UC Merced
UC Merced Electronic Theses and Dissertations

Title

Modeling of Pulsing Corals and Optimization of Mixing

Permalink

https://escholarship.org/uc/item/79336000

ISBN

9798197875068

Author

Malone, Sarah

Publication Date

2026-05-07

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, MERCED

Modeling of Pulsing Corals and Optimization of Mixing

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Sarah Malone

Committee in charge:

Shilpa Khatri, Chair

Roummel F. Marcia, Co-Chair

Harish Bhat

Laura Miller

Kevin Mitchell

2026

All chapters © 2026 Sarah Malone

The dissertation of Sarah Malone is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Harish Bhat)

(Laura Miller)

(Kevin Mitchell)

(Roummel F. Marcia, Co-Chair)

(Shilpa Khatri, Chair)

University of California, Merced

2026

TABLE OF CONTENTS

Signature Page . . . iii

List of Figures . . . vii

List of Tables . . . xi

List of Algorithms . . . xiii

Acknowledgements . . . xiv

Curriculum Vitae . . . xv

Abstract . . . xvii

Chapter 1 Introduction . . . 1

Chapter 2 Fluid Flow Analysis of Pulsing Soft Corals . . . 4
2.1 Introduction . . . 4
2.2 Model Equations and Numerical Methods . . . 5
2.3 Kinematics . . . 10
2.4 Kinematic and Flow Analysis . . . 16
2.4.1 Kinematic Comparison . . . 16
2.4.2 Vorticity . . . 16
2.4.3 Volumetric Flux . . . 17
2.4.4 Energy and Power Expenditure . . . 18
2.5 Parameter Variation . . . 19
2.6 Results . . . 21
2.6.1 Fluid Parameter Variation . . . 21
2.6.2 Muscle Parameter Variation . . . 32
2.7 Conclusion . . . 45

Chapter 3 Surrogate Modeling of Fluid Flow Analysis . . . 46
3.1 Introduction . . . 46
3.2 Methods . . . 46
3.2.1 Gaussian Process Regression . . . 46
3.2.2 Kernel Functions . . . 48
3.2.3 Implementation . . . 50
3.3 Results . . . 50
3.3.1 Kinematic Comparison . . . 52
3.3.2 Vorticity . . . 56
3.3.3 Volumetric Flux . . . 59
3.3.4 Energy and Power Expenditure . . . 64

3.4 Conclusion . . . 69

Chapter 4 Optimization of Muscle and Parameters . . . 70
4.1 Methods . . . 71
4.1.1 Generalized Pattern Search . . . 71
4.1.2 Nelder-Mead Simplex Method . . . 74
4.1.3 Gradient Descent . . . 75
4.1.4 Trust Region Methods . . . 77
4.1.5 Multi-objective Optimization . . . 81
4.1.6 Implementation . . . 84
4.2 Results . . . 84
4.2.1 Single-Objective Optimization of GP Models . . . 85
4.2.2 Multi-Objective Optimization of GP Models . . . 88
4.3 Conclusion . . . 90

Chapter 5 Iterative Surrogate Model Refinement . . . 91
5.1 Model Verification . . . 92
5.1.1 Kinematic Comparison . . . 93
5.1.2 Vorticity . . . 95
5.1.3 Volumetric Flux . . . 95
5.1.4 Energy and Power Expenditure . . . 99
5.2 Updated Surrogate Model . . . 100
5.2.1 Hyperparameter Interpretation . . . 101
5.2.2 Comparison of Original and Updated Surrogates . . . 101
5.3 Optimization of Updated Surrogates . . . 104
5.4 Conclusion and Next Steps . . . 106

Chapter 6 Conclusion . . . 107

Bibliography . . . 108

Appendix A Gaussian Processes as a Surrogate for Fluid Flow Analysis of Soft Corals . . . 119
A.1 Introduction . . . 119
A.2 Methods . . . 120
A.2.1 Model Equations and Numerical Methods . . . 120
A.2.2 Kinematics . . . 123
A.2.3 Kinematic & Flow Analysis . . . 126
A.2.4 Gaussian Processes Regression . . . 127
A.3 Results & Discussion . . . 129
A.3.1 Kinematic Comparison . . . 129
A.3.2 Vertical Flux . . . 132
A.4 Conclusion . . . 133

Appendix B Wasserstein-Based Similarity Constrained Matrix Factorization for Drug-Drug Interaction Prediction . . . 134
B.1 Introduction . . . 134
B.2 Methods . . . 135
B.2.1 Drug molecules as probability measures . . . 135
B.2.2 Wasserstein distance on drug molecules . . . 135
B.2.3 Similarity-Constrained Matrix Factorization . . . 137
B.3 Numerical Experiments . . . 138
B.3.1 Data . . . 138
B.3.2 Parameter Settings . . . 139
B.4 Results and Discussion . . . 140
B.4.1 Evaluation Metrics . . . 140
B.4.2 Comparison with Other Similarity Measures . . . 140
B.5 Conclusion . . . 141

Appendix C Triple Matrix Factorization for Drug-Drug Interaction Prediction Using Fused Gromov-Wasserstein Distances . . . 142
C.1 Introduction . . . 142
C.2 Methods . . . 143
C.2.1 Drug molecules as probability measures . . . 143
C.2.2 Wasserstein distance of drug molecules . . . 144
C.2.3 Gromov-Wasserstein distance of drug molecules . . . 146
C.2.4 Fused Gromov-Wasserstein distance of drug molecules . 147
C.2.5 Similarity-Based Triple Matrix Factorization . . . 148
C.3 Numerical Experiments . . . 149
C.3.1 Data . . . 149
C.3.2 Parameter Settings . . . 150
C.4 Results and Discussion . . . 150
C.4.1 Evaluation Metrics . . . 150
C.5 Conclusion . . . 152

Appendix D Detailed Results for Single-Objective Optimization . . . 153

Appendix E Detailed Results for Multi-Objective Optimization . . . 155

Appendix F Detailed Results for Optimization of Refined Surrogates . . . 160

List of Figures

Figure 1.1: Proposed dissertation workflow 2

Figure 2.1: IFED material configuration 6
Figure 2.2: Tentacle elasticity 12
Figure 2.3: Muscle activation function 13
Figure 2.4: Muscle location and tension 13
Figure 2.5: Coral model 15
Figure 2.6: Tentacle tip tracking 16
Figure 2.7: Regions of study for vorticity and velocity studies 17
Figure 2.8: Tentacle tip path for varied Re 22
Figure 2.9: Range of kinematic motion for varied Re 22
Figure 2.10: Spatially averaged tentacle velocity 23
Figure 2.11: Comparative tentacle trajectories 23
Figure 2.12: Vorticity contours for various Re values 25
Figure 2.13: Average magnitude of vorticity in V_A 26
Figure 2.14: Average vertical flow results 27
Figure 2.15: Average horizontal flow results 29
Figure 2.16: Comparison of tension- and force-based energy and power 30
Figure 2.17: Time-averaged tentacle displacement for various Reynolds numbers . . 31
Figure 2.18: Spatially averaged tentacle force 31
Figure 2.19: Tentacle kinematics with constant muscle tension and varied tension-elasticity ratio 33
Figure 2.20: Tentacle kinematics with varied muscle tension and varied tension-elasticity ratio 33
Figure 2.21: Tentacle kinematics with varied muscle tension and constant tension-elasticity ratio 34
Figure 2.22: Tentacle kinematics with varied muscle length 35
Figure 2.23: Comparison of tentacle tip tracking for $Re = 32$ with varied muscle length 35
Figure 2.24: Magnitude of vorticity with constant muscle tension and varied tension-elasticity ratio 36

Figure 2.25: Magnitude of vorticity with varied muscle tension and varied tension-elasticity ratio . . . 37
Figure 2.26: Magnitude of vorticity with varied muscle tension and constant tension-elasticity ratio . . . 37
Figure 2.27: Magnitude of vorticity with varied muscle length . . . 38
Figure 2.28: Averaged vertical flow in V_A with varied muscle tension and varied tension-elasticity ratio . . . 38
Figure 2.29: Averaged vertical flow in V_A with varied muscle tension and constant tension-elasticity ratio . . . 39
Figure 2.30: Averaged vertical flow in V_A with constant muscle tension and varied tension-elasticity ratio . . . 39
Figure 2.31: Averaged vertical flow in V_A with varied muscle length . . . 40
Figure 2.32: Averaged vertical flow in V_B with constant muscle tension and varied tension-elasticity ratio . . . 40
Figure 2.33: Averaged vertical flow in V_B with varied muscle tension and varied tension-elasticity ratio . . . 41
Figure 2.34: Averaged vertical flow in V_B with varied muscle tension and constant tension-elasticity ratio . . . 41
Figure 2.35: Averaged vertical flow in V_B with varied muscle length . . . 42
Figure 2.36: Averaged tension-based energy across muscle parameter variations . . . 43
Figure 2.37: Averaged tension-based power across muscle parameter variations . . . 43
Figure 2.38: Averaged force-based energy across muscle parameter variations . . . 44
Figure 2.39: Averaged force-based power across muscle parameter variations . . . 45

Figure 3.1: Example of kernel function. Top: Exponential, Squared Exponential, and Rational Quadratic; Bottom: Matern 3/2 and 5/2. . . . 49
Figure 3.2: Surrogate model for tentacle path . . . 53
Figure 3.3: Predicted tentacle kinematics with constant muscle tension and varied tension-elasticity ratio . . . 54
Figure 3.4: Predicted tentacle kinematics with varied muscle tension and varied tension-elasticity ratio . . . 54
Figure 3.5: Predicted tentacle kinematics with varied muscle tension and constant tension-elasticity ratio . . . 55
Figure 3.6: Predicted tentacle kinematics with varied muscle length . . . 55
Figure 3.7: True vs predicted magnitude of vorticity . . . 56
Figure 3.8: Predicted magnitude of vorticity with constant muscle tension and varied tension-elasticity ratio . . . 57
Figure 3.9: Predicted magnitude of vorticity with varied muscle tension and varied tension-elasticity ratio . . . 57
Figure 3.10: Predicted magnitude of vorticity with varied muscle tension and constant tension-elasticity ratio . . . 58
Figure 3.11: Predicted magnitude of vorticity with varied muscle length . . . 58
Figure 3.12: True vs predicted vertical flow for varied Re . . . 59

Figure 3.13: Predicted vertical flow in V_A with constant muscle tension and varied tension-elasticity ratio . . . 60
Figure 3.14: Predicted vertical flow in V_A with varied muscle tension and varied tension-elasticity ratio . . . 60
Figure 3.15: Predicted vertical flow in V_A with varied muscle tension and constant tension-elasticity ratio . . . 61
Figure 3.16: Predicted vertical flow in V_A with varied muscle length . . . 61
Figure 3.17: Predicted vertical flow in V_B with constant muscle tension and varied tension-elasticity ratio . . . 62
Figure 3.18: Predicted vertical flow in V_B with varied muscle tension and varied tension-elasticity ratio . . . 62
Figure 3.19: Predicted vertical flow in V_B with varied muscle tension and constant tension-elasticity ratio . . . 63
Figure 3.20: Predicted vertical flow in V_B with varied muscle length . . . 63
Figure 3.21: True vs predicted vertical flow for increased muscle length . . . 64
Figure 3.22: True vs predicted tension-based energy and tension with varied Re . . 65
Figure 3.23: Predicted tension-based energy across muscle parameter variations . . 65
Figure 3.24: Predicted tension-based power across muscle parameter variations . . 66
Figure 3.25: True vs predicted force-based energy and tension with varied Re . . . 66
Figure 3.26: Predicted force-based energy across muscle parameter variations . . . 67
Figure 3.27: Predicted force-based power across muscle parameter variations 68

Figure 4.1: Examples of pattern search simplices . . . 72
Figure 4.2: Simplex transformations in the Melder-Mead method . . . 74
Figure 4.3: Example of Trust Region Method . . . 78
Figure 4.4: Predicted tension-based energy across muscle parameter variations for low Reynolds numbers . . . 86
Figure 4.5: Predicted tension-based energy across muscle parameter variations for low Reynolds numbers . . . 87

Figure 5.1: Iterative workflow pipeline . . . 91
Figure 5.2: Tentacle kinematics surrogate accuracy on unseen data . . . 94
Figure 5.3: Magnitude of vorticity surrogate accuracy on unseen data . . . 96
Figure 5.4: Vertical flow in V_A surrogate accuracy on unseen data . . . 97
Figure 5.5: Vertical flow in V_B surrogate accuracy on unseen data . . . 98
Figure 5.6: Surrogate accuracy for tension-based metrics on unseen data . . . 99
Figure 5.7: Surrogate accuracy for force-based metrics on unseen data . . . 100
Figure 5.8: Tension-based energy and power: true vs. predicted with varied Re . . 102
Figure 5.9: Force-based energy and power: true vs. predicted with varied Re . . . 102
Figure 5.10: Tentacle kinematics: true vs. predicted . . . 103
Figure 5.11: Tentacle kinematics: true vs. predicted . . . 103
Figure 5.12: Magnitude of vorticity: true vs. predicted . . . 104

Figure A.1: Model of xeniid pulsing coral . . . 120

Figure A.2: Immersed boundary material configuration 121
Figure A.3: Regions of vertical flux study 127
Figure A.4: Tentacle tip trajectory 129
Figure A.5: Maximum height and width achieved during a pulse 130
Figure A.6: Inputs and outputs of GPR model for tentacle tip trajectory 131
Figure A.7: Tentacle tip trajectory, as modeled using the GPR 131
Figure A.8: Vertical flux averaged over region A for one pulse 132
Figure A.9: Inputs and outputs of GPR model for average vertical flux 133

Figure C.1: Drug-drug interaction problem 143
Figure C.2: The distance between two molecules is computed by representing the molecules as graphs with atoms as vertices 144
Figure C.3: Example of a coupling matrix, $Q \in \mathbb{R}^{K \times N}$ 145
Figure C.4: The Gromov-Wasserstein distance is measured as a function of the difference between the similarity of the atoms composing the drug . . . 147

List of Tables

Table 2.1: Base fluid parameters for simulations 7
Table 2.2: Base muscle parameters for simulations 15
Table 2.3: Summary of varied fluid and muscle parameters 20
Table 2.4: Summary of muscle parameter combinations and the number of simulations performed with varying Reynolds number. 20

Table 3.1: Summary of GP surrogate model accuracy across the full parameter space 51
Table 3.2: Relative error of surrogate models when evaluated on training data . . . 52

Table 4.1: Preliminary additive multi-objective optimization results 83
Table 4.2: Preliminary multiplicative multi-objective optimization results 83
Table 4.3: Preliminary AC multi-objective optimization results 84
Table 4.4: Optimization parameter bounds 85
Table 4.5: Single objective optimization results for vorticity and vertical flow . . . 86
Table 4.6: Single objective optimization results for energy and power 88
Table 4.7: Multi-objective optimization results for vorticity and tension-based energy 88
Table 4.8: Multi-objective optimization results for vorticity and tension-based power 89
Table 4.9: Multi-objective optimization results for vertical flow and tension-based energy 89
Table 4.10: Multi-objective optimization results for vertical flow and tension-based power 89

Table 5.1: Simulation parameters based on optimization results 92
Table 5.2: Relative error of surrogate models evaluated on new simulations 93
Table 5.3: Summary of GP surrogate model accuracy, including the new training data 100
Table 5.4: Updated optimal simulation parameters based on optimization results . . 105

Table A.1: Base values for pulsing coral simulations 122
Table A.2: Parameters used to define the tension-driven muscle coral model's motion [96]. 126

Table B.1: DDI prediction accuracy metrics for various similarity metrics 140

Table C.1: DDI prediction accuracy metrics for various Wasserstein-based similarity metrics 151
Table C.2: Further DDI accuracy metrics 151

Table D.1: Optimization results: Vorticity and Vertical Flow 153
Table D.2: Optimization results: Energy and Power 154

Table E.1: Multi-objective optimization results for the magnitude of vorticity and tension-based energy 156
Table E.2: Multi-objective optimization results for the magnitude of vorticity and tension-based power 157
Table E.3: Multi-objective optimization results for the vertical flow in V_A and tension-based energy 158
Table E.4: Multi-objective optimization results for the vertical flow in V_A and tension-based power 159

Table F.1: Updated multi-objective optimization results for the magnitude of vorticity and tension-based energy 160
Table F.2: Updated multi-objective optimization results for the magnitude of vorticity and tension-based power 161
Table F.3: Updated multi-objective optimization results for the vertical flow in V_A and tension-based energy 161
Table F.4: Updated multi-objective optimization results for the vertical flow in V_A and tension-based power 162

List of Algorithms

Algorithm 2.1: Solution procedure for nodal IFED simulations, computed each timestep 9
Algorithm 4.1: Generalized Pattern Search 73
Algorithm 4.2: Nelder-Mead Simplex Method 76
Algorithm 4.3: Gradient Descent with Armijo–Wolfe Line Search 77
Algorithm 4.4: Trust-Region Method 79
Algorithm 4.5: Steihaug Conjugate Gradient Method for the Trust-Region Subproblem 81

ACKNOWLEDGEMENTS

> "The only good way to ruin everything is by accident—if you make an effort to do it you've *tried*, and sadly, trying is its own success."
>
> —George Watsky

Having spent ten years at UC Merced, as both an undergraduate and graduate student, I can sincerely call this place home. I am grateful to the countless people who have supported me along the way.

I would like to start by thanking my advisors, Shilpa Khatri and Roummel Marcia, for their time, patience, and effort in helping shape me into the researcher and professional I am today. I also thank my committee members Harish Bhat, Laura Miller, and Kevin Mitchell for their thoughtful feedback along the way.

This work would not have been possible without the support of several funding organizations. Since August of 2022, I have been funded by the Department of Defense SMART Scholarship for Service Program, funded by the Under Secretary of Defense for Research and Engineering, National Defense Education Program. Computation time on the MERCED and Pinnacles clusters was supported by NSF-MRI #1429783 and NSF-MRI #2019144, respectively, through the Cyberinfrastructure and Research Technologies (CIRT) group at UC Merced. I also thank the UC Merced Applied Math Department, supported by NSF-DMI #1840265. Finally, I was fortunate to receive scholarships from the Phi Mu Foundation during the 2020-2021 and 2021-2022 academic years through the Maud Tachell Sagle Memorial Fund and the Mary McAteer Scholarship Fund.

On a more personal note, I want to thank my parents for always encouraging me to pursue my interests and for supporting me throughout my education. I also thank my sister, Emily, for being a constant source of supporteven if our phone schedules rarely align.

I want to thank my friends specifically Natalie Meacham, Morgan Lavenstein Bendall, Scott West, and Steven Ellenberger. Grad school had its challenges, but between weekly trivia, movie nights, League of Legends sessions, and countless other adventures, you made the past six years fun and memorable.

Finally, I want to thank my husband, Zach Malone. I cannot begin to put into words how grateful I am to have you as my partner in life and constant support system. I dont think I would be where I am today without you.

Sarah Malone

sdowns@ucmerced.edu | sdowns409@gmail.com

Education

University of California, Merced, Ph.D. in Applied Mathematics May 2026
- Dissertation: *Modeling of Pulsing Corals and Optimization of Mixing*
- GPA: 3.790
- Coursework: Special Topics in Machine Learning, Computational Fluid Dynamics, Scientific Computing

University of California, Merced, B.S. in Applied Mathematics May 2020
- GPA: 3.494
- Coursework: GIS Decision Analysis, Mathematical Modeling, Modern Applied Statistics
- Research: Statistical Analysis of African Soils Using Spectral Profiles

Research Experience

Research Intern, National Geospatial-Intelligence Agency Summers 2024, 2025
- Supported maritime domain awareness and navigation safety initiatives using Python, MATLAB, and GIS.
- Processed and visualized geospatial datasets with IDL and GIS.

Research Intern, Autonomy Technology Research Center, Wright State University Summer 2021
- Developed hierarchical classification models for synthetic aperture radar imagery.
- Implemented algorithms in Python and MATLAB.

Research Intern, Center for Ecosystem Climate Solutions, UC Irvine Summer 2020
- Classified forest disturbance using GEDI lidar data with machine learning.
- Integrated Python, GIS, and Google Earth Engine for large-scale modeling.

Deep Learning Intern, Intel Corporation Summer 2019
- Developed telemetry visualization for Neural Network Processor performance using C++ and Python.

Teaching Experience

Teaching Fellow, UC Merced Living Learning Communities Aug 2021 - May 2022
Instructor of record for first-year seminar on technology and society.

Teaching Assistant, UC Merced Applied Mathematics Aug 2020 - May 2021
Supported courses in Calculus II and Vector Calculus.

Publications

Gaussian Processes as a Surrogate for Fluid Flow Analysis of Soft Corals Oct 2025
OCEANS 2025 Great Lakes, *in-person*
Authors: Sarah Malone, Roummel F. Marcia, Shilpa Khatri
https://doi.org/10.23919/OCEANS59106.2025.11245103

Triple Matrix Factorization for Drug-Drug Interaction Prediction Jul 2025
IEEE EMBC 47th, *in-person*
Authors: Sarah Malone, Mohammed Aburidi, Roummel F. Marcia
https://doi.org/10.1109/EMBC58623.2025.11252781

Wasserstein-Based Similarity Constrained Matrix Factorization Nov 2024
IEEE SiPS Workshop, *in-person*
Authors: Sarah Malone, Mohammed Aburidi, Roummel F. Marcia
https://doi.org/10.1109/SiPS62058.2024.00017

Presentations

Contributed Talks

Gaussian Processes as a Surrogate for Soft Coral Fluid Flow Analysis Oct 2025
OCEANS 2025 Great Lakes, *in-person*

Modeling of Pulsing Corals and Mixing Nov 2024
APS Division of Fluid Dynamics Meeting, *in-person*

Simulations of Pulsing Soft Corals Nov 2022
APS Division of Fluid Dynamics Meeting, *in-person*

Posters

Utilizing GEDI data to classify changes in forest structure Dec 2020
AGU Fall Meeting, *online*

Technical Skills

Programming: Python, MATLAB, C++
Geospatial: GIS, IDL

Fellowships and Awards

Department of Defense SMART Scholar Aug 2022 – May 2026
UC Merced Applied Mathematics Excellence in Service Award May 2026
UC Merced Applied Mathematics Distinguished Service Award May 2025, 2026
UC Merced Grad Slam Top 10 Finalist Feb 2026

Leadership and Service

Graduate Student Leadership Council, UC Merced Aug 2024 - May 2026
- Liaison between graduate students and faculty.
- Coordinated department events and community-building initiatives.

Chapter Adviser, Phi Mu Fraternity Feb 2022 - May 2026
- Advise chapter officers and oversee operations, programming, and budget.
- Mentor members to promote leadership and personal development.

GradEXCEL Peer Mentor, UC Merced Graduate Division Sep 2025 - Dec 2025
- Mentored first-year graduate students in Applied Mathematics.

Fluids in Yosemite Conference Organizing Committee, UC Merced Oct 2023 - Apr 2024
- Assisted with program organization, logistics, and participant outreach.

ABSTRACT OF THE DISSERTATION

Modeling of Pulsing Corals and Optimization of Mixing

by Sarah Malone
Doctor of Philosophy in Applied Mathematics

University of California Merced, 2026

Committee Chair: Shilpa Khatri

Soft corals in the Xeniidae family pulse in an energetically expensive manner to generate fluid flow, which benefits them biologically. This flow has been modeled using computationally intensive numerical methods, such as the Immersed Boundary Method, which limits the ability to explore a wide parameter space. To address this challenge, we introduce Gaussian Process Regression as a surrogate model to capture the kinematic and flow outcomes of these simulations. By training on a limited set of simulations, the surrogates enable accurate predictions across a broader range of parameters than those originally sampled. These surrogate models can then be optimized to identify the parameter combinations which maximize fluid flow. The resulting optimal parameters can further guide additional numerical simulations, creating an iterative framework for targeted exploration of the parameter space. This approach offers new insights into how the coral polyp musculature and local flow conditions influence the resulting kinematics and fluid transport while substantially reducing computational cost.

Chapter 1

Introduction

Soft pulsing corals in the Xeniidae family are known for their unique movement, which drives fluid flow around them. Because these sessile animals do not use this energetically expensive motion for locomotion, researchers have long questioned what purpose it serves. While some soft corals use bending or flexing to feed [58], this is not believed to be the case for xeniid corals. They are rarely found with prey in their gastric cavity [28, 58, 98] and exhibit low primary energy production [19]. Instead, these corals depend on endosymbiotic algae (zooxanthella) to photosynthesize and provide energy to their coral host [106]. This has led to the prevailing hypothesis that pulsation enhances both the influx of carbon dioxide necessary for photosynthesis and the efflux of oxygen around the coral polyp, thereby improving the photosynthetic ability of these algae [40, 54, 68]. Additionally, pulsation can enhance the uptake of other key nutrients, such as nitrogen and phosphorus, which benefit both the coral host and its symbiotic algae [22, 40, 56].

Several numerical studies have examined this pulsing motion to better understand the result role of the fluid flow. Three-dimensional immersed boundary models have demonstrated that pulsation enhances local mixing and re-filtration of the surrounding fluid [5, 91]. Later work explored interactions among multiple coral polyps, showing that mixing effects are strongest in the region immediately surrounding an individual polyp [92]. Incorporating laboratory data improved model resolution and accuracy [93], while two-dimensional models captured oxygen concentration dynamics and transport [97]. Most recently, a tension-based model has been developed to explicitly capture the muscle-driven pulsing behavior, rather than following a phenomenological motion [96]. Collectively, these studies support the hypothesis that Xeniid corals have evolved to circulate fluid in ways that benefit both the polyp and its symbionts.

Despite these prior works, the interaction between fluid properties and muscle mechanics creates a high-dimensional parameter space that is difficult to explore using direct numerical simulations alone. While some studies have explored minor variations in biological parameters, most focus on fixed values and do not systematically investigate how changes in muscle mechanics or environmental conditions influence the resulting flow [5, 91, 92, 93]. Santiago et al. [96] developed a new mechanistic model and

examined key parameters in greater depth, but many questions remain. Consequently, it remains unclear how changes in these parameters affect the induced flow dynamics or whether biological coral polyps operate near conditions that maximize fluid transport.

To address these questions, this dissertation conducts a parameter study examining how fluid properties and muscle mechanics influence the pulsing motion. Specifically, we address two main questions:

1. Have biological coral polyps evolved to efficiently drive fluid flow?
2. What conditions lead to the greatest fluid transport and mixing?

We hypothesize that biological corals are tuned to efficiently generate local fluid motion and that the environments in which they thrive correspond to conditions that maximize fluid flow. Consequently, we expect that the parameter values producing the greatest fluid motion will align with the biological values.

In chapter 2 we conduct this parameter study. We develop a model of the fluid-structure interaction using the immersed boundary method [83], which couples a Lagrangian elastic structure with an Eulerian fluid domain. Specifically, we employ the nodal immersed finite element finite difference formulation [35, 119], allowing us to compute velocity and pressure fields while updating the structures motion accordingly. We vary both the Reynolds numbera dimensionless measure of inertial versus viscous forcesand the muscle parameters that drive pulsing, such as muscle length, maximum muscle tension, and tentacle elasticity. After running these simulations, we analyze the resulting kinematics and flow fields through kinematic, flow, and cost metrics to quantify the impact of each parameter on the induced fluid flow. This workflow is summarized in the top panel of Fig. 1.1, illustrating the sequence from parameter selection to the resulting kinematic and flow metrics.

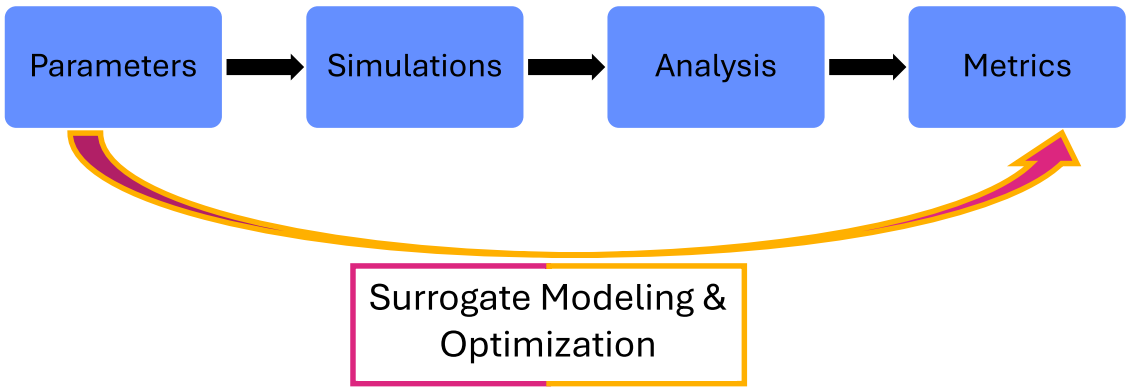


Figure 1.1: Top: Direct numerical workflow. Bottom: Proposed plan to bypass direct numerical simulations.

While these methods are well established, they are computationally intensive: a single simulation can take several days to reach steady state on a high-performance cluster and

produce datasets exceeding one terabyte. Thus, only a limited number of simulations can be performed. To overcome this limitation, chapter 3 develops surrogate models using Gaussian process regression [62, 69, 88], a nonparametric technique which interpolates between known results to reveal underlying functional relationships. This allows us to learn general trends from a limited dataset and identify promising regions of the parameter space for further refinement. Portions of this surrogate modeling framework were previously published in the proceedings of IEEE OCEANS 2025 [66], and are substantially expanded here to include additional training data and analyses. The original publication is included in Appendix A. This process, represented in the bottom panel of Fig. 1.1, allows us to estimate the kinematic, flow, and cost outputs across a uniform grid, bypassing the full simulation workflow once an initial set of simulations has been performed.

Finally, the surrogate models are optimized to determine the parameter combinations that yield the greatest fluid flow and mixing in chapter 4. We compare four optimization methods: generalized pattern search and Nelder-Mead (derivative-free), and gradient descent and trust-region methods (derivative-based). Because the smoothness of the surrogate surfaces is uncertain, both approaches are tested for convergence and efficiency. We also compare optimization time and performance, with the aim of finding a computationally efficient approach to identify optimal parameter combinations. In some cases, we optimize a single objective, while in others we use weighted combinations (such as maximizing upward flow while minimizing energetic cost).

Related methodological work in optimization was previously conducted in separate conference studies on drug-drug interaction prediction [64, 65]. Although those studies addressed a different application domain, they informed the optimization framework employed here. The full publications are included in Appendices B–C.

After generating and optimizing the initial surrogate model, the workflow then enters an iterative phase. Additional simulations, focused on the current optimal parameters, are used to construct an updated surrogate model, which can then inform subsequent rounds of optimization. We present one iteration in chapter 5.

The methods and insights presented here contribute to a broader understanding of coral reef ecosystem dynamics. Soft corals are generally more resilient to climate change than stony corals [24, 71, 112], and shifts toward soft-coral dominance have been observed in several benthic communities [48]. Understanding how pulsing corals interact with their surrounding fluid environment may therefore provide insight into the ecological roles these organisms play in evolving reef systems. Beyond ecological implications, the modeling and optimization framework developed here may also inform biomimetic engineering applications, where animal-inspired mechanisms are used to enhance fluid transport and mixing in industrial systems.

Chapter 2

Fluid Flow Analysis of Pulsing Soft Corals

2.1 Introduction

This work begins by characterizing the fluid flow generated by pulsing corals across different muscle parameters and local flow regimes. We employ the immersed boundary (IB) method, a numerical method developed by Charles Peskin, which allows for the modeling of coupled fluid-structure interactions [85]. A key advantage of this method, and one reason for its popularity in biological fluid studies, is that it models the fluid in an Eulerian framework, tracking flow properties on a fixed Cartesian grid, while representing the elastic structure in a Lagrangian framework, where points move with the boundary. These two frameworks communicate through a set of interaction equations, enabling conversion between the Eulerian and Lagrangian grids. Thus, fluid velocity and pressure can be computed on a simple Cartesian grid, while still accommodating for complex, moving structures that would be difficult to represent otherwise. Applications of this methodology have expanded well beyond its original purpose of studying flow through a heart [83, 84] to now include the study of heart valves [36, 73], the inner ear [6], sperm motility [20], aquatic animal location [21, 43], immersed winds [75], insect flight [49], and jellyfish swimming [39, 42, 44, 45].

Additionally, the immersed boundary method has been extensively studied, both in terms of its stability and accuracy [8, 33, 46, 57, 108, 113] and through the development of extensions aimed at improving stability, accuracy, computational efficiency as well as adapting to particular applications. This includes a discretization of the entire system through finite elements [61, 117, 125, 126] as well as a discretization of the fluid through finite difference and the structure through finite elements [35, 119]. Aside from discretization, adaptations have also focused on utilizing energy functionals to compute the elastic force [18], applying a projection method as a preconditioner [32], introducing an adaptive Eulerian grid [34], implementing a fully variational method which removed the need for regularized delta functions [7, 38, 90], using a meshless method [26, 27], accounting for the mass of a structure [51, 76], and using radial basis functions to define structure deformation [99].

We utilize the nodal finite element / finite difference variation to understand the fluid dynamics of pulsing corals belonging to the Xeniidae family. This is an ongoing collaboration that has produced a breadth of results. Early studies examined the three-dimensional fluid-structure interaction using the standard immersed boundary method and modeled the tentacle's motion through a set of prescribed equations [5, 91]. These results were further refined by tracking the motion and fluid flow of corals in a lab environment using particle image velocimetry [93]. This led to the introduction of a more accurate prescribed motion. Up to this point, each of these studies had only focused on the dynamics of a single coral polyp. However, these corals live in colonies and therefore, the flow of a single polyp has the potential to affect its neighbors' flow. Subsequent studies then found that pulsing corals only impact the flow of the fluid in their local area and that there is no benefit to coordinated pulsing between corals when placed nearby to each other [92]. This study also found that polyp pairs would produce slightly less flow than single coral polyps and that as coral colonies became bigger, the flow generated by a single polyp within the colony was not significantly altered from that of a pair. This indicated that corals form colonies for reasons other than gaining a hydrodynamic advantage. This also showcases how studies with only one coral polyp can be extrapolated to generally understand colony dynamics as a whole [92]. More recently, there have been efforts to understand not only the fluid flow but also the mixing generated by this pulsing. Results for a two-dimensional computational coral quantified the mixing through the use of Poincaré maps and found that as the Reynolds number increases, so does the amount of fluid entering regions of interest near the coral polyp, indicating that more mixing was occurring [95, 97]. This study also explored photosynthesis of the symbiotic algae by modeling the advection and diffusion of oxygen, serving as a proxy for the photosynthesis, and found that the flow and mixing parameters that govern the coral system play a key role in both the resulting mixing and photosynthesis [97]. Most recently, new work has been conducted to update the pulsing motion of these simulations to a mechanistic model, modeling the coral polyp's muscle dynamics and solving for the kinematics [96].

This chapter extends and expands upon these previous studies by adding a parameter sweep of the variables that determine the muscle-driven model. This will allow us to understand the conditions that allow coral polyps to generate the most fluid flow.

2.2 Model Equations and Numerical Methods

We employ the nodal immersed boundary finite element / finite difference method (IFED) to solve for the coral motion and resulting fluid flow [35, 96, 119]. IFED utilizes a finite element discretization of the structure's Lagrangian framework and a finite difference discretization of the fluid's Eulerian framework. This is an improvement over the standard immersed boundary method, as it allows for coarser Lagrangian meshes relative to the Eulerian mesh, while ensuring that the structure remains "watertight" throughout the structure's deformation, preventing numerical and discretization errors [35].

The Eulerian fluid domain has positions $\vec{x} \in \Omega$ at which the fluid velocity and pressure

are solved, where Ω is the entire fluid domain. IFED makes use of a reference configuration for the Lagrangian coral structure, which will be the structure with no movement applied. The Lagrangian grid is defined as $\vec{X} \in S$, where $\vec{X}$ is a grid point on the reference configuration. To arrive at the current configuration, $\vec{\chi}(\vec{X}, t)$, we apply the deformation tensor $\mathbb{F}$, which allows us to find the physical Lagrangian position in time t after some movement has been applied. This setup is illustrated in Fig. 2.1.

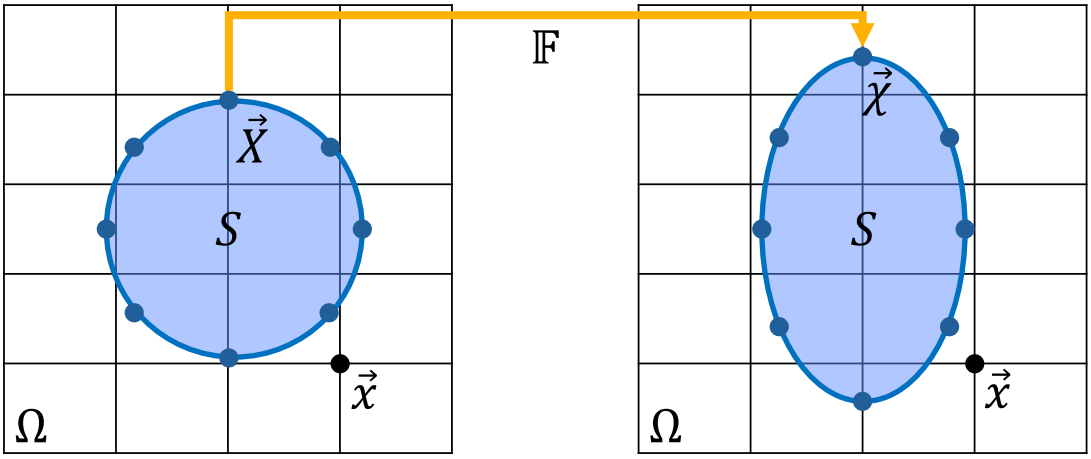


Figure 2.1: IFED configuration. Here, Ω is the Eulerian grid, while S is the Lagrangian grid. The structure is deformed through the deformation tensor $\mathbb{F}$, allowing the structure to move from its initial configuration $\vec{X}$ to its current configuration $\vec{\chi}$. Redrawn from Santiago [95].

The IFED method is a numerical method which solves a system of equations for the movement of the coral and resulting fluid flow in a coupled manner. The model system includes the Lagrangian force of the coral on the fluid, $\vec{F}(\vec{X}, t)$, given in weak form by

$$\begin{aligned} \int_S \vec{F}(\vec{X}, t) \cdot \vec{\phi}(\vec{X}) d\vec{X} = &- \int_S \mathbb{P}(\vec{X}, t) : \nabla_{\vec{X}} \vec{\phi}(\vec{X}) d\vec{X} \\ &+ \int_S \vec{F}_T(\vec{X}, t) \cdot \vec{\phi}(\vec{X}) d\vec{X}, \end{aligned} \tag{2.1}$$

where $\mathbb{P}$ is the first Piola-Kirchhoff (PK1) stress tensor, $\vec{F}_T$ is the tether force holding our structure in place, $\vec{\phi}(\vec{X})$ is a test function, : is the double dot product, and $\nabla_{\vec{X}}$ is the gradient with respect to $\vec{X}$ [35, 96]. Details on $\mathbb{P}$ and $\vec{F}_T$ will be given in Section 2.3.

The model also includes interaction equations, which are used to convert between the Lagrangian and Eulerian frameworks. First, the Lagrangian force applied by the immersed structure onto the fluid is spread to the Eulerian fluid grid by

$$\vec{f}(\vec{x}, t) = \int_S \vec{F}(\vec{X}, t) \delta \left(\vec{\chi}(\vec{X}, t) - \vec{x} \right) d\vec{X}, \tag{2.2}$$

where $\vec{F}(\vec{X},t)$ is the Lagrangian force (solved for in Eq. (2.1)) being spread to the Cartesian grid and $\delta(\cdot)$ is the Dirac delta function. This results in the Eulerian force applied to the fluid by the immersed structure, $\vec{f}(\vec{x},t)$.

The Navier-Stokes equations are solved on the Eulerian grid using this force, given by

$$\begin{aligned} \rho\left(\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\nabla\vec{u}\right)+\nabla p &= \mu\Delta\vec{u}+\vec{f}, \\ \nabla\cdot\vec{u} &= 0. \end{aligned} \tag{2.3}$$

Here, $\vec{u}(\vec{x},t)$ is the fluid velocity, $p(\vec{x},t)$ is the fluid pressure, ρ is the fluid density, and μ is the fluid viscosity. In this equation, the advection term, $\vec{u}\cdot\nabla\vec{u}$, represents the change in fluid velocity due to the motion of the fluid itself, while the diffusion term, $\mu\Delta\vec{u}$, describes how viscosity diffuses momentum through the fluid. The pressure gradient, ∇p, enforces incompressibility.

IB methods offer a clear advantage: we only need to solve for the fluid velocity and pressure on the fixed Eulerian grid. A central non-dimensional parameter for these equations is the Reynolds number, Re, which is the ratio of inertial to viscous forces in the fluid,

$$Re=\frac{\rho\pi L_T^2}{2\mu T_c}. \tag{2.4}$$

In this formulation, which is based on the coral tentacle kinematics, L_T is the tentacle length and T_c is the contraction period. We take $Re=46.5892$ as our base value, with other relevant parameters listed in Table 2.1. These values are drawn from prior experimental studies of xeniid corals, ensuring our simulations are biologically realistic [93].

Parameter	Variable	Value	Units
Fluid domain volume	Ω	0.0992^3	m^3
Fluid density	ρ	1029	$\mathrm{kg\,m^{-3}}$
Dynamic viscosity	μ	0.00108	$\mathrm{kg\,m^{-1}\,s^{-1}}$
Tentacle length	L_T	0.0041	m
Contraction period	T_c	0.54	s
Expansion period	T_e	0.71	s
Resting period	T_r	0.64	s
Pulsing period	T_{per}	1.89	s
Reynolds number	Re	46.5892	dimensionless

Table 2.1: Base fluid and kinematic parameters used in the pulsing coral simulations.

The second of our interaction equations interpolates the fluid velocity to the Lagrangian structure,

$$\begin{aligned} \frac{\partial\vec{\chi}(\vec{X},t)}{\partial t} &= \vec{U}(\vec{X},t) \\ &= \int_\Omega \vec{u}(\vec{x},t)\delta\left(\vec{\chi}(\vec{X},t)-\vec{x}\right)d\vec{x}, \end{aligned} \tag{2.5}$$

where $\vec{U}(\vec{X}, t)$ is the velocity of the structure. This allows the coral to be moved as a result of the updated fluid velocity.

The IFED method temporally and spatially discretizes this system of equations. The Eulerian fluid variables are discretized using finite differences on a marker-and-cell (MAC) staggered grid. Adaptive mesh refinement is employed to increase resolution in regions of large fluid vorticity. The Lagrangian structure is discretized using a finite-element triangulation, denoted $\mathcal{F}^h$, consisting of m nodes and associated finite-element basis functions $\phi_q(\vec{X})$.

The interaction between the Eulerian fluid and the Lagrangian structure is represented through the interaction equations, given in Eq. (2.2) and Eq. (2.5). In the IFED formulation, these interactions are evaluated using quadrature rules defined on the Lagrangian mesh. In particular, a nodal quadrature rule $\mathbb{N}_q = \{(\vec{X}_q, w_q)\}$ is defined such that each quadrature point $\vec{X}_q$ corresponds to a node of $\mathcal{F}^h$ with weights given by

$$w_q = \int \phi_q(\vec{X}_q) \cdot \vec{1} d\vec{X}_q, \tag{2.6}$$

where ϕ_q is the finite-element basis function associated with node q.

The Lagrangian elastic forces are obtained by discretizing Eq. (2.1) using the finite-element basis. This results in a linear system for the nodal force vector,

$$\mathbb{M}\vec{F} = \vec{L}, \tag{2.7}$$

where $\mathbb{M}$ is the finite-element mass matrix and $\vec{L}$ is the load vector. The entries of the load vector are given by

$$\vec{L}_i = - \sum_{(\vec{X}_q, w_q) \in \mathbb{H}_q} \mathbb{P}_e(\vec{X}_q, t) : \nabla_{\vec{X}} \phi_i(\vec{X}_q) w_q, \tag{2.8}$$

where $\mathbb{P}_e$ denotes the passive elastic stress tensor and $\mathbb{H}_q = \{(\vec{X}_q, w_q)\}$ is the Gaussian quadrature rule defined over the elements of the finite-element mesh.

In practice, the load vector is evaluated using Gaussian quadrature to accurately integrate the stress over each element. The mass matrix is evaluated using the nodal quadrature rule. Once the nodal forces have been obtained, the Eulerian-Lagrangian interaction operators are applied. The force spreading operator distributes the Lagrangian nodal forces to the surrounding Eulerian grid using a regularized discrete delta function corresponding to the four-point Peskin kernel, which spreads forces over four Eulerian grid cells in each coordinate direction [85]. The velocity interpolation operator evaluates the fluid velocity at the Lagrangian nodes. In the IFED formulation, both operators are evaluated using the nodal quadrature rule.

The coupled system is advanced in time using a semi-implicit scheme. In particular, the nonlinear advection term is treated explicitly using a second-order Adams-Bashforth method, while the viscous term is treated implicitly using Crank-Nicolson. The resulting

discretized momentum equation takes the form

$$\rho\left(\frac{\vec{u}^{n+1}-\vec{u}^{n}}{\Delta t}\right)+\frac{3}{2}\left(\vec{u}^{n}\cdot\nabla\vec{u}^{n}\right)-\frac{1}{2}\left(\vec{u}^{n-1}\cdot\nabla\vec{u}^{n-1}\right)$$
$$=-\nabla p^{n+1/2}+\mu\Delta\left(\frac{\vec{u}^{n+1}+\vec{u}^{n}}{2}\right)+\vec{f}^{n+1/2}. \tag{2.9}$$

where $\vec{f}^{n+1/2}$ denotes the Eulerian force evaluated at the half timestep. The Lagrangian nodes are updated explicitly using the interpolated fluid velocities at each timestep via Eq. (2.5). When adaptive mesh refinement is used, Lagrangian forces are spread to the finest fluid level, and the fluid velocity is interpolated back to the Lagrangian nodes across the hierarchy. The timestep is chosen to satisfy both a CFL condition based on the maximum fluid velocity and a stability constraint arising from the Lagrangian elasticity.

A summary of this procedure can be found in Alg. 2.1. Additional details of the nodal IFED formulation can be found in Wells et al. [119].

Algorithm 2.1 Solution procedure for nodal IFED simulations, computed each timestep

1. Evaluate Lagrangian forces by discretizing Eq. (2.1) and solving the resulting linear system for $\vec{F}$ (Eq. (2.7)).
2. Spread Lagrangian forces to the Eulerian grid using the interaction equation (Eq. (2.2)) with the four-point Peskin kernel.
3. Advance the Eulerian fluid equations (Eq. (2.9)) using a semi-implicit scheme.
4. Interpolate updated fluid velocities to the Lagrangian nodes (Eq. (2.5)).
5. Update Lagrangian node positions using the interpolated velocities.

To implement this, we utilized the software library Immersed Boundary Adaptive Mesh Refinement (IBAMR), version 0.13, to solve the discretized equations numerically [34, 35, 47]. Simulations were executed on the High Performance Computing (HPC) system at UC Merced, initially using the MERCED cluster and later the Pinnacles cluster, running on a single compute node utilizing all available CPU cores, typically 56. At the base Re, simulations required approximately 2 days of wall-clock time to reach periodic steady state, defined as consistent fluid behavior across successive pulses. Steady state was determined by examining the horizontal and vertical velocities in a cylindrical region near the coral, and was typically achieved by the 15th pulse, although higher Re simulations required more pulses to reach steady state.

Once the simulations reached steady state, kinematic and fluid flow analyses were completed using the visualization software VisIt [10, 37] and MATLAB [109].

2.3 Kinematics

We employ a tension-based mechanistic model for the musculature of the coral polyp, which was developed in Santiago et al. [96]. This model incorporates the first Piola-Kirchhoff, PK1, stress tensor,

$$\mathbb{P} = \mathbb{P}_e + \mathbb{P}_t, \tag{2.10}$$

which is a summation of the passive elasticity $\mathbb{P}_e$ and the active tension $\mathbb{P}_t$. The PK1 stress tensor defines stress relative to a reference configuration, unlike the Cauchy stress tensor, which defines stress in the current configuration. This is advantageous for our model, as it removes the need to track stress in the evolving deformed state.

The passive elasticity, $\mathbb{P}_e$, determines the elastic forces of our structure, guiding its return to the resting configuration. This is described by a neo-Hookean material model

$$\mathbb{P}_e = \eta J^{-2/3}\left(\mathbb{F} - \frac{I_1}{3}\mathbb{F}^{-\top}\right) + \beta \ln(J)\mathbb{F}^{-\top}. \tag{2.11}$$

Here, $\mathbb{F} = \partial\vec{\chi}/\partial\vec{X}$ is the deformation tensor, $J = \det(\mathbb{F})$ is the determinant of the deformation tensor, and $I_1 = \text{tr}(\mathbb{F}^\top\mathbb{F})$ is a tensor invariant of the right Cauchy-Green tensor. Additionally, the shear elastic modulus, η, is defined in relation to the material parameters, the Young's modulus, E, and Poisson's ratio, $\nu = 0.4$:

$$\eta = \frac{E}{2(1+\nu)}. \tag{2.12}$$

We will define η directly below as varying over the coral tentacles. Lastly, the term $\beta \ln(J)\mathbb{F}^{-\top}$ in Eq. (2.11) is a volume stabilization term to reinforce incompressibility and improve the volume conservation of the material, where β is the bulk density [114], given by

$$\beta = \frac{2\eta(1+\nu)}{3(1-2\nu)}. \tag{2.13}$$

The active tension, $\mathbb{P}_t$, is used to model the musculature of the coral polyp, which we compute as follows. First, we define the muscle fiber direction

$$\vec{\text{fl}}_o' = \left(-X_1, \frac{R_t(\vec{X})}{X_2 - 0.01}, -X_3\right), \tag{2.14}$$

where $R_t(\vec{X})$ is a scalar determining how far along the tentacle a point $\vec{X}$ is

$$R_t(\vec{X}) = \sqrt{X_1^2 + X_3^2}. \tag{2.15}$$

We normalize $\vec{\text{fl}}_o'$ as a unit vector,

$$\vec{\text{fl}}_o = \frac{\vec{\text{fl}}_o'}{\|\vec{\text{fl}}_o'\|}. \tag{2.16}$$

This properly orients the muscles, making it so the coral tentacles pull inward towards the oral disc [29, 30]. Then, we define the active tension as

$$\mathbb{P}_t = JT\mathbb{F}\vec{\mathrm{fl}}_o \mathrm{tr}\left(\vec{\mathrm{fl}}_o\right), \tag{2.17}$$

where T is the magnitude of the tension and $\mathrm{tr}\left(\vec{\mathrm{fl}}_o\right)$ is the trace of the muscle fiber direction.

The edges of the tentacles make up the pinnule region consists of fine branches covered in cilia, hair-like organelles, that generate local ciliary flow by driving water through and around the tentacle [80, 81, 100]. Modeling this phenomenon would be computationally expensive, so instead Eq. (2.13) is modeled through variable elasticity. Specifically, we vary the shear elastic modulus, η, over the tentacle domain,

$$\eta = \eta_p + \left(\eta_t - \eta_p l_\eta(R_t) \max\left(\tau(\vec{X}), \sigma(R_t)\right)\right), \tag{2.18}$$

where η_p is the elasticity applied to the tentacle edges, representing the pinnule region, and η_t is the elasticity applied to the tentacle center. We also include a spatial variation along the tentacle length through l_η. Altogether, this formulation produces a tentacle that is most rigid near the center (stalk region) and most flexible toward the edges (pinnule region).

The transition in elasticity between the central and pinnule regions is determined by two sigmoidal terms: (i) the longitudinal transition between the oral disc and tentacle, given by

$$\sigma(\vec{X}) = 1 - \frac{1}{1 + \exp\left[-\lambda_s\left(R_t(\vec{X}) - r_o\right)\right]}, \tag{2.19}$$

where λ_s is a smoothing parameter and r_o is the radius of the oral disc; and (ii) the transverse transition across the tentacle width, given by

$$\tau(\vec{X}) = -\frac{1}{1 + \exp\left[\lambda_\tau\left(r_m(\vec{X}) - e_\tau\right)\right]} + \frac{1}{1 + \exp\left[\lambda_\tau\left(r_m(\vec{X}) + e_\tau\right)\right]}, \tag{2.20}$$

where λ_τ is a smoothing parameter, $e_\tau = 0.0002$ m defines the boundary of the central (stiff) region, and $r_m(\vec{X})$ is the distance from the tentacles midline in the transverse direction (i.e., across the tentacle thickness).

The variable elasticity along the length of the tentacle, l_η is defined through a piecewise function:

$$l_\eta(\vec{X}) = \begin{cases} 1 & R_t(\vec{X}) < 0.01 \\ l_{\eta_c}(\vec{X}) & 0.01 \leq R_t(\vec{X}) < r_p \\ 0 & r_p < R_t(\vec{X}) \end{cases}, \tag{2.21}$$

where $r_p = 0.9L_T + r_o$ is the length along which the tentacle starts to recede into the pinnule region, with L_T being the tentacle length and

$$l_{\eta_c}(\vec{X}) = \frac{1}{2} + \frac{1}{2} * \cos\left(\frac{\pi}{0.9L_T}\left(R_t(\vec{X}) - r_o\right)\right) \tag{2.22}$$

is the ramp down through the stalk. This produces variable elasticity along the length of the tentacle, making it most rigid near the oral disc and most flexible toward the tentacle tip.

Finally, note that the implementation of η in Eq. (2.18) serves as an approximation to the formal definition given in Eq. (2.12), and an example of its distribution is shown in Fig. 2.2.

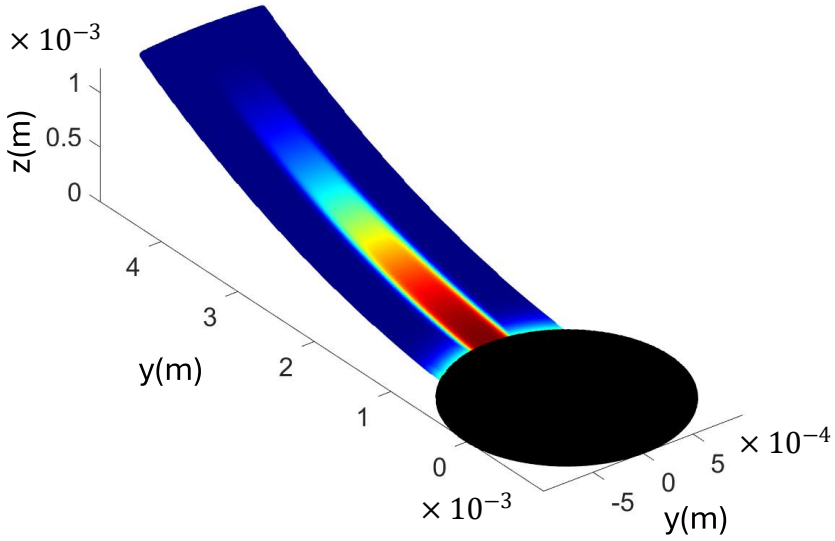


Figure 2.2: Tentacle elasticity, η. One tentacle and the oral disc is presented here. Warmer colors represent greater tentacle elasticity.

Next, we introduce the muscle tension, $T(\vec{X}, t)$, for Eq. (2.17), given by

$$T(\vec{X}, t) = T_{\max} a(t) m_{\text{loc}}(\vec{X}) \tag{2.23}$$

where $T_{\max}$ is the maximum tension, $a(t)$ is an activation function for the tension, and $m_{\text{loc}}(\vec{X})$ is the muscle location function corresponding to where the tension is applied [29, 30]. The activation function

$$a(t) = A_1 \left(\frac{1}{1 + \exp\left(-A_a(t_p - 0.15)\right)} - \frac{1}{1 + \exp\left(-A_d(t_p - 0.4)\right)} \right) \tag{2.24}$$

determines how the tension is applied along the muscles as time varies and is responsible for the resulting periodic pulsing behavior of the coral polyp. Here, A_a and A_d are scalars that control the ramp-up and ramp-down respectively, A_1 is a scalar such that $\max\left(a(t)\right) = 1$, and $t_p = t \bmod T_{\text{per}}$ which ensures that the activation function is periodic within the desired pulsing period, T_{per}. This can be seen plotted over the course of a single pulse in Fig. 2.3.

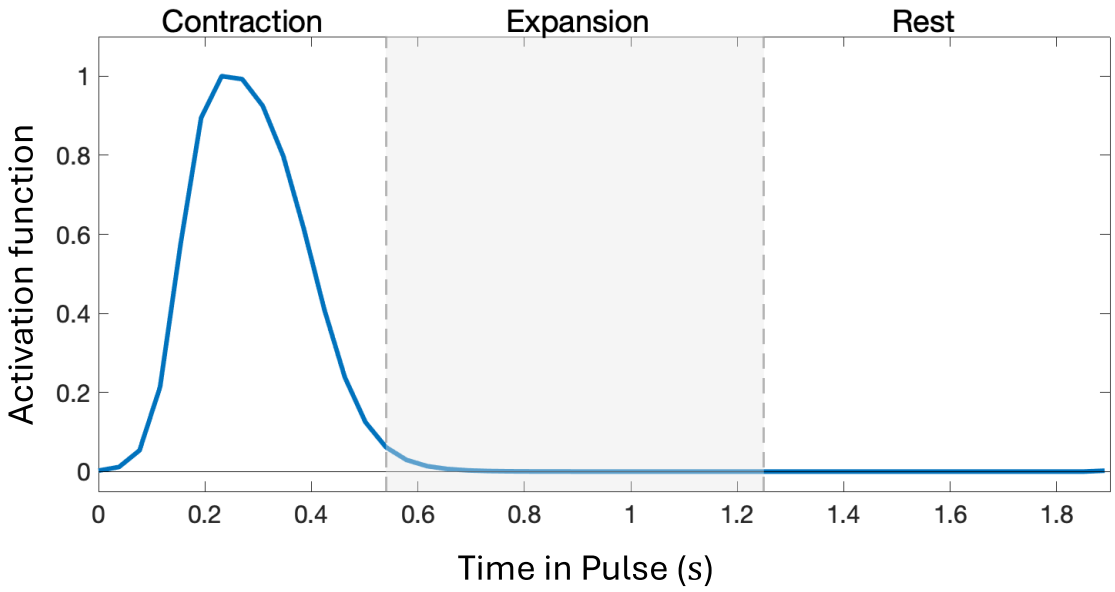


Figure 2.3: Muscle activation function, $a(t)$, over a pulsing period.

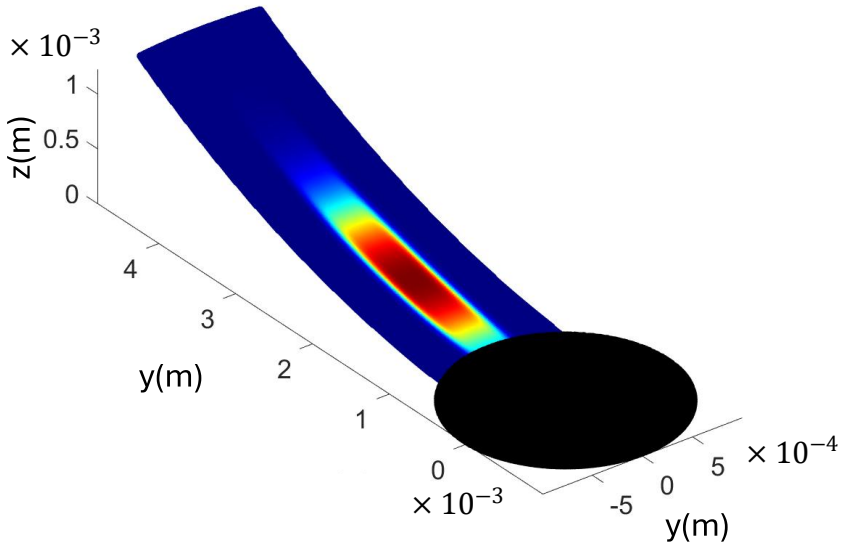


Figure 2.4: Muscle location, m_{loc}, and tension, $T(\vec{X}, 0)$. One tentacle and the oral disc is presented here. Warmer colors represent areas with greater muscle tension.

The muscle location function, $m_{\text{loc}}(\vec{X})$, is the location where the tension is applied and is given by

$$m_{\text{loc}}(\vec{X}) = h(\vec{X}) l_m(\vec{X}) \tau(\vec{X}) \tag{2.25}$$

where

$$h(\vec{X}) = \frac{1}{1 + \exp\left(-\lambda_c R_c(\vec{X})\right)} \tag{2.26}$$

determines if a point is on the top or bottom of a tentacle, with λ_c a smoothing parameter and

$$R_c(\vec{X}) = 0.0099 - \sqrt{x_1^2 + (x_2 - 0.01)^2 + X_3^2}, \tag{2.27}$$

such that $R_c(\vec{X}) = -0.0001$m on the bottom of the coral body and $R_c(\vec{X}) = 0.0001$m on the top of the coral body. Lastly,

$$\begin{aligned} l_m(\vec{X}) =& \frac{1}{1 + \exp\left(-\lambda_s \left(R_t(\vec{X}) - m_s\right)\right)} + \frac{1}{1 + \exp\left(-\lambda_c \left(R_t(\vec{X}) - m_e\right)\right)}, \\ m_s =& L_s L_T + r_o, \\ m_e =& L_e L_T + r_o, \end{aligned} \tag{2.28}$$

determines the length of the muscle, where $L_s = 0.05$ and $L_e = 0.45$ are dimensionless scalars indicating that the muscle starts at 5% of the tentacle length and ends at 45% of the tentacle length. This is used to calculate the actual starting and stopping point of the muscle though m_s and m_e, respectively. Finally, λ_s and λ_e are smoothing parameters for the transition at the start of the muscle and end of the muscle. Altogether, the muscle location for the reference configuration can be seen in Fig. 2.4 where warmer colors represent increased muscle tension.

This model results in a kinematic motion similar to that of the biological corals [93]. Motion starts with contraction, where the muscles tense and drive the tentacle in an inward and upward direction, as seen on the right of Fig. 2.5. Once fully contracted, expansion begins where the muscles relax and elasticity returns the coral to its resting configuration, seen on the left of Fig. 2.5. Finally, the polyp enters its resting phase where the muscles do not move, resulting in no active motion. Altogether, the total pulse period, T_{per}, is the sum of these three kinematic phases: the contraction period (T_c), the expansion period (T_e), and the resting period (T_r).

A summary of the parameters used to generate the biologically realistic (base) case of this muscle model can be seen in Table2.2. These are the parameters used in all example figures in this section.

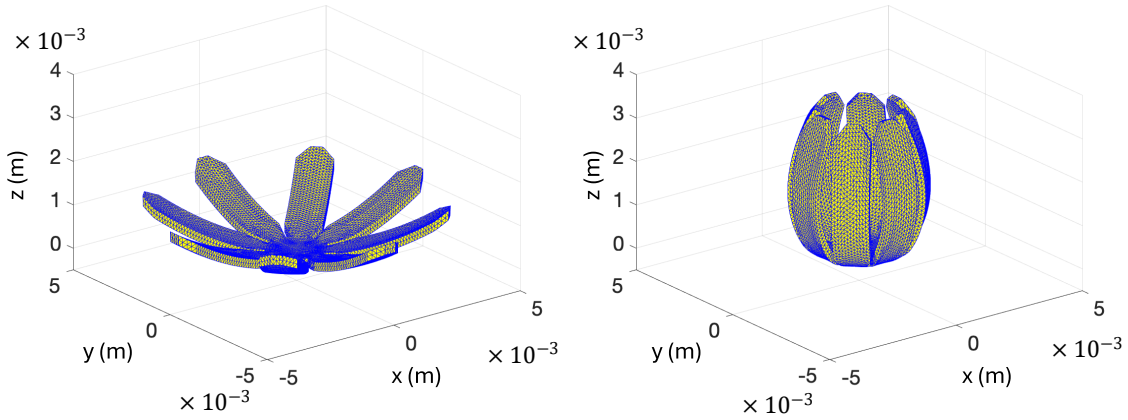


Figure 2.5: Coral model when fully expanded (left) and fully contracted (right).

Parameter	**Variable**	**Value**	**Units**
Radius of oral disc	r_o	0.001	m^{-1}
Elasticity of pinnule region	η_p	20	Pa
Elasticity of central tentacle region	η_t	5,000	Pa
Maximum tension	$T_{\max}$	4,500	N
Muscle width (from tentacle's centerline)	e_τ	0.0002	m
Muscle start fraction	L_s	0.05	-
Muscle end fraction	L_e	0.45	-
Elasticity smoothing parameter	λ_τ	50,000	m^{-1}
Surface smoothing parameter	λ_c	100,000	m^{-1}
Muscle start smoothing parameter	λ_s	4,000	m^{-1}
Muscle end smoothing parameter	λ_e	12,000	m^{-1}
Activation function normalization parameter	A_1	1/0.935	-
Activation function ramp-up	A_a	40	s^{-1}
Activation function ramp-down	A_d	20	s^{-1}

Table 2.2: Parameters used to define the tension-driven muscle coral model's motion [96].

2.4 Kinematic and Flow Analysis

As described in Section 2.2, simulations are run until they reach a periodic steady state, which typically occurs within 15 pulses. At this stage, the coral polyp has settled into a repetitive motion, allowing for meaningful analysis of its fluid-structure interactions. To characterize the system, we perform a series of analyses using the softwares VisIt [10, 37] and MATLAB [109]. These include the evaluation of tentacle kinematics to quantify motion patterns, vorticity to assess local mixing and vortex formation, volumetric flux to measure fluid transport (assessed via vertical and horizontal fluid velocities), and energy and power expenditure to determine the mechanical effort required for pulsation. Together, these analyses provide a comprehensive picture of how the coral polyp interacts with and influences its surrounding fluid environment.

2.4.1 Kinematic Comparison

A comparison of the motion of the corals throughout a pulse is conducted to understand both how the muscle and flow parameters impact the movement of the tentacles, as well as how this changing movement impacts the resulting fluid flow. This is done by tracking a two-dimensional slice of a single tentacle throughout a pulse and plotting the path that the tip of the tentacle takes. This process can be seen in Fig. 2.6. Additionally, we can compare the maximum ranges of motion achieved during a pulse, specifically the maximum height reached during contraction and the maximum width reached during expansion. This analysis is computed for only a single tentacle, as we assume symmetry in the coral polyp as it is pulsing.

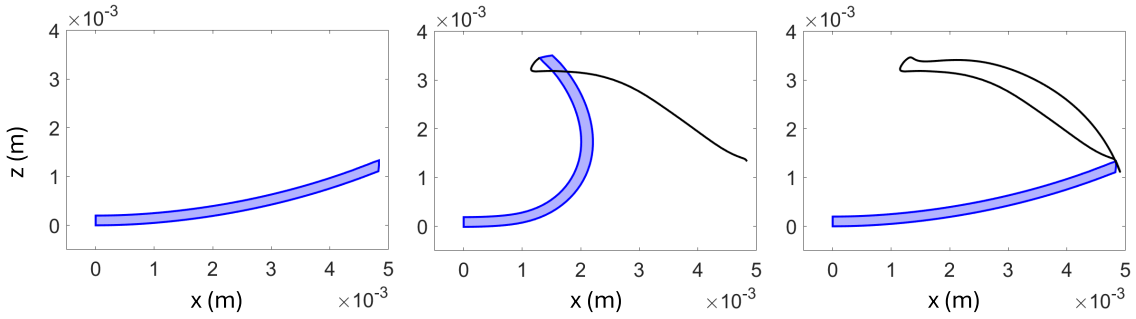


Figure 2.6: Tentacle tip tracking used for kinematic comparison. From left to right, the tentacle starts at rest, rises to its contracted position, and then expands to its resting position.

2.4.2 Vorticity

Vorticity, which quantifies the local rotation of a fluid, is used here to understand how vortices are generated as a result of tentacle pulsation. Mathematically, it is defined as the

curl of the velocity field:

$$\omega = \nabla \times \vec{u}. \tag{2.29}$$

Previous studies have shown that vorticity is generated at the tentacle tips, with oppositely spinning vortices forming on opposing tentacles. These vortices support an upward jet of fluid flow around the polyp [93]. Moreover, at higher Reynolds numbers, vortices can separate from the tentacles and continue transporting fluid upward, while at lower Reynolds numbers they dissipate rapidly [92].

In this study, we continue this analysis using three-dimensional vorticity contours. Our primary focus is on the generation of vortices and vortex rings, which serve as indicators of upward fluid transport.

To further quantify vorticity, we examine the average magnitude within a region of interest near the coral. Specifically, we use the cylinder V_A, shown in Fig. 2.7, positioned above the contracted coral polyp with a height of L_T and a radius of 0.002 m. This approach provides a measure of fluid rotation near the coral and serves as an indicator of fluid mixing.

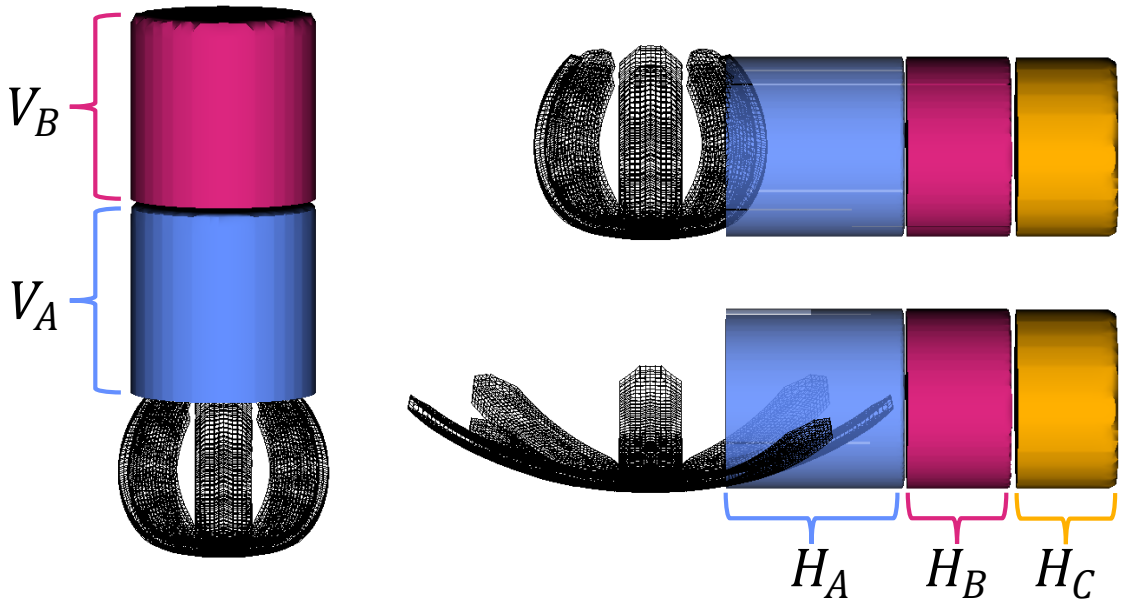


Figure 2.7: Left: Regions of study for magnitude of vorticity and vertical flow. Right: Regions of interest for horizontal flow, as compared to a fully contracted (top) and fully expanded (bottom) coral polyp.

2.4.3 Volumetric Flux

We measure volumetric flux through the average vertical, u_y, and horizontal, u_x, velocities in regions of interest near the coral polyp. Starting with vertical flow, we average u_y in a cylinder placed above the coral polyp, as seen in Fig. 2.7. While previous

studies have computed this metric in boxes [91, 92] or along planes [93, 96], we have chosen to utilize cylinders as they more closely mirror the morphology of the coral polyp and the flow regions generated by its pulsing. This metric is computed for two regions which each have a height of L_T meters and a radius of 0.002 meters, but vary in their location as follows: V_A directly on top of the fully contracted coral polyp, V_B directly on top of region V_A.

Secondly, we explore horizontal flow by averaging u_x in cylinders placed to the side of the coral polyp, as seen in Fig. 2.7. This metric is computed for three total regions which have a diameter of a fully contracted coral polyp but vary in their length and placement: H_A directly to the side of the fully contracted coral polyp with a length that ends at the fully expanded coral polyp, H_B directly to the side of the fully expanded coral polyp with a length of $L_T/2$, H_C directly to the side of H_B with a length of $L_T/2$.

Since we do not compute volumetric flux directly, these velocities serve as a proxy for the fluid transport generated by the coral. These measurements allow us to estimate the spatial extent and strength of flow induced by the coral. Previous studies found that corals only impact the fluid flow locally, implying that we should see the greatest generation of fluid flow in the regions closest to the coral polyp, while relatively little fluid flow should be found as we move further away from the coral polyp [92].

2.4.4 Energy and Power Expenditure

There are several ways to quantify the energy and power expenditure of pulsing corals throughout their motion. Here, we focus on measures that capture both the mechanical effort of the coral structure and the energy transmitted to the surrounding fluid.

Energy, or the capacity to do work, can be defined in multiple ways. One approach is a tension-based formulation:

$$E_{\text{tension}} = \frac{1}{T_{\text{per}}} \int_0^{T_{\text{per}}} \sum_{i=1}^{N} T_i(\vec{X}, t) \left\| \Delta\chi_i(\vec{X}, t) \right\|_2 \, dt, \tag{2.30}$$

where T_{per} is the duration of a pulse, N is the number of Lagrangian points and $\Delta\chi(\vec{X}, t)$ is the structure's displacement from the reference configuration [96]. The 2-norm in $\|\Delta\chi_i(\vec{X}, t)\|_2$ is taken over the three displacement components of mesh point i at each time t within the pulse period. Alternatively, a force-based formulation defines energy as

$$E_{\text{force}} = \int_0^{T_{\text{per}}} \sum_{i=1}^{N} \left| \vec{F}_i(t) \cdot \vec{U}_i(t) \right| \, dt, \tag{2.31}$$

where $\vec{F}_i$ is the Lagrangian force at mesh point i in the Lagrangian discretization (see Eq. (2.1)), and $\vec{U}_i$ is the corresponding Lagrangian velocity. In both cases, energy has units of N m. The tension-based energy characterizes the instantaneous work done by the corals structure, while the force-based energy captures the total work transmitted to the fluid over time.

Similarly, power, the rate of energy transfer, can be expressed through either model. The tension-based power is given by

$$P_{\text{tension}} = \frac{1}{T_{\text{per}}} \int_0^{T_{\text{per}}} \sum_{i=1}^{N} T_i(\vec{X}, t) \left\| \vec{U}_i(\vec{X}, t) \right\|_2 \, dt. \tag{2.32}$$

The force-based definition is

$$P_{\text{force}} = \sum_{i=1}^{N} \left| \vec{F}_i(t) \cdot \vec{U}_i(t) \right|. \tag{2.33}$$

Both have units of N m s. Tension-based power quantifies the corals internal energetic expenditure, while force-based power represents how effectively that energy is transmitted into the fluid.

2.5 Parameter Variation

As previously discussed, we first establish a base case representative of a biological coral polyp, using the fluid parameters listed in Table 2.1 and the muscle parameters in Table 2.2. We then conduct a parameter study in which these values are systematically varied. Previous studies have primarily examined parameter values close to those observed biologically [92, 93] or explored only a limited set of variations [96, 97]. By considering a broader range of parameters, we aim to better understand how these variables influence the resulting flow dynamics and energetic costs. This approach also allows us to identify parameter regimes that generate specific kinematic motions and enhance flow production.

The parameter study begins by varying the Reynolds number according to $Re = 2^k$, where $k = 0, 1, 2, \ldots, 9$, through adjustments to the fluid dynamic viscosity μ in Eq. (2.4). As Re characterizes the balance between inertial and viscous effects in the fluid, it provides a fundamental measure for understanding how changes in the flow regime influence coral kinematics and fluid transport.

Beyond Re, several muscle-related parameters governing coral pulsing can also be varied, providing deeper insight into how these parameters influence kinematics and fluid flow. In Santiago et al. [96], the authors varied: (1) the maximum tension exerted by the corals muscles, $T_{\max}$; (2) the tentacle elasticity parameter, η_t, which determines the shear elastic modulus η in Eq. (2.18) with proportional muscle tension such that $T_{\max} = 0.9\eta_t$; and (3) the pulsing frequency, $1/T_c$, where T_c is the contraction period. These parameters were tested at biologically relevant values, with each varied incrementally to assess its effect on coral performance.

Building upon this framework, we explore four main muscle parameters:

1. Maximum muscle tension (with constant elasticity): $T^*_{\max} = \frac{1}{2}T_{\max}, \frac{3}{2}T_{\max}$,
2. Maximum muscle tension with proportional elasticity: $T^*_{\max} = \frac{1}{2}T_{\max}, \frac{3}{2}T_{\max}$ with $\eta_t = \frac{T_{\max}}{0.9}$,

3. Tension-elasticity ratio: $\psi = T_{\max}/\eta_t$ varied by adjusting η_t such that $\psi = 0.45, 1.5$,

4. Muscle end fraction: $m_e = L_e L_T + r_0$ in Eq. (2.28) with $L_e = 0.5, 1.5$.

Each of these simulations is conducted for $Re = 1$, 16, 32, 64, 128, 512. For higher Reynolds numbers (e.g. $Re = 512$) or extreme muscle parameters, some simulations were either too slow to run to completion (e.g., one case ran for 20 days of compute time to reach 15 pulses and had not yet reached steady state) or failed to converge after many extra pulses, and were therefore omitted.

A summary of the varied parameters is provided in Table 2.3, while the specific parameter combinations and the number of simulations performed for each set are listed in Table 2.4.

Name	**Parameter Varied**
Reynolds number (initial simulations)	Re
Maximum muscle tension (constant elasticity)	$T_{\max}$
Maximum muscle tension with proportional elasticity	$T_{\max}$, η_t
Tension–elasticity ratio	$\psi = T_{\max}/\eta_t$
Muscle end fraction	L_e

Table 2.3: Summary of fluid and muscle parameters varied across simulation studies.

$T_{\max}$	η_t	$\psi = T_{\max}/\eta_t$	L_e	# Varied Re Simulations (with varied Re)
4,500	5,000	0.9	0.45	16
2,250	5,000	0.45	0.45	6
2,250	2,500	0.9	0.45	6
6,750	5,000	1.35	0.45	5
6,750	7,500	0.9	0.45	5
4,500	10,000	0.45	0.45	6
4,500	3,000	1.5	0.45	5
4,500	5,000	0.9	0.225	6
4,500	5,000	0.9	0.675	5

Table 2.4: Summary of muscle parameter combinations and the number of simulations performed with varying Reynolds number.

We elected not to vary the pulse frequency or other time-related parameters. Previous work found that altering pulse frequency prevented full contraction of the coral polyp [96]. Additionally, preliminary tests with varied resting times showed minimal impact on overall fluid transport, aside from allowing more time for the fluid to settle toward the coral polyp. As such, these parameters were excluded from further analysis. Other parameters could

also be varied within the model, but we focused on those believed to have the largest impact on the resulting kinematics and fluid flow.

2.6 Results

The following sections examine how variations to both fluid and muscle parameters influence the kinematics, fluid flow, and overall energetic performance of the pulsing coral system. Together, these analyses provide a comprehensive view of how both environmental and biological factors shape the coral polyps fluid-structure interactions. All analyses are performed after the simulations reach steady state, so that each pulse exhibits similar behavior.

2.6.1 Fluid Parameter Variation

In this section, we present results from the fluid parameter variations, with muscle parameters held constant at the values listed in Table 2.2.

Kinematic Comparison

The influence of Reynolds number (Re) on tentacle kinematics is shown in Fig. 2.8. As Re increases, the tentacles exhibit a larger range of motion.

The shape of the tentacle trajectory also changes with Re. At lower Re, the path forms an approximate figure-8, as seen for $Re = 8$ in Fig. 2.8. Laboratory studies using particle image velocimetry have observed a similar, though more pronounced, figure-8 motion in coral polyps [67]. Since this motion is only reproduced at low Re in our simulations, it suggests that our baseline parameters may not exactly capture the biological conditions of the coral.

To quantify this behavior, we measured the maximum height (reached during contraction) and the maximum width (reached during expansion), presented in Fig. 2.9. At low Re, both measures increase sharply with Re, indicating that viscous forces strongly limit movement. Near the biological Re, however, the range stabilizes, suggesting that the coral musculature can overcome viscous resistance and achieve its characteristic movement.

Additionally, we can compare the time points at which the tentacle tip reaches its point of greatest contraction and expansion. For each of the varied Reynolds number simulations, the time at which the tentacle reaches its point of greatest contraction is relatively stable, occurring between $t = [0.54, 0.66]$. This occurs because contraction is controlled by tension driving the tentacle inward. The time at which the tentacle reaches its greatest expansion, however, varies substantially, occurring between $t = [1.12, 1.67]$, with maximum expansion being achieved more quickly as Re increases. Note that for $Re \leq 16$, the greatest point of expansion is achieved before contraction begins, and has therefore been left out of this comparison. This is due to the fact that expansion is dominated by passive elastic forces pulling the coral tentacle down to its resting position.

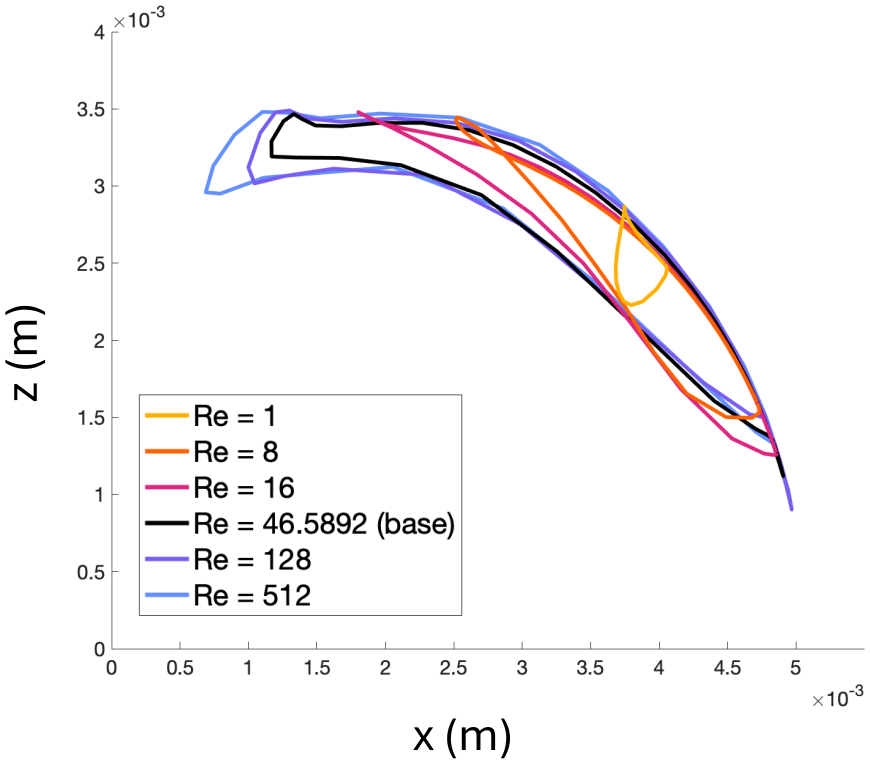


Figure 2.8: Tentacle tip path for varied Reynolds numbers.

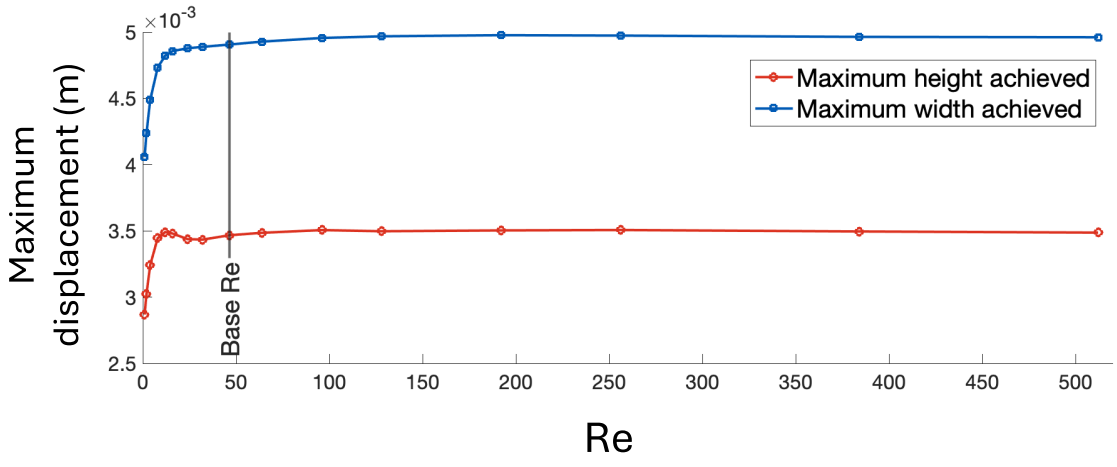


Figure 2.9: Maximum displacement of coral tentacle during pulsation. The maximum height is achieved during contraction while the maximum width is achieved during expansion.

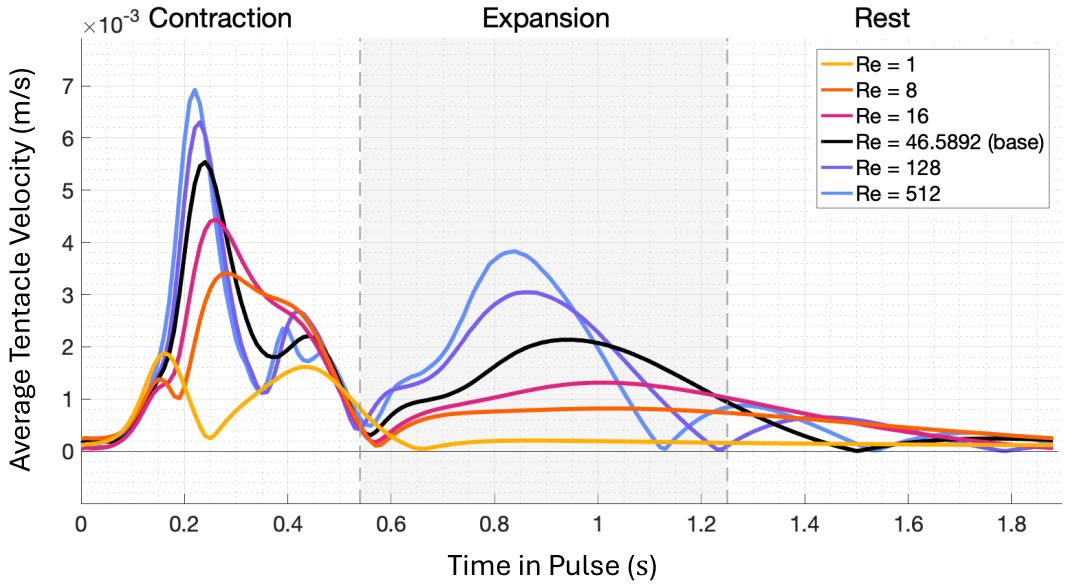


Figure 2.10: Average magnitude of tentacle velocity (2-norm) over a single tentacle mesh, for different Reynolds numbers during the pulsing cycle.

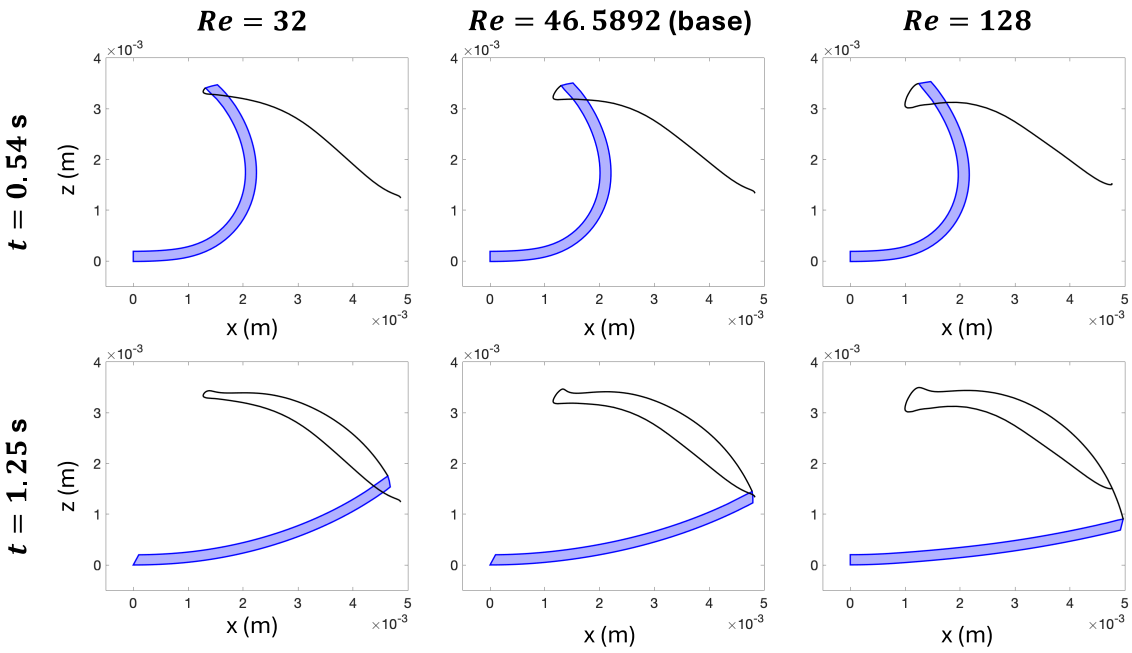


Figure 2.11: Comparative tentacle trajectories. Top row: $t = 0.54$ (end of contraction), Bottom row: $t = 1.25$ (end of expansion), Left column: $Re = 32$, Middle column: $Re = 46.5892$ (base), Right column: $Re = 128$.

As demonstrated in Fig. 2.10, the maximum tentacle velocity increases as Re increases (we will also see that as Re increases, greater fluid flow is achieved), reflecting the reduced viscous effects and stronger inertial effects at higher Reynolds numbers. This allows the tentacle to return to its resting position more quickly.

This trend is demonstrated in Fig. 2.11, where the tentacle tip path is shown for three increasing Reynolds numbers at two representative time points: the end of contraction and the end of the expansion phase. Here it can be seen that the $Re = 32$ and base cases reach similar expansion positions by the end of the expansion period. However, the higher Reynolds number case ($Re = 128$) has progressed further along the expansion trajectory, reaching a lower vertical position.

Vorticity

Vorticity contours for snapshots over the pulsing period are shown in Fig. 2.12 for a few cases, specifically $Re = 1$, 46.5892 (base), and 128. At $Re = 1$, there is minimal vortex generation, consistent with the limited tentacle motion observed at lower Reynolds numbers. At the base Re, a vortex ring is formed, indicating an upward jet of fluid flow that pushes water away from the coral. This vortex ring appears as the rest period is ending and expansion begins, maintaining upward flow even while the coral is not actively pulsing. Despite this, some of the vorticity generated in the previous pulse dissipates, allowing the fluid to partially relax before the next pulse. This creates a biological advantage as the coral polyp does not need to continuously expend energy to maintain flow generation. Finally, at $Re = 128$, vortex generation is much more pronounced, with strong vortex shedding and complex flow structures, including vortex rings that are formed and pushed upward, suggesting enhanced fluid transport and mixing at higher Re values. Biologically, this may be disadvantageous, as the strong upward flow prevents the local accumulation of carbon-rich water needed for photosynthesis. In contrast, such an effect may be inconsequential in industrial applications. This is consistent with prior work, which found that pulsing corals generate oppositely swirling vortices during contraction that form vortex rings [92, 93, 96].

Looking at the spatially averaged magnitude of the vorticity, presented in Fig. 2.13, tells a similar story to the vorticity contours. As Re increases, the mean vorticity magnitude rises correspondingly. This demonstrates that increasing Re amplifies local fluid flow and mixing near the coral polyp.

Interestingly, we find that greater tentacle motion is not tied to increased vortex generation. Beyond our low Re simulations ($Re \leq 16$), the tentacle kinematics only changed modestly, yet vorticity continues to rise. This indicates a shift in the resulting fluid dynamics: once inertial effects begin to dominate over viscosity, even similar motions can shed stronger vortices and drive greater fluid flow. The enhanced vortex generation thus reflects a regime transition rather than increased mechanical input.

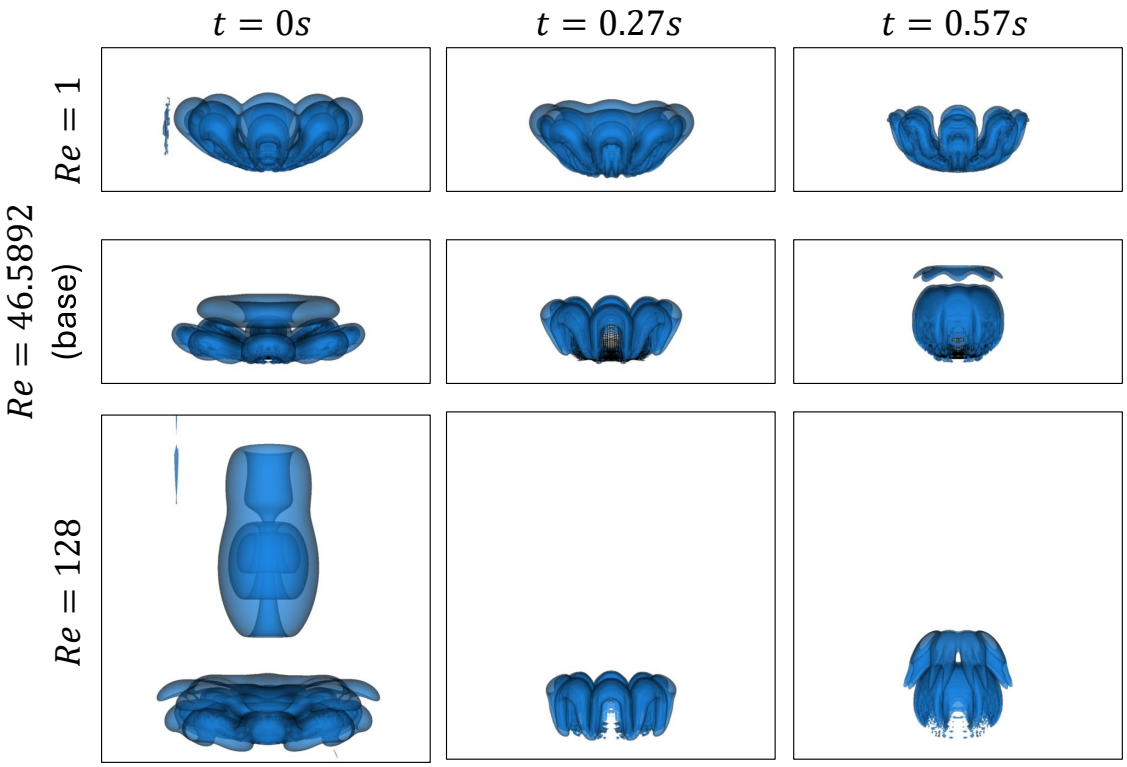


Figure 2.12: Vorticity contours for various Reynolds numbers at snapshots of the pulsing period. Top row: $Re = 1$, Middle row: $Re = 46.5892$ (base), Bottom row: $Re = 128$. Left column: $t = 0$ (start of contraction), Middle column: $t = 0.27$ (halfway through contraction), Right column: $t = 0.57$ (beginning of expansion).

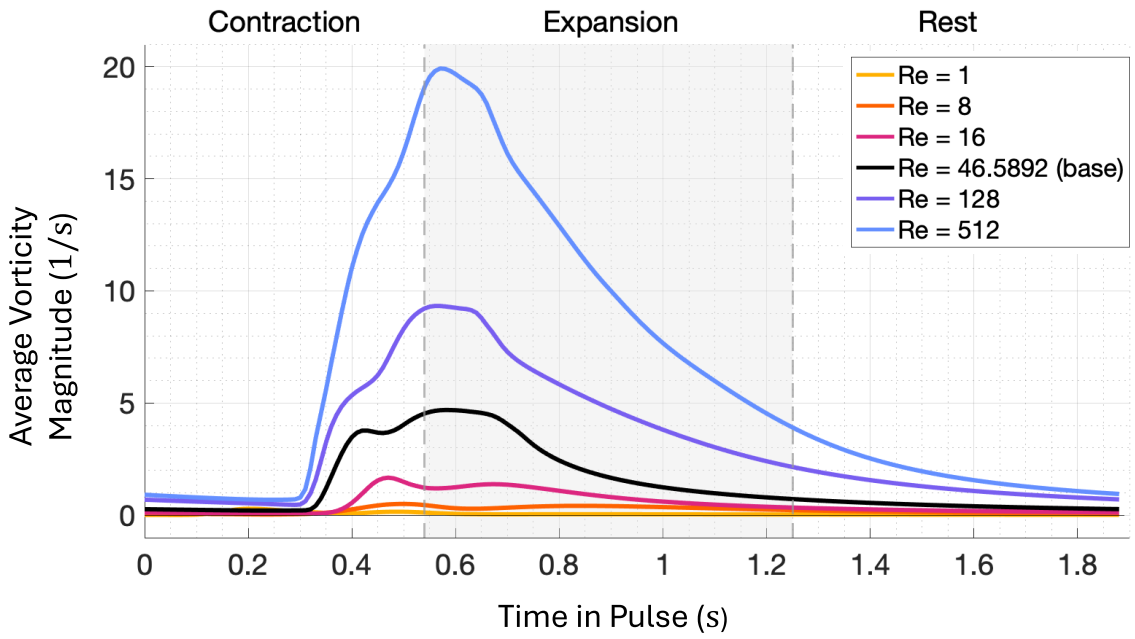


Figure 2.13: Average magnitude of vorticity in region V_A (see Fig. 2.7) for varying Reynolds numbers.

Volumetric Flux

The average vertical flow for regions V_A and V_B for varying Reynolds numbers is shown in Fig. 2.14 (see Fig. 2.7 for region definitions).

We first consider cylinder V_A, the region closest to the coral polyp. It can be seen that the vertical flow begins near zero, increases as contraction ends, and decreases during expansion before returning to near zero by the end of the rest period. The maximum vertical flow achieved by the end of contraction increases monotonically with Re, indicating that higher Reynolds numbers drive stronger upward flow directly above the coral polyp. This corresponds with the results of our vorticity studies, where stronger vorticity was also generated as Re increased. During expansion, the base and lower Re cases exhibit negative flow, meaning fluid falls back toward the coral. Higher Re cases maintain upward flow that gradually diminishes during expansion and rest. As with our vorticity study, this result is consistent with findings from previous studies [92, 93, 96].

From a biological standpoint, the lower Re cases generate little upward transport and thus would not effectively drive the efflux of oxygen-rich water. Conversely, very high Re cases may generate excessive upward flow, enhancing oxygen removal but reducing the opportunity for carbon dioxide-rich water to flow toward the coral polyp. The base case appears to strike a balance, producing sufficient flow for both oxygen efflux and carbon uptake.

Moving farther from the coral, cylinder V_B probes the extent of the polyps influence on

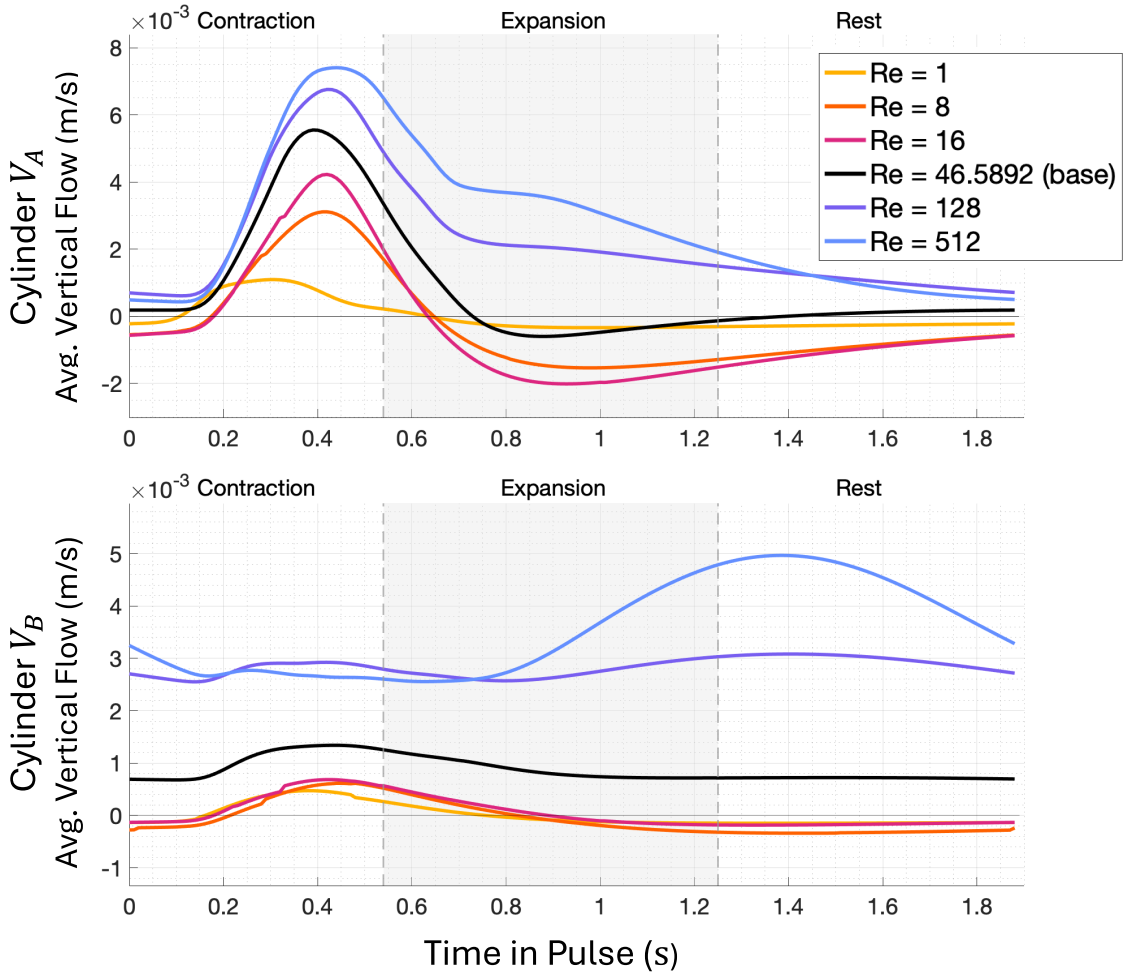


Figure 2.14: Results for average vertical flow. Top: Vertical flow in cylinder V_A. Bottom: Vertical flow in cylinder V_B.

the surrounding fluid. Here, very little flow occurs in the base and lower Re cases, with upward flow only observed at higher Re. This is consistent with previous findings that corals primarily alter flow in their immediate vicinity. This finding may be significant for industrial mixing applications, where a larger Re is needed to generate fluid flow outside of the local region.

Finally, we again see that increased vertical fluid transport is not driven by changes in the tentacle kinematics beyond a certain Re. This implies that vertical velocity scales with vorticity generation, not with the extent of tentacle motion. This reinforces that once inertial forces dominate, the fluids own dynamics, not increased deformation of the coral, govern transport efficiency.

Shifting focus to the horizontal fluid flow, the average horizontal flow for regions H_A, H_B, and H_C for varying Reynolds numbers can be found in Fig. 2.15 (see Fig. 2.7 for region definitions).

In H_A, the flow begins near zero, becomes negative during contraction, shifts positive during expansion, and returns to near zero during rest. Physically, this reflects fluid being pulled toward the polyp during contraction and expelled outward during expansion. Differences across Re are relatively minor: smaller Re cases reach their minimum earlier in contraction, and slight variations occur in maximum flow and its timing. Thus, near the pulsing tentacles, Re has little effect on horizontal transport.

Farther away, in H_B and H_C, the smaller Re cases generate the strongest negative flow during contraction, though at reduced magnitudes compared to H_A. At the highest Re, greater horizontal flow persists farther from the polyp. Physically, this indicates the base case and lower Re simulations drive an influx of carbon dioxide-rich water towards the polyp. Alternatively, higher Re simulations have such a high generation of flow away from the polyp that this phenomena cannot occur. Additionally, these results support the theory that biological corals mainly influence their local environment, while higher Re may extend the range of hydrodynamic impact.

Finally, it should be noted that vertical flow reaches much greater magnitudes than horizontal flow. This indicates that pulsing is highly effective at driving upward flow while less effective at generating lateral transport. For coral colonies, this is advantageous: investing energy in horizontal mixing would be unnecessary, as neighboring polyps influence those regions. In contrast, for biomimetic mixer applications, this suggests that multiple devices would be required for effective lateral transport, whereas a single mixer could be optimized for vertical mixing. Based on this result, we will only examine the vertical flow moving forward.

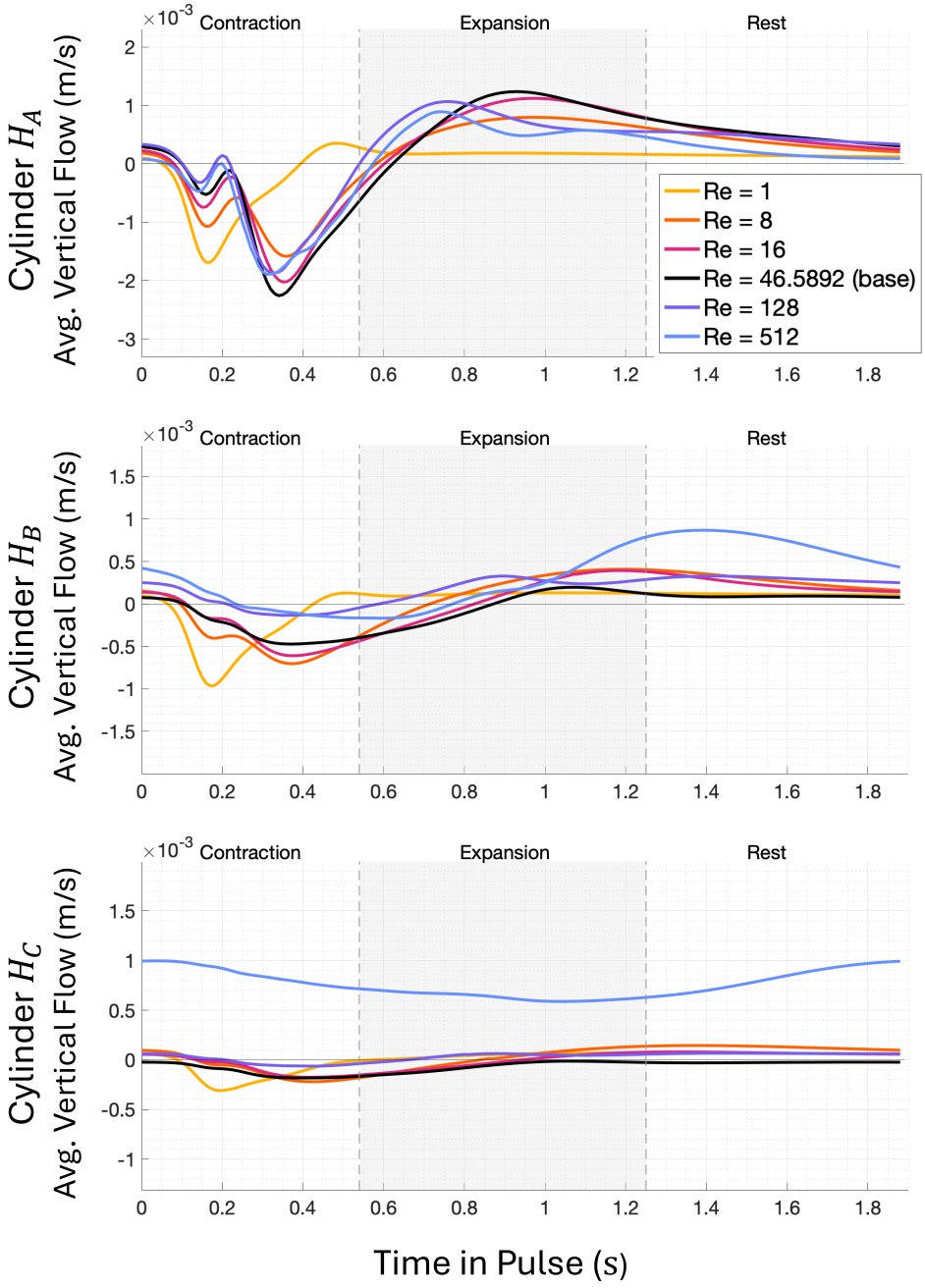


Figure 2.15: Results for average horizontal flow with the top plot corresponding to cylinder H_A, the middle to H_B, and the bottom to H_C.

Energy and Power Expenditure

A comparison of energy and power computed using the tension-based and force-based definitions is shown in Fig. 2.16 for varying Reynolds numbers.

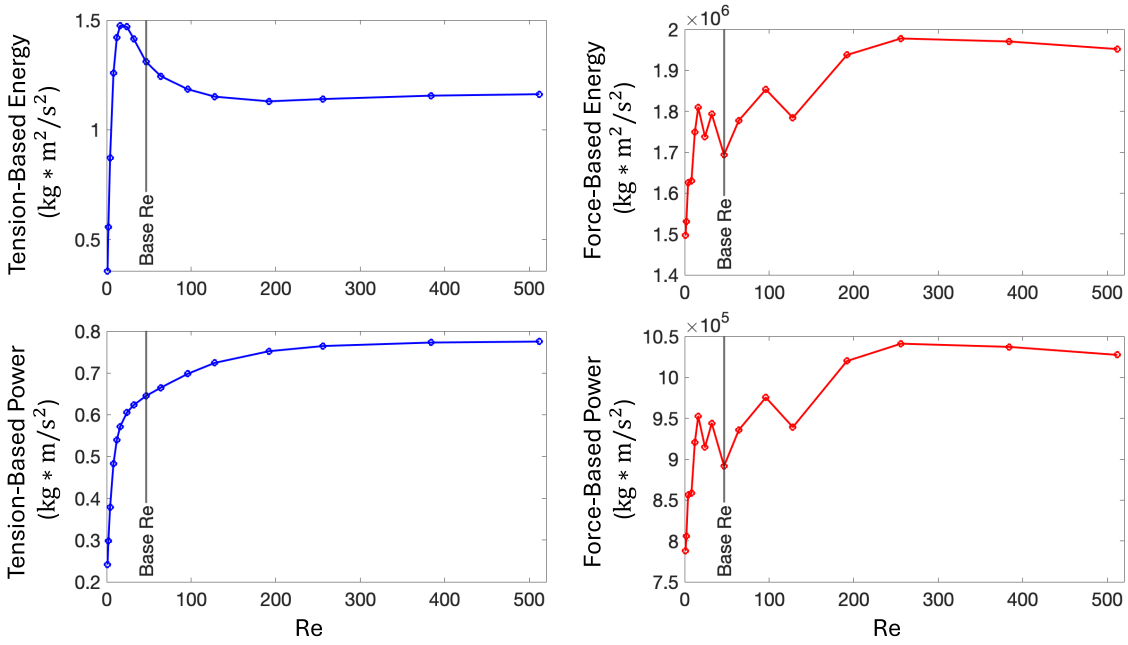


Figure 2.16: Comparison of energy and power with tension-based (left) and force-based (right) definitions.

Starting with energy (top panel of Fig. 2.16), the two definitions display distinct trends. Because muscle tension is held constant across all Re simulations, the tension-based energy primarily reflects the tentacles displacement from its reference configuration. As expected, this measure is lowest at low Re, where motion is minimal. This metric peaks at intermediate Re and then decreases again at higher Re values. The decline occurs because simulations at high Re return to their resting position more quickly, as previously evidenced in Fig. 2.11. Additionally, higher Re simulations achieve both a greater contraction and a lower expansion point, resulting in a smaller average displacement, as quantified in Fig. 2.17. In contrast, the force-based energy shows more variability, generally starting low, decreasing slightly at intermediate Re, and then increasing at higher values. As shown in Fig. 2.10, the maximum tentacle velocity increases with Re, and at low Re not only is the peak velocity smaller, but the velocity remains low for most of the pulsing cycle. At the same time, the Lagrangian force decreases as Re increases (Fig. 2.18), because viscous resistance is lower and the coral must apply less force to achieve contraction. The combination of increasing velocity and decreasing force produces the observed trend in the force-based energy: low at small Re due to slow motion, slightly decreasing at intermediate Re as forces drop faster than

velocities rise, increasing at higher Re as velocities become large enough to dominate the integral, and finally beginning to decrease at extreme Re values as force nears zero.

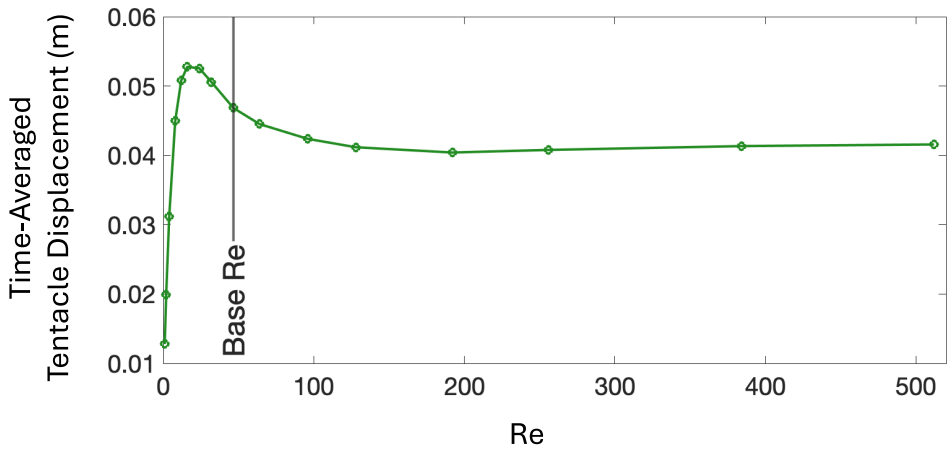


Figure 2.17: Time-averaged tentacle displacement for various Reynolds numbers. Computed as the 2-norm from the reference configuration and then averaged over the pulse period.

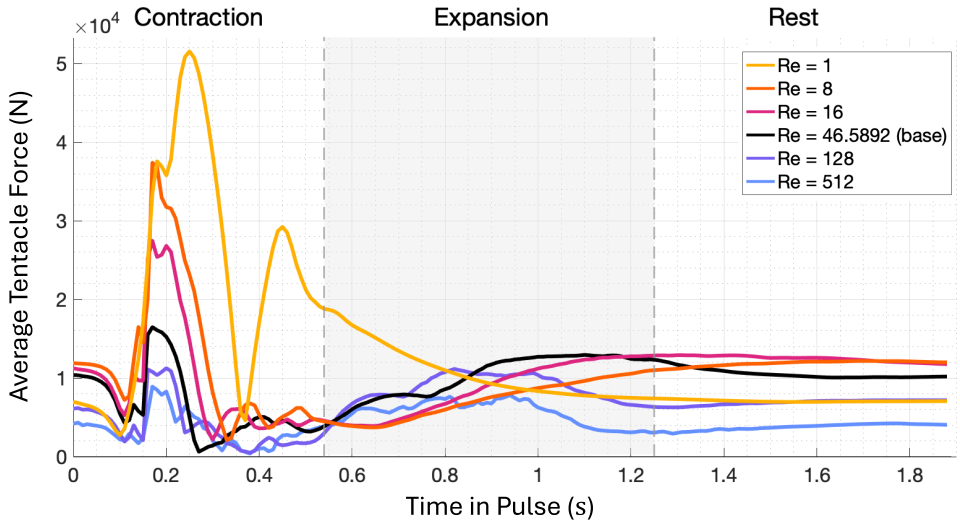


Figure 2.18: Average magnitude of tentacle force (2-norm) over a single tentacle mesh, for different Reynolds numbers during the pulsing cycle.

Turning to power (bottom panel of Fig. 2.16), the tension-based power rises monotonically with Re,scaling with the tentacles Lagrangian velocity. Consequently, faster motion at higher Re produces greater power, with the largest increase observed from low to intermediate Re where the Lagrangian velocity rises most sharply. The

force-based power mirrors the behavior of the force-based energy: it begins low, shows a small drop at intermediate Re, and generally increases as Re grows.

Together, these results highlight that the two approaches capture different aspects of the coral polyps energetic output. The tension-based metrics describe the internal cost of motion, how much and how quickly the coral moves, whereas the force-based metrics quantify the effort exerted on the surrounding fluid. We expect greater force generation at low and intermediate Re, where the coral must overcome strong viscous resistance.

In summary, lower viscous forces, which restrict kinematic activity and fluid transport, are associated with reduced energetic costs. The Reynolds number at which cost peaks depends on the metric used, yet overarching patterns emerge, with energy and power generally rising with increasing Re.

2.6.2 Muscle Parameter Variation

In this section, we present results for the muscle parameter variations, listed in Tables 2.3 and 2.4.

Kinematic Comparison

We begin by examining how variations in $T_{\max}$, $\psi = T_{\max}/\eta_t$, and L_e influence tentacle kinematics. As with changes in Re, these parameters can significantly alter the range and trajectory of polyp motion. Results are shown in Figs. 2.19–2.22, which illustrate the effects of varying the tension-elasticity ratio ψ, maximum muscle tension $T_{\max}$, and muscle length L_e. Detailed tentacle tip trajectories for varied L_e are provided in Fig. 2.23.

Across all muscle parameter variations, the range of tentacle motion increases with Re, with a pronounced increase between $Re = 1$ and $Re = 32$. At low Re, variations in muscle parameters result in different resting positions of the tentacle tip, which determine the start and end points of the kinematic path. The specific fluid-structure mechanisms driving these differences are not fully clear, but they highlight the sensitivity of resting positions to muscle properties.

Examining each muscle parameter individually, we first consider the muscle-tension ratio, ψ, with constant muscle tension, $T_{\max} = 4,500$, shown in Fig. 2.19. As ψ increases (i.e., tentacle elasticity decreases), the range of motion generally increases. This trend persists across different $T_{\max}$ values, demonstrated in Fig. 2.20. In this figure, decreases in $T_{\max}$ are shown in orange while increases in $T_{\max}$ are shown in red, with dashed lines indicating lower ψ (greater elasticity) and solid lines with markers indicating higher ψ (reduced elasticity). The exception occurs at high Re (e.g., $Re = 512$) with increased $T_{\max}$, where the tentacle has already reached near-maximal motion and differences due to ψ are less apparent. Focusing solely on $T_{\max}$, we also observe that increasing tension generally increases the range of motion, though this effect can be partially offset by greater tentacle elasticity. To isolate the effect of $T_{\max}$, we held ψ constant at 0.9 by adjusting η_t as $T_{\max}$ was varied (Fig. 2.21). As in the other cases, increasing $T_{\max}$ leads to a larger tentacle range of motion.

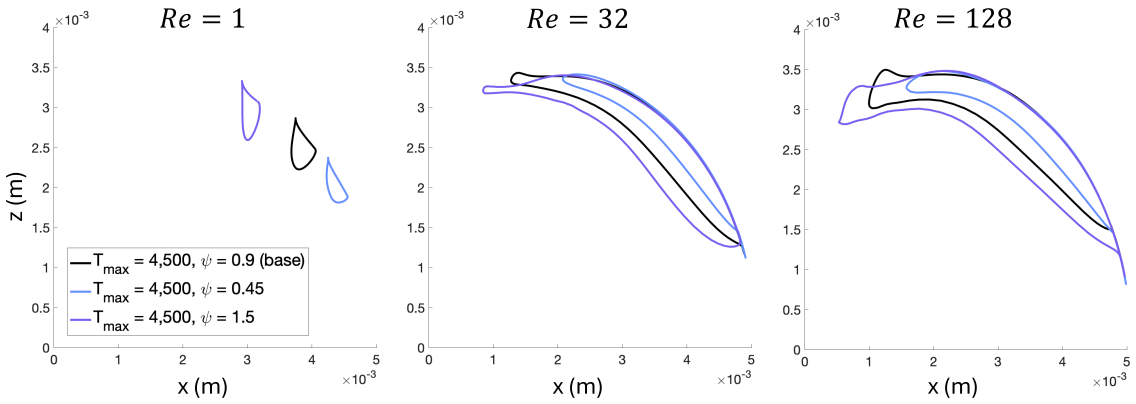


Figure 2.19: Tentacle kinematics with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

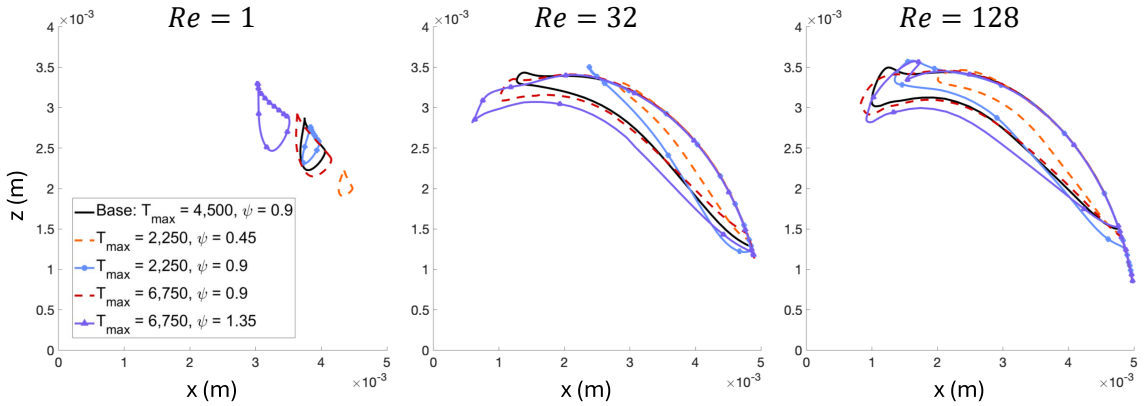


Figure 2.20: Tentacle kinematics with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

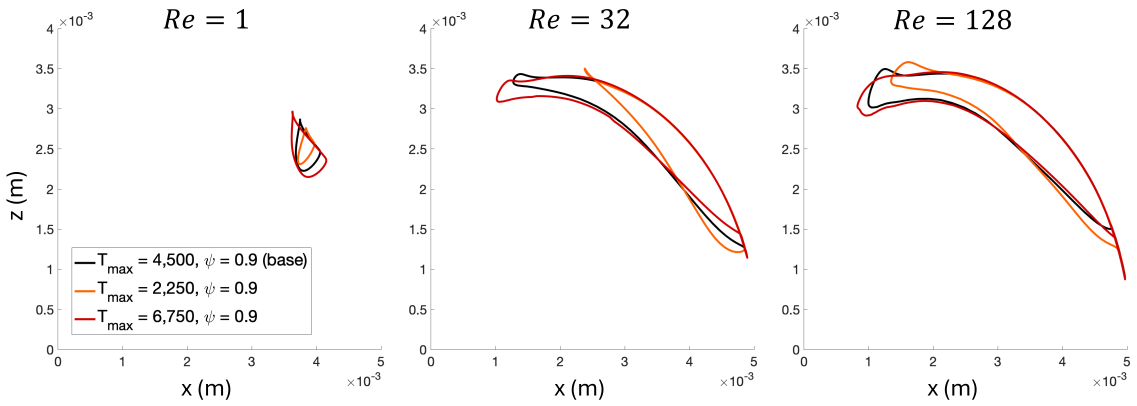


Figure 2.21: Tentacle kinematics with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Turning to variations in muscle length, shown in Fig. 2.22, increasing L_e both increases the kinematic range and alters the pulse pattern of the tentacle. The tentacle with decreased muscle length follows a muted path similar to the base case, whereas increased length causes the tentacle to curve inward, producing a pronounced bulging tip trajectory (Fig. 2.23). It should be noted that for $Re = 1$, the tentacle path corresponding to the simulation with increased muscle length (the yellow path) does not fully close. This simulation was run to steady state, indicating that the small offset is not due to the initial conditions. Instead, it likely results from small numerical effects in the fluid-structure interaction solver or slight asymmetries introduced by the increased muscle length, which can produce a minor drift in the final tentacle-tip position over a pulsing period.

Overall, muscle length has the strongest effect on tentacle kinematics, while changes in tension and elasticity produce more moderate adjustments. These results highlight that while Re modulates overall motion amplitude, muscle parameters, particularly length, determine the detailed shape and extent of tentacle trajectories.

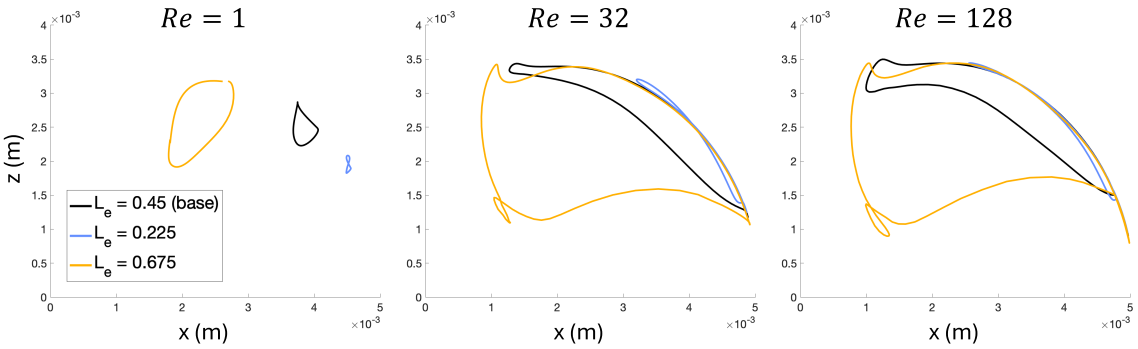


Figure 2.22: Tentacle kinematics with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

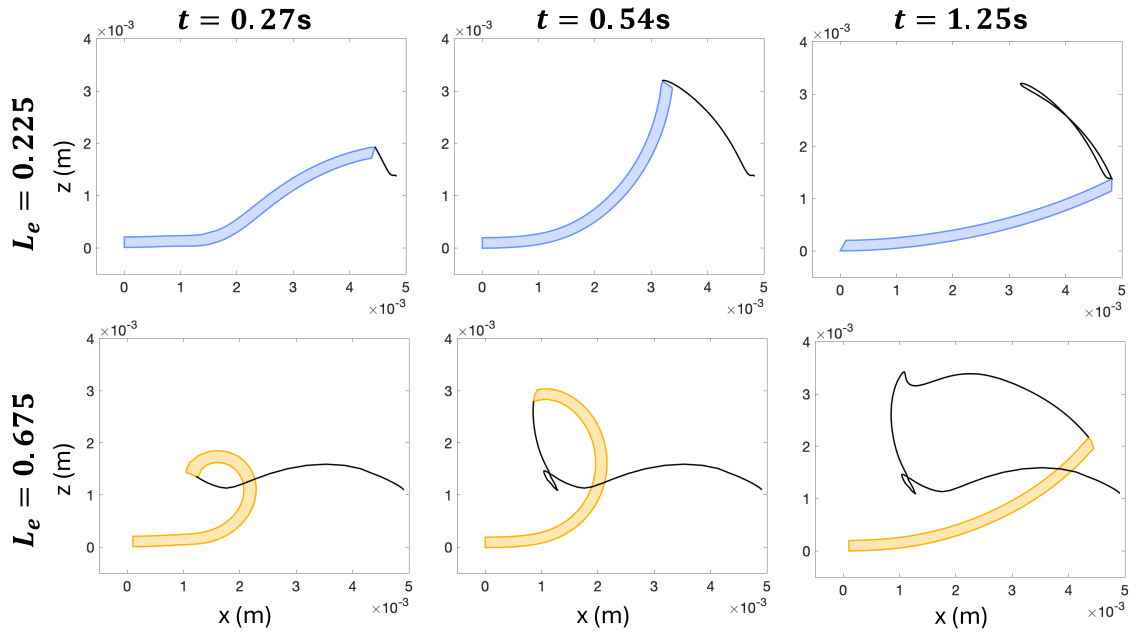


Figure 2.23: Comparison of tentacle tip tracking for $Re = 32$ with varied muscle length, L_e. Top: Decreased muscle length, Bottom: Increased muscle length.

Vorticity

We next examine the spatially averaged magnitude of vorticity within the region V_A (see Fig. 2.7). Results are shown in Figs. 2.24–2.27, which illustrate the effects of varying the tension-elasticity ratio ψ, maximum muscle tension $T_{\max}$, and muscle length L_e.

In general, we find that at $Re = 1$, vorticity generation is minimal even when muscle parameters are varied, consistent with the dominance of viscous effects at low Reynolds numbers. As Re increases, the peak vorticity increases across all parameter variations.

We again examine each muscle parameter individually, beginning with the muscle–tension ratio, ψ, with constant muscle tension, $T_{\max} = 4,500$, shown in Fig. 2.24. Here we observe that a decreased ψ (i.e., a more elastic tentacle) produces notably less vorticity than cases with larger ψ. Increasing ψ produces the greatest vorticity, although only marginally more than the base case. This coincides with the result of our kinematic analysis, where it was found that a greater value of ψ produces an increased tentacle range of motion.

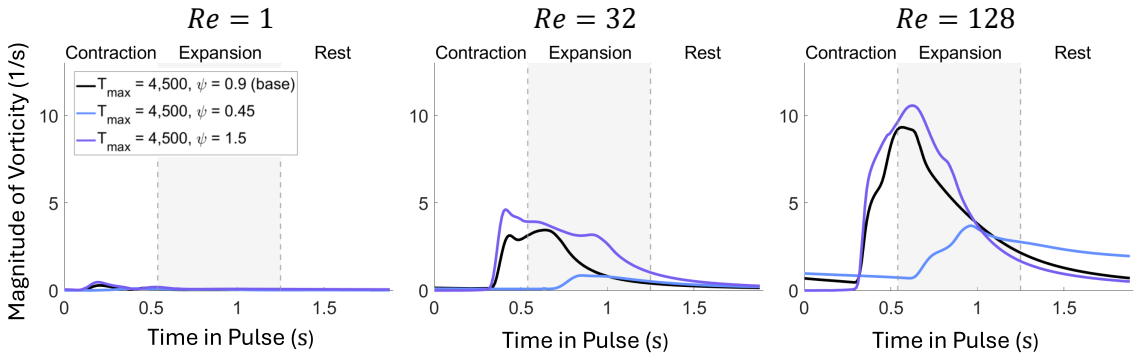


Figure 2.24: Magnitude of vorticity with constant muscle tension, $T_{\max}$ and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

We next consider variations in $T_{\max}$. As $T_{\max}$ increases, greater vortex generation is observed, as demonstrated in Fig. 2.25 and Fig. 2.26. In the case with decreased $T_{\max}$ and decreased ψ, we observe almost no vortex generation across all Reynolds numbers, illustrated by the orange dashed line. Holding $T_{\max}$ low while increasing ψ still produces minimal vortex generation, even at $Re = 128$. For the cases with increased $T_{\max}$, similar peak vorticity magnitudes are achieved, although the value of ψ shifts the timing of the peak. This shift is likely driven by differences in contraction timing: the tentacle with $\psi = 1.35$ (corresponding to the purple line in Fig. 2.25) reaches its maximum contraction at $t = 0.49$ s, whereas the case with $\psi = 0.9$ (corresponding to the red dashed line in Fig. 2.25) reaches maximum contraction at $t = 0.88$ s. Although active contraction has ended in the latter case, the tentacle elasticity and fluid momentum continue to drive an inward motion, shifting the timing of peak vorticity.

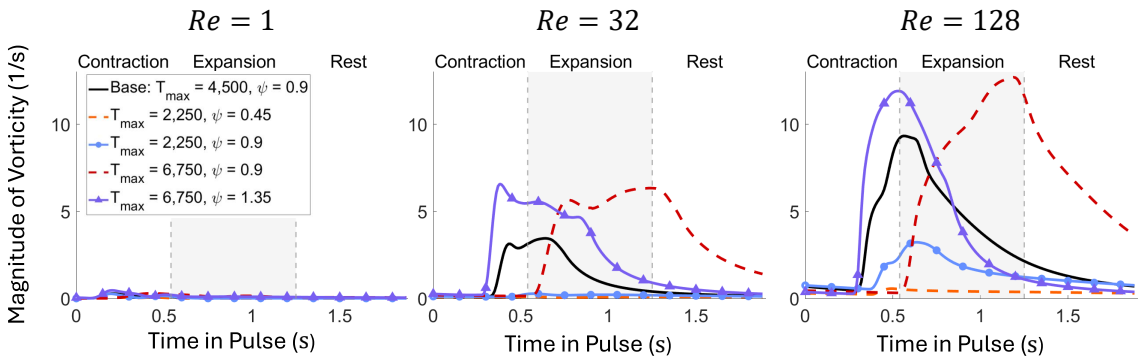


Figure 2.25: Magnitude of vorticity with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

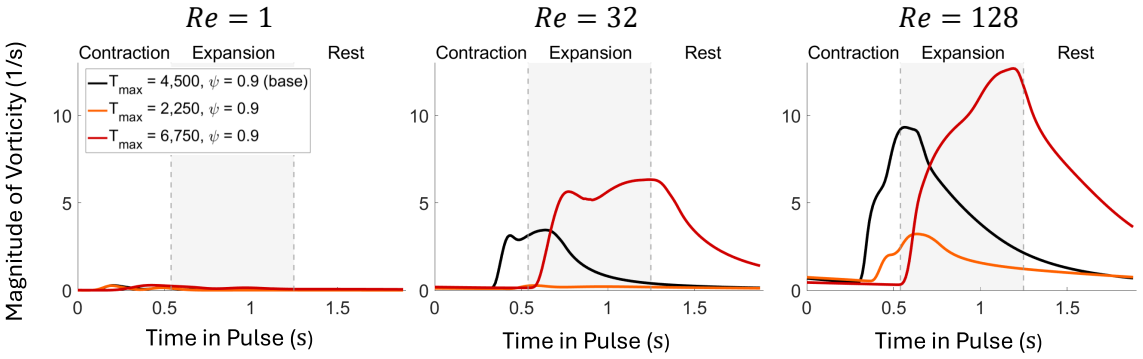


Figure 2.26: Magnitude of vorticity with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Finally, we examine the impact of increasing and decreasing muscle length (Fig. 2.27). Across all Re, there is little vortex generation when L_e is halved from the base value. Additionally, increasing the muscle length also leads to less vortex generation than the base parameter. This suggests that the coral polyp may operate near an optimal muscle length with respect to this metric.

Overall, these results indicate that vorticity, and therefore fluid transport, is strongly enhanced at higher Re and with increased muscle parameters, namely increased $T_{\max}$ and ψ, whereas low-Re conditions suppress vortex formation regardless of parameter variation. Additionally, the timing of fluid transport can be modified by adjusting the tentacle's elasticity.

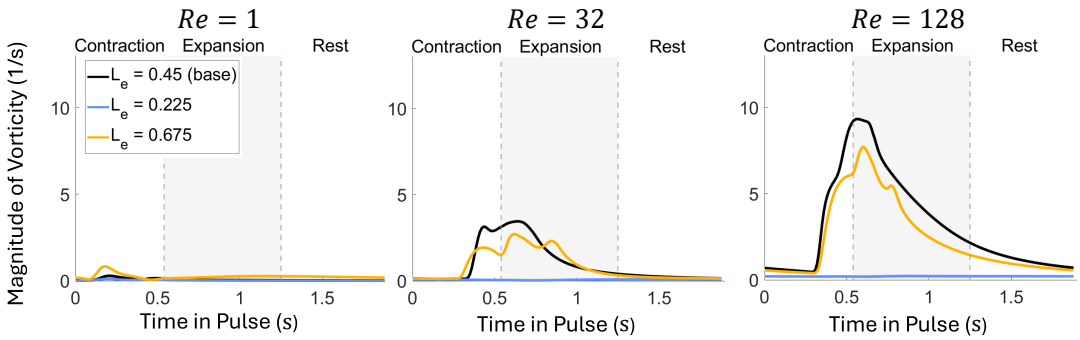


Figure 2.27: Magnitude of vorticity with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Volumetric Flux

We next examine vertical flow under the muscle parameter variations. Results for region V_A are shown in Figs. 2.30–2.31, while the corresponding results for region V_B are shown in Figs. 2.32–2.35. The figures illustrate the effects of varying the tension-elasticity ratio ψ, maximum muscle tension $T_{\max}$, and muscle length L_e (see Fig. 2.7 for region definitions). Results for horizontal flow are omitted as they were found to be negligible.

Closest to the coral polyp, in region V_A, the trend observed for varying Re persists: the maximum vertical flow during a pulse increases with Reynolds number. At the lowest Re ($Re = 1$), fluid motion remains minimal regardless of muscle parameter changes.

Consistent with the vorticity results, increasing $T_{\max}$ leads to increased vertical flow, as shown in Fig. 2.28 and Fig. 2.29. The greatest vertical flow occurs when both $T_{\max}$ and ψ are increased. The influence of ψ alone appears weaker than in the vorticity results; minor shifts in the peak do occur, but it is not clear that these are driven primarily by ψ. For example, this shift is not observed when $T_{\max}$ is held constant (Fig. 2.30), but does occur when ψ is constant and $T_{\max}$ is varied (Fig. 2.29).

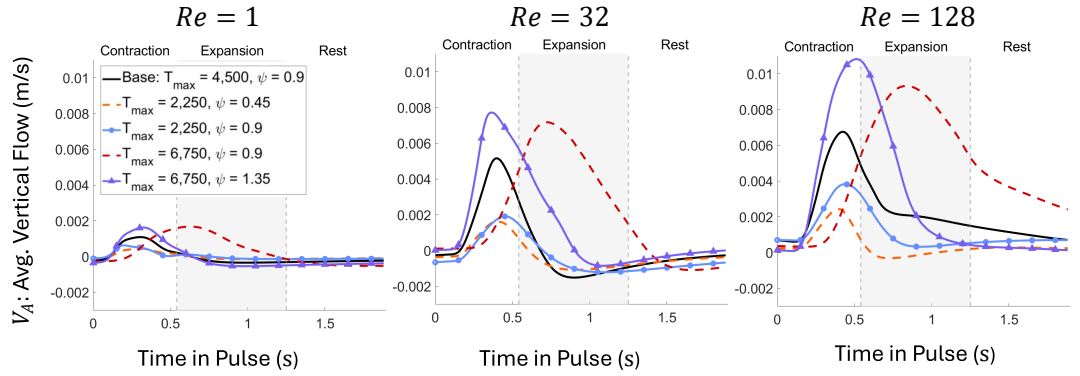


Figure 2.28: Averaged vertical flow in V_A with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

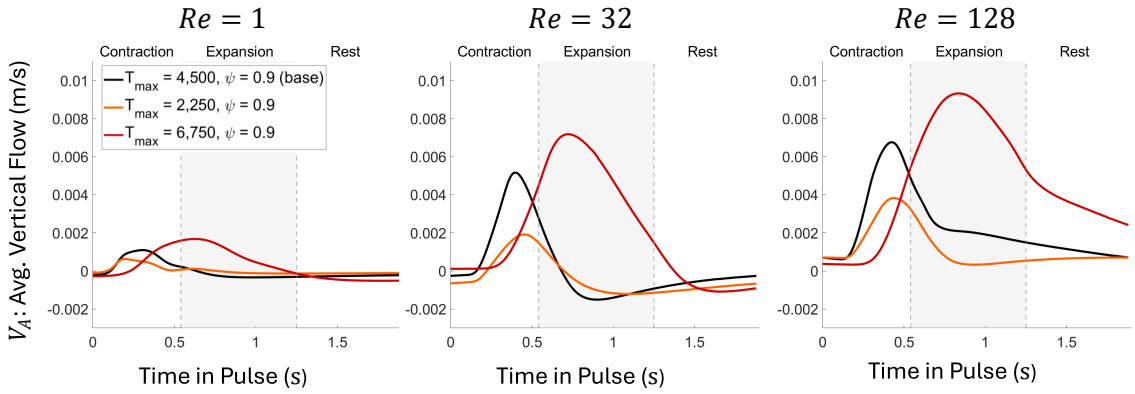


Figure 2.29: Averaged vertical flow in V_A with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

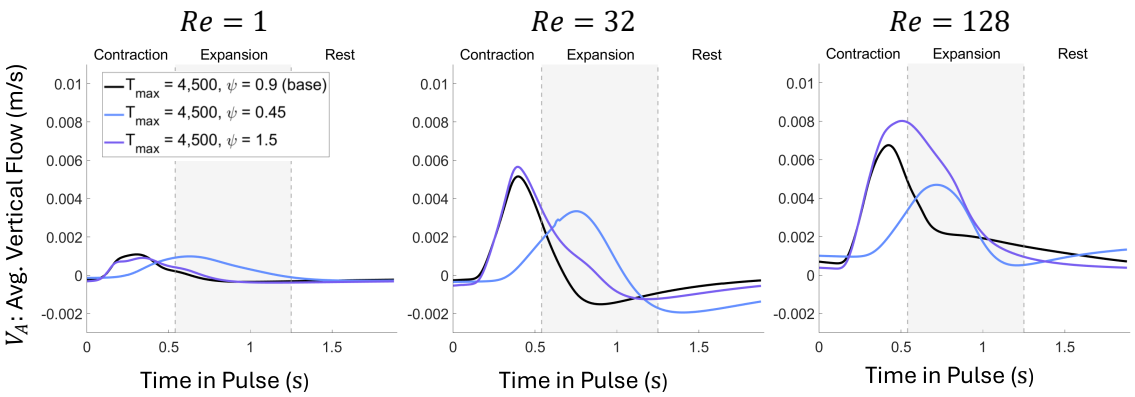


Figure 2.30: Averaged vertical flow in V_A with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Shifting focus to the muscle length, shown in Fig. 2.31, we find that the greatest vertical flow coincides with the base value, consistent with the vorticity results.

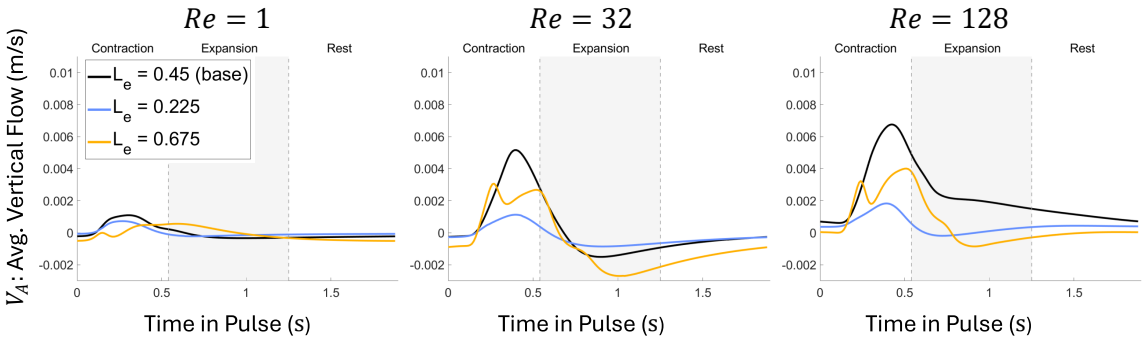


Figure 2.31: Averaged vertical flow in V_A with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

For region V_B, we assess whether parameter variations drive fluid motion beyond the corals immediate neighborhood. The trend observed in the Re-only variations again persists: appreciable vertical flow outside the local region occurs only at high Re. In the low-to-medium Re regimes, variations in muscle parameters do not substantially alter the surrounding flow field. At high Re, increasing ψ (Figs. 2.32-2.33) produces a modest increase in fluid transport away from the organism, with the strongest effect occurring when both maximum muscle tension and the tension-elasticity ratio are increased, as shown by the solid purple curve in Fig. 2.33. This suggests that ψ plays a primary role in modulating transport in this outer region, as variations in $T_{\max}$ alone (with ψ held constant; Fig. 2.34) do not produce a significant change in the flow.

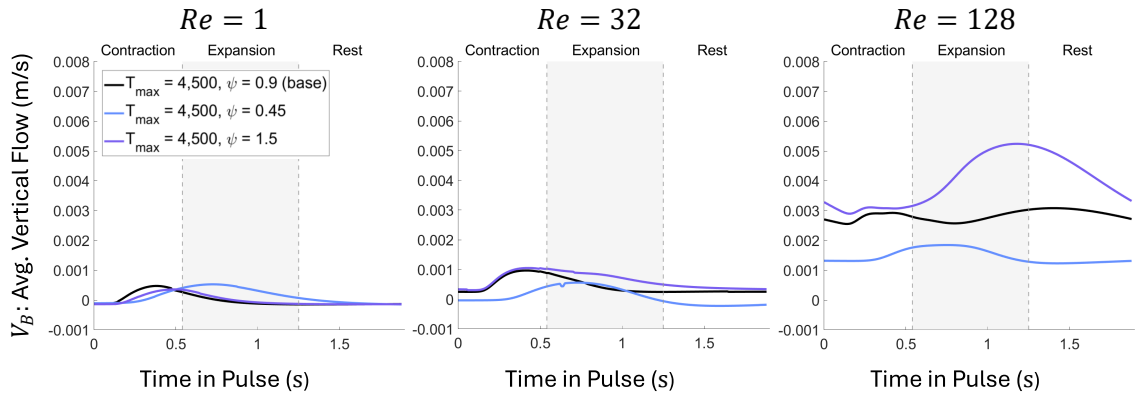


Figure 2.32: Averaged vertical flow in V_B with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

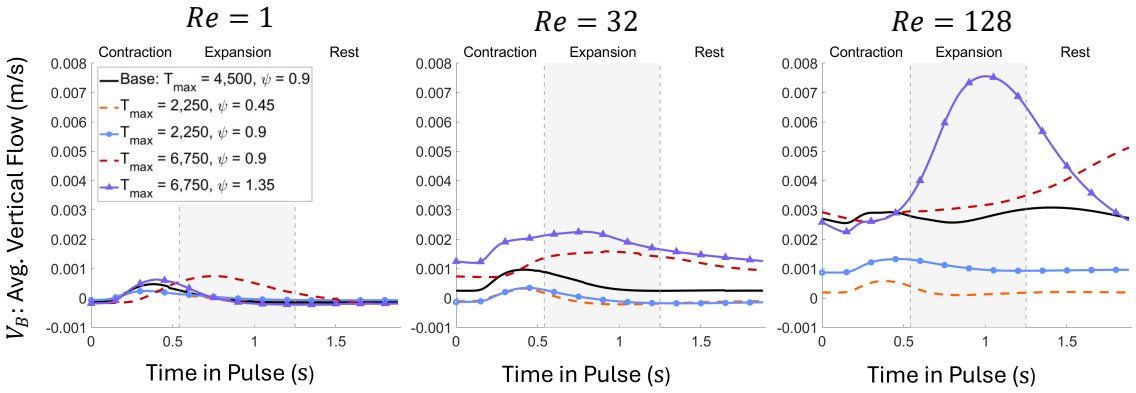


Figure 2.33: Averaged vertical flow in V_B with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

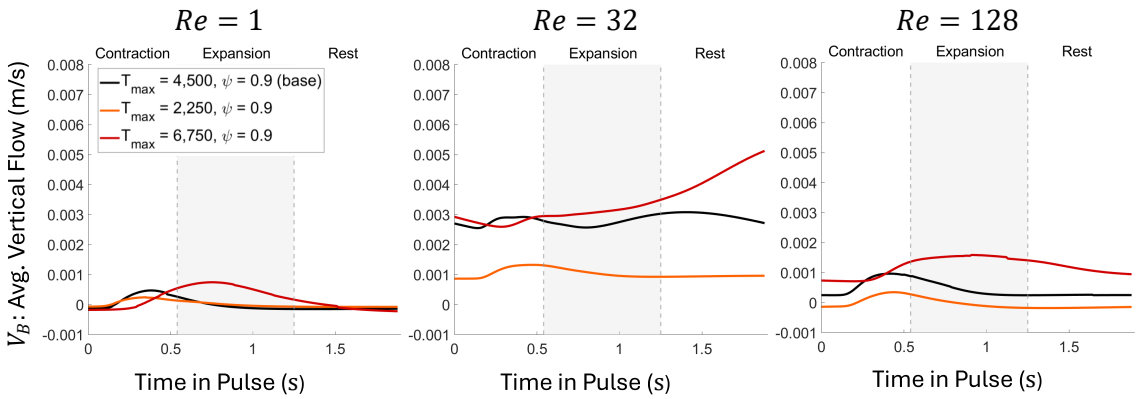


Figure 2.34: Averaged vertical flow in V_B with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Turning to muscle length (Fig. 2.35), the greatest vertical transport continues to occur at the base parameter value, indicating that the base muscle length is well tuned for driving fluid motion in this regime.

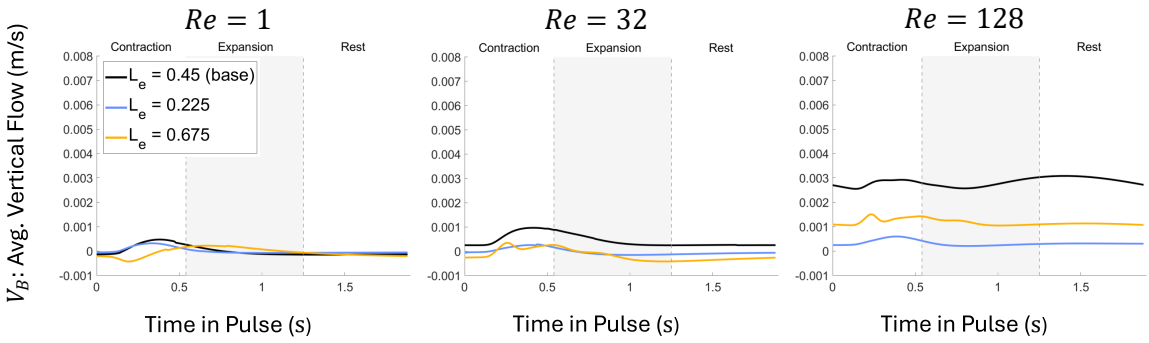


Figure 2.35: Averaged vertical flow in V_B with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$.

Overall, these results suggest that within the parameter ranges studied, biomimetic mixers inspired by this morphology would primarily function as local mixers, with limited ability to influence flow at a distance, even under parameter tuning.

Energy and Power Expenditure

Finally, we examine the cost metrics for our varied muscle parameters. The tension-based energy and power are shown in Fig. 2.36 and Fig. 2.37, respectively, while the force-based energy and power are shown in Fig. 2.38 and Fig. 2.39.

We focus first on the tension-based metrics. The energy expenditure (Fig. 2.36) begins relatively low, increases to a peak around $Re = 16$, and then dips to a plateau. Power (Fig. 2.37) starts low and increases to a plateau as Re increases. In both metrics, the greatest energy and power are achieved when the muscle length is increased (yellow line), while the least occurs when the muscle length is decreased. Parameters that increase vorticity and flow also increase energetic cost: higher $T_{\max}$ raises energy expenditure, and a smaller ψ (less elastic tentacle) also leads to higher cost. These trends highlight the role of muscle kinematics and the tension-elasticity balance in governing the energetic demands of polyp pulsing.

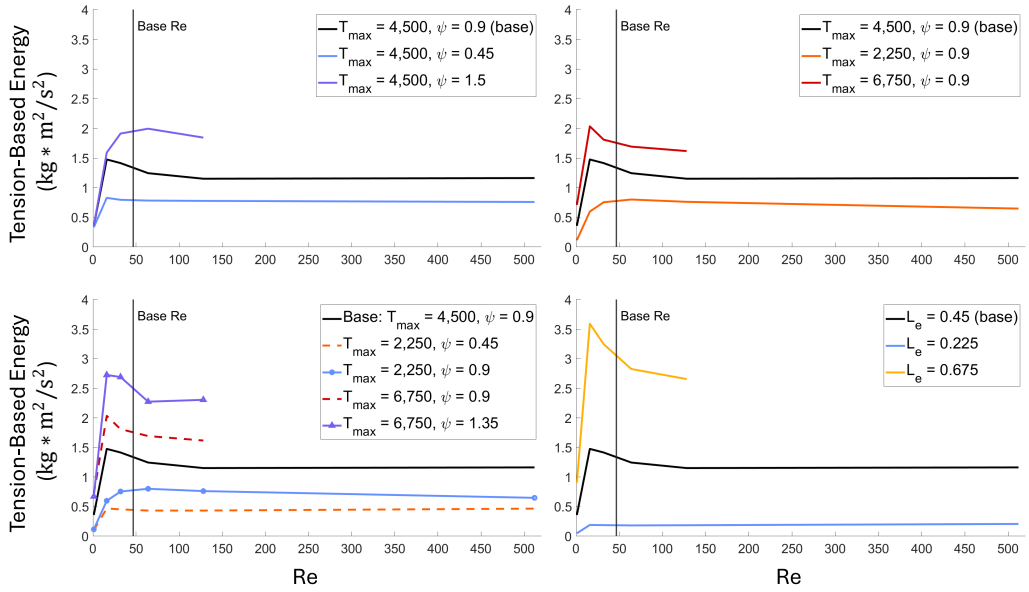


Figure 2.36: Averaged tension-based energy across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e.

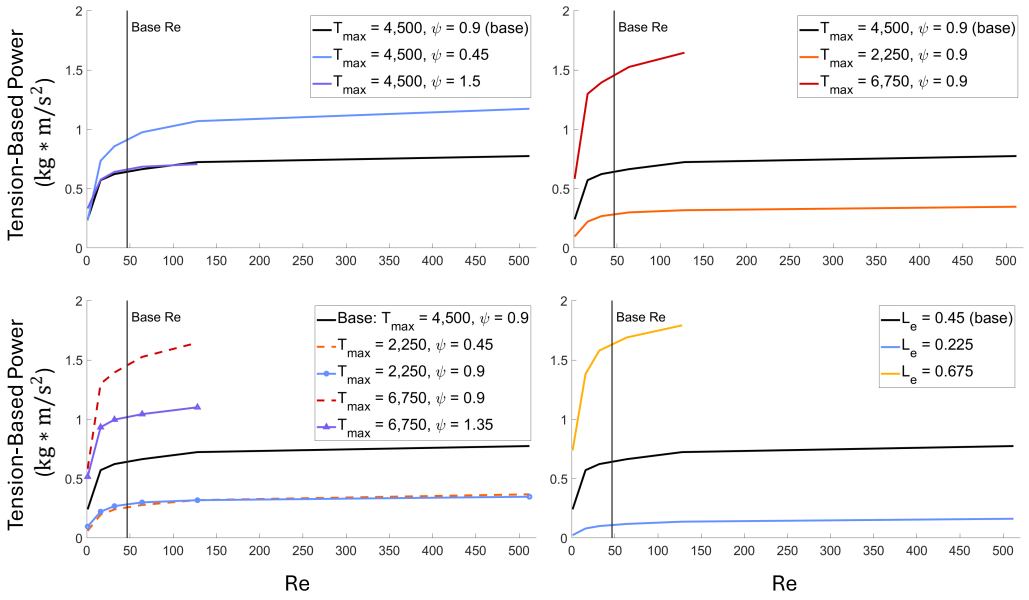


Figure 2.37: Averaged tension-based power across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e.

Next we examine the force-based metrics, presented in Fig. 2.38 and Fig. 2.39 for the energy and power respectively. The force-based energy and power exhibit similar overall trends, with modest increases from $Re = 1$ to $Re = 16$, followed by plateaus or slight decreases. As with tension-based metrics, increased muscle length yields the highest expenditures, while decreased muscle length yields the lowest. However, variations in ψ play a smaller role, and the effects of $T_{\max}$ dominate. Overall, these results indicate clear patterns for tension-based metrics and more subtle trends for force-based metrics, informing our choice of metrics for subsequent surrogate modeling.

These observations suggest that the force-based metrics are less smooth and may be harder to capture accurately with our surrogate models in the next phase of work, whereas tension-based metrics are more predictable. In both cases, reducing muscle length or tension lowers energetic cost, but only the tension-based metrics reflect the influence of tentacle elasticity. Across all metrics, the biological base parameters result in intermediate energy expenditures, suggesting that corals may be naturally tuned to balance energetic cost and fluid transport efficiency. Consequently, tension-based metrics are prioritized for further modeling due to their smoother behavior and ease of analysis.

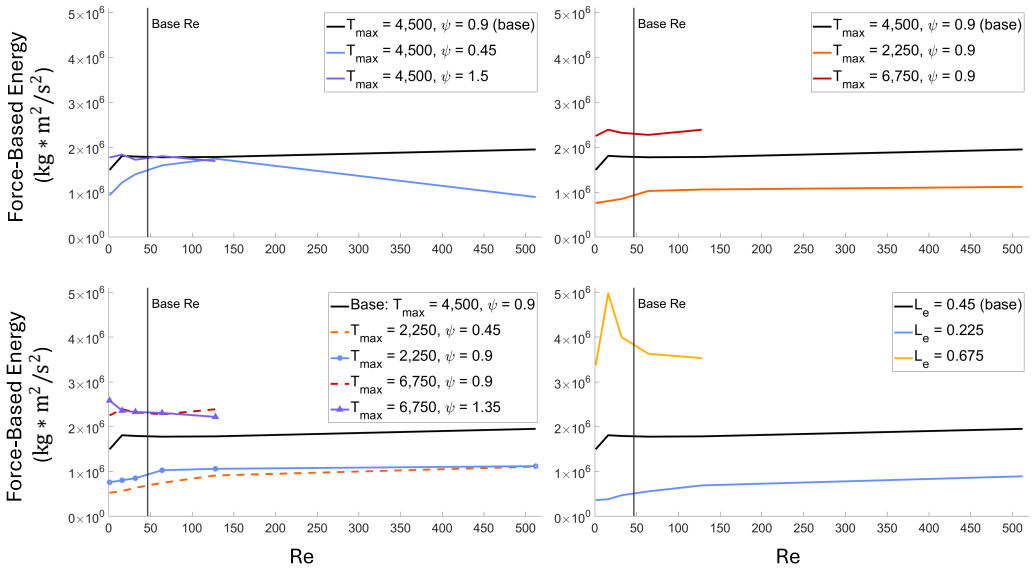


Figure 2.38: Averaged force-based energy across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e.

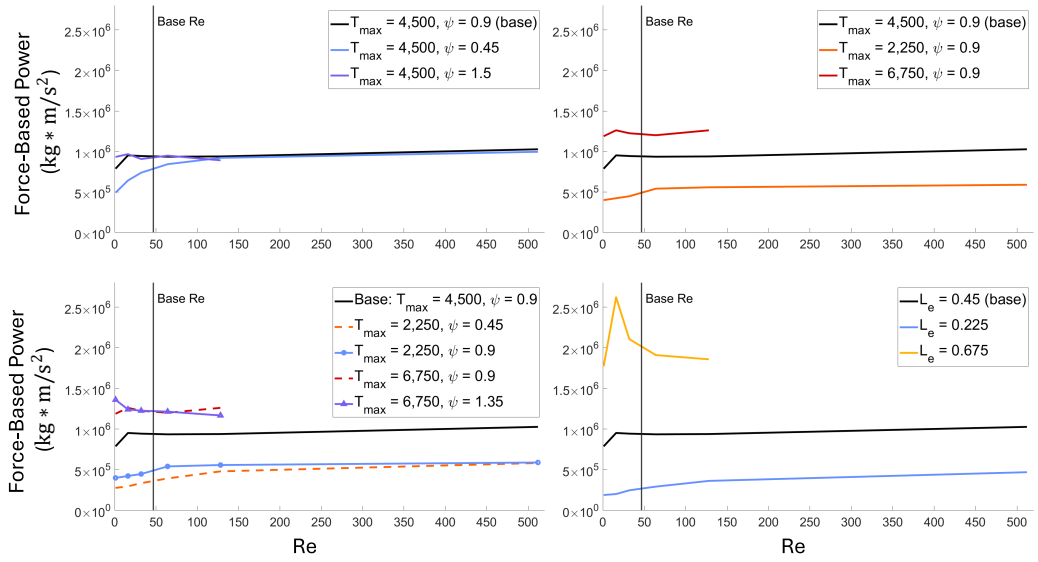


Figure 2.39: Averaged force-based power across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e.

2.7 Conclusion

Overall, varying the muscle and fluid parameters that define a pulsing coral simulation can substantially affect the resulting kinematics, fluid flow, and energetic cost to the coral. In general, increasing these parameters produces greater kinematic motion and stronger fluid flow. However we find that beyond a certain Re threshold, the tentacle motion stagnates while the fluid transport continues to increase as inertial effects begin to dominate. Likewise, increasing these parameters generally leads to an increase in the energy expenditure. This implies that the biological corals may have evolved to pulse in a way which expends enough energy to drive the necessary fluid flow, without expending excess energy for unnecessary fluid transport. Overall, these results highlight that small mechanical changes can lead to vastly different hydrodynamic responses.

Although we examined a broad range of parameter combinations, it is infeasible to test all possibilities to identify those that maximize fluid flow. The simulations used in this study took approximately 181 days of compute time to generate. Additionally, each simulation is on the order of 1 terabyte in size, making data processing and storage an issue. To address this, the next chapter (Ch. 3) employs statistical learning techniques, specifically Gaussian process regression, to model these results. This approach will enable us to uncover the underlying trends within the parameter space without the need to conduct more simulations. From there, we can more efficiently identify additional regions of interest for further study within the parameter space.

Chapter 3

Surrogate Modeling of Fluid Flow Analysis

3.1 Introduction

A downside to conducting numerical studies such as that presented in Ch. 2 is that they can be computationally expensive. The immersed boundary method, computed using the IBAMR library [35, 47], which solves for our fluid velocity and pressure, takes about 38 hours when running on a high-performance computer, and results in files that can be upwards of one terabyte in size. After this, additional time and computing cost are required for the subsequent analyses. This makes it difficult to study the large set of parameters necessary to gain an encompassing understanding of the parameters that drive fluid flow. Therefore, we propose the use of Gaussian processes (GPs), which can learn the underlying flow dynamics from a relatively small set of parameter variations [62, 88]. Gaussian processes are a nonparametric statistical model that use the probability distribution of observed data to make inferences about new data. Intuitively, GPs define a distribution over possible functions that fit the observed data. Instead of outputting a single prediction, GPs provide a mean estimate along a measure of uncertainty, making them especially valuable for probabilistic modeling and decision-making. This tool has been widely used in geostatistics, where it is referred to as kriging [69]. However, GPs have since found applications in many fields where the goal is to model or infer the distribution of some quantity of interest over a space of input parameters. With this in mind, we hope to model and interpolate our results from Ch. 2 onto a uniform parameter space, facilitating a deeper understanding of the parameters that drive kinematic motion and fluid flow.

3.2 Methods

3.2.1 Gaussian Process Regression

We follow the standard GP process to construct surrogate models for each output variable independently. Given a dataset with inputs $\mathbf{Q} = [\vec{q_1}, \vec{q_2}, ..., \vec{q_n}]$ (i.e. our muscle

and fluid parameters) and corresponding outputs $\vec{O} = [o_1, o_2, ..., o_n]$ (i.e. the result being modeled), our goal is to recover the relationship

$$o_i = \hat{g}(\vec{q}_i) + \epsilon_i, \tag{3.1}$$

where $\epsilon_i \sim \mathcal{N}(0, \sigma_i^2)$ represents zero-mean Gaussian noise with variance σ_i^2. Here, $\hat{g}(\vec{q})$ is an unknown function describing the underlying system, and the GP framework allows us to infer this function from limited observations. This is done by defining a distribution,

$$g(\vec{q}) \sim \text{GP}\left(\mathbb{E}[g(\vec{q})], k(\vec{q}, \vec{q}')\right), \tag{3.2}$$

where $\mathbb{E}[g(\vec{q})]$ is the mean function and $k(\vec{q}, \vec{q}')$ is the covariance (kernel) function [88]. Intuitively, inputs that are close in the input space should produce similar outputs, and the kernel encodes this similarity quantitatively through the covariance function.

The prior distribution over the training outputs is

$$g(\mathbf{Q}) \sim \mathcal{N}\left(\mathbb{E}[g(\mathbf{Q})], K\right), \quad \text{where } K = k(\mathbf{Q}, \mathbf{Q}). \tag{3.3}$$

Here, K is the symmetric, positive-semidefinite covariance matrix of the GP prior evaluated at the training inputs $\mathbf{Q}$. Including observational noise gives

$$\text{cov}(\vec{O}) = K + \text{diag}(\sigma_1^2, \sigma_2^2, \ldots, \sigma_n^2). \tag{3.4}$$

The joint distribution of observed outputs $\vec{O}$ and test outputs $\vec{O}^*$ at new inputs $\mathbf{Q}^*$ is

$$\begin{bmatrix} \vec{O} \\ \vec{O}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbb{E}[g(\mathbf{Q})] \\ \mathbb{E}[g(\mathbf{Q}^*)] \end{bmatrix}, \begin{bmatrix} \text{cov}(\vec{O}) & K^* \\ (K^*)^\intercal & K^{**} \end{bmatrix} \right), \tag{3.5}$$

where

$$K^* = k(\mathbf{Q}, \mathbf{Q}^*) \quad \text{and} \quad K^{**} = k(\mathbf{Q}^*, \mathbf{Q}^*).$$

This distribution captures dependencies between observed and test outputs, enabling predictions at new points.

The posterior predictive distribution at new inputs combines prior beliefs with observed data:

$$g^* | \mathbf{Q}, \vec{O}, \mathbf{Q}^* \sim \mathcal{N}(\bar{o}^*, \text{cov}(o^*)), \tag{3.6}$$

with

$$\bar{o}^* = \mathbb{E}[g(\mathbf{Q}^*)] + (K^*)^\intercal K_y^{-1}(\vec{O} - \mathbb{E}[g(\mathbf{Q})]), \tag{3.7}$$

$$\text{cov}(o^*) = K^{**} - (K^*)^\intercal K_y^{-1} K^*. \tag{3.8}$$

Here, $\bar{o}^*$ is the predicted mean, and $\text{cov}(o^*)$ quantifies prediction uncertainty [79, 88].

Specifying a meaningful mean function can be difficult in practice. Instead, we include basis functions to account for a nonzero mean:

$$o(\mathbf{Q}) = h(\mathbf{Q})^\intercal \zeta + g(\mathbf{Q}), \tag{3.9}$$

$$g(\mathbf{Q}) \sim \text{GP}\left(0, k(\mathbf{Q}, \mathbf{Q}')\right), \tag{3.10}$$

where $h(\mathbf{Q})$ is a set of fixed basis functions and ζ is a coefficient vector. In this work, we use a constant basis $h(\mathbf{Q}) = 1$, which assumes that the outputs fluctuate around a single central value. The GP then captures the structure of the residuals via the covariance matrix.

Finally, two standard limitations of GP models should be noted. First, conventional GP regression treats each output independently and does not account for correlations between multiple outputs. In applications where several outputs must be modeled simultaneously, multi-output GP regression (co-kriging) can be used; however, this approach is computationally expensive. We therefore adopt the simpler strategy of training a separate GP model for each output, with the option to incorporate a multi-output framework if needed. Second, standard GPs scale cubically with the number of training points, $\mathcal{O}(n^3)$, making them costly for large datasets. To mitigate this computational burden, sparse GP approximations, which rely on a selected subset of the training data, can significantly reduce cost while retaining predictive performance [60, 87].

3.2.2 Kernel Functions

Intuitively, the kernel function, $k(\vec{q}, \vec{q}')$, controls how quickly outputs change as inputs vary. It therefore determines many important properties of the surrogate model, including smoothness and characteristic length scales. Selecting an appropriate kernel is thus crucial for obtaining an accurate and well-behaved model.

We test several standard kernels commonly used in GP regression:

- Exponential: $k(\vec{q}, \vec{q}') = \sigma_o^2 \exp\left(- \frac{||\vec{q}-\vec{q}'||_2}{\ell} \right)$
- Squared Exponential: $k(\vec{q}, \vec{q}') = \sigma_o^2 \exp\left(- \frac{||\vec{q}-\vec{q}'||_2^2}{\ell^2} \right)$
- Rational Quadratic: $k(\vec{q}, \vec{q}') = \sigma_o^2 \left(1 + \frac{||\vec{q}-\vec{q}'||_2^2}{2\alpha_{RQ}\ell^2} \right)^{-\alpha_{RQ}}$
- Matern 3/2: $k(\vec{q}, \vec{q}') = \sigma_o^2 \left(1 + \frac{\sqrt{3}||\vec{q}-\vec{q}'||_2}{\ell} \right) \exp\left(- \frac{\sqrt{3}||\vec{q}-\vec{q}'||_2}{\ell} \right)$
- Matern 5/2: $k(\vec{q}, \vec{q}') = \sigma_o^2 \left(1 + \frac{\sqrt{5}||\vec{q}-\vec{q}'||_2}{\ell} + \frac{5||\vec{q}-\vec{q}'||_2^2}{3\ell^2} \right) \exp\left(- \frac{\sqrt{5}||\vec{q}-\vec{q}'||_2}{\ell} \right)$

Here, σ_o^2 is the output variance, ℓ is the length scale controlling how far correlations extend in input space, and α_{RQ} is a non-negative scale-mixture parameter specific to the Rational Quadratic kernel [88]. To provide intuition for how these kernels affect the GP predictions, an illustration of each of these functions is provided in Fig. 3.1. The Exponential kernel produces rough functions, the Squared Exponential yields very smooth curves, and the Rational Quadratic exhibits intermediate smoothness with multiple length scales. The Matern kernels provide intermediate smoothness between these extremes.

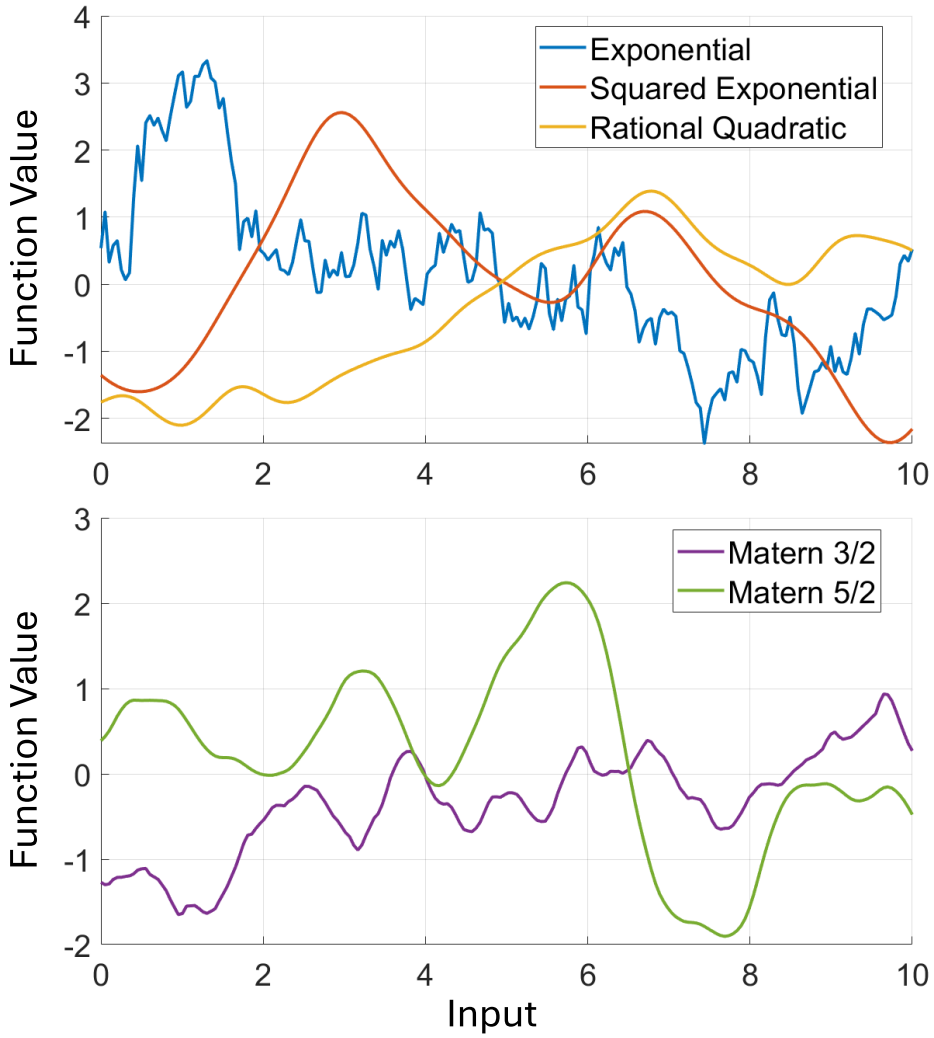


Figure 3.1: Example of kernel function. Top: Exponential, Squared Exponential, and Rational Quadratic; Bottom: Matern 3/2 and 5/2.

The hyperparameters σ_o^2, ℓ, and α_{RQ} are learned from the data during GP training. Note that σ_o^2 has the same units as the square of the output, while α_{RQ} is unitless.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(o_i - \hat{o}_i\right)^2, \tag{3.11}$$

where n is the number of training points and $\hat{o}_i$ is the true value. This MSE provides a straightforward measure of how well the GP captures the underlying function at the observed points, allowing comparison of the relative fit across different kernel choices. Note that the MSE has units equal to the square of the output units.

3.2.3 Implementation

Surrogate GP models are generated in MATLAB using `fitrgp` [109]. When data points are sparse, i.e., separated by distances larger than the kernel length scale ℓ, the GP prior dominates, causing predictions to revert toward the prior mean and variance.

To mitigate this effect, we apply three preprocessing steps:

1. Transform the Reynolds number as $\hat{Re} = \log_2(Re)$ to reduce skew in its distribution;
2. Standardize all input parameters to zero mean and unit variance, ensuring that the kernel treats each input dimension consistently;
3. Normalize outputs such that $\max(y) = 1$ to prevent differences in output scale from dominating the similarity measure.

Hyperparameters, including the noise variance σ and basis function coefficient ζ, are optimized internally by `fitrgp` to minimize cross-validation loss. We evaluate automatic relevance determination (ARD) kernels for the Exponential, Squared Exponential, Matern 3/2, Matern 5/2, and Rational Quadratic kernels [25, 63]. ARD assigns a separate length scale to each input, allowing the GP to automatically learn which parameters are most influential. For each surrogate model, the kernel type yielding the lowest mean-squared error (MSE) on the training data is selected, providing a quantitative measure of predictive accuracy.

For models that include time t as an input, a sparse GP approximation is employed due to the increased data volume [60, 87]. Sparse approximations reduce computational cost by selecting a subset of representative training points, making GP training feasible for large datasets. For datasets of manageable size, exact GP models are used.

3.3 Results

We evaluate the GP surrogates across the full set of simulations outlined in Sec. 2.5, allowing us to assess model accuracy and predictive capability across the entire input parameter space. Prior to this comprehensive study, a preliminary GP surrogate was developed for the vertical velocity with only Re and t varied. These initial results were

published and presented at the IEEE OCEANS 2025 conference and the published paper is included in Appendix A [66].

The results of the present analysis are reported in terms of overall performance, quantified using the MSE, relative error (RE), and visual comparisons between predicted and true values. Relative error,

$$RE_i = \frac{|\hat{o}_i - o_i|}{|\hat{o}_i|}, \tag{3.12}$$

quantifies the magnitude of the prediction error relative to the true output $\hat{o}_i$, providing a normalized measure of accuracy across outputs with different scales. Detailed evaluations for individual cases are provided in the following subsections, while an overview of each models performance appears in Table 3.1, with summaries of the relative error given in Table 3.2.

Result being Modeled	**Training MSE (units^2)**	Kernel Type (ARD)	$\boldsymbol{\sigma_o}$ **(units^2)**	$\boldsymbol{\alpha_{RQ}}$
Kinematic Motion–X	6.53×10^{-5}	Rational Quadratic	0.1497	2.11
Kinematic Motion–Z	5.96×10^{-5}	Rational Quadratic	0.4328	2.73
Magnitude of Vorticity	3.57×10^{-5}	Matern 5/2	0.1728	-
Vertical flow in V_A	4.43×10^{-5}	Rational Quadratic	0.1420	1.69
Vertical flow in V_B	7.09×10^{-6}	Rational Quadratic	0.2444	0.60
Tension-based Energy	7.10×10^{-4}	Rational Quadratic	0.4663	4.46
Tension-based Power	1.65×10^{-4}	Rational Quadratic	0.6063	9.47×10^5
Force-based Energy	4.97×10^{-8}	Rational Quadratic	0.0890	0.25
Force-based Power	0.0017	Rational Quadratic	6.6393×10^{-5}	0.68

Table 3.1: GP surrogate model accuracy across the full parameter space. All models except magnitude of vorticity employ a Rational Quadratic kernel, reflecting the smoothness and regularity of the input data.

All models that depend on time, that is, all models except energy and power, are trained using a subset-of-regressors (SR) sparse approximation, a common approach for large datasets. Although the dataset contains only 60 simulations, each simulation provides data at $dt = 0.01$ from $t = 0$ to $t = 1.89$, quickly producing a large number of points. Training a full GP on every timestep would be computationally expensive and increase the risk of overfitting. The SR approximation mitigates this by using a random 10% of the training points, which can lead to slightly higher relative error at some points [102, 103]. Predictions were made using the fully independent conditional (FIC) method. While SR tends to underestimate predictive variance, especially far from the training subset, FIC adds a diagonal correction to the predictive covariance to account for variance

Metric	Mean Relative Error (% over all points)	Median Relative Error (% over all points)	Max Relative Error (% over all points)
Kinematic Motion X	0.98%	0.28%	16.69%
Kinematic Motion Y	0.66%	0.21%	36.07%
Kinematic Motion Combined	0.69%	0.33%	25.15%
Vorticity	12,347%	5.53%	7.31×10^{7}%
Vertical Flow in V_A	66.17%	2.16%	253,880%
Vertical Flow in V_B	14.88%	0.88%	24,449%
Tension-based Energy	8.04%	6.26%	59.08%
Tension-based Power	5.24%	3.09%	39.13%
Force-based Energy	0.05%	0.02%	0.24%
Force-based Power	7.01%	4.58%	39.55%

Table 3.2: Relative errors of GP surrogate models when evaluated on the training data. Reported values are the mean, median, and maximum of the relative error computed for all time points and simulations. Percentages are relative to the observed output at each point.

not captured by the subset [104]. This correction ensures that uncertainty estimates remain meaningful across the input space.

These modeling choices also influence how different error metrics behave. In particular, the use of a sparse subset and approximate predictive variance can lead to small deviations that appear more pronounced when measured as relative error, especially for outputs with small magnitudes. While MSE captures the overall squared deviation across all points, relative error provides insight into performance relative to the size of the predicted values. Consequently, models were primarily selected based on MSE, with relative error used as a complementary measure to assess performance across all data scales.

3.3.1 Kinematic Comparison

A common limitation of Gaussian Process (GP) models is that they are typically formulated to handle only a single output variable at a time. However, the tentacle kinematics problem inherently has the x and z positions correlated with one another. Although multi-output GP models, also known as co-kriging, have been developed to jointly model correlated outputs such as this, these approaches significantly increase the training cost and model complexity. Because the primary goal of the surrogate is to provide a computationally inexpensive approximation of the full simulation, we instead model the x and z components independently and evaluate whether the full tentacle trajectory can be accurately reconstructed from these separate models.

The resulting models yield MSEs of 6.5340×10^{-5} for x and 5.9634×10^{-5} for z. Since these errors are of comparable magnitude and remain small, this indicates that each

component can be modeled accurately on its own, suggesting that treating the outputs independently does not introduce a significant loss of fidelity.

The reconstructed tentacle path evaluated over a uniform grid of t and Re with base muscle parameters is presented in Fig. 3.2. This also demonstrates how the x and z positions are modeled separately and then combined to generate a single trajectory. Additionally, we recreate the muscle parameter variations from Section 2.6.2, presented in Figs. 3.3–3.6.

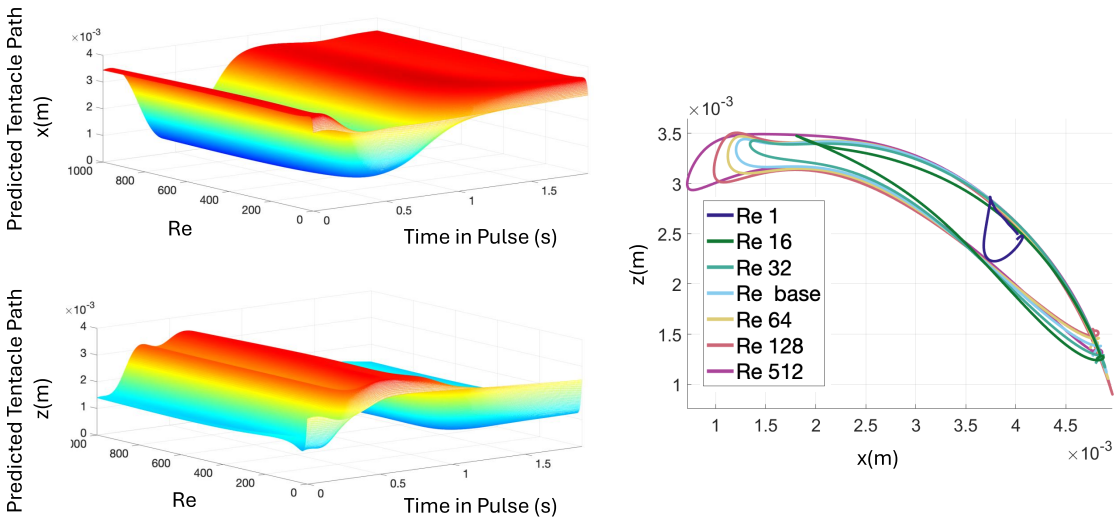


Figure 3.2: Surrogate model for the tentacle path evaluated over a uniform t and Re. The x (top-left) and z (bottom-left) components are modeled independently and combined to reconstruct the tentacle path (right).

In both cases, deviations from the input data largely occur near the beginning and end of the pulsing period (near $t = 0$ and $t = 1.89$). In these regions, the model does not fully close the loop, whereas the true motion begins and ends at approximately the same spatial position. This is most evident in the muscle length variations (Fig. 3.6), where the predicted tentacle-tip trajectories for both increased and decreased muscle length do not end at the same position where they begin. For the increased muscle length case at $Re = 1$, this behavior is partially inherited from a small simulation error discussed in Section 2.6.2. However, this error was not present in other simulations and therefore does not fully explain the observed behavior.

This discrepancy highlights a limitation of surrogate models such as this: while they can capture overall trends in the data, they do not explicitly enforce the physical constraints of the system. Aside from these boundaries, the overall trajectory is well captured, indicating that the independent GP models reproduce the general kinematic behavior accurately.

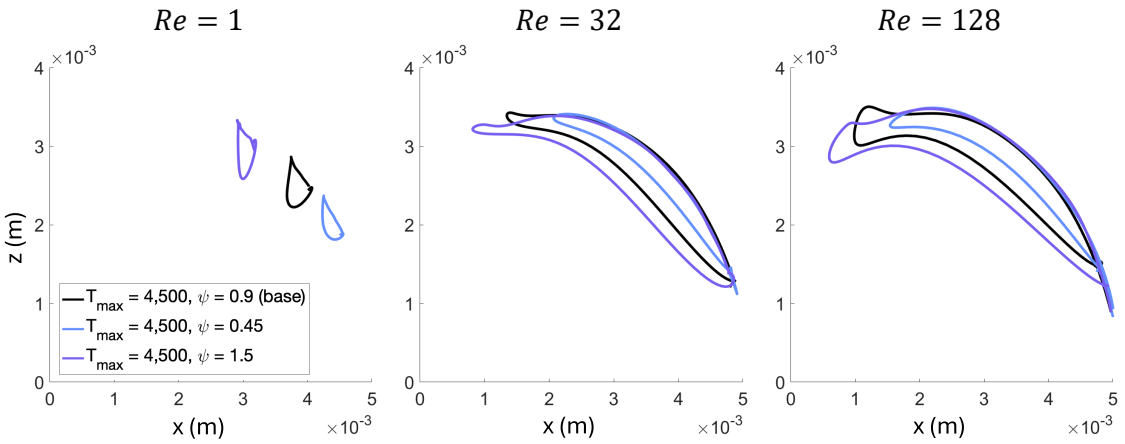


Figure 3.3: Predicted tentacle kinematics with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.19.

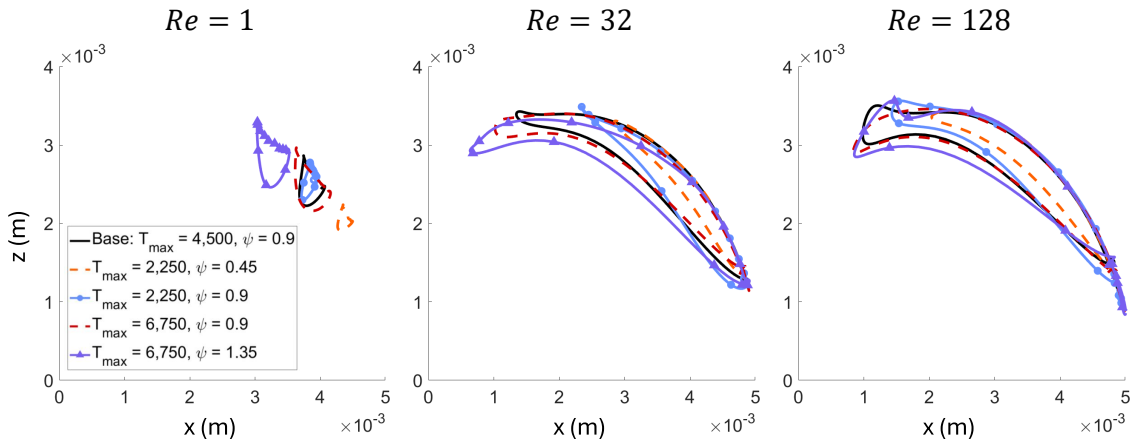


Figure 3.4: Predicted tentacle kinematics with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.20.

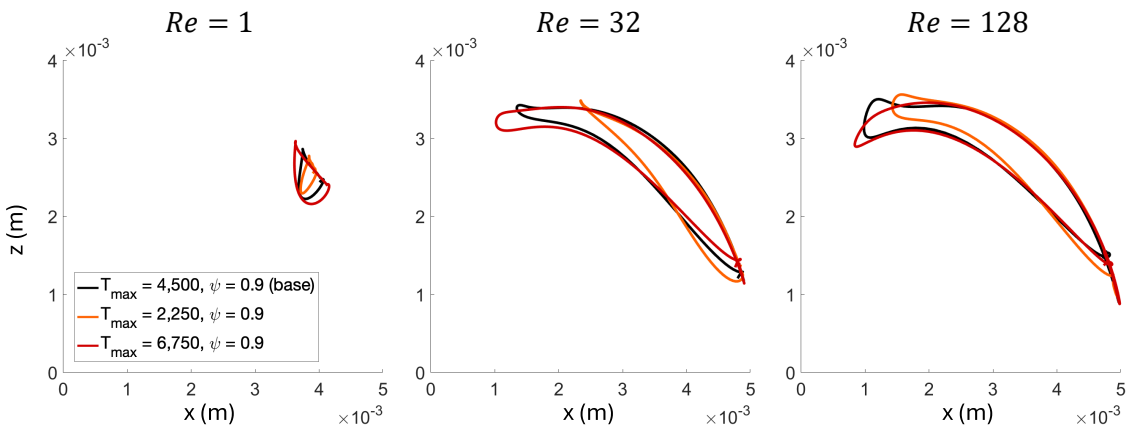


Figure 3.5: Predicted tentacle kinematics with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.21.

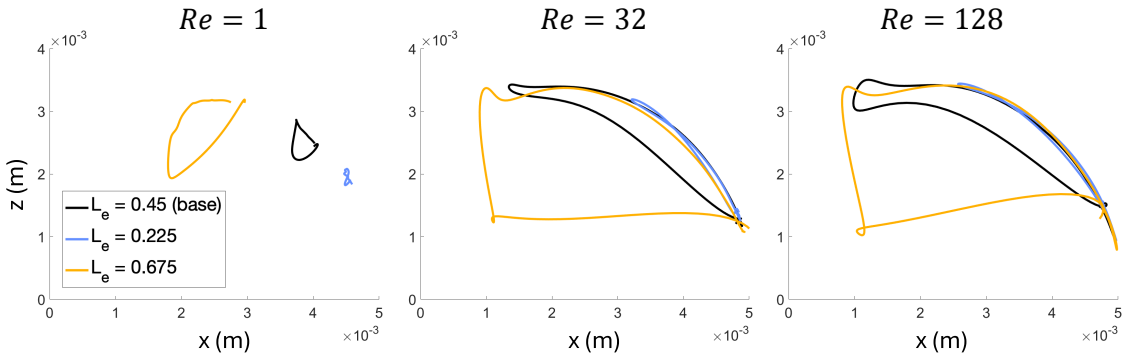


Figure 3.6: Predicted tentacle kinematics with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.22.

3.3.2 Vorticity

The surrogate model for vorticity magnitude achieves an MSE of 3.57×10^{-5}. This results in the accurate reproduction of the data trends as Re and t are varied and other muscle parameters are held constant, presented in Fig. 3.7 Fig. 3.7. We also recreate the muscle parameter variations from Section 2.6.2, seen in Figs. 3.8–3.11. While the predictions remain accurate for the base muscle parameters, some muscle variations produce small negative predictions, even though vorticity magnitude is strictly nonnegative. One such example occurs when $\psi = 1.5$ and $Re = 128$, presented in both Fig. 3.8 and Fig. 3.9. This behavior again reflects a limitation of the surrogate model, which does not explicitly enforce physical constraints, namely that vorticity magnitude cannot be negative. These discrepancies occur near dataset boundaries where the true vorticity is near zero. Since the main interest is identifying parameter combinations producing strong vorticity and mixing, these minor deviations should not be consequential.

Notably, this is the only surrogate model for which the rational quadratic kernel is not optimal; instead, the Matern $5/2$ kernel provides superior performance. This suggests that the vorticity magnitude varies less smoothly with the input parameters than the other outputs and is therefore better captured by a kernel that allows for slightly sharper changes.

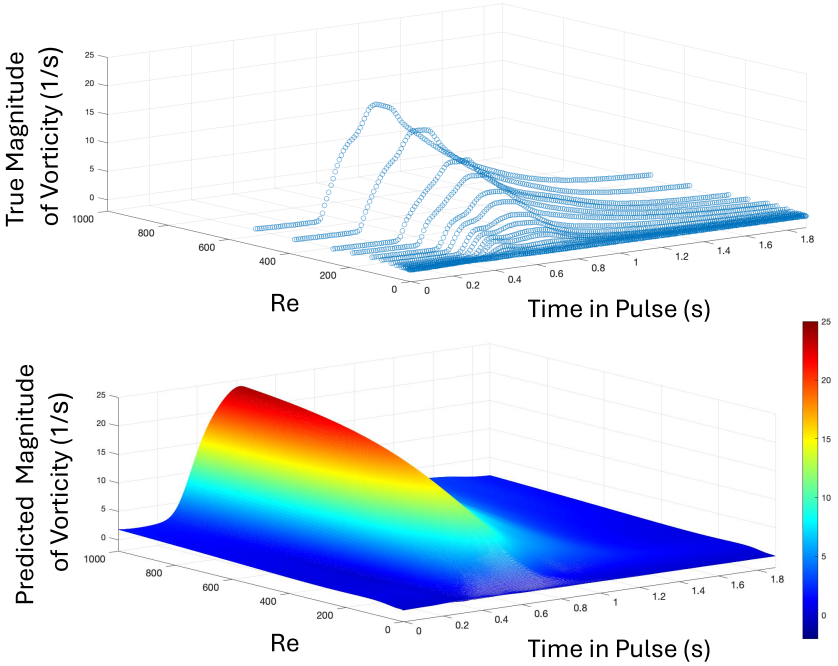


Figure 3.7: True (top) vs predicted (bottom) magnitude of vorticity for region V_A plotted along a uniform t and Re.

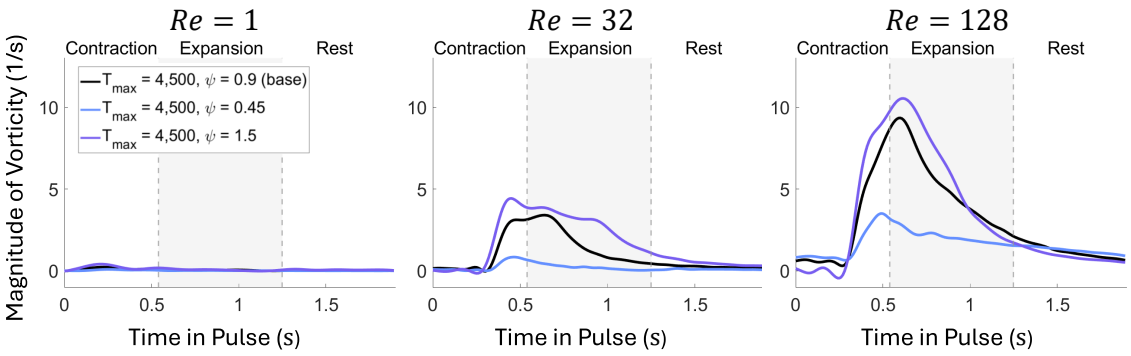


Figure 3.8: Predicted magnitude of vorticity with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.24.

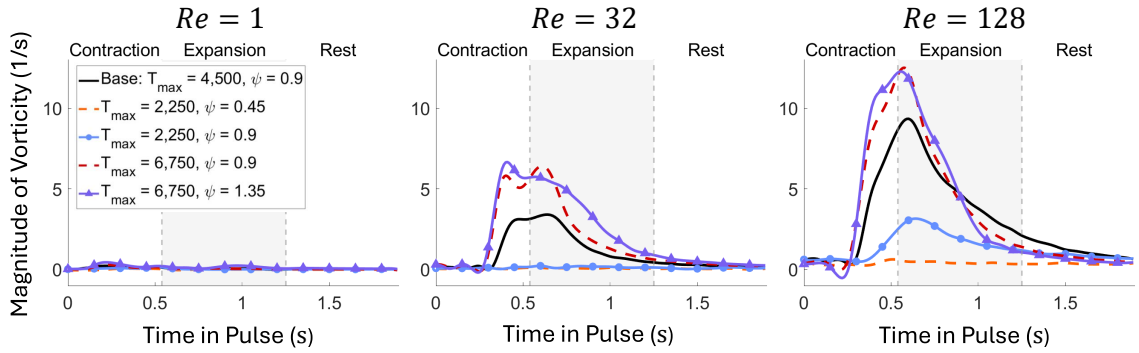


Figure 3.9: Predicted magnitude of vorticity with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.25.

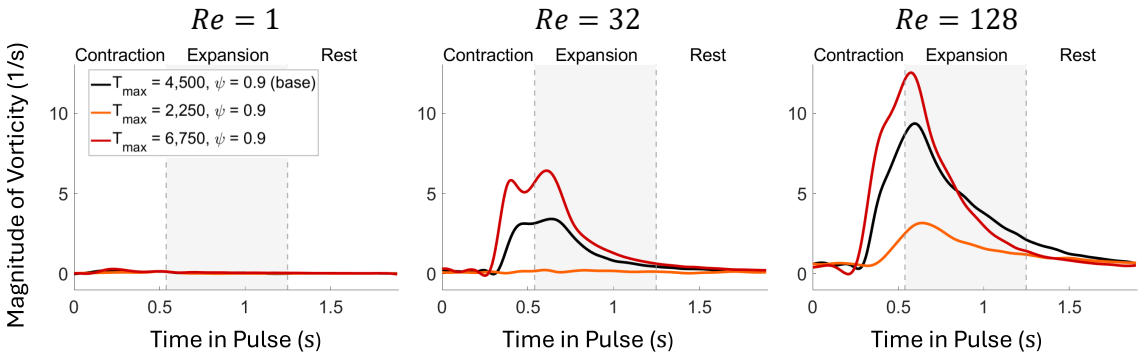


Figure 3.10: Predicted magnitude of vorticity with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.26.

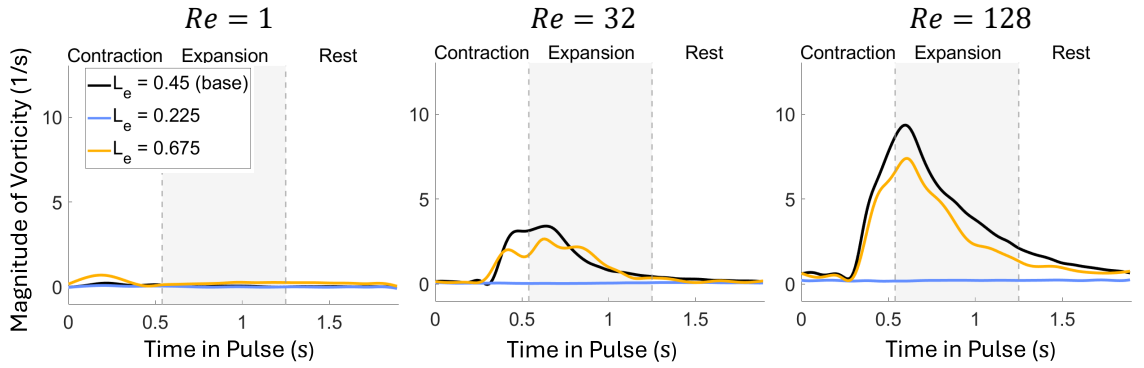


Figure 3.11: Predicted magnitude of vorticity with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.27.

3.3.3 Volumetric Flux

The GP surrogate models for the vertical flow in regions V_A and V_B achieve mean squared errors of 4.43×10^{-5} and 7.09×10^{-6}, respectively, demonstrating strong agreement with the simulation data. Visual comparisons of the true and predicted fields, presented in Fig. 3.12, where Re and t vary while the muscle parameters are held at baseline confirm that the surrogate accurately captures dominant trends and generalizes smoothly to higher Re without introducing artifacts.

Muscle parameter variations from Section 2.6.2 are recreated in Figs. 3.13–3.16 for region V_A and Figs. 3.17–3.20 for region V_B. Across all cases, the surrogate produces smooth, physically consistent fields that visually match the training data.

Unlike the kinematics and vorticity results, no nonphysical behavior is observed in the vertical flow predictions. However, some smoothing of the predicted curves is evident. This effect is most pronounced when the muscle length is increased, seen in Fig. 3.16, Fig. 3.20, and Fig. 3.21, where oscillations present in the simulation data are not fully captured. This behavior is likely driven by the relatively low smoothness of this parameter case compared to others, leading the GP to favor smoother trends. Although modeling all parameter variations is desirable, the practical emphasis is on accurately capturing regimes that generate the most fluid flow. Since increased muscle length consistently produces less vertical flow than the baseline, the associated smoothing and errors are less significant than for parameter combinations that enhance flow.

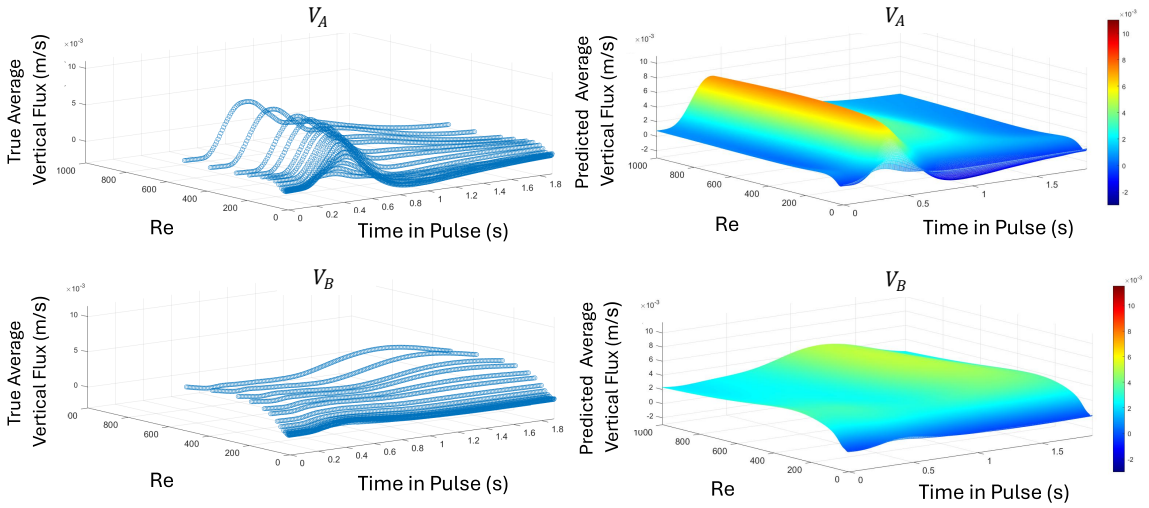


Figure 3.12: True (left) vs predicted (right) vertical flow in regions V_A (top) and V_B (bottom) with Re and t varied while muscle parameters are held at their base values.

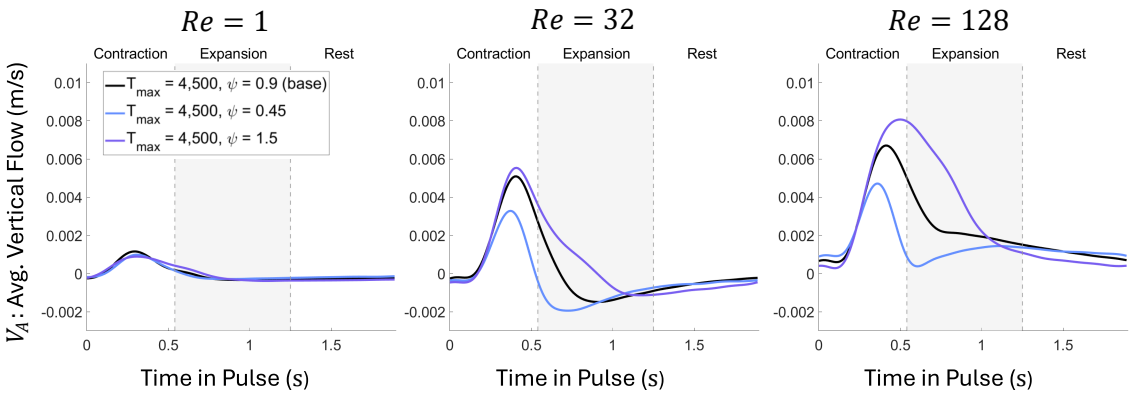


Figure 3.13: Predicted vertical flow in V_A with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.30.

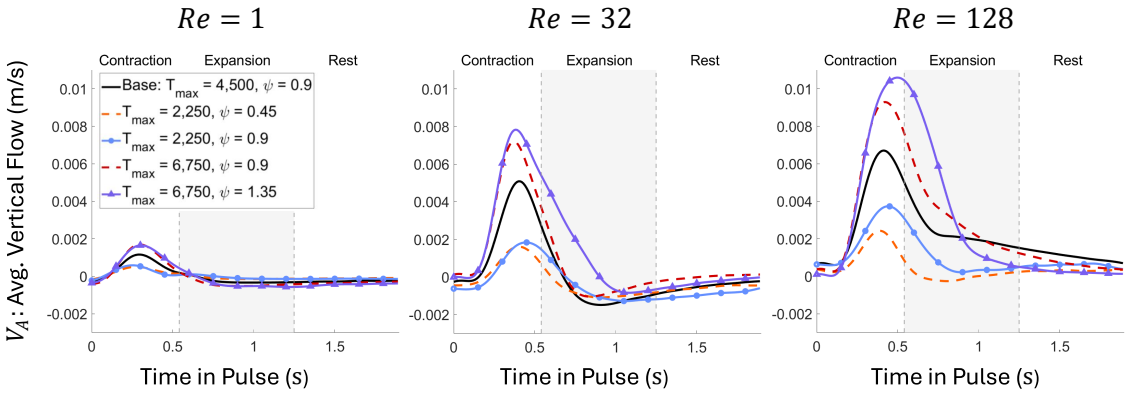


Figure 3.14: Predicted vertical flow in V_A with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.28.

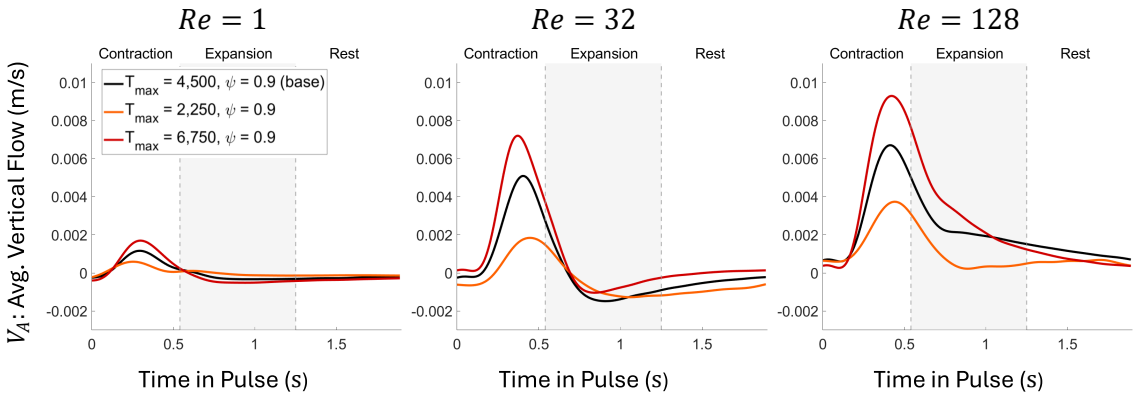


Figure 3.15: Predicted vertical flow in V_A with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.29.

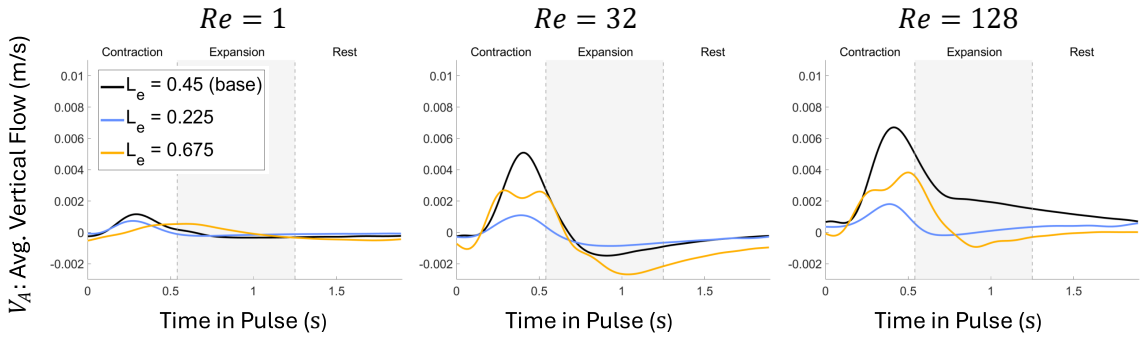


Figure 3.16: Predicted vertical flow in V_A with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.31.

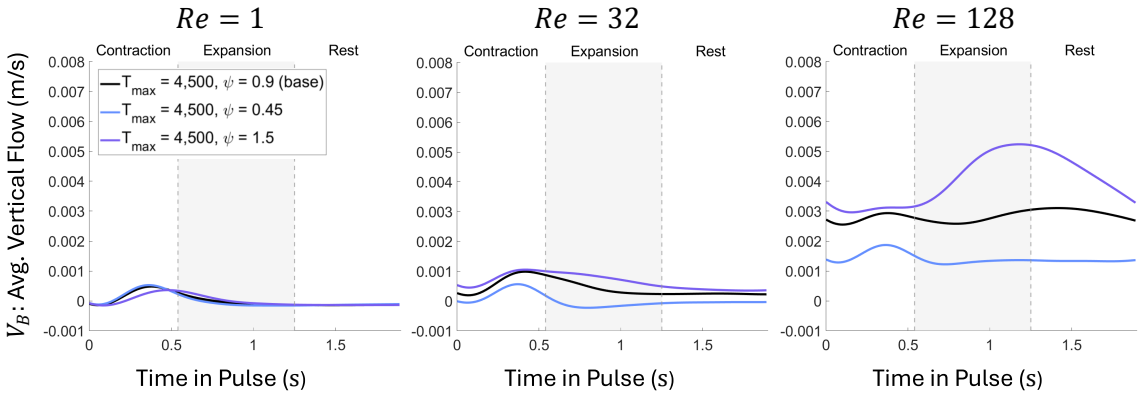


Figure 3.17: Predicted vertical flow in V_B with constant muscle tension, $T_{\max}$, and varied tension-elasticity ratio, ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.32.

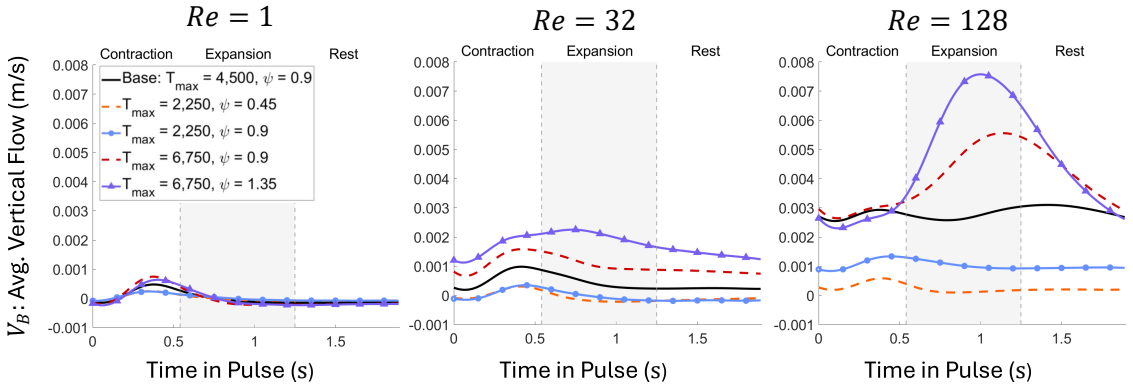


Figure 3.18: Predicted vertical flow in V_B with varied $T_{\max}$ and ψ. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.33.

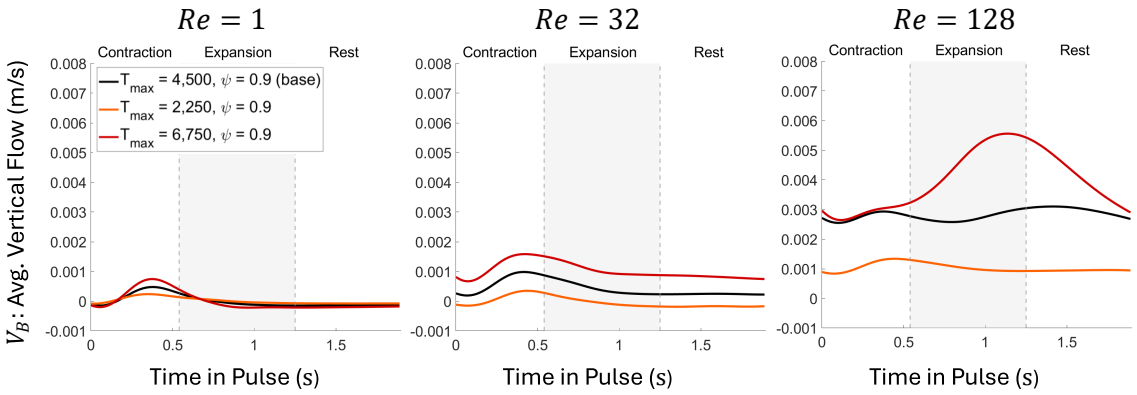


Figure 3.19: Predicted vertical flow in V_B with varied $T_{\max}$ and constant $\psi = 0.9$. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.34.

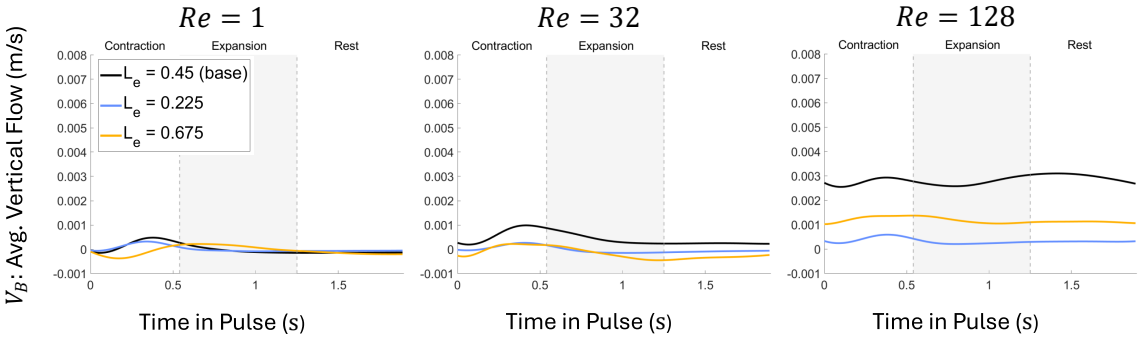


Figure 3.20: Predicted vertical flow in V_B with varied muscle length, L_e. Left to right: $Re = 1$, $Re = 32$, and $Re = 128$. These results correspond to the parameter variations presented in Fig 2.35.

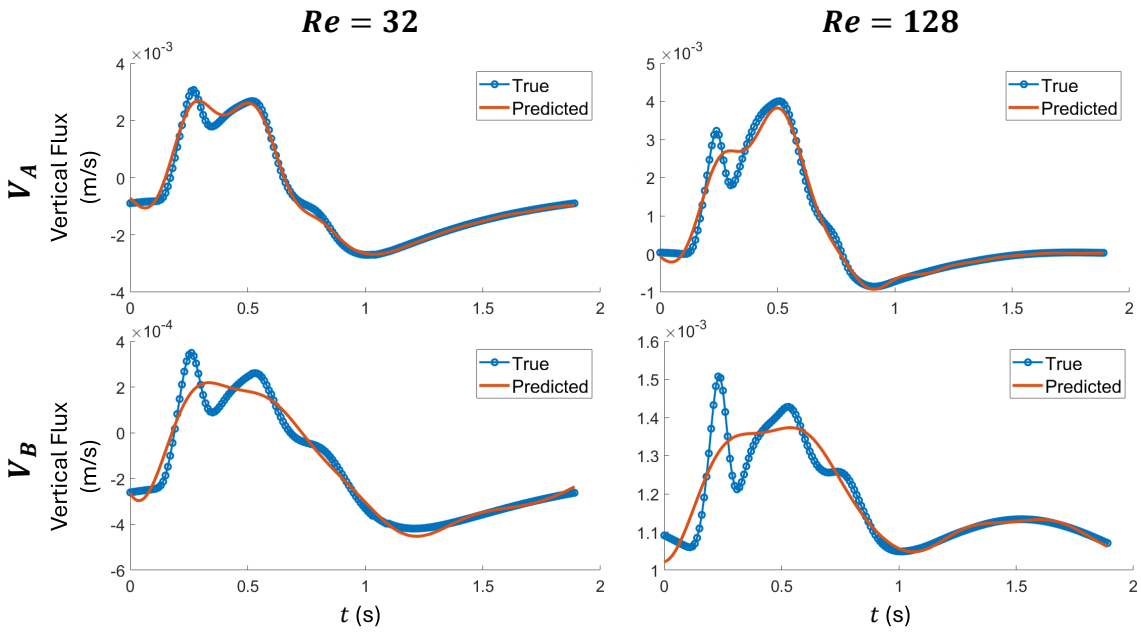


Figure 3.21: True vs predicted vertical flow in regions V_A (top) and V_B (bottom) with $1.5 * L_e$ and other parameters held constant at their base.

3.3.4 Energy and Power Expenditure

The final surrogate models correspond to tension- and force-based energy and power. The tension-based models achieve MSEs of 7.10×10^{-4} for energy and 1.65×10^{-4} for power, while the force-based models attain MSEs of 4.97×10^{-8} and 1.7×10^{-3}, respectively.

Beginning with the tension-based metrics, we compare the true and predicted energy and power as Re varies with muscle parameters held constant (Fig. 3.22). In these cases, the predicted outputs generally follow the simulation data, though small deviations are evident. We also examine surrogate predictions under the muscle parameter variations introduced in Section 2.6.2, shown in Figs. 3.23–3.24. Across these parameter sweeps, the surrogates produce smooth responses that capture the primary minima and maxima of the data.

For the force-based metrics, the energy surrogate performs substantially better than the power surrogate, as reflected by the difference in magnitude of their respective MSEs. The force-based energy results are shown in the left panel of Fig. 3.25 for varied Re and base muscle parameters, and in Fig. 3.26 for the muscle parameter variations from Section 2.6.2. For the varied Re case, the surrogate closely follows the simulation data, despite the data exhibiting greater variability. Additionally, for the parameter variations, the surrogate produces smooth predictions that reproduce the major trends observed in the data.

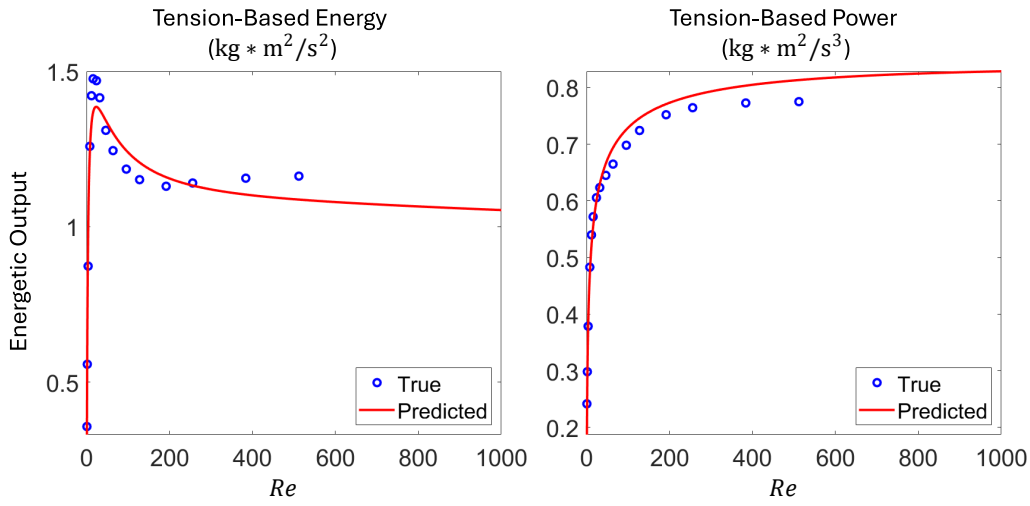


Figure 3.22: True vs predicted tension-based energy (left) and power (right) with Re varied while muscle parameters are held at their base values.

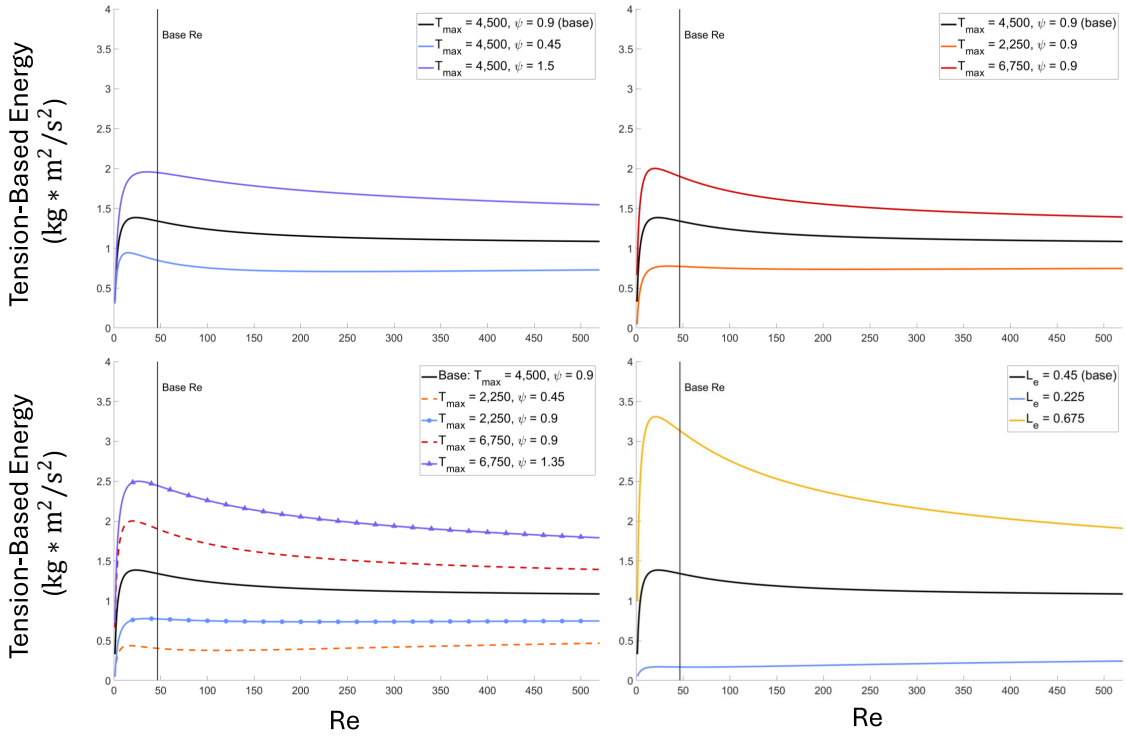


Figure 3.23: Predicted tension-based energy across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e. These results correspond to the parameter variations presented in Fig 2.36.

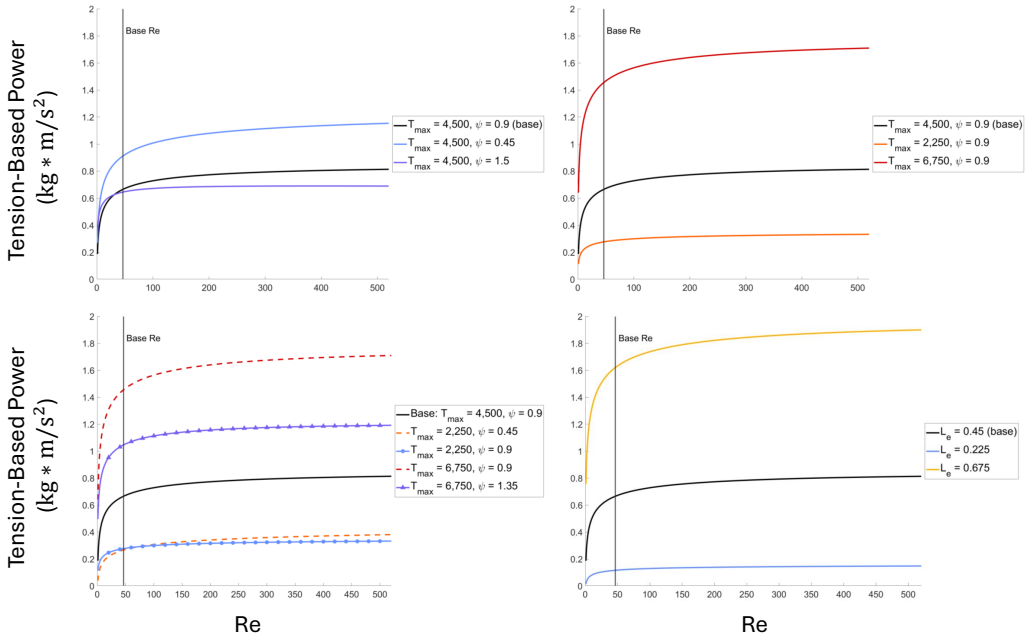


Figure 3.24: Predicted tension-based power across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e. These results correspond to the parameter variations presented in Fig 2.37.

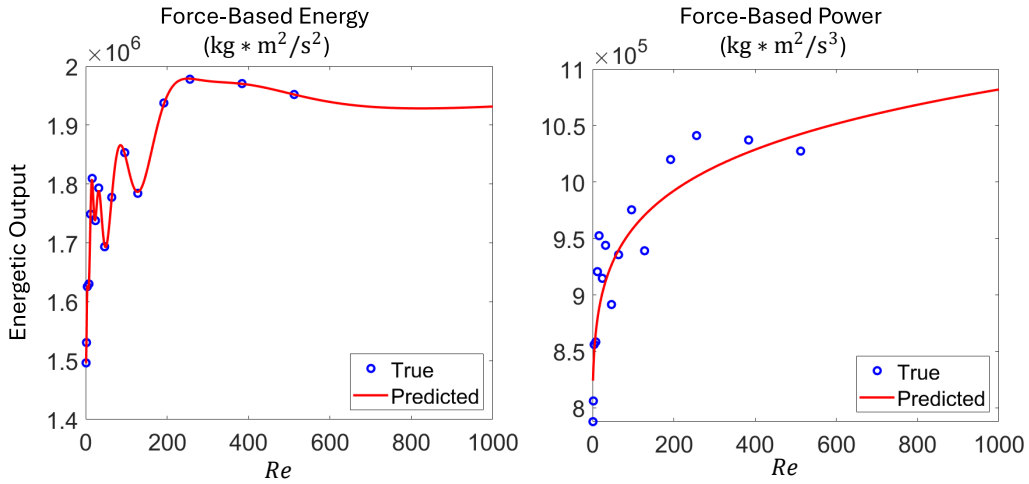


Figure 3.25: True vs predicted force-based energy (left) and power (right) with Re varied while muscle parameters are held at their base values.

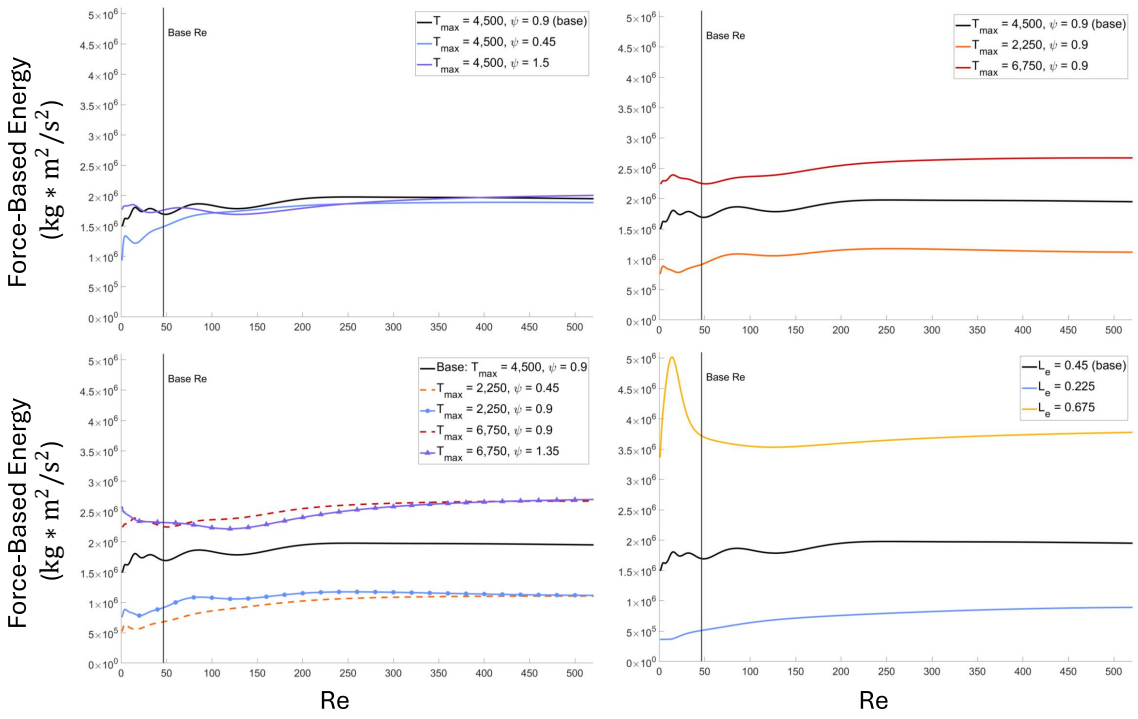


Figure 3.26: Predicted force-based energy across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e. These results correspond to the parameter variations presented in Fig 2.38.

The force-based power surrogate, however, performs less consistently. In the case of varied Re with the base muscle parameters, presented in the right panel of Fig. 3.25, the surrogate produces an overly smooth approximation that falls between neighboring data points rather than reproducing their local variations. When the muscle parameters are varied to recreate the results from Section 2.6.2, presented in Fig. 3.27, a similar effect occurs. This is especially evident for the increased muscle length (yellow curve in Fig. 3.27), which fails to reproduce the large peak present in the simulation data.

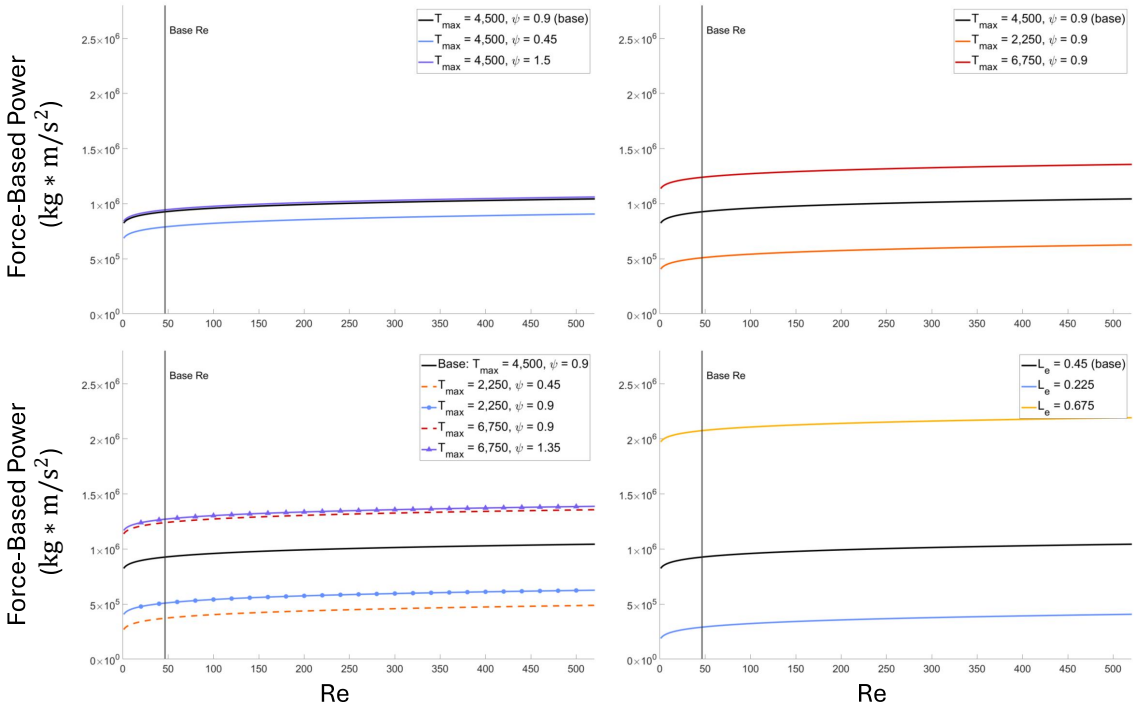


Figure 3.27: Predicted force-based power across muscle parameter variations. The subplots, starting at the top left, show the effect of varying tension-elasticity ratio ψ with constant $T_{\max}$, maximum muscle tension $T_{\max}$ with constant ψ, varied $T_{\max}$ and ψ, and muscle length L_e. These results correspond to the parameter variations presented in Fig 2.39.

From an optimization perspective, however, accurately capturing parameter combinations associated with extreme energy or power costs remains important. Despite the discrepancies observed in the force-based power model, the surrogate models remain useful for identifying regions of parameter space associated with large energetic costs. Notably, despite visual similarities in overall trends, the force-based energy surrogate performs exceptionally well, suggesting that force-based energy varies more smoothly across the parameter space and is therefore more amenable to GP approximation.

Further insight can be gleaned by examining the optimized noise variance σ_o and rational quadratic parameter α_{RQ} (Table 3.1). The tension-based models exhibit large

α_{RQ} values, causing the rational quadratic kernel to behave similarly to a squared exponential kernel with a single characteristic length scale. Combined with a relatively large σ_o, this leads to smoother surrogates that emphasize generalizable trends. In contrast, the force-based surrogates exhibit very small σ_o, indicating that the GP attributes nearly all observed variability to the underlying signal. For the force-based power model, this low noise level, together with the comparatively higher MSE, suggests the presence of sharper or more complex features that are not fully resolved by the available training data.

Finally, although the surrogate models in previous sections were primarily used to interpolate between existing simulation data, they can also be used to estimate system behavior for parameter combinations that were not explicitly simulated. An example occurs when muscle tension is increased while ψ remains at its base value (red curve in the top-right panels of Figs. 3.23–3.24 and Figs. 3.26–3.27). This case was not simulated for $Re = 512$ due to computational constraints. The surrogate therefore provides an estimate of the corresponding energy and power values at this Reynolds number.

3.4 Conclusion

Overall, the GP surrogates demonstrate strong predictive capability across the full parameter space for all evaluated outputs. The models accurately reproduce the key trends of the reference data, with MSEs ranging from 10^{-10} to 10^{-4} depending on the quantity being modeled. Although small discrepancies remain such as slight offsets in the final kinematic position and minor numerical artifacts in the vorticity field, the surrogates successfully capture the dominant extrema and overall trends of the outputs, which are the primary features of interest.

Together, these results demonstrate that the GP framework provides an effective approach for surrogate modeling of both kinematic and flow metrics, capturing the coupled effects of fluid and muscle parameters. The resulting models enable rapid evaluation of system behavior across the parameter space and can subsequently be used for optimization to identify parameters combinations of interest, such as those that minimize energy expenditure or maximize fluid flow, which will be explored in Ch. 4.

Chapter 4

Optimization of Muscle and Parameters

In Chapter 3, Gaussian Process (GP) surrogate models were constructed to approximate several quantities of interest, including kinematic metrics, fluid flow measures, and energetic costs, as functions of the muscle and fluid parameters $\vec{q}$. Each surrogate model, $g(\vec{q})$ represents the GP approximation of the underlying system response. In the following optimization, we treat $g(\vec{q})$ as the objective function to be maximized or minimized with respect to $\vec{q}$.

With these surrogate models, we can efficiently explore the kinematic and fluid flow responses for a range of fluid and muscle parameters beyond those originally sampled. This naturally raises the question: do the biologically observed parameters maximize fluid flow under specific conditions, and if not, which parameter combinations do? More generally, the optimal combinations may depend on the metric being considered, for example, maximizing fluid flow, minimizing energetic cost, or balancing both, and on the surrounding fluid environment. To investigate these possibilities, we optimize the GP surrogate model developed in Ch. 3. After identifying promising parameters, we can run the immersed boundary simulations under these conditions to validate the GP predictions and refine the parameter space in further regions of particular interest.

To identify optimal parameters, we compare several optimization approaches spanning both gradient-free and gradient-based methods. The goal of this comparison is to determine which class of algorithms is best suited for optimizing the GP model. Specifically, we test generalized pattern search, Nelder-Mead optimization, gradient descent, and trust-region optimization. Pattern search and Nelder-Mead are gradient-free methods [77, 111], often used when gradients are unavailable or expensive to compute. While gradients of the GP surrogate could in principle be obtained analytically, the simplicity and interpretability of these methods make them attractive candidates. In contrast, gradient descent and trust-region optimization are gradient-based methods [16, 78], which typically converge more rapidly but may be sensitive to local minima in complex objective landscapes.

In our optimization, the objective functions are the GP surrogate predictions $g(\vec{q})$, which provide smooth approximations of the system responses for the quantities of interest (e.g., kinematic metrics, fluid flow measures, energetic costs). Since an exact analytical gradient of $g(\vec{q})$ is not used here, we approximate gradients and Hessians

numerically using central finite differences. For a scalar parameter q, the first and second derivatives are approximated as

$$g'(q) \approx \frac{g(q+h) - g(q-h)}{2h}, \tag{4.1}$$

$$g''(q) \approx \frac{g(q+h) - 2g(q) + g(q-h)}{h^2}, \tag{4.2}$$

where h is a small step size chosen to match the scale of the variable being optimized. Consequently, our "gradient-based" methods are effectively approximate-gradient-based, though the underlying optimization methodology remains unchanged.

Related optimization studies were conducted on drug-drug interaction and the published works are given in Appendix B [65] and Appendix C [64].

4.1 Methods

4.1.1 Generalized Pattern Search

Pattern search methods evaluate the objective function at multiple points arranged along a lattice or "pattern." Based on these evaluations, the lattice is shifted or reshaped, and the process is iterated until convergence is achieved.

While this class of methods was originally proposed by Hooke-Jeeves [41], a unifying framework that generalizes all pattern search methods and guarantees their convergence are discussed in detail in Torczon [111]. In this framework, sampling the objective function along the pattern is referred to as the *exploratory moves*. While the specific form of these exploratory moves varies by method, two requirements must be met to ensure convergence:

- The step direction s_k must be defined by the pattern PS_k, and its length by a step length parameter Δ_k^{PS}.
- If a decrease in the objective function exists among the trial points, the exploratory moves must include at least one step that achieves a decrease.

These conditions ensure that the exploratory moves produce a simple descent by returning the step that minimizes the objective function value.

Mathematically, the pattern can be defined through a non-singular basis matrix $B \in \mathbb{R}^{n\times n}$ and a generating matrix $C_k \in \mathbb{Z}^{n\times m}$, with $m > 2n$. A common choice is $B = I$, though this may vary depending on the problem. The generating matrix can be expressed as

$$C_k = [M_k \quad -M_k \quad L_k] = [\Gamma_k \quad L_k], \tag{4.3}$$

where $M_k \in M \subset \mathbb{Z}^{n\times n}$, M is a finite set of non-singular matrices, and $L_k \in \mathbb{Z}^{n\times(m-2n)}$ includes at least one column of zeros.

The pattern PS_k is then defined by the columns of

$$PS_k = BC_k = [BM_k \quad -BM_k \quad BL_k] = [B\Gamma_k \quad BL_k]. \tag{4.4}$$

Each trial step s_k^i at iteration k is given by

$$s_k^i = \Delta_k B c_k^i, \tag{4.5}$$

where c_k^i is the ith column of $C_k = [c_k^1, \ldots, c_k^p]$, Bc_k^i defines the step direction, and $\Delta_k^{PS} > 0$ is the step length parameter. This step generates a trial point

$$q_k^i = q_k + s_k^i, \tag{4.6}$$

where q_k is the current iterate. For details, see Torczon [111].

After the exploratory moves, the algorithm performs an *update* step. If any trial point reduces the objective function, i.e.,

$$g(q_k + s_k^i) < g(q_k), \tag{4.7}$$

the corresponding point becomes the new base point for the next iteration. Otherwise, the step size Δ_k^{PS} is reduced by a contraction factor $\alpha \in (0, 1)$, and the exploratory moves are repeated until convergence.

In practice, the initial simplex can significantly influence convergence speed. In the original Hooke-Jeeves paper, a coordinate axis pattern is utilized, as illustrated in Fig. 4.1 [41]. Another common choice is a simplex with $n + 1$ vertices, where n is the problem dimension. For example, a 2D problem uses a triangular simplex, while a 3D problem uses a triangular prism simplex [111]. More complex simplices may cover a larger area in a single exploratory move, but also require additional function evaluations, which can slow convergence.

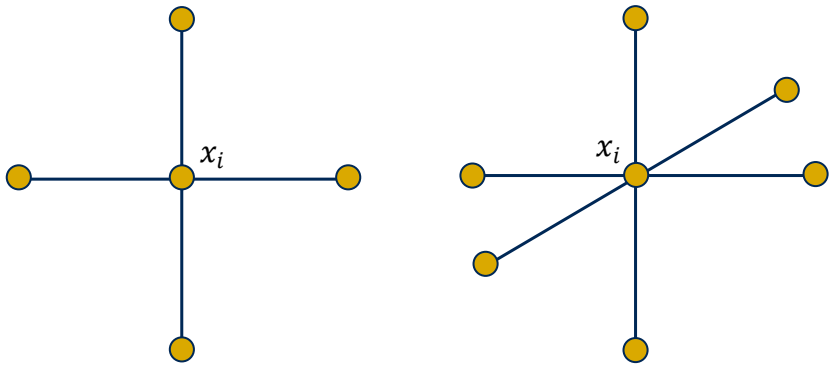


Figure 4.1: Examples of 2D (left) and 3D (right) pattern search simplices.

An outline of this method is provided in Algorithm 4.1.

Algorithm 4.1 Generalized Pattern Search

Require: Objective function $g(\vec{q})$, initial point $q_0 \in \mathbb{R}^n$, initial step size $\Delta_0^{PS} > 0$, contraction factor $0 < \alpha < 1$, tolerance ε, maximum iterations $I_{\max}$, pattern directions $D = \{d_1, \ldots, d_m\} \subset \mathbb{R}^n$

Output: Approximate minimizer q

$q \leftarrow q_0$
Evaluate $g(q)$
$\Delta^{PS} \leftarrow \Delta_0^{PS}$
$i \leftarrow 0$
while stopping criterion not met (e.g., step size $> \varepsilon$) **and** $i < I_{\max}$ **do**
    Construct pattern points:

$$PS = \{q + \Delta^{PS} d \mid d \in D\} \cup \{q\}$$

    Evaluate g at all points in PS
    Let $q_{\text{best}} = \arg\min_{q \in PS} g(q)$ with value g_{best}
    Update center: $q \leftarrow q_{\text{best}}$ ▷ Exploratory move: move to best trial point
    if q_{best} is the original center **then**
        $\Delta^{PS} \leftarrow \alpha \cdot \Delta^{PS}$ ▷ shrink step size
    end if
    $i \leftarrow i + 1$
end while
return q

4.1.2 Nelder-Mead Simplex Method

The Nelder-Mead optimization method is a widely used direct search technique, related to pattern search methods [77]. Although similar in spirit, it does not follow the strict definition of pattern and exploratory moves, placing it in a different class of methods. The algorithm maintains an initial simplex consisting of $n + 1$ vertices for an n-dimensional problem, which is iteratively updated through reflection, expansion, contraction (either inside or outside), or shrink operations.

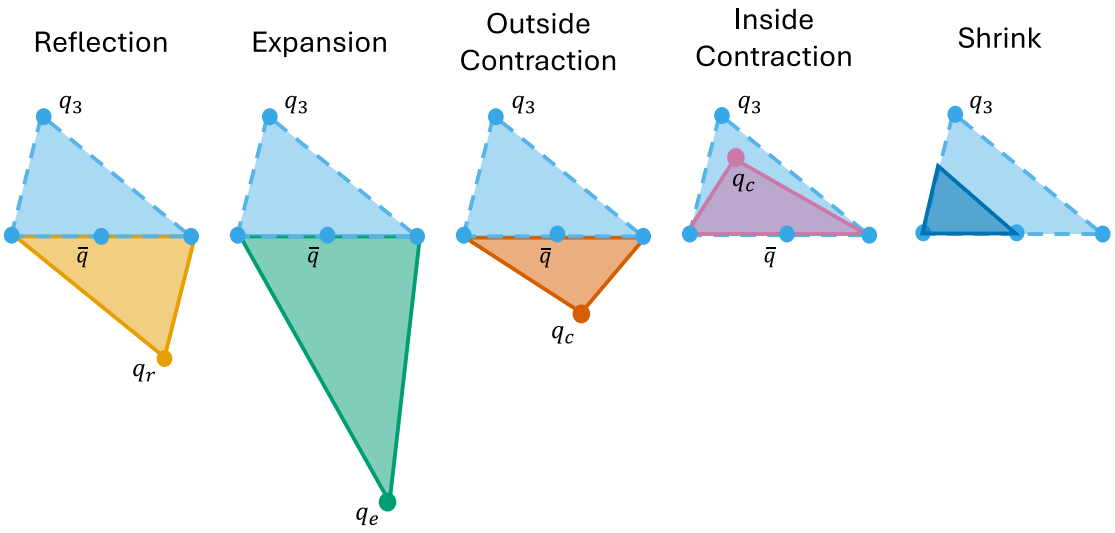


Figure 4.2: Illustration of the simplex transformations in the Nelder-Mead method, where q_3 is the worst point which is being transformed. From left to right: reflection, expansion, outside contraction, inside contraction, and shrink. Blue dotted triangle: original simplex; red solid triangle: updated simplex. Redrawn from Wright [124].

The points are first ordered such that $g(q_1) \leq g(q_2) \leq \cdots \leq g(q_{n+1})$, where $g(q)$ (the GP surrogate) is the objective function being minimized. The centroid of all points except the worst point q_{n+1} is denoted $\bar{q}$.

The reflection point is computed as

$$q_r = \bar{q} + \alpha_{NM}(\bar{q} - q_{n+1}), \quad \alpha_{NM} > 0. \tag{4.8}$$

If $g(q_1) \leq g(q_r) \leq g(q_n)$, then q_{n+1} is replaced by q_r and the process restarts.

If the reflection point improves upon the best point ($g(q_r) \leq g(q_1)$), we attempt an expansion:

$$q_e = \bar{q} + \gamma_{NM}(q_r - \bar{q}), \quad \gamma_{NM} > 1. \tag{4.9}$$

If $g(q_e) < g(q_r)$, q_{n+1} is replaced by q_e, otherwise, by q_r.

For contraction, two cases are considered based on the reflection point. If the reflection point lies between the best and worst vertices, i.e. $g(q_r) < g(q_{n+1})$, an outside contraction

is performed:

$$q_c = \bar{q} + \rho_{NM}(q_r - \bar{q}), \quad 0 < \rho_{NM} \leq 0.5. \tag{4.10}$$

If $g(q_c) < g(q_r)$, then the worst vertex q_{n+1} is replaced by q_c. Otherwise, if $g(q_r) \geq g(q_{n+1})$, an inside contraction is performed:

$$q_c = \bar{q} + \rho_{NM}(q_{n+1} - \bar{q}). \tag{4.11}$$

If $g(q_c) < g(q_{n+1})$, the worst vertex q_{n+1} is replaced by q_c.

If contraction fails to yield improvement, the simplex is shrunk toward the best point:

$$q_i = q_1 + \sigma_{NM}(q_i - q_1). \tag{4.12}$$

Standard parameter values are $\alpha_{NM} = 1$, $\gamma_{NM} = 2$, $\rho_{NM} = 0.5$, and $\sigma_{NM} = 0.5$. An illustration of these moves can be found in Fig. 4.2. An outline of this method can be found in Algorithm 4.2.

4.1.3 Gradient Descent

Gradient descent, also known as the method of steepest descent, is a first-order optimization method that minimizes a function by iteratively moving in the direction of steepest decrease [78]. For an objective function $q(q)$ (our GP surrogate), the gradient $\nabla g(q)$ points in the direction of steepest ascent; thus, the next iterate is

$$q_{k+1} = q_k - \alpha_k^{GD} \nabla g(q_k), \quad \alpha_k^{GD} > 0 \tag{4.13}$$

where q_k is the current iterate and α_k^{GD} is the step size [13, 78].

Intuitively, gradient descent can be thought of as taking small steps directly downhill along the steepest slope, adjusting the step size to ensure each move decreases the objective function without overshooting the minimum.

The ideal step length provides a substantial decrease in the objective function, but computing it exactly can be expensive. Instead we use backtracking line search with the Armijo-Wolfe conditions to adaptively determine the step size α_k^{GD} at each iteration [3, 121, 122]. These conditions ensure sufficient decrease and curvature control:

$$g(q_k - \alpha_k^{GD} \nabla g(q_k)) \leq g(q_k) - c_1^{GD} \alpha_k^{GD} \|\nabla g(q_k)\|^2, \tag{4.14}$$

$$\nabla g(q_k - \alpha_k^{GD} \nabla g(q_k))^T (-\nabla g(q_k)) \geq c_2^{GD} \|\nabla g(q_k)\|^2, \tag{4.15}$$

where $0 < c_1^{GD} < c_2^{GD} < 1$, with standard values being $c_1^{GD} = 10^{-4}, c_2^{GD} = 0.9$ [78]. We iteratively adjust α_k^{GD} until both conditions are satisfied, ensuring sufficient reduction of the objective function along each step.

These steps are outlined in Algorithm 4.3.

Algorithm 4.2 Nelder-Mead Simplex Method

Require: Objective function $g(q)$, initial simplex $\{q_1, \ldots, q_{n+1}\}$, tolerance ε, reflection α_{NM}, expansion γ_{NM}, contraction ρ_{NM}, shrink σ_{NM}, maximum iterations $I_{\max}$

Output: Approximate minimizer q_{best}

$k \leftarrow 0$ ▷ iteration counter
while stopping criterion not met (e.g., simplex size $> \varepsilon$) and $k < I_{\max}$ **do**
    Order the simplex: $g(q_1) \leq g(q_2) \leq \cdots \leq g(q_{n+1})$ ▷ q_1 best, q_{n+1} worst
    Compute centroid excluding worst: $\bar{q} = (1/n) \sum_{i=1}^{n} q_i$
    Reflect worst vertex: $q_r = \bar{q} + \alpha_{NM}(\bar{q} - q_{n+1})$
    if $g(q_1) \leq g(q_r) \leq g(q_n)$ **then**
        Replace $q_{n+1} \leftarrow q_r$
    else if $g(q_r) < g(q_1)$ **then** ▷ Expansion
        $q_e = \bar{q} + \gamma_{NM}(q_r - \bar{q})$
        if $g(q_e) < g(q_r)$ **then**
            Replace $q_{n+1} \leftarrow q_e$
        else
            Replace $q_{n+1} \leftarrow q_r$
        end if
    else ▷ Contraction
        if $g(q_r) < g(q_{n+1})$ **then** ▷ Outside contraction
            $q_c = \bar{q} + \rho_{NM}(q_r - \bar{q})$
        else ▷ Inside contraction
            $q_c = \bar{q} + \rho_{NM}(q_{n+1} - \bar{q})$
        end if
        if $g(q_c) < g(q_{n+1})$ **then**
            Replace $q_{n+1} \leftarrow q_c$
        else
            Shrink all vertices toward best: $q_i \leftarrow q_1 + \sigma_{NM}(q_i - q_1),\ i = 2, \ldots, n+1$
        end if
    end if
    $k \leftarrow k + 1$
end while
return q_1 ▷ Best vertex as approximate minimizer

Algorithm 4.3 Gradient Descent with Armijo–Wolfe Line Search

Require: Initial point q_0, tolerance ε, maximum iterations $I_{\max}$, Armijo parameter $c_1^{GD} \in (0, 1)$, Wolfe parameter $c_2^{GD} \in (c_1^{GD}, 1)$, initial step size $\alpha_0^{GD} > 0$, backtracking factor $\tau \in (0, 1)$

Output: Approximate minimizer q

$k \leftarrow 0$
while Stopping criterion not met (e.g. $\|\nabla g(q_k)\| > \varepsilon$) **and** $k < I_{\max}$ **do**
  Compute descent direction: $d_k = -\nabla g(q_k)$
  Initialize step size: $\alpha \leftarrow \alpha_0^{GD}$
  while Armijo or Wolfe condition not satisfied **do**
    if $g(q_k + \alpha d_k) > g(q_k) + c_1^{GD} \alpha \nabla g(q_k)^T d_k$ **then**
      $\alpha \leftarrow \tau\alpha$ ▷ Reduce step for Armijo condition
    else if $\nabla g(q_k + \alpha d_k)^T d_k < c_2^{GD} \nabla g(q_k)^T d_k$ **then**
      $\alpha \leftarrow \tau\alpha$ ▷ Reduce step for Wolfe curvature condition
    end if
  end while
  Update iterate: $q_{k+1} \leftarrow q_k + \alpha d_k$
  $k \leftarrow k + 1$
end while
return q_k

4.1.4 Trust Region Methods

Trust-region methods define a neighborhood around the current iterate within which a model is "trusted" to approximate the objective function accurately. A step is then chosen to minimize the model within this region. If the step is deemed unacceptable, the trust-region radius is reduced, and a new minimizer is computed [16, 78]. An example of this process is illustrated in Fig. 4.3.

Within the trust region, we assume that a quadratic model, m_k^{TR}, adequately represents the objective function. This is based on a Taylor expansion of g (the surrogate model) around the current iterate q_k:

$$g(q_k + s) = g(q_k) + g'(q_k)^T s + \frac{1}{2} s^T g''(q_k + ts)s, \quad t \in [0, 1], \tag{4.16}$$

where $g'(q_k)$ and $g''(q_k)$ are the gradient and Hessian of g at q_k. Replacing the Hessian with a symmetric approximation H_k gives the simplified quadratic model:

$$m_k^{TR}(s) = g(q_k) + g'(q_k)^T s + \frac{1}{2} s^T H_k s. \tag{4.17}$$

As a reminder, we use finite differences to calculate both the derivative and Hessian of g.

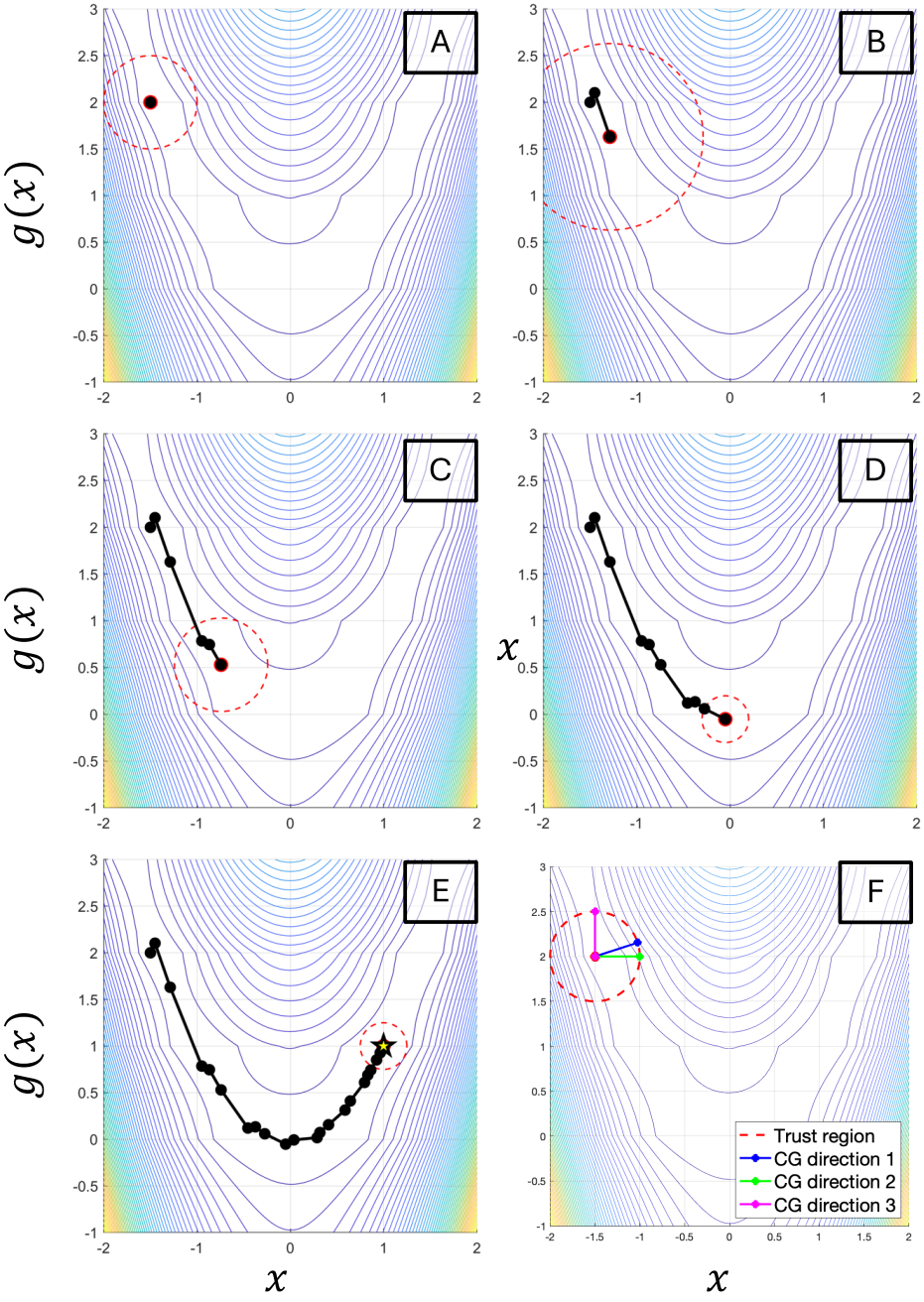


Figure 4.3: A-E: Example of trust-region steps as the algorithm searches for the minimum of the Rosenbrock function [89]. F: Steihaug conjugate gradient iterations for finding s_k.

The trust-region step is defined as the solution to the constrained quadratic minimization problem:

$$\min_{s \in \mathbb{R}^n} m_k^{TR}(s) \quad \text{subject to } \|s\| \leq \Delta_k^{TR}, \quad \Delta_k^{TR} > 0, \tag{4.18}$$

where Δ_k^{TR} is the trust-region radius at iteration k.

The quality of a step is assessed via a ratio of the actual and predicted reductions:

$$\rho_k^{TR} = \frac{g(q_k) - g(q_k + s_k)}{m_k^{TR}(0) - m_k^{TR}(s_k)}, \tag{4.19}$$

where s_k is the computed trust-region step. If $\rho_k^{TR} \approx 1$, the model closely approximates g and the trust region can be expanded. If $\rho_k^{TR} \ll 1$, the step is poor and the trust region should be shrunk. If $\rho_k^{TR} < 0$, the step increases the objective and should be rejected [12].

An outline of the trust region method is given in Algorithm 4.4.

Algorithm 4.4 Trust-Region Method

Require: Initial point q_0, trust-region radius Δ_0^{TR}, tolerance ε, max iterations $I_{\max}$, parameters $0 < \eta_1 < \eta_2 < 1$, scaling factors $0 < \gamma_1 < 1 < \gamma_2$

Output: Approximate minimizer q

$k \leftarrow 0$, $\Delta_k^{TR} \leftarrow \Delta_0^{TR}$

while $\|g'(q_k)\| > \varepsilon$ and $k < I_{\max}$ **do**

  Compute gradient $g'(q_k)$ and Hessian approximation H_k

  Solve the trust-region subproblem approximately using an iterative method, for example a conjugate gradient-based method (Alg. 4.5), to obtain s_k.

  Compute ratio of actual to predicted reduction:

$$\rho_k^{TR} = \frac{g(q_k) - g(q_k + s_k)}{m_k^{TR}(0) - m_k^{TR}(s_k)}$$

  if $\rho_k^{TR} < \eta_1$ **then**

    Reject step: $q_{k+1} = q_k$, $\Delta_{k+1}^{TR} = \gamma_1 \Delta_k^{TR}$

  else if $\rho_k^{TR} > \eta_2$ and $\|s_k\| \approx \Delta_k^{TR}$ **then**

    Accept step: $q_{k+1} = q_k + s_k$, $\Delta_{k+1}^{TR} = \gamma_2 \Delta_k^{TR}$

  else

    Accept step: $q_{k+1} = q_k + s_k$, $\Delta_{k+1}^{TR} = \Delta_k^{TR}$

  end if

  $k \leftarrow k + 1$

end while

return q_k

Steihaug Conjugate Gradient Method for the Trust-Region Subproblem

Solving Eq. (4.18) exactly is expensive. Steepest descent may be too slow, and dogleg methods require H_k to be positive definite [86]. Instead, we use Steihaug's conjugate gradient (CG) method [105], which approximately solves the trust-region subproblem. This method generates a search path using CG iterations rather than a piecewise linear dogleg path. An example of directions considered by this method is shown in Fig. 4.3.

The quadratic subproblem in Steihaug's CG method is:

$$m_k^{TR}(s) = g'(q_k)^T s^{CG} + \frac{1}{2}(s^{CG})^T H_k s^{CG}, \tag{4.20}$$

where H_k is a real symmetric matrix and $g'(q_k) \neq 0$. If H_k were positive definite and unconstrained, the minimizer would satisfy

$$H_k s^{CG} = -g'(q_k), \tag{4.21}$$

which is solved iteratively by CG.

Starting from $s_0^{CG} = 0$, $r_0^{CG} = g'(q_k)$, $d_0^{CG} = -r_0^{CG}$, the standard CG iterations are:

$$\alpha_i^{CG} = \frac{(r_i^{CG})^T r_i^{CG}}{(d_i^{CG})^T H_k d_i^{CG}}, \tag{4.22}$$

$$s_{i+1}^{CG} = s_i^{CG} + \alpha_i^{CG} d_i^{CG}, \tag{4.23}$$

$$r_{i+1}^{CG} = r_i^{CG} + \alpha_i^{CG} H_k d_i^{CG}, \tag{4.24}$$

$$\beta_i^{CG} = \frac{(r_{i+1}^{CG})^T r_{i+1}^{CG}}{(r_i^{CG})^T r_i^{CG}}, \tag{4.25}$$

$$d_{i+1}^{CG} = -r_{i+1}^{CG} + \beta_i^{CG} d_i^{CG}. \tag{4.26}$$

Steihaug's method introduces early stopping conditions:

1. **Boundary check:** If the next CG step would leave the trust region ($\|s_i^{CG} + \alpha_i^{CG} d_i^{CG}\| > \Delta_k^{TR}$), compute a truncated step:

$$s^* = s_i^{CG} + \tau d_i^{CG}, \quad \tau \geq 0 \text{ such that } \|s^*\| = \Delta_k^{TR}. \tag{4.27}$$

2. **Negative curvature check:** If $(d_i^{CG})^T H_k d_i^{CG} \leq 0$, the model is non-convex along d_i^{CG}. Truncate along the direction of negative curvature:

$$s^* = s_i^{CG} + \tau d_i^{CG}, \quad \tau \geq 0 \text{ chosen to approximately minimize } m_k^{TR}(s). \tag{4.28}$$

The step returned by Steihaug CG, s^*, is then assigned as the trust-region step: $s_k = s^*$. This ensures robustness even when H_k is indefinite or poorly conditioned.

Steihaug refers to the method as a preconditioned CG approach. The "preconditioning" here refers to early truncation via the stopping conditions, not the

classical use of a preconditioner [105]. Classical preconditioning can still be applied, but we use the standard method [15].

Intuitively, Steihaug's CG method can be thought of as taking a step downhill on the quadratic surface, pausing whenever it would step outside of the trust region or encounter a direction where the surface curves downward. Each step is therefore safe, feasible, and makes consistent progress toward minimizing the model, even if the Hessian is indefinite.

An outline of the Steihaug CG subproblem solver can be found in Algorithms 4.5.

Algorithm 4.5 Steihaug Conjugate Gradient Method for the Trust-Region Subproblem

Require: Gradient $g'(q_k)$, Hessian H_k (or approximation), trust-region radius Δ_k^{TR}, tolerance ε

Output: Approximate solution s_k of $\min_{\|s\| \leq \Delta_k^{TR}} g'(q_k)^T s + \frac{1}{2} s^T H_k s$

Initialize $s_0^{CG} = 0$, $r_0^{CG} = g'(q_k)$, $d_0^{CG} = -r_0^{CG}$
for $i = 0, 1, 2, \ldots$ **do**
  if $(d_i^{CG})^T H_k d_i^{CG} \leq 0$ **then** ▷ Negative curvature
    Find $\tau \geq 0$ such that $\|s_i^{CG} + \tau d_i^{CG}\| = \Delta_k^{TR}$
    return $s_k = s_i^{CG} + \tau d_i^{CG}$
  end if
  $\alpha_i^{CG} = \frac{(r_i^{CG})^T r_i^{CG}}{(d_i^{CG})^T H_k d_i^{CG}}$
  if $\|s_i^{CG} + \alpha_i^{CG} d_i^{CG}\| \geq \Delta_k^{TR}$ **then** ▷ Hit trust-region boundary
    Find $\tau \geq 0$ such that $\|s_i^{CG} + \tau d_i^{CG}\| = \Delta_k^{TR}$
    return $s_k = s_i^{CG} + \tau d_i^{CG}$
  end if
  Update:

$$s_{i+1}^{CG} = s_i^{CG} + \alpha_i^{CG} d_i^{CG}, \quad r_{i+1}^{CG} = r_i^{CG} + \alpha_i^{CG} H_k d_i^{CG}$$

  if $\|r_{i+1}^{CG}\| \leq \varepsilon$ **then**
    return $s_k = s_{i+1}^{CG}$
  end if
  $\beta_i^{CG} = \frac{(r_{i+1}^{CG})^T r_{i+1}^{CG}}{(r_i^{CG})^T r_i^{CG}}$
  $d_{i+1}^{CG} = -r_{i+1}^{CG} + \beta_i^{CG} d_i^{CG}$
end for

4.1.5 Multi-objective Optimization

To evaluate competing objective functions, multi-objective optimization can be used. This approach combines multiple objective functions into one composite objective function which will then be minimized. A simple example for two objective functions, $g_1(q)$ and $g_2(q)$, is the weighted additive objective function:

$$h_+(q) = g_1(q) + \omega_+ g_2(q), \tag{4.29}$$

where $g_1(q)$ and $g_2(q)$ are surrogate models representing different outputs, such as fluid flow and energy usage. In this formulation, the weight of $g_1(q)$ is fixed at 1, while the second weight, ω_+, adjusts the relative importance of $g_2(q)$ in the optimization.

An alternative is the multiplicative composite objective:

$$h_\times(q) = g_1(q)g_2(q)^{\omega_\times}. \tag{4.30}$$

Here, $\omega_\times$ controls the influence of $g_2(q)$, emphasizing solutions where both objectives perform well simultaneously rather than allowing one to dominate.

In this work, we adopt a scalarization-based approach, in which multiple objectives are combined into a single composite objective that can be minimized using standard optimization techniques. Common scalarization strategies include weighted additive or multiplicative formulations. While these approaches are straightforward to implement, their performance is highly sensitive to the choice of weights, which can be difficult to tune in practice, particularly when objectives differ in scale or exhibit conflicting behavior.

While additive and multiplicative scalarizations are straightforward to implement, they are highly sensitive to the choice of weights, especially when objectives differ in scale or exhibit conflicting trends. To address this limitation, we introduce the Augmented Chebyshev (AC) scalarization, which combines a weighted Chebyshev norm with a small augmentation term to smooth the objective landscape [11, 53]. For two objectives, the AC scalarization is defined as

$$\begin{aligned} h_{AC}(q) = &\max\left[\omega_{AC}|g_1(q) - z_1^*|,\, (1-\omega_{AC})|g_2(q) - z_2^*|\right] \\ &+ \rho_{AC}\Big(|g_1(q) - z_1^*| + |g_2(q) - z_2^*|\Big), \end{aligned} \tag{4.31}$$

where $\omega_{AC} \in [0, 1]$ controls the relative weighting of the objectives, with larger values favoring $g_1(q)$. The bounded and interpretable nature of ω enables systematic exploration of trade-offs between objectives by varying a single parameter. The reference values z_1^* and z_2^* are typically the maximum observed or ideal values of $g_1(q)$ and $g_2(q)$. The first term in Eq. (4.31) penalizes solutions that perform poorly on either objective, while the second term, scaled by a small augmentation parameter $\rho_{AC} \ll 1$, smooths the objective landscape, producing a gradual transition between the optima of the individual objectives. Following common practice in the literature, we adopt $\rho_{AC} = 0.05$ [11, 53].

Preliminary Multi-Objective Optimization Tests

To determine an appropriate multi-objective composite function, we first tested additive, multiplicative, and augmented Chebyshev (AC) scalarizations on simple surrogate models. Here, $g_1(q)$ represents the vertical flow in region V_A, and $g_2(q)$ represents the tension-based energy, with the optimization parameters limited to t and Re. All surrogate outputs were normalized and standardized prior to optimization, though reported values for $g_1(q)$, $g_2(q)$, and parameter combinations are converted back to physical units. The composite objectives were minimized to balance the competing goals of maximizing vertical fluid flow while minimizing energy expenditure.

For this test problem, the maximum of $g_1(q)$ occurs at $(t, Re) = (0.4313, 1000)$, while the minimum of $g_2(q)$ occurs at $Re = 1$, as $g_2(q)$ is independent of t. Tables 4.1-4.3 summarize results for the additive, multiplicative, and AC objectives across several weighting parameters.

For the additive and multiplicative objectives, the pattern search method consistently achieved the lowest objective values with the shortest compute times, reflecting its ability to take large exploratory steps with relatively few function evaluations. In contrast, for the AC formulation, pattern search performed best when $\omega_{AC} \leq 0.25$, whereas gradient descent dominated for $\omega_{AC} \geq 0.5$. Gradient descent required more computational time in these regimes but converged to lower objective values, highlighting how the weighting parameter shapes the optimization landscape and favors different methods under different conditions.

The additive and multiplicative objectives also exhibited sharp transitions in the location of the optimum as their relative weights changed. For $h_+(q)$, when $\omega_+ \leq 4.9$, the optimizer converged to the maximum of $g_1(q)$, but for $\omega_+ \geq 4.95$, the optimum abruptly shifted to the minimum of $g_2(q)$, indicating dominance of the energy term. Similarly, $h_\times(q)$ remained at the maximum of $g_1(q)$ for $\omega_\times \leq 1.75$ but jumped to the minimum of $g_2(q)$ for $\omega_\times \geq 1.8$. These sharp transitions make meaningful weighting difficult without extensive parameter exploration.

ω_+	Method	$h_+(q)$	$g_1(q)$	$g_2(q)$
2	Pattern Search	0.549	0.008	1.054
4.9	Pattern Search	-0.302	0.008	1.054
4.95	Pattern Search	-0.400	0.001	0.331
5	Pattern Search	-0.3145	0.0011	0.3313

Table 4.1: Best-performing results for the additive composite objective $h_+(q)$. g_1 is vertical flow in V_A, g_2 is tension-based energy. Other optimization methods exhibited similar trends but required more iterations or longer compute times.

$\omega_\times$	Method	$h_\times(q)$	$g_1(q)$	$g_2(q)$
1.5	Pattern Search	7.148	0.008	1.054
1.75	Pattern Search	9.713	0.008	1.054
1.8	Pattern Search	10.681	0.011	0.331
2	Pattern Search	17.205	0.001	0.331

Table 4.2: Best-performing results for the multiplicative composite objective $h_\times(q)$. g_1 is vertical flow in V_A, g_2 is tension-based energy. Other methods followed similar trends but required more iterations or compute time.

In contrast, the AC formulation produces a smoother transition between objectives. By construction, it balances maximizing g_1 and minimizing g_2 across $\omega_{AC} \in [0, 1]$. Small

ω_{AC} values favor g_2, while larger values shift the solution toward maximizing g_1. The weighting parameter is bounded and interpretable, allowing systematic exploration of trade-offs between objectives. For the simple surrogate problem, $\omega_{AC} = 0.25$ provided the closest balance between objectives while still favoring g_2.

ω_{AC}	Method	h_{AC}	$g_1(q)$	$g_2(q)$
0.10	Pattern Search	-0.094	0.001	0.331
0.25	Pattern Search	-0.205	0.002	0.898
0.50	Gradient Descent	-0.152	0.008	1.058
0.75	Gradient Descent	-0.077	0.008	1.075
0.90	Gradient Descent	-0.031	0.008	1.081

Table 4.3: Best-performing results for the augmented Chebyshev (AC) composite objective h_{AC} with $\rho = 0.05$. g_1 is vertical flow in V_A, g_2 is tension-based energy. Pattern Search dominates for low ω_{AC}, while Gradient Descent is best for higher ω_{AC}. Other methods followed similar trends but required more iterations or compute time.

Based on these observations, we adopt the AC scalarization with a range of ω_{AC} values for the full parameter-space exploration. For notational simplicity, we refer to ω_{AC} as ω in subsequent sections.

4.1.6 Implementation

All optimization methods are implemented in MATLAB. Upper and lower bounds are imposed on each parameter to ensure feasible solutions: $t \in [0, 1.89]$, $Re \in [1, 1000]$, $T_{\max} \in [500, 10000)$, $\psi = T_{\max}/\eta_T \in [0.25, 3)$, and $L_e \in (0.1, 0.9)$. These ranges were selected to represent both the physical and computational limits of the system.

Because the initial guess can strongly influence optimization outcomes, the initial point is chosen as the average of the parameter space, i.e. $q_0 = \{\bar{t}, \bar{Re}, \bar{T}_{\max}, \bar{\psi}, \bar{L}_e\}$ with $\bar{\cdot}$ denoting the average. The optimization bounds and initial starting guess are provided in Table 4.4. To mitigate local minima, additional start points are generated through randomized grid sampling [72]. The final solution is selected as the best result among all runs.

Full parameter-space optimizations were performed on the Pinnacles cluster, a High Performance Computing system at UC Merced.

4.2 Results

We now discuss optimization of the full parameter space. We start by optimizing only a single surrogate objective function at a time. Specifically, we will be maximizing the vorticity and vertical flow surrogates and minimizing the energy and power surrogates. This will allow us to find the parameters that produce either the greatest fluid flow or are

Name	Variable	Baseline	Initial Guess	Bounds	
				Min	Max
Time in pulse	t	–	0.9450	0	1.89
Reynolds number	Re	46.5892	500.5	1	1000
Maximum muscle tension	$T_{\max}$	4,500	5,250	500	10,000
Tension–elasticity ratio	$\psi = T_{\max}/\eta_t$	0.9	1.625	0.25	3.0
Muscle end ratio	L_e	0.45	0.5	0.1	0.9

Table 4.4: Baseline values, initial guesses, and optimization bounds for surrogate model optimization.

the least energetically expansive. At this time, we omit the kinematic surrogate, though this could also be optimized to determine the parameters which result in the greatest tentacle motion. After single-objective optimization, we move onto multi-objective optimization, where we take into account both the fluid flow and energetic expenditure. Whether a surrogate function is minimized or maximized will stay consistent throughout both the single- and multi-objective optimization.

We compare results for each of the four methods previously presented for both single- and multi-objective optimization. Each optimization method is allowed a maximum of 10,000 iterations and has 100 start points sampled via a Latin hypercube sample.

4.2.1 Single-Objective Optimization of GP Models

A summary of the optimization results for vorticity and vertical flow, both of which were maximized, is presented in Table 4.5. Among the methods tested, Nelder-Mead consistently produced the best performance. Pattern search also performed reasonably well, but appeared to converge prematurely, a behavior that could potentially be mitigated by relaxing its termination criteria.

Across both vorticity and vertical flow optimizations, the optimal end length of the tentacle was found to be $L_e \approx 0.45$-0.48, closely matching the biological value of $L_e = 0.45$. In contrast, the optimal tension-elasticity ratio falls in the range $\psi = T_{\max}/\eta_t \approx 1.35$-$1.90$, which is substantially larger than the biological value of $\psi = 0.9$ used in the base model. Similarly, the optimal tension values lie in the range $T_{\max} \approx 4,675$-$6,650$, slightly exceeding the base value of $T_{\max} = 4,500$. Taken together, these results suggest that maximizing fluid transport within the current framework favors increased muscle stiffness and tension relative to the base parameter choices. This discrepancy may reflect either limitations of the base parameter assumptions and the fact that the optimization prioritizes fluid transport efficiency over other physiological considerations.

Metric	Method	$o(q)$	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
Vorticity	Nelder-Mead	28.870	0.62	1,000	4,675	1.35	0.48	0.072
Flow in V_A	Nelder-Mead	0.012	0.55	266	6,280	1.55	0.46	0.083
Flow in V_B	Nelder-Mead	0.009	0.98	167	6,650	1.90	0.45	0.102

Table 4.5: Optimization results showing only the best-performing method for each metric. All function outputs are maximized. Full results for all methods are provided in Appendix D.

During the initial optimization of energy and power expenditures, the surrogate models predicted negative minima, which were not observed in the corresponding simulations.This discrepancy is attributed to Gaussian process extrapolation, which produces an artificial drop-off at low Reynolds numbers, as evident in Figs. 4.4–4.5, which are recreations of the tension-based results presented in Fig. 3.23–3.24 with a focus on low Reynolds numbers. Physically, these negative predictions are not meaningful: tension-based energy depends on the pulse-averaged muscle tension and the norm of the structures displacement, while tension-based power depends on muscle tension and the norm of the Lagrangian tentacle velocity. Similarly, force-based energy and power depend on the magnitude of the Lagrangian force and tentacle velocity. All of these quantities are inherently non-negative.

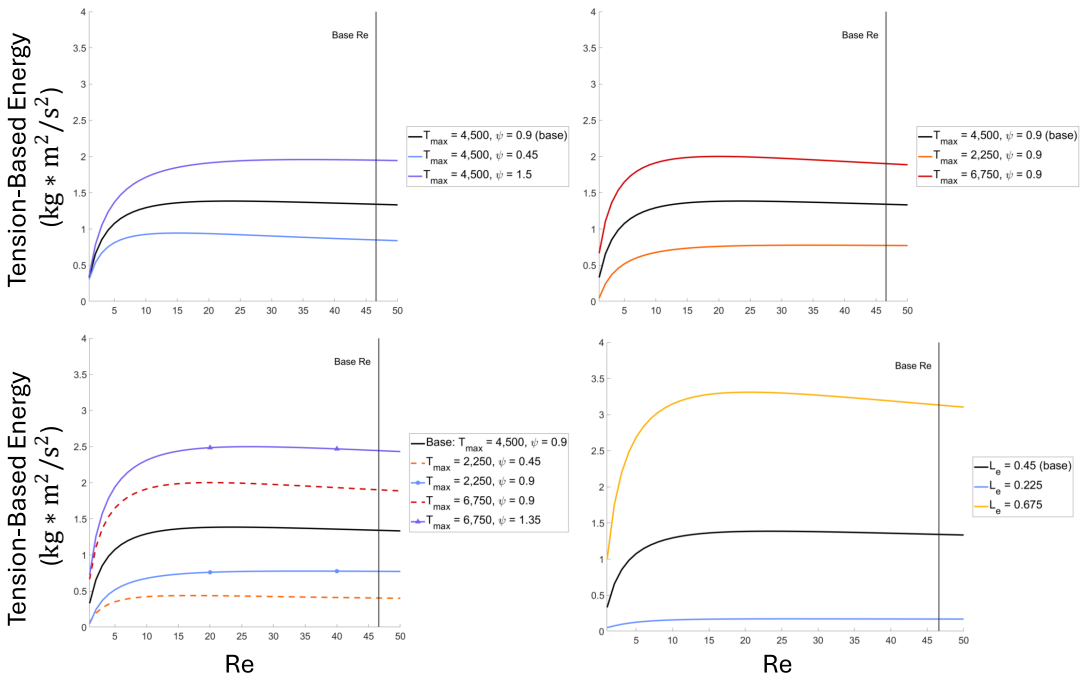


Figure 4.4: Predicted tension-based energy across muscle parameter variations. Recreation of Fig. 3.23 with a focus on low Reynolds numbers.

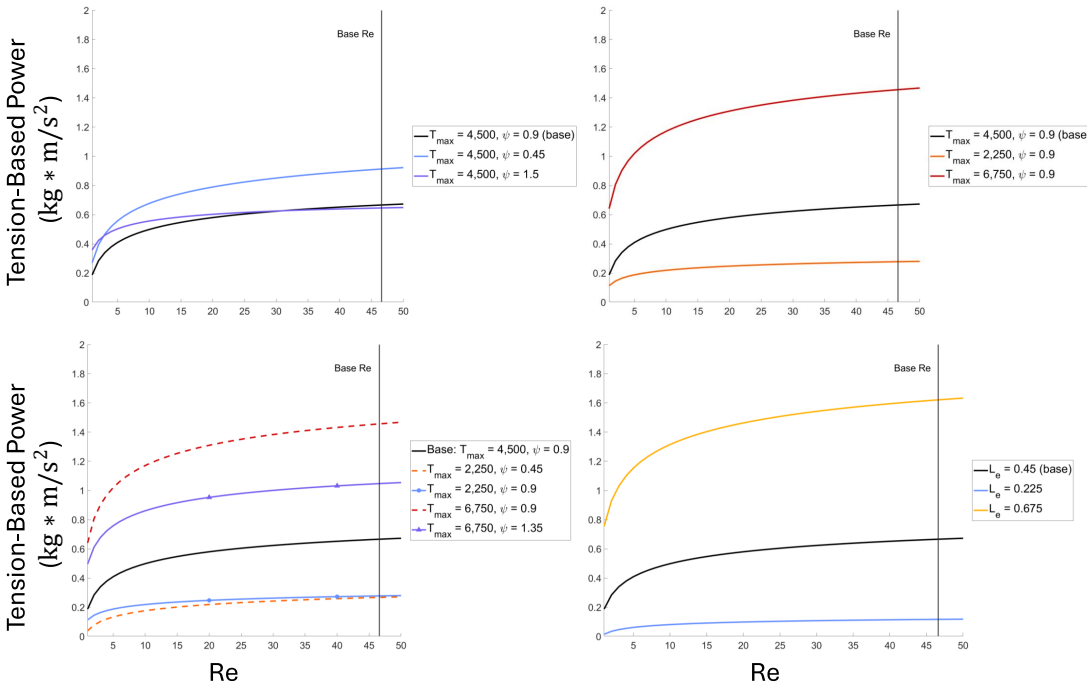


Figure 4.5: Predicted tension-based energy across muscle parameter variations. Recreation of Fig. 3.24 with a focus on low Reynolds numbers.

To enforce this physical condition, we impose a hard constraint requiring the predicted objective values at the optima to be non-negative for each metric. Enforcing this constraint allows for the possibility that multiple parameter combinations yield near-zero energy or power, as observed for the tension-based energy metric, where both the Nelder-Mead and trust-region methods converge to similar objective values using different parameter sets. With this in mind, a result of zero energy or power would come from a stationary (or nearly-stationary) coral. Therefore parameter combinations that produce this result are unlikely to have such a low output if they were to be run as full simulations, but can help point to parameter regimes that are more cost effective.

With this hard constraint enforced, the Nelder-Mead method consistently achieved the lowest values across all cost metrics (Tab. 4.6). The minimum energy and power typically occur when Re, $T_{\max}$, and L_e are small. This behavior is expected, as reducing any of these parameters limits the coral polyps ability to pulse, resulting in muted motion and weaker fluid flow. Comparing tension- and force-based results, we observe that the force-based optimization tends to favor a higher tension-elasticity ratio, ψ and a lower muscle length, L_e. Physically, a high tension-elasticity ratio indicates a more tension-dominated, less elastic response, while a shorter muscle length reduces the overall kinematic motion of the coral polyp. Together, these effects likely suppress the generation of Lagrangian forces used to compute this metric.

While these single-optimization results point towards the parameters which produce the greatest flow or least energy cost, the coral has many other competing priorities. Therefore,

it is more realistic to consider an optimization scheme which attempts to balance these needs. This will be explored in the following section, where multi-objective optimization is employed.

Metric	Method	$o(q)$	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
Tension Energy	Nelder-Mead	0	-	7	2,481	0.49	0.37	0.069
Tension Power	Nelder-Mead	0	-	3	3,279	0.64	0.37	0.082
Force Energy	Nelder-Mead	0	-	29	2,449	1.23	0.25	0.103
Force Power	Nelder-Mead	0	-	2	3,488	1.45	0.19	0.104

Table 4.6: Optimization results showing only the best-performing method for each metric. All function outputs are minimized. Function outputs smaller than 1×10^{-6} have been rounded to 0. Full results for all methods are provided in Appendix D.

4.2.2 Multi-Objective Optimization of GP Models

Multi-objective optimization was performed for the vertical flow in region V_A and for vorticity, each paired with tension-based energy and power. We focus on V_A because flow is strongest near the coral polyp, and on tension-based metrics because they produce smoother surrogate models. Tables 4.7 and 4.8 summarize the best-performing method for vorticity, while Tables 4.9 and 4.10 report results for vertical flow, each paired with energy and power. Full results for all methods and parameter sets are provided in Appendix E. As a reminder, $\omega \in [0, 1]$ is the weight parameter for the Augmented Chebyshev scalarization, with $\omega \approx 0$ favoring g_2 and $\omega \approx 1$ favoring g_1. Throughout this results section, g_1 will be represented by either the surrogate for vertical flow or vorticity and will thus be maximized. Alternatively, g_2 will be the surrogate for tension-based energy or power and will be minimized. Therefore, a smaller ω will favor minimizing cost, while a larger ω will favor increasing vertical flow.

ω	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
0.10	Trust Region	-0.088	3.85	0.32	0.67	348	6,497	0.61	0.25	0.68
0.25	Trust Region	-0.160	8.31	0.70	0.78	441	4,451	1.02	0.34	153.91
0.50	Trust Region	-0.149	14.51	0.97	0.59	533	5,847	0.61	0.44	77.34
0.65	Trust Region	-0.108	16.90	1.01	0.67	466	4,580	1.03	0.40	0.79
0.75	Gradient Descent	-0.073	18.16	0.95	0.62	541	4,054	1.00	0.42	0.89
0.90	Pattern Search	-0.021	27.41	1.31	0.65	1,000	4,904	1.25	0.44	0.04

Table 4.7: Best-performing method for each ω in the augmented Chebyshev optimization for vorticity and tension-based energy. Full results for all methods are provided in Appendix E.

ω	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
0.10	Trust Region	-0.085	4.48	0.15	0.79	424	4,492	1.04	0.27	1.02
0.25	Gradient Descent	-0.159	8.39	0.35	0.85	369	4,607	1.37	0.35	2.65
0.50	Trust Region	-0.175	13.59	0.57	0.64	320	3,280	1.17	0.47	155.04
0.65	Gradient Descent	-0.099	17.15	0.46	0.60	374	4,607	1.35	0.39	0.62
0.75	Gradient Descent	-0.077	18.06	0.50	0.61	375	4,607	1.36	0.40	0.35
0.90	Gradient Descent	-0.033	19.24	0.55	0.62	442	4,492	1.14	0.41	0.48

Table 4.8: Best-performing method for each ω in the augmented Chebyshev optimization for vorticity and tension-based power. Full results for all methods are provided in Appendix E.

ω	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
0.10	Gradient Descent	-0.080	0.003	0.29	0.30	484	4,054	0.93	0.25	0.40
0.25	Gradient Descent	-0.152	0.005	0.66	0.40	232	3,627	0.68	0.43	0.59
0.50	Pattern Search	-0.162	0.008	1.03	0.46	483	6,081	0.94	0.38	0.02
0.65	Pattern Search	-0.132	0.009	1.23	0.49	468	5,046	1.06	0.43	0.05
0.75	Pattern Search	-0.110	0.010	1.46	0.50	358	4,904	1.25	0.44	0.02
0.90	Pattern Search	-0.049	0.011	1.63	0.49	341	6,333	1.23	0.43	0.05

Table 4.9: Best-performing method for each ω in the augmented Chebyshev optimization for vertical flow in V_A and tension-based energy. Full results for all methods are provided in Appendix E.

ω	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$ (N)	ψ	L_e	Compute Time (s)
0.10	Gradient Descent	-0.070	0.004	0.11	0.35	485	4,054	0.96	0.29	0.35
0.25	Gradient Descent	-0.131	0.006	0.28	0.41	196	3,702	1.23	0.38	0.50
0.50	Pattern Search	-0.146	0.008	0.48	0.60	508	5,187	1.68	0.36	0.02
0.65	Pattern Search	-0.123	0.009	0.57	0.55	398	5,857	1.63	0.36	0.01
0.75	Pattern Search	-0.096	0.010	0.59	0.61	344	5,014	1.66	0.40	0.05
0.90	Gradient Descent	-0.046	0.010	0.75	0.53	466	4,733	1.53	0.46	2.81

Table 4.10: Best-performing method for each ω in the augmented Chebyshev optimization for vertical flow in V_A and tension-based power. Full results for all methods are provided in Appendix E.

While no single optimization method consistently performs best, clear patterns emerge. In the case of vorticity with energy, the trust-region method often performs the best. Here gradient-free methods fall into a local minimum at $Re = 1000$ and fail to progress further. Similarly, for vorticity with power, gradient-based methods continue to dominate, though it is more split between gradient descent and the trust-region. Often, these methods converge to similar enough minima that gradient descent could be used moving forward as it is the simpler technique. Shifting to vertical flow, we see that pattern search and gradient descent alternate as the dominant method, depending on the value of ω. These results suggest that pattern search and gradient descent together provide adequate coverage of the optimization landscape and are the preferred choices for future analyses.

Turning to the optimal parameters, several consistent trends emerge. There is a clear trade-off between flow and cost, unlike the preliminary test problem, where solutions jumped abruptly between extremes. Here, higher flow is increasingly favored as ω grows. Most solutions converge to $Re \geq 200$, even at low ω, while shifts in other parameters are non-uniform. This highlights the non-convexity of the problem and shows that different parameter combinations can produce similar optimal outputs. When compared with biological data, solutions for $\omega = 0.5$-0.75 most closely match observed polyp parameters, capturing the expected balance between strong vertical flow and minimal energetic expenditure.

For vertical flow, multi-objective optimization favors a larger Reynolds number than the single-objective optimization (Table 4.5), while simultaneously reducing $T_{\max}$ and increasing ψ. This suggests that reducing energetic cost can be achieved by lowering overall muscle tension and elasticity. In contrast, optimizing vorticity shifts the solution toward lower Reynolds numbers and reduces elasticity by increasing ψ, indicating a different balance when prioritizing fluid rotation versus upward transport. Overall, the multi-objective formulation reveals biologically meaningful trade-offs between flow generation and energetic expenditure.

4.3 Conclusion

In this chapter, we demonstrated that both single- and multi-objective optimization can be applied to our surrogate models to identify parameter values that maximize or minimize desired outputs. The results reveal biologically meaningful trade-offs between flow generation and energetic expenditure, showing how the coral balances its competing needs, and highlight how different objective functions lead to distinct optimal strategies. This demonstrates that the surrogate models are effective tools for guiding optimization across a high-dimensional parameter space, despite localized limitations. Using these optimal parameters, full numerical simulations can be re-run to validate surrogate predictions, explore regions of interest in greater detail, and develop improved surrogate models. This iterative process of optimization and model refinement will be explored further in Ch. 5.

Chapter 5

Iterative Surrogate Model Refinement

The workflow introduced in previous chapters can be iterated to progressively improve model accuracy and insight. In this chapter, the optimal parameters identified in Ch. 4 are used to re-run the numerical simulations from Ch. 1, generating additional data in regions of interest. From these simulations, updated surrogate models are constructed and re-optimized, forming an iterative cycle illustrated in Fig. 5.1.

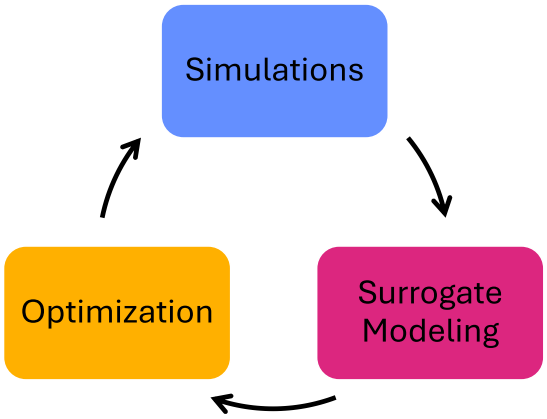


Figure 5.1: After the initial simulations, modeling, and optimization are complete, the resulting optimal parameters can be used to generate additional simulations in regions of interest. This process can then be repeated to progressively improve the surrogate model accuracy.

This iterative approach has two key objectives. First, it validates the accuracy of the initial surrogate by comparing predicted optima with high-fidelity numerical results. Second, it refines the parameter space around regions of interest, allowing subsequent simulations to be conducted more efficiently and with improved resolution.

5.1 Model Verification

We selected parameter sets from the multi-objective optimization corresponding to $\omega = 0.25$ and $\omega = 0.65$. The lower value, $\omega = 0.25$, emphasizes minimizing energy expenditure, yielding smaller Reynolds numbers that align closely with the biologically relevant regime for Vertical Flow. The intermediate value, $\omega = 0.65$, represents a balance between fluid transport and energetic cost, falling within the range ($\omega = 0.5$-0.75) that most closely matches observed polyp behavior.

The chosen parameter sets are summarized in Table 5.1. Full simulations were run with these parameter settings, and their kinematic and flow results were used to test the surrogate models predictive accuracy, with this new data serving as out-of-sample evaluation data which was not in the original training set. Following Section 3.3, the relative error

$$RE_i = \frac{|o_i - \hat{o}_i|}{|\hat{o}_i|}, \tag{5.1}$$

quantifies the magnitude of the error of the prediction o_i relative to the true output $\hat{o}_i$, providing a normalized measure of accuracy across outputs with different scales. The relative error of the original surrogate model is summarized in Table 5.2, with further discussion provided in the following subsections. For the kinematic motion, vorticity, and vertical flow, which are all dependent on time, the relative error is calculated at each time step and then averaged over all time points for each of the eight simulations. For energy and power, which are independent of time, the relative error is calculated using the eight simulation values directly. Further details and results are presented in the following subsections.

Test ID	Maximized Parameter	Minimized Parameter	ω	Re	$T_{\max}$ (N)	ψ	L_e	Source Table
1	Vorticity	Energy	0.25	441	4,451	1.02	0.34	Table 4.7
2	Vorticity	Power	0.25	369	4,607	1.37	0.35	Table 4.8
3	Vertical Flow	Energy	0.25	232	3,627	0.68	0.43	Table 4.9
4	Vertical Flow	Power	0.25	196	3,702	1.23	0.38	Table 4.10
5	Vorticity	Energy	0.65	466	4,580	1.03	0.40	Table 4.7
6	Vorticity	Power	0.65	374	4,607	1.35	0.39	Table 4.8
7	Vertical Flow	Energy	0.65	468	5,046	1.06	0.43	Table 4.9
8	Vertical Flow	Power	0.65	398	5,857	1.63	0.36	Table 4.10

Table 5.1: Simulation parameters based on optimization results, with test identifiers used for reference in subsequent figures. Note that energy and power each refer to the tension-based definitions.

Metric	Average Relative Error	Median Relative Error	Maximum Relative Error
Kinematic Motion	7.12%	5.46%	39.31%
Vorticity– full curve	40.02%	32.36%	541.44%
Vorticity– at peak value	17.85%	16.62%	33.44%
Vertical Flow in V_A– full curve	35.44%	31.14%	140.65%
Vertical Flow in V_A– at peak value	9.74%	7.97%	22.22%
Vertical Flow in V_B– full curve	17.92%	16.78%	46.73%
Vertical Flow in V_B– at peak value	15.23%	12.03%	28.76%
Tension-based Energy	11.41%	11.17%	27.93%
Tension-based Power	17.96%	17.03%	31.21%
Force-based Energy	14.12%	12.96%	31.34%
Force-based Power	6.38%	4.26%	14.60%

Table 5.2: Relative error of surrogate models evaluated on new simulations not used in training. Note that relative error at the peak values, which are the values of interest, are relatively small.

5.1.1 Kinematic Comparison

We evaluated the surrogate model for both the x and z position of the tentacle kinematics across the eight test cases. Highlights of these results are shown in Fig. 5.2. The top panel presents the time-averaged relative error for each new simulation, computed using the combined 2D displacement: at each time point, the error is calculated as the Euclidean distance between the predicted and true positions normalized by the true position, and then averaged over time. This combined metric ranges from 0.044 to 0.141. Test ID #5 exhibits the lowest error, with predicted and true trajectories closely aligned; the largest deviations occur near the loop extrema, where the polyp transitions between contraction and expansion.

By contrast, Test ID #8 shows the highest relative error. Here, the predicted trajectory is less smooth and shorter than the full simulation result. This behavior is associated with the parameter regime. Both $T_{\max}$ and ψ are at their largest values, representing a high muscle tension-to-elasticity ratio. This is because the training dataset only included ψ values up to 1.5, meaning this test case partially exceeds the surrogates trained domain.

Even in this extreme case, the surrogate maintains relative errors below 40% across all unseen simulations at any time point, demonstrating that it reliably captures the overall pattern of tentacle kinematics both within and slightly beyond the training parameter space.

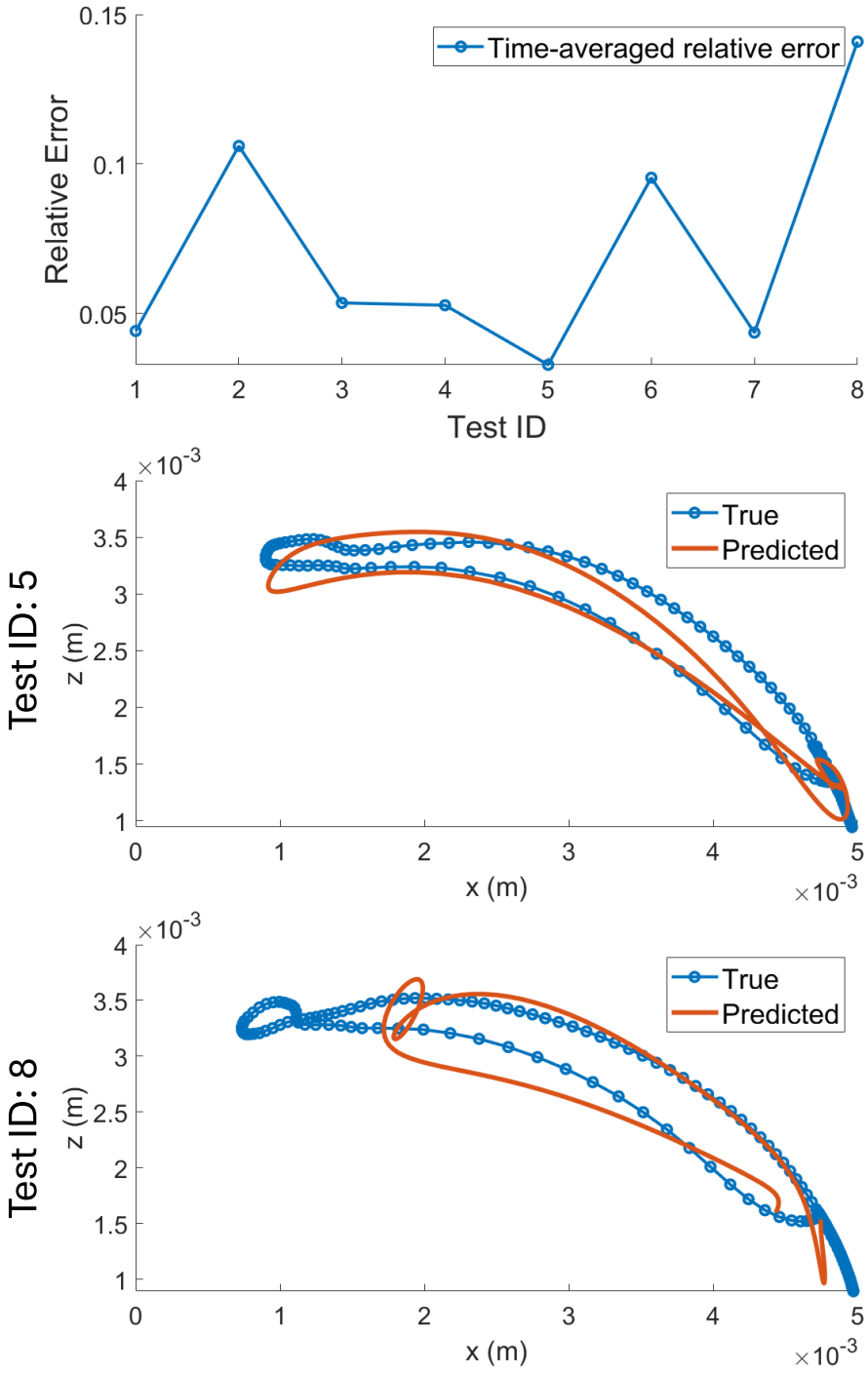


Figure 5.2: Top: Surrogate model accuracy for tentacle kinematics on unseen data. Middle and bottom: Predicted versus true trajectories for the best (ID #5) and worst (ID #8) relative errors. Test IDs correspond to Table 5.1.

5.1.2 Vorticity

Next, we examine the spatially averaged magnitude of vorticity, shown in Fig. 5.3. A key question is whether the surrogate captures the full result over time or mainly reproduces the peak value. Because the goal of optimization emphasizes peak vorticity, we focus on evaluating the surrogate's performance in reproducing the peak. Unlike the time-averaged relative error, which is computed over all time points, the relative error at the peak is calculated only at the peak point in each simulation and then averaged across the eight simulations.

Across the test cases, relative error at the peak ranges from 0.031 to 0.334, presented in the top panel of Fig. 5.3. Test ID #4 exhibits the highest error in this regard, while Test ID #8 exhibits the lowest. Test ID #8 accurately reproduces the peak vorticity but fails to reproduce the spatially averaged vorticity over time, highlighting that surrogate performance is most critical in the context of the modeling objective. While the surrogate captures peak behavior effectively, improving accuracy over the entire time series would likely require additional training data.

5.1.3 Volumetric Flux

We next examine the surrogate models for vertical flow in regions V_A and V_B (see Fig. 2.7 for region definitions). Relative errors at the peak and over time are presented in the top panels of Figs. 5.4–5.5, while the middle and bottom panels show the simulations with the best and worst peak predictions. As with vorticity, the primary interest is accurately predicting peak flow rather than reproducing the full temporal profile, so we focus on cases that performed best and worst in this regard.

For V_A, Fig. 5.4, the relative error at the peak is substantially lower than the time-averaged error. Test ID #1 achieves the lowest peak error (0.027), whereas Test ID #6 performs worst (0.222). Notably, Test ID #6 has a $T_{\max}$ similar to the baseline but a much larger ψ, corresponding to a coral polyp with reduced elasticity. This suggests the surrogate captures peaks well under most conditions but struggles to reproduce the general output trends.

For V_B (Fig. 5.5), the peak and time-averaged relative errors are more similar. Test ID #7 performs best, while Test ID #6 again performs worst. When the peak flow is predicted accurately, the surrogate also reproduces the overall data trends, but poor peak predictions coincide with less accurate predictions over the entire time period.

Overall, relative errors remain below 22% for V_A and 28.8% for V_B, even in challenging regions of the parameter space. This demonstrates that the surrogate reliably captures the dominant features of vertical flow, particularly the peaks most relevant for evaluating coral performance. While overall data trends may be less precise in some cases, the model provides a robust and efficient tool for identifying conditions that maximize flow.

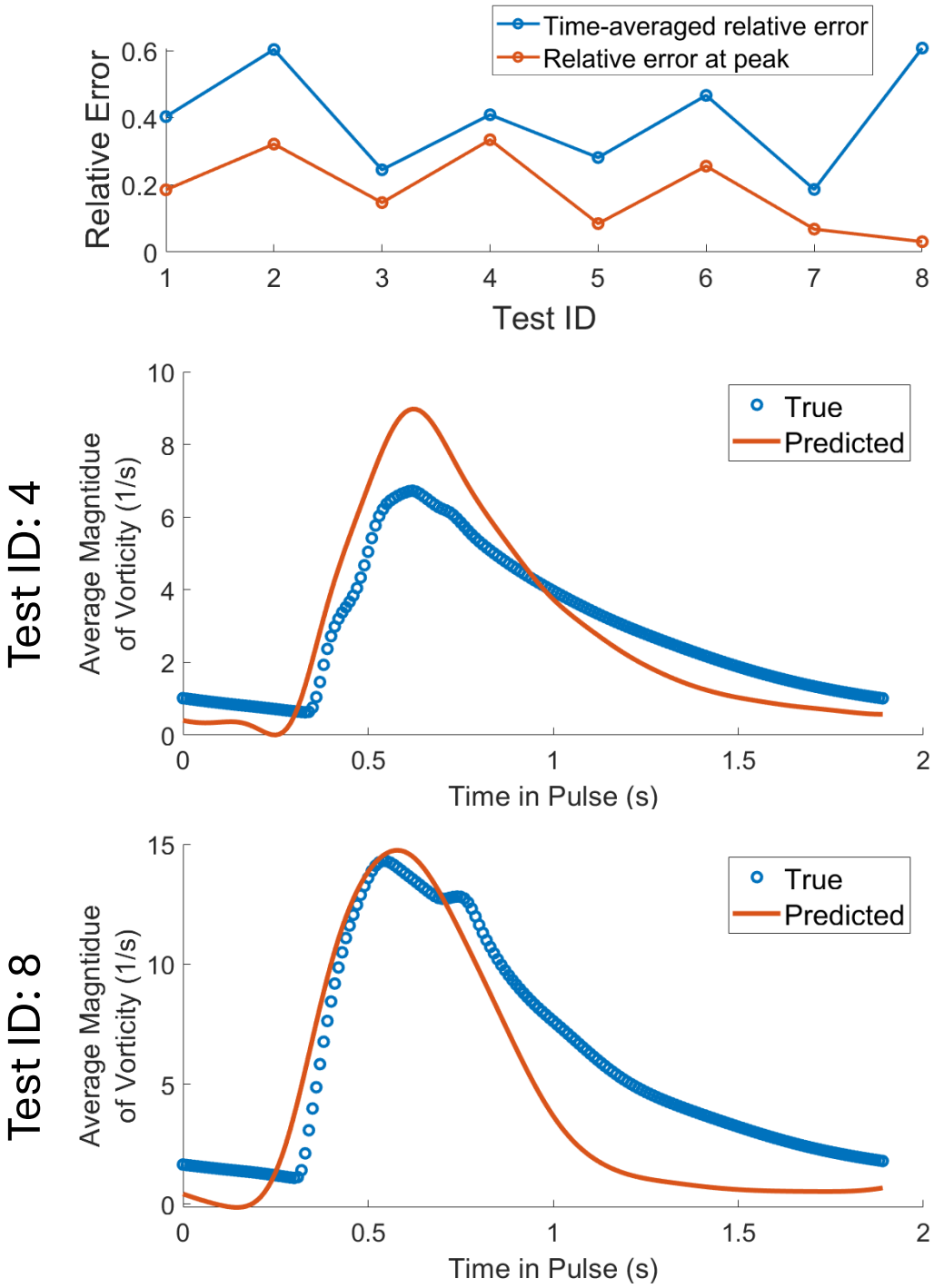


Figure 5.3: Top: Surrogate model accuracy for the spatially averaged magnitude of vorticity for unseen simulations. Middle and bottom: Predicted versus true outputs for the best (ID #8) and worst (ID #4) relative errors at the peak value. Test IDs correspond to Table 5.1.

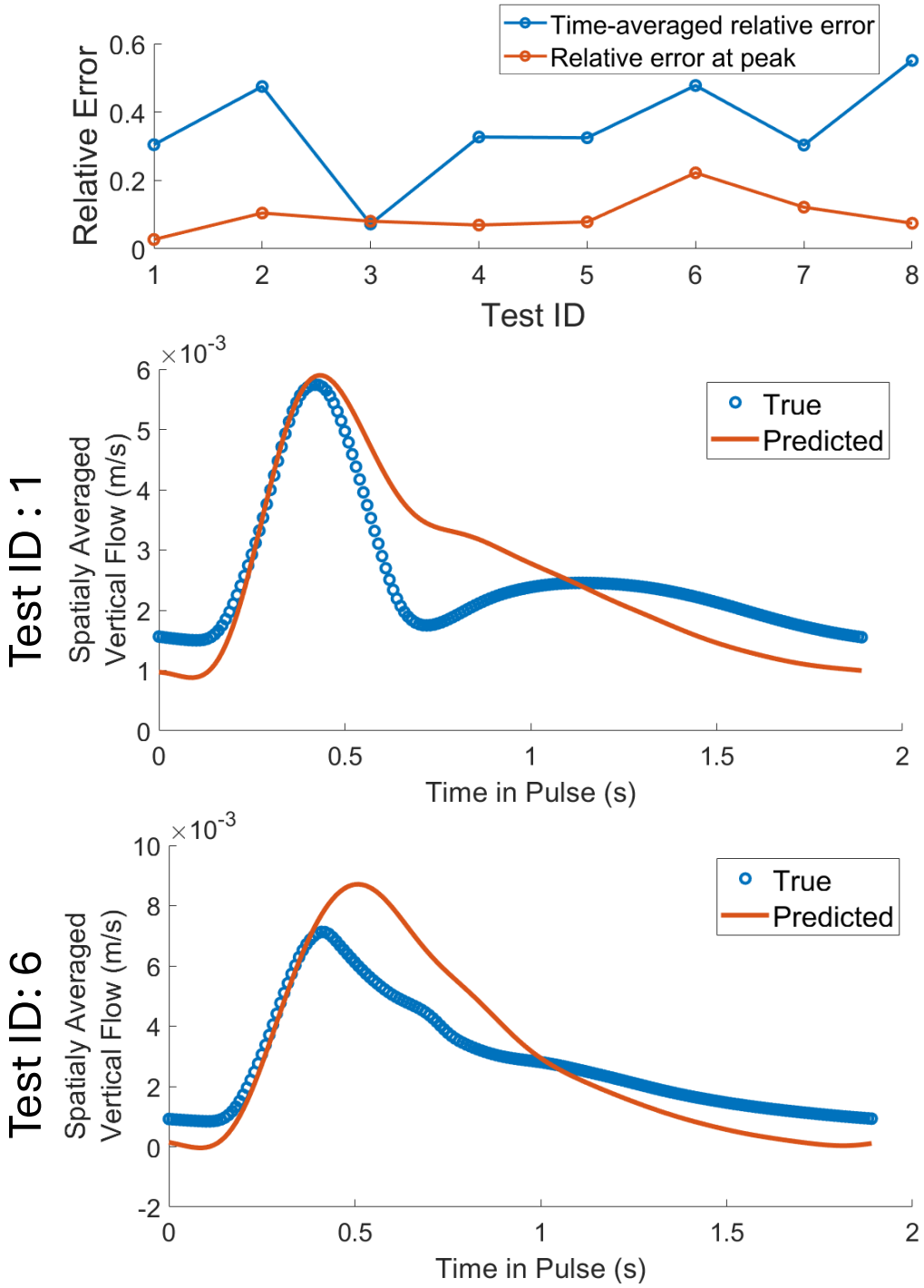


Figure 5.4: Top: Surrogate model accuracy for the spatially averaged vertical flow in V_A. Middle and bottom: Predicted versus true outputs for the best (ID #1) and worst (ID #6) relative errors at the peak value. Test IDs correspond to Table 5.1.

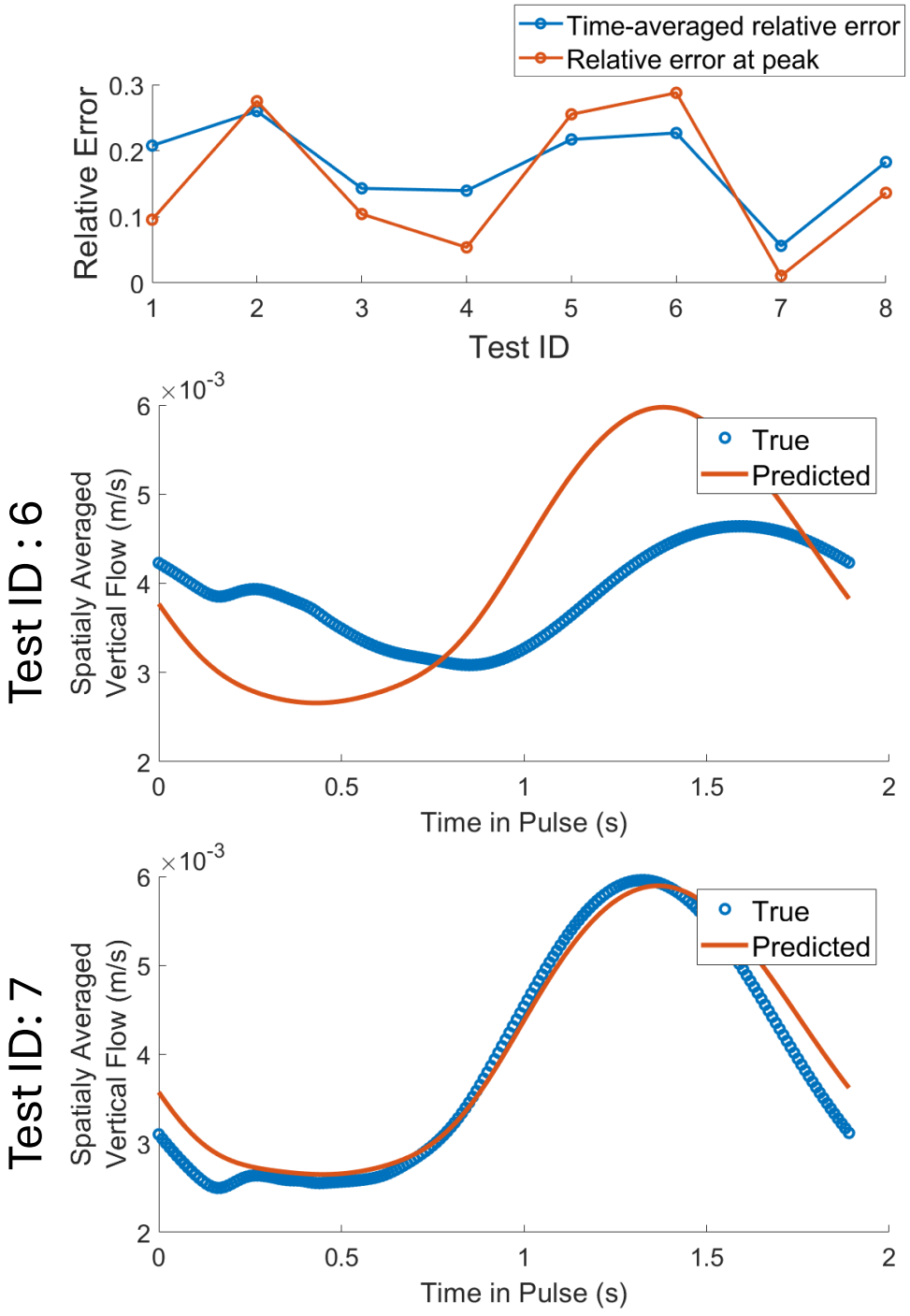


Figure 5.5: Top: Surrogate model accuracy for the spatially averaged vertical flow in V_B for unseen simulations. Middle and bottom: Predicted versus true outputs for the worst (ID #6) and best (ID #7) relative errors at the peak value. Test IDs correspond to Table 5.1.

5.1.4 Energy and Power Expenditure

Finally, we evaluate the surrogate models for tension- and force-based energy and power. Tension-based results are shown in Fig. 5.6, while force-based quantities are presented in Fig. 5.7. Among these, the force-based power surrogate performs best, with a maximum relative error of 0.146. In contrast, the remaining surrogates exhibit maximum errors of roughly 30%. Interestingly, the force-based power surrogate shows the lowest maximum error on unseen data despite having the largest training MSE, emphasizing that training error alone is not a reliable indicator of generalization.

No test ID consistently yields the best or worst performance across all energy and power metrics. This demonstrates that surrogate accuracy depends on the output quantity rather than the parameter set, and that no parameter combination uniformly produces highly accurate or highly inaccurate predictions.

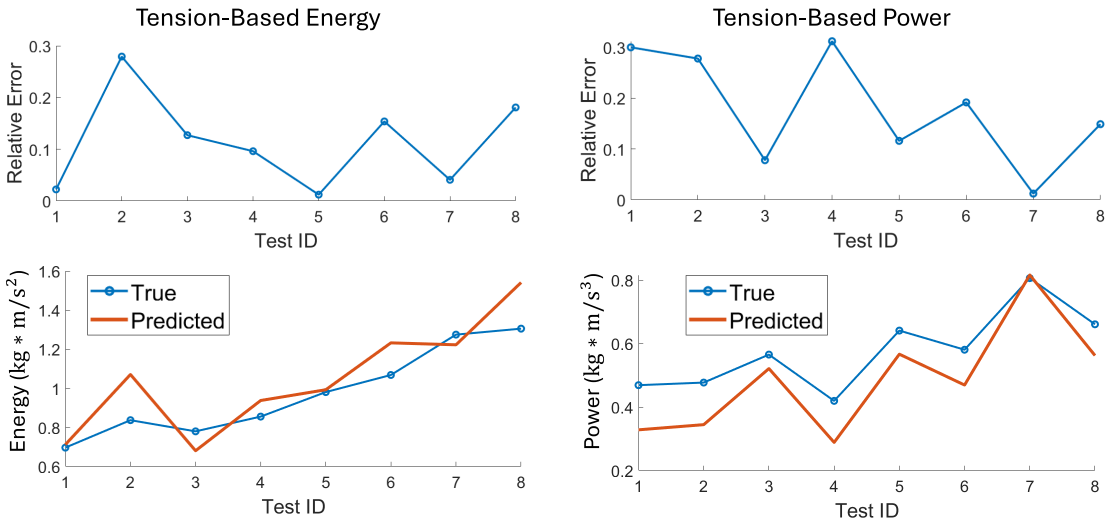


Figure 5.6: Top: Surrogate accuracy for tension-based energy (left) and power (right) on unseen data. Bottom: Predicted versus true energy (left) and power (right). Test IDs correspond to Table 5.1.

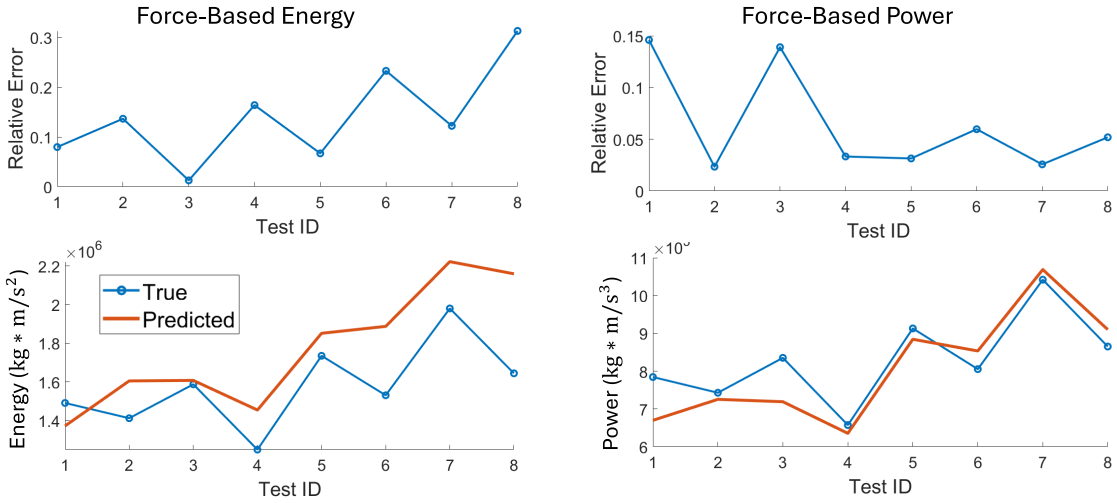


Figure 5.7: Top: Surrogate accuracy for force-based energy (left) and power (right) on unseen data. Bottom: Predicted versus true energy (left) and power (right). Test IDs correspond to Table 5.1.

5.2 Updated Surrogate Model

Using the new simulation data, we updated the surrogate models for each output quantity. Table 5.3 summarizes the MSE and optimized kernel hyperparameters for the updated surrogates. The MSE associated with kinematic motion and magnitude of vorticity decreases slightly with the expanded training set, while the MSE for the remaining outputs remains similar to the original surrogates. These results indicate that the underlying functional relationships are preserved, with minor improvements in select outputs.

Result being Modeled	**MSE**	ARD Kernel	σ_o	α_{RQ}
Kinematic Motion– X	0.0084	Rational Quadratic	0.2077	2.33
Kinematic Motion– Z	0.0066	Rational Quadratic	0.2674	10.13
Magnitude of Vorticity	0.0077	Matern 5/2	0.2422	-
Vertical Flow in V_A	5.15×10^{-5}	Rational Quadratic	0.1609	1.50
Vertical Flow in V_B	1.29×10^{-5}	Rational Quadratic	0.1894	5.52
Tension-based Energy	6.85×10^{-4}	Rational Quadratic	0.4432	8.87
Tension-based Power	2.79×10^{-5}	Rational Quadratic	0.5722	59.47
Force-based Energy	5.18×10^{-9}	Rational Quadratic	0.7088	0.03
Force-based Power	1.05×10^{-12}	Rational Quadratic	0.7091	0.03

Table 5.3: Summary of GP surrogate model accuracy, including the new training data.

5.2.1 Hyperparameter Interpretation

The optimized kernel parameters are generally consistent with the original models, indicating that underlying functional relationships are preserved. Notable exceptions include:

- An increase in α_{RQ} for the vertical (z) component of kinematic motion;
- A substantial decrease in α_{RQ} for tension-based power;
- An increase in signal variance σ_o for force-based power.

Higher α_{RQ} corresponds to a smoother inferred function, suggesting that additional simulations reduce apparent variability in the vertical kinematics. Conversely, the lower α_{RQ} for tension-based power indicates increased local variability, requiring a less smooth kernel to capture the observed behavior. Finally, the increased σ_o for force-based power reflects broader output variance, consistent with the expanded dataset exposing a wider range of power outputs.

5.2.2 Comparison of Original and Updated Surrogates

To evaluate the effects of the updated surrogates, we compare original and updated models for tension- and force-based energy and power, presented in Figs .5.8–5.9, with Re varied while the muscle parameters are held at their base values. For tension-based metrics, the updated surrogate model closely reproduces predictions from the original model. By contrast, notable differences are observed for the force-based metrics. The energy surrogates align well up to $Re = 512$, after which the updated model decreases while the original remains constant. Additionally, the power predictions now match the observed data trends much better. These changes reflect the increased σ_o for force-based outputs, producing less smooth but more data-specific surrogate models. In essence, these updated surrogates improve accuracy but raise potential concerns about overfitting.

We next compare original and updated surrogates for tentacle kinematics, presented in Fig. 5.10 for an increased muscle length with other parameters at their base, and in Fig. 5.11 for an increased muscle tension (and $\psi = 1.35$) with other parameters at their base. While differences are less pronounced than for the force-based metrics, the updated model does exhibit noticeable changes in predicted tentacle motion. Specifically, for $Re = 1$, the updated surrogates produce more oscillatory trajectories, resulting in paths that are less smooth and biologically unrealistic. This behavior appears to result from the increase in α_{RQ} for the z-component of the kinematic motion. Although α_{RQ} increased from 2 to 10, this change does not substantially improve the fit to the observed data points. The effect is most evident at low Re, where constrained trajectories exaggerate local variations, while at higher Re, predictions remain smooth and accurate.

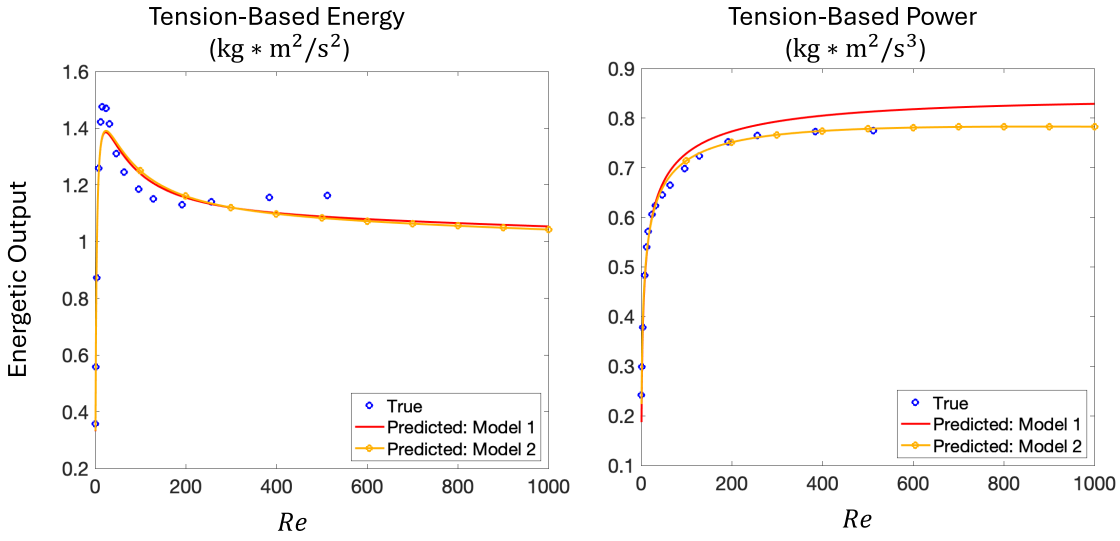


Figure 5.8: True versus predicted tension-based energy (left) and power (right) for varying Re, with muscle parameters held at base values, using the original (Model 1) and updated (Model 2) surrogate models.

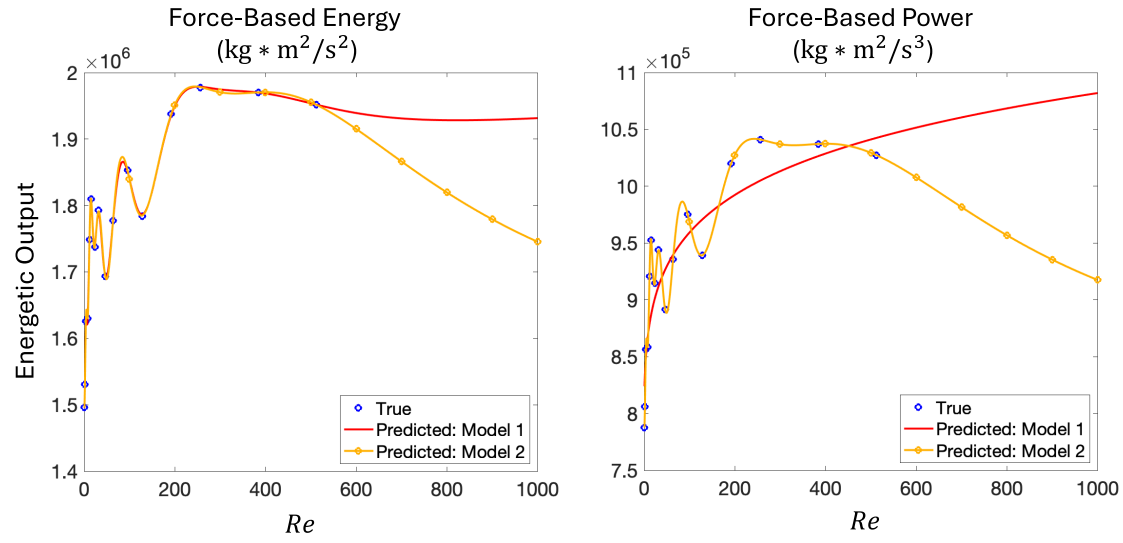


Figure 5.9: True versus predicted force-based energy (left) and power (right) for varying Re, with muscle parameters held at base values, using the original (Model 1) and updated (Model 2) surrogate models.

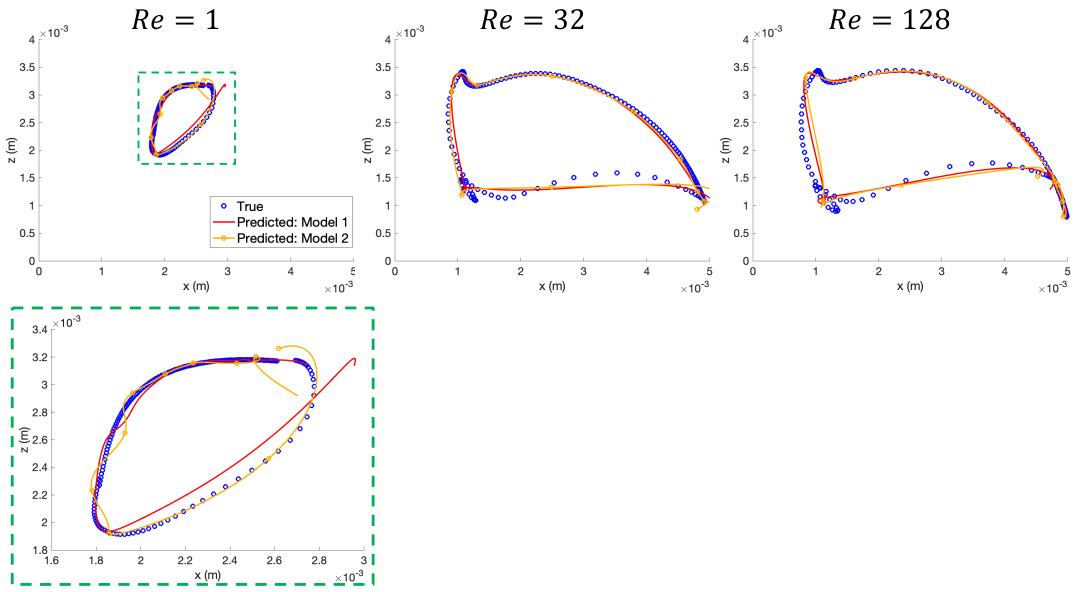


Figure 5.10: Top: True versus predicted tentacle kinematics for $Re = 1$ (left), $Re = 32$ (middle), and $Re = 128$ (right) with $1.5 * L_e$ and other parameters held at base, using the original (Model 1) and updated (Model 2) surrogate models.. Bottom: Zoom-in of $Re = 1$.

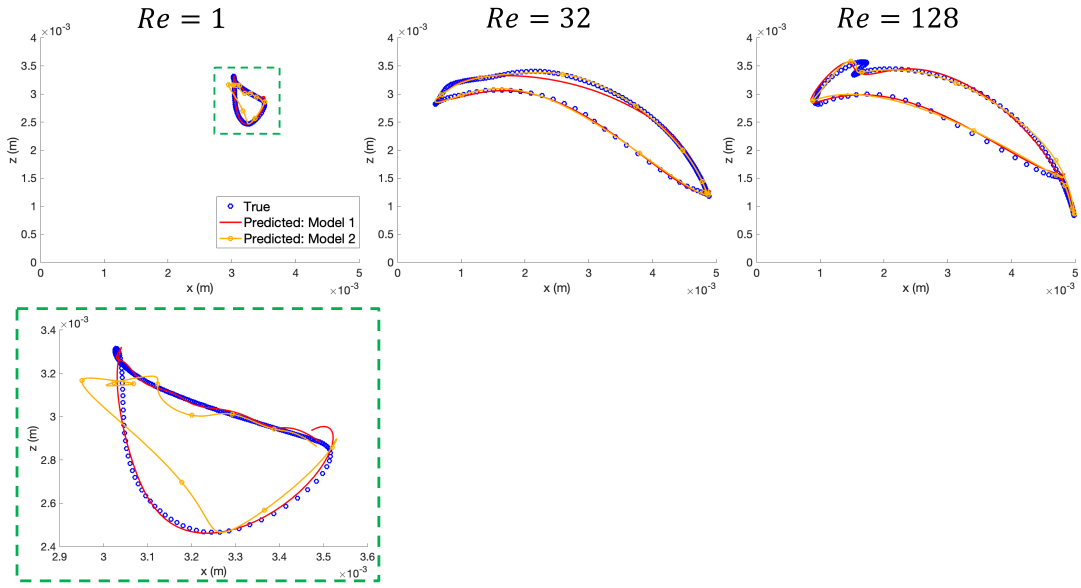


Figure 5.11: Top: True versus predicted tentacle kinematics for $Re = 1$ (left), $Re = 32$ (middle), and $Re = 128$ (right) with $1.5 * T_{\max}$, $\psi = 1.35$, and other parameters held at base, using the original (Model 1) and updated (Model 2) surrogate models.. Bottom: Zoom-in of $Re = 1$.

Finally, it should be noted that changes in the surrogate model can arise even when the hyperparameters are only slightly adjusted, as is the case for the magnitude of vorticity. Because vorticity was modeled using a Matern 5/2 kernel, there is no α_{RQ} parameter, however, the signal variance, σ_o, increased slightly from 0.17 to 0.22 between the original and updated models. Despite this modest change, Fig. 5.12 shows that the updated surrogate does not reproduce the same peak vorticity at $Re = 128$ as the original model. The increased σ_o, though small, slightly shifted the models predictions in sparsely sampled regions, highlighting the sensitivity of surrogate outputs to both hyperparameter and data changes.

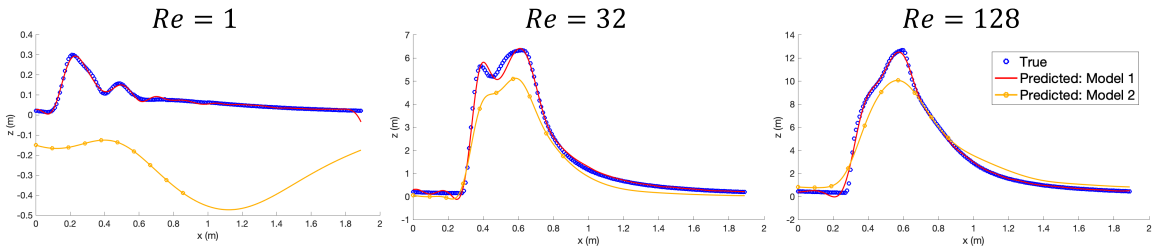


Figure 5.12: True versus predicted magnitude of vorticity for $Re = 1$ (left), $Re = 32$ (middle), and $Re = 128$ (right) with $1.5 * T_{\max}$, $\psi = 0.9$, and other parameters held at base.

5.3 Optimization of Updated Surrogates

Using the updated surrogate models, we repeat the multi-objective optimization procedure. The goal remains to identify parameter combinations that maximize fluid transport while minimizing energetic cost to the coral polyp. Because the updated surrogates are refined in regions associated with optimal performance, modest shifts in the predicted optima are expected.

We compare the same four optimization methods introduced in Chapter 4: generalized pattern search, Nelder-Mead, gradient descent (with an approximate gradient), and trust region methods (with approximate gradient and Hessian). In the original optimization, pattern search and gradient descent were the most effective approaches; the present study evaluates whether this conclusion holds after surrogate refinement.

Optimization is performed using augmented Chebyshev scalarizations that (i) maximize vorticity while minimizing tension-based energy, (ii) maximize vorticity while minimizing tension-based power, (iii) maximize vertical flow in V_A while minimizing tension-based energy, and (iv) maximize vertical flow in V_A while minimizing tension-based power. Each objective pairing is evaluated for $\omega = 0.25$ and $\omega = 0.65$, corresponding to the parameter regimes targeted by the additional high-fidelity simulations. A summary of the resulting optimal parameters is provided in Table 5.4, with detailed results in Appendix F.

Best Method	Maximized Parameter	Minimized Parameter	ω	o_1	o_2	Re	$T_{\max}$ (N)	ψ	L_e
Pattern Search	Vorticity	Energy	0.25	12.52	0.48	1,000	5,187	0.25	0.36
Gradient Descent	Vorticity	Power	0.25	8.65	0.34	367	2,829	0.76	0.42
Gradient Descent	Vertical Flow in V_A	Energy	0.25	0.005	0.62	183	4,439	0.94	0.34
Gradient Descent	Vertical Flow in V_A	Power	0.25	0.005	0.32	196	3,702	1.32	0.32
Gradient Descent	Vorticity	Energy	0.65	17.32	0.87	544	4,525	0.79	0.43
Gradient Descent	Vorticity	Power	0.65	17.11	0.47	852	3,310	1.09	0.44
Pattern Search	Vertical Flow in V_A	Energy	0.65	0.009	1.36	567	6,206	1.18	0.42
Pattern Search	Vertical Flow in V_A	Power	0.65	0.008	0.71	141	4,904	1.35	0.44

Table 5.4: Updated simulation parameters based on iterated optimization results. Note that energy and power each refer to the tension-based definitions.

Across all updated optimization runs, gradient descent and pattern search consistently outperform Nelder-Mead and trust region methods. Although the trust region method performed best for one objective pairing in the original optimization, it is substantially more computationally expensive and does not justify its use in the refined surrogate settings. Consequently, Nelder-Mead and trust region methods are not recommended for future optimization rounds.

While the refined optimization produces shifts in several optimal parameters, compared to Tables. 4.7–4.10, certain trends persist. Muscle length remains stable, with $L_e \approx 0.4$, and the optimal Reynolds number stays within the same range, except when maximizing vorticity while minimizing tension-based energy at $\omega = 0.25$ and minimizing tension-based power at $\omega = 0.65$. Values of $T_{\max}$ also remain stable, with a single exception for maximizing vorticity when minimizing tension-based power at $\omega = 0.25$. Notably, the optimal tension–elasticity ratio decreases substantially across all cases, indicating that the optimization now favors more elastic regimes.

Among the updated results, the optimal parameters obtained when maximizing vorticity while minimizing energy warrant closer examination for two reasons. First, the optimization problem appears to lack a unique solution: multiple parameter combinations can produce comparably optimal objective values. In this case, the refined optimization identifies an optimal parameter set that differs substantially from the original solution. The composite landscape is sensitive to changes in surrogate data, which can guide the optimization toward a different, yet similarly optimal, region of parameter space. Second, the feasibility of the resulting parameters must be considered. For example, with $T_{\max} = 5,187$ and $\psi = 0.25$, the resulting tentacle elasticity $\eta_t = 20,748$ may prevent the simulation from reaching steady state or require computationally infeasible runtimes. This suggests that future optimization rounds may require additional constraints to exclude parameter regimes that are not practically realizable.

Overall, these shifts highlight that changes in surrogate training data can alter predicted optima or guide the optimization toward different, but similarly effective, regions of parameter space. The trends observed- stable parameters, occasional outliers, and sensitivity to surrogate updates, support careful interpretation of optimal solutions rather than assuming strict uniqueness

5.4 Conclusion and Next Steps

Moving forward, the iterative surrogate refinement and optimization procedure can be repeated until the optimal parameter sets converge, producing surrogate models increasingly focused on regions of parameter space most relevant to optimal performance. This process simultaneously refines estimates of the fluid and muscle parameters that maximize transport efficiency subject to energetic constraints.

The refinement process can also highlight additional parameters that merit further exploration. For example, muscle length rapidly converged to a consistent value across cases, suggesting a robust optimum, whereas the tension-elasticity ratio exhibited substantial variation. Targeted additional simulations that vary these sensitive parameters could further clarify their influence on the objective space.

Overall, this chapter demonstrates an iterative surrogate-optimization pipeline provides a practical framework for efficiently exploring complex parameter spaces, guiding both computational and experimental investigations toward regions of high performance.

Chapter 6

Conclusion

This dissertation examined the fluid flow generated by pulsing corals through numerical simulations, statistical modeling, and optimization. The central goal was to determine whether pulsing soft corals operate in regimes that efficiently drive fluid transport and to identify parameter combinations that maximize flow while accounting for energetic costs.

In Chapter 2, we presented numerical simulations and conducted a parameter sweep across fluid and muscle parameters governing coral polyp kinematics and the resulting flow. We found that increasing parameters such as muscle tension, muscle length, and Reynolds number generally enhanced fluid transport. However, these increases also entail higher energetic costs for the coral polyp, highlighting an inherent trade-off between maximizing flow and minimizing energy expenditure. Computational expense further limited the number of parameter combinations that could be directly analyzed, motivating the need for surrogate modeling.

Chapter 3 introduced Gaussian process regression to address the computational limitations of directly simulating many parameter combinations. Surrogate models mapped fluid and muscle parameters to kinematic and flow outputs, enabling efficient exploration of trends across the parameter space. This predictive framework allowed us to infer general patterns in the complex system and prepare the data for targeted optimization.

Building on this, Chapter 4 applied optimization techniques to these surrogate models. Single- and multi-objective optimization identified parameter combinations that maximize fluid transport or minimize energetic cost, as well as those that provide a balanced trade-off between the two objectives.

Chapter 5 further refined this approach through iterative modeling. Optimized parameter sets were used to generate new simulations, which in turn produced updated surrogate models. This iterative framework allows for progressively more efficient sampling of the parameter space, focusing computational resources on regions that produce results of greatest interest.

Our results indicate that biological corals largely occupy near-optimal regions of parameter space. In particular, the optimal maximum muscle tension closely matched the baseline values observed in natural polyps, and muscle length rapidly converged to its

biological value during optimization. These results suggest that the physical and muscular properties of coral polyps are well tuned to promote efficient fluid transport and mixing. The most notable deviation from baseline occurred in the Reynolds number: higher Re values were associated with increased fluid transport, indicating potential for enhanced flow under alternative environmental conditions.

Overall, these findings suggest that coral polyps operate in regimes that promote effective fluid mixing. However, the evolutionary mechanisms underlying these parameter choices remain uncertain, and it is not clear whether corals evolved specifically to optimize fluid transport or whether these properties arise from other biological constraints.

The insights gained from studying pulsing corals also have practical implications beyond the original biological motivation. For example, the iterative surrogate-optimization framework developed here could guide the design of biomimetic systems that mimic coral kinematics to enhance fluid transport. By leveraging the trade-offs between transport efficiency and energetic cost revealed in the coral system, engineers could create artificial systems that achieve robust performance with minimal energy input. More generally, the methods and workflow of this dissertation, combining numerical simulations, surrogate modeling, and iterative optimization, are applicable to other computationally expensive problems, enabling efficient exploration of complex parameter spaces and informed decision-making in both biological and engineering contexts. In the context of pulsing corals, this work contributes to a growing understanding of how organisms interact with their surrounding fluid environments.

Bibliography

[1]M. Aburidi and R. Marcia, "Optimal Transport Based Graph Kernels for Drug Property Prediction", IEEE Open Journal of Engineering in Medicine and Biology **6**, 152–157 (2025).

[2]M. Aburidi and R. Marcia, "Wasserstein Distance-Based Graph Kernel for Enhancing Drug Safety and Efficacy Prediction", in 2024 IEEE First International Conference on Artificial Intelligence for Medicine, Health and Care (AIMHC) (Feb. 2024), pp. 113–119.

[3]L. Armijo, "Minimization of Functions Having Lipschitz Continuous First Partial Derivatives", Pacific Journal of Mathematics **16**, 1–3 (1966).

[4]A. Batta, B. S. Kalra, and R. Khirasaria, "Trends in FDA Drug Approvals Over Last 2 Decades: An Observational Study", en-US, Journal of Family Medicine and Primary Care **9**, 105 (2020).

[5]N. A. Battista, J. E. Samson, S. Khatri, and L. A. Miller, "Under the Sea: Pulsing Corals in Ambient Flow", Texts in Biomathematics **1**, arXiv:1709.05058 [physics, q-bio], 22 (2017).

[6]R. P. Beyer, "A Computational Model of the Cochlea Using the Immersed Boundary Method", Journal of Computational Physics **98**, 145–162 (1992).

[7]D. Boffi, L. Gastaldi, L. Heltai, and C. S. Peskin, "On the Hyper-Elastic Formulation of the Immersed Boundary Method", Computer Methods in Applied Mechanics and Engineering, Immersed Boundary Method and Its Extensions **197**, 2210–2231 (2008).

[8]K.-Y. Chen, K.-A. Feng, Y. Kim, and M.-C. Lai, "A Note on Pressure Accuracy in Immersed Boundary Method for Stokes Flow", Journal of Computational Physics **230**, 4377–4383 (2011).

[9]F. Cheng, I. A. Kovács, and A.-L. Barabási, "Network-Based Prediction of Drug Combinations", en, Nature Communications **10**, Number: 1, 1197 (2019).

[10]H. Childs, E. Brugger, B. Whitlock, J. Meredith, S. Ahern, D. Pugmire, K. Biagas, M. C. Miller, C. Harrison, G. H. Weber, H. Krishnan, T. Fogal, A. Sanderson, C. Garth, E. W. Bethel, D. Camp, O. Rubel, M. Durant, J. M. Favre, and P. Navratil, *High Performance VisualizationEnabling Extreme-Scale Scientific Insight*, 2012.

[11]T. Chugh, “Scalarizing Functions in Bayesian Multiobjective Optimization”, in 2020 IEEE Congress on Evolutionary Computation (CEC) (July 2020), pp. 1–8.

[12]A. R. Conn, N. I. M. Gould, and P. L. Toint, *Trust-Region Methods* (Society for Industrial and Applied Mathematics, USA, June 2000).

[13]H. B. Curry, “The Method of Steepest Descent for Non-Linear Minimization Problems”, en, Quarterly of Applied Mathematics **2**, 258–261 (1944).

[14]J. J. Darrow, J. Avorn, and A. S. Kesselheim, “FDA Approval and Regulation of Pharmaceuticals, 1983-2018”, en, JAMA **323**, 164 (2020).

[15]R. S. Dembo and T. Steihaug, “Truncated-Newton Algorithms for Large-Scale Unconstrained Optimization”, en, Mathematical Programming **26**, 190–212 (1983).

[16]J. E. Dennis and R. B. Schnabel, “Globally Convergent Modifications of Newton’s Method”, in *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Classics in Applied Mathematics (Society for Industrial and Applied Mathematics, Jan. 1996), pp. 111–154.

[17]W. C. Dennison and D. J. Barnes, “Effect of Water Motion on Coral Photosynthesis and Calcification”, Journal of Experimental Marine Biology and Ecology **115**, 67–77 (1988).

[18]D. Devendran and C. S. Peskin, “An Immersed Boundary Energy-Based Method for Incompressible Viscoelasticity”, Journal of Computational Physics **231**, 4613–4642 (2012).

[19]K. Fabricius and D. Klumpp, “Widespread Mixotrophy in Reef-Inhabiting Soft Corals: The Influence of Depth, and Colony Expansion and Contraction on Photosynthesis”, en, Marine Ecology Progress Series **125**, 195–204 (1995).

[20]L. J. Fauci and A. McDonald, “Sperm Motility in the Presence of Boundaries”, eng, Bulletin of Mathematical Biology **57**, 679–699 (1995).

[21]L. J. Fauci and C. S. Peskin, “A Computational Model of Aquatic Animal Locomotion”, Journal of Computational Physics **77**, 85–108 (1988).

[22]W. Fitt and C. Cook, “The Effects of Feeding or Addition of Dissolved Inorganic Nutrients in Maintaining the Symbiosis Between Dinoflagellates and a Tropical Marine Cnidarian”, en, Marine Biology **139**, 507–517 (2001).

[23]M. Frank and P. Wolfe, “An Algorithm for Quadratic Programming”, en, Naval Research Logistics Quarterly **3**, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/nav.3800030109, 95–110 (1956).

[24]Y. Gabay, Y. Benayahu, and M. Fine, “Does Elevated pCO2 Affect Reef Octocorals?”, en, Ecology and Evolution **3**, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ece3.351, 465–473 (2013).

[25]M. G. Genton, “Classes of Kernels for Machine Learning: A Statistics Perspective”, en, Journal of Machine Learning Research **2**, 299–312 (2001).

[26]A. J. Gil, A. Arranz Carreño, J. Bonet, and O. Hassan, "An Enhanced Immersed Structural Potential Method for FluidStructure Interaction", Journal of Computational Physics **250**, 178–205 (2013).

[27]A. J. Gil, A. Arranz Carreño, J. Bonet, and O. Hassan, "The Immersed Structural Potential Method for Haemodynamic Applications", Journal of Computational Physics **229**, 8613–8641 (2010).

[28]H. Gohar, "Studies on the Xeniidae of the Red Sea: Their Ecology, Physiology, Taxonomy, and Phylogeny", Publications of the Marine Biological Station, Ghardaqa, (Red Sea) **2** (1940).

[29]H. Gohar and H. M. Roushdy, "On the Physiology of the Neuromuscular System of Heteroxenia (Alcyonaria)", Publications of the Marine Biological Station Ghardaqa **10**, 91–144 (1959).

[30]H. Gohar and H. Roushdy, "The Neuromuscular System of the Xeniidae (Alcyonaria) I. Histological", Publ. Mar. Biol. Stat. Ghardaqa **10**, 63–81 (1959).

[31]A. Gottlieb, G. Y. Stein, Y. Oron, E. Ruppin, and R. Sharan, "INDI: A Computational Framework for Inferring Drug Interactions and Their Associated Recommendations", Molecular Systems Biology **8**, 592 (2012).

[32]B. E. Griffith, "An Accurate and Efficient Method for the Incompressible NavierStokes Equations Using the Projection Method as a Preconditioner", en, Journal of Computational Physics **228**, 7565–7595 (2009).

[33]B. E. Griffith, "On the Volume Conservation of the Immersed Boundary Method", en, Communications in Computational Physics **12**, 401–432 (2012).

[34]B. E. Griffith, R. D. Hornung, D. M. McQueen, and C. S. Peskin, "An Adaptive, Formally Second Order Accurate Version of the Immersed Boundary Method", en, Journal of Computational Physics **223**, 10–49 (2007).

[35]B. E. Griffith and X. Luo, "Hybrid Finite Difference/Finite Element Immersed Boundary Method", eng, International Journal for Numerical Methods in Biomedical Engineering **33**, 10.1002/cnm.2888 (2017).

[36]B. E. Griffith, X. Luo, D. M. McQueen, and C. S. Peskin, "Simulating the Fluid Dynamics of Natural and Prosthetic Heart Values Using the Immersed Boundary Method", en, International Journal of Applied Mechanics, 10.1142/S1758825109000113 (2012).

[37]B. T. N. Gunney, "Scalable Mesh Management for Patch-Based AMR", en, (2012).

[38]L. Heltai and F. Costanzo, "Variational Implementation of Immersed Finite Element Methods", Computer Methods in Applied Mechanics and Engineering **229-232**, 110–127 (2012).

[39]G. Herschlag and L. Miller, "Reynolds Number Limits for Jet Propulsion: A Numerical Study of Simplified Jellyfish", Journal of Theoretical Biology **285**, 84–95 (2011).

[40]C. E. L. Hill, S. G. Abbass, G. Caporale, Y. C. ElKhaled, L. Kuhn, T. Schlenzig, C. Wild, and A. Tilstra, "Physiology of the Widespread Pulsating Soft Coral Xenia Umbellata is Affected by Food Sources, but Not by Water Flow", Ecology and Evolution **13**, e10483 (2023).

[41]R. Hooke and T. A. Jeeves, ""Direct Search" Solution of Numerical and Statistical Problems", en, Journal of the ACM **8**, 212–229 (1961).

[42]A. Hoover and L. Miller, "A Numerical Study of the Benefits of Driving Jellyfish Bells at Their Natural Frequency", en, Journal of Theoretical Biology **374**, 13–25 (2015).

[43]A. P. Hoover, J. Daniels, J. C. Nawroth, and K. Katija, "A Computational Model for Tail Undulation and Fluid Transport in the Giant Larvacean", en, Fluids **6**, 88 (2021).

[44]A. P. Hoover, B. E. Griffith, and L. A. Miller, "Quantifying Performance in the Medusan Mechanospace with an Actively Swimming Three-Dimensional Jellyfish Model", en, Journal of Fluid Mechanics **813**, 1112–1155 (2017).

[45]A. P. Hoover, A. J. Porras, and L. A. Miller, "Pump or Coast: The Role of Resonance and Passive Energy Recapture in Medusan Swimming Performance", en, Journal of Fluid Mechanics **863**, 1031–1061 (2019).

[46]M. Hua and C. S. Peskin, "An Analysis of the Numerical Stability of the Immersed Boundary Method", Journal of Computational Physics **467**, 111435 (2022).

[47]*IBAMR: An Adaptive and Distributed-Memory Parallel Implementation of the Immersed Boundary Method*.

[48]S. Inoue, H. Kayanne, S. Yamamoto, and H. Kurihara, "Spatial Community Shift From Hard to Soft Corals in Acidified Water", en, Nature Climate Change **3**, 683–687 (2013).

[49]S. K. Jones, R. Laurenza, T. L. Hedrick, B. E. Griffith, and L. A. Miller, "Lift vs. Drag Based Mechanisms for Vertical Force Production in the Smallest Flying Insects", Journal of Theoretical Biology **384**, 105–120 (2015).

[50]S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang, and E. E. Bolton, "PubChem 2023 update", Nucleic Acids Research **51**, D1373–D1380 (2023).

[51]Y. Kim and C. S. Peskin, "Penalty Immersed Boundary Method for an Elastic Boundary with Mass", Physics of Fluids **19**, 053103 (2007).

[52]A. Kirzinger, A. Montero, G. Sparks, I. Valdes, and L. H. Published, *Public Opinion on Prescription Drugs and Their Prices*, en-US, Aug. 2023.

[53]J. Knowles, "ParEGO: A Hybrid Algorithm with On-Line Landscape Approximation for Expensive Multiobjective Optimization Problems", IEEE Transactions on Evolutionary Computation **10**, 50–66 (2006).

[54]M. Kremien, U. Shavit, T. Mass, and A. Genin, "Benefit of Pulsation in Soft Corals", Proceedings of the National Academy of Sciences **110**, 8978–8983 (2013).

[55]M. Kuhn, I. Letunic, L. J. Jensen, and P. Bork, "The SIDER Database of Drugs and Side Effects", eng, Nucleic Acids Research **44**, D1075–1079 (2016).

[56]L. Muscatine and James W. Porter, “Reef Corals: Mutualistic Symbioses Adapted to Nutrient-Poor Environments”, BioScience **27** (1977).

[57]M.-C. Lai and C. S. Peskin, “An Immersed Boundary Method with Formal Second-Order Accuracy and Reduced Numerical Viscosity”, Journal of Computational Physics **160**, 705–719 (2000).

[58]J. B. Lewis, “Feeding Behaviour and Feeding Ecology of the Octocorallia (Coelenterata: Anthozoa)”, Journal of Zoology **196**, 371–384 (1982).

[59]P. Li, C. Huang, Y. Fu, J. Wang, Z. Wu, J. Ru, C. Zheng, Z. Guo, X. Chen, W. Zhou, W. Zhang, Y. Li, J. Chen, A. Lu, and Y. Wang, “Large-Scale Exploration and Analysis of Drug Combinations”, eng, Bioinformatics (Oxford, England) **31**, 2007–2016 (2015).

[60]H. Liu, Y.-S. Ong, X. Shen, and J. Cai, “When Gaussian Process Meets Big Data: A Review of Scalable GPs”, IEEE Transactions on Neural Networks and Learning Systems **31**, 4405–4423 (2020).

[61]W. K. Liu, Y. Liu, D. Farrell, L. Zhang, X. S. Wang, Y. Fukui, N. Patankar, Y. Zhang, C. Bajaj, J. Lee, J. Hong, X. Chen, and H. Hsu, “Immersed Finite Element Method and Its Applications to Biological Systems”, Computer Methods in Applied Mechanics and Engineering, A Tribute to Thomas J.R. Hughes on the Occasion of his 60th Birthday **195**, 1722–1749 (2006).

[62]D. J. C. Mackay, “Introduction to Gaussian Processes”, en, (1998).

[63]D. J. C. MacKay, “Bayesian Non-Linear Modeling for the Prediction Competition”, en, in *Maximum Entropy and Bayesian Methods: Santa Barbara, California, U.S.A., 1993*, edited by G. R. Heidbreder (Springer Netherlands, Dordrecht, 1996), pp. 221–234.

[64]S. Malone, M. Aburidi, and R. F. Marcia, “Triple Matrix Factorization for Drug-Drug Interaction Prediction Using Fused Gromov-Wasserstein Distances”, in 2025 47th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC), ISSN: 2694-0604 (July 2025), pp. 1–6.

[65]S. Malone, M. Aburidi, and R. F. Marcia, “Wasserstein-Based Similarity Constrained Matrix Factorization for Drug-Drug Interaction Prediction”, in 2024 IEEE Workshop on Signal Processing Systems (SiPS), ISSN: 2374-7390 (Nov. 2024), pp. 49–53.

[66]S. Malone, R. F. Marcia, and S. Khatri, “Gaussian Processes as a Surrogate for Fluid Flow Analysis of Soft Corals”, in OCEANS 2025 - Great Lakes, ISSN: 2996-1882 (Sept. 2025), pp. 1–8.

[67]D. Malul, R. Holzman, and U. Shavit, “Spatio-Temporal Patterns of the Incoming Water Flow in Pulsating Corals”, en, Journal of Experimental Biology **228**, jeb250262 (2025).

[68]T. Mass, A. Genin, U. Shavit, M. Grinstein, and D. Tchernov, “Flow Enhances Photosynthesis in Marine Benthic Autotrophs by Increasing the Efflux of Oxygen from the Organism to the Water”, Proceedings of the National Academy of Sciences **107**, 2527–2531 (2010).

[69]G. Matheron, “The Intrinsic Random Functions and Their Applications”, en, Advances in Applied Probability **5**, 439–468 (1973).

[70]G. Matheron, “Principles of Geostatistics”, Economic Geology **58**, 1246–1266 (1963).

[71]D. G. Maucieri and J. K. Baum, “Impacts of Heat Stress on Soft Corals, An Overlooked and Highly Vulnerable Component of Coral Reef Ecosystems, At a Central Equatorial Pacific Atoll”, Biological Conservation **262**, 109328 (2021).

[72]M. D. McKay, R. J. Beckman, and W. J. Conover, “A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code”, Technometrics **21**, 239–245 (1979).

[73]D. M. McQueen and C. S. Peskin, “Heart Simulation by an Immersed Boundary Method with Formal Second-order Accuracy and Reduced Numerical Viscosity”, en, in Mechanics for a New Mellennium, edited by H. Aref and J. W. Phillips (2002), pp. 429–444.

[74]F. Mémoli, “GromovWasserstein Distances and the Metric Approach to Object Matching”, Foundations of Computational Mathematics **11**, 417–487 (2011).

[75]L. A. Miller and C. S. Peskin, “A Computational Fluid Dynamics of ‘Clap and Fling’ in the Smallest Insects”, Journal of Experimental Biology **208**, 195–212 (2005).

[76]Y. Mori and C. S. Peskin, “Implicit Second-Order Immersed Boundary Methods with Boundary Mass”, Computer Methods in Applied Mechanics and Engineering, Immersed Boundary Method and Its Extensions **197**, 2049–2067 (2008).

[77]J. A. Nelder and R. Mead, “A Simplex Method for Function Minimization”, The Computer Journal **7**, 308–313 (1965).

[78]J. Nocedal and S. J. Wright, *Numerical Optimization*, en, Second edition, Springer series in operations research and financial engineering (Springer, New York, NY, 2006).

[79]A. O’Hagan, “Curve Fitting and Optimal Design for Prediction”, Journal of the Royal Statistical Society. Series B (Methodological) **40**, 1–42 (1978).

[80]C. O. Pacherres, S. Ahmerkamp, K. Koren, C. Richter, and M. Holtappels, “Ciliary Flows in Corals Ventilate Target Areas of High Photosynthetic Oxygen Production”, Current Biology **32**, 4150–4158.e3 (2022).

[81]C. O. Pacherres, S. Ahmerkamp, G. M. Schmidt-Grieb, M. Holtappels, and C. Richter, “Ciliary Vortex Flows and Oxygen Dynamics in the Coral Boundary Layer”, en, Scientific Reports **10**, 7541 (2020).

[82]B. Percha and R. B. Altman, “Informatics Confronts DrugDrug Interactions”, Trends in Pharmacological Sciences **34**, 178–184 (2013).

[83]C. S. Peskin, “Flow Patterns Around Heart Valves: A Numerical Method”, Journal of Computational Physics **10**, 252–271 (1972).

[84]C. S. Peskin, “Numerical Analysis of Blood Flow in the Heart”, Journal of Computational Physics **25**, 220–252 (1977).

[85]C. S. Peskin, “The Immersed Boundary Method”, en, Acta Numerica **11**, 479–517 (2002).

[86]M. J. D. Powell, “A New Algorithm for Unconstrained Optimization”, in *Nonlinear Programming*, edited by J. B. Rosen, O. L. Mangasarian, and K. Ritter (Academic Press, Jan. 1970), pp. 31–65.

[87]J. Quiñonero-Candela and C. E. Rasmussen, “A Unifying View of Sparse Approximate Gaussian Process Regression”, Journal of Machine Learning Research **6**, 1939–1959 (2005).

[88]C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*, en, 3. print, Adaptive computation and machine learning (MIT Press, Cambridge, Mass., 2008).

[89]H. H. Rosenbrock, “An Automatic Method for Finding the Greatest or Least Value of a Function”, The Computer Journal **3**, _eprint: https://academic.oup.com/comjnl/article-pdf/3/3/175/988633/030175.pdf, 175–184 (1960).

[90]S. Roy, L. Heltai, and F. Costanzo, “Benchmarking the Immersed Finite Element Method for FluidStructure Interaction Problems”, Computers & Mathematics with Applications **69**, 1167–1188 (2015).

[91]J. E. Samson, N. A. Battista, S. Khatri, and L. A. Miller, “Pulsing Corals: A Story of Scale and Mixing”, BIOMATH **6**, arXiv:1709.04996 [q-bio], 10.11145/j.biomath.2017.12.169 (2017).

[92]J. E. Samson and L. A. Miller, “Collective Pulsing in Xeniid Corals: Part II-Using Computational Fluid Dynamics to Determine if There are Benefits to Coordinated Pulsing”, eng, Bulletin of Mathematical Biology **82**, 67 (2020).

[93]J. E. Samson, L. A. Miller, D. Ray, R. Holzman, U. Shavit, and S. Khatri, “A Novel Mechanism of Mixing by Pulsing Corals”, en, Journal of Experimental Biology, jeb.192518 (2019).

[94]F. Santambrogio, *Optimal Transport for Applied Mathematicians* (Birkhäuser, 2015).

[95]M. Santiago, “Numerical Methods for Modeling the Fluid Flow of Pulsing Soft Corals and the Photosynthesis of Their Symbiotic Algae”, English, Ph.D. (University of California, Merced, United States – California, 2021).

[96]M. Santiago, A. Hoover, and L. A. Miller, “Emergent Kinematics and Flow Structure of Tension Driven Pulsing Xeniid Corals”, en, Bulletin of Mathematical Biology **87**, 133 (2025).

[97]M. Santiago, K. A. Mitchell, and S. Khatri, “Numerical Method for Modeling Photosynthesis of Algae on Pulsing Soft Corals”, Physical Review Fluids **7**, 033102 (2022).

[98]D. Schlichter, “Nutritional Strategies of Cnidarians: The Absorption, Translocation and Utilization of Dissolved Nutrients by Heteroxenia fuscescens”, American Zoologist **22**, 659–669 (1982).

[99]V. Shankar, G. B. Wright, R. M. Kirby, and A. L. Fogelson, "Augmenting the Immersed Boundary Method with Radial Basis Functions (RBFs) for the Modeling of Platelets in Hemodynamic Flows", en, International Journal for Numerical Methods in Fluids **79**, 536–557 (2015).

[100]O. H. Shapiro, V. I. Fernandez, M. Garren, J. S. Guasto, F. P. Debaillon-Vesque, E. Kramarsky-Winter, A. Vardi, and R. Stocker, "Vortical Ciliary Flows Actively Enhance Mass Transport in Reef Corals", Proceedings of the National Academy of Sciences **111**, 13391–13396 (2014).

[101]J.-Y. Shi, A.-Q. Zhang, S.-W. Zhang, K.-T. Mao, and S.-M. Yiu, "A Unified Solution for Different Scenarios of Predicting Drug-Target Interactions via Triple Matrix Factorization", BMC Systems Biology **12**, 136 (2018).

[102]B. W. Silverman, "Some Aspects of the Spline Smoothing Approach to Non-Parametric Regression Curve Fitting", Journal of the Royal Statistical Society: Series B (Methodological) **47**, 1–21 (1985).

[103]A. Smola and P. Bartlett, "Sparse Greedy Gaussian Process Regression", in Advances in Neural Information Processing Systems, Vol. 13 (2000).

[104]E. Snelson and Z. Ghahramani, "Sparse Gaussian Processes using Pseudo-Inputs", in Advances in Neural Information Processing Systems, Vol. 18 (2005).

[105]T. Steihaug, "The Conjugate Gradient Method and Trust Regions in Large Scale Optimization", SIAM Journal on Numerical Analysis **20**, 626–637 (1983).

[106]M. Streamer, Y. R. McNeil, and D. Yellowlees, "Photosynthetic Carbon Dioxide Fixation in Zooxanthellae", en, Marine Biology **115**, 195–198 (1993).

[107]N. P. Tatonetti, P. P. Ye, R. Daneshjou, and R. B. Altman, "Data-Driven Prediction of Drug Effects and Interactions", Science Translational Medicine **4**, 125ra31–125ra31 (2012).

[108]J. M. Teran and C. S. Peskin, "Tether Force Constraints in Stokes Flow by the Immersed Boundary Method on a Periodic Domain", SIAM Journal on Scientific Computing **31**, 3404–3416 (2009).

[109]The Mathworks Inc., *MATLAB version: 9.12.0 (R2022a)*, Natick, Massachusetts, United States, 2022.

[110]V. Titouan, N. Courty, R. Tavenard, C. Laetitia, and R. Flamary, "Optimal Transport for Structured Data with Application on Graphs", in Proceedings of the 36th International Conference on Machine Learning, Vol. 97, edited by K. Chaudhuri and R. Salakhutdinov, Proceedings of Machine Learning Research (June 2019), pp. 6275–6284.

[111]V. Torczon, "On the Convergence of Pattern Search Algorithms", SIAM Journal on Optimization **7**, 1–25 (1997).

[112]G. Tsounis and P. J. Edmunds, "Three Decades of Coral Reef Community Dynamics in St. John, USVI: A Contrast of Scleractinians and Octocorals", en, Ecosphere **8**, _eprint: https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.1002/ecs2.1646, e01646 (2017).

[113]C. Tu and C. S. Peskin, "Stability and Instability in the Computation of Flows with Moving Immersed Boundaries: A Comparison of Three Methods", SIAM Journal on Scientific and Statistical Computing **13**, 1361–1376 (1992).

[114]B. Vadala-Roth, S. Acharya, N. A. Patankar, S. Rossi, and B. E. Griffith, "Stabilization Approaches for the Hyperelastic Immersed Boundary Method for Problems of Large-Deformation Incompressible Elasticity", Computer Methods in Applied Mechanics and Engineering **365**, 112978 (2020).

[115]S. Vilar, R. Harpaz, E. Uriarte, L. Santana, R. Rabadan, and C. Friedman, "DrugDrug Interaction Through Molecular Structure Similarity Analysis", Journal of the American Medical Informatics Association : JAMIA **19**, 1066–1074 (2012).

[116]S. Vilar, E. Uriarte, L. Santana, N. P. Tatonetti, and C. Friedman, "Detection of Drug-Drug Interactions by Modeling Interaction Profile Fingerprints", en, PLOS ONE **8**, e58321 (2013).

[117]X. Wang and L. T. Zhang, "Interpolation Functions in the Immersed Boundary and Finite Element Methods", en, Computational Mechanics **45**, 321–334 (2010).

[118]D. Weininger, "SMILES, a Chemical Language and Information System. 1. Introduction to Methodology and Encoding Rules", Journal of Chemical Information and Computer Sciences **28**, 31–36 (1988).

[119]D. Wells, B. Vadala-Roth, J. H. Lee, and B. E. Griffith, "A Nodal Immersed Finite Element-Finite Difference Method", Journal of computational physics **477**, 111890 (2023).

[120]D. S. Wishart, Y. D. Feunang, A. C. Guo, E. J. Lo, A. Marcu, J. R. Grant, T. Sajed, D. Johnson, C. Li, Z. Sayeeda, N. Assempour, I. Iynkkaran, Y. Liu, A. Maciejewski, N. Gale, A. Wilson, L. Chin, R. Cummings, D. Le, A. Pon, C. Knox, and M. Wilson, "DrugBank 5.0: A Major Update to the DrugBank Database for 2018", eng, Nucleic Acids Research **46**, D1074–D1082 (2018).

[121]P. Wolfe, "Convergence Conditions for Ascent Methods", SIAM Review **11**, 226–235 (1969).

[122]P. Wolfe, "Convergence Conditions for Ascent Methods. II: Some Corrections", SIAM Review **13**, 185–188 (1971).

[123]M. K. Wood and G. B. Dantzig, "Programming of Interdependent Activities: I General Discussion", Econometrica **17**, 193–199 (1949).

[124]M. H. Wright, "Nelder, Mead, and the Other Simplex Method", en, in *Documenta Mathematica Series*, Vol. 6, 1st ed. (EMS Press, Jan. 2012), pp. 271–276.

[125]L. T. Zhang and M. Gay, "Immersed Finite Element Method for Fluid-Structure Interactions", Journal of Fluids and Structures **23**, 839–857 (2007).

[126]L. Zhang, A. Gerstenberger, X. Wang, and W. K. Liu, "Immersed Finite Element Method", Computer Methods in Applied Mechanics and Engineering, Flow Simulation and Modeling **193**, 2051–2067 (2004).

[127]W. Zhang, Y. Chen, D. Li, and X. Yue, "Manifold Regularized Matrix Factorization for Drug-Drug Interaction Prediction", Journal of Biomedical Informatics **88**, 90–97 (2018).

[128]W. Zhang, Y. Chen, F. Liu, F. Luo, G. Tian, and X. Li, "Predicting Potential Drug-Drug Interactions by Integrating Chemical, Biological, Phenotypic and Network Data", BMC Bioinformatics **18**, 18 (2017).

Appendix A

Gaussian Processes as a Surrogate for Fluid Flow Analysis of Soft Corals

The following is a copy of a previously published paper which supports this dissertation work [66].

A.1 Introduction

Soft corals in the Xeniidae family, seen modeled in Fig. A.1, have the unique characteristic of pulsing in a rhythmic manner to generate fluid flow around themselves. This energetically expensive motion is not for locomotion, leading researchers to question its motive. The prevailing theory is that this is for the benefit of the endosymbiotic algae which live inside the coral's cells and produce energy for the coral polyp through photosynthesis. Through this process, the algae take in carbon dioxide rich water, producing both energy for the algae and coral polyp, as well as oxygen, a waste product of photosynthesis. The accumulation of oxygen near the coral polyp can lead to a decrease in photosynthesis [68], however, by pulsing, the coral polyp enhances the efflux of oxygen rich water. This has been supported by several studies that found that photosynthesis is increased when fluid flow is present [17, 54]. Together, these studies support the theory that the pulsing motion of xeniid corals is derived to best support the coral ecosystem in which it exists.

In addition to these biological studies, several numerical studies have been completed with the goal of better understanding this pulsing motion [5, 91, 92, 93, 96, 97]. Each of these has supported the hypothesis that these corals evolved to circulate flow in a way that is advantageous to themselves and their symbionts.

One downside to computing such numerical studies is the computational cost required to solve for the fluid flow and to analyze the subsequent fluid flow and mixing. This makes it difficult to gain a broad understanding of the impact of the various kinematic, flow, and muscle parameters that determine this system. Therefore, we propose the introduction of Gaussian processes, which can learn the underlying flow dynamics from a relatively small

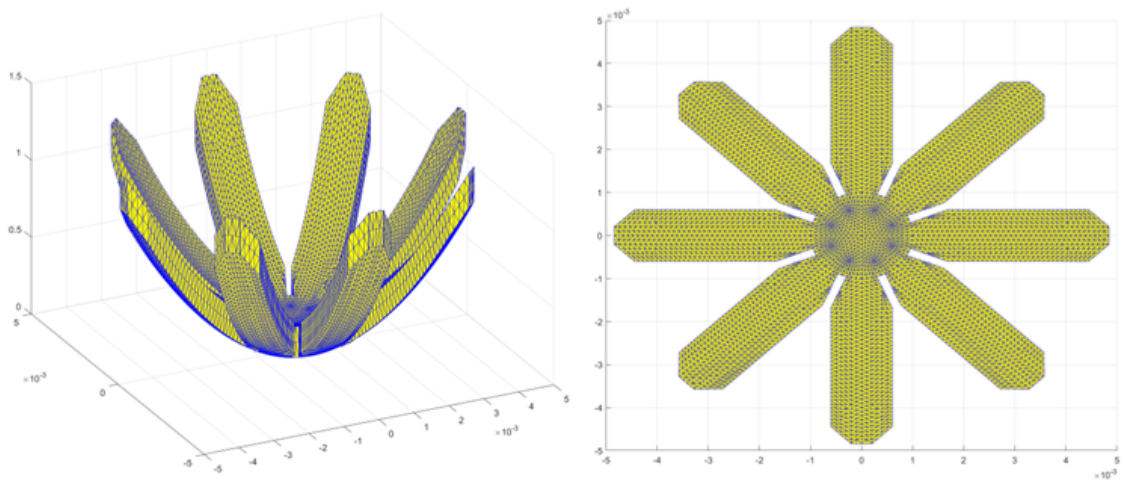


Figure A.1: Model of xeniid pulsing coral at contraction shown from different angles.

set of parameter variations [62, 88]. Through this application, we can better understand the conditions under which the most fluid flow and mixing are achieved without incurring a high computational cost.

A.2 Methods

A.2.1 Model Equations and Numerical Methods

The immersed boundary method [83] is a widely used framework for studying fluid-structure interactions, as it allows problems to be formulated using an Eulerian grid for the fluid and a moving Lagrangian grid for the structure. Since its introduction, many extensions and alternate versions of this method have been developed, including the hybrid finite difference / finite element immersed boundary method (IBFE) [35]. IBFE utilizes a finite element discretization of the structure's Lagrangian framework and a finite difference discretization of the fluid's Eulerian framework. This is an improvement over the standard immersed boundary method, as it allows for coarser Lagrangian meshes relative to the Eulerian mesh, while ensuring that the structure remains "watertight" throughout the structure's deformation, preventing numerical and discretization errors [35].

The Eulerian fluid domain has positions, $\vec{x} \in \Omega$, at which the fluid velocity and pressure are solved where Ω is the fluid domain. IBFE makes use of a reference configuration for the Lagrangian structure. The Lagrangian grid is defined as $\vec{X} \in S$, where $\vec{X}$ is a grid point on the resting configuration. To arrive at the current configuration, $\vec{\chi}(\vec{X}, t)$, we apply the deformation gradient $\mathbb{F}$, which allows us to find the physical Lagrangian position in time t. This set-up is illustrated in Fig. A.2.

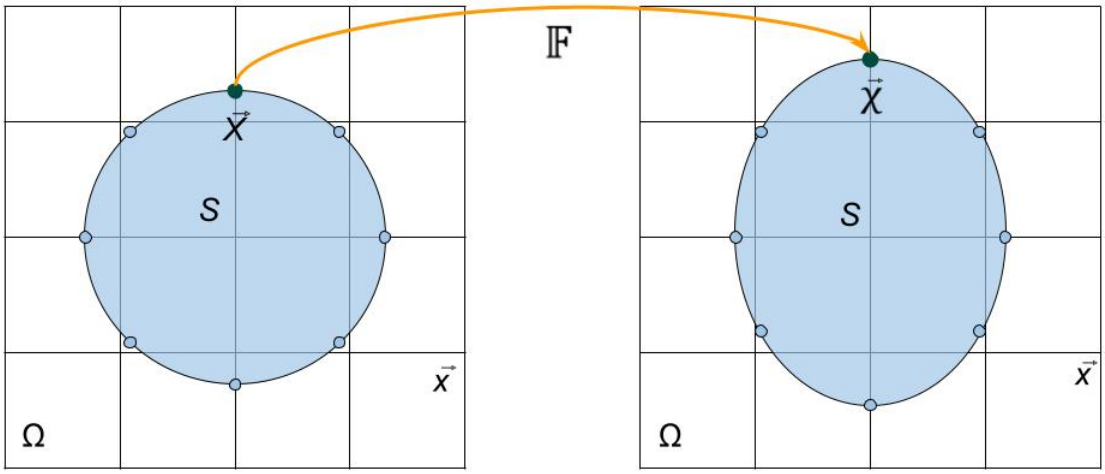


Figure A.2: Immersed boundary material configuration. Here, Ω is the Eulerian grid while S is the Lagrangian grid. The structure is deformed through the deformation gradient $\mathbb{F}$, allowing the structure to move from its initial configuration $\vec{X}$ to its current configuration $\vec{\chi}$. Redrawn from Santiago [95].

The IBFE method is a numerical method that solves a system of equations for the movement of the coral and resulting fluid flow in a coupled manner. The model system includes the Lagrangian force of the coral on the fluid given in weak form by

$$\begin{aligned}\int_S \vec{F}(\vec{X},t)\cdot\vec{\phi}(\vec{X})d\vec{X} = &-\int_S \mathbb{P}(\vec{X},t):\nabla_{\vec{X}}\vec{X}d\vec{X}\\ &+\int_S \vec{F}_T(\vec{X},t)\cdot\vec{\phi}(\vec{X})d\vec{X}\end{aligned} \tag{A.1}$$

where $\mathbb{P}$ is the first Piola-Kirchhoff (PK1) stress tensor, $\vec{F}_T$ is the tether force holding our structure in place, $\vec{\phi}(\vec{X},t)$ is a test function, and : is the double dot product and ∇_x is the gradient with respect to $\vec{X}$ [35, 96]. Details on $\mathbb{P}$ and $\vec{F}_T$ will be given below in Section A.2.2 [96].

The model also includes interaction equations which are used to convert between the Lagrangian and Eulerian frameworks. First, the Lagrangian force applied by the immersed structure onto the fluid is spread to the Eulerian fluid grid by

$$\vec{f}(\vec{x},t) = \int_S \vec{F}(\vec{X},t)\delta\left(\vec{x}-\vec{\chi}(\vec{X},t)\right)d\vec{X} \tag{A.2}$$

where $\vec{F}(\vec{X},t)$ is the Lagrangian force solved for in Eq. ((A.1)) being spread to the Cartesian grid and $\delta(\cdot)$ is the Dirac delta function.

The Navier-Stokes equations are solved for on the Eulerian grid using this force, given

by

$$\rho\left(\frac{\partial\vec{u}}{\partial t}+\vec{u}\cdot\nabla\vec{u}\right)+\nabla p=\mu\Delta\vec{u}+\vec{f},$$
$$\nabla\cdot\vec{u}=0, \tag{A.3}$$

Here, $\vec{u}(\vec{x},t)$ is the fluid velocity, $\vec{f}$ is the force applied to the fluid by the immersed structure, $p(\vec{x},t)$ is the fluid pressure, ρ is the fluid density, and μ is the fluid viscosity. This introduces a key advantage to this method, namely, we only solve for the fluid velocity and pressure on the fixed Eulerian grid. A key parameter for these equations is the Reynolds number, Re, which is the ratio of inertial to viscous forces,

$$Re=\frac{\rho\pi L_T^2}{2\mu T_c} \tag{A.4}$$

where L_T is the tentacle length and T_c is the contraction time. For the biologically relevant coral, we define $Re = 46.5892$ to be our base value with other relevant base parameters seen in Tab A.1. Initially, this sweep varied Re along $Re = 2^k, k = \{0, 1, 2, \ldots, 9\}$, in addition to the base case. To ensure sufficient resolution for GP regression, additional points were added, resulting in a final dataset of 16 simulations: Re = [1, 2, 4, 8, 12, 16, 24, 32, 36, 46.5892, 64, 96, 128, 192, 256, 512].

Parameter	**Variable**	**Units**	**Base Value**
Domain Size	D	m	0.0992
Fluid Density	ρ	kgm$^-$3	1029
Fluid Dynamic Viscosity	μ	kgms$^-$1	0.00108
Tentacle Length	L_T	m	0.0041
Contraction Period	T_c	s	0.54
Expansion Period	T_e	s	0.71
Resting Period	T_r	s	0.64
Pulsing Period	T_{per}	s	1.89
Reynolds Number	Re	-	46.5892

Table A.1: Base values for pulsing coral simulations

The second of our interaction equations interpolates the fluid velocity to the structure,

$$\begin{aligned}\frac{\partial\vec{\chi}(\vec{X},t)}{\partial t}&=\vec{U}(\vec{X},t)\\&=\int_\Omega\vec{u}(\vec{x},t)\delta\left(\vec{x}-\vec{\chi}(\vec{X},t)\right)d\vec{x},\end{aligned} \tag{A.5}$$

where $\vec{U}(\vec{X},t)$ is the velocity of the structure. This allows the structure to move as a result of the updated fluid velocity.

The IBFE method temporally and spatially discretizes this system of equations. We utilize the Immersed Boundary Method Adaptive Mesh Refinement (IBAMR) software

to solve this discretization numerically [34, 35, 47]. These simulations were executed on the High Performance Computing system at UC Merced, with early iterations using the MERCED cluster and newer simulations using the Pinnacles cluster. On this system, simulations at the base Re take approximately 2 days to reach a periodic steady state. This is defined as when the fluid flow exhibits a consistent behavior across successive pulses and was determined by examining the horizontal and vertical velocities of a simulation in a cylinder near the coral. Steady state was typically achieved by the 15th pulse although higher Re simulations required more pulses to converge.

Once the simulations reached steady state, fluid flow analyses were completed using the visualization software, VisIt [10]. These results were then imported to MATLAB where GP regression was computed using the built-in function `fitrgp` [109].

A.2.2 Kinematics

We employ a tension-based mechanistic model to mimic the musculature of the coral polyp, which was developed in [96]. This model incorporates the Piola-Kirchoff stress tensor,

$$\mathbb{P} = \mathbb{P}_e + \mathbb{P}_t, \tag{A.6}$$

which is a summation of the passive elasticity $\mathbb{P}_e$ and the active tension $\mathbb{P}_t$.

The passive elasticity, $\mathbb{P}_e$, determines the elastic forces of our structure, guiding its return to the resting configuration. This is described by a neo-Hookean material model

$$\mathbb{P}_e = \eta J^{-2/3}\left(\mathbb{F} - \frac{I_1}{3}\mathbb{F}^{-\top}\right) + \beta \ln(J)\mathbb{F}^{-\top}. \tag{A.7}$$

Here, $\mathbb{F} = \partial\vec{\chi}/\partial\vec{X}$ is the deformation tensor, $J = \det(\mathbb{F})$ is the determinant of the deformation tensor, and $I_1 = \text{tr}(\mathbb{F}^\top\mathbb{F})$ is a tensor invariant of the right Cauchy-Green tensor. Additionally, the shear elastic modulus, η, is defined in relation to the material parameters, the Young's modulus, E, and Poisson's ratio, $\nu = 0.4$:

$$\eta = \frac{E}{2(1+\nu)}. \tag{A.8}$$

We will define η below as varying over the coral tentacles. Lastly, the term $\beta \ln(J)\mathbb{F}^{-\top}$ in Eq. (A.7) is a volume stabilization term to reinforce incompressibility and improve the volume conservation of the material, where β is the bulk density [114], given by

$$\beta = \frac{2\eta(1+\nu)}{3(1-2\nu)}. \tag{A.9}$$

The active tension, $\mathbb{P}_t$, is used to model the musculature of the coral polyp, which we compute as follows. First, we define the muscle fiber direction

$$\vec{f}l_0 = \left(-X_1, \frac{R_t(\vec{X})}{X_2 - 0.01}, -X_3\right), \tag{A.10}$$

where $R_t(\vec{X})$ is a scalar determining how far along the tentacle a point $\vec{X}$ is

$$R_t(\vec{X}) = \sqrt{X_1^2 + X_3^2}. \tag{A.11}$$

We let $\vec{fl}_0$ be the unit vector in the direction of $\vec{fl}_0$, which allows for proper orientation of the muscles, making it so that the coral pulls in an inward direction towards the oral disc [29, 30]. Finally, we define the active tension as

$$\mathbb{P}_t = JT\mathbb{F}\vec{fl}_0 \text{tr}\Big(\vec{fl}_0\Big), \tag{A.12}$$

where T is the magnitude of the tension and $\text{tr}\Big(\vec{fl}_0\Big)$ is the trace of the muscle fiber direction.

To account for the varied elasticity that would be found along the coral tentacle, we vary the shear elastic modulus, η, over the tentacle domain,

$$\eta = \eta_p + \Big(\eta_t - \eta_p l_\eta(R_t) \max\Big(\tau(\vec{X}), s(R_t)\Big)\Big), \tag{A.13}$$

where η_p is the elasticity applied to the edges of the tentacle model, representing the pinnule region, and η_t is the elasticity applied to the center of the tentacle. This allows for the model to be most stiff in the center of the tentacle where the stalk is located, and most flexible at the edges where the pinnules are located. The elasticity between these two regions is then determined by two terms: (i) the elasticity $s(\vec{X})$ in the region between the oral disk and the tentacle:

$$s(\vec{X}) = 1 - \frac{1}{1 + \exp\Big(-\lambda_s(R_t(\vec{X}) - r_o)\Big)}, \tag{A.14}$$

where λ_s is a smoothing parameter and r_o is the radius of the oral disk, and (ii) the elasticity $\tau(\vec{X})$ along the width of the tentacle:

$$\begin{aligned}\tau(\vec{X}) = &\frac{-1}{1 + \exp\Big(\lambda_\tau(D(\vec{X}) - e_\tau)\Big)} \\ &+ \frac{1}{1 + \exp\Big(\lambda_\tau(D(\vec{X}) + e_\tau)\Big)},\end{aligned} \tag{A.15}$$

where λ_τ is another smoothing parameter, e_τ is the end of the central region in regards to the tentacle width, and $D(\vec{X})$ is the distance from the middle of the central region of the tentacle in regards to tentacle width. Note that when we say width, we are referring to the thickness of the tentacles.

We also include an elasticity $l_\eta(\vec{X})$ along the length of the tentacle:

$$l_\eta(\vec{X}) = \begin{cases} 1 & R_t(\vec{X}) < 0.01 \\ l_{\eta_s}(\vec{X}) & 0.01 \leq R_t(\vec{X}) < e_l \\ 0 & e_l < R_t(\vec{X}) \end{cases}, \tag{A.16}$$

where $e_l = 0.9L_T + r_o$ is the length along which the tentacle starts to recede into the pinnule region with L_T being the tentacle length and

$$l_{\eta_s}(\vec{X}) = \frac{1}{2} + \frac{1}{2} * \cos\left(\frac{\pi}{0.9L_T}\left(R_t(\vec{X}) - r_o\right)\right) \tag{A.17}$$

is the ramp down through the stalk.

Note that this representation of η in Eq. (A.13) serves as an approximation to the definition given in Eq. (A.8).

Next, we introduce the muscle tension, $T(\vec{X}, t)$, given by

$$T(\vec{X}, t) = T_{\max} a(t) m(\vec{X}) \tag{A.18}$$

where $T_{\max}$ is the maximum tension, $a(t)$ is an activation function for the tension, and $m(\vec{X})$ is the muscle location function corresponding to where the tension is applied [29, 30]. The activation function

$$\begin{aligned} a(t) = A_1 \Bigg(& \frac{1}{1 + \exp\left(-A_a(t_p - 0.15)\right)} \\ & - \frac{1}{1 + \exp\left(-A_d(t_p - 0.4)\right)} \Bigg) \end{aligned} \tag{A.19}$$

determines how the tension is applied along the muscles as time varies and is responsible for the resulting periodic pulsing behavior of the coral polyp. Here, A_a and A_d are scalars that control the ramp-up and ramp-down, A_1 is a scalar such that $\max\left(a(t)\right) = 1$, and $t_p = t \bmod T_{\text{per}}$ which ensures that the activation function is periodic within the desired pulsing period. The muscle location function, $m(\vec{X})$, is the location where the tension is then applied and is given by

$$m(\vec{X}) = h(\vec{X}) l_m(\vec{X}) \tau(\vec{X}) \tag{A.20}$$

where

$$h(\vec{X}) = \frac{1}{1 + \exp\left(-\lambda_c R_c(\vec{X})\right)} \tag{A.21}$$

determines if a point is on the top or bottom of a tentacle, with λ_c a smoothing parameter and

$$R_c(\vec{X}) = 0.0099 - \sqrt{x_1^2 + (x_2 - 0.01)^2 + X_3^2}, \tag{A.22}$$

and

$$\begin{aligned} l_m(\vec{X}) = & \frac{1}{1 + \exp\left(-\lambda_b\left(R_t(\vec{X}) - b_m\right)\right)} \\ & + \frac{1}{1 + \exp\left(-\lambda_c\left(R_t(\vec{X}) - e_m\right)\right)}, \end{aligned} \tag{A.23}$$

determines the length of the muscle, where $b_m = 0.05L_T + r_o$ and $e_m = 0.45L_T + r_o$ are scalars that define the start and end of the muscle in relation to the tentacle length and λ_b

and λ_e are smoothing parameters for the transition between the oral disc and the muscle and the muscle and the end of the tentacle.

This model results in a kinematic motion similar to that of biological corals [93]. Motion starts with contraction, where the muscles tense and drive the tentacle in an inward and upward direction. Once fully contracted, expansion begins where the muscles relax and elasticity returns the coral to its resting configuration. Finally, the polyp enters its resting phase where the muscles do not move, resulting in no active motion.

A summary of the parameters used to generate the biologically relevant base case of this muscle model can be seen in Tab. A.2.

Parameter	Value	Units
r_s	0.01	m^{-1}
r_o	0.001	m^{-1}
η_{min}	20	Pa
η_{max}	5000	Pa
T_{max}	4500	N
e_τ	0.0002	m^{-1}
λ_τ	50000	m^{-1}
λ_σ	10000	m^{-1}
λ_c	100000	m^{-1}
λ_b	4000	m^{-1}
λ_e	12000	m^{-1}
A_1	$\frac{1}{0.935}$	-
A_a	40	s^{-1}
A_d	20	s^{-1}

Table A.2: Parameters used to define the tension-driven muscle coral model's motion [96].

A.2.3 Kinematic & Flow Analysis

Kinematic Comparison

A comparison of the motion of the corals throughout a pulse is conducted to understand how the movement of the tentacles depends on the coupling of the muscle dynamics and fluid parameters. This is done by tracking the tip of a tentacle throughout a pulse and finding the path it takes. Additionally, we compare the maximum height achieved during contraction and width achieved during expansion to see if the coral polyp is able to fully contract and expand as compared to the biologically relevant case. Note that this analysis is done on a slice in 2-dimensions, as we assume there is symmetry in the coral polyp as it is pulsing.

Vertical Flux

Vertical flux is quantified by finding the average vertical velocity, u_y, in a cylinder placed above the coral polyp, as seen in Fig. A.3. Previous studies used boxes to compute this metric, however we have chosen to utilize cylinders as they more closely mirror the morphology of the coral polyp and the flow regions generated by its pulsing. We compute this metric in four total regions which each have a height of L_T meters but vary in their radius and location as follows: (A) directly on top of the fully contracted coral polyp with a radius 0.002 meters (B) directly on top of region A with a radius of 0.002 meters (C) directly on top of the fully contracted coral polyp with a radius of 0.003509 meters (D) directly on top of region C with a radius of 0.003509 meters. The value of the radii for these regions have been chosen as they closely approximate the radius of the coral model when fully contracted and expanded.

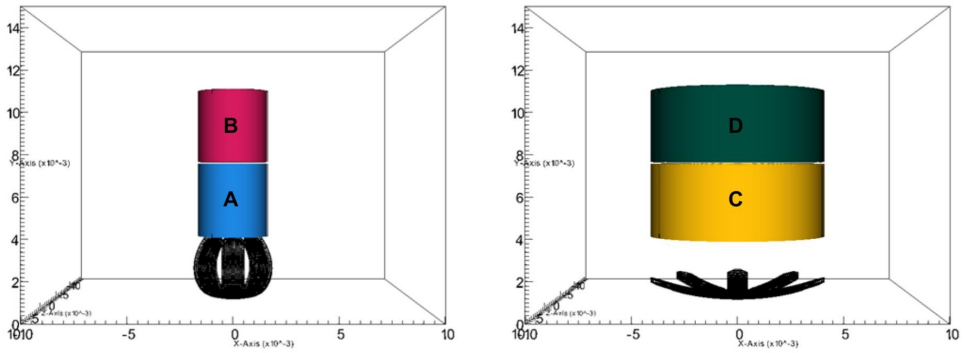


Figure A.3: Regions in which average vertical flux are computed.

Previous studies of corals using biological parameters found the peak in vertical flux to occur just before the end of the contraction phase, followed by a steep drop-off during the expansion phase [91, 92, 93]. Additionally, it has been shown that the vertical flux for a live coral polyp, as opposed to a simulated one, is found to always be positive in planes above the coral polyp, meaning it never draws fluid back down towards it [93]. Therefore, we expect to find similar dynamics for our simulated pulsing coral with biological parameters, referred to as the base case.

A.2.4 Gaussian Processes Regression

Gaussian processes (GPs) are nonparametric statistical models that use the probability distribution of the observed data to make inferences about new data. Intuitively, Gaussian processes define a distribution over possible functions that fit the observed data. Instead of outputting a single prediction, GPs provide a mean estimate along with a measure of uncertainty, making them especially valuable for probabilistic modeling and decision-making under uncertainty. This tool has been widely used in geostatistics, where it is referred to as kriging [69, 70]. However, GPs have since found applications in many

fields where the goal is to model or infer the distribution of some quantity of interest over a space of input parameters. With this in mind, we hope to use GPs as a surrogate model for the numerical modeling and analyses previously presented.

In the context of our flow analyses (Section A.2.3) we assume the existence of an underlying function $f(Re, t)$ which determines the final fluid flow metric as a function of the Reynolds number and time within the pulse. Using GP regression (GPR), our goal is to infer this function, f, which serves as a surrogate for the previously computed flow analyses.

Given a generic dataset with inputs $\mathbf{X} = [\vec{x_1}, \vec{x_2}, ..., \vec{x_n}]$ and corresponding outputs $\vec{y} = [y_1, y_2, ..., y_n]$ we seek to model the relationship $y = f(\vec{x}) + \epsilon$ where $\epsilon \sim \mathcal{N}(0, \sigma^2)$ represents zero-mean Gaussian noise with variance σ^2. A GP is defined as a distribution,

$$f(\vec{x}) \sim \text{GP}\left(\mathbb{E}[f(\vec{x})], k(\vec{x}, \vec{x}')\right) \tag{A.24}$$

where $\mathbb{E}[f(\vec{x})]$ is the expected value of our function, also known as the mean function, and $k(\vec{x}, \vec{x}') = \mathbb{E}\left[\left(f(\vec{x}) - \mathbb{E}[f(\vec{x})]\right)\left(f(\vec{x}') - \mathbb{E}[f(\vec{x}')]\right)\right]$ is the covariance function, or kernel [88].

To construct a GP model, we start by defining a prior probability distribution,

$$f(\mathbf{X}) \sim \mathcal{N}\left(\mathbb{E}(f(\mathbf{X})), K\right) \tag{A.25}$$

where $K = k(\mathbf{X}, \mathbf{X})$ is the covariance matrix computing using a kernel function. Assuming some observational noise, the covariance of $\vec{y}$ becomes $\text{cov}(\vec{y}) = K(\mathbf{X}, \mathbf{X}) + \sigma^2 I$, with I being the identity matrix. We use the exponential kernel, given by

$$k(\vec{x}, \vec{x}') = \sigma_y^2 \exp\left(-\frac{||\vec{x} - \vec{x}'||_2}{\ell}\right) \tag{A.26}$$

with σ_y^2 being the standard deviation of the output y while ℓ is the length scale of the input space. In practice, the hyper-parameters σ_y^2 and ℓ are learned from the data by minimizing the cross-validation loss.

Finally, we form a joint probability distribution

$$\begin{bmatrix} \vec{y} \\ \vec{y}^* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mathbb{E}(f(\mathbf{X})) \\ \mathbb{E}(f(\mathbf{X}^*)) \end{bmatrix}, \begin{bmatrix} K_y & K^* \\ (K^*)^\intercal & K^{**} \end{bmatrix}\right) \tag{A.27}$$

where $\mathbf{X}^*$ is a test input point, y^* is the resulting test output value, $K = k(\mathbf{X}, \mathbf{X})$, $K^* = k(\mathbf{X}, \mathbf{X}^*)$, $K^{**} = k(\mathbf{X}^*, \mathbf{X}^*)$, and $K_y = K + \sigma_n^2 I$. From this joint distribution, we can compute the posterior predictive distribution [79, 88] for the test outputs

$$f^*|\mathbf{X}, \vec{y}, \mathbf{X}^* \sim \mathcal{N}(\bar{y}^*, \text{cov}(y^*)), \tag{A.28}$$

where

$$\bar{y}^* = \mathbb{E}(f(\mathbf{X}^*)) + (K^*)^\intercal K_y^{-1}\left(\vec{y} - \mathbb{E}((\mathbf{X}))\right), \tag{A.29}$$

$$\text{cov}(y^*) = K^{**} - (K^*)^\intercal K_y^{-1} K^*. \tag{A.30}$$

In practice, it is often difficult to specify a meaningful mean function as given in Eq. (A.29). Instead, a more flexible approach involves incorporating basis functions. Doing so alters our problem formulation:

$$g(\mathbf{X}) = f(\mathbf{X}) + h(\mathbf{X})^{\intercal}\beta, \tag{A.31}$$

$$f(\mathbf{X}) \sim \mathrm{GP}\left(0, k(\mathbf{X}, \mathbf{X}')\right), \tag{A.32}$$

where $h(\mathbf{X})$ is a set of fixed basis functions and β is a basis function coefficient. We will specifically be using a constant basis function, meaning that $h(\mathbf{X}) = 1$ while $\beta \in \mathbb{R}$ is a constant which offsets the model since we now assume a zero-mean GP.

It should be noted that a standard limitation of GPs is their computational complexity, being $\mathcal{O}(n^3)$, due to the inverse of the covariance matrix, K_y^{-1}. A common work-around is the use of sparse approximations of the covariance matrix [60].

A.3 Results & Discussion

A.3.1 Kinematic Comparison

When tracking the tentacle tip, we find that the range of kinematic motion increases as the Reynolds number increases, as seen in Fig. A.4. Additionally, the path the tentacle takes changes as the Reynolds number varies. For Reynolds 16 and below, the tentacle moves in a figure-eight like motion, while the larger Reynolds number simulations generate more of a circular path which never crosses itself.

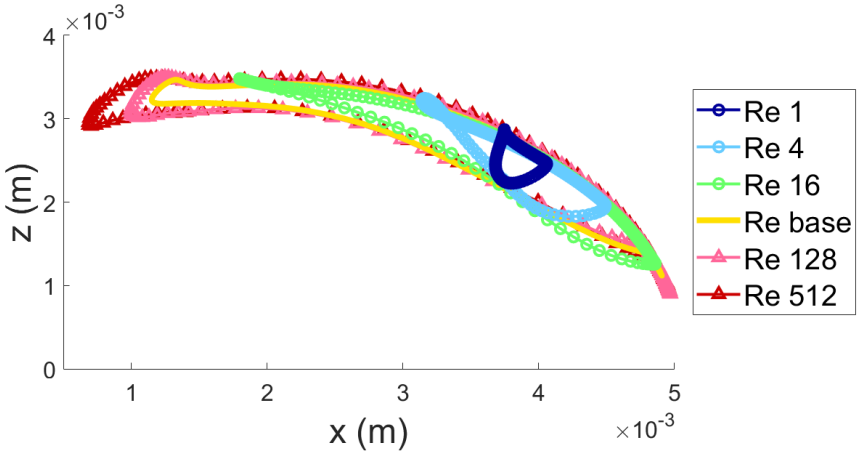


Figure A.4: Path of tentacle tip over the course of a pulse. Cooler colors represent lower Reynolds numbers and warmers colors represent higher Reynolds numbers, with the base Reynolds of 46.5892 being an intermediate value.

We can further quantify the tentacle motion by plotting the maximum height and width achieved during a pulse, which can be seen in Fig. A.5. Here, the maximum height achieved occurs as contraction is ending, while the maximum width occurs as expansion is ending.

We see a rapid ramp-up in both the height and width as the Reynolds number increases from one to sixteen. This indicates that this may be a transition point where the muscles are able to overcome the viscous forces to complete a fuller pulsing motion. Interestingly, the maximum height dips a bit after this, likely a result of the change in pulsing motion from the figure-eight motion to the rounder path.

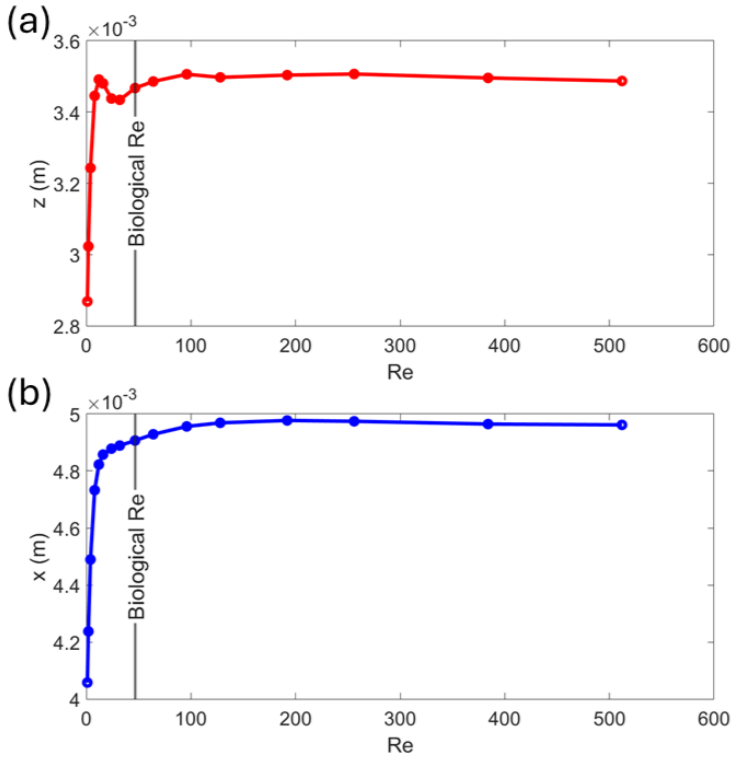


Figure A.5: Maximum (a) height and (b) width achieved during a pulse. Note that the maximum height is achieved near the end of contraction, while the maximum width is achieved near the end of expansion.

This phenomenon can be modeled through a GP model where Reynolds and time are our independent variables and the X and Y of the tentale tip are our dependent variables. We de-couple the X and Y position for this modeling, but can re-couple them after the fact to plot the predicted tentacle tip path. The uncoupled version of both the inputs and outputs can be seen in Fig. A.6. When re-coupled, it can be seen that we accurately recreate the tentacle paths that were originally generated, with the modeled outputs seen in Fig. A.7 matching nearly exactly to Fig. A.4. We can confirm that the GPR model is working accurately as we have a mean squared error of 1.71×10^{-17} when modeling the x position and 6.966×10^{-14} when modeling the y position.

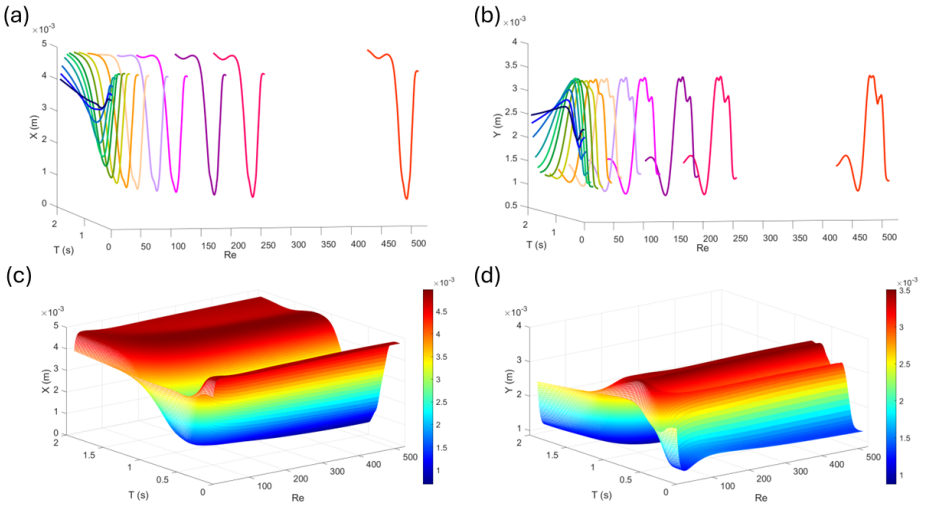


Figure A.6: Inputs and outputs of GPR model for tentacle tip path. (a) X position input to model (b) Y position input to model (c) X position output by model (d) Y position output by model.

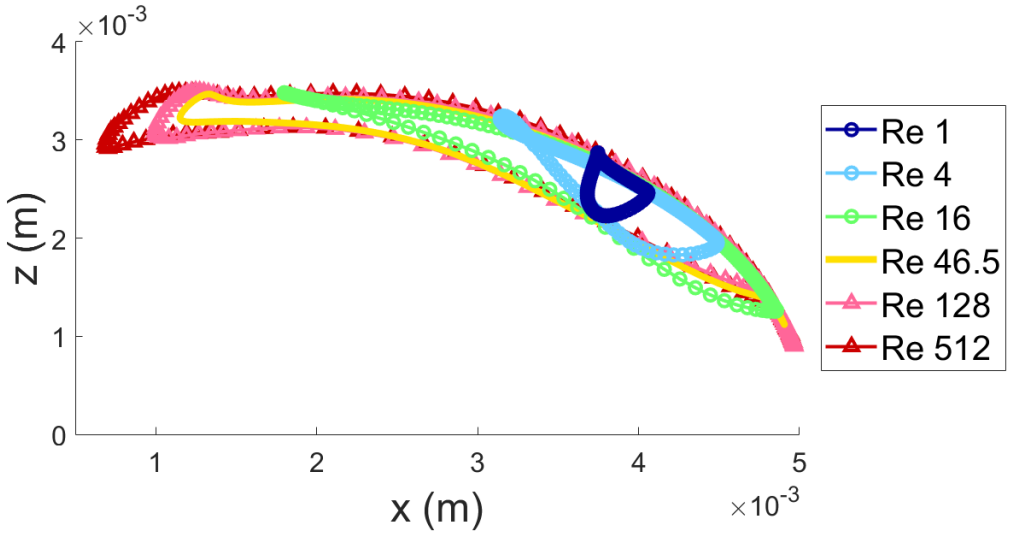


Figure A.7: Path of tentacle tip over the course of a pulse, as modeled using the GPR. Cooler colors represent lower Reynolds numbers and warmers colors represent higher Reynolds numbers, with the base Reynolds being closest modeled by a Reynolds value of 46.5.

A.3.2 Vertical Flux

The average vertical flux for region A can be seen in Fig. A.8, where cooler colors signify lower Reynolds numbers and warmer colors represent higher Reynolds numbers. Additionally, analysis can be broken down into the three phases of motion: contraction, expansion, and rest. Looking first at contraction, there is a monotonic increase in the vertical flux as the Reynolds number increases. During expansion, vertical flux decreases such that the biological value dips just below zero, whereas the lower Reynolds numbers dip into negative vertical flux and the higher Reynolds numbers maintain a positive vertical flux. Finally during rest, vertical flux stabilizes around zero. It is believed that this pattern achieved by the biological polyps is what allows the coral to generate enough upward flow to push oxygen rich water away during contraction, but then having flow settle down so that new carbon dioxide rich water can enter the region near the tentacles during expansion and rest [93].

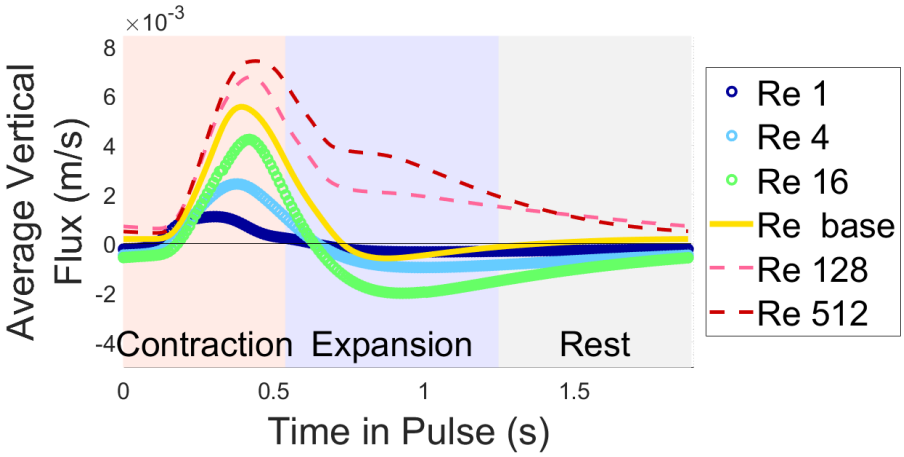


Figure A.8: Vertical flux (m/s) averaged over region A for one pulse. Cooler colors represent lower Reynolds numbers and warmers colors represent higher Reynolds numbers, with the base Reynolds of 46.5892 being an intermediate value.

Note that this analysis was similarly completed for the other regions outlined in of interest and can be made available upon request.

Similarly to the GP modeling of the kinematic motion, we create a GP model where the Reynolds number and time are our independent variables, and the vertical flux for our regions of interest are our dependent variables which we are hoping to recover. The result of this can be seen in Fig. A.9, where we model the input average vertical flux with a mean squared error of 2.7193×10^{-16}.

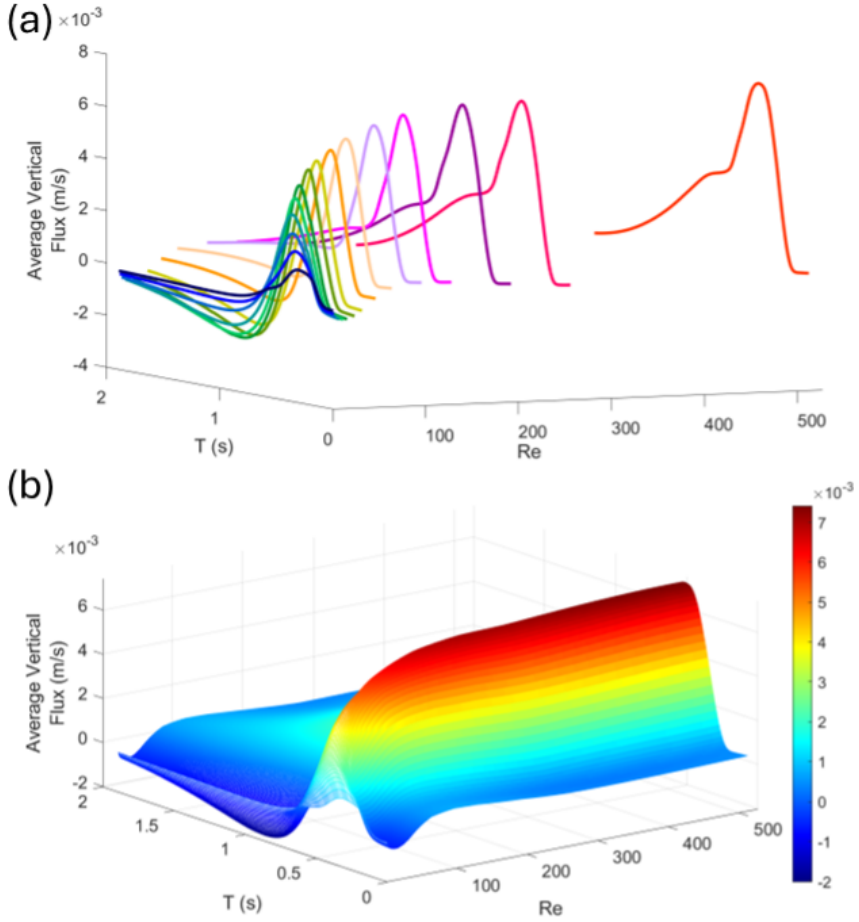


Figure A.9: Inputs and outputs of GPR model for average vertical flux in region A. (a) Input to model (b) Output of model.

A.4 Conclusion

Through the use of Gaussian process regression modeling, we are able to accurately re-create the kinematic and fluid flow results from varied Reynolds number computational fluid dynamics simulations and analyses. This allows us to gain a deeper understanding of how the Reynolds number impacts these results, without needing to expend the computational cost to compute a vast number of simulations. Moving forward, this work can be expanded to include other flow and mixing analysis. Additionally, we can varying other kinematic and flow parameters, such as the tentacle's tension and elasticity, and recreate these models with these are additional input parameters.

Appendix B

Wasserstein-Based Similarity Constrained Matrix Factorization for Drug-Drug Interaction Prediction

The following is a copy of a previously published paper which supports this dissertation work [65].

B.1 Introduction

As of August 2023, it was estimated that approximately 60% of the adult population in the United States had taken at least one prescription medication within the previous year, with 37% reporting the use of three or more in the same time period [52]. Given the large percentage of the population that takes prescription drugs, it is paramount for public health initiatives to understand potential drug interactions.

A significant hurdle in achieving this understanding lies in the timeline from a drug's conception to its approval, spanning nearly eight years on average from the initiation of clinical trials to final regulatory clearance, excluding the initial phases of drug development [14]. While clinical trials address some potential interactions, the focus tends to center on commonly co-prescribed medications [82]. Compounding this issue is the accelerated rate at which drugs are gaining approval. Between 2011 and 2018, the FDA's Center for Drug Evaluation and Research sanctioned an average of 26 drugs annually, compared to an average of 23 approvals per year from 2000 to 2010 [4]. This influx of medications makes it increasingly impractical to assess all possible drug combinations that a patient might encounter. Consequently, healthcare practitioners often only recognize interactions between frequently prescribed drugs or in cases where adverse effects are severe.

In response to these challenges, researchers have turned to data analytics and machine learning techniques to enhance the prediction of drug-drug interactions (DDIs). Leveraging graphical representations of drugs, these methods employ matrix completion algorithms to predict interactions based on drug features and similarities [31, 59, 115, 116]. We aim

to add to this class of understanding by proposing the Wasserstein distance as an alternate similarity measure to those already studied, as it allows us to look at features of a drug at the atomic level which ultimately provides greater insight into the drug itself. Specifically, we apply the Wasserstein distance to the Manifold Regularized Matrix Factorization method for drug interaction prediction, which has previously seen success at predicting DDIs.

B.2 Methods

B.2.1 Drug molecules as probability measures

Graphs serve as a powerful depiction of small molecules, where atoms are nodes and chemical bonds are edges. This graph-based representation efficiently captures both the structural and chemical characteristics of these molecules. Our methodology of obtaining the similarity matrix unfolds in several stages: (1) converting each drug molecule into a set of node embeddings derived from its atomic properties; (2) measuring the Wasserstein distance between all pairs of graphs by solving the drug matching problem as an optimal transport problem; and (3) building the similarity matrix using the computed Wasserstein distance. By solving the graph matching of the chemical atomic-level and topological features of the drugs, we produce an informative similarity matrix that is used to enforce the drug similarity constraints on our matrix factorization optimization problem.

To formalize the attributed graph-matching problem, we consider undirected labeled graphs, denoted as tuples in the form: $\mathcal{G}(\mathcal{V}, \mathcal{E}, l_f)$. Here, $(\mathcal{V}, \mathcal{E})$ represents the set of vertices and edges within the graph. The function l_f assigns each vertex $v_i \in \mathcal{V}$ a feature vector $a_i = l_f(v_i)$ in a specific feature metric space. In our specific application, this feature vector captures atomic properties in a chemical compound. Following [1, 2], we enhance the graph by introducing a histogram to convey the relative significance of its vertices. In this enhancement, assuming the graph comprises N vertices, individual weights h_i are assigned to each vertex. Consequently, our graph takes the form $\mathcal{G}(\mathcal{V}, \mathcal{E}, l_f, h_\mathcal{G})$, where $h_\mathcal{G}$ is a function associating a weight with each vertex with $h_i = h_\mathcal{G}(v_i)$. This definition enables us to represent the graph as a probability measure with comprehensive support across the feature space. In scenarios where all weights are equal ($h_i = \frac{1}{N}$), every vertex holds an equivalent degree of relative importance.

Our objective is to define a matching distance metric between two graphs (i.e., drug molecules), labeled $\mathcal{G}_1$ and $\mathcal{G}_2$, each possessing N and K vertices (atoms), respectively. These graphs are distinguished by their corresponding probability measures, $h_{\mathcal{G}_1}$ and $h_{\mathcal{G}_2}$. Subsequently, we proceed to calculate the pairwise distances between the drugs using optimal transport. This begins with the computation of ground distances for each node pair.

B.2.2 Wasserstein distance on drug molecules

Optimal transport, as outlined in [94], offers a mathematical paradigm tailored to solving the challenge of finding the most efficient way of relocating objects, often

probability distributions, from one configuration to another while minimizing costs. Efficiency, in this context, pertains to achieving such transfers at minimal expense.

To mathematically formulate the transportation problem in the context of graphs, consider two sets of nodes, denoted as $\mathcal{X} = \{x_j\}_{j=1}^{N}$ representing the source samples, which correspond to the nodes of the first graph (representing the molecules of one drug), and $\mathcal{Y} = \{y_i\}_{i=1}^{K}$, corresponding to the target nodes of the second graph (representing a second drug). We introduce two discretized distributions of interest, denoted as $\mathbf{p} \in \mathcal{H}_N$ and $\mathbf{q} \in \mathcal{H}_K$. Here, $\mathcal{H}_N$ and $\mathcal{H}_K$ signify histograms with N and K bins, respectively. The vector $\mathbf{p}$ serves to convey the relative significance of the vertices (atoms) within the first drug, and the vector $\mathbf{q}$ for the second drug. These distributions adhere to the following constraints: $\mathbf{p} \in \mathbb{R}_{+}^{N}$, indicating that each element of $\mathbf{p}$ is a non-negative real number, and the sum of its elements equals 1, i.e., $\sum_j p_j = 1$. Similarly, $\mathbf{q} \in \mathbb{R}_{+}^{K}$ with the constraint $\sum_i q_i = 1$.

Define $Q \in \mathbb{R}+^{K \times N}$ as the transportation plan or coupling matrix, with elements denoted as Q_{ij}. This matrix precisely quantifies the amount situated at the source x_j, having a total mass of p_j, which necessitates transportation to the target location y_i, possessing a total mass of q_i. From a linear algebra perspective, the entries in the j^{th} column of Q represent the amounts from the j^{th} source to be transported to the targets, while the entries in the i^{th} row signify the amounts from the sources transported to the i^{th} target. Mathematically,

$$\begin{aligned} \sum_{j=1}^{N} Q_{ij} &= q_i \qquad \text{for all } \; i \in \{1, .., K\}, \\ \sum_{i=1}^{K} Q_{ij} &= p_j \qquad \text{for all } \; j \in \{1, .., N\}. \end{aligned} \tag{B.1}$$

The transportation process involves a cost matrix C, where C_{ij} represents the geometric distance between x_i and y_j. The total cost related to a transport plan is defined as follows:

$$\langle C, Q \rangle_F = \sum_{i=1}^{K} \sum_{j=1}^{N} C_{ij} Q_{ij} \tag{B.2}$$

where $\langle \cdot , \cdot \rangle$ is the Frobenius dot-product of two matrices. The optimal transport is thus formulated as the following optimization problem:

$$\begin{aligned} \underset{Q \in \mathbb{R}^{K \times N}}{\text{minimize}} \quad & \langle C, Q \rangle_F \\ \text{subject to} \quad & \sum_{j=1}^{N} Q_{ij} = q_i \quad \text{for all } \; i \in \{1, .., K\} \\ & \sum_{i=1}^{K} Q_{ij} = p_j \quad \text{for all } \; j \in \{1, .., N\} \\ & Q_{ij} \geq 0 \quad \text{for all } \; (i, j) \in \{1, .., K\} \times \{1, .., N\} \end{aligned} \tag{B.3}$$

In each column, the transportation process must precisely allocate the mass p_i to achieve a perfect match with the intended mass q_j. This optimization problem can be expressed more succinctly as follows:

$$\begin{aligned} &\underset{Q\in\mathbb{R}^{K\times N}}{\text{minimize}} \quad \langle C, Q\rangle_F \\ &\text{subject to} \quad Q\mathbf{1}_N = \mathbf{q}, \qquad Q^T\mathbf{1}_K = \mathbf{p}, \qquad Q \geq 0, \end{aligned} \tag{B.4}$$

where $\mathbf{1}_N$ and $\mathbf{1}_K$ represent vectors of ones in dimension N and K, respectively. We define the set of all admissible couplings or transport plans $\mathbf{Q}(\mathbf{p}, \mathbf{q})$ between histograms as follows:

$$\mathbf{Q}(\mathbf{p}, \mathbf{q}) = \{Q \in \mathbb{R}_+^{K\times N} \mid Q\mathbf{1}_N = \mathbf{q}, Q^T\mathbf{1}_K = \mathbf{p}\}.$$

In a specific scenario where the cost function C aligns with a distance matrix, the optimal transport problem leads to the Wasserstein distance on $\mathcal{H}_N \times \mathcal{H}_K$. The Wasserstein distance is defined as:

$$W(\mathbf{p}, \mathbf{q}) = \underset{Q\in\mathbf{Q}(\mathbf{p},\mathbf{q})}{\text{minimize}} \ \langle C, Q\rangle_F. \tag{B.5}$$

We employ the simplex method [123] to solve (B.4), which iteratively moves from one feasible solution to another along the edges of the feasible region until an optimal solution is attained. The method employs a tableau representation, with each iteration selecting a pivot column and row for tableau update. The algorithm iterates until optimality conditions are satisfied.

We solve (B.4) for each pair of drugs and obtain the distance matrix $D \in \mathbb{R}^{m\times m}$, where m is the number of drugs and where the entries are given by (B.5). We define our similarity matrix S as follows:

$$S = \frac{1}{\max(D)}\Big(\max(D)\mathbf{1}_{m\times m} - D\Big) \tag{B.6}$$

where $\max(D)$ is the maximum element in the matrix D and $\mathbf{1}_{m\times m}$ is the matrix of ones of size $m \times m$. Note that the entries s_{ij} of S are between 0 and 1. Also, the larger the distance, i.e., the entry in D, the closer the corresponding entry in S is to 0, and vice versa.

B.2.3 Similarity-Constrained Matrix Factorization

Our implementation follows the methodology introduced by Zhang[127]. We assume we have a DDI matrix, A, which is an $m\times m$ symmetric matrix with zeroes on the diagonal. The elements of A indicate whether or not an interaction is known between each drug, with a value of 1 indicating a known interaction while 0 indicates either no interaction or no known interaction.

Once A and S are known, the goal is to decompose A into two low-rank matrices $X \in \mathbb{R}^{m\times k}$, $Y \in \mathbb{R}^{m\times k}$ such that $A = XY^T$, where k is the dimension of a latent feature vector. The objective function we consider in this work is

$$\begin{aligned} L(X, Y) &= \frac{1}{2}\|A - XY^T\|_F^2 + \frac{\lambda}{2}\left(\|X\|_F^2 + \|Y\|_F^2\right) \\ &= \frac{1}{2}\sum_{i,j}\left(a_{ij} - \vec{x}_i\vec{y}_j^T\right)^2 + \frac{\lambda}{2}\sum_i\left(\|\vec{x}_i\|_2^2 + \|\vec{y}_j\|_2^2\right) \end{aligned} \tag{B.7}$$

where the first term in (B.7) is a fitting term and the second is a regularization term to make the problem well-posed. Here $\|\cdot\|_F$ is the Frobenius norm, $\|\cdot\|_2$ is the Euclidean norm, $\vec{x}_i$ is the ith row of X, $\vec{y}_j$ is the jth row of Y, and $\lambda > 0$ is the Tikhonov regularization parameter.

This method assumes that drug-drug similarities can be represented by manifolds in the drug feature space, meaning that each drug feature has a neighborhood that maps to the drug similarity. By making this assumption, we can utilize the fact that nearby points in the drug feature space will likely have a similar mapping to their drug similarity. Therefore, the objective function L can then be regularized over the manifolds by

$$L_{reg}^{row}(X) = \frac{1}{2}\sum_{i,j} s_{ij}\|\vec{x}_i - \vec{x}_j\|_2^2 \tag{B.8}$$

$$L_{reg}^{col}(Y) = \frac{1}{2}\sum_{i,j} s_{ij}\|\vec{y}_i - \vec{y}_j\|_2^2 \tag{B.9}$$

where s_{ij} is the similarity between the ith and jth drugs. Finally, we attain the Manifold Regularized Matrix Factorization (MRMF) [127] by combining (B.7) with (B.8) and (B.9) resulting in the following optimization problem:

$$\begin{aligned}\underset{X,Y\in\mathbb{R}^{m\times k}}{\text{minimize}}\ & L_{mix}(X,Y) \\ &= L(X,Y) + \mu\left(L_{reg}^{row}(X) + L_{reg}^{col}(Y)\right),\end{aligned} \tag{B.10}$$

where $\mu > 0$ is a manifold regularization parameter.

To find the minimizer of (B.10), we utilize an alternating descent method in [127]. This method is equivalent to a gradient descent where $\vec{x}_i$ and $\vec{y}_j$ are successively fixed. We start by randomly initializing $\vec{x}_i$ and $\vec{y}_j$. Then, we use Newton's method to compute each update to $\vec{x}_i$ and $\vec{y}_j$:

$$\vec{x}_i \leftarrow \vec{x}_i - \nabla_{\vec{x}_i} L_{mix}\left(\nabla^2_{\vec{x}_i} L_{mix}\right)^{-1} \tag{B.11}$$

$$\vec{y}_j \leftarrow \vec{y}_j - \nabla_{\vec{y}_j} L_{mix}\left(\nabla^2_{\vec{y}_j} L_{mix}\right)^{-1} \tag{B.12}$$

where $\nabla_{\vec{x}_i} L_{mix}$ and $\nabla_{\vec{y}_j} L_{mix}$ are the gradients of L_{mix} with respect to $\vec{x}_i$ and $yvec_j$, respectively, and $\nabla^2_{\vec{x}_i} L_{mix}$ and $\nabla^2_{\vec{y}_j} L_{mix}$ are the Hessians of L_{mix} with respect to $\vec{x}_i$ and $yvec_j$, respectively. The updates (B.11) and (B.12) are continued until convergence.

B.3 Numerical Experiments

B.3.1 Data

As pharmaceutical data becomes increasingly accessible, researchers have turned to curated databases for comprehensive drug information. TWOSIDEs, for instance, aggregates reported drug interactions without attribution to a specific drug [107].

Complementary feature data, encompassing side effects, off-label effects, and chemical compositions, are drawn from established repositories such as SIDER [55], OFFSIDES [107], PubChem [50], and DrugBank [120]. By linking drugs with known interactions to those with detailed feature information, a comprehensive dataset is formed. This mapping, compiled by Zhang [128], identifies 548 drugs with both interaction data and detailed features. The dataset, including the drug-drug interaction matrix, A, is publicly accessible via Github[1].

The key feature utilized to construct our similarity matrix is the Simplified Molecular-Input Line-Entry System (SMILES) string associated with each drug. SMILES leverages molecular graph theory to provide atomic-level feature representation for drugs [118]. However, it's worth noting that four drugs in the original dataset lack SMILES indices, resulting in a final dataset comprising 544 drugs with a total of 47,537 known interactions.

To evaluate the accuracy of our method, 10% of the known interactions are randomly withheld. Subsequently, our objective is to accurately reconstruct the original drug-drug interaction matrix A.

B.3.2 Parameter Settings

The three parameters in the MRMF method that can be altered are (i) the latent feature vector size, k, (ii) the Tikhonov regularization parameter, λ, and (iii) the manifold regularization parameter, μ. Several experiments were run altering these parameters to find the optimal parameter setting. Zhang [127] found $k = 0.08 * m$, $\lambda = 2^2$, and $\mu = 2^2$ to produce the best result, so we initially started with this combination. In their study, this was found to be the best combination of parameters, regardless of the similarity metric used. Our results show that this is not an optimal parameter combination when using the Wasserstein distance metric, as it predicts that no drug interactions are present.

We then swept over the parameter space field to find the optimal settings. Our first sweep held $k = 0.1 * m$ while looping over $\lambda, \mu \in J = \{2^\kappa \colon \kappa \in -4, -3, -2, \cdots, 2, 3\}$. Next, we conducted four variations of an experiment where μ and λ are held constant while looping over $k = 0.01\beta * m$, where $\beta \in \{2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$. In the first of these we set $\mu = \lambda = 2^2$, in the second $\mu = \lambda = 2^{-2}$, in the third $\mu = \lambda = 2^{-4}$, and finally in the fourth $\mu = \lambda = 2^{-6}$.

Of these experiments, the greatest results were achieved when $k = 0.1 * m$ and $\mu = \lambda = 2^{-6}$. The results presented from here on will be based on these parameters.

Ultimately, these experiments found that the value for the latent feature vector, k, did not have as great of an impact on the results as the regularization parameters μ and λ. When these regularization parameters are properly tuned, nearly identical results are achieved, regardless of k. This allows us to project a large DDI matrix into a much smaller space, saving computational cost. In our numerical experiments, we found that $k = 0.1 * m$ is sufficient.

[1] `https://github.com/zw9977129/drug-drug-interaction/tree/master`

B.4 Results and Discussion

B.4.1 Evaluation Metrics

To analyze the performance of our method, we compare recall, precision, accuracy, and F-score values:

$$\text{Recall} = \frac{TP}{TP + FN}, \tag{B.13}$$

$$\text{Precision} = \frac{TP}{TP + FP} \tag{B.14}$$

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{B.15}$$

$$\text{F-Score} = \frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}} \tag{B.16}$$

where TP is the number of true positives, TN is the number of true negatives, FP is the number of false positives, and FN is the number of false negatives. Additionally, we utilize the area under the ROC Curve (AUC), which is a metric of the true positives vs false positive rates, and the area under the precision-recall curve (AUPRC), which measures the true positives to predicted positives.

Of these metrics, the AUPRC takes precedence in assessing method performance, as it directly measures the ratio of correctly predicted true positives to all predicted positives. This has been chosen as the leading performance metric as in clinical settings where this may be applied, the ability to correctly predict true positives is highly valued, and therefore it is acceptable to have a slight decrease in overall performance if this measure is being met. However, the remaining metrics do still provide supplementary insight into model performance.

B.4.2 Comparison with Other Similarity Measures

Similarity	AUPRC	AUC	Recall	Precision	Accuracy	F-score
Jaccard	0.7958	0.9585	0.6812	0.7722	0.9546	0.7237
Cosine	0.7958	0.9584	0.6906	0.7729	0.9546	0.7237
Gauss	0.7959	0.9584	0.6806	0.7726	0.9546	0.7236
Wasserstein	0.9449	0.9690	0.7777	0.9292	0.9095	0.8467

Table B.1: Accuracy metrics for various similarity metrics. The first three rows come from Zhang [127] with $\mu = \lambda = 2^2$, $k = 0.8 * m$, while the last row comes from our approach (using the Wasserstein distance) with $\mu = \lambda = 2^{-6}$, $k = 0.1 * m$.

We conducted a comparative analysis between the results obtained using the Wasserstein distance to generate our similarity matrix and those reported by Zhang [127],

as summarized in Table B.1. While the Wasserstein distance looks at features of a drug on an atomic level, the Jaccard similarity, Cosine similarity, and Gauss similarity each look at the drug features holistically. As such, these measures look to understand how similar two drugs may be in terms of their overall composition and effects but may miss key information that can only be seen when exploring the chemical structure of a drug. Each of these are common measures in bioinformatics problems and therefore serve as a basis to compare our proposed Wasserstein-based similarity measure. Zhang's study indicated that, under their optimal parameters, similar accuracy was achieved regardless of the similarity metric employed, suggesting that the accuracy of the Manifold Regularized Matrix Factorization (MRMF) method was independent of the similarity measure utilized. In contrast, our approach using the Wasserstein similarity measure yielded a significantly higher AUPRC of 0.9449. This outcome underscores the effectiveness of our metric in accurately identifying potential drug-drug interactions.

It's important to note that our measure's high AUPRC primarily reflects its ability to predict true positives, while having slightly lower performance with regards to predicting true negatives. We surmise that this is because the Wasserstein distance captures the similarity between features that drive drug interaction, while excluding the features that do not. This may result in a lower overall accuracy score, however, in practical application, the focus lies on identifying potential interactions for further investigation, thus prioritizing true and false positives over true negatives.

B.5 Conclusion

This study proposes an alternative similarity metric for similarity-constrained drug-drug interaction prediction. Our findings demonstrate that the Wasserstein similarity measure outperforms previously studied metrics when integrated with the MRMF method for DDI prediction, particularly in terms of true positive prediction rates.

Appendix C

Triple Matrix Factorization for Drug-Drug Interaction Prediction Using Fused Gromov-Wasserstein Distances

The following is a copy of a previously published paper which supports this dissertation work [64].

C.1 Introduction

Understanding drug-drug interactions (DDIs) is a vital issue as prescription drug use becomes more prevalent. Typically, the timeline of drug discovery to marketing takes 10 to 15 years, during which a proposed drug goes through multiple phases of clinical trials to determine if it is safe for approval. This includes standard methods for DDI prediction, such as in vitro methods and clinical studies, but as more drugs become available, it is harder to test all potential interactions. The result is that interactions are typically only known when there are severe side effects or between drugs that are commonly prescribed [82].

With this in mind, researchers have begun exploring the use of data analytics and machine learning methods to better predict DDIs. Of these, a popular method for prediction is to look at how similar two drugs are and predict interaction based on this measure, illustrated in Figure C.1, otherwise known as similarity-based methods. Previous methods have explored this class of methods using both different features and different similarity measures [9, 31, 59, 116, 127, 128].

We propose a class of Wasserstein-based distance metrics as an alternative similarity measure to those already studied. Wasserstein-based methods allow us to look at the features of a drug at the atomic level, providing greater insight into the drug itself. We then use a matrix factorization approach using these distances. In particular, we build upon the triple matrix factorization in [101], employing Wasserstein-based distance metrics similarity between drugs as our sole feature information when predicting

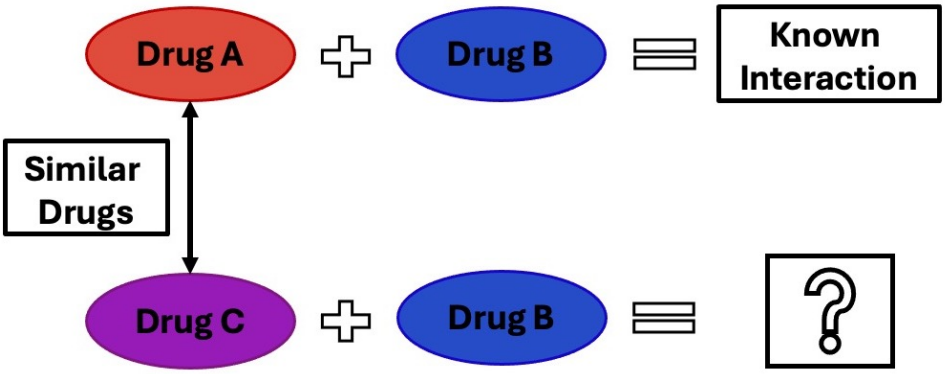


Figure C.1: Drug-drug interaction problem where if Drugs A and B are known to interact, and Drug C is known to be similar to Drug A, we want to predict whether Drug C will interact with Drug B.

drug-drug interaction.

C.2 Methods

C.2.1 Drug molecules as probability measures

Graphs are a valuable tool to depict small molecules, allowing atoms to be represented by nodes and chemical bonds as edges, capturing both the structural and chemical characteristics of these molecules. Once molecules are represented via graphs, we can easily compare molecules and find how similar they are to each other, resulting in a similarity matrix. This process starts by converting each drug molecule into a set of node embeddings derived from its atomic properties. We then measure the distance between all pairs of graphs by solving the drug-matching problem as an optimal transport problem. Finally, we build the similarity matrix using the computed distance. This produces an informative similarity matrix that is used to enforce drug similarity constraints on our matrix factorization optimization problem.

We consider undirected labeled graphs, denoted as tuples in the form $\mathcal{G}(\mathcal{V}, \mathcal{E}, l_f)$, where $(\mathcal{V}, \mathcal{E})$ represents the set of vertices and edges within the graph. The function l_f assigns each vertex $v_i \in \mathcal{V}$ a feature vector $a_i = l_f(v_i)$ in a feature vector space which captures atomic properties in a chemical compound. We enhance the graph by introducing a histogram to convey the relative significance of its vertices (see [1, 2] for further details). This enhancement assumes the graph comprises N vertices and individual weights h_i are assigned to each vertex. Therefore, our graph takes the form $\mathcal{G}(\mathcal{V}, \mathcal{E}, l_f, h_\mathcal{G})$, where $h_\mathcal{G}$ is a function associating a weight with each vertex with $h_i = h_\mathcal{G}(v_i)$. This allows us to represent the graph as a probability measure with comprehensive support across the feature space. If all weights are equal, then each vertex holds an equivalent degree of

relative importance.

Our goal is to define a matching distance metric between two graphs of drug molecules, $\mathcal{G}$ and $\tilde{\mathcal{G}}$, which each possesses N and K vertices, or atoms, respectively. These graphs are differentiated through their corresponding probability measures $h_{\mathcal{G}}$ and $h_{\tilde{\mathcal{G}}}$. We then calculate the pair-wise distance between the drugs using optimal transport, which begins with the computation of ground distances of each node pair. This process can be seen illustrated in Figure C.2.

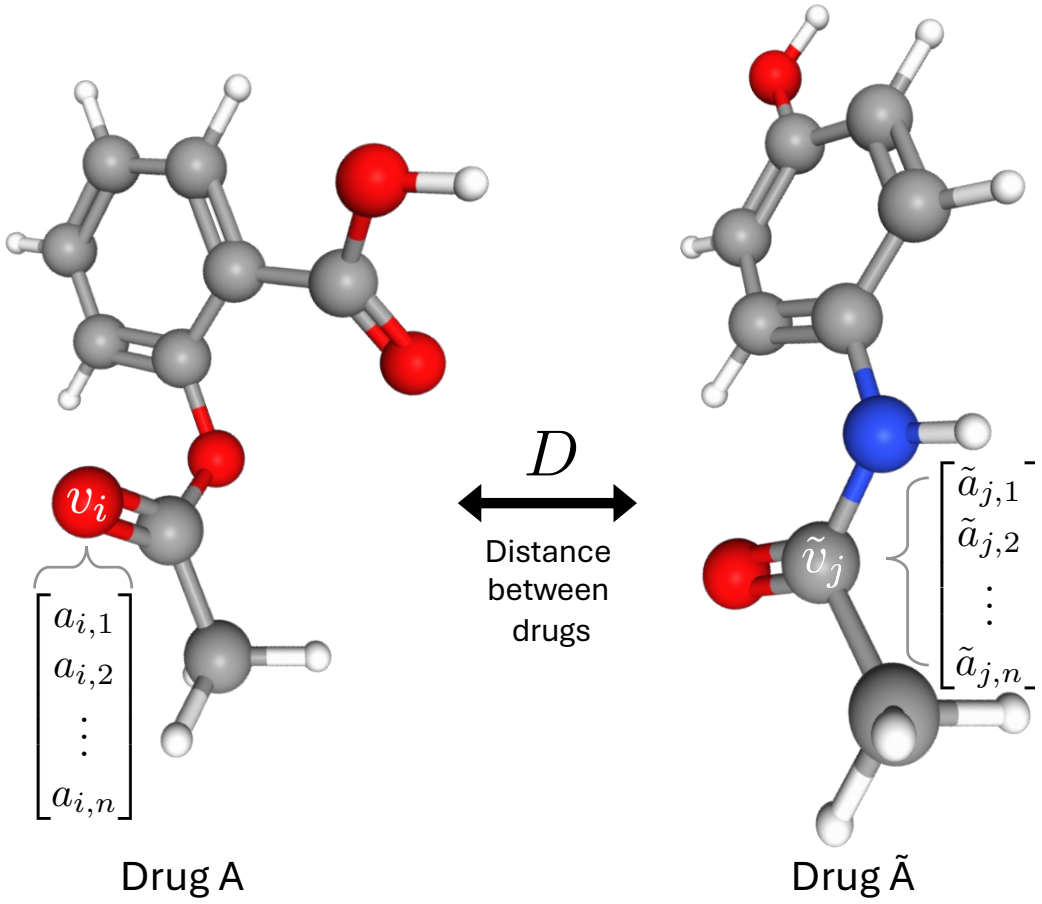


Figure C.2: The distance between two molecules is computed by representing the molecules as graphs with atoms as vertices. Each vertex, v_i, is associated with a corresponding feature information vector, a_i.

C.2.2 Wasserstein distance of drug molecules

Optimal transport is the field of mathematics focused on finding the most efficient way of moving objects from one configuration to another while minimizing costs [94]. In this frame, efficient transport requires not only successfully transporting the objects, but doing so at minimal expense.

Formulating this in terms of graphs, consider two sets of nodes, $\mathcal{X} = \{x_j\}_{j=1}^{N}$ representing the source samples and $\mathcal{Y} = \{y_i\}_{i=1}^{K}$ representing the target nodes of the second graph, where each graph represents the molecules of different drugs. We introduce two discretized distributions, denoted as $\mathbf{p} \in \mathcal{H}_N$ and $\mathbf{q} \in \mathcal{H}_K$, where $\mathcal{H}_N$ and $\mathcal{H}_K$ are histograms with N and K bins, respectively. The vectors $\mathbf{p}$ and $\mathbf{q}$ convey the relative significance of the vertices, or atoms, within the first drug and second drugs, accordingly. These distributions are constrained by $\mathbf{p}, \mathbf{q} \in \mathbb{R}_+^N$, meaning that each element of these distributions is a non-negative real number. Since $\mathbf{p}$ and $\mathbf{q}$ correspond to discretized distributions, the sum of their elements equals 1.

Define $Q \in \mathbb{R}_+^{K \times N}$ as the transportation plan or coupling matrix, with elements denoted as Q_{ij}. This matrix quantifies the amount at the source x_j with a total mass of p_j, which requires transport to the target location y_i with a total mass of q_i. The entries in the j^{th} column of Q represent the amounts from the j^{th} source to be transported to the targets. The entries in the i^{th} row represent the amounts from the sources to be transported to the i^{th} target. Mathematically, this can be written as the constraints

$$\sum_{j=1}^{N} Q_{ij} = q_i \qquad \text{for all } \; i \in \{1, .., K\},$$
$$\sum_{i=1}^{K} Q_{ij} = p_j \qquad \text{for all } \; j \in \{1, .., N\}.$$

An example of Q can be seen in Figure C.3, demonstrating how the columns of Q must satisfy the first constraint, while the rows of Q must satisfy the second.

	0.15	0.20	0.17	0.26	0.22	} q
0.18	0.02	0.01	0.05	0.03	0.07	
0.25	0.03	0.14	0.00	0.04	0.04	
0.22	0.04	0.03	0.08	0.03	0.04	
0.35	0.06	0.02	0.04	0.16	0.07	

p $\qquad\qquad$ $Q \in \mathbb{R}^{K \times N}$

Figure C.3: Example of a coupling matrix, $Q \in \mathbb{R}^{K \times N}$, where $K = 4$ and $N = 5$. Here the first column corresponds to the source mass vector, $p \in \mathbb{R}^4$, and the first row corresponds to the target mass vector, $q \in \mathbb{R}^5$. The elements in the rows of Q sum to the corresponding value of p (see e.g., the row in beige), and the elements in the columns of Q sum to the corresponding value of q (see e.g., the column in light blue).

We also consider a cost matrix, C, where C_{ij} represents the cost of transporting mass from source x_j to target y_i. The optimal transport can therefore be written as the following

optimization problem:

$$\begin{aligned} \underset{Q\in\mathbb{R}^{K\times N}}{\text{minimize}} \quad & \mathcal{F}_W(Q) = \sum_{i=1}^{K}\sum_{j=1}^{N} C_{ij}Q_{ij} \\ \text{subject to} \quad & Q_{ij} \geq 0 \quad \text{for all } (i,j) \in \{1,..,K\}\times\{1,..,N\} \\ & \sum_{j=1}^{N} Q_{ij} = q_i \quad \text{for all } i \in \{1,..,K\} \\ & \sum_{i=1}^{K} Q_{ij} = p_j \quad \text{for all } j \in \{1,..,N\} \end{aligned} \tag{C.1}$$

We define the set of all admissible couplings or transport plans $\mathbf{Q}(\mathbf{p},\mathbf{q})$ between histograms as

$$\mathbf{Q}(\mathbf{p},\mathbf{q}) = \{Q \in \mathbb{R}_{+}^{K\times N} \mid Q\mathbf{1}_N = \mathbf{q}, Q^T\mathbf{1}_K = \mathbf{p}\},$$

where $\mathbf{1}_N$ and $\mathbf{1}_K$ represent vectors of ones in dimension N and K, respectively. When the cost function C aligns with a distance matrix, the optimal transport problem results in the Wasserstein distance on $\mathcal{H}_N \times \mathcal{H}_K$. The Wasserstein distance is defined as

$$W(\mathbf{p},\mathbf{q}) = \underset{Q\in\mathbf{Q}(\mathbf{p},\mathbf{q})}{\text{minimize}}\ \mathcal{F}_W(Q) \tag{C.2}$$

which is solved using the simplex method [123].

C.2.3 Gromov-Wasserstein distance of drug molecules

The Wasserstein distance primarily focuses on the inherent features of graph elements, addressing them in isolation. The Gromov-Wasserstein distance overcomes this limitation by extending the Wasserstein distance to capture the relational aspects among the vertices in a graph, revealing the structural composition of the graph. It depends on the intra-domain structure matrices of both graphs, denoted as $S^\sigma = (S^\sigma(x_i,x_k))_{i,k} \in \mathbb{R}^{N\times N}$ for the source and $S^\tau = (S^\tau(y_j,y_l))_{j,l} \in \mathbb{R}^{K\times K}$ for the target. The Gromov-Wasserstein distance is defined by the following optimization problem:

$$GW(\mathbf{p},\mathbf{q}) = \underset{Q\in\mathbf{Q}(\mathbf{p},\mathbf{q})}{\text{minimize}}\ \mathcal{F}_{GW}(Q) \tag{C.3}$$

where

$$\mathcal{F}_{GW}(Q) = \sum_{i,k=1}^{N}\sum_{j,l=1}^{K} |S_{ik}^\sigma - S_{jl}^\tau| Q_{ij}Q_{kl} \tag{C.4}$$

which can be seen depicted in Figure C.4. The choice of structural matrices denoted as S^σ and S^τ, embodies a flexible aspect of our approach, enabling the integration of an array of structural and topological information that defines the graph. These matrices can manifest in diverse forms, encompassing representations of shortest path data, adjacency connections, weighted shortest paths, harmonic distances, or dissimilarity matrices based on node-to-node features. The Gromov-Wasserstein optimization problem (C.3) is solved using conditional gradient [23, 110] (see [74] for further details.)

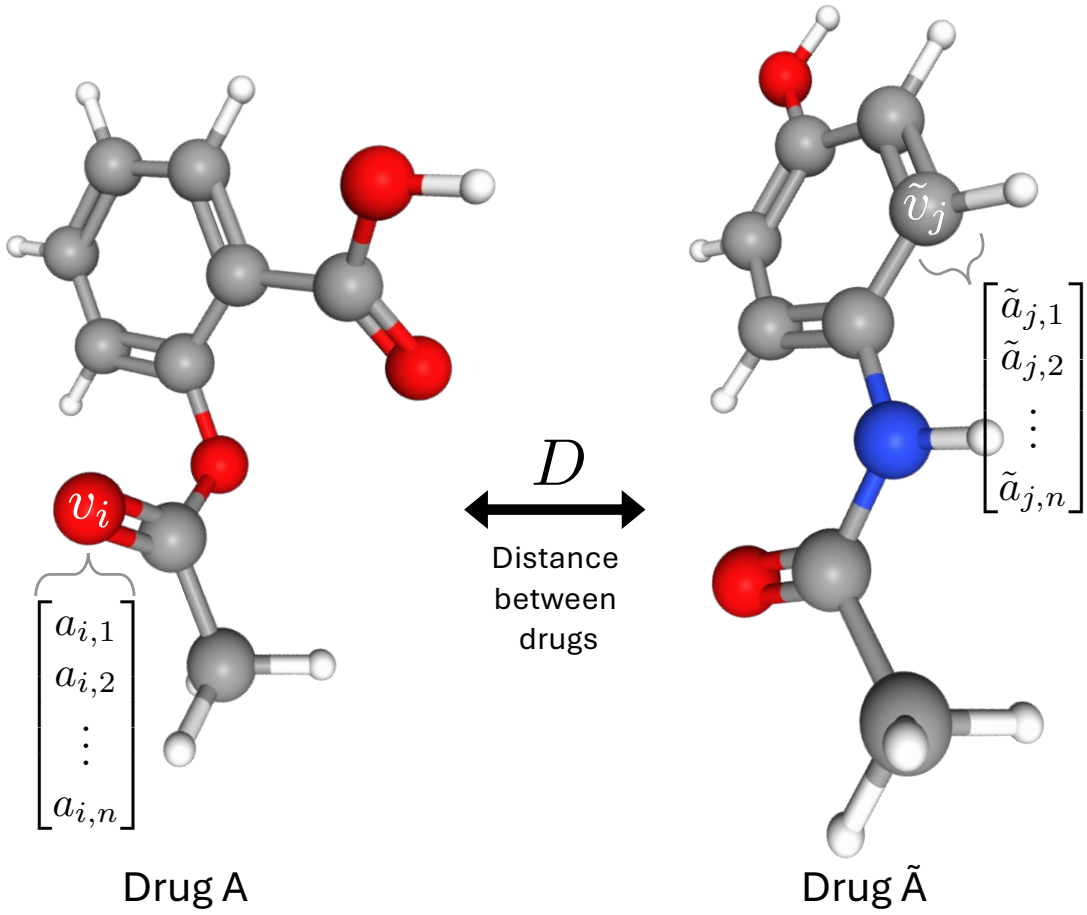


Figure C.4: The Gromov-Wasserstein distance calculates the distance between two molecules, A and $\tilde{A}$, as a function of the difference between the similarity, S, of the atoms composing the drug.

C.2.4 Fused Gromov-Wasserstein distance of drug molecules

Now, we introduce the concept of the Fused Gromov-Wasserstein distance [110], an extension of the Wasserstein and Gromov-Wasserstein distances. This measure extends the capabilities of both distances by simultaneously integrating and harmonizing feature and structural information, enabling a holistic analysis of graph data. The Fused Gromov-Wasserstein distance is defined by the following optimization problem:

$$FGW(\mathbf{p}, \mathbf{q}) = \underset{Q \in \mathbf{Q}(\mathbf{p},\mathbf{q})}{\text{minimize}} \ (1-\alpha)\mathcal{F}_W(Q) + \alpha\mathcal{F}_{GW}(Q) \tag{C.5}$$

where α (with $0 \leq \alpha \leq 1$) is a scalar trade-off parameter that balances the Wasserstein and Gromov-Wasserstein distances. The Fused Gromov-Wasserstein optimization problem (C.5) is also solved using conditional gradient [23, 110].

We solve the distance equation for each pair of drugs to retrieve the distance matrix $D_\alpha \in \mathbb{R}^{m \times m}$, where m is the number of drugs and the entries are given by (C.5) for a given

value of α. Our similarity matrix is then defined as a function of the distance matrix:

$$S_\alpha = \frac{1}{\max(D_\alpha)}\Big(\max(D_\alpha)\mathbf{1}_{m\times m} - D_\alpha\Big) \tag{C.6}$$

where $\max(D_\alpha)$ is the maximum element in the matrix D_α and $\mathbf{1}_{m\times m}$ is the matrix of ones of size $m \times m$. Note that the entries in S_α are between 0 and 1, where a smaller value signifies a greater distance between the points, and therefore less similarity. For notational convenience, we drop the subscript α on S hereon.

C.2.5 Similarity-Based Triple Matrix Factorization

We now begin the process of matrix completion to find potential DDIs using the triple matrix factorization (TMF) approach [101] with drug similarity as our sole feature. Consider a DDI matrix, $A \in \mathbb{R}^{m\times m}$ which is a symmetric matrix with a 0 diagonal with m being the number of drugs in the study. The element A_{ij} denotes whether an interaction is known between drugs i and j, with $A_{ij} = 1$ being a known interaction, while $A_{ij} = 0$ represents either no interaction or no known interaction. We aim to recover those values where interaction is not known. This is completed by decomposing A as

$$A = S_d\widehat{\Theta}S_t^T \tag{C.7}$$

where $\widehat{\Theta} \in \mathbb{R}^{m\times m}$ is a low-rank matrix used to project from the known feature space to the latent feature space and the $m \times m$ matrices S_d and S_t come from computing the singular value decomposition

$$S = U_S\Sigma_S V_S^T = \Big(U_S\sqrt{\Sigma_S}\Big)\left(\sqrt{\Sigma_S^T}V_S^T\right) = S_dS_t^T. \tag{C.8}$$

The decomposition problem (C.7) might be ill-posed or might not have a solution. Thus we formulate the related optimization problem

$$\underset{\widehat{\Theta}}{\text{minimize}}\ J(\widehat{\Theta}) = \|A - S_d\widehat{\Theta}S_t^T\|_F^2 + \lambda\|\widehat{\Theta}\|_F^2 \tag{C.9}$$

which can be solved by setting the gradient of $J(\widehat{\Theta})$ to zero and solving for $\widehat{\Theta}$:

$$\nabla J(\widehat{\Theta}) = -2S_d^TAS_t + 2S_d^TS_d\widehat{\Theta}S_t^TS_t + 2\lambda\widehat{\Theta} = 0 \tag{C.10}$$

However, this is computationally difficult to solve when m is sufficiently large, as is common in DDI problems.

Alternatively, we write $\widehat{\Theta} = \widehat{B}_d\widehat{B}_t^T$, where $\widehat{B}_d \in \mathbb{R}^{m\times k}$ (along with S_d) contains information corresponding to the known drug data, while $\widehat{B}_t \in \mathbb{R}^{m\times k}$ (along with S_t) contains information corresponding to the unknown targets that we aim to predict. Here, $k \ll m$ because $\widehat{\Theta}$ is a low rank matrix.

To compute $\widehat{B}_d$ and $\widehat{B}_t^T$, we introduce the intermediary matrices $\widehat{A}_d, \widehat{A}_t \in \mathbb{R}^{m\times k}$ such that

$$\widehat{A}_d \approx S_d\widehat{B}_d, \quad \widehat{A}_t \approx S_t\widehat{B}_t, \quad \text{and} \quad A \approx \widehat{A}_d\widehat{A}_t^T. \tag{C.11}$$

Doing so allows us to formulate an optimization problem similar to (C.10) but with the objective function

$$\begin{aligned}\widehat{J}(\widehat{A}_d, \widehat{A}_t, \widehat{B}_d, \widehat{B}_t) = ||A - \widehat{A}_d\widehat{A}_t^T||_F^2 + ||\widehat{A}_d - S_d\widehat{B}_d||_F^2 \\ + ||\widehat{A}_t - S_t\widehat{B}_t||_F^2 + \lambda||\widehat{A}_d||_F^2 \\ + \mu||\widehat{A}_t||_F^2 + \gamma||\widehat{B}_d||_F^2 + \beta||\widehat{B}_t||_F^2.\end{aligned}$$

The function $\widehat{J}$ can be minimized using the method of alternating least squares, which works by fixing all sets of variables except for one, setting the corresponding partial derivative to 0, and solving for that set of variable. The resulting solutions at each step are given by

$$\begin{aligned}\widehat{B}_d &= \left(S_d^T S_d + \gamma I\right)^{-1} S_d^T \widehat{A}_d, \\ \widehat{B}_t &= \left(S_t^T S_t + \beta I\right)^{-1} S_t^T \widehat{A}_t, \\ \widehat{A}_d &= \left(A\widehat{A}_t + S_d\widehat{B}_d\right)\left(\widehat{A}_t^T\widehat{A}_t + I + \lambda I\right)^{-1}, \\ \widehat{A}_t &= \left(A^T\widehat{A}_d + S_t\widehat{B}_t\right)\left(\widehat{A}_d^T\widehat{A}_d + I + \mu I\right)^{-1}.\end{aligned} \tag{C.12}$$

Note that the matrix inverses in (C.12) can be easily computed since they are small $k \times k$ matrices. To initialize the optimization method, we compute $\widehat{A}_d$ and $\widehat{A}_t$ using the SVD of A similar to (C.8) but using only the first k rows for each, and B_d and B_t are randomly initialized. Once the optimal matrices $\widehat{B}_d^*$ and $\widehat{B}_t^*$ are found, we can create our prediction matrix $A_{\text{pred}} = S_d\widehat{B}_d(\widehat{B}_t^*)^T S_t^T$.

C.3 Numerical Experiments

C.3.1 Data

In recent years, pharmaceutical information has become much more publicly available, leading to increased accessibility to actual drug data for researchers. TWOSIDES is a database that contains known and reported drug interactions that cannot be attributed to either drug [107]. Feature information such as side effects, off-label side effects, and chemical and molecular composition comes from the databases SIDER [55], OFFSIDES [107], PubChem [50], and DrugBank [120]. Drugs with known interactions are mapped to drugs with known feature information, creating the final dataset. Note that this data is binary, indicating only if an interaction occurred or not, and does not include information such as the severity of the interaction, whether it was a positive or negative, or if it was a dose-dependent or time-dependent. This mapping has been compiled by Zhang [128], which found 548 drugs with known interactions and feature data. This dataset, including the drug-drug interaction matrix, A, is publicly accessible via Github: `https://github.com/zw9977129/drug-drug-interaction/tree/master`.

The specific feature that we use to create our similarity matrix is a drugs simplified molecular-input line-entry system (SMILES) string. SMILES uses molecular graph theory

to label each drug and gives feature information at the atomic level for a drug [118]. Unfortunately, four drugs in the original dataset do not have SMILES indices, making it so the final dataset we use is made of $m = 544$ drugs with 47,537 known interactions.

To test our methods accuracy, 10% of the known interactions are randomly removed. We therefore aim to recover the original DDI matrix, A.

C.3.2 Parameter Settings

The parameters that can be varied in this implementation are the latent feature size, k, and the regularization weights $\lambda, \mu, \gamma, \beta$. For preliminary testing, we have set $\lambda = \mu = \gamma = \beta$ and conducted five variations of an experiment where $\mu = \lambda = \gamma = \beta$ are kept constant while looping over $k = 0.01\eta * m$, where $\eta \in \{2, 4, 6, 8, 10, 12, 14, 16, 18, 20\}$. In the first of these, we set $\mu = \lambda = 2^2$, in the second $\mu = \lambda = 2^0$, in the third $\mu = \lambda = 2^{-2}$, in the fourth $\mu = \lambda = 2^{-4}$, and finally in the fifth $\mu = \lambda = 2^{-6}$.

In addition, our objective is to see which α value in the Fused Gromov-Wasserstein distance metric produces the greatest results. Therefore, we will repeat each of the above experiments with $\alpha \in \{0.10, 0.25, 0.5, 0.75, 0.90\}$.

In these experiments, the greatest results were achieved regardless of the α value when $\lambda = \mu = \gamma = \beta = 2^{-6}$, and $k = 0.04 * m$. The results presented going forward will be based on these parameters. Moreover, the Fused Gromov-Wasserstein distance metric with $\alpha = 0.1$ ultimately performs the best of the distance metrics when using this parameter setting.

C.4 Results and Discussion

C.4.1 Evaluation Metrics

To analyze the performance of our method, we compare recall, precision, and accuracy. Additionally, we utilize the area under the ROC curve (AUC), a metric of true versus false positive rates, and the area under the precision-recall curve (AUPRC), a metric of true positives to predicted positives. The AUPRC will be the primary metric for evaluating the performance of these methods, since we care most about correctly predicted true positives, while the others provide performance analysis secondarily.

We compare the results of the Wasserstein distance, Gromov-Wasserstein distance, and Fused Gromov-Wasserstein distance with varied α values in Table C.1, which contains the AUPRC and AUC, and Table C.2 which shows further evaluation metrics. Focusing on the AUPRC in Table C.1, we see that the Fused Gromov-Wasserstein with $\alpha = 0.1$ performs the best with an AUPRC of 0.9559. As α gets larger, the performance drops slightly, signifying that the atomic level information of the drug is vital to the prediction capability of this method, but we do need to keep in mind the features of the drug as a whole structure.

The bottom portion of Table C.1 shows the results achieved in [101], which proposed the TMF method. Their implementation utilized four schemas, of which the third most closely represents our scenario, where we want to identify new DDIs using existing drug

	Experiment	AUPRC	AUC
Proposed Method	Wasserstein	0.918	0.945
	FGW ($\alpha = 0.10$)	0.956	0.969
	FGW ($\alpha = 0.25$)	0.955	0.968
	FGW ($\alpha = 0.50$)	0.951	0.966
	FGW ($\alpha = 0.75$)	0.953	0.968
	FGW ($\alpha = 0.90$)	0.960	0.965
	Gromov-Wasserstein	0.947	0.962
Shi [101]	Enzymes	0.866	0.967
	Ion Channels	0.853	0.972
	G-Protein Coupled Receptors	0.677	0.959
	Nuclear Receptors	0.675	0.929

Table C.1: Accuracy metrics for our implementation when using various Wasserstein-based similarity metrics compared to results from Shi [101]. Our implementation uses a parameter setting of $\lambda = \mu = \gamma = \beta = 2^{-6}$, and $k = 0.04 * m$ where FGW stands for Fused Gromov-Wasserstein. Note that the results from Shi [101] correspond to scheme three (S3) with the data broken down into subsets by target type.

Experiment	Recall	Precision	Accuracy
Wasserstein	0.4644	0.9874	0.8260
FGW ($\alpha = 0.10$)	0.6336	0.9942	0.8813
FGW ($\alpha = 0.25$)	0.6204	0.9949	0.8772
FGW ($\alpha = 0.50$)	0.5934	0.9941	0.8684
FGW ($\alpha = 0.75$)	0.5916	0.9953	0.8681
FGW ($\alpha = 0.90$)	0.5845	0.9945	0.8657
Gromov-Wasserstein	0.5826	0.9934	0.8649

Table C.2: Further accuracy metrics corresponding to the results of our proposed method as seen in Table C.1. Note that we have a parameter setting of $\lambda = \mu = \gamma = \beta = 2^{-6}$, and $k = 0.04 * m$ where FGW stands for Fused Gromov-Wasserstein.

information. Their dataset was broken down into four subgroups based on what the drug is targeted to interact with: Enzymes (EN), Ion Channels (IC), G-Protein Coupled Receptors (GPCR), and Nuclear Receptors (GPCR). Since our dataset is not broken down based on drug targeting, we can most directly compare to the mean of these results, where we see that our experiments do indeed outperform theirs in terms of AUPRC. This highlights how Wasserstein-based similarity metrics convey feature information regarding drugs and can predict potential interactions.

Another interesting aspect of these results is the low recall rate among each of the similarity metrics tested, seen in Table C.2. A low recall means that our method is risk-averse, and will only predict a drug interaction when it is sure of it. This can be seen when analyzing the results of our best prediction with $\alpha = 0.1$ and $\lambda = \mu = \gamma = \beta = 2^{-6}$. We correctly predict whether a DDI occurs $197,615$ time, with only 443 false positive predictions. However, there are an additional $35,209$ false negatives that our method did not classify. Ideally, further tuning of the parameters $\lambda = \mu = \alpha = \beta$, and specifically sweeping the parameter space of these variables rather than setting them all equal, will improve this metric. A risk-averse method results in fewer resources being spent verifying drug interactions where there are not any, but will also leave many more unknown drug interactions hidden. Therefore, finding a balance where a method can make some errors while positively identifying a larger pool of interactions is important.

Finally, it should be noted that the small value of k needed for our latent feature size makes it so this problem can be projected onto a much smaller space than it originally existed. This saves computational cost, making it so this implementation performs quicker than other methods.

C.5 Conclusion

This work builds upon the triple matrix factorization [101] while using Wasserstein-based distances as our sole prediction feature for predicting drug-drug interaction. While the Wasserstein distance, which measures similarity at the atomic level, and the Gromov-Wasserstein distance, which account for structural similarity, each produce fine results individually, the best results can be achieved when combining them to generate the Fused Gromov-Wasserstein distance. With this metric, we can take into account both atomic and structural features to improve prediction performance.

Acknowledgments

The drugs depicted in Figures 2 and 4 were generated using molecular structure data and visualization tools from the PubChem website `https://pubchem.ncbi.nlm.nih.gov`.

Appendix D

Detailed Results for Single-Objective Optimization

This appendix presents the detailed results of the initial round of single-objective optimization on the surrogate models. A summary and discussion of these results are given in Sec. 4.2.1.

	Method	$o(q)$	t (s)	Re	$T_{\max}$	ψ	L_e	# iters	Compute Time (s)
Vorticity	Pattern Search	28.368	0.62	1,000	4,809	1.50	0.48	41	0.113
	Nelder-Mead	**28.870**	**0.62**	**1,000**	**4,675**	**1.35**	**0.48**	**170**	**0.072**
	Gradient Descent	28.870	0.62	1,000	4,681	1.35	0.48	1,344	12.236
	Trust Region	28.870	0.62	1,000	4,675	1.35	0.48	70	1.074
Vertical Flow in V_A	Pattern Search	0.011	0.54	398	5,857	1.63	0.49	7	0.017
	Nelder-Mead	**0.012**	**0.55**	**266**	**6,280**	**1.55**	**0.46**	**197**	**0.083**
	Gradient Descent	0.012	0.55	266	6,256	1.55	0.46	956	13.427
	Trust Region	0.012	0.55	266	6,280	1.55	0.46	2,100	31.351
Vertical Flow in V_B	Pattern Search	0.009	0.95	160	6,892	1.72	0.45	6	0.24
	Nelder-Mead	**0.009**	**0.98**	**167**	**6,650**	**1.90**	**0.45**	**256**	**0.102**
	Gradient Descent	0.009	0.99	167	6,633	1.90	0.45	218	2.814
	Trust Region	0.009	0.99	167	6,650	1.90	0.45	555	8.287

Table D.1: Optimization results for vorticity and vertical flow, all of which are maximized. The best-performing method for each metric is in bold.

	Method	$o(q)$	t (s)	Re	$T_{\max}$	ψ	L_e	# iters	Compute Time (s)
Tension-Based Energy	Pattern Search	$2.27\text{e}-4$	-	4	5,438	0.31	0.10	5	0.011
	Nelder-Mead	**0**	**-**	**7**	**2,481**	**0.49**	**0.37**	**132**	**0.069**
	Gradient Descent	0.115	-	565	10,000	0.25	0.10	5,655	45.033
	Trust Region	0	-	363	4,607	0.92	0.10	48	0.715
Tension-Based Power	Pattern Search	$6.59\text{e}-4$	-	1,000	3,168	1.42	0.20	8	0.019
	Nelder-Mead	**0**	**-**	**3**	**3,279**	**0.64**	**0.37**	**171**	**0.082**
	Gradient Descent	0.004	-	1,000	4,439	1.22	0.15	10,000	80.413
	Trust Region	0	-	1	4,381	0.97	0.26	40	0.605
Force-Based Energy	Pattern Search	25.149	-	1	2,290	1.00	0.29	6	0.014
	Nelder-Mead	**0**	**-**	**29**	**2,449**	**1.23**	**0.25**	**209**	**0.103**
	Gradient Descent	7.12e5	-	1	10,000	0.25	0.10	5,954	49.309
	Trust Region	0.052	-	64	5,930	1.08	0.10	40	0.598
Force-Based Power	Pattern Search	386.855	-	7	2,7450	1.54	0.26	5	0.011
	Nelder-Mead	**0**	**-**	**2**	**3,488**	**1.45**	**0.19**	**207**	**0.104**
	Gradient Descent	251.245	-	1	10,000	3	0.22	10,000	242.180
	Trust Region	0.018	-	18	3,784	0.89	0.10	39	0.605

Table D.2: Optimization results for energy and power, all of which are minimized. The best-performing method for each metric is in bold. Function outputs smaller than 1×10^{-6} have been rounded to 0.

Appendix E

Detailed Results for Multi-Objective Optimization

This appendix presents the detailed results of the initial round of multi-objective optimization on the surrogate models. A summary and discussion of these results are given in Sec. 4.2.2.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.10$	Pattern Search	-0.075	6.25	0.27	0.71	1,000	2,750	1.03	0.26	0.01
	Nelder-Mead	-0.074	6.55	0.27	0.81	1,000	1,799	1.16	0.28	0.13
	Gradient Descent	-0.086	4.09	0.32	0.53	1,000	10,000	0.25	0.20	33.61
	Trust Region	**-0.088**	**3.85**	**0.32**	**0.67**	**348**	**6,497**	**0.61**	**0.25**	**0.68**
$\omega = 0.25$	Pattern Search	-0.129	10.65	0.56	0.78	1,000	3,079	1.21	0.313	0.02
	Nelder-Mead	-0.118	11.34	0.52	0.65	1,000	3,762	0.83	0.34	0.17
	Gradient Descent	-0.158	8.47	0.69	0.65	348	6,497	0.64	0.35	525.50
	Trust Region	**-0.160**	**8.31**	**0.70**	**0.78**	**441**	**4,451**	**1.02**	**0.34**	**153.91**
$\omega = 0.50$	Pattern Search	-0.117	18.28	0.79	0.59	1,000	4,439	0.88	0.38	0.01
	Nelder-Mead	-0.106	16.08	0.69	0.63	1,000	4,096	0.89	0.37	0.13
	Gradient Descent	-0.124	15.42	0.81	0.61	595	4,213	0.78	0.42	3.73
	Trust Region	**-0.149**	**14.51**	**0.97**	**0.59**	**533**	**5,847**	**0.61**	**0.44**	**77.34**
$\omega = 0.65$	Pattern Search	-0.083	18.28	0.79	0.59	1,000	4,439	0.88	0.38	0.01
	Nelder-Mead	-0.080	17.68	0.75	0.63	1,000	4,162	0.91	0.38	0.07
	Gradient Descent	-0.097	17.22	0.90	0.64	537	4,054	1.01	0.40	0.84
	Trust Region	**-0.108**	**16.90**	**1.01**	**0.67**	**466**	**4,580**	**1.03**	**0.40**	**0.79**
$\omega = 0.75$	Pattern Search	-0.062	18.53	0.81	0.61	1,000	3,760	0.95	0.40	0.04
	Nelder-Mead	-0.060	18.48	0.78	0.63	1,000	4,193	0.92	0.38	0.20
	Gradient Descent	**-0.073**	**18.16**	**0.95**	**0.62**	**541**	**4,054**	**1.00**	**0.42**	**0.89**
	Trust Region	-0.071	18.20	0.93	0.62	808	3,364	1.21	0.40	1.28
$\omega = 0.90$	**Pattern Search**	**-0.021**	**27.41**	**1.31**	**0.65**	**1,000**	**4,904**	**1.25**	**0.44**	**0.04**
	Nelder-Mead	-0.018	27.21	1.19	0.62	1,000	4,493	1.16	0.44	0.06
	Gradient Descent	-0.018	27.30	1.20	0.62	1,000	4,580	1.16	0.44	4.83
	Trust Region	-0.018	27.21	1.19	0.62	1,000	4,493	1.16	0.44	0.31

Table E.1: Multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the magnitude of vorticity in V_A, and $g_2(q)$ being the tension-based energy. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.10$	Pattern Search	-0.043	12.58	0.08	0.68	1,000	3,702	1.23	0.31	0.01
	Nelder-Mead	-0.040	12.73	0.07	0.67	1,000	3,412	1.17	0.32	0.12
	Gradient Descent	-0.082	4.99	0.15	0.82	428	4,492	1.12	0.27	0.68
	Trust Region	**-0.085**	**4.48**	**0.15**	**0.79**	**424**	**4,492**	**1.04**	**0.27**	**1.02**
$\omega = 0.25$	Pattern Search	-0.081	13.99	0.13	0.72	1,000	3,339	1.14	0.35	0.02
	Nelder-Mead	-0.065	15.20	0.14	0.65	1,000	3,613	1.21	0.34	0.14
	Gradient Descent	**-0.159**	**8.39**	**0.35**	**0.85**	**369**	**4,607**	**1.37**	**0.35**	**2.65**
	Trust Region	-0.158	8.47	0.34	0.53	198	3,702	1.24	0.40	155.26
$\omega = 0.50$	Pattern Search	-0.084	17.56	0.28	0.69	1,000	3,248	1.30	0.38	0.02
	Nelder-Mead	-0.064	17.58	0.21	0.65	1,000	3,786	1.25	0.35	0.11
	Gradient Descent	-0.137	14.96	0.45	0.70	370	4,607	1.35	0.38	0.59
	Trust Region	**-0.175**	**13.59**	**0.57**	**0.64**	**320**	**3,280**	**1.17**	**0.47**	**155.04**
$\omega = 0.65$	Pattern Search	-0.066	19.62	0.32	0.63	1,000	4,525	1.42	0.35	0.04
	Nelder-Mead	-0.051	18.50	0.24	0.64	1,000	3,848	1.26	0.36	0.15
	Gradient Descent	**-0.099**	**17.15**	**0.46**	**0.60**	**374**	**4,607**	**1.35**	**0.39**	**0.62**
	Trust Region	-0.098	17.19	0.46	0.65	397	3,946	1.37	0.41	1.00
$\omega = 0.75$	Pattern Search	-0.047	19.62	0.32	0.63	1,000	4,525	1.42	0.35	0.05
	Nelder-Mead	-0.039	18.99	0.25	0.64	1,000	3,880	1.27	0.36	0.19
	Gradient Descent	**-0.077**	**18.06**	**0.50**	**0.61**	**375**	**4,607**	**1.36**	**0.40**	**0.35**
	Trust Region	-0.063	18.40	0.41	0.62	542	3,766	1.41	0.40	0.87
$\omega = 0.90$	Pattern Search	-0.017	21.67	0.35	0.64	1,000	4,054	1.41	0.38	0.04
	Nelder-Mead	-0.015	23.70	0.41	0.63	1,000	4,174	1.33	0.39	0.11
	Gradient Descent	**-0.033**	**19.24**	**0.55**	**0.62**	**442**	**4,492**	**1.14**	**0.41**	**0.48**
	Trust Region	-0.016	24.69	0.46	0.63	1,000	4,437	1.36	0.40	0.36

Table E.2: Multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the magnitude of vorticity in V_A, and $g_2(q)$ being the tension-based power. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.10$	Pattern Search	-0.070	0.004	0.23	0.36	282	4,845	0.53	0.31	0.17
	Nelder-Mead	-0.067	0.004	0.24	0.34	258	5,230	0.25	0.37	0.24
	Gradient Descent	**-0.080**	**0.003**	**0.29**	**0.30**	**484**	**4,054**	**0.93**	**0.25**	**0.40**
	Trust Region	-0.077	0.003	0.28	0.33	485	4,054	0.92	0.25	47.20
$\omega = 0.25$	Pattern Search	-0.142	0.006	0.62	0.37	483	6,081	0.47	0.38	0.02
	Nelder-Mead	-0.130	0.006	0.57	0.37	249	5,730	0.48	0.38	0.14
	Gradient Descent	**-0.152**	**0.005**	**0.66**	**0.40**	**232**	**3,627**	**0.68**	**0.43**	**0.59**
	Trust Region	-0.139	0.005	0.60	0.42	487	4,054	0.93	0.35	0.65
$\omega = 0.50$	**Pattern Search**	**-0.162**	**0.008**	**1.03**	**0.46**	**483**	**6,081**	**0.94**	**0.38**	**0.02**
	Nelder-Mead	-0.153	0.008	1.00	0.41	266	6,024	0.80	0.39	0.17
	Gradient Descent	-0.158	0.008	1.03	0.49	426	4,492	1.22	0.38	0.61
	Trust Region	-0.155	0.008	1.01	0.50	606	4,733	1.16	0.38	0.74
$\omega = 0.65$	**Pattern Search**	**-0.132**	**0.009**	**1.23**	**0.49**	**468**	**5,046**	**1.06**	**0.43**	**0.05**
	Nelder-Mead	-0.127	0.009	1.18	0.49	574	4,939	1.19	0.40	0.18
	Gradient Descent	-0.130	0.009	1.21	0.47	350	5,366	1.18	0.39	0.90
	Trust Region	-0.129	0.009	1.21	0.48	380	5,366	1.09	0.41	0.67
$\omega = 0.75$	**Pattern Search**	**-0.110**	**0.010**	**1.46**	**0.50**	**358**	**4,904**	**1.25**	**0.44**	**0.02**
	Nelder-Mead	-0.099	0.010	1.30	0.50	541	5,099	1.25	0.41	0.25
	Gradient Descent	-0.102	0.010	1.34	0.48	546	4,733	1.24	0.44	2.23
	Trust Region	-0.102	0.010	1.33	0.53	617	4,733	1.36	0.42	0.88
$\omega = 0.90$	**Pattern Search**	**-0.049**	**0.011**	**1.63**	**0.49**	**341**	**6,333**	**1.23**	**0.43**	**0.05**
	Nelder-Mead	-0.045	0.010	1.48	0.51	469	5,390	1.32	0.43	0.12
	Gradient Descent	-0.046	0.010	1.50	0.51	546	5,673	1.31	0.43	1.58
	Trust Region	-0.046	0.010	1.50	0.50	368	5,366	1.33	0.42	0.85

Table E.3: Multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the vertical flow in V_A, and $g_2(q)$ being the tension-based energy. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	$h_{\mathbf{AC}}$	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.10$	Pattern Search	-0.063	0.005	0.09	0.48	1,000	3,689	1.33	0.30	0.02
	Nelder-Mead	-0.058	0.005	0.10	0.45	661	3,539	1.17	0.33	0.15
	Gradient Descent	**-0.070**	**0.004**	**0.11**	**0.35**	**485**	**4,054**	**0.96**	**0.29**	**0.35**
	Trust Region	-0.067	0.004	0.12	0.36	486	4,054	0.96	0.29	50.53
$\omega = 0.25$	Pattern Search	-0.119	0.006	0.25	0.51	1,000	4,244	1.33	0.34	0.02
	Nelder-Mead	-0.110	0.007	0.24	0.48	590	3,912	1.28	0.35	0.11
	Gradient Descent	**-0.131**	**0.006**	**0.28**	**0.41**	**196**	**3,702**	**1.23**	**0.38**	**0.50**
	Trust Region	-0.131	0.006	0.28	0.41	196	3,702	1.23	0.38	2.20
$\omega = 0.50$	**Pattern Search**	**-0.146**	**0.008**	**0.48**	**0.60**	**508**	**5,187**	**1.68**	**0.36**	**0.02**
	Nelder-Mead	-0.127	0.008	0.41	0.51	490	4,451	1.41	0.38	0.13
	Gradient Descent	-0.136	0.008	0.44	0.49	425	4,492	1.23	0.39	0.37
	Trust Region	-0.129	0.008	0.42	0.49	362	4,607	1.41	0.37	0.68
$\omega = 0.65$	**Pattern Search**	**-0.123**	**0.009**	**0.57**	**0.55**	**398**	**5,857**	**1.63**	**0.36**	**0.01**
	Nelder-Mead	-0.108	0.009	0.50	0.52	435	4,755	1.48	0.39	0.28
	Gradient Descent	-0.110	0.009	0.51	0.51	360	4,607	1.43	0.40	0.70
	Trust Region	-0.119	0.009	0.55	0.51	426	4,492	1.27	0.42	0.63
$\omega = 0.75$	**Pattern Search**	**-0.096**	**0.010**	**0.59**	**0.61**	**344**	**5,014**	**1.66**	**0.40**	**0.05**
	Nelder-Mead	-0.085	0.010	0.56	0.53	400	4,963	1.52	0.40	0.37
	Gradient Descent	-0.091	0.010	0.59	0.54	361	4,607	1.41	0.42	0.48
	Trust Region	-0.089	0.010	0.58	0.51	451	4,580	1.41	0.42	0.78
$\omega = 0.90$	Pattern Search	-0.044	0.010	0.71	0.59	316	5,675	1.85	0.42	0.02
	Nelder-Mead	-0.039	0.010	0.64	0.55	352	5,291	1.57	0.41	0.14
	Gradient Descent	**-0.046**	**0.010**	**0.75**	**0.53**	**466**	**4,733**	**1.53**	**0.46**	**2.81**
	Trust Region	-0.040	0.010	0.65	0.55	270	5,256	1.56	0.42	0.76

Table E.4: Multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the vertical flow in V_A, and $g_2(q)$ being the tension-based power. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

Appendix F

Detailed Results for Optimization of Refined Surrogates

This appendix presents the detailed results from the second round of optimization performed using the refined surrogate models. A summary and discussion of these results are given in Sec. 5.3.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.25$	**Pattern Search**	**-0.110**	**12.51**	**0.48**	**0.60**	**1,000**	**5,187**	**0.25**	**0.36**	**0.01**
	Nelder-Mead	-0.098	12.80	0.43	0.65	768	4,096	0.25	0.39	0.21
	Gradient Descent	-0.109	12.03	0.47	0.64	489	4,145	0.34	0.40	14.57
	Trust Region	-0.100	12.71	0.43	0.68	727	4,324	0.25	0.38	0.72
$\omega = 0.65$	Pattern Search	-0.076	17.82	0.71	0.61	937	5,014	0.48	0.40	0.01
	Nelder-Mead	-0.075	17.84	0.70	0.62	903	5,463	0.34	0.41	0.30
	Gradient Descent	**-0.093**	**17.32**	**0.87**	**0.60**	**544**	**4,525**	**0.70**	**0.43**	**204.12**
	Trust Region	-0.080	17.70	0.74	0.62	844	4,525	0.68	0.40	0.93

Table F.1: Updated multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the magnitude of vorticity, and $g_2(q)$ being the tension-based energy. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.25$	Pattern Search	-0.126	10.74	0.25	0.65	1,000	1,682	0.54	0.43	0.02
	Nelder-Mead	-0.112	11.82	0.24	0.64	1,000	2,113	0.75	0.40	0.21
	Gradient Descent	**-0.156**	**8.47**	**0.34**	**0.65**	**367**	**2,829**	**0.76**	**0.42**	**1.21**
	Trust Region	-0.156	8.63	0.34	0.67	368	2,829	0.77	0.42	70.30
$\omega = 0.65$	Pattern Search	-0.099	17.17	0.46	0.65	1,000	3,364	1.08	0.43	0.02
	Nelder-Mead	-0.093	17.33	0.43	0.62	1,000	3,471	1.02	0.42	0.10
	Gradient Descent	**-0.101**	**17.11**	**0.47**	**0.60**	**852**	**3,310**	**1.09**	**0.44**	**19.23**
	Trust Region	-0.095	17.27	0.44	0.61	913	3,536	1.09	0.41	1.04

Table F.2: Updated multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the magnitude of vorticity, and $g_2(q)$ being the tension-based power. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.25$	Pattern Search	-0.140	0.005	0.61	0.34	348	6,497	0.34	0.38	0.01
	Nelder-Mead	-0.133	0.006	0.58	0.42	490	4,534	1.19	0.30	0.15
	Gradient Descent	**-0.142**	**0.005**	**0.62**	**0.44**	**183**	**4,439**	**0.94**	**0.34**	**0.57**
	Trust Region	-0.139	0.005	0.61	0.44	489	4,054	1.03	0.34	36.47
$\omega = 0.65$	**Pattern Search**	**-0.144**	**0.009**	**1.36**	**0.45**	**567**	**6,206**	**1.18**	**0.42**	**0.02**
	Nelder-Mead	-0.135	0.009	1.26	0.49	788	5,437	1.09	0.44	0.25
	Gradient Descent	-0.139	0.009	1.30	0.43	351	6,497	0.85	0.44	0.82
	Trust Region	-0.141	0.009	1.31	0.45	353	6,497	0.87	0.43	0.64

Table F.3: Updated multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the vertical flow in V_A, and $g_2(q)$ being the tension-based energy. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.

	Method	h_{AC}	g_1	g_2	t (s)	Re	$T_{\max}$	ψ	L_e	Compute Time (s)
$\omega = 0.25$	Pattern Search	-0.136	0.006	0.30	0.44	514	5,212	1.55	0.26	0.02
	Nelder-Mead	-0.129	0.006	0.28	0.42	455	4,736	1.42	0.28	0.07
	Gradient Descent	**-0.153**	**0.005**	**0.32**	**0.40**	**196**	**3,702**	**1.32**	**0.32**	**0.32**
	Trust Region	-0.146	0.005	0.32	0.39	488	4,054	1.04	0.32	0.24
$\omega = 0.65$	**Pattern Search**	**-0.157**	**0.008**	**0.71**	**0.50**	**141**	**4,904**	**1.35**	**0.44**	**0.03**
	Nelder-Mead	-0.148	0.009	0.69	0.50	427	4,327	1.26	0.47	0.14
	Gradient Descent	-0.154	0.009	0.72	0.50	205	4,213	1.31	0.48	3.20
	Trust Region	-0.150	0.009	0.70	0.49	190	4,439	1.34	0.47	0.70

Table F.4: Updated multi-objective optimization results using the augmented Chebyshev objective h_{AC} with $\rho = 0.05$, $g_1(q)$ being the vertical flow in V_A, and $g_2(q)$ being the tension-based power. In bold is the optimization method which performed best for each ω. Function outputs smaller than 1×10^{-6} have been rounded to 0.