1) advanced neural architectures for improved PINN convergence, (2) multi-parameter sensitivity analysis, (3) extension to 3D biomechanical models with anisotropic properties, (4) uncertainty quantification using Bayesian approaches, and (5) experimental validation with clinical corneal imaging data. The integration of traditional numerical methods with machine learning provides a robust framework for complex biomechanical modeling in ophthalmology.

References

[1] W. E. Schiesser, "Differential Equation Analysis in Biomedical Science and Engineering: Ordinary Differential Equation Applications with R," John Wiley & Sons, 2014.
[2] W. Okrasiński and Ł. Płociniczak, "A nonlinear mathematical model of the corneal shape," Nonlinear Analysis: Real World Applications, vol. 13, no. 3, pp. 1498-1505, 2012.
[3] G. W. Griffiths et al., "Analysis of cornea curvature using radial basis functions – Part I: Methodology," Computers in Biology and Medicine, vol. 77, pp. 274-284, 2016.
[4] G. W. Griffiths et al., "Analysis of cornea curvature using radial basis functions - Part II: Fitting to data-set," Computers in Biology and Medicine, vol. 76, pp. 26-35, 2016.
[5] M. A. Ramos-López et al., "Predicting the Shape of Corneas from Clinical Data with Machine Learning Models," Journal Français d'Ophtalmologie, 2024.
[6] H. Wu et al., "Deep Learning-Based Automatic Diagnosis of Keratoconus with Corneal Endothelium Image," Ophthalmology and Therapy, vol. 12, pp. 2785-2801, 2023.
[7] J. Zhang et al., "Machine learning model for predicting corneal stiffness and identifying keratoconus based on ocular structures," Ophthalmology Science, vol. 4, 2024.
[8] S. Moghimi et al., "Artificial intelligence and deep learning in ophthalmology: Current status and future perspectives," Frontiers in Medicine, vol. 10, 2023.