

AFRL-AFOSR-VA-TR-2023-0118

Network-Based Feedback Control of Fluid Flows

Taira, Kunihiko
FLORIDA STATE UNIV TALLAHASSEE
874 TRADITIONS WAY
TALLAHASSEE, FL, 32306 - 0001
US

10/31/2022
Final Technical Report

DISTRIBUTION A: Distribution approved for public release.

Air Force Research Laboratory
Air Force Office of Scientific Research
Arlington, Virginia 22203
Air Force Materiel Command

DISTRIBUTION A: Distribution approved for public release.

REPORT DOCUMENTATION PAGE

PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ORGANIZATION.

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED	
20221031	Final	START DATE 20160930	END DATE 20200929

4. TITLE AND SUBTITLE
Network-Based Feedback Control of Fluid Flows

5a. CONTRACT NUMBER	5b. GRANT NUMBER	5c. PROGRAM ELEMENT NUMBER
	FA9550-16-1-0650	61102F
5d. PROJECT NUMBER	**5e. TASK NUMBER**	**5f. WORK UNIT NUMBER**

6. AUTHOR(S)
Kunihiko Taira

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
FLORIDA STATE UNIV TALLAHASSEE
874 TRADITIONS WAY
TALLAHASSEE, FL 32306 - 0001
US

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)	11. SPONSOR/MONITOR'S REPORT NUMBER(S)
Air Force Office of Scientific Research 875 N. Randolph St. Room 3112 Arlington, VA 22203	AFRL/AFOSR RTA1	AFRL-AFOSR-VA-TR-2023-0118

12. DISTRIBUTION/AVAILABILITY STATEMENT
A Distribution Unlimited: PB Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
Unsteady fluid flows have complex dynamics due to the nonlinear interactions amongst vortical elements. In this report, a network-theoretic framework is developed to describe vortical and modal (coherent structure) interactions in unsteady fluid flows. A sparsifieddynamics model and a networked-oscillator model describe the complex dynamics in fluid flows in terms of vortical and modal networks, respectively. Based on the characterized network interactions, model-based feedback control laws are established, particularly for controlling the flow unsteadiness. Furthermore, to characterize model-free feedback control laws for suppressing flow separation in turbulent flows, a data-driven approach leveraging unsupervised clustering is developed. This approach alters the Markov transition dynamics of fluid flow trajectories in an optimal manner using a cluster-based control strategy. To describe
vortical interactions, dense fluid flow graphs are constructed using discrete point vortices as nodes and induced velocity as edge weights. Sparsification techniques are then employed on these graph representations based on spectral graph theory to construct sparse graphs of the overall vortical interactions which maintain similar spectral properties as the original setup. Utilizing the sparse vortical graphs, a sparsified-dynamics model is developed which
drastically reduces the computational cost to predict the dynamical behavior of vortices, sharing characteristics of reduced-order models.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U	UU	174

19a. NAME OF RESPONSIBLE PERSON	19b. PHONE NUMBER *(Include area code)*
GREGG ABATE	425-1779

DISTRIBUTION A: Distribution approved for public release.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.
PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
02/11/2021	Final	09/30/2016 - 09/29/2020

4. TITLE AND SUBTITLE
Network-based feedback control of fluid flows

5a. CONTRACT NUMBER

5b. GRANT NUMBER
FA9550-16-1-0650

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)
Taira, Kunihiko; Brunton, Steven, L; Shih, Chiang; Nair, Aditya G.; Yeh, Chi-An; Bai, Zhe; and Callaham, Jared

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Florida State University, Sponsored Research Services,
874 Traditions Way, Tallahassee, FL 32310
University of California, Los Angeles, CA 90095, USA
University of Washington, Seattle, WA, 98195, USA

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Air Force Office of Scientific Research
875 Randolph Street
Arlington, VA 22203

10. SPONSOR/MONITOR'S ACRONYM(S)
AFOSR

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Distribution A - Approved for Public Release

13. SUPPLEMENTARY NOTES

14. ABSTRACT
The effort from this project led to the development of novel network-theoretic and data-based frameworks for analyzing and controlling fluid flows. The framework incorporates advanced mathematical principles from network science, graph theory and dynamical systems to extract fundamental interactions in fluid flows. On manipulating these interactions, wake unsteadiness in bluff body flow is reduced leading to drag reduction. Furthermore, data-based methods are developed to deduce optimal feedback control laws for post-stall separated flows. The network-theoretic and data-based approaches provide insights on fundamental interactions in fluid flows which pave the way for design of novel flow control strategies.

15. SUBJECT TERMS
Unsteady Aerodynamics, Networks, Flow Control

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Kunihiko Taira
					19b. TELEPHONE NUMBER *(Include area code)*
U	U	U		172	310-825-7660

INSTRUCTIONS FOR COMPLETING SF 298

1. REPORT DATE. Full publication date, including day, month, if available. Must cite at least the year and be Year 2000 compliant, e.g. 30-06-1998; xx-06-1998; xx-xx-1998.

2. REPORT TYPE. State the type of report, such as final, technical, interim, memorandum, master's thesis, progress, quarterly, research, special, group study, etc.

3. DATE COVERED. Indicate the time during which the work was performed and the report was written, e.g., Jun 1997 - Jun 1998; 1-10 Jun 1996; May - Nov 1998; Nov 1998.

4. TITLE. Enter title and subtitle with volume number and part number, if applicable. On classified documents, enter the title classification in parentheses.

5a. CONTRACT NUMBER. Enter all contract numbers as they appear in the report, e.g. F33315-86-C-5169.

5b. GRANT NUMBER. Enter all grant numbers as they appear in the report. e.g. AFOSR-82-1234.

5c. PROGRAM ELEMENT NUMBER. Enter all program element numbers as they appear in the report, e.g. 61101A.

5e. TASK NUMBER. Enter all task numbers as they appear in the report, e.g. 05; RF0330201; T4112.

5f. WORK UNIT NUMBER. Enter all work unit numbers as they appear in the report, e.g. 001; AFAPL30480105.

6. AUTHOR(S). Enter name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. The form of entry is the last name, first name, middle initial, and additional qualifiers separated by commas, e.g. Smith, Richard, J, Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES). Self-explanatory.

8. PERFORMING ORGANIZATION REPORT NUMBER. Enter all unique alphanumeric report numbers assigned by the performing organization, e.g. BRL-1234; AFWL-TR-85-4017-Vol-21-PT-2.

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES). Enter the name and address of the organization(s) financially responsible for and monitoring the work.

10. SPONSOR/MONITOR'S ACRONYM(S). Enter, if available, e.g. BRL, ARDEC, NADC.

11. SPONSOR/MONITOR'S REPORT NUMBER(S). Enter report number as assigned by the sponsoring/ monitoring agency, if available, e.g. BRL-TR-829; -215.

12. DISTRIBUTION/AVAILABILITY STATEMENT. Use agency-mandated availability statements to indicate the public availability or distribution limitations of the report. If additional limitations/ restrictions or special markings are indicated, follow agency authorization procedures, e.g. RD/FRD, PROPIN, ITAR, etc. Include copyright information.

13. SUPPLEMENTARY NOTES. Enter information not included elsewhere such as: prepared in cooperation with; translation of; report supersedes; old edition number, etc.

14. ABSTRACT. A brief (approximately 200 words) factual summary of the most significant information.

15. SUBJECT TERMS. Key words or phrases identifying major concepts in the report.

16. SECURITY CLASSIFICATION. Enter security classification in accordance with security classification regulations, e.g. U, C, S, etc. If this form contains classified information, stamp classification level on the top and bottom of this page.

17. LIMITATION OF ABSTRACT. This block must be completed to assign a distribution limitation to the abstract. Enter UU (Unclassified Unlimited) or SAR (Same as Report). An entry in this block is necessary if the abstract is to be limited.

DISTRIBUTION A: Distribution approved for public release.

Final report: Network-based feedback control of fluid flows

for the Air Force Office of Scientific Research
(FA9550-16-1-0650, Sep 30, 2016 to Sep 29, 2020)
Program Managers: Drs. Gregg Abate & Douglas Smith

Authors: Kunihiko Taira†[1,2], Steven L. Brunton‡[3], Chiang Shih¶[2], Aditya G. Nair[2,3], Chi-An Yeh[1,2], Zhe Bai[3], Jared Callaham[3]

[1]Department of Mechanical and Aerospace Engineering, University of California, Los Angeles, CA 90095, USA

[2]Department of Mechanical Engineering, Florida State University, Tallahassee, FL 32310, USA

[3]Department of Mechanical Engineering, University of Washington, Seattle, WA, 98195, USA

† PI, ktaira@seas.ucla.edu
‡ co-PI, sbrunton@uw.edu
¶ PI/Adminstrative POC, shih@eng.famu.fsu.edu

DISTRIBUTION A: Distribution approved for public release.

CONTENTS

1. Abstract 4
2. Resulting Publications and Presentations 6
2.1. Journal Publications 6
2.2. Conference Publications 7
3. Conference Talks 7
4. Introduction 9
4.1. Motivation and background 9
4.2. Outline 10
5. Network Analysis and Graph Theory 13
5.1. Network-theoretic framework 13
5.2. Spectral sparsification 15
6. Vortical Interaction Networks 18
6.1. Problem setup 18
6.2. Representing vortex interactions on a graph 19
6.3. Sparsification of vortex interactions 20
6.4. Sparsified-dynamics model 22
6.5. Conservation properties 29
6.6. Comparison with reduced-order model 30
7. Turbulent Flow Networks 33
7.1. Problem setup 33
7.2. Network representation 33
7.3. Network characterization 35
7.4. Network dynamics 36
7.5. Resilience of turbulence networks 37
8. Modal Interaction Networks 40
8.1. Modal oscillators 40
8.2. Networked oscillator representation 43
8.3. Application to cylinder flow 44
8.4. Computational setup 45
8.5. Baseline (unperturbed flow) 45
8.6. Perturbed flow 47
9. Interaction-based Feedback Control 57
9.1. Disturbance rejection 59
9.2. Total oscillation control 60
10. Markov Transition Network Control 64
10.1. Cluster-based control framework 64
10.2. Actuator setup 66
10.3. Cluster-based discretization 67
10.4. Optimized feedback control design 68
10.5. Separation control of the flow around a two-dimensional airfoil 71
10.6. Baseline feature space clustering 71
10.7. Optimized feedback control 75
10.8. Separation control of the three-dimensional flow around an airfoil 79
10.9. Baseline feature space clustering 80
10.10. Optimized feedback control 82
11. Phase-based control of periodic fluid flows 88
11.1. Introduction 88
11.2. Formulation 90

DISTRIBUTION A: Distribution approved for public release.

11.3. Results 94
11.4. Conclusion 105
11.5. Appendix: Model-predictive control (MPC) 106
12. Randomized Dynamic Mode Decomposition 108
12.1. Related Work 109
12.2. Assumptions, Limitations, Applications, and Extensions 110
12.3. Technical Preliminaries 111
12.4. The Singular Value Decomposition 111
12.5. Dynamic Mode Decomposition 112
12.6. Randomized Methods for Linear Algebra 115
12.7. Randomized Dynamic Mode Decomposition 119
12.8. Numerical Results 122
12.9. Conclusion 127
13. Supervised Machine Learning for Flow Regression 129
13.1. Introduction 129
13.2. Machine learning methods 130
13.3. Examples 137
13.4. Concluding remarks 151
14. Concluding Remarks 154
14.1. Conclusions 154
14.2. Future work 156
15. Acknowledgements 158

DISTRIBUTION A: Distribution approved for public release.

1. Abstract

Unsteady fluid flows have complex dynamics due to the nonlinear interactions amongst vortical elements. In this report, a network-theoretic framework is developed to describe vortical and modal (coherent structure) interactions in unsteady fluid flows. A sparsified-dynamics model and a networked-oscillator model describe the complex dynamics in fluid flows in terms of vortical and modal networks, respectively. Based on the characterized network interactions, model-based feedback control laws are established, particularly for controlling the flow unsteadiness. Furthermore, to characterize model-free feedback control laws for suppressing flow separation in turbulent flows, a data-driven approach leveraging unsupervised clustering is developed. This approach alters the Markov transition dynamics of fluid flow trajectories in an optimal manner using a cluster-based control strategy.

To describe vortical interactions, dense fluid flow graphs are constructed using discrete point vortices as nodes and induced velocity as edge weights. Sparsification techniques are then employed on these graph representations based on spectral graph theory to construct sparse graphs of the overall vortical interactions which maintain similar spectral properties as the original setup. Utilizing the sparse vortical graphs, a sparsified-dynamics model is developed which drastically reduces the computational cost to predict the dynamical behavior of vortices, sharing characteristics of reduced-order models. The model retains the nonlinearity of the interactions and also conserves the invariants of discrete vortex dynamics.

The network structure of vortical interactions in two-dimensional incompressible homogeneous turbulence is then characterized. The strength distribution of the turbulence network reveals an underlying scale-free structure that describes how vortical structures are interconnected. Strong vortices serve as network hubs with smaller and weaker eddies predominantly influenced by the neighboring hubs. The time evolution of the fluid flow network informs us that the scale-free property is sustained until dissipation overtakes the flow physics. The types of perturbations that turbulence network is resilient against is also examined.

To describe modal interactions in fluid flows, a networked-oscillator-based analysis is performed. The analysis examines and controls the transfer of kinetic energy for periodic bluff body flows. The dynamics of energy fluctuations in the flow field are described by a set of oscillators defined by conjugate pairs of spatial POD modes. To extract the network of interactions among oscillators, impulse responses of the oscillators to amplitude and phase perturbations are tracked. Using linear regression techniques, a networked oscillator model is constructed that reveals energy exchanges among the modes. In particular, a large collection of system responses are aggregated to capture the general network structure of oscillator interactions. The present networked oscillator model describes the modal perturbation dynamics more accurately than the empirical Galerkin reduced-order model. The linear network model for nonlinear dynamics is subsequently utilized to design a model-based feedback controller. The controller suppresses the modal fluctuations and amplitudes that result in wake unsteadiness leading to drag reduction. The strength of the approach is demonstrated for a canonical example of two-dimensional unsteady flow over a circular cylinder. The network-based formulation enables the characterization and control of modal interactions to control fundamental energy transfers in unsteady bluff body flows.

Finally, unsupervised clustering and data-driven optimization of coarse-grained control laws is leveraged to manipulate post-stall separated flows. Optimized feedback control laws are deduced in high-fidelity simulations in an automated, model-free manner. The

DISTRIBUTION A: Distribution approved for public release.

approach partitions the baseline flow trajectories into clusters, which corresponds to a characteristic coarse-grained phase in a low-dimensional feature space constituted by feature variables (sensor measurements). The feedback control law is then sought for each and every cluster state which is iteratively evaluated and optimized to minimize aerodynamic power and actuation power input. The control optimally transforms the Markov transition network associated with the baseline trajectories to achieve desired performance objectives. The approach is applied to two and three-dimensional separated flows over a NACA 0012 airfoil at an angle of attack of 9°, Reynolds number $Re = 23,000$ and free-stream Mach number $M_\infty = 0.3$. The optimized control law minimizes power consumption for flight enabling flow to reach a low-drag state. The analysis provides insights for feedback flow control of complex systems characterizing global cluster-based control laws based on a data-driven, low-dimensional characterization of fluid flow trajectories.

The effort from this project led to the development of novel network-theoretic and data-based frameworks for analyzing and controlling fluid flows. The framework incorporates advanced mathematical principles from network science, graph theory and dynamical systems to extract fundamental interactions in fluid flows. On manipulating these interactions, wake unsteadiness in bluff body flow is reduced leading to drag reduction. Finally, data-based methods are developed to deduce optimal feedback control laws for post-stall separated flows. The network-theoretic and data-based approaches provide insights on fundamental interactions in fluid flows which pave the way for design of novel flow control strategies. In what follows, we summarize our major findings from this work. Since a large number of papers have been produced from this project, we limit the report to highlight only the key findings. The full list of papers can be found in section 2. Some work contained therein were also supported by the Army Research Office and the Office of Naval Research in addition to the Air Force Office of Scientific Research.

DISTRIBUTION A: Distribution approved for public release.

2. Resulting Publications and Presentations

2.1. Journal Publications

- K. Fukami, R. Maulik, N. Ramachandra, K. Fukagata, & K. Taira, "Global field reconstruction from sparse sensors with Voronoi tessellation-assisted deep learning," in review, 2021. (Fukami *et al.* 2021*b*)
- C. A. Yeh, M. G. Meena & K. Taira, "Network broadcast analysis and control of turbulent flows," *Journal of Fluid Mechanics*, 915, A15, 2021. (Yeh *et al.* 2021)
- K. Fukami, K. Fukagata, & K. Taira, "Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows," *Journal of Fluid Mechanics*, 909, A9, 2021. (Fukami *et al.* 2021*a*)
- K. Taira, S. L. Brunton, S. T. M. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley, "Modal Analysis of Fluid Flows: An Overview," AIAA Journal, 55(12), 4013-4041, 2017. (Taira *et al.* 2017*a*)
- K. Taira, M. S. Hemati, S. L. Brunton, Y. Sun, K. Duraisamy, S. Bagheri, S. T. M. Dawson, and C.-A. Yeh, "Modal Analysis of Fluid Flows: Applications and Outlook," AIAA Journal, 58(3), 998-1022, 2020. (Taira *et al.* 2020*b*)
- K. Taira, M. S. Hemati, and L. S. Ukeiley, "Modal Analysis of Fluid Flows: Introduction to the Virtual Collection," AIAA Journal, 58(3), 991-993, 2020. (**?**)
- M. A. Khodkar & K. Taira, "Phase-synchronization properties of laminar cylinder wake for periodic external forcings," *Journal of Fluid Mechanics*, 904, R1, 2020. (Khodkar & Taira 2020)
- M. A. Khodkar, J. T. Klamo, & K. Taira, "Phase-locking of laminar wake to periodic vibrations of a circular cylinder," in review, 2020.
- R. Maulik, K. Fukami, N. Ramachandra, K. Fukagata, and K. Taira, "Probabilistic Neural Networks for Fluid Flow Model-Order Reduction and Data Recovery," *Physical Review Fluids*, 5:104401, 2020. (Maulik *et al.* 2020*a*)
- Z. Bai, E. Kaiser, J. L. Proctor, B. W. Brunton, J. N. Kutz, and S. L. Brunton, "Dynamic mode decomposition for compressive system identification," *AIAA Journal*, 58(2):561–574, 2020.
- K. Fukami, K. Fukagata, & K. Taira, "Assessment of supervised machine learning methods for fluid flows," *Theoretical and Computational Fluid Dynamics*, 34, p. 497-519, 2020. (Fukami *et al.* 2020)
- Z. Bai, N. B. Erichson, M. G. Meena, K. Taira, and S. L. Brunton, "Randomized methods to characterize large-scale vortical flow networks: A scalable framework for low-rank approximation." *PLOS One*, 14(11), e0225265, 2019. (Bai *et al.* 2019)
- K. Fukami, K. Fukagata, & K. Taira, "Super-resolution reconstruction of turbulent flows with machine learning," *Journal of Fluid Mechanics*, 870, p. 106-120, 2019. (Fukami *et al.* 2018)
- K. Fukami, K. Fukagata, & K. Taira, "Applications of a machine-learned super-resolution algorithm to two-dimensional flow fields," *Nagare-Journal of Japan Society of Fluid Mechanics*, 38(6), p. 395-398, 2019.
- J. L. Callaham, K. Maeda, and S. L. Brunton, "Robust flow reconstruction from limited measurements via sparse representation," *Physical Review Fluids*, 4:103907, 2019.
- N. B. Erichson, L. Mathelin, J. N. Kutz, and S. L. Brunton, "Randomized dynamic mode decomposition," *SIAM Journal on Applied Dynamical Systems*, 18(4):1867–1891, 2019.
- A. G. Nair, B. R. Noack, C. A. Yeh, E. Kaiser, S. L. Brunton & K. Taira, "Cluster-based feedback control of turbulent post-stall separated flows," *Journal of Fluid Mechanics*, 875, 345-375, 2019. (Nair *et al.* 2019)

DISTRIBUTION A: Distribution approved for public release.

• J. C. Loiseau and S. L. Brunton, "Constrained sparse Galerkin regression," *Journal of Fluid Mechanics*, 838:42–67, 2018.

• E. Kaiser, M. Morzynski, G. Daviller, J. N. Kutz, B. Brunton, and S. L. Brunton, "Sparsity enabled cluster reduced-order modeling for control," *Journal of Computational Physics*, 352:388–409, 2018.

• A. G. Nair, S. L. Brunton, & K. Taira, "Networked oscillator-based modeling and control of unsteady wake flows," *Physical Review E*, 97 (063107), 2018. (Nair *et al.* 2018)

• M. Gopalakrishnan Meena, A. G. Nair, & K. Taira, "Network community-based model reduction for vortical flows," *Physical Review E*, 97 (063103), 2018. (Gopalakrishnan Meena *et al.* 2018)

• N. M. Mangan, J. N. Kutz, S. L. Brunton, and J. L. Proctor, "Model selection for dynamical systems via sparse regression and information criteria," *Proceedings of the Royal Society A*, 473: 1–16, 2017.

2.2. Conference Publications

• K. Fukami, R. Maulik, N. Ramachandra, K. Fukagata, & K Taira, "Probabilistic neural network-based reduced-order surrogate for fluid flows," Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020), Vancouver, Canada, 2020.

• K. Fukami, K. Fukagata, & K. Taira, "Super-resolution analysis with machine learning for low-resolution flow data," 11th International Symposium on Turbulence and Shear Flow Phenomena, Southampton, UK, 2019.

• K. Fukami, K. Fukagata, & K. Taira, "Application of machine-learned super resolution for two-dimensional flow," Japan Society of Fluid Mechanics Annual Meeting, Tokyo, Japan, 2019 (in Japanese)

• K. Fukami, K. Fukagata, & K. Taira, "Machine-learned super-resolution analysis of three-dimensional turbulent channel flow," Japan Society of Mechanical Engineers Fluids Engineering Conference, Aichi, Japan, 2019 (in Japanese)

• A. G. Nair, K. Taira, & S. L. Brunton, "Oscillator network based control of unsteady fluid flows", *20th World Congress of the International Federation of Automatic Control*, Toulouse, France, July 9-14, 2017.

• K. Taira, A. G. Nair, and S. L. Brunton,"Complex Network Analysis of Unsteady Fluid Flows," *XXIV International Congress of Theoretical and Applied Mechanics*, Montreal, August 21-26, 2016.

• K. Taira, A. G. Nair, and S. L. Brunton, "Vortex Interaction Analysis using Complex Network Framework," *Annual Meeting of the Japan Society of Fluid Mechanics*, Nagoya, 2016 (in Japanese).

3. Conference Talks

• C.-A. Yeh, M. Gopalakrishnan Meena, and K. Taira, "Time-evolving network analysis of two-dimensional turbulence," IUTAM Symposium on Laminar-Turbulent Transition, London, United Kingdom, Sep. 2-6, 2019.

• K. Fukami, K. Fukagata, and K. Taira, "Super-resolution analysis with machine learning for low-resolution flow data," 11th International Symposium on Turbulence and Shear Flow Phenomena, Southampton, UK, Jul. 30 - Aug. 2, 2019.

• A. G. Nair "Data-driven modeling and control strategies," *Telluride Neuromorphic Cognition Engineering Workshop*, Telluride, June 30-July 19, 2019.

• A. G. Nair, B. R. Noack, C. A. Yeh, E. Kaiser, S. L. Brunton & K. Taira, 'Design of feedback control laws for turbulent post-stall separated flows," *SIAM Conference on Applications of Dynamical Systems*, Snowbird, May 19-23, 2019.

DISTRIBUTION A: Distribution approved for public release.

• S. L. Brunton, J. N. Kutz, J.-Ch. Loiseau, and B. R. Noack, "Interpretable Nonlinear Models of Unsteady Flow Physics", SIAM DS, Snowbird, UT, May 2019.

• K. Fukami, K. Fukagata, and K. Taira, "Image-based super-resolution analysis with machine learning for two-dimensional turbulence," 13th Southern California Flow Physics Symposium, Santa Barbara, CA, Apr. 20, 2019.

• A. G. Nair, K. Taira, & S. L. Brunton, "Data-driven feedback control strategies for unsteady flows," *SIAM Conference on Computational Science and Engineering*, Spokane, Feb. 25-Mar. 1, 2019.

• S. L. Brunton, "Data-Driven Discovery and Control of Complex Systems: Uncovering Interpretable and Generalizable Nonlinear Models", SIAM CSE, Spokane, WA, February 2019.

• A. Mendible, S. L. Brunton, and J. N. Kutz, "The space-time problem with model reduction for traveling waves," SIAM CSE, Spokane, WA, February 2019.

• A. G. Nair, B. R. Noack, C. A. Yeh, E. Kaiser, S. L. Brunton & K. Taira, "Optimal cluster-based feedback control for separated flows," *71th Annual Meeting of the APS Division of Fluid Dynamics*, Atlanta, Nov. 18-20, 2018.

• S. L. Brunton, J. N. Kutz, J.-Ch. Loiseau, and B. R. Noack, "Interpretable Models of Nonlinear Flow Physics," APS DFD, Atlanta, GA, November 2018.

• Z. Bai, N. B. Erichson, M. G. Meena, K. Taira, and S. L. Brunton, "Sparse and randomized sampling methods for scalable turbulent flow networks," APS DFD, Atlanta, GA, November 2018.

• J. Callaham, K. Maeda and S. L. Brunton, "Robust reconstruction of flow fields from limited measurements," APS DFD, Atlanta, GA, November 2018.

• K. Taira, A. G. Nair, and C.-A. Yeh, "Modal Analysis Based Characterization, Modeling, and Control of Fluid Flows," *JSASS 50th Fluid Dynamics Conference/36th Aerospace Numerical Simulation Symposium*, Miyazaki, Japan July 4-6, 2018.

• A. G. Nair, B. R. Noack, C.-A. Yeh, E. Kaiser, S. L. Brunton, and K. Taira, "Local characterization of feedback control laws for unsteady wake flows," *SIAM Annual Meeting*, Portland, July 9-13, 2018.

• A. G. Nair, K. Taira, and S. L. Brunton, "On extracting vortical and modal networks in unsteady fluid flows," *2nd Machine Learning Control Workshop*, Valenciennes, July 5-6, 2017.

• A. G. Nair, K. Taira, & S. L. Brunton, "Network-theoretic approaches for characterizing, modeling, and controlling fluid flows," *AIAA Aviation and Aeronautics Forum and Exposition*, Denver, June 5-9, 2017.

• A. G. Nair, K. Taira, & S. L. Brunton, "Data-based extraction of modal interaction networks," *SIAM Conference on Computational Science and Engineering*, Atlanta, Feb. 27-Mar. 3, 2017.

• M. Gopalakrishnan Meena, A. G. Nair, and K. Taira, "Network Representation and Analysis of Bluff Body Wake," *SIAM Conference on Computational Science and Engineering*, Atlanta, Feb. 27-Mar. 3, 2017.

• A. G. Nair, M. Gopalakrishnan Meena, and K. Taira, "Vortical and Modal Network Analysis of Unsteady Cylinder Wake," *69th Annual Meeting of the APS Division of Fluid Dynamics*, Portland, Nov. 20-22, 2016.

DISTRIBUTION A: Distribution approved for public release.

4. Introduction

4.1. Motivation and background

In the field of fluid mechanics, there have been extensive studies performed to capture the behavior of unsteady fluid flows that have rich dynamics generated from the nonlinear interactions that take place over a broad range of spatial and temporal scales. The nonlinear and multi-scale nature of the interactions along with the associated high-dimensionality is what makes fluid mechanics such an exciting and challenging field. One of the most complex unsteady flow features is turbulence, which is ubiquitous in nature, characterized by the resulting chaos from the multitude of fluid flow interactions Davidson (2004); Holmes *et al.* (2012*a*). However, quantifying these interactions in a tractable manner has remained a challenge to date. We can view these interactions to be present between vortical elements in physical space or between modal structures in wave space.

In the review by Saffman & Baker Saffman & Baker (1979), the importance of vortex interactions was highlighted. In particular, the idea of representing turbulence as a superposition of interacting vortices was emphasized. These vortical interactions are closely tied with the generation of coherent structures in fluid flows Roshko (1976). The interactions between the modes in unsteady fluid flows can be captured by reduced-order models while dramatically reducing the computational expense to model fluid flows of interest (Aubry *et al.* 1988; Noack *et al.* 2003*a*). The projection of the Navier–Stokes equations onto the modal basis results in an empirical Galerkin formulation (Holmes *et al.* 2012*a*). However, mode deformations and truncation of energy cascade in unsteady fluid flows pose a significant challenge in developing accurate reduced-order models (Lassila *et al.* 2014; Brunton & Noack 2015*a*).

The strong nonlinear dynamics in unsteady fluid flows are characterized by spatio-temporal oscillations. In flows past bluff bodies, oscillatory behavior of the flows is revealed through shedding of coherent vortices observed in the wake. Such periodic shedding generates unsteady forces on the body which can lead to detrimental increase in drag associated structural fatigue due to the emergence of flow-induced vibrations (Williamson & Govardhan 2004; Sarpkaya 2004). In the work of Roshko (1954), the relationship between form drag and vortex shedding was explored in detail. It was demonstrated that unsteady force oscillations and drag can be reduced by mitigating the wake unsteadiness. Since then, a myriad of studies using active and passive flow control strategies have focussed on controlling bluff-body wake vortex shedding and the resulting unsteady forces, summarized in a review by Choi *et al.* (2008). Although there have been tremendous breakthroughs in applying flow control techniques for drag reduction, only a few studies make use of the fundamental energy transfer mechanisms and interactions in unsteady fluid flows and controlling the flow unsteadiness therein. Thus, analysis and control of fluid flows necessitates a knowledge of the fundamental interactions involved in fluid flows.

Network science provides a unique framework for analyzing complex fluid flows in terms of a network of vortical and modal interactions. A network describes how components are linked to one another through a web of interactions. We can represent the components by points (nodes) and the interactions by lines (edges) through a mathematical abstraction. The structure comprised of these nodes and edges is called a graph, which has been studied in detail in the field of graph theory Bollobás (1998). Network analysis is concerned with the study of graphs as well as the interaction and evolution of the variables of interest on graphs over space and time Newman (2010). The framework developed in network analysis and graph theory can provide insights into how the structure of a

DISTRIBUTION A: Distribution approved for public release.

network can influence the overall dynamics taking place on the network. With the network structure established, we can also analyze resilience of network-based activities (Albert *et al.* 2000). Characterization of these networks has helped answer many fundamental questions of practical relevance and device new standards and protocols for efficient functioning of many systems. These analysis techniques are founded on graph theory, dynamical systems, and operator theory but place unique emphasis on interactions and connectivity amongst the elements that establish a network. Modifying the network interactions in fluid flows can be transformative in developing model-based flow control strategies.

Flow control design by first-principle models provides an accurate control framework, however, the computational burden is prohibitively large in terms of resources, processing time, and data storage, even for simple geometries. In contrast, data-driven control of fluid flows is transforming, driven by the confluence of big data, advanced algorithms in machine learning, and modern computational hardware (Brunton & Noack 2015*a*). Instead of learning models and then performing flow control, especially for high Reynolds number flows, there is a need for model-free control of fluid flows leveraging data-based techniques.

Our primary focus lies in developing a network-based framework for analyzing fluid flow interactions. Thereafter, we focus on developing model-based feedback control laws modifying the network of interactions. Finally, model-free alternatives for data-driven feedback control are developed.

4.2. Outline

In section 5, network analysis and graph theory are briefly introduced laying the principle framework for the rest of the Sections. In addition to discussing the framework, the notion of graph sparsification is introduced which is used to sparsify the interactions between components of the graph.

In Section 6, network analysis and graph theory are extended to describe discrete point-vortex dynamics. Here, we consider representing vortices with nodes and the interactions amongst the vortices with edges. By utilizing the network-theoretic framework to study vortex dynamics, we highlight the connections (edges) that the vortices have in the flow field. Such analysis emphasizes how a collection of vortices influence each other through a causal point of view on a network structure. We believe that the present study can provide an alternative tool to analyze how vortices or flow structures interact in the flow field and support the development of interaction-based models to capture unsteady vortex dynamics. Furthermore, we consider the use of graph sparsification for sparsifying the interaction between the point vortices. These models capture the essential physics of unsteady fluid flow by keeping the nodes intact and reduce the number of edges maintaining the dimensionality of the original system unlike reduced-order models. The removal of edges can drastically reduce the computational cost to model the full dynamical behavior, sharing the same spirit as reduced-order models. We refer to the dynamical model based on sparsifying the interactions as *sparsified dynamics* model.

In Section 7, the underlying structure and organization of turbulent flows is highlighted using the vortical network based perspective illuminated in Section 6. In this Section, we demonstrate that two-dimensional decaying isotropic turbulence Kraichnan & Montgomery (1980); McWilliams (1984); Benzi *et al.* (1990); Benzi & Colella (1992); Davidson (2004); Boffetta & Ecke (2012) has a scale-free network structure reminiscent of other networks found in nature Barabási & Albert (1999); Caldarelli (2007); Barrat *et al.* (2004*b*). We show that the variations in the strength of interactions in the extracted turbulence network follows a scale-free distribution. This scale-free characterization of

DISTRIBUTION A: Distribution approved for public release.

turbulence helps us in capturing key players in the network and gives us insights on how it may be possible to control the dynamics of turbulent vortices Liu *et al.* (2011); Farazmand *et al.* (2011); Brunton & Noack (2015*b*). Also, once the turbulence network structure is obtained, dynamics and resilience studies are performed.

In Section 8, we leverage the use of network analysis to examine how modal structures in unsteady fluid flows interact to distribute energy in non-equilibrium conditions. Utilizing modal decomposition techniques in conjunction with networked oscillator models, we highlight the interactive physics involved in unsteady fluid flows. In this Section, we examine the nonlinear energy transfer between modes and construct a tractable model for amplitude and phase perturbations in unsteady fluid flows. To accomplish these goals, mode pairs describing individual coherent structures in baseline (unforced and unperturbed) time-periodic flows are considered as a set of oscillators (self-sustaining components). A network representation with the oscillators as nodes and interactions between them as edges is constructed. Perturbation energy is introduced in the unsteady fluid flow to examine its propagation among the modes which are tracked using a networked oscillator model. The associated network structure is extracted using a simple linear regression procedure (Brunton *et al.* 2016*b*). With the network structure established, we are able to study the amplitude and phase dynamics of these perturbations on networks. We demonstrate the strength of our approach using a canonical example of two-dimensional cylinder flow.

In Section 9, the interactive dynamics between modes described from a network-theoretic perspective is leveraged to developed model-based feedback control laws. Modifying the interactions between oscillators, we control the perturbations about the limit cycle state of periodic flows. Flow control strategies are then designed to suppress modal amplitudes consequently leads to drag reduction.

In Section 10, we propose a cluster-based network and control with optimization for learning coarse-grained, nonlinear control laws directly from data. The framework is based on the notion of local trajectory diversion. In contrast to manipulating the energy transfer between coherent structures to promote patterns with desirable properties, as in the model-based framework in Section 9, this Section aims to re-route a network of trajectories in an automated fashion. The control design takes advantage of data measurements alone, which benefits its applicability to both computational and experimental methods, alike. the feature space of trajectories from baseline separated flows is partitioned into discrete clusters using unsupervised clustering. To define the feature space, full flow field information is not required. As flow separation can be gauged by monitoring aerodynamic forces, a low number of intrinsic coordinates are sufficient to define the feature space. Each cluster corresponds to a characteristic coarse-grained phase of the flow in the feature space spanned by aerodynamic forces. Assigning a control law to each discretized cluster and actively monitoring and sensing the variables of interest enable the feedback control of flows (Kaiser *et al.* 2017*b*). These control laws can then be optimized to minimize a cost functional consisting of state and control variables. Previous efforts have primarily focussed on deducing optimal or suboptimal control laws in physical space co-ordinates. The cluster-based control approach provides a discrete representation of the control law in terms of feature space coordinates tieing the control design directly with data.

In Section 12, we describe an approach to leverage coherent structures in high-dimensional fluid dynamics data to accelerate computations using randomized linear algebra. In particular, we introduce a *randomized dynamic mode decomposition* to accelerate spatiotemporal coherent structures with orders of magnitude computational speed ups and tunable error bounds.

DISTRIBUTION A: Distribution approved for public release.

In Section 13, we survey machine learning based approaches to solve regression problems in unsteady fluid mechanics. These techniques were tested for flow and force reconstructions based sensor measurements and under-resolved flow field data. It is shown that machine learning based technique can reconstruct nonlinear data in an accurate and efficient manner with only modest amount of data input. These machine learning based techniques will be valuable in performing future control efforts with enhanced performance and robustness.

In Section 14, we provide concluding remarks based on the utilization of network theory and data-based approaches to analyze, model and control unsteady fluid flows. Some discussions on future directions of research are provided.

DISTRIBUTION A: Distribution approved for public release.

5. Network Analysis and Graph Theory

Network analysis and graph theory by nature are very fundamental, which enable them to impact a wide range of applications, including the analysis of technological, biological and social networks, study of traffic flows, and design of robust power grids Albert & Barabási (2002); Newman (2003); Barrat *et al.* (2004*a*); Newman (2010). Network analysis can also reveal how a group of people are socially connected to one another and examine how subgroups within a population are interlinked in a complex manner Porter *et al.* (2005). Moreover, network analysis has been utilized in electrical engineering to determine the voltages and currents associated with electrical circuits via graph representations Chen (2004). In the aforementioned applications of network analysis, the connections between people or elements are highlighted.

Biologists and medical scientists use network analysis to determine how electrical signals travel inside the brain and how abnormality in the brain network connections can affect the normal functionality Duarte-Cavajalino *et al.* (2012); Owen *et al.* (2013). In epidemiology, researchers model the outbreak of diseases on the population network. Locations with high concentration of population, such as airports, stations, schools, and hospitals, can be represented on a network with large number of connections. Identifying such locations is especially critical when containment measures are designed to control outbreaks of HIV Morris (1993), SARS Lloyd-Smith *et al.* (2005), and Influenza Glass *et al.* (2006); Cauchemeza *et al.* (2011). Each of these diseases have different dynamics and an associated network structure. Once the high-risk groups and areas are identified, network analysis can assist in designing and implementing prevention and combat strategies in the most swift manner with limited resources Salathé & Jones (2010); Robinson *et al.* (2012).

We first describe the network-theoretic framework in §5.1 and then discuss spectral sparsification technique in §5.2.

5.1. Network-theoretic framework

A network is defined by a collection of vertices joined by edges. These edges form the bonds that connect the vertices or the various entities of the system together. The edges can be associated with weights describing the importance of the connections between the vertices. Such a graph is called as a weighted graph. If the edges of the graph have an associated directivity (i.e., one vertex influences the other but not vice versa), the graph is called a directed graph. If a vertex influences itself, a self-loop edge (edge reconnected to itself) can represent such an effect. Any undirected graph $\mathcal{G}$ can be described by a set of vertices $V = \{v_1, v_2, ...v_N\}$, a set of edges E, and a set of weights w associated with the edges, i.e., $\mathcal{G} = \{V, E, w\}$ Chung (1997); Newman (2010). An example of a weighted graph is shown in figure 1 (left) with weights associated with the edges displayed. A complete graph $\mathcal{K}_N$ with $N = 5$, shown in figure 1 (right), has all the vertices connected to each other. In other words, this graph has N vertices with complete set of possible $N(N-1)/2$ edges without any self-loops. We later use a weighted version of the complete graph to describe the interactions amongst a set of discrete point vortices in the context of fluid dynamics.

Network connections can be summarized by its adjacency matrix $\boldsymbol{A}_{\mathcal{G}} \in \mathbb{R}^{N\times N}$, which is given by

$$[\boldsymbol{A}_{\mathcal{G}}]_{ij} = \begin{cases} w_{ij} & \text{if } (i,j) \in E \\ 0 & \text{otherwise.} \end{cases} \tag{5.1}$$

DISTRIBUTION A: Distribution approved for public release.

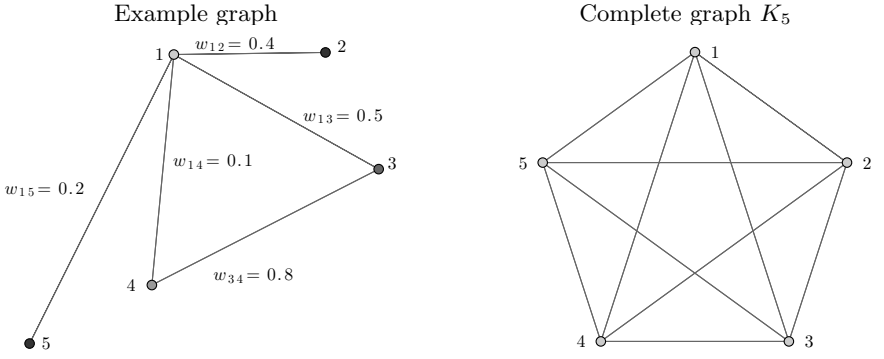


Figure 1: (left) An example weighted graph; and (right) a complete graph $\mathcal{K}_5$.

In an unweighted graph, the weight is set to unity if $(i, j) \in E$. The diagonal entries of the adjacency matrix relate to the weight of the self-connecting (loops) edges. The adjacency matrix of an undirected graph is symmetric. The degree k_i of a vertex i represents the summation of the weights of the edges connected to it given by $k_i = \sum_{j=1}^{N} [\boldsymbol{A}_{\mathcal{G}}]_{ij}$. The adjacency matrix for an example graph $\boldsymbol{A}_{\mathcal{G}}$ and the complete graph $\boldsymbol{A}_{\mathcal{K}_5}$ shown in figure 1 are given by

$$\boldsymbol{A}_{\mathcal{G}} = \begin{pmatrix} 0 & 0.4 & 0.5 & 0.1 & 0.2 \\ 0.4 & 0 & 0 & 0 & 0 \\ 0.5 & 0 & 0 & 0.8 & 0 \\ 0.1 & 0 & 0.8 & 0 & 0 \\ 0.2 & 0 & 0 & 0 & 0 \end{pmatrix} \text{ and } \boldsymbol{A}_{\mathcal{K}_5} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{pmatrix},$$

where $\mathcal{G}$ is a weighted graph and $\mathcal{K}_5$ is an unweighted complete graph.

Another important matrix in graph theory is the graph Laplacian matrix $\boldsymbol{L}_{\mathcal{G}} \in \mathbb{R}^{N \times N}$, which is given by

$$[\boldsymbol{L}_{\mathcal{G}}]_{ij} = \begin{cases} k_i & \text{if } (i,j) \in E \text{ and } (i = j) \\ -w_{ij} & \text{if } (i,j) \in E \text{ and } (i \neq j) \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

The graph Laplacian matrix can also be deduced from the adjacency matrix, $\boldsymbol{L}_{\mathcal{G}} = \boldsymbol{D}_{\mathcal{G}} - \boldsymbol{A}_{\mathcal{G}}$, where $\boldsymbol{D}_{\mathcal{G}} \in \mathbb{R}^{N \times N}$ is a diagonal matrix with elements equal to degrees of vertices, $\boldsymbol{D}_{\mathcal{G}} = \text{diag}([k_i]_{i=1}^{N})$. For undirected networks, it is symmetric and positive semidefinite (singular). This Laplacian matrix is a discrete analog of the negative continuous Laplacian operator $(-\nabla^2)$ and is also called as the discrete Laplacian. It is naturally defined by its quadratic form. If we have a linear system of the form $\boldsymbol{L}_{\mathcal{G}}\boldsymbol{x} = \boldsymbol{b}$, for the vector $\boldsymbol{x} \in \mathbb{R}^N$, the Laplacian quadratic form of a weighted graph $\mathcal{G}$ is given by Mohar (1991)

$$\boldsymbol{x}^T \boldsymbol{L}_{\mathcal{G}} \boldsymbol{x} = \sum_{(i,j) \in E} w_{ij} (x_i - x_j)^2. \tag{5.3}$$

The discrete Laplacian is a smoothness indicator of $\boldsymbol{x}$ over the edges in $\mathcal{G}$. The Laplacian quadratic form becomes large as $\boldsymbol{x}$ jumps over the edges of $\mathcal{G}$. The definitions of adjacency matrix and graph Laplacian form the building blocks of the graph-theoretic framework. The Laplacian quadratic form will be used below to introduce the notion of spectral similarity of graphs and graph sparsification.

DISTRIBUTION A: Distribution approved for public release.

5.2. Spectral sparsification

The sparse approximation of dense graphs can provide numerous benefits for numerical computation. Graph sparsification is useful for designing computationally efficient algorithms and helps in identifying representative edges and associated weights Spielman & Teng (2011). Graph similarity, which forms the basis for graph sparsification, can be derived in a number of ways depending on the desired similarity properties. Distance similarity of graphs Peleg & Ullman (1989) can be achieved by sparse graphs called spanners that keep the same shortest-path distance between each pair of vertices as in the original graph while cut similarity Benczúr & Karger (1996) can be achieved by maintaining the weight of the edges leaving a set of vertices of the sparse graph to be the same as that of the original graph. A much stronger notion of similarity referred to as *spectral similarity* was introduced by Spielman & Teng (2011). Spectral similarity is closely tied to the Laplacian quadratic form of the graphs as defined by equation (5.3). The concept of spectral similarity directly leads to spectral sparsification of graphs; that is to create sparse graphs, which are spectrally similar to the original graph.

Spectral sparsification is a more general abstraction than cut sparsifiers and maintains spectral similarity between the sparsified and original graphs. In particular, spectral sparsification can remove some of the edges in the graph, while maintaining similar adjacency and Laplacian eigenspectra. One of the key features of spectral sparsification is that it keeps the sum of the weights leaving the vertex of a graph constant. A spectral sparsifier is a subgraph of the original graph whose Laplacian quadratic form is approximately the same as the original graph Spielman & Teng (2011).

Sparsification involves the creation of a sparse graph $\mathcal{G}_S$ from the original graph $\mathcal{G}$ based on an approximation order of ϵ. The quadratic form induced by graph Laplacian of $\mathcal{G}$ is maintained upto a multiplicative $(1 \pm \epsilon)$ factor by spectral sparsification Kelner & Levin (2011). Thus, the sparse graph $\mathcal{G}_S$ is a $(1 \pm \epsilon)$-spectral approximation of $\mathcal{G}$. The approximation order ϵ can vary from zero to unity. The approximation with $\epsilon = 0$ indicates that $\mathcal{G}_S$ is same as original graph $\mathcal{G}$ and none of the edges are sparsified, while $\epsilon = 1$ relaxes the quadratic form induced by the sparse graph to within twice of that induced by the original graph. An approximation of $\epsilon = 1$ leads to a heavily sparsified graph. Denoting the Laplacian matrices of $\mathcal{G}$ and $\mathcal{G}_S$ by $\boldsymbol{L}_{\mathcal{G}}$ and $\boldsymbol{L}_{\mathcal{G}_S}$, respectively, the spectrally sparsified Laplacian satisfies

$$(1-\epsilon)\boldsymbol{x}^T\boldsymbol{L}_{\mathcal{G}}\boldsymbol{x} \leqslant \boldsymbol{x}^T\boldsymbol{L}_{\mathcal{G}_S}\boldsymbol{x} \leqslant (1+\epsilon)\boldsymbol{x}^T\boldsymbol{L}_{\mathcal{G}}\boldsymbol{x} \tag{5.4}$$

at least with probability $1/2$ with large N for all $\boldsymbol{x} \in \mathbb{R}^N$ Spielman & Srivastava (2011). For the example problem considered later, these bounds are much tighter. This tells us that $\boldsymbol{L}_{\mathcal{G}_S}$ holds eigenvalues similar to those of $\boldsymbol{L}_{\mathcal{G}}$. These spectrally similar sparse graphs are found using the spectral sparsification algorithm based on sampling by effective resistance discussed below.

Before we discuss how a graph can be sparsified, let us first follow the works of Bollobás (1998) and Srivastava (2010) to establish an analogy of a graph to an electrical circuit. If the entire graph is viewed as a resistive circuit, we can define a resistance on the individual edges $e = (i, j)$ of the graph. According to Thomson's principle, the potentials and currents in a resistive circuit distribute themselves so as to minimize the total energy in the network Bollobás (1998). This energy minimization principle leads to the concept of effective resistance.

Effective resistance between vertices i and j is the potential difference induced between them when a unit current is injected at one vertex and extracted at the other Bollobás (1998); Srivastava (2010). The effective graph resistance (also called as resistance dis-

DISTRIBUTION A: Distribution approved for public release.

tance) is the sum of the effective resistance over all the pairs of vertices in the graph $\mathcal{G}$ Klein & Randić (1993); Ellens *et al.* (2011). Rayleigh's monotonicity law states that pairwise effective resistance is a non-increasing function of the edge weights Mieghem (2011).

In order to obtain an expression of effective resistance for graph sparsification, we orient the edges of the original weighted undirected graph $\mathcal{G}$ with N vertices and M edges. We can represent any directed graph by a signed edge (e)-vertex (v) incidence matrix $\boldsymbol{B}_{\mathcal{G}} \in \mathbb{R}^{M\times N}$ given by

$$[\boldsymbol{B}_{\mathcal{G}}]_{ev} = \begin{cases} 1 & \text{if } e \in E \text{ and } v \text{ is the head of } e \\ -1 & \text{if } e \in E \text{ and } v \text{ is the tail of } e \\ 0 & \text{otherwise.} \end{cases} \tag{5.5}$$

The row of $\boldsymbol{B}_{\mathcal{G}}$ corresponding to an edge $e = (i, j)$ is given by $(\boldsymbol{p}_i - \boldsymbol{q}_j)$, where $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$ are elementary unit vectors in the i and j directions, respectively. If the edge weights of the graph are represented in a diagonal matrix given by $\boldsymbol{C}_{\mathcal{G}} \in \mathbb{R}^{M\times M}$, we can express the Laplacian matrix based on the incidence matrix as

$$\boldsymbol{L}_{\mathcal{G}} = \boldsymbol{B}_{\mathcal{G}}^T \boldsymbol{C}_{\mathcal{G}} \boldsymbol{B}_{\mathcal{G}} = \sum_{i,j\in E} w_{ij}(\boldsymbol{p}_i - \boldsymbol{q}_j)(\boldsymbol{p}_i - \boldsymbol{q}_j)^T. \tag{5.6}$$

The above relation holds true only for undirected graphs. The incidence matrix $\boldsymbol{B}_{\mathcal{G}}$ and the diagonal matrix $\boldsymbol{C}_{\mathcal{G}}$ for the example graph shown in figure 1 (left) are given by

$$\boldsymbol{B}_{\mathcal{G}} = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 1 & 0 & 0 & 0 & -1 \end{pmatrix} \text{ and } \boldsymbol{C}_{\mathcal{G}} = \begin{pmatrix} 0.4 & 0 & 0 & 0 & 0 \\ 0 & 0.5 & 0 & 0 & 0 \\ 0 & 0 & 0.1 & 0 & 0 \\ 0 & 0 & 0 & 0.8 & 0 \\ 0 & 0 & 0 & 0 & 0.2 \end{pmatrix}.$$

For undirected graphs, each edge is counted only once in the incidence matrix and the convention for head and tail of an edge can be fixed arbitrarily. Here, for edge $e = (1, 2)$ of the example graph shown in figure 1 (left), the head and tail are considered to be vertex 1 and 2, respectively. The corresponding elementary unit vectors for the edge are given by $p_1 = (1, 0, 0, 0, 0)$ and $q_2 = (0, 1, 0, 0, 0)$, which is apparent from the first row of $\boldsymbol{B}_{\mathcal{G}}$ being $\boldsymbol{p}_1 - \boldsymbol{q}_2$.

For an edge corresponding to $e = (i, j)$, the unit current is injected at vertex i and extracted at vertex j. Thus, we set the electrical current across the edge to be $(\boldsymbol{p}_i - \boldsymbol{q}_j)$. The potential induced by this current at the vertices is given by $\boldsymbol{L}_{\mathcal{G}}^+(\boldsymbol{p}_i - \boldsymbol{q}_j)$, where $\boldsymbol{L}_{\mathcal{G}}^+$ is the Moore–Penrose pseudoinverse of the Laplacian matrix $\boldsymbol{L}_{\mathcal{G}}$ Srivastava (2010). The potential difference across edge $e = (i, j)$ is then given by $(\boldsymbol{p}_i - \boldsymbol{q}_j)^T \boldsymbol{L}_{\mathcal{G}}^+(\boldsymbol{p}_i - \boldsymbol{q}_j)$. Thus, for unit current, the effective resistance across edge $e = (i, j)$ which corresponds to the potential difference can be expressed as

$$[R_e]_{ij} = (\boldsymbol{p}_i - \boldsymbol{q}_j)^T L_{\mathcal{G}}^+(\boldsymbol{p}_i - \boldsymbol{q}_j). \tag{5.7}$$

The above expression is used for computing effective resistance of the edges of the graph. The sparsification of the original graph $\mathcal{G} = \{V, E, w\}$ is performed with an algorithm `Sparsify` Spielman & Srivastava (2011) to produce a sparse graph $\mathcal{G}_S = \{V, \widetilde{E}, \widetilde{w}\}$ where $\widetilde{w}$ are the weights corresponding to the sparse graph $\mathcal{G}_S$. This algorithm is based on the concept of effective resistance and yields a $(1 + \epsilon)$ sparse graph $\mathcal{G}_S$. This sparse graph

DISTRIBUTION A: Distribution approved for public release.

contains a reduced number of $\mathcal{O}(N\log(N)/\epsilon^2)$ edges. The procedure for sparsification $\mathcal{G}_S = \texttt{Sparsify}(\mathcal{G})$ is summarized below.

We first create a list of the edges E with the associated weights of the original graph $\mathcal{G}$. The adjacency and Laplacian matrices of graph $\mathcal{G}$ are constructed from equations (8.10) and (5.2). The Moore–Penrose pseudoinverse of the Laplacian matrix $\boldsymbol{L}_{\mathcal{G}}^{+}$ is then computed. For each edge in the edge list, the elementary unit vectors, $\boldsymbol{p}_i$ and $\boldsymbol{q}_j$, and edge weights w_e are obtained. Next, the effective resistance $[R_e]_{ij}$ corresponding to each edge is computed from equation (11.11).

A random edge $e = (i, j)$ from the edge list of graph $\mathcal{G}$ is chosen with probability p_e proportional to $w_e[R_e]_{ij}$. The edge e is added to the sparse graph $\mathcal{G}_S$ with the weight given by $\widetilde{w}_e = w_e/qp_e$, where $q = 8N\log_2(N)/\epsilon^2$. We take integer$(q)$ number of samples independently without replacement and sum the weights if an edge is chosen more than once. The resulting graph becomes the sparsified graph $\mathcal{G}_S$ that satisfies equation (5.4) with eigenvalues similar to those of $\mathcal{G}$.

After the random sampling procedure, a sparsified adjacency matrix $\boldsymbol{A}_{\mathcal{G}_S}$ is obtained. For convenience, we define the ratio of the sparsified and original adjacency matrix weights as W_{ij},

$$W_{ij} \equiv \begin{cases} \widetilde{w}_{ij}/w_{ij} & \text{if } (i,j) \in \widetilde{E} \\ 0 & \text{otherwise.} \end{cases} \tag{5.8}$$

This sparsification factor W_{ij} is related to the probability of an edge $e = (i, j)$ of graph $\mathcal{G}$ being sampled. The edges of the original graph that are not sampled during the random sampling procedure have zero weights in the sparsified graph. These edges are cut during sparsification. The spectral sparsification procedure also redistributes the weights of the cut edges among the other edges of the sparsified graph. Thus, cutting of graph edges is compensated by redistribution of the weights to preserve spectral properties of the original graph.

The spectral sparsification algorithm described above produces a $(1 \pm \epsilon)$ expander graph, i.e., sparsifier with strong connectivity properties when $\epsilon \geqslant 1/\sqrt{N}$, where N is the number of vertices Spielman & Srivastava (2011). With the concept of graphs and spectral sparsification now discussed, we consider the application of network analysis to discrete vortex dynamics in the next chapter.

DISTRIBUTION A: Distribution approved for public release.

6. Vortical Interaction Networks

Vortex dynamics is at the heart of understanding fluid flow behavior and the ensuing chaotic motions Saffman (1992). Lagrangian based methods, such the vortex methods, allow us to simulate unsteady fluid flows Leonard (1980); Cottet & Koumoutsakos (2000). These methods involve modeling of fluid flow with point vortices, vortex sheets, vortex filaments or vortex patches Saffman (1992); Cottet & Koumoutsakos (2000). Low-order representation of these vortex models Wang & Eldredge (2013); Hemati *et al.* (2014) have been proposed in recent years. The evaluation of the velocity field for vortex methods rely often on fast summation methods Greengard & Rokhlin (1987) for reduced computational time. In this chapter, we apply network-theoretic framework discussed in the previous chapter to study the dynamics of a collection of point vortices Saffman (1992); Newton (2001).

In §6.1, we discuss the problem setup and in §6.2, we describe the interactions amongst a set of point vortices with graph theory and network analysis. The dynamics of the vortices will be modeled on a sparsified graph (discussed in §6.3) in place of the full (complete) graph to derive a sparsified model (discussed in §6.4) that accurately captures the motion of the vortices while conserving physical properties, such as circulation, Hamiltonian, as well as linear and angular impulse (discussed in §6.5). Comments on comparison of sparisified dynamics model with reduced-order models is offered in §6.6.

6.1. Problem setup

The flow is assumed to be two-dimensional, incompressible, and inviscid. The spatial domain $\mathcal{D}$ is taken to be infinitely large and the position of the vortices in the flow field is denoted by $\boldsymbol{r} = (x, y) \in \mathcal{D}$. We consider a collection of N vortices to be in the domain $\mathcal{D}$ resulting in the vorticity field of

$$\omega(\boldsymbol{r}) = \sum_{j=1}^{N} \frac{\kappa_j}{2\pi} \delta(\boldsymbol{r} - \boldsymbol{r}_j), \tag{6.1}$$

where κ_j and $\boldsymbol{r}_j$ are the strength (circulation) and position of the j-th vortex, respectively, and $\delta(\cdot)$ is the Dirac delta function. The motion of the vortices is described by the Biot–Savart law

$$\frac{d\boldsymbol{r_i}}{dt} = \sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{\kappa_j}{2\pi} \frac{\hat{\boldsymbol{k}} \times (\boldsymbol{r}_i - \boldsymbol{r}_j)}{|\boldsymbol{r}_i - \boldsymbol{r}_j|^2}, \tag{6.2}$$

where $\hat{\boldsymbol{k}}$ is the out-of-plane unit normal vector. To determine the trajectories of the vortices, we numerically integrate the above equation. This equation is the basis of the discrete vortex methods that are used to simulate unsteady vortical flows in place of discretizing the two-dimensional Euler equations.

We consider a configuration of N point vortices in n_c clusters. This setup allows us to examine the effect of sparsification on a cluster of vortices as discussed in the previous chapter. We consider a collection of $N = 100$ point vortices with $n_c = 5$ with each cluster containing 20 vortices. The point vortices in a particular cluster are given a normal distribution in space with a radial standard deviation of $\sigma_r = 0.2$. The strength of the point vortices, κ_i, within the individual clusters have a normal distribution with the mean of $\bar{\kappa} = 0.1$ and a standard deviation of $\sigma_\kappa = 0.01$. The setup for this configuration is shown in figure 2. The positions of the clusters are normalized by the average radial distance of the centroid of the clusters from the geometric center of the system (R_o).

DISTRIBUTION A: Distribution approved for public release.

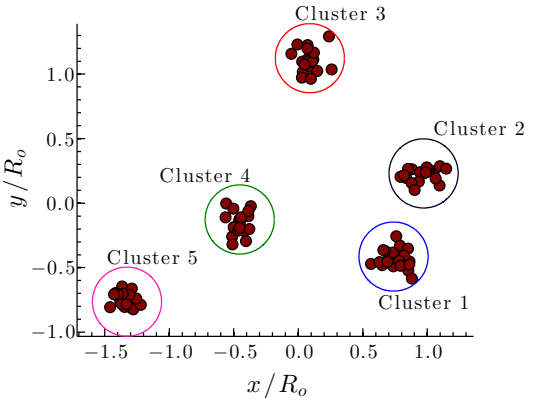


Figure 2: The spatial arrangement of $N = 100$ point vortices with five clusters.

6.2. Representing vortex interactions on a graph

A vortex is advected by the induced velocity from the other vortices and not by its own velocity as seen in equation (6.2). Each vortex is influenced by all other vortices, which means that each and every vortex has a connection to all other vortices without any self-loops. Thus, the discrete system of N point vortices can be represented by a weighted complete graph, $\mathcal{K}_N$. The vertices or nodes of the graph represent the discrete point vortices and the edge weights represent the strength of the connections between them.

The motion of point vortices is influenced by the strengths (circulation) of the individual point vortices and the distances between them. The shorter the distance between the point vortices, the higher is the influence of the vortices on each other. Also, a vortex with higher strength has more ability to influence the surrounding vortices compared to that with lower strength. We assign the weight for the interaction between two vortices to be dependent on the characteristic vortex induced velocity based on the strengths of the two vortices and the distance between them. Thus, this implies that the edge weight for the vortex network should be proportional to κ/r, where κ is the characteristic strength of the two vortices and r is the distance between them within the network.

The distance between the vortices is dependent on the location of the two point vortices $|\boldsymbol{r}_i - \boldsymbol{r}_j|$ and the characteristic strength can be established by considering their geometric mean $\sqrt{|\kappa_i \kappa_j|}$. This choice makes the adjacency matrix symmetric which allows the use of the spectral sparsification algorithm discussed in §5.2. Therefore, the adjacency matrix for the graph representation of a set of point vortices can be defined as,

$$[\boldsymbol{A}_{\mathcal{G}}]_{ij} = \begin{cases} \frac{\sqrt{|\kappa_i \kappa_j|}}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} & \text{if } (i,j) \in E \text{ and } i \neq j \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

Note that the diagonal elements of the adjacency matrix are zero as the self-induced velocity components are zero. This translates to the graph not having any self-loops. The interaction between the system of discrete point vortices represented on a complete graph can be sparsified for deriving the sparse vortex interaction model.

In addition to considering the geometric mean of the circulation for the weights, algebraic mean of the circulation was also considered. The computation using the algebraic mean (i.e., $\frac{1}{2}(|\kappa_i| + |\kappa_j|)/|\boldsymbol{r}_i - \boldsymbol{r}_j|$) for the weights perform similar to that with the above geometric mean. While we show that geometric mean sufficiently captures the

vortex interaction, there may be some room for optimizing the choice of weights. In what follows, we utilize the geometric mean in equation (6.3) for the network weight.

6.3. Sparsification of vortex interactions

In this section, we construct sparsified representation of vortex interactions by sparsifying and redistributing the weights of the connections between the point vortices as discussed in §5.2. The edge weights represented in the adjacency matrix are given by equation (6.3). It must be noted that the clusters in the problem setup considered can be clearly identified by the use of graph clustering algorithms. An algorithm maximizing modularity Newman (2004) can be utilized for the identification of clusters. One could also use k-means for cluster identification similar to the work performed by Kaiser *et al.* (2014*b*).

The original point vortex distribution modeled as a complete graph is shown in figure 3 (top left). The colors of the point vortices (network vertices) indicate their unweighted degree (number of edge connections). As the original graph here is a complete graph $\mathcal{K}_N$, each vortex is connected to every other vortex. Thus, the unweighted degree of each vortex in this example is $N-1=99$. The sparsity pattern of the adjacency matrix for the complete graph $\boldsymbol{A}_{\mathcal{K}_N}$ is shown in figure 3 (top right). We also note in the figure the sparsity index S, defined as $S \equiv n_{\text{non-zero}}/N^2$ with $n_{\text{non-zero}}$ being the number of non-zero weights in the adjacency matrix and N being the number of vortices. As the diagonal elements of the adjacency matrix have zero weights, $n_{\text{non-zero}} = N(N-1)$ for the complete graph. For $N=100$, a sparsity index of $S=0.99$ is obtained for the original graph.

We now perform spectral sparsification on the original graph ($\mathcal{G}=\mathcal{K}_N$) using the algorithm $\mathcal{G}_S = \texttt{Sparsify}(\mathcal{G})$ described in §5.2 to obtain the spectrally similar sparse representations of the complete graph based on approximation order ϵ. The corresponding adjacency matrix, $\boldsymbol{A}_{\mathcal{G}_S}$ can also be found. The vortex network for sparse graph and the sparsity pattern of $\boldsymbol{A}_{\mathcal{G}_S}$ corresponding to approximation order of $\epsilon=0.5$ are shown in figure 3 (middle). We observe that the number of edges (connections) between the vortices in cluster 3 and 5 are reduced. Because the weights of the adjacency matrix are inversely proportional to the distance between the vortices, a large number of edges between the vortices of the clusters with larger distances are cut during sparsification. The sparsity index decreases from $S=0.99$ for the original complete graph to $S=0.741$ for the sparse graph ($\epsilon=0.5$), reducing the number of connections by approximately 25 percent.

The vortex network and sparsity patterns for approximation order of $\epsilon=1$ are shown in figure 3 (bottom). We observe dense representation of the connections between the clusters 1 and 2. Thus, the proximity in clusters is identified both in the network structure and the sparsity patterns. Also, we realize from the sparsity patterns that the connections between the vortices in a particular cluster are maintained while majority of the inter-cluster ties are cut resulting in a sparse graph. The number of connections in the sparse graph for $\epsilon=1$ are reduced dramatically by nearly 60 percent with $S=0.4006$.

For weighted graphs, the effective resistance is computed in such as way so as to maintain spectral similarity. The eigenspectra (σ and λ) of the adjacency and Laplacian matrices for the sparse and original graphs are compared in figure 4. We observe from figure 4 that the spectra of the sparse and original configurations are in good agreement as expected. The second smallest eigenvalue of the Laplacian matrix represents the spectral gap or algebraic connectivity of the graph, i.e., vertices connected by at least a single path Newman (2010). As observed from figure 4, the spectral gap is greater than zero for original and sparse graphs indicating that the graph is connected. Thus,

DISTRIBUTION A: Distribution approved for public release.

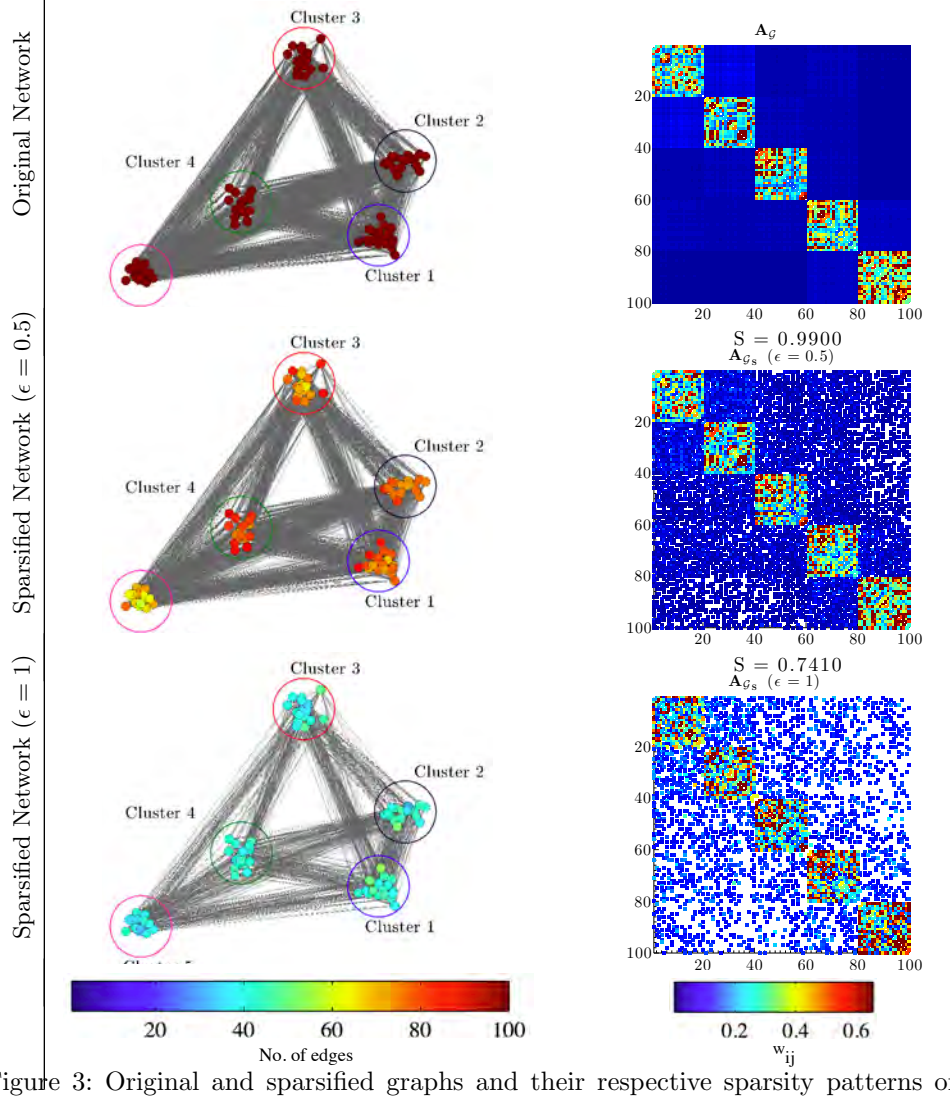


Figure 3: Original and sparsified graphs and their respective sparsity patterns of the adjacency matrix for $N = 100$ vortices. The color of the nodes of the vortex network represents the unweighted connections (number of connected edges) of the individual point vortices. The color in the sparsity patterns of the adjacency matrix indicates the adjacency weights w_{ij} for original graph $\boldsymbol{A}_{\mathcal{G}}$ and $\widetilde{w}_{ij}$ for sparse graphs $\boldsymbol{A}_{\mathcal{G}_S}$. The empty white spaces indicate sparsity in the adjacency matrix. Sparsity index is $S \equiv n_{\text{non-zero}}/N^2$, where $n_{\text{non-zero}}$ is the number of non-zero weights (elements) in adjacency matrix and N is the number of vortices. The colored circles in the vortex network represent the individual cluster groups.

sparsification preserves the connectivity properties of the original graph. The sparsified graphs slightly underpredict the maximum eigenvalues of the adjacency and Laplacian matrices compared to the original graph. The preservation of spectral properties is attractive if the sparsified network is to be used to describe the dynamics of the

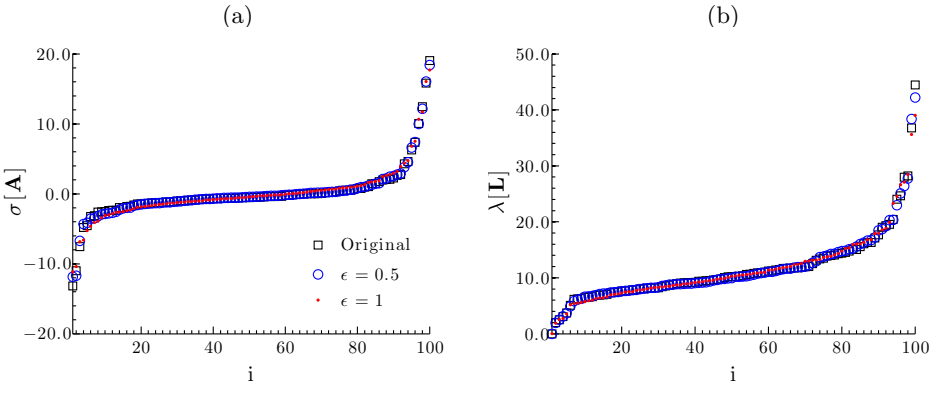


Figure 4: Eigenvalue spectra of (a) adjacency and (b) Laplacian matrices of the sparsified graphs with $\epsilon = 0.5$ and 1 in comparison to the original graph.

vortices. We expect the overall motion of the vortices to remain similar with sparsified representations of the vortex interactions.

Sparsification helps identify the individual clusters by creating subgraphs within a graph where the density of the edges between vertices is much greater than that outside it. The spectral sparsification procedure leads to the reduction in the number of edges to $\mathcal{O}(N\log(N)/\epsilon^2)$ for large N. We examine the performance of spectral sparsification on the current example for increasing the total number of vortices. As in the example that was previously considered, we maintain a constant number of clusters $n_c = 5$ but increase the number of vortices in each cluster N/n_c. The sparsity patterns of the adjacency matrix for $N = 500$ and 2000 with approximation order $\epsilon = 1$ are shown in figure 5. We observe that as the number of vortices increases, the adjacency matrix of the sparsified graph becomes increasingly sparse. Similar to the case with $N = 100$ vortices, the majority of the ties within a cluster are maintained while a large number of inter-cluster ties are cut. The sparsity index decreases from $S = 0.1784$ to 0.0707 for $N = 500$ to 2000, which is a substantial amount of sparsification. Let us further show the sparsity index S for increasing N with different approximation order ϵ in figure 6. The sparsity index S decreases with an increase in ϵ and the number of vortices. The expected behavior of the sparsity index $S = \mathcal{O}(N\log(N))/(N^2) = \mathcal{O}(\log(N)/N)$ is observed for larger N. The trend deviates from the expected behavior for lower N as the availability of the connections for redistribution of the weights of the sparsified connections decreases.

We observe that the sparsification algorithm provides us with a heavily sparsified model to produce a computationally tractable representation of the vortex-to-vortex interaction. We further note that the sparsification algorithm can be easily parallelized, which is attractive to further reduce the computational wait time. The bulk motion of clusters of point vortices could be tracked effectively by tracking their individual centroids. The present approach may lead to algorithms similar to fast particle summation methods where near and far-field effects are taken into consideration Greengard & Rokhlin (1987).

6.4. Sparsified-dynamics model

The sparsified vortex interactions can be used for analyzing the dynamics of the system of discrete point vortices. Spectral sparsification can be viewed as adjusting the strength of the vortices in the Biot-Savart law through the sparsification factor W given by equation (11.18). We can incorporate the weights of the sparse graph into the Biot-Savart

DISTRIBUTION A: Distribution approved for public release.

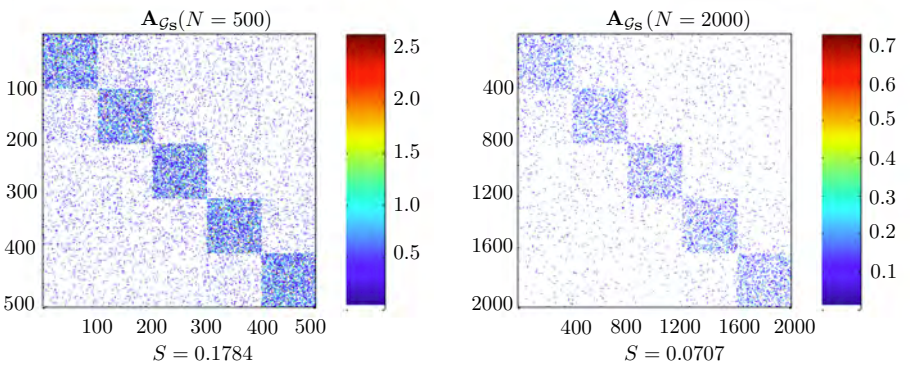


Figure 5: Sparsity patterns of adjacency matrix for $N = 500$ and 2000 for approximation order $\epsilon = 1$. The color in the sparsity patterns of the adjacency matrix denotes the adjacency weights $\widetilde{w}_{ij}$ with the empty white space indicating sparsity.

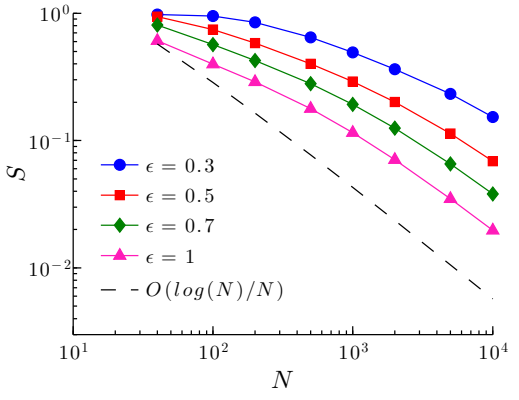


Figure 6: Variation of sparsity index S for sparsification with increasing number of point vortices. The reference black line represents $\log(N)/N$.

law as

$$\frac{d\boldsymbol{r_i}}{dt} = \sum_{\substack{j=1 \\ W_{ij}\neq 0}}^{N} W_{ij} \frac{\kappa_j}{2\pi} \frac{\widehat{\boldsymbol{k}} \times (\boldsymbol{r}_i - \boldsymbol{r}_j)}{|\boldsymbol{r}_i - \boldsymbol{r}_j|^2}, \tag{6.4}$$

where W_{ij} is the sparsification factor. The above expression significantly reduces the amount of computation compared to the original Biot–Savart law, because the sparsification has made a significant number of elements of W to be zero, as seen in figures 3 and 5.

For evaluating the effect of sparsification on dynamics of the point vortices, we consider the same example presented in §6.2 with $N = 100$ vortices as the initial condition. The objective here is to accurately capture the dynamics of the vortex clusters with the sparsified Biot–Savart law given by equation (6.4). The bulk motions of the clusters of point vortices are well represented by the centroid of the individual clusters. The original and the sparsified dynamics are given by equations (6.2) and (6.4), respectively, and

DISTRIBUTION A: Distribution approved for public release.

are integrated in time with the fourth-order Runge–Kutta method. The results from the sparsified-dynamics model are shown in figures 7 and 8.

For the present analysis, time is non-dimensionalized by considering the cumulative circulation of the vortices of the cluster (Γ) and average radial distance of the centroid of the clusters from the geometric center of the overall system at the initial time (R_o). As the advective velocity of an isolated cluster of point vortices of strength Γ at R_o is given by $u^\star = \Gamma/2\pi R_o$, the non-dimensional time can be deduced as $tu^\star/R_o = t\Gamma/2\pi R_o^2$. In the current example, each individual cluster has a collection of vortices with their strength having a mean of $\bar{\kappa} = 0.1$ (with a normal distribution). Thus, for n vortices in a cluster, $\Gamma = n\bar{\kappa}$. The error in position of the centroid of the clusters for the sparsified setting with respect to the original setting given by $|\boldsymbol{r}_\epsilon - \boldsymbol{r}|$ is non-dimensionalized by the average radial distance of the centroid of the clusters from the geometric center of the system at the initial time (R_o).

Let us consider the case where sparsification is performed only once to determine the sparsification factor W_{ij} before initiating the time integration. The same sparsification factor is used throughout the time integration. This procedure is inexpensive as sparsification is performed only once *a priori*. The trace of the positions of the centroid of the individual clusters and their error for approximation orders of $\epsilon = 0.5$ and 1 for a single sparsification are shown in figures 7 (top) and 8 (top), respectively. Considering the number of connections cut between the vortices for sparse approximations, the trajectories of the centroid of the clusters based on sparse vortex dynamics agrees reasonably with those from full dynamics. On comparing the errors, we find that the errors with approximation order of $\epsilon = 0.5$ is less than those with $\epsilon = 1$, which is expected. Despite the Biot–Savart law being nonlinear, the present sparsified approach achieves a reasonable agreement with the full nonlinear solution.

If the clusters of vortices are far from each other, the connections cut between their vortices do not cause a large change in the dynamics while the connections cut between clusters close to each other affect the dynamics considerably. As time progresses, the relative distance between vortices change, resulting in an increase in error. For $t\Gamma/2\pi R_o^2 > 1$, we observe that the errors in sparse vortex dynamics increases to $\approx 0.5R_o$. As the adjacency weights w_{ij} and the sparsification factor W_{ij} are based on the initial position of the vortices, the ratio does not hold over large times. This calls for adjustment of the weights and periodic sparsification to adapt to the dynamically changing position of the vortices.

We thus consider resparsification of the graph representing the system of point vortices periodically at $(0.1t\Gamma/2\pi R_o^2)$ and $(0.01t\Gamma/2\pi R_o^2)$ for both approximation orders. The trace of the centroids and error for approximation orders of $\epsilon = 0.5$ and 1 with resparsification are shown in middle and bottom subfigures, respectively, of figures 7 and 8. We observe that the error decreases significantly with resparsification. The error in position reduces from $\mathcal{O}(10^{-1})$ to $\mathcal{O}(10^{-2})$ with resparsification. We notice that the centroid trajectories based on the sparse and original calculations become increasingly similar with resparsification. Resparsification updates the sparsification factor periodically based on the position and strength of the vortices and decreases the error by an order of magnitude. Thus, the nonlinear evolution of discrete vortex dynamics is well predicted by the sparsification techniques.

One of the advantages of sparsification is the decreased computational cost due to increased sparsity of the connections between the vortices. This could potentially lead to design of faster algorithms based on sparsification strategies. Let us first evaluate the offline cost of computing effective resistance and random sampling required for spectral sparsification. The time required for computing effective resistance t_r and time required

DISTRIBUTION A: Distribution approved for public release.

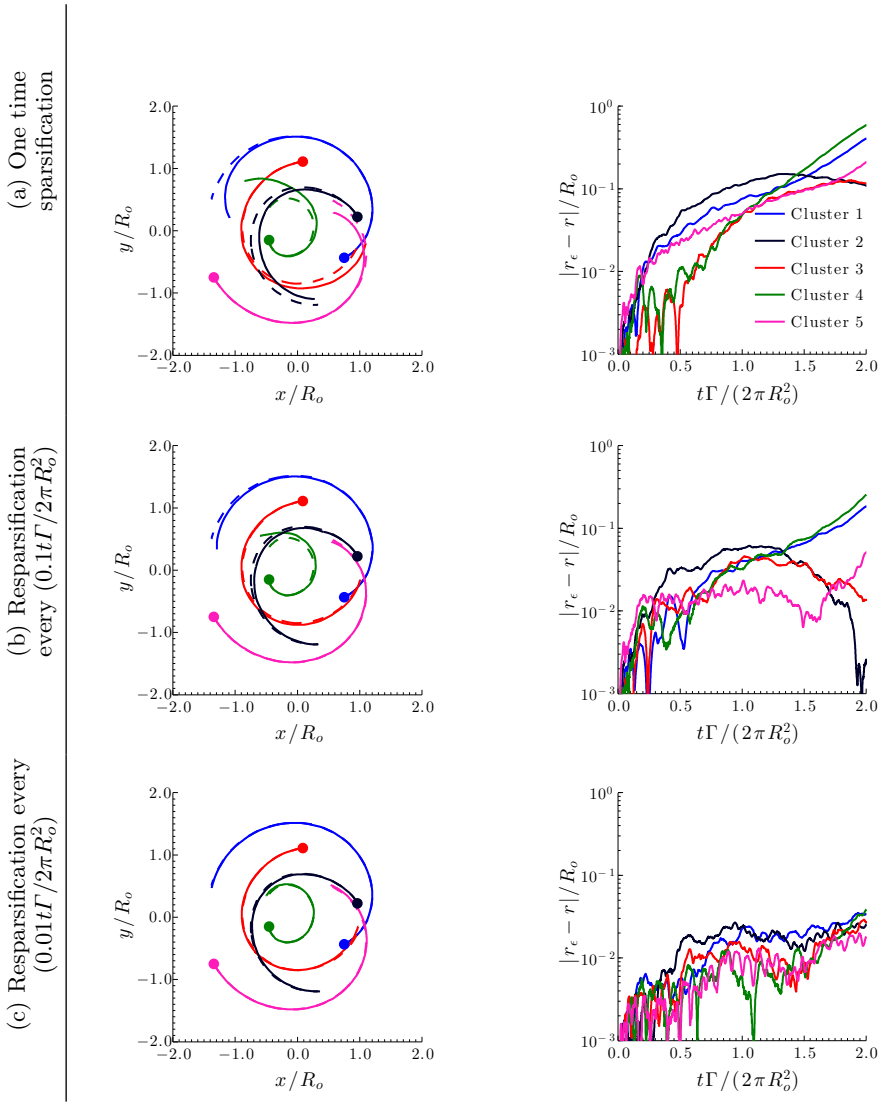


Figure 7: Trajectories of the centroids (left) and error in position of centroids (right) of the vortex clusters based on sparse graph ($\epsilon = 0.5$) and original graph with time for $N = 100, \bar{\kappa} = 0.1$. (a) trajectories and error based on sparsification performed only at the initial step while (b) results based on resparsification performed at every $(0.1t\Gamma/2\pi R_o^2)$ and (c) results based on resparsification performed at every $(0.01t\Gamma/2\pi R_o^2)$. The full nonlinear solution is shown with dashed lines.

for random sampling t_s is shown in figures 9(a) and (b) respectively. All computations were performed in MATLAB on an iMac with 3.4 GHz Intel Core i7 processor. We can observe that for each edge, the computation time of effective resistance is $\mathcal{O}(\log(N))$, hence requiring $\mathcal{O}(N^2 \log(N))$ for the overall effective resistance computation. The random sampling procedure takes $\mathcal{O}(N^2)$ time for all the approximation orders. We compare

DISTRIBUTION A: Distribution approved for public release.

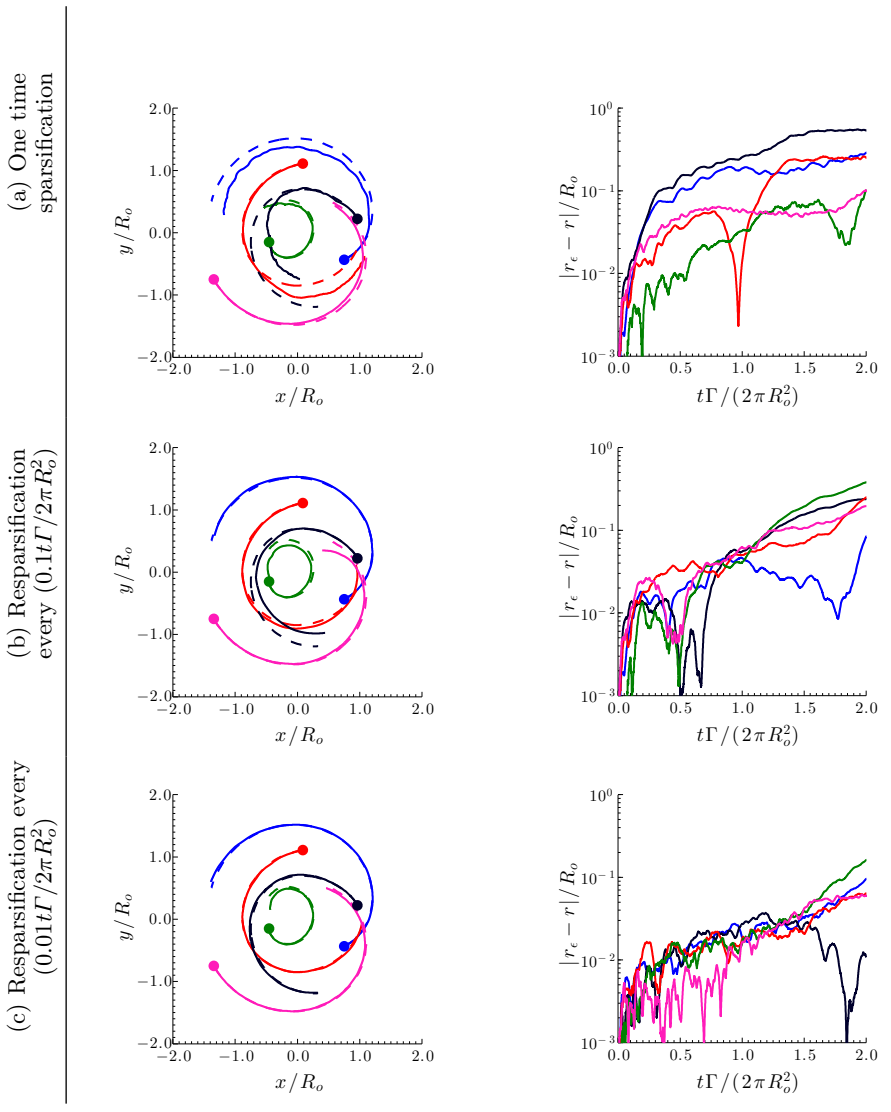


Figure 8: Trajectories of the centroids (left) and error in position of centroids (right) of the vortex clusters based on sparse graph ($\epsilon = 1$) and original graph with time for $N = 100, \bar{\kappa} = 0.1$. (a) trajectories and error based on sparsification performed only at the initial step while (b) results based on resparsification performed at every $(0.1t\Gamma/2\pi R_o^2)$ and (c) results based on resparsification performed at every $(0.01t\Gamma/2\pi R_o^2)$. The full nonlinear solution is shown with dashed lines.

the computational time t_d required at each step of numerical integration of the Biot–Savart law for the original and sparse configuration. We can observe from figure 9(c) that the original configuration takes $\mathcal{O}(N^2)$ time equivalent to the number of edges in the complete graph, while for the sparse configuration the computation time is reduced to $\mathcal{O}(N \log(N))$.

DISTRIBUTION A: Distribution approved for public release.

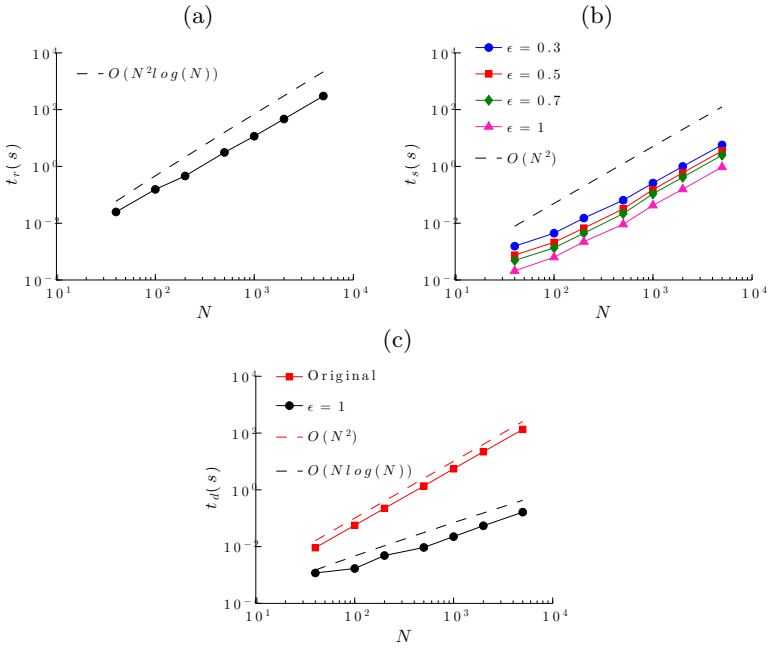


Figure 9: The time required for computation of (a) effective resistance and (b) random sampling for different approximation orders for one-time sparsification. The dashed lines indicate the expected trends. (c) Time required for numerical integration at each step for original and sparsified configuration with $\epsilon = 1$.

There is always a tradeoff amongst the level of sparsification, the error that appears in the dynamics, and the associated computational cost. The sparser the network is, the faster the computation can become but may compromise accuracy. To increase the computational accuracy, resparsification can be performed but can introduce an additional computational load. We evaluate the computational cost for numerical integration until a total time of $0.1t\Gamma/2\pi R_o^2$. The time required for resparsification and numerical integration for the two different resparsification frequencies (performed every $0.1t\Gamma/2\pi R_o^2$ and $0.01t\Gamma/2\pi R_o^2$) considered in this study for original and sparse configuration with $\epsilon = 1$ is shown in figure 10. It should be noted that for the resparsification performed every $0.1t\Gamma/2\pi R_o^2$, resparsification is performed once for the total time considered. For resparsification conducted every $0.01t\Gamma/2\pi R_o^2$, resparsfication is performed ten times. We can see that the time required for resparsification at every $0.1t\Gamma/2\pi R_o^2$ is considerably less compared to the original configuration. With increase in resparsification frequency, though the time required is less compared to the original configuration, there is increased cost associated with resparsification.

To illustrate that the trajectory prediction using sparsification cannot be performed naively, we compare the results from spectral sparsification with those based on random edge removal (without redistribution of edge weights). We keeep the number of cuts identical to that achieved by spectral sparsification for fairness. The sparsity patterns for random cuts with sparsity $S = 0.7410$ and 0.4006 and the corresponding Laplacian eigenspectra are shown in figure 11. We observe that the spectra from the randomly sparsified graphs grossly under-predicts that of the original graph.

The trace of the positions of the centroid of the individual clusters and their error for

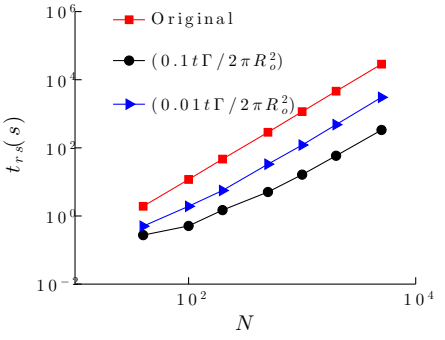


Figure 10: The time required for numerical integration for a total time of $0.1t\Gamma/2\pi R_o^2$ with different resparsification frequencies.

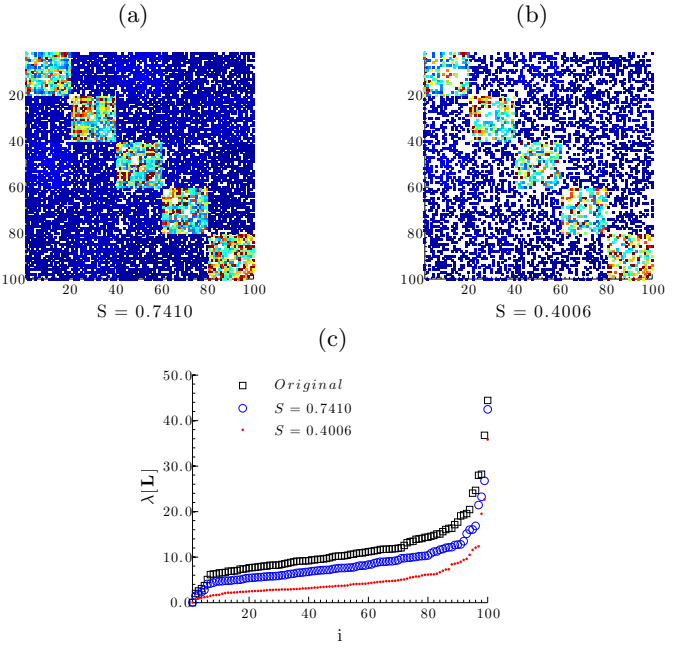


Figure 11: Sparsity patterns of the adjacency matrix for $N = 100$ vortices with random cuts with sparsity (a) $S = 0.7410$ and (b) $S = 0.4006$. The color in the sparsity patterns of the adjacency matrix indicates the adjacency weights $\widetilde{w}_{ij}$ for $S = 0.7410$ and 0.4006. (c) Eigenvalue spectra of Laplacian matrices of the randomly sparsified graphs with $S = 0.7410$ and 0.4006 in comparison to the original graph. The legend for the sparsity patterns is similar to that of sparsity patterns in figure 3.

random cuts with sparsity $S = 0.7410$ and 0.4006 for one-time sparsification are shown in figure 12. The random-cut approximation performs poorly as compared to spectral sparsification and the dynamics of the centroid of the vortex clusters are not captured. As the connections between the vortices are randomly cut, the centroids of the clusters move slower compared to the original configuration as expected. This is due to the loss of induced velocity with the absence of weight redistribution from random sparsification. Spectral sparsification, on the other hand, redistributes the weights to prevent the loss of the overall interactions amongst the set of point vortices.

DISTRIBUTION A: Distribution approved for public release.

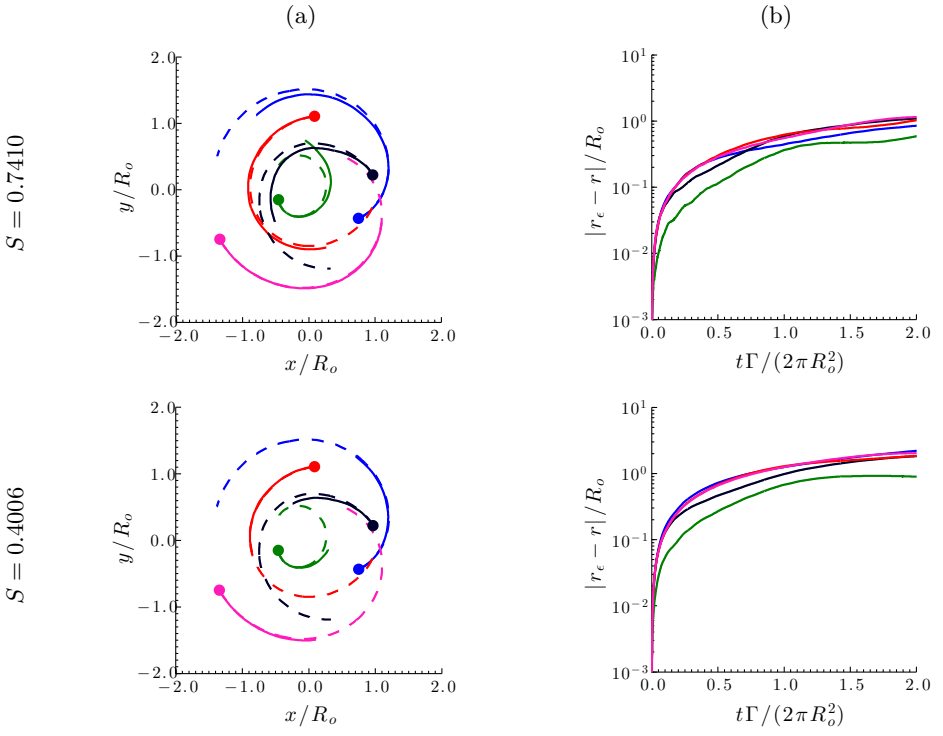


Figure 12: (a) Trajectories of the centroids and (b) error in position of centroids (right) of the vortex clusters based on random cuts with sparsity $S = 0.7410$ and 0.4006 for $N = 100, \bar{\kappa} = 0.1$.

It is well known that fast multipole methods Greengard & Rokhlin (1987) can also reduce the $\mathcal{O}(N^2)$ velocity evaluation significantly. The particle-box and box-box schemes reduce the computational complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N\log(N))$ and $\mathcal{O}(N)$, respectively. As seen in this section, spectral sparsification reduces the computational load to $\mathcal{O}(N\log(N))$ for computing the velocity of the vortices. Fast multipole methods approximate the effect of cluster of particles at a certain distance by multipole expansions and organizing the particles to a hierarchy of clusters Cottet & Koumoutsakos (2000). In order to compensate for the sparsified connections in spectral sparsification, the weights are redistributed among the other edge connections to compute the sparsified dynamics. In an analogy to fast multipole methods, the interaction list depends on the connections that have direct impact on the vortices and computational accuracy may degrade when the vortices move in space. Similar to reconstruction of tree data structures in fast multipole methods, performing resparsification periodically increases the accuracy by re-evaluating the associated weights as the positions of the vortices evolve over time. We note that even with sparsification, invariants of discrete vortices are conserved as discussed below.

6.5. **Conservation properties**

For a discrete set of point vortices, angular impulse and linear impulse are among the conserved quantities. The coordinates of center of vorticity obtained from linear impulse and the length of dispersion about the center of vorticity obtained from angular impulse, when the strength of vortices is of the same sign, remain constant. The center of vorticity

DISTRIBUTION A: Distribution approved for public release.

(X, Y) and length of dispersion of vorticity D are given by

$$X = \frac{\sum_{i=1}^{N} \kappa_i x_i}{\sum_{j=1}^{N} \kappa_j}, \quad Y = \frac{\sum_{i=1}^{N} \kappa_i y_i}{\sum_{j=1}^{N} \kappa_j}, \tag{6.5}$$

$$D^2 = \frac{\sum_{i=1}^{N} \kappa_i \{(x_i - X)^2 + (y_i - Y)^2\}}{\sum_{j=1}^{N} \kappa_j}, \tag{6.6}$$

where κ_i is the strength of the i-th vortex and (x, y) are the vortex positions Batchelor (2000); Newton (2001). The circulation for n vortices in a cluster is $\Gamma = n\bar{\kappa}$. The total circulation of the system of $n_c = 5$ clusters is given by $\Gamma_t = \sum_{i=1}^{N} \kappa_i = n_c \Gamma$. Another conserved quantity is the Hamiltonian. The Hamiltonian is defined as the interaction energy of the system of point vortices and is given by

$$H = -\frac{1}{4\pi} \sum_{i,j=1, i \neq j}^{N} \kappa_i \kappa_j \log \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}. \tag{6.7}$$

We non-dimensionalize the Hamiltonian using the total circulation of n_c clusters and the average radial distance of centroid of the clusters at initial time R_o.

We compare the error in the Hamiltonian $(H_\epsilon(r/R_o) - H(r/R_o))/\Gamma_t^2$, the center of vorticity $((X_\epsilon - X)/R_o, (Y_\epsilon - Y)/R_o)$, and the square of length of dispersion of vorticity $(D_\epsilon^2 - D^2)/R_o^2$ for approximation orders of $\epsilon = 0.5$ and 1 over time in figure 13. The preservation of center of vorticity and length of dispersion implies the conservation of linear and angular impulse. Here, the variables with subscript ϵ denote those based on a sparsified model. We observe that the error in the Hamiltonian is of $\mathcal{O}(10^{-4})$ for $\epsilon = 0.5$ and $\mathcal{O}(10^{-3})$ for $\epsilon = 1$. The error in the square of dispersion is of $\mathcal{O}(10^{-6})$ while the error in the center of vorticity is of $\mathcal{O}(10^{-8})$ with sparsification. While not shown, resparsification of these vortices periodically performed at every $(0.1t\Gamma/2\pi R_o^2)$ and $(0.01t\Gamma/2\pi R_o^2)$ maintain similar error levels for the invariants. We also note that the circulation of each vortices has not been altered, which conserves the individual and overall circulation over time.

As seen in §5.2, sparsification by effective resistance is based on the energy-minimization principle. The effective resistance is defined such that the total energy of the system remains constant. This is reflected well in the conservation of the Hamiltonian representing the interaction energy of the discrete point vortices. In addition, sparsification preserves other invariants including linear and angular impulse of the discrete set of point vortices. We note in passing that even with resparsification, the invariants were observed to be conserved. The conservation of the invariants for the sparse configurations further encourages the applicability of sparsification strategies on discrete vortex dynamics.

6.6. Comparison with reduced-order model

We comment on the similarities and differences between the present sparsified-dynamics model and the reduced-order model. Both models share the general objective of deriving a model that captures the full-order physics in a distilled manner. Reduced-order models achieve such goals by reducing the dimension of the state variable. One such technique is the Galerkin projection of the Navier–Stokes equations using a set of spatial bases Noack *et al.* (2005); Rowley *et al.* (2004), such as those determined from POD Berkooz *et al.* (1993*a*); Holmes *et al.* (1996). One can further consider incorporating the effect of input and output dynamics by using balanced truncation or the eigensystem realization algorithm Rowley (2005); Ma *et al.* (2009). The resulting

DISTRIBUTION A: Distribution approved for public release.

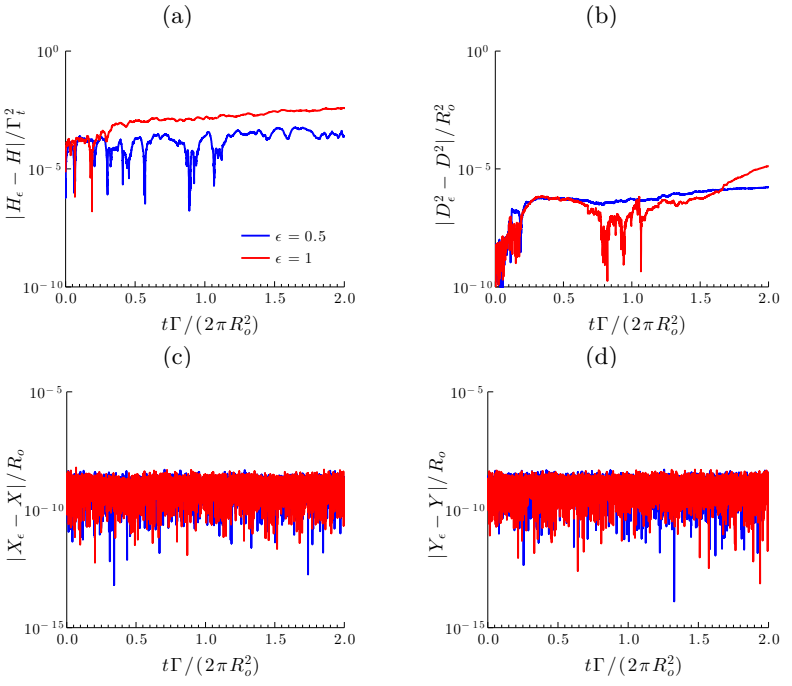


Figure 13: Error in the (a) Hamiltonian, (b) square of length of dispersion, (c) x-position of center of vorticity and (d) y-position of center of vorticity based on single sparsification for $N = 100, \bar{\kappa} = 0.1$ and $\epsilon = \{0.5, 1\}$.

reduced-order models are generally described by ordinary-differential equations for the temporal coefficients with reduced dimensions.

On the other hand, the present sparsified-dynamics model does not reduce the dimensionality of the state variable. We instead focus on reducing the number of connections used to capture the overall dynamics of the fluid flow. The reduction in the number of connections is performed based on the concepts from network analysis and graph theory. Although we cut a large number of the edges in the graph representation of the dynamical interaction, we redistribute the weights associated with the edges to maintain properties of the graph. This procedure is based upon approximating the spectral properties of the graph and does not require selecting the spatial basis functions unlike the Galerkin-projection based models. One nice feature of the sparsified-dynamics model is its ability to conserve physical variables such as the Hamiltonian, angular impulse, and linear impulse, as discussed previously. Hence, the sparsified dynamics is able to predict the full dynamics as demonstrated by the example with discrete vortices. By highlighting the interactions amongst a set of vortices, we are able to determine which interactions amongst the vortices are important in guiding the overall motion of vortices.

We believe the sparsified-dynamics model has promising potential to model various types of fluid flow by considering the modal structures as a abstraction of graph nodes. The application of sparsified dynamics models towards flow control problems may also become fruitful as the the computational time necessary to capture the complex behavior of the flow is reduced and the interaction of nodes or flow structures are well-captured, which has been a lacking feature in linear dynamics model and linear stability analysis. We anticipate that the presently proposed model can lead to potential feedback control

DISTRIBUTION A: Distribution approved for public release.

of fluid flow Bagheri *et al.* (2009); Ahuja & Rowley (2010) but with nonlinear interactions emphasized. We extend the network-theoretic framework to turbulent flows in the next chapter.

DISTRIBUTION A: Distribution approved for public release.

7. Turbulent Flow Networks

The rich and complex nature of turbulent flows stem from vortical interactions in the flow field that take place over a wide range of length scales Tennekes & Lumley (1972); Hinze (1975); Frisch (1995); Pope (2000); Davidson (2004); Lesieur (2008). The chaotic motion of a large number of vortices in turbulent flows is caused by the induced velocities of the vortices themselves. The vortical network discussed in the previous chapter neatly summarizes these interactions. In this chapter, we apply this network-theoretic approach to characterize two-dimensional incompressible decaying isotropic turbulence problem.

We discuss the problem setup in §7.1. Vortical network representation and characterization is described in §7.2 and §7.3, respectively. In §7.4, we examine the dynamics of the characterized vortical networks and discuss their resilience properties in §7.5.

7.1. Problem setup

We perform direct numerical simulation on a square bi-periodic computational domain $(x, y) \in [0, L] \times [0, L]$ with a grid size of $m_x \times m_y = 1024 \times 1024$. The unforced two-dimensional incompressible isotropic turbulent flow is simulated by numerically solving the two-dimensional vorticity transport equation given by,

$$\frac{\partial \omega}{\partial t} + u_j \frac{\partial \omega}{\partial x_j} = \frac{1}{Re} \frac{\partial^2 \omega}{\partial x_j \partial x_j}, \tag{7.1}$$

where $\boldsymbol{u}$ and $\omega = \boldsymbol{\omega} \cdot \hat{\boldsymbol{k}}_z$ are the velocity and vorticity variables, respectively and $\hat{\boldsymbol{k}}_z$ is the out-of-plane unit normal vector. A Fourier spectral method and fourth-order Runge–Kutta time integration scheme (Canuto *et al.* 1988) is used to perform the simulation. We use a smooth distribution comprised of a large number (≈ 100) of superposed vortices (Taylor 1918) with random strengths, core sizes, and locations to initialize the vorticity field for the simulation. To avoid domain effects from interfering with the analysis, the initial core sizes of the vortices are chosen to be sufficiently small compared to the size of the computational domain (McWilliams 1984). The velocity variable is normalized by the square root of the spatial average of the initial kinetic energy $u^*(t_0) \equiv [\overline{u^2}(t_0)]^{1/2}$, where the overline denotes the spatial average. The spatial length and time scales are non-dimensionalized by the initial integral length scale $l^*(t_0) \equiv [2\overline{u^2}(t_0)/\overline{\omega^2}(t_0)]^{1/2}$ and the initial eddy turnover time $t_0^* \equiv l^*(t_0)/u^*(t_0)$, respectively. The Reynolds number is defined accordingly as $Re \equiv u^* l^*/\nu$ where ν is the kinematic viscosity. In this study, turbulent flows with initial Reynolds numbers of $Re(t_0) = 75$, 439, 814, 1607, and 2485 are examined.

In what follows, we discuss regarding the extraction of turbulent network structure in §7.2. We expand on network characterization and dynamics in §7.3 and §7.4, respectively. Finally, we discuss some resilience properties of turbulence networks in §7.5.

7.2. Network representation

First, we extract the network structure of the turbulent flow. For this purpose, we quantify the interactions between fluid elements based on the vortical interactions. The velocity $\boldsymbol{u}$ at position $\boldsymbol{x}$ induced by the vorticity distribution $\boldsymbol{\omega}$ of the flow is

$$\boldsymbol{u}(\boldsymbol{x}, t) = \frac{1}{4\pi} \int \frac{\boldsymbol{\omega}(\tilde{\boldsymbol{x}}, t) \times (\boldsymbol{x} - \tilde{\boldsymbol{x}})}{|\boldsymbol{x} - \tilde{\boldsymbol{x}}|^3} d\tilde{\boldsymbol{x}}. \tag{7.2}$$

In Chapter 6, we considered a Lagrangian representation of the flow field for discrete point vortices. The circulation of the individual point vortices and their location were used to define the network metric in §6.2. In this study for unforced two-dimensional

DISTRIBUTION A: Distribution approved for public release.

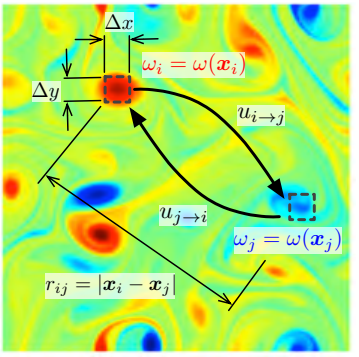


Figure 14: Interaction of fluid elements in two-dimensional turbulence. The strength of the vortical interaction between elements i and j having vorticity ω_i and ω_j is quantified through the induced velocities $u_{i\to j}$ and $u_{j\to i}$, respectively. For discretizing the Cartesian domain, we take n_x and n_y points in the horizontal and vertical directions, respectively, providing the adjacency matrix $\boldsymbol{A}$ of size $n \times n$ with $n = n_x n_y$. Shown in the background with a contour plot is the corresponding vorticity field with initial $Re(t_0) = 814$ at $t = 18$.

isotropic turbulence in a periodic box, we assess the influence of the vorticity distribution over a Cartesian domain. Here, the two-dimensional vorticity field reduces to $\boldsymbol{\omega}(\boldsymbol{x}, t) = \omega(x, y, t)\hat{\mathbf{e}}_z$ with $\hat{\mathbf{e}}_z$ denoting the unit normal plane vector. Following the Eulerian representation of the flow field, the vortical component for each discrete Cartesian element is modeled as a line vortex. We then evaluate the interactions between these vorticity elements of the flow, as shown in Fig. 34. Similar to Eq. (6.3), the magnitude of the induced velocity from fluid element i on another element j reduces from Eq. (7.2) to

$$u_{i\to j} = \frac{|\Gamma_i|}{2\pi|\boldsymbol{x}_i - \boldsymbol{x}_j|}, \tag{7.3}$$

where $\Gamma_i = \omega(\boldsymbol{x_i})\Delta x \Delta y$ is the circulation of fluid element i with side lengths of Δx and Δy. The superposition of the induced velocity from all other fluid elements provides the advective velocity of the fluid element. Although the adjacency matrix weights are defined to be positive, vortical interactions can also be generalized to accommodate negative weights.

As discussed above, the nodes $\mathcal{V}$ in this study are taken to be the vortical elements residing within the Cartesian cells and the edges $\mathcal{E}$ represent the vortical interactions between them. Similar to the last chapter, we consider a symmetric version of network adjacency matrix defined as the average induced velocity

$$A_{ij} = \begin{cases} \frac{1}{2}(u_{i\to j} + u_{j\to i}) & \text{if } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{7.4}$$

This leads to an undirected graph. We also account for the influence from the neighboring periodic vortex images in the analysis. In assessing the strength of the vortical interaction between two fluid elements, we utilize Eq. (7.4) to perform network analysis to extract the spatial connectivity structure. Geometric mean of induced velocity can be alternatively be chosen as weights and yields similar results. This approach has been successful in capturing the nonlinear vortex dynamics and modeling the trajectories of vortex clusters (Nair & Taira 2015). In general, we can formulate the adjacency matrix in an asymmetric

DISTRIBUTION A: Distribution approved for public release.

manner leading to directed graphs as

$$A_{ij} = \begin{cases} \phi u_{i\to j} + (1-\phi) u_{j\to i} & \text{if } i \neq j \\ 0 & \text{otherwise.} \end{cases} \tag{7.5}$$

where, the parameter ϕ takes a value between 0 and 1. We can recover the symmetric formulation by setting $\phi = 1/2$. When $\phi = 0$ and 1, the adjacency matrix A_{ij} are defined by the velocity imposed to the other elements ($A_{ij} = u_{j\to i}$) and upon themselves ($A_{ij} = u_{i\to j}$), respectively, for $i \neq j$. We also briefly consider this asymmetric formulation to highlight the difference from a physical point of view in the next section.

7.3. Network characterization

We identify the underlying network structure and characteristics of two-dimensional turbulence using symmetric adjacency matrix defined by Eq. 7.4. The time-evolving vorticity field is obtained from a two-dimensional incompressible bi-periodic direct numerical simulation (Canuto *et al.* 1988) for unforced isotropic turbulence. Given the vorticity field over a Cartesian grid, each fluid element is considered to be connected to all other elements through vortical network edges. The resulting fluid flow network is described by a complete graph with a range of weights. We visualize the network edges with transparent gray scale corresponding to the adjacency weight, as shown in Fig. 15(a). The captured structure reveals the turbulent network. Some regions in the flow have a large number of strong connections corresponding to larger stronger vortices seen in red, serving as primary network hubs. Note that these strong vortices induce velocities over long distances. Moderate size vortices that act as secondary hubs also possess dominant connections to primary hubs and other secondary hubs. In contrast, fluid elements corresponding to smaller, weaker eddies, shown in blue, generally have influence only in their vicinity. The node strength distribution ($s_i = \sum_j A_{ij}$) over space shows that the vortices with large circulation have larger strength, as illustrated in Fig. 15(b). The node strength distribution over space enables us to distinguish secondary and primary hubs, which may not be easily differentiated from simply visualizing the vorticity field or the Q criterion in a traditional manner. For instance, see the green vortices in (b) which can appear similar to primary ones in vorticity level.

The probability of strength distribution $P(s)$ over the strength s of fluid elements is shown in Fig. 15(c). We find that two-dimensional isotropic turbulence network has a power-law distribution $P(s) \sim s^{-\gamma}$ with $\gamma = 2.7$ at the time shown. This tells us that the vortex interactions in turbulence can be characterized by a weighted scale-free network. This realization enables the interaction-based analysis of turbulent flows from a new perspective through network theory (Newman 2010; Cohen & Havlin 2010). In particular, this type of network is known to have certain resilience properties as we will explore later in this section. Also shown in Fig. 15(c) in gray are the degree distributions for asymmetric adjacency formulations. The out and in-degree distributions can be found by setting $\phi = 0$ and 1, respectively, in Eq. (7.5). It can be observed that the scale-free symmetric distribution is mostly comprised of the out-degree components, which describe how each vortical element influences all other elements (i.e., $u_{j\to i}$). In contrast, we find that the in-degree distribution has a single peak which conveys that all fluid elements receive a similar amount of collective influence from vortices in the flow field. We have found that the scale-free property of two-dimensional isotropic turbulence is most well-captured by the symmetric weights compared to the other asymmetric formulations. It is also possible to examine the strength distribution taking positive and negative values of circulations, as we have briefly discussed in Section 2. Utilizing positive and negative

DISTRIBUTION A: Distribution approved for public release.

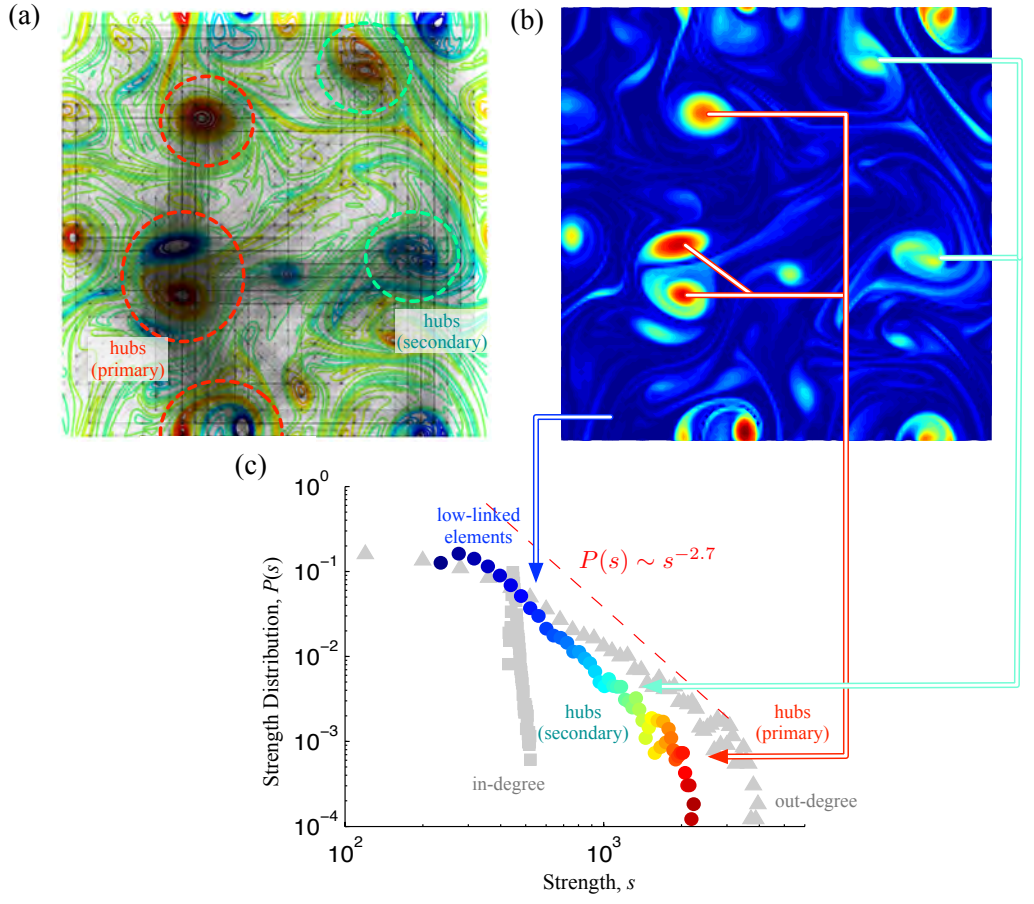


Figure 15: The scale-free network of vortical interactions in two-dimensional turbulence with initial $Re(t_0) = 814$. (a) Turbulent network structure overlaid on the vorticity field with the darkness of the network edges corresponding to the values of the adjacency weights ($t = 18$). (b) Contour plot of the node strength s distribution. Vortex cores having high degree of connectivity act as hubs in the turbulent vortical network. (c) The corresponding node strength probability distribution exhibiting the scale-free characteristics with $P \sim s^{-2.7}$. The same contour level is shared by (b) and (c). Also shown in the background of (c) in gray are the out and in-degree distributions ($\phi = 0$ and 1, respectivley). The network visualized in (a) does not show interactions from periodic images and uses 32×32 nodes for graphical clarity.

weights, their strength distribution can also exhibit a scale-free behavior but with network strength having both negative and positive values. This leads to a symmetric strength distribution over the strength with resemblance to the probability density function of scaled displacements (Weiss *et al.* 1998). In what follows, results based on the symmetric adjacency matrix (using the magnitude of induced velocity) are presented.

7.4. Network dynamics

We further examine the time-varying properties of the turbulent network. In unforced turbulence, the kinetic energy of the flow decreases over time due to viscous dissipation

as shown in Fig. 16(a). The strength distribution $P(s)$ of the turbulence network and the corresponding flow field snapshots are presented in Fig. 16(b). Turbulent flow is comprised of vortical structures over a wide range of spatial scales initially. The distribution $P(s)$ exhibits scale-free characteristics for $t \lesssim 30$ with $P(s) \sim s^{-\gamma}$, where $\gamma \approx 2.7$. For the flow under consideration, a bend in the strength distribution appears after $t \approx 30$ as the system starts to exhibit scale separation. This is caused by the diffusion of smaller scale structures and their merging with other structures. Over time, viscous dissipation removes kinetic energy through the smaller eddies and leaves only the larger vortices. This behavior can be described by two power laws $P(s) \sim s^{-\gamma_1}$ and $P(s) \sim s^{-\gamma_2}$, where they capture the weaker fluid elements and the larger stronger vortices, respectively. The bifurcation of these power laws is shown in Fig. 16(c) indicated by the vertical dashed line.

We have considered a range of Reynolds numbers and observed that γ takes values of $\gamma = 2.7 \pm 0.5$. The variations observed in γ, γ_1, and γ_2 shown in Fig. 16(c) are influenced by the chaotic nature of turbulence. These parameters however appear to exhibit a coalescing behavior when they are plotted over the product of the characteristic velocity and length, $u^*(t)l^*(t)$. Here, we interpret $u^*(t)l^*(t)$ as the circulation of vortices that have the characteristic velocity and length scales. As shown in Figure 17, we observe that the turbulence network shows coalescence of the scale-free parameter γ to $\gamma_{\rm cr} \approx 2.7$ over time for different cases of turbulent flows. Once the flows reach a state where the characteristic strength of vortices is $(u^*l^*)_{\rm cr} \approx 0.063$, the network distribution bifurcates to display two different slopes with γ_1 and γ_2, as previously illustrated in Figure 16. This observation reveals that a scale-free turbulent network is present until the unforced turbulent flow field loses the smaller-scale vortices and mostly contains vortices with strengths larger than the critical value of $(u^*l^*)_{\rm cr}$.

7.5. Resilience of turbulence networks

Characterizing turbulent flow with a scale-free network enables us to view turbulent interactions in a systematic manner and provides insights into how vortical structures influence each other. It is known from network analysis that scale-free networks are resilient to random perturbations but attacks towards network hubs can affect network dynamics in a detrimental manner (Albert *et al.* 2000). Network resilience for fluid flow translates to the difficulty of modifying the vortical interaction network and, consequently, the collective behavior of the vortices over time. To measure the change in vortical interaction caused by network disturbance, we can consider how the removal of turbulence network nodes (percolation) modifies the characteristic network length

$$l_{\rm network} \equiv \frac{1}{n(n-1)} \sum_{i \neq j} \min d(i,j), \tag{7.6}$$

which is the average shortest network distance $d(i,j)$ between any two nodes on a network. Here, we perform node percolation by setting the vorticity values at the chosen nodes to be zero. The above metric quantifies how well vortical elements are connected within a turbulent network. Note that the distance here refers to network distance based on the adjacency matrix and not the spatial distance.

In particular, we take the inverse of each adjacency weight $1/a_{ij}$ and evaluate the minimal sum

$$d(i,j) = 1/a_{ik_1} + 1/a_{k_1k_2} + \cdots + 1/a_{k_mj} \tag{7.7}$$

over a network path that connects nodes i and j for this metric (Rubinov & Sporns 2010). This metric $l_{\rm network}$ can be thought of as the average of the minimal characteristic

DISTRIBUTION A: Distribution approved for public release.

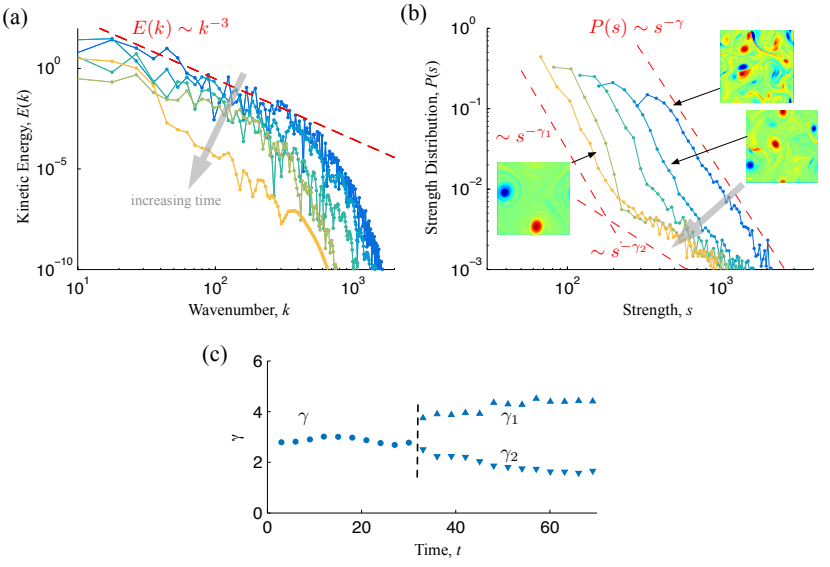


Figure 16: The dynamics of turbulent network with $Re(t_0) = 814$. (a) Kinetic energy and (b) strength distribution of two-dimensional isotropic turbulence for $t = 15$, 30, 75, 150, and 300 (line colors represent time). The inset plots in (b) show the corresponding vorticity fields. The kinetic energy $E(k)$ is shown over the wavenumber k exhibiting the asymptotic profile of $E(k) \sim k^{-3}$. The strength distribution $P(s)$ displays the scale-free property of $P(s) \sim s^{-\gamma}$ over node strength s. (c) The corresponding exponents γ, γ_1, and γ_2 are shown. Later in time the strength distribution exhibits the emergence of two distributions, $P(s) \sim s^{-\gamma_1}$ and $s^{-\gamma_2}$.

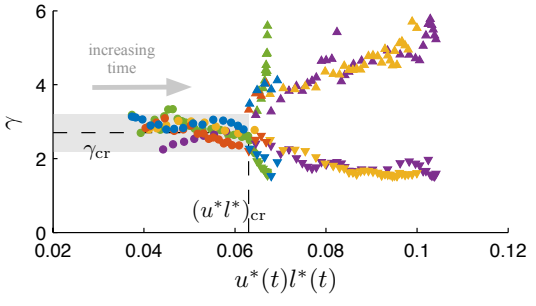


Figure 17: Exponent γ for the network strength distribution $P(s) \sim s^{-\gamma}$ plotted over $u^*(t)l^*(t)$ with different initial Reynolds numbers (green: $Re(t_0) = 75$, purple: $Re(t_0) = 439$, yellow: $Re(t_0) = 814$, red: $Re(t_0) = 1607$, and blue: $Re(t_0) = 2485$). Scale-free distributions are observed with γ coalescing to $\gamma_{cr} \approx 2.7$ up until a bifurcation at $(u^*l^*)_{cr} \approx 0.063$. The gray box shows $\gamma = -2.7 \pm 0.5$ as reference.

advective (commute) time per unit length between every pair of fluid elements in the domain. This minimal network distance is determined using the Floyd–Warshall algorithm (Floyd 1962).

The changes in the turbulence network characteristic length l_{network} when network nodes are removed in a random fashion and a coordinated manner targeting hub nodes are summarized in Fig. 18. Here, the changes in the normalized characteristic network

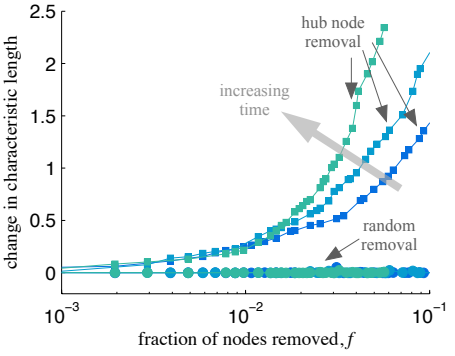


Figure 18: The resilience of turbulence network against node removals for $t = 15$, 30, and 75 with $Re(t_0) = 814$. Shown are relative changes in the characteristic network length $\Delta\tilde{l}_{\text{network}}$ of turbulent flow for random node and hub node removals. The colors of the curves represent the time when node removal is considered and follows Figure 16.

length

$$\Delta\tilde{l}_{\text{network}} \equiv \frac{l_{\text{network}}(t,f) - l_{\text{network}}(t,f=0)}{l_{\text{network}}(t,f=0)} \tag{7.8}$$

for varied fraction of node removal are shown. While it would be difficult to completely remove nodes as we have performed in this investigation, the present analysis sheds light on how external forcing or perturbations can alter the turbulent flow from an interaction-based analysis. We observe that turbulent flow is resilient against random forcing, as evident from the characteristic network length being unaffected even for a large fraction f of nodes being removed. This behavior is consistently observed over time. On the other hand, we find that the global vortical interaction network can be greatly modified by targeting large vortex cores (hubs), as exhibited by the substantial change in the characteristic length. It may be more energetically expensive to remove well-connected hub nodes, which often correspond to regions of concentrated vorticity. However, it is clear from Figure 18 that even the smallest fraction of hub node removal can influence the overall interaction, which suggests that hub removal still provides a more effective and efficient way to modify the flow than random node removal.

When the vortical interaction network is grossly altered, the dynamics of the collection of vortices would be significantly modified (Nair & Taira 2015). These observations also agree with past studies in flow control that identified effective actuation frequencies to be associated with the length scale of the large coherent structures in turbulent flows (Joslin & Miller 2009; Gad-el-Hak 2000). With increasing time, we can further notice that network connectivity decreases with hub removal due to viscous dissipation of smaller vortical structures and the influence of removing the core structures becomes more evident. The present network based understanding reveals which type of flow structures should be targeted with flow control if we aim to alter the behavior of the turbulent flow field in a global manner. In the next chapter, we describe interactions between coherent structures in unsteady fluid flows.

DISTRIBUTION A: Distribution approved for public release.

8. Modal Interaction Networks

The previous two chapters (6 & 7) dealt with describing interactions between vortical elements in physical space. In this chapter, we discuss interactions between coherent structures in wave space. Modal decomposition techniques extract these coherent structures (low-order basis) embedded in fluid flows. In particular, proper orthogonal decomposition (POD) (Sirovich 1987; Aubry *et al.* 1988; Berkooz *et al.* 1993*a*) and dynamic mode decomposition (DMD) (Schmid 2010*a*) decomposes snapshot flow field data into discrete modes based on energy content and frequency content, respectively. Global behavior of nonlinear flows can also be described by spectral analysis of linear, infinite-dimensional Koopman operator which yields Koopman modes (Rowley *et al.* 2009*a*; Mezic 2013).

In this chapter, we are interested in the unsteady fluctuations of flows with well-identified coherent structures. These are especially observed in laminar flows. Although the applicability of the approach utilized is not limited to such flows, these flows provide a tractable testbed for identifying interactions between the coherent structures (modes). We are interested in capturing the energy transfer dynamics between modes. In this regard, POD provides an optimal set of minimum number of modes capturing maximum energy content of the dynamical system. Also, POD is especially important in connection with constructing reduced-order models for capturing the dominant dynamics of the flow.

We first describe the procedure to extract these modes in unsteady fluid flows using the POD technique in §8.1. The network representation is discussed in 11.2. We apply the network-theoretic formulation to examine the modal interactions in a canonical problem of two-dimensional incompressible flow over a cylinder in §8.3.

8.1. Modal oscillators

We consider a baseline case corresponding to time-periodic flow without any forcing or perturbations introduced in the Navier–Stokes equations. For POD analysis, we use the method of snapshots approach (Sirovich 1987). Using this approach, the unsteady velocity field $\boldsymbol{u}$ can be approximated by a finite expansion in terms of mean (time-averaged) velocity field $\boldsymbol{u}_0$ and N spatial POD modes $\boldsymbol{\phi}_j^{\boldsymbol{u}}$ as

$$\boldsymbol{u}(\boldsymbol{x},t) = \boldsymbol{u}_0(\boldsymbol{x}) + \sum_{j=1}^{N} a_j(t)\boldsymbol{\phi}_j^{\boldsymbol{u}}(\boldsymbol{x}), \tag{8.1}$$

where $a_j(t) = \left\langle \boldsymbol{u} - \boldsymbol{u}_0, \boldsymbol{\phi}_j^{\boldsymbol{u}} \right\rangle$ are the temporal coefficients and $\langle \cdot, \cdot \rangle$ denotes the inner product over the computational domain. The fluctuation kinetic energy of the modes are given by $a_j^2/2$, amounting to a total modal energy of $E = \sum_{j=1}^{N} a_j^2/2$. As mentioned earlier, for time-periodic flows, we obtain modes in conjugate pairs from POD. Each conjugate mode pair describing periodic coherent structures in the baseline case define an oscillator in our analysis. From a set of N modes obtained from POD analysis, we can obtain $M = N/2$ such oscillators. These oscillators or linked modes can also be identified using the procedure described in the work of Sieber *et al.* (2016). It is noteworthy that such oscillators can only be extracted for periodic components of the flow captured by POD. In this chapter, we track the variations in the modal fluctuation energy.

The temporal oscillations associated with each oscillator can be described by their coefficients in the complex plane. Conjugate mode pairs $(\boldsymbol{\phi}_{2j-1}^{\boldsymbol{u}}, \boldsymbol{\phi}_{2j}^{\boldsymbol{u}})$ with temporal coefficients (a_{2j-1}, a_{2j}) can be represented on the complex plane as,

$$z_m = a_{2j-1} + i a_{2j} = r_m \exp(i\theta_m), \tag{8.2}$$

DISTRIBUTION A: Distribution approved for public release.

where $m = \mathrm{I}, \mathrm{II} \ldots M$, $j = 1, 2 \ldots N/2$, $r_m = |z_m|$, and $\theta_m = \angle z_m$. Throughout this chapter, the oscillators will be numbered in roman numerals, {I, II ... M} denoted by index m to distinguish from mode numbering, $\{1, 2, \ldots N\}$ denoted by index j. Each oscillator m is associated with temporal coefficient defined on the complex plane (z_m) which consists of an odd-number mode with coefficient a_{2j-1} and an even-number mode with coefficient a_{2j}. The temporal coefficient corresponding to the mean flow ($\boldsymbol{u}_0$) is fixed to unity, $z_0 = 1$. Equivalent to Eq. (8.1), we can recover the velocity field in terms of oscillators and their associated temporal coefficients as

$$\boldsymbol{u}(\boldsymbol{x}, t) = \boldsymbol{u}_0(\boldsymbol{x}) + \sum_{m=1}^{M} \left[\Re(z_m) \boldsymbol{\phi}^{\boldsymbol{u}}_{2m-1} + \Im(z_m) \boldsymbol{\phi}^{\boldsymbol{u}}_{2m} \right], \tag{8.3}$$

where $\Re(z_m)$ and $\Im(z_m)$ represent real and imaginary components of the temporal coefficient z_m, respectively.

In the baseline case, oscillators follow a natural limit-cycle, the generalized dynamics of which is described by the Stuart–Landau equation as

$$\dot{z}^b_m = z^b_m(\lambda_m - |z^b_m|^2 + i\Omega^b_m), \tag{8.4}$$

with $z^b_m = r^b_m \exp(i\theta^b_m)$, $\lambda_m = (\overline{r^b_m})^2$ and Ω^b_m is the oscillator frequency, where $\bar{}$ denotes time average. Here, the superscript $()^b$ denotes the baseline case. From a dynamical system point of view, the individual oscillators in the baseline case do not exhibit any interactions with each other. The conjugate mode pairs of each oscillator exchanges energy among themselves to describe the self-sustaining behavior of each coherent structure in unsteady fluid flows. Similar to the present analysis, the work of Bagheri (2013) demonstrates the use of Stuart-Landau equation to perform multi-scale Koopman expansion of the cylinder wake. Stable limit cycles are a characteristic feature of systems exhibiting such self-sustaining oscillations (Strogatz 2014). This self-sustaining behavior may not be true for systems with chaotic limit-cycles (attractors) or turbulent flows.

The baseline temporal dynamics of the individual oscillators are independent of each other as given by Eq. (11.7) and does not involve any coupling between oscillators. To highlight interactions between oscillators, we introduce perturbations impulsively to the baseline temporal coefficients of oscillators at $t = t_0$. These perturbations introduced in DNS to the baseline shedding state leads to the emergence of nonlinear interactions and energy exchange among the oscillators resulting in fluctuations described by

$$z'_m = a'_{2j-1} + i a'_{2j} = \epsilon'_m r^b_m \exp(i[\theta^b_m + \theta'_m]), \tag{8.5}$$

where $()'$ denotes the perturbation quantity and $\epsilon'_m(t_0)$ and $\theta'_m(t_0)$ are the impulsively added amplitude and phase perturbations to the mth oscillator, respectively. The total temporal coefficient for each oscillator in the perturbed case can be described by combining the baseline temporal coefficients and fluctuation as

$$z_m = r_m \exp(i\theta_m) = z^b_m + z'_m. \tag{8.6}$$

The oscillator model highlighting perturbations introduced is shown in Figure 68 (a). The blue circle in the Figure indicates the natural limit-cycle of oscillator m with baseline temporal coefficient (z^b_m) lying on it. The perturbations in amplitude (ϵ'_m) and phase (θ'_m) result in a total temporal coefficient (z_m) as shown. We need to consider three main factors to characterize the perturbations introduced, (1) the oscillator number m we introduce the perturbations to, (2) the initial amplitude of perturbation $\epsilon'_m(t_0)$ of the oscillators and (3) the initial phase perturbation size $\theta'_m(t_0)$ of the oscillators. Once these

DISTRIBUTION A: Distribution approved for public release.

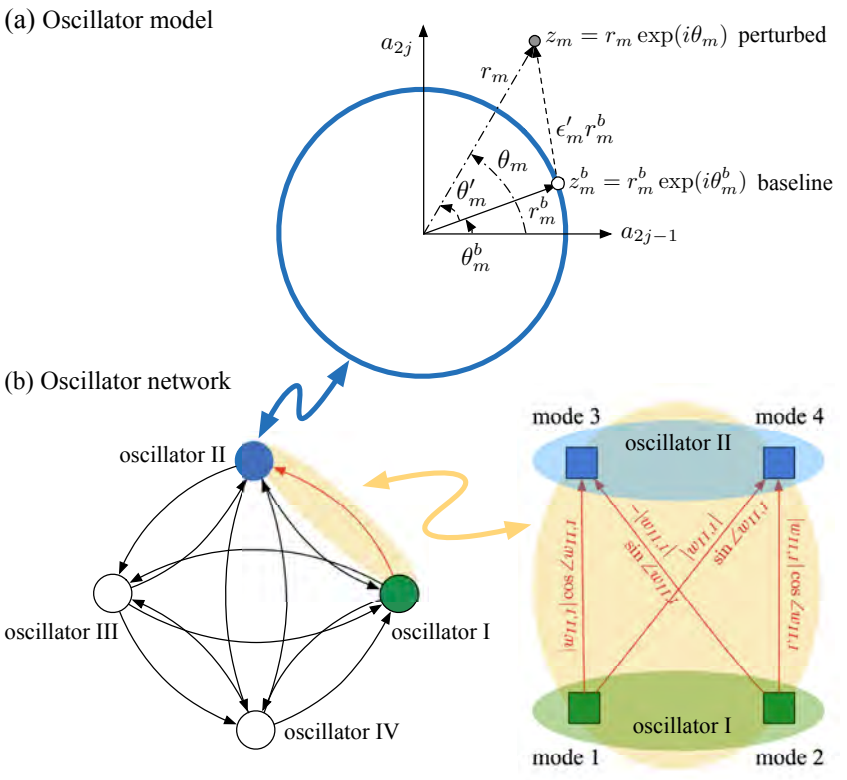


Figure 19: (a) Modal oscillator model in complex space and (b) oscillator network. The circle in oscillator model describes the limit-cycle trajectory of the oscillators (z^b_m) and the perturbed temporal coefficients (z_m). The nodes of the network correspond to oscillators with directed edges showing interactions. Modal interactions corresponding to edge from oscillator I to II highlighted in red.

factors for introducing impulse perturbations to the flow are determined, using Eq. (8.5) and (11.11) in (11.6), the initial velocity field for DNS of the perturbed case is prescribed.

Once the perturbations are introduced, the unsteady flow tries to relax to the baseline flow (natural limit cycle) by distributing the effect of the perturbation among the POD modes. The resulting fluctuations in the temporal dynamics of the modes are attributed to the interactions between them resulting primarily from the convective term of the Navier–Stokes equation. These interactions causes energy transfers among the modes in unsteady fluid flow. Thus, the baseline flow serves as the natural (limit cycle) attractor for the perturbed flow with the transient temporal dynamics of the modes capturing the interactive physics involved. As the POD modes are orthogonal to each other, the temporal coefficients in the perturbed case can be extracted by projection as $a_j = (\boldsymbol{u} - \boldsymbol{u}_0, \boldsymbol{\phi}^{\boldsymbol{u}}_j)$. We then construct the temporal coefficients in complex plane z_m for each oscillator m using Eq. (8.2). To capture the fluctuating amplitude and phase of the temporal coefficients, we track the normalized fluctuation

$$\zeta_m = z'_m / z^b_m = \epsilon'_m \exp(i\theta'_m) \tag{8.7}$$

Thus, the normalized fluctuation tracks the amplitude $\epsilon'_m(t)$ and phase $\theta'_m(t)$ fluctuation

DISTRIBUTION A: Distribution approved for public release.

of the modal oscillators due to interactions. It must be noted that the amplitude fluctuation of the oscillators is related to the variation in oscillator energy $E'_m(t)$ compared to the baseline given by

$$E'_m = \frac{1}{2}\left(|z_m|^2 - |z_m^b|^2\right) = \frac{1}{2}[(1+\epsilon'_m)^2 - 1](r_m^b)^2, \qquad E'_m(t_0) = \beta_m E^b, \tag{8.8}$$

where $m = \{\mathrm{I}, \mathrm{II} \ \ldots \ M\}$, $E'_m(t_0)$ represents the oscillator energy input, E^b is the total baseline modal energy given by $E^b = \sum_{j=1}^{N}(a_j^b)^2/2 = \sum_{m}^{N/2}(r_m^b)^2/2$ and β_m represents the factor of input energy introduced to oscillator m at initial time (t_0) compared to the total baseline energy. If no amplitude perturbation is introduced in the simulation, $E'_m(t_0) = 0$. If only phase perturbation is introduced in the simulation, though oscillator energy input is zero, variation in oscillator energy may be obtained due to oscillator interactions. To be consistent with the tracking of normalized fluctuation ζ_m in Eq. (8.7), the normalized quantities $E'_m/(r_m^b)^2$ and a'_j/r_m^b are used to track the energy perturbations of the oscillators and amplitude perturbations of the modes, respectively while θ'_m tracks the phase of the oscillators. This normalization is so as to facilitate observations regarding tracking the energy transfer between modes. Multiple combinations of oscillator energy input and phase perturbations can be used to quantify energy exchange in each perturbed case.

8.2. Networked oscillator representation

Using the oscillators extracted from POD and their normalized fluctuations extracted from perturbations impulsively added to baseline flow, we can now create a network-theoretic representation of the system. A network or graph $\boldsymbol{G} = \{\mathcal{V}, \mathcal{E}, w\}$ consists of a set of nodes V connected by edges $\mathcal{E}$ with associated edge weights w (Newman 2010). In the present study, we consider the oscillators (modes pairs) characterizing unsteady fluctuations to be the nodes $(\mathcal{V})$ of the graph and the interaction between them as edges $(\mathcal{E})$. To characterize the interactions between oscillators, i.e., to define the edges $\mathcal{E}$ and associated edge weights w of the network, we need a model that entails coupling between oscillators. This leads us to a networked oscillator model of M linearly coupled oscillators given by

$$\dot{\zeta}_m = \sum_{n=1}^{M}[\boldsymbol{A}_{\mathcal{G}}]_{mn}\,(\zeta_n - \zeta_m) = -\sum_{n=1}^{M}[\boldsymbol{L}_{\mathcal{G}}]_{mn}\zeta_n, \tag{8.9}$$

where $[\boldsymbol{A}_{\mathcal{G}}]_{mn}$ and $[\boldsymbol{L}_{\mathcal{G}}]_{mn}$ are the adjacency and the in-degree Laplacian matrices, respectively. The networked oscillator model considers diffusion processes on networks and is used as a simple model for examining the spread of kinetic energy among oscillators. It should be emphasized that this is a linear model. The dynamics of normalized fluctuations of temporal coefficients are nonlinear in general due to convective physics in unsteady fluid flows and networked oscillator model is a linear approximation, reminiscent of Koopman ideas (Mezic 2013). For a set of M oscillators, the adjacency matrix $\boldsymbol{A}_{\mathcal{G}} \in \mathbb{C}^{M\times M}$ concisely describes the network connectivity given as

$$[\boldsymbol{A}_{\mathcal{G}}]_{mn} = \begin{cases} w_{mn} = |w_{mn}|\exp\left(i\angle w_{mn}\right) & \text{if there is an edge from } n \text{ to } m \\ 0 & \text{otherwise.} \end{cases} \tag{8.10}$$

In our formulation, no self-loops are present in the network connectivity of oscillators. Thus, the oscillators have no influence over themselves and hence the diagonal entries of the adjacency matrix are zero. The rows of the adjacency matrix indicates the dependence

DISTRIBUTION A: Distribution approved for public release.

of the oscillators n on the dynamics of oscillators in column m, i.e., w_{mn}. The in-degree (k_m) of a node m represents the summation of the incoming weights of the edges connected to it given by $k_m = \sum_{n=1}^{M} [\boldsymbol{A}_\mathcal{G}]_{mn}$. The (in-degree) graph Laplacian is closely related to the adjacency matrix as $\boldsymbol{L}_\mathcal{G} = \boldsymbol{D}_\mathcal{G} - \boldsymbol{A}_\mathcal{G}$ where $\boldsymbol{D}_\mathcal{G}$ is a diagonal matrix with elements equal to the in-degree of the nodes, $\boldsymbol{D}_\mathcal{G} = \mathrm{diag}([k_m]_{m=1}^{M})$. Graph Laplacian is a discrete analog of the continuous Laplacian operator. In the current study, it encodes structural properties of the networked oscillator model. Diffusion processes on networks are neatly described by this graph Laplacian.

Using the knowledge of time evolution of normalized fluctuations from DNS for each oscillator, i.e. $\boldsymbol{\zeta}$ and $\dot{\boldsymbol{\zeta}}$, we use a linear regression procedure to compute the adjacency matrix weights. Linear regression allows us to examine the relationships between predictive (independent) and response (dependent) variables. It enables us to find the best possible linear fit for estimating the response of a system by regressing the dynamics onto the predictor variables. Here, the time derivative of the normalized fluctuations of the individual oscillators ($\dot{\zeta}_m$) are the response variables and ($\zeta_n - \zeta_m$) are the predictor variables. The result of such a multiple regression over m oscillators yields the adjacency matrix weights for the networked oscillator model. It must be noted that as the temporal coefficient associated with the mean flow is constrained to unity, $z_0' = 0$ and does not contribute to normalized fluctuations of the modes. Thus, mean flow is considered as an isolated node in this formulation.

The oscillator network representation is shown in Figure 68 (b). The individual oscillators form nodes of the network. The oscillator model in Figure 68 (a) highlights the temporal coefficient dynamics associated with the nodes of the network, here, oscillator II (shown in blue). The adjacency weights obtained from linear regression w_{mn} can be decomposed into a magnitude ($|w_{mn}|$) and phase ($\angle w_{mn}$). The magnitude of the edge weights signify the influence of oscillator n on oscillator m as illustrated in Figure 68 (b). The phase of the edge weights emphasize the individual modal contributions in oscillator interactions. In particular, it highlights the phase advances or delays between modes of interacting oscillators. Consider the complex-valued edge weight from oscillator I (mode pair 1 and 2) to oscillator II (mode pair 3 and 4), i.e. $w_{\mathrm{II,I}} = |w_{\mathrm{II,I}}| \exp \angle w_{\mathrm{II,I}}$ shown in red in Figure 68 (b). This edge corresponds to 4 interactions, mode 1 $\rightarrow$ mode 3, mode 1 $\rightarrow$ mode 4, mode 2 $\rightarrow$ mode 3, mode 2 $\rightarrow$ mode 4. The odd-odd mode interactions and even-even mode interactions are given by $|w_{n,m}| \cos \angle w_{n,m}$ while the odd-even and even-odd mode interactions are given by $|w_{n,m}| \sin \angle w_{n,m}$ and $-|w_{n,m}| \sin \angle w_{n,m}$, respectively.

As such, the networked oscillator model describes both amplitude and phase dynamics of collective oscillation and captures the energy exchange and phase dynamics among the modes. As discussed earlier, the model follows from impulsively adding perturbations to DNS to characterize oscillator interactions. In §8.3, we describe the application of the oscillator network-based formalism to a canonical problem of two-dimensional incompressible flow over a cylinder. The model can also serve as a foundation to perform network-inspired feedback control as will be discussed in §**??**.

8.3. Application to cylinder flow

Two-dimensional flow over a cylinder is a canonical problem in fluid mechanics and well-studied from the viewpoint of modal decomposition techniques. The coherent structures identified are well-reported in literature and forms an excellent testbed for our analysis. Also, for the cylinder flow problem considered in this study, the POD modes are the same as DMD modes as demonstrated in work by Chen *et al.* (2012). Thus, as an alternative to POD for the current approach, DMD can be performed. To ensure

DISTRIBUTION A: Distribution approved for public release.

orthogonality of the modes required for the current approach, recursive DMD can be performed to obtain orthogonal pure frequency modes which has niche applications in analyzing nonlinear transient dynamics (Noack *et al.* 2016*a*).

8.4. Computational setup

We perform DNS of incompressible flow over a cylinder using the immersed boundary projection method (Taira & Colonius 2007*a*) at a diameter-based Reynolds number of $Re = 100$. This method employs a Cartesian grid with the immersed body represented using a boundary force (satisfying no-slip) applied along the boundary surface. We take advantage of the multi-domain technique of immersed boundary projection (Colonius & Taira 2008*a*) which is based on vorticity-streamfunction formulation. The innermost domain is chosen as $-1 \leqslant x \leqslant 29, -15 \leqslant y \leqslant 15$ with the grid resolution of 600×600 grid points. The outermost domain is chosen far enough so as to not affect the results in the near-field. Uniform flow is prescribed at the domain boundaries. For time integration, the method uses an implicit Crank-Nicholson scheme for the viscous term and an Adam-Bashforth discretization for the convective term.

From the simulation, the drag coefficient (C_D) and lift coefficient (C_L) are computed as

$$C_D = \frac{F_D}{\frac{1}{2}\rho U^2 d}, \quad C_L = \frac{F_L}{\frac{1}{2}\rho U^2 d}$$

where, F_D and F_L are the drag and lift forces on the cylinder, d is the cylinder diameter and U is the free stream velocity field. These are obtained by integrating the boundary force representing the cylinder in immersed boundary formulation. We obtain a Strouhal number of $St = 0.164$, drag coefficient $C_D = 1.35 \pm 0.009$ and lift coefficient $C_L = \pm 0.325$ which agree well with literature (Liu *et al.* 1998). The flow exhibits vortex shedding behavior in the cylinder wake as shown by the instantaneous vorticity field in Figure 70 (a). Such vortex shedding characterizes a von Kármán vortex street due to the repetitive pattern of vortices in the unsteady wake. The time-averaged (mean) vorticity field is shown in Figure 70 (b). We then perform POD decomposition using method of snapshots (Sirovich 1987) using velocity field data ($\boldsymbol{u}$) gathered from DNS. The modes and temporal coefficients obtained are in agreement with those from the work by Noack *et al.* (2003*a*).

8.5. Baseline (unperturbed flow)

The extracted spatial modes ($\boldsymbol{\phi}_j^{\boldsymbol{\omega}} = \nabla \times \boldsymbol{\phi}_j^{\boldsymbol{u}}$) and associated temporal coefficients (a_j^b) are shown in Figure 70 (c) and (d) respectively. We define the conjugate mode pairs with commensurate energy levels as independent oscillators in our formulation. Oscillator I constitutes mode pair $(1, 2)$, oscillator II constitutes mode pair $(3, 4)$ and so on. The mode pairs (oscillators) are ordered in terms of decreasing energy content, $E_j = (a_j^b)^2/2$, in Figure 70 (top to bottom). As the first eight POD modes captures 99.98% of the modal kinetic energy, we choose $N = 8$ for our analysis. The energy content of oscillator I, II, III and IV are $96.87\%, 2.18\%, 0.88\%$ and 0.05%, respectively. The oscillator temporal coefficients in the complex plane (z_m^b) corresponding to temporal coefficients of the individual mode pairs from Eq. (8.2) is shown in Figure 70 (e). For this canonical problem, the frequency associated with the higher-order oscillators are harmonics of the lowest-order oscillator, $\Omega_n^b = n\Omega_{\mathrm{I}}^b$. The frequency of oscillator I corresponds to the Strouhal number associated with the natural shedding cycle of the flow, $\Omega_{\mathrm{I}} = 0.164$. As the frequency ($\Omega_n^b$) associated with the temporal coefficients of the mode pairs increases, the size of the spatial modal structures decreases. With regards to the spatial structure, oscillators II and IV are symmetric with respect to the streamwise

DISTRIBUTION A: Distribution approved for public release.

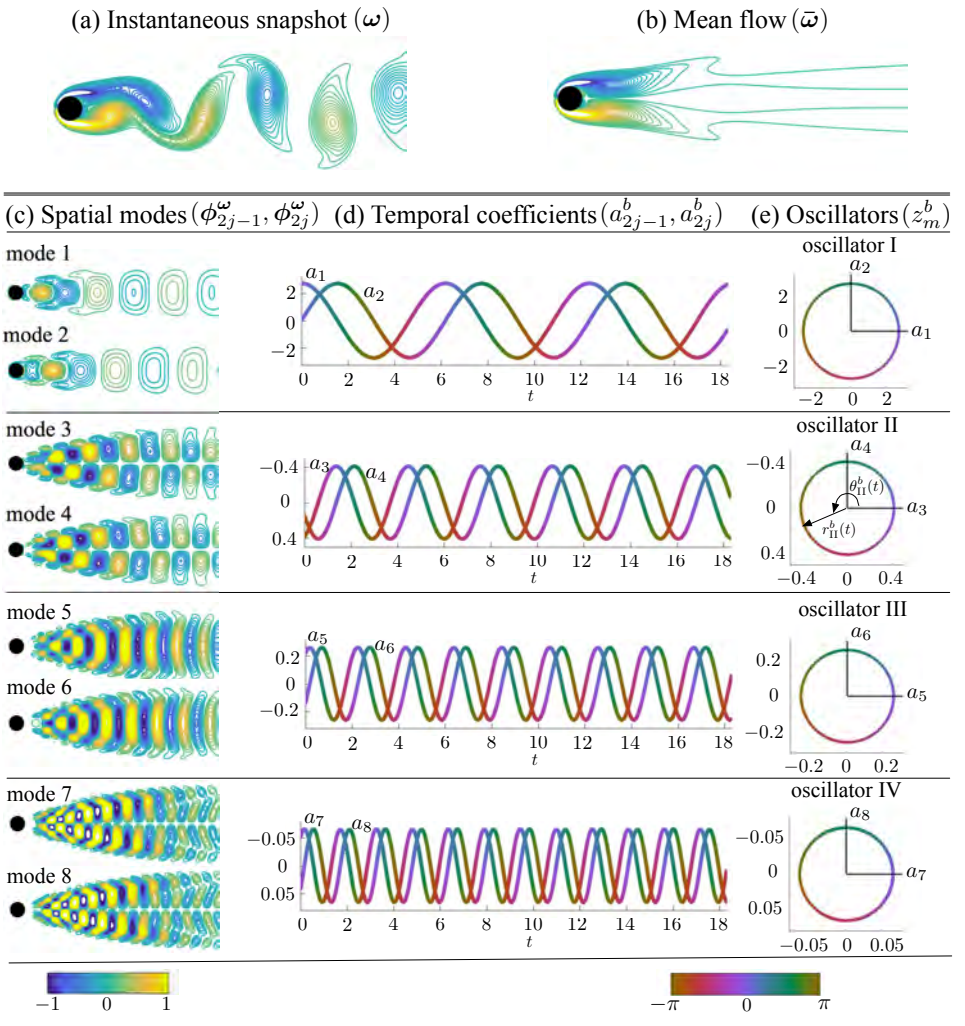


Figure 20: (a) Instantaneous snapshot of vorticity field and (b) time-averaged (mean) vorticity field. Proper orthogonal decomposition (POD) technique applied to cylinder flow problem results in (c) spatial modes and (d) temporal coefficients associated with the modes. (e) temporal coefficients of oscillators in complex plane. The colorbar of the spatial modes indicates the contour level and colorbar of the temporal coefficients and oscillators indicates phase of the oscillators varying from $[-\pi, \pi]$ in a periodic limit-cycle manner. The colors (Thyng *et al.* 2016) of the lines in Figure 70 (d) and (e) helps us easily identify the correspondence of the temporal coefficients in the time domain and phase space. Polar coordinates associated with the oscillator II is shown.

coordinate while oscillators I and III are anti-symmetric. From modal decomposition of the baseline case, we can extract the the temporal coefficients of the oscillators z^b_n associated with baseline frequency Ω^b_n. The oscillators in the baseline flow are associated with limit cycle temporal dynamics which can be described by Eq. (11.7) independent of each other with no coupling between them. As discussed earlier, each oscillator exhibits self-sustaining behavior in the baseline flow due to exchange of energy between their

conjugate mode pairs. Thus, the individual oscillators do not exhibit any interactions in the baseline (unperturbed) flow.

8.6. **Perturbed flow**

For the perturbed flow, additional energy and phase perturbation impulsively added to the simulation through the initial condition causes fluctuation in temporal coefficients of the modes. The projected coefficients $a_n = (\boldsymbol{u} - \boldsymbol{u}_0, \boldsymbol{\phi}_n^{\boldsymbol{u}})$ from the perturbed case are extracted from DNS and the normalized fluctuation $\zeta_m(t)$ is tracked using the networked oscillator model discussed in §11.2. We expect that as the perturbation convects downstream and out of the computational domain, the normalized fluctuation of the oscillators will decay to zero and the perturbed field will return to the baseline state. However, addition of a perturbation to the flow results in a constant phase shift of each oscillator in the final limit cycle compared to the limit cycle without perturbation.

In two-dimensional unsteady cylinder flow, the leading POD mode pair corresponding to oscillator I contains maximum energy and any deviations in the final limit cycle is immediately detected by the oscillator I phase (θ_{I}). Thus, to construct the normalized fluctuation time history, we align the phase of oscillator I (holding a majority of the energy) for the perturbed case and the baseline case as $\zeta_m(\theta_I) = [z_m(\theta_I) - z_m^b(\theta_I)]/z_m^b(\theta_I)$. Once the normalized fluctuation ζ_m is constructed, we also compute its time derivative $\dot{\zeta}_m$. We then construct the library of functions $\zeta_n - \zeta_m$ for each oscillator m. In general, the library of functions could also include higher-order terms $(\zeta_n - \zeta_m)^2, (\zeta_n - \zeta_m)^3 \ldots$. We perform a simple regression procedure to obtain the network structure $[\boldsymbol{A}_{\mathcal{G}}]_{mn}$. Once the coupling function (network structure) is obtained, we solve the linear networked oscillator model in Eq. (8.9) with a prescribed initial condition and compare the fluctuations with those obtained in DNS. Following this procedure, we discuss some representative cases with amplitude and phase perturbations impulsively added to highlight the approach.

We first introduce an amplitude perturbation to oscillator II. Oscillator II is associated with a frequency corresponding to the second harmonic of the natural shedding frequency of the flow. Introducing perturbations at this frequency perturbs not only the natural shedding frequency but also the oscillators associated with higher harmonics of the flow due to interactions. Considering $\beta_{\mathrm{II}} = 0.2$, additional 20% of baseline modal energy is added to oscillator II. No perturbation in phase is added, i.e. $\theta'_{\mathrm{II}}(t_0) = 0$. With these initial perturbations determined, we can construct the initial flow field required for DNS of the perturbed case. We can observe that the initial vorticity field as shown in 71 (a) for the perturbed case is different as compared to Figure 70 (a) for the baseline. Here, as oscillator II is energized, we notice the prevalence of spatial vortical structures associated with mode pair $(3, 4)$. The vorticity magnitude associated with these structures increases due to the added energy and correspondingly appear in the initial vorticity field reconstruction for the perturbed case. As the introduced perturbation in oscillator II convects downstream in the numerical simulation, it interacts with other oscillators. Using the procedure highlighted above, we construct the normalized fluctuations $\boldsymbol{\zeta}$ and extract the adjacency matrix $\boldsymbol{A}_{\mathcal{G}}$ that highlights network interactions.

The magnitude and phase of the adjacency matrix are shown in Figure 71 (b). In this example, the dynamics of oscillator I (leading oscillator) is not affected much by the other oscillators. The dynamics of oscillator II and III show maximum dependence on oscillator I. This is consistent with our expectation as most of the energy in the flow is contained in oscillator I. Also, if we evaluate the strength of the incoming edge weights of each oscillator ($\sum_n w_{nm}$) for this perturbed case, oscillator III shows maximum interactions with the other oscillators. The complex matrix structure of the extracted network summarizes the interactions in the flow on introduction of the impulse

DISTRIBUTION A: Distribution approved for public release.

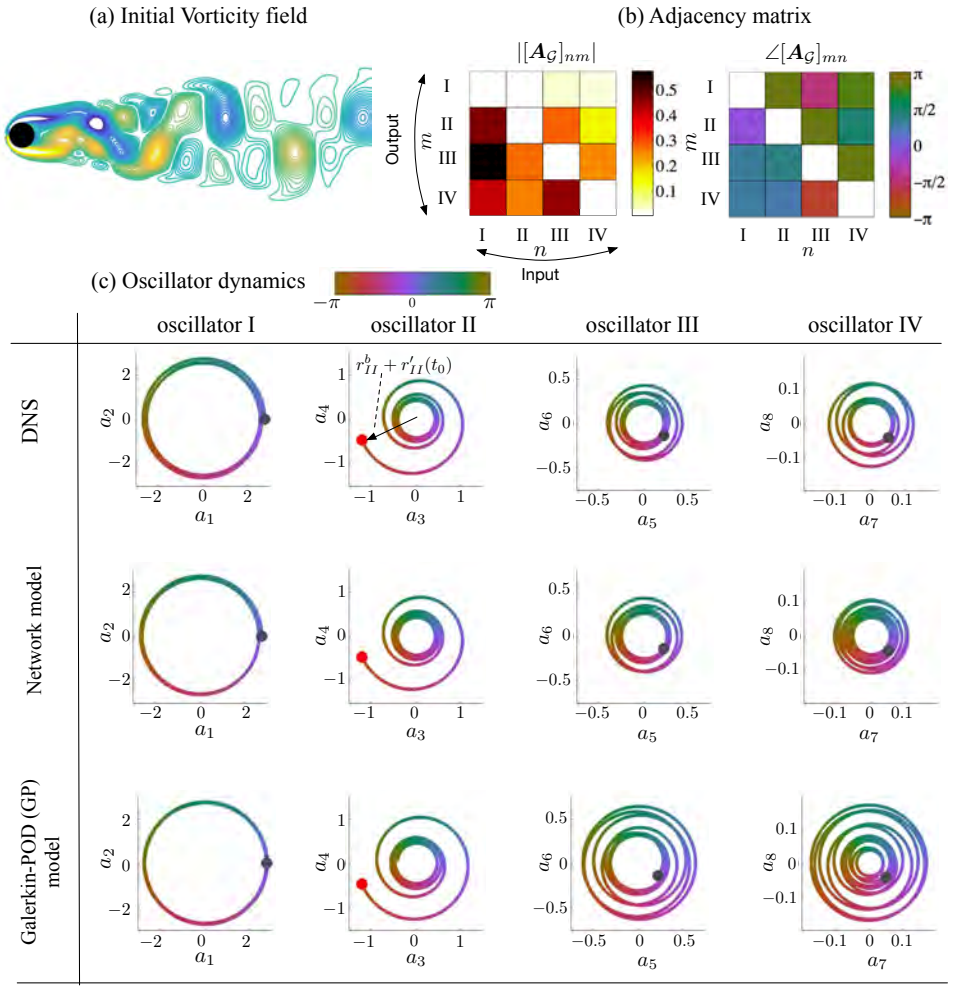


Figure 21: Amplitude perturbation introduced at oscillator II. (a) Initial vorticity field corresponding to amplitude perturbation, (b) Adjacency matrix of networked oscillator model. (c) Oscillator dynamics from DNS, networked oscillator model and Galerkin-POD model (color of temporal dynamics of oscillators indicates phase). Red dot indicates the initial perturbation.

perturbations. To reveal individual modal contributions to the oscillator interactions, we evaluate the phase of the edge weights in Figure 71 (b). For dynamics of oscillator II, i.e., mode pair $(3, 4)$, the phase of the edge weights indicates that interactions between modes $1 \to 3$, $2 \to 4$, $8 \to 3$ and $7 \to 4$ are larger than other modal interactions. For this perturbed case, in general, phase advancing effects are predominant for the oscillators affecting each other.

Once the network structure is obtained, we can solve the linear networked oscillator model with the prescribed initial condition corresponding to the perturbation. For comparison, we also implement Galerkin projection of the incompressible Navier-Stokes equations onto the POD modes to construct the Galerkin-POD reduced-order model

DISTRIBUTION A: Distribution approved for public release.

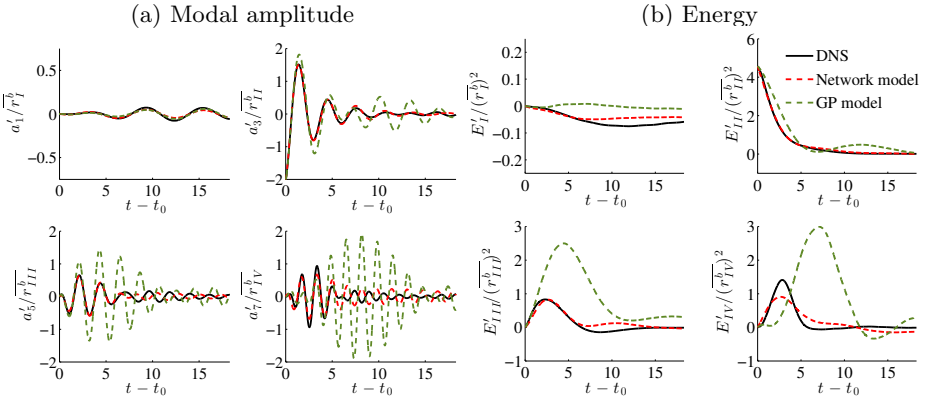


Figure 22: (a) Modal amplitude and (b) energy tracking for amplitude perturbation introduced at oscillator II.

(Noack *et al.* 2011; Holmes *et al.* 2012*a*). The resulting galerkin system is given by

$$\dot{a_j} = \sum_{k=1}^{N} \psi_{jk} a_k + \sum_{k,\,l=1}^{N} \chi_{jkl} a_k a_l, \quad j = 1, 2, \ldots N, \tag{8.11}$$

where ψ_{jk} and χ_{jkl} are the linear and quadratic coefficient term, respectively. Here, the linear coefficient term contains diffusive physics between the modes and mean while the quadratic term contains the convective physics. The networked oscillator model combines the effect of both diffusion and convection into the network structure extracted. Both the models require the specification of initial condition of the temporal coefficients of the mode pairs only.

The oscillator dynamics from DNS (reference) and those predicted from networked oscillator and Galerkin-POD models are shown in Figure 71 (c). We notice that the initial amplitude of temporal coefficients in oscillator II corresponding to the red dot, $r_{\mathrm{II}}(t_0)$, is larger for the perturbed case compared to baseline case. However, the initial phase of the oscillator II is the same as that of the baseline case as no phase perturbation is introduced. The initial amplitude and phase of the other oscillators are the same for the perturbed and baseline cases as no perturbations are introduced in these oscillators. It can be seen in Figure 71 (c) that the oscillator dynamics in the networked oscillator model agrees well with DNS trajectories, particularly the dynamics of oscillators I, II and III. The networked oscillator model and DNS show negligible differences in tracking the trajectory of oscillator II from the perturbed initial condition to its final limit cycle baseline orbit. The trajectory of oscillator IV shows some similarities with the DNS. Due to its low energy content, any small deviations in the dynamics of lower-order oscillators causes comparable changes in the trajectory of oscillator IV. On the other hand, the POD-Galerkin model overpredicts the fluctuations in oscillators. As discussed in the work by Rempfer (2000), restricted completeness property of the POD basis creates problems for integrating the POD-Galerkin system for any initial conditions lying outside the ensemble of trajectories used to compute the basis. Thus, small perturbations or disturbances created by numerical error in integration of the Galerkin model leads to incomplete representation of the Navier–Stokes operator.

There are two mechanisms of kinetic energy transfers in incompressible fluid flows, one from advection and the other from dissipation. In the incompressible Navier–Stokes

DISTRIBUTION A: Distribution approved for public release.

equations, the advective term $-(\nabla \cdot \boldsymbol{u})\boldsymbol{u}$ results in advection of kinetic energy in the flow while the diffusion term $\nu\nabla^2\boldsymbol{u}$ results in dissipation of energy (Noack *et al.* 2008). In the Galerkin model, the energy dissipation is modeled by the linear term and the advective energy transfer is modeled by the quadratic term (Noack *et al.* 2005). In the networked oscillator model, the diagonal terms of the Laplacian contribute to the dissipative physics of the individual oscillators and the off-diagonal terms model the advective physics. The absence of the mean in the network oscillator model results in energy exchanges only between the oscillators.

To further compare the details of the predicted trajectories, we track the fluctuations in modal amplitude and oscillator energy in Figure 72 (a) and (b) respectively. Networked oscillator model shows an excellent agreement in tracking the amplitude perturbation (a'_{2n-1}) and energy transfers (E'_m) compared to the DNS. Also, we see agreement in the long-time behavior of the fluctuations as the flow returns to the baseline state for the networked oscillator model. The Galerkin-POD model is not well designed for modeling the long-time behavior of modal fluctuations and hence is not expected to work well as time progresses. This is especially true as POD modes are used for the empirical galerkin formulation. As indicated by the gray dashed line, the Galerkin-POD model overpredicts these fluctuations and their associated time scales.

Similar to tracking amplitude perturbations, we can also track phase perturbations introduced in the flow. These phase perturbations are introduced by shifting the initial phase of the oscillators at t_0. As energy input is related to the amplitude of the perturbations, introducing only phase perturbations does not introduce any additional energy to the flow. We introduce a phase perturbation of size $\theta'_{\mathrm{II}}(t_0) = -\pi/2$ by shifting the phase associated with oscillator II at t_0 in the baseline case in the clockwise direction by $\pi/2$. The initial vorticity field resulting from the phase perturbation is shown in Figure 74 (a). Over time, both amplitude and phase of the oscillators get affected with the introduction of phase perturbations in DNS. These fluctuations are tracked and network structure is extracted as shown in Figure 74 (b). As seen from the magnitude of the edge weights, similar to the amplitude perturbation case, oscillator I dynamics is negligibly affected by the other oscillators. Oscillator IV does not contribute much to the dynamics of oscillators II and III. Deductions of modal interactions can be made from the phase corresponding to the edge weights.

The oscillator dynamics for the phase perturbation case is shown in Figure 74 (c). We notice that the phase perturbation introduced in oscillator II at t_0 results in a phase shift to an initial condition indicated by the red dot. As the introduced perturbation in oscillator II convects downstream, the oscillators synchronize to the limit cycle. Similar to the amplitude perturbation case, the oscillator dynamics in the networked oscillator model shows better agreement with the DNS than the Galerkin-POD model. The Galerkin-POD model incorrectly tracks the dynamics of oscillator II and overpredicts the higher-order oscillator dynamics. The amplitude fluctuations (a'_{2n-1}) of the individual modes and corresponding phase dynamics (θ'_m) tracked using the networked oscillator model agree well with DNS fluctuations as shown in Figure 75 (a) and (b), respectively. The Galerkin-POD model is unable to correctly track the phase of the oscillators and hence overpredicts the amplitude of the fluctuations.

The network structure of interactions can be extracted individually by tracking the trajectory of the oscillator fluctuations from DNS as demonstrated above. While the corresponding linear networked oscillator models can be built for each specific case, the resulting network structure obtained depends on the initial perturbations introduced. Thus, one may argue that such individually tuned models do not necessarily capture the overall interactions in response to a generic perturbation. A general network interaction

DISTRIBUTION A: Distribution approved for public release.

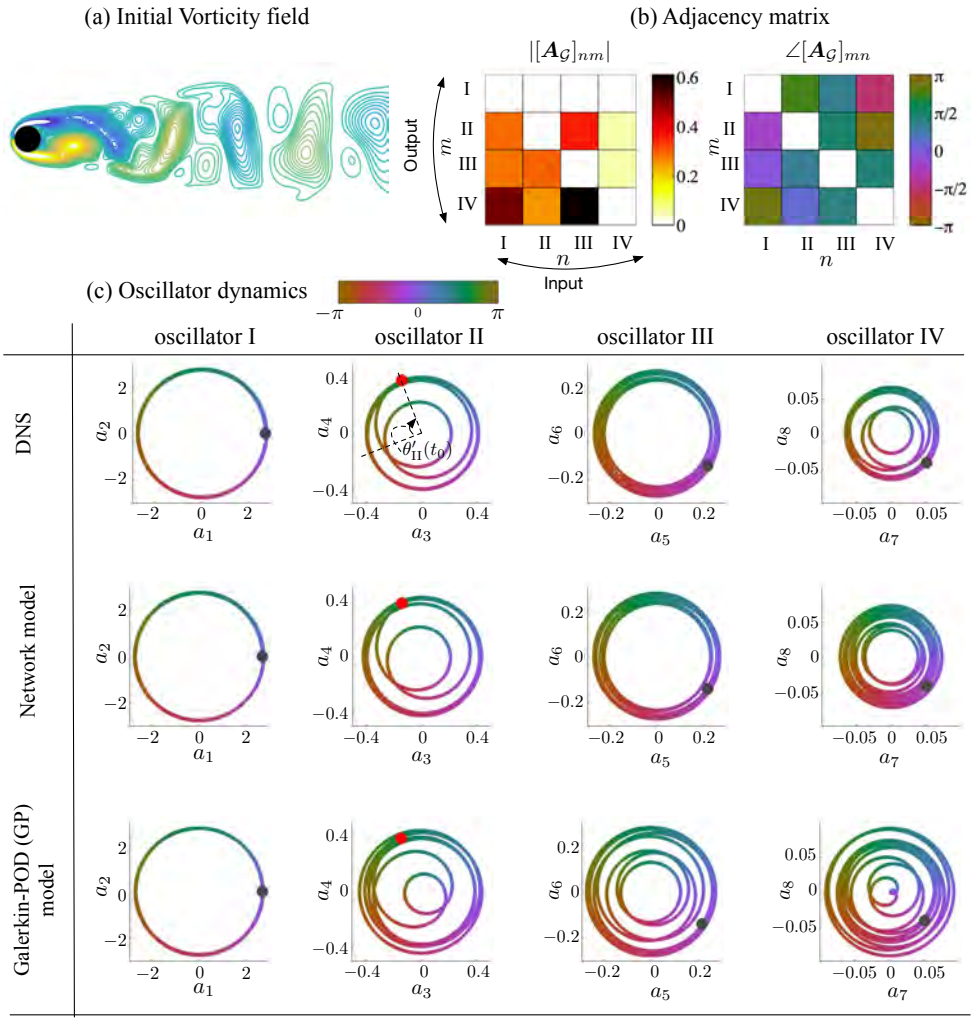


Figure 23: Phase perturbation introduced at oscillator II. (a) Initial vorticity field corresponding to phase perturbation, (b) Adjacency matrix of networked oscillator model. (c) Dynamics of DNS, networked oscillator model and Galerkin-POD model (color of oscillators indicates phase). Red dot indicates perturbed state.

model is desired not only to capture interactions for any combination of perturbations but also to design effective strategies for flow control. In this section, we describe our approach to build an aggregate network model that captures the fluid flow response to a variety of perturbation inputs.

We consider a range of perturbed flow cases by varying the amplitude and phase of perturbations to different combinations of oscillators. For a particular oscillator perturbed, we vary the energy input from 5% to 100% additional baseline modal fluctuation energy in the flow. We also vary the phase perturbation size (θ'_m) from $-2\pi/3$ to $2\pi/3$ to cover a broad coverage of plausible initial perturbations introduced. The range of amplitude and phase perturbations considered here are quite large. These large perturbations lead to the emergence of strong nonlinear interactions between the oscillators. For the variety

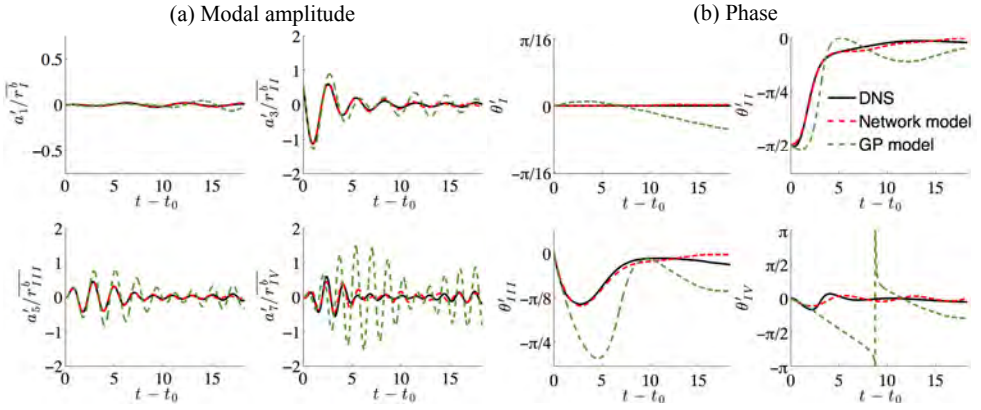


Figure 24: (a) Modal amplitude and (b) phase tracking for amplitude perturbation introduced at oscillator II.

of perturbation cases, we collect the oscillator fluctuation data from DNS. To build an aggregate network model, we concatenate the trajectories of all the perturbed cases obtained from DNS instead of individually tracking perturbations and monitoring the oscillator fluctuations in each perturbed case.

We then segregate the collection of data into training and test sets to perform cross-validation and evaluate the predictive capabilities of the network model. Varying fractions of the combined input-output data are randomly chosen as the training sets. For each training set, regression analysis is performed to extract a corresponding network model. We then examine the in-degree ($k_m = \sum_{n=\mathrm{I}}^{M} [\boldsymbol{A}_{\mathcal{G}}]_{mn}$) and out-degree ($k_n = \sum_{m=\mathrm{I}}^{M} [\boldsymbol{A}_{\mathcal{G}}]_{mn}$) of the network nodes for each model extrated. The variation of the network degrees with respect to the fraction of the chosen training set is shown in figure 76 (a). We observe that as the fraction of the training data used increases, convergence of the network degree is obtained. We also learn that the average in-degree increases from low-order oscillators to higher-order oscillators and the average out-degree decreases. Oscillators I and IV have maximum out-degree and in-degree, respectively. Oscillator I influences the other oscillators most, while oscillator IV is the most influenced. Oscillators II and III have a more balanced in and out-degrees indicating more balanced energy transfer for each mode.

We observe a convergence of network degree for a fraction of training data, $f_{\text{train}} \geqslant 0.8$. We randomly choose 80% of the combined input-output data as the training set and use the remaining 20% of the data to assess the performance of the network model extracted. Using the training set, the adjacency matrix for aggregate network model extracted is shown in figure 76 (b). The magnitude of the network reveals that the lower-order oscillators have more influence on the dynamics of the higher-order oscillators. This follows from our earlier discussion regarding the network node degrees and agrees with our intuition that energy is passed from lower-order oscillators to higher-order oscillators. Alternatively, we can construct the aggregate network model by considering complete trajectories of randomly chosen perturbed cases yielding a very similar aggregate network model.

We also compare the aggregate network model with network models built with selectively chosen training sets. In particular, we consider trajectories of perturbed cases of individual oscillators. Combining the perturbed trajectories for various amplitude and

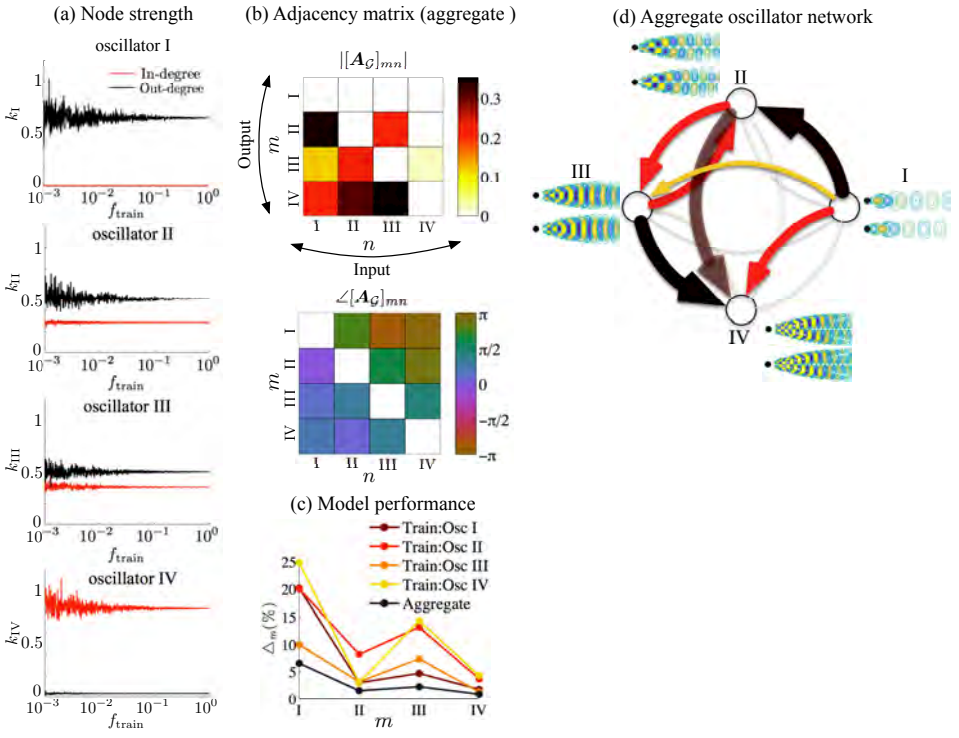


Figure 25: (a) Network degrees for varying fractions of training data. (b) Adjacency matrix of the aggregate network model and (c) the associated model performance. (d) Aggregate network structure of oscillator interactions. The color and thickness of the edges in the network represent the magnitude of the elements of $\boldsymbol{A}_{\mathcal{G}}$.

phase perturbation sizes for each oscillator, we build network models corresponding to oscillators I, II, III and IV. We use the same test data set as before to assess the performance of each of these models along with the aggregate model. We predict the time derivative of the normalized fluctuation $\dot{\zeta}_m$ with each of the models on the test data set and compare with DNS reference values. In figure 76 (c), we assess the prediction error through a normalized root mean square deviation ($\triangle_m$), given by

$$\triangle_m = \frac{1}{\max(|\dot{\zeta}_m|)} \sqrt{\frac{\sum_{k=1}^{n_t} |\hat{\dot{\zeta}}_m(k) - \dot{\zeta}_m(k)|^2}{n_t}}, \tag{8.12}$$

where n_t is the length of test data set. Here, $\dot{\zeta}_m(k)$ is the time derivative of oscillator m obtained from DNS and $\hat{\dot{\zeta}}_m(k)$ is the predicted time derivative value based on oscillator network model for the kth test data point. The root mean square deviation is normalized by the range of the measured data. We can observe from figure 76 (c) that the aggregate model achieves the lowest error level in predicting the normalized fluctuation time derivative. As the aggregate model contains more information of the general interactions based on various oscillator perturbations, it yields enhanced predictive capabilities for energy transfer amongst all oscillators. Thus, the aggregate network structure shown in figure 76 (d) yields a global oscillator interaction model. For the cylinder flow problem,

the passage of energy from lower-order oscillators to the higher-order ones is highlighted. Based on the modal interactions characterized using the aggregate network model, we design flow control strategies to suppress modal fluctuations.

Using the aggregate network structure shown in Figure 76 (a), we evaluate the performance of networked oscillator model in tracking the fluctuations of the perturbation cases examined in this study. The oscillator dynamics using the aggregate network oscillator model with introduction of amplitude perturbations corresponding to $\beta_m = 0.2$ to each oscillator individually is compared with DNS trajectories in Figure 77. The aggregate networked oscillator model is able to track the amplitude and phase dynamics for each case with reasonable accuracy. We can see that introduction of amplitude perturbation in oscillator I does not cause large deviations in the higher-order oscillators. Also, the prescribed energy input requires only slight deviation from the limit cycle orbit for perturbations introduced on oscillator I as it has large energy to begin with. Perturbation introduced to oscillator III results large fluctuations in oscillator II and IV while perturbation introduced to oscillator IV results in comparable fluctuations in oscillator III but really small deviations in oscillator I and II dynamics. All these features are well captured by the aggregate networked oscillator model. We also note in passing that the real part of the eigenvalues of aggregate network Laplacian are inversely related to the settling time of the individual oscillators perturbed.

We also show the comparison of predicted trajectories of aggregate networked oscillator model for phase perturbation cases of each oscillator corresponding to $\theta'_m(t_0) = -\pi/2$ in Figure 79. As oscillator I contains large amounts of energy, phase perturbations of θ'_{I} causes an effective phase perturbation of $m\theta'_n$ for oscillators II, III and IV. Thus, the phase perturbation on oscillator I can be viewed as combined phase perturbations on the higher-order oscillators that try to synchronize their phases with respect to oscillator I. Thus large deviations are observed in this case. Phase perturbations introduced in oscillator III causes fluctuations in oscillator II and IV while phase perturbations introduced in oscillator IV does not cause any fluctuations in lower-order oscillators. As with the amplitude perturbation cases, aggregate network model also captures the features associated with the phase perturbation cases. Thus one aggregate network structure is able to capture a multitude of perturbation cases yielding a global oscillator interaction model.

Before we perform design strategies for controlling the flow behavior using the aggregate network structure, we briefly discuss the residuals generated in the flow due to nonlinear interactions in each perturbed case. In this study, we only capture the fluctuation with respect to the baseline POD modes in the flow. The nonlinear interactions induced by the perturbations cause the flow to be modified resulting in generation of new flow structures not captured by the baseline POD modes. The magnitude of the residual energy increases with energy input β_m. Once all the residual flow structures exit the computational domain, the energy corresponding to the residuals sharply drops to zero. The present study does not describe the residual energy dynamics in the flow but characterizes the interactions between the baseline POD modes. Based on these modal interactions, we design flow control strategies to suppress the modal temporal dynamics in next chapter.

DISTRIBUTION A: Distribution approved for public release.

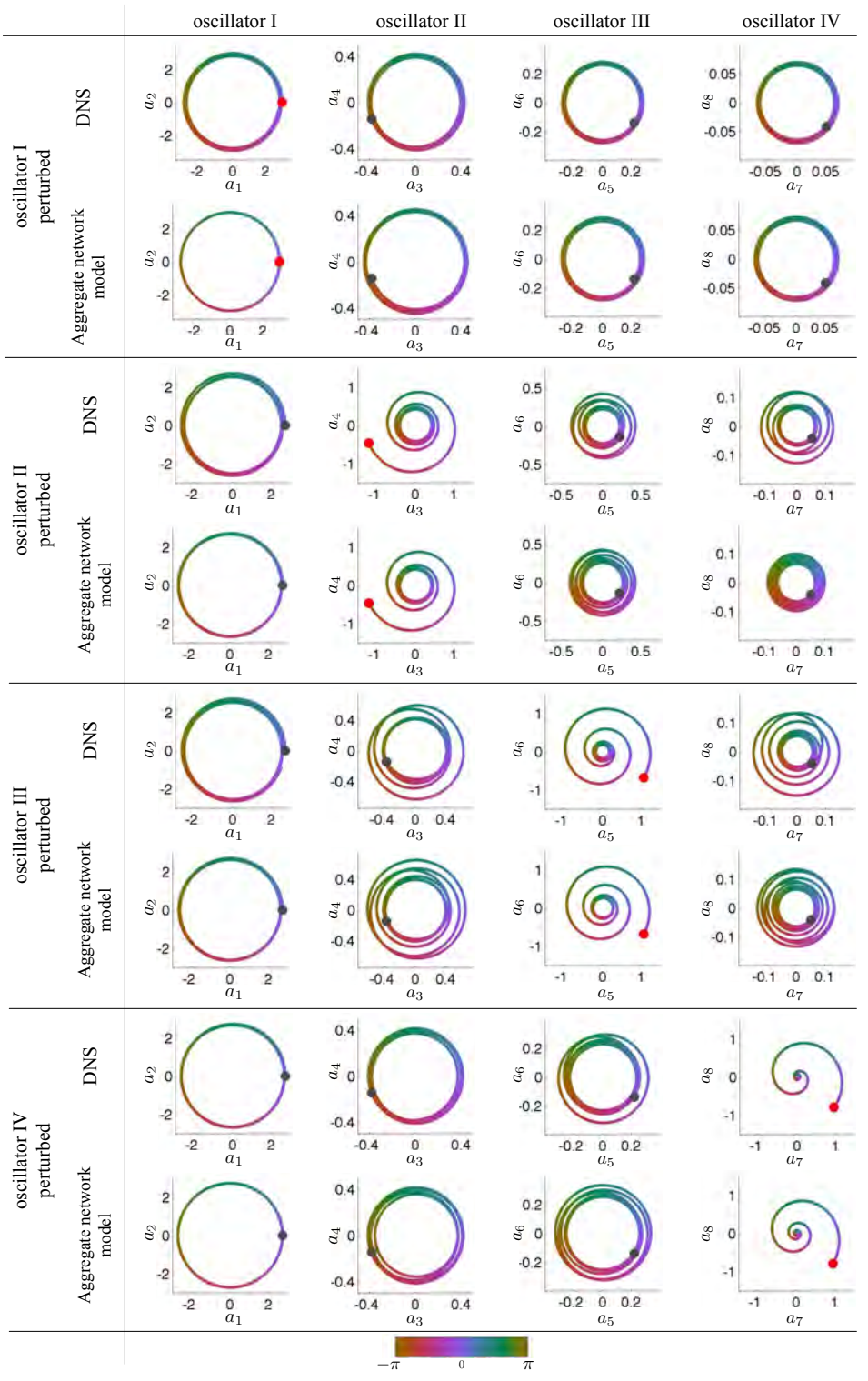


Figure 26: Oscillator dynamics of DNS and aggregate networked oscillator model for amplitude perturbations to oscillators corresponding to energy input of $\beta_m = 0.2$.

DISTRIBUTION A: Distribution approved for public release.

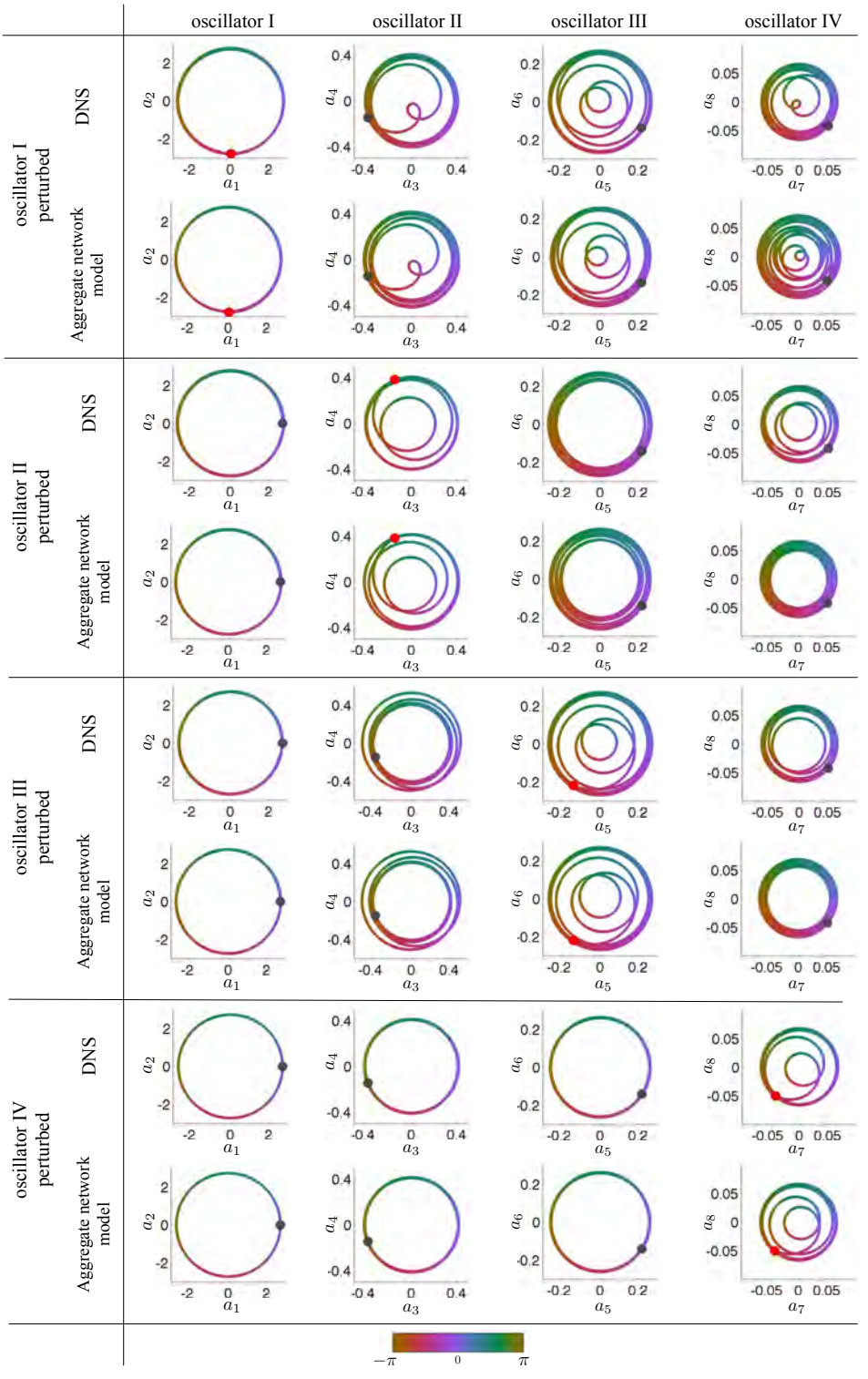


Figure 27: Oscillator dynamics of DNS and aggregate networked oscillator model for phase perturbations to oscillators corresponding to $\theta'_n(t_0) = -\pi/2$.

DISTRIBUTION A: Distribution approved for public release.

9. Interaction-based Feedback Control

With the global network interactions characterized and aggregate network model developed in the previous chapter, we develop a control strategy to suppress modal fluctuations. Suppression of modal oscillations is critical in reducing the wake unsteadiness. Roshko (1954) and Mao *et al.* (2015) reported the strong relationship between unsteadiness in the wake and the drag force acting on the bluff body. As mentioned previously, modal oscillations in the flow are reflected in the temporal coefficients associated with the modes. If the application of control forces the modal temporal coefficients to zero, the flow will approach the mean flow. However, the mean flow is not a steady solution to the Navier–Stokes equation in general, and hence the flow diverges from the mean towards the unstable steady state in this cylinder flow case.

In the seminal work of Noack *et al.* (2003*a*), it was shown that the mean flow and the unstable steady state are connected by a shift mode. The shift mode for the cylinder flow problem is shown in figure 28 (top left). This shift mode captures the transient dynamics between the onset of vortex shedding near the unstable steady state shown in figure 28 (bottom left) and the baseline mean flow of the periodic von Kármán vortex street in the globally stable limit cycle. This evolution takes place along the parabolic inertial manifold shown in grey. The vertical axis represents the change in the temporal dynamics associated with the shift mode with application of control. The drag coefficient also varies between the mean baseline value and the unstable fixed point value. A minimum drag coefficient is attained at the unstable steady state C_D^*, which also gives zero lift force on the cylinder. The kinetic energy associated with the shift mode varies as $a_\triangle^2$ and the drag force on the cylinder in the mean shift regime scales as $\sqrt{C_D'} \propto a_\triangle$, as shown in figure 28 where, $C_D' = C_D - C_D^*$. Thus, a mean shift towards the unstable steady state achieves a reduction in drag force.

We first develop a low-dimensional control framework based on the networked oscillator model to attenuate perturbations in the flow. We then extend the formulation to suppress the overall flow unsteadiness. As the networked oscillator model given by Eq. (8.9) is linear, we can exploit the use of the linear quadratic regulator (LQR) to control the modal fluctuations. Adding a forcing input to the networked oscillator model (in vector form), we arrive at

$$\dot{\boldsymbol{\zeta}} = -\boldsymbol{L}_{\mathcal{G}}\boldsymbol{\zeta} + \boldsymbol{B}\boldsymbol{v}, \tag{9.1}$$

where $\boldsymbol{\zeta} = [\zeta_{\mathrm{I}}, \zeta_{\mathrm{II}} \ldots \zeta_M]^T$, $\boldsymbol{v} \in \mathbb{C}^{M\times 1}$ is the forcing input and $\boldsymbol{B} \in \mathbb{R}^{M\times 1}$ is the actuation input matrix. The mth entry of $\boldsymbol{B}$ corresponds to forcing being added to oscillator m.

We implement an optimal full-state control with $\boldsymbol{v} = -\boldsymbol{K}\boldsymbol{\zeta}$ such that

$$\dot{\boldsymbol{\zeta}} = (-\boldsymbol{L}_{\mathcal{G}} - \boldsymbol{B}\boldsymbol{K})\boldsymbol{\zeta}, \tag{9.2}$$

where the gain matrix $\boldsymbol{K}$ is determined from the Riccati equation for LQR. Here, we assume that the POD mode pairs can be used as forcing inputs. While this viewpoint may appear as naive, it provides tremendous insights and guidance in designing localized actuation techniques. An optimal control strategy using LQR minimizes the quadratic cost function of the form,

$$J = \int_0^\infty \left[\boldsymbol{\zeta}(t)^T \boldsymbol{Q}\boldsymbol{\zeta}(t) + \boldsymbol{v}(t)^T \boldsymbol{S}\boldsymbol{v}(t)\right] \mathrm{d}t, \tag{9.3}$$

where $\boldsymbol{Q}$ and $\boldsymbol{S}$ are the state deviation and input usage weights respectively. Here, we set $\boldsymbol{Q} = \boldsymbol{I}$ and $\boldsymbol{S} = \sigma\boldsymbol{I}$ and consider a range of values for σ.

We can force the individual oscillators or a combination of oscillators. To aid the selection of which oscillators to force, we examine the movement of the eigenvalues

DISTRIBUTION A: Distribution approved for public release.

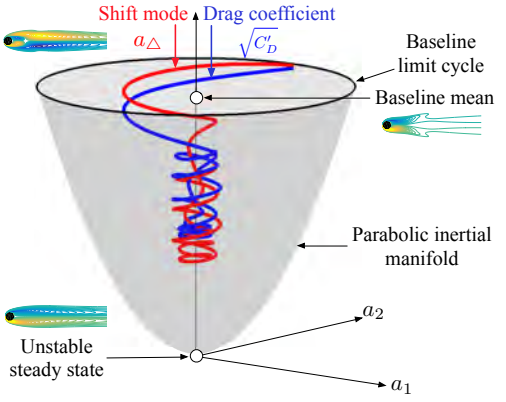


Figure 28: Shift mode temporal coefficient (red) and drag coefficient (blue) variation (indicated by arrows) with application of control. Parabolic inertial manifold shown in grey.

of $\boldsymbol{L}_\mathcal{G}$ with control for different combinations of input matrix $\boldsymbol{B}$ as shown in figure 29. Eigenvalues of Laplacian matrix $\lambda(-\boldsymbol{L}_\mathcal{G})$ reveal the dynamical characteristics of the system. As four oscillators are considered in this work, we obtain four eigenvalues $(\lambda_\mathrm{I} \ldots \lambda_\mathrm{IV})$ for varying σ in figure 29. Note that $\lambda_m \leqslant 0$ for all eigenvalues which is a characteristic feature of Laplacian-based systems.

We compare the response for input matrix $\boldsymbol{B}$ for both single oscillator input and some multiple oscillator input cases. For each input case, we determine the LQR gain matrix $\boldsymbol{K}$ for values of σ ranging from 0.1 to 1000 and examine the movement of eigenvalues of operator, $(-\boldsymbol{L}_\mathcal{G} - \boldsymbol{B}\boldsymbol{K})$ which govern the behavior of the controlled system. For the single-oscillator input cases, we observe large movement of the eigenvalue corresponding to the forced oscillator as σ is decreased, i.e., the real part of λ_m decreases when more forcing input is provided to oscillator m. We also notice that an input in oscillator I affects all eigenvalues while inputs to the higher-order oscillators do not move the λ_I eigenvalue. This is expected as oscillator I has maximum out-degree and correspondingly has the highest influence in the network.

As an input to oscillator I is required to affect the λ_I eigenvalue, we consider multiple oscillator input cases including oscillator I. A noticeable movement in the eigenvalues is observed with inputs on oscillators I and IV. For forcing inputs added to all oscillators, only λ_I eigenvalue is affected with no influence on the other system eigenvalues. To summarize the effectiveness of forcing input on controlling the system behavior, we track $\max(\Re(\lambda))$ for multiple oscillator input cases in figure 29 (b). We observe that for small σ, oscillator input combinations of I and IV outperform the other input combinations.

We consider the LQR controller such that system eigenvalues move towards the left side of the complex plane as much as possible. Thus, we select the input matrix $\boldsymbol{B} = [1\ 0\ 0\ 1]^T$ which adds forcing input to oscillators I and IV. Considering full-state feedback control with this choice of $\boldsymbol{B}$, we compute the control gain matrix $\boldsymbol{K}$ using LQR for a particular σ. Introducing control modifies the interaction between oscillators. We extract this modified network structure of interactions for $\sigma = 0.1$ from the controlled Laplacian $(-\boldsymbol{L}_\mathcal{G} - \boldsymbol{B}\boldsymbol{K})$ as shown in 29 (c). We can clearly notice that control added to oscillators I and IV result in self-loops which attenuate their fluctuations. Strong interactions are observed between oscillators I and IV. In addition, we also observe interactions corresponding

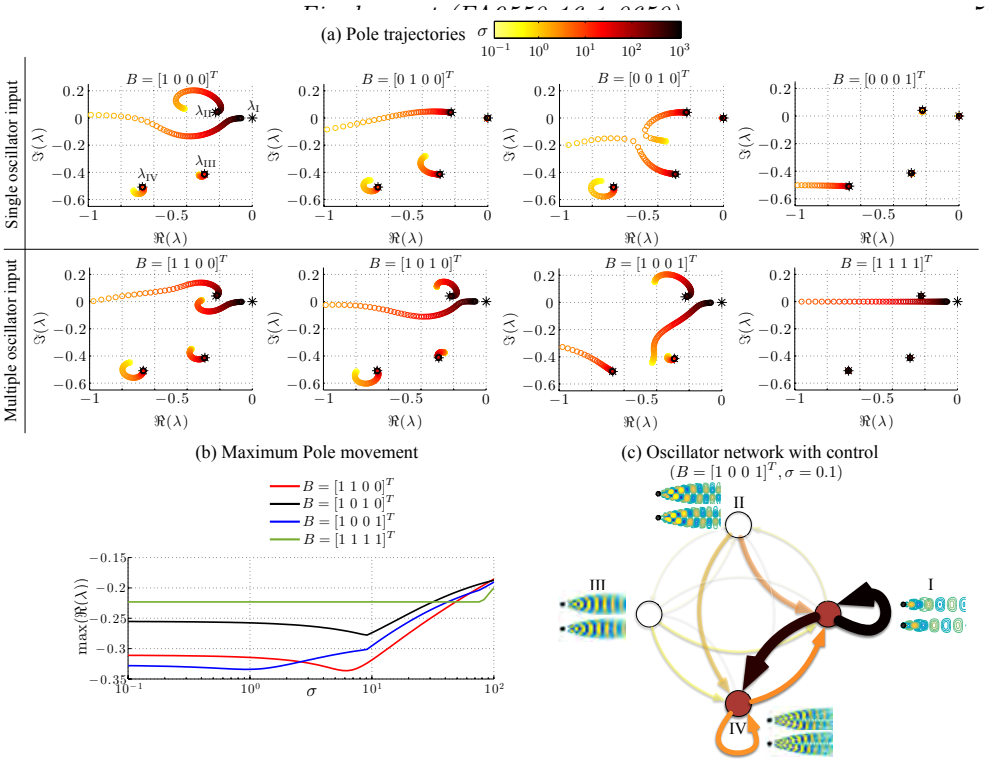


Figure 29: (a) Pole trajectories with application of different control inputs to the aggregate network model for a range of σ. Stars indicate $\lambda(-\boldsymbol{L}_{\mathcal{G}})$. (b) Movement of $\max(\Re(\lambda))$ for the multiple oscillator input cases. (c) Network structure with control. The oscillators to which control is added is indicated by red (filled) circles. The color and thickness of the network edges in the network represent the magnitude of elements of $(-\boldsymbol{L}_{\mathcal{G}} - \boldsymbol{BK})$.

to transfer of energy from oscillators II and III to oscillators I and IV. The incoming interactions to oscillators II and III are weakened considerably. Thus, energy transfers are concentrated to the oscillators where control is added which are ultimately attenuated due to the forcing resulting in self-loops. In the following discussion, we examine two control scenarios. First, we attenuate modal perturbations introduced in the flow. Then, we apply the LQR-based feedback control to suppress all modal amplitudes altogether.

9.1. Disturbance rejection

We first illustrate the control of modal disturbances introduced to the cylinder flow. This entails the control of fluctuations in the modal temporal coefficients (z'_m). To demonstrate the control, we consider random amplitude perturbations added to all oscillators about 2% of respective baseline fluctuations. The temporal coefficients of the oscillators for this perturbed case obtained from DNS are shown in figure 81 (left). The networked oscillator control system given by Eq. (9.2) describes control of temporal coefficients associated with the modes.

To implement control design in DNS, a body force corresponding to $-\boldsymbol{BK\zeta}$ is added to the momentum equation with the spatial modal information incorporated along with the temporal coefficients. The results with the application of control in DNS for $\sigma = 0.1$ are shown in figure 81 (right). We can drive the oscillators to the natural limit cycle

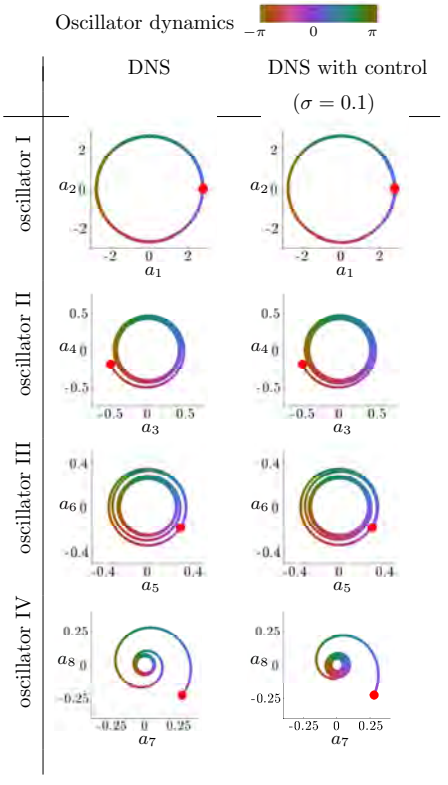


Figure 30: Oscillator dynamics without control and with control implemented in DNS for suppressing perturbations to limit cycle.

much faster with control. We notice that as control is introduced in oscillators I and IV, the effectiveness of control is more pronounced with these oscillators. This also follows from figure 29 where the correspondence between the Laplacian eigenvalues and oscillator inputs was discussed. Similar analysis can be performed for controlling a variety of modal perturbations introduced.

9.2. Total oscillation control

We then consider the control of total modal oscillations in the flow associated with wake unsteadiness. This requires the control of the temporal coefficients associated with the oscillators (z_m). Though the oscillator interactions characterized in this work are based on fluctuations with respect to the natural limit cycle (baseline) state, we consider these interactions to be characteristic of the modal oscillations in the flow. In fact, the energy transfer mechanism should be similar for z'_m and $z'_m + z^b_m$. We realize that by suppressing z_m, the flow is attracted towards the unstable steady state. Since we can assume that the above networked oscillator based LQR control is applicable near the baseline limit cycle, we expect that there is some region of validity of control to achieve drag reduction. Inhibiting nonlinear energy transfer should remove the energy input to sustain wake oscillations. Hence, we expect to use the same aggregate model extracted to suppress the limit cycle along with fluctuations of the modes to some degree.

DISTRIBUTION A: Distribution approved for public release.

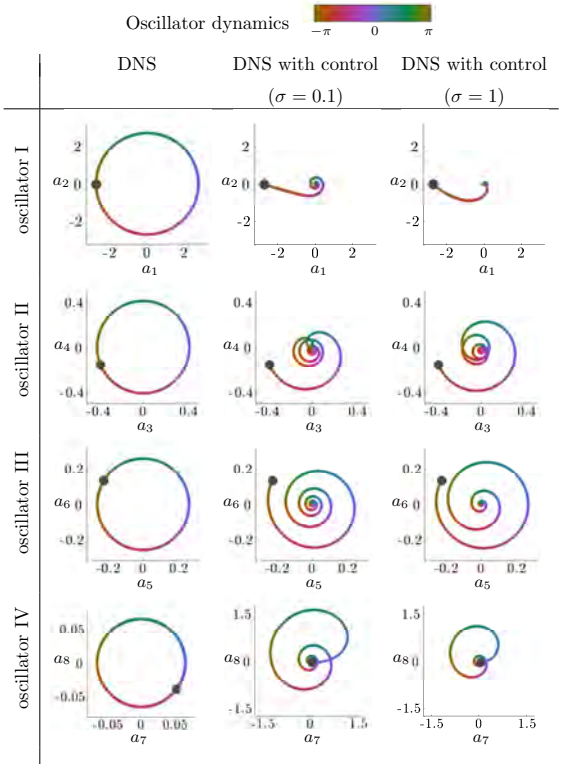


Figure 31: Oscillator dynamics without control and with control implemented in DNS for suppressing modal oscillations.

Our objective is to control the temporal coefficients associated with the modes, yielding a control system of the form,

$$\dot{z}_m = -\sum_{n=\mathrm{I}}^{M} [\boldsymbol{L}_{\mathcal{G}} - \boldsymbol{BK}]_{mn} z_n. \tag{9.4}$$

In DNS, this amounts to adding a body force corresponding to $-\boldsymbol{BKz}$. This control input steers the modal amplitudes corresponding to the baseline limit cycle to zero as shown in figure 82 (a). Here, we show the control performance for $\sigma = 0.1$ and 1. With control, the temporal coefficient associated with oscillator I first decays to zero followed by the higher-order oscillators. The energy lost by oscillator I is compensated with an initial increase in energy associated with oscillator IV. This is expected as oscillator IV is the most influenced node in the network. This highlights the energy transfers from lower-order to higher-order oscillators. As control input is added to oscillator IV, we suppress the corresponding modal oscillations. Once the modal amplitudes are forced to zero, the flow returns to the mean flow. This is expected as the unsteady flow field is decomposed into mean flow and modal components, so that forcing the modal components to zero reduces the flow to the mean. However, the mean flow is not an equilibrium and a shift in the mean flow is observed as time progresses. This mean flow deformation is attributed to the Reynolds stress generated by the modal fluctuations which modifies the base flow (Brunton & Noack 2015*a*). The change in the base flow leads to a corresponding decrease in modal energy until equilibrium is achieved. In the case of the cylinder flow problem,

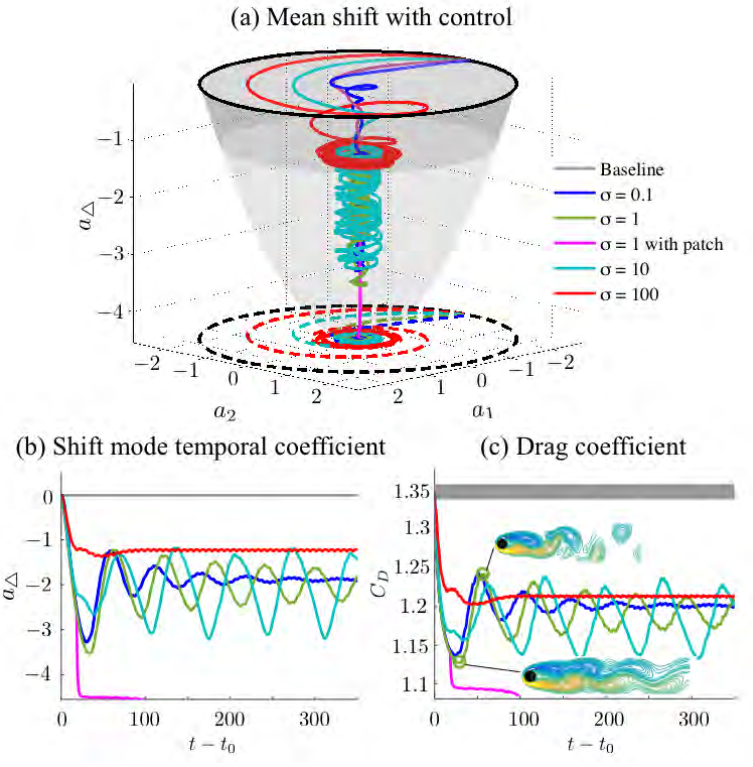


Figure 32: (a) Shift mode temporal dynamics on parabolic inertial manifold for a range of σ and (b) shift mode temporal dynamics in the time domain. (c) Drag force compared to the baseline.

the flow tends towards the unstable steady state.

The mean shift between the mean flow and the unstable steady state can be described by the shift mode. The temporal coefficient ($a_\triangle$) corresponding to the shift mode ($\boldsymbol{u}_\triangle$) can be obtained by projection as $a_\triangle = \langle \boldsymbol{u} - \bar{\boldsymbol{u}}, \boldsymbol{u}_\triangle \rangle$. We also perform control considering $\sigma = 10$ and 100. The variation of shift mode temporal coefficients with respect to temporal coefficients of the dominant POD modes for the range of σ considered is shown in figure 85 (a). For reference, the evolution of the flow from unstable steady state to the mean flow that follows a parabolic inertial manifold is shown in grey (Noack *et al.* 2003*a*). The darker grey region of the manifold indicates the region of effective control using the networked oscillator model. In this region, the nonlinear energy transfers are inhibited to reduce wake oscillations. Thus, in suppressing the modal oscillations, we achieve a mean shift in the flow resulting in drag reduction. For the case corresponding to $\sigma = 1$, we collect velocity snapshots in the time interval between $t = 19.3$ and 80 and perform POD analysis on the controlled data. Considering the first 10 additional POD modes for control, we construct a linearized model by curve-fitting the temporal coefficients. Designing an additional controller (patch controller) based on these coefficients, we can force the flow to the bottom of the manifold corresponding to the unstable steady state shown in magenta ($\sigma = 1$ with patch) in figure 85 (a).

The temporal dynamics associated with the shift mode is shown in figure 85 (b). It can be observed that initially $a_\triangle = 0$ in the limit cycle and as control is applied, the

DISTRIBUTION A: Distribution approved for public release.

shift mode temporal coefficient decreases and approaches the unstable steady state. As σ decreases, we achieve a steeper decrease in $a_\triangle$. Examining the long time history of the temporal variation of the shift mode, the choice of lower σ results in low frequency oscillations in $a_\triangle$. These low frequency oscillations are attributed to the exchange of energy between the low and moderate drag states (Munday & Taira 2013). For control effort corresponding to $\sigma = 100$, a steady state with application of control is achieved more rapidly.

The unsteady forces on the cylinder with application of control compared to the baseline drag variation is shown in figure 85 (c). The variation in the drag coefficient is similar to the shift mode variation as discussed previously. Almost a 12% reduction in drag is achieved with the application of control. With the addition of patch controller, minimum drag state is achieved. The networked oscillator control framework considered here is based on interactive dynamics of the baseline POD modes and does not include any other modes. We however note that the shift mode itself can be incorporated into the controller (Noack *et al.* 2011; Luchtenburg *et al.* 2009).

The unsteady forces on the cylinder can be reduced by suppressing modal oscillations in the flow and inhibiting energy transfers therein. Using a networked oscillator model in conjunction with optimal control, we can control energy transfer dynamics effectively for unsteady wake flows. In the next chapter, we discuss a model-free feedback control strategy for unsteady separated flows.

DISTRIBUTION A: Distribution approved for public release.

10. Markov Transition Network Control

Modal interpretation of physics yields low-order models for control of fluid flows, as discussed in the previous chapter. However, stochastic and broadband dynamics limit capabilities of these models. To describe such dynamics, cluster-based reduced-order modeling (CROM) was proposed in the work by Kaiser *et al.* (2014*a*). Partitioning high-dimensional data into groups sharing similar attributes, also called *clusters*, the system dynamics are modeled as a linear, probabilistic Markov chains. This framework yields a low-dimensional, statistical closure model for turbulence with intrinsic robustness and systematic incorporation of nonlinear control mechanisms (Kaiser *et al.* 2017*b*).

The objective of this work is three-fold; (i) partition the baseline flow trajectories (sensor measurements) into discrete clusters and examine the linear Markov transition dynamics, (ii) design an optimal feedback control law in a model-free manner using the discretized clusters, and finally (iii) transform the baseline Markov transition dynamics to achieve desired performed objectives with control.The overall approach is outlined in Figure 33. In §**??**, we discuss the details of the computational setup for baseline simulations along with numerical validation. The actuator setup for performing flow active control studies is elaborated in §10.2. To design feedback control laws for separated flows, collected baseline feature-space trajectories are discretized into clusters as discussed in §10.3. Each coarse-grained phase of the flow (cluster) has an associated wall-normal blowing/suction jet velocity input for actuation. As the flow transitions through the clusters, the influence of the associated actuation command dominates the control input. Path of control trajectories determines the feedback to the flow enabling the controller to adapt in time. Our focus is to reduce the aerodynamic power in post-stalled flows with minimum actuation power from data-based principles which has implications in improving flight endurance (Anderson 1999). The details of optimization of cluster-based feedback control laws are outlined in §10.4. We demonstrate the effectiveness of the approach for two–dimensional (2D) and three–dimensional (3D) separated flow over an airfoil in §10.5 and §10.8, respectively.

10.1. Cluster-based control framework

We perform 2D and 3D large eddy simulations (LES) of separated flows over a NACA 0012 airfoil at an angle of attack $\alpha = 9°$, with Reynolds number of $Re = U_\infty L_c/\nu = 23,000$ and Mach number of $M_\infty = U_\infty/a_\infty = 0.3$. Here, U_∞ is the free stream velocity, L_c is the chord length, ν is the kinematic viscosity, and a_∞ is the free stream speed of sound. Spanwise periodicity is enforced in the 3D simulations. LES is performed using a compressible flow solver CharLES (Brès *et al.* 2017). The solver uses a second-order accurate finite-volume scheme and a third-order Runge-Kutta method for time integration.

The computational domain is chosen to be $x/L_c \in [-19, 26]$, $y/L_c \in [-20, 20]$, $z/L_c \in [-0.1, 0.1]$, following the work by Yeh *et al.* (2017), where x, y and z are streamwise, normal and spanwise coordinate directions, respectively. For performing LES, Vremen's subgrid-scale model (Vreman 2004) is utilized. We apply Dirichlet boundary conditions corresponding to $[\rho, u_x, u_y, u_z, p] = [\rho_\infty, U_\infty, 0, 0, p_\infty]$ at inflow and far-field with a sponge zone specified near the outlet over $x/L_c \in [16, 26]$ to avoid numerical reflections. Here, u_x, u_y, and u_z stand for streamwise, normal and spanwise velocity, respectively. A no-slip adiabatic boundary condition is prescribed over the airfoil. A structured mesh consisting of 32 million volume elements is utilized for the 3D LES simulations. The 2D LES simulations consider the same $x - y$ plane corresponding to the 3D mesh consisting 0.32 million volume elements.

DISTRIBUTION A: Distribution approved for public release.

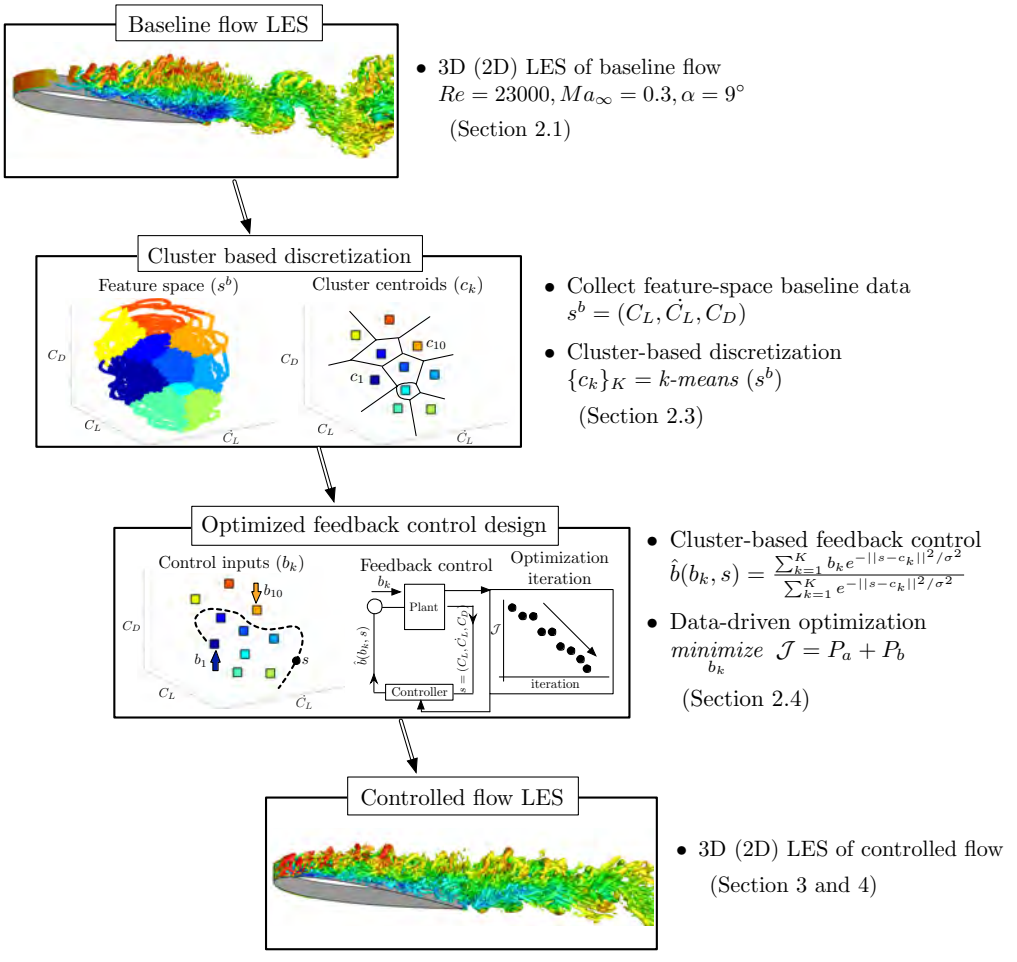


Figure 33: Overview of cluster-based control framework.

The computational domain ($x - y$ plane) and mesh is shown in Figure 34(a) (left). The streamlines for the time-averaged 3D baseline separated flow is shown in Figure 34(a) (right). We can see the presence of a recirculation bubble in this separated flow. To characterize the extent of the separation region, a contour line of time- and spanwise-averaged streamwise velocity, $\bar{u}_x/U_\infty = 0$, is shown by the blue dashed line, which extends over the length of the airfoil. Here, $\bar{q}$ indicates time-average (mean) of flow variable q.

The instantaneous flow field and turbulent kinetic energy (TKE) for 3D baseline flow are shown in Figure 34(b). TKE is defined as $\overline{u_x'^2 + u_y'^2 + u_z'^2}/U_\infty^2$, where $u_\bullet' = u_\bullet - \bar{u}_\bullet$ and '$\bullet$' assumes x, y, or z. The vortical structures in the flow field are highlighted by a level set of the Q-criterion (Hunt *et al.* 1998), which is colored by streamwise velocity. It can be seen that laminar separation at the leading edge forms a shear layer that rolls up and evolves into spanwise vortical structures. TKE increases in this region reaching a maximum value at $x/L_c \approx 0.55$. A subsequent laminar-turbulent transition is observed.

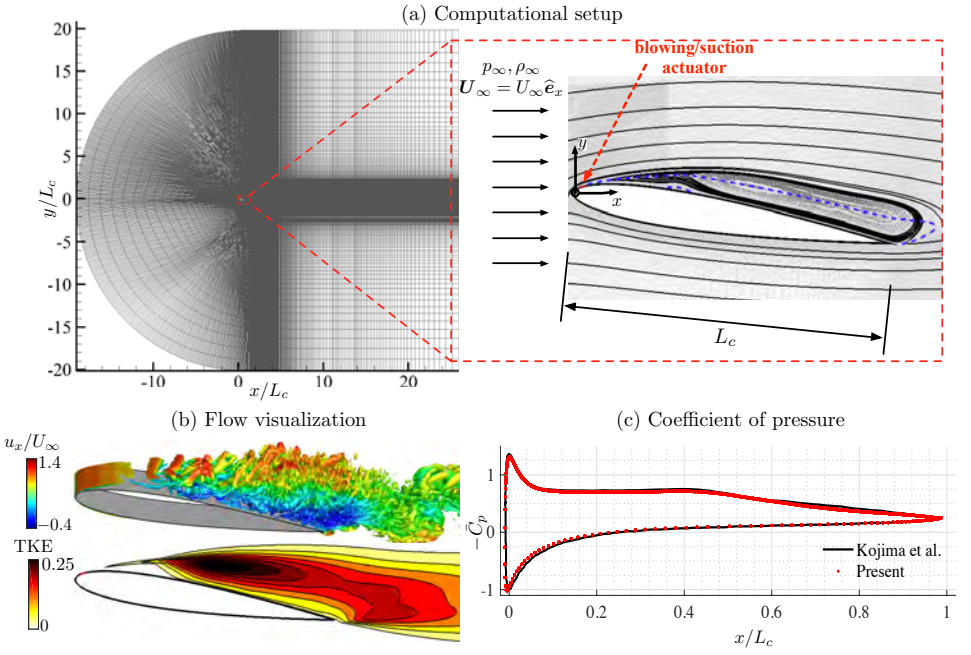


Figure 34: (a) Computational domain, $x - y$ plane (left) and near field of a NACA 0012 airfoil at an angle of attack of $\alpha = 9°$ (right) indicating the streamlines for 3D spanwise-periodic baseline flow. The actuator surface is indicated in red. The blue dashed line indicates contour line corresponding to $\bar{u}_x/U_\infty = 0$. (b) Instantaneous flow field (highlighted by Q-criterion) colored by streamwise velocity and turbulent kinetic energy (TKE) and (c) time-averaged coefficient of pressure distribution on suction and pressure side of the airfoil for 3D baseline flow.

From the simulations, the drag coefficient C_D, lift coefficient C_L, and pressure coefficient C_p are obtained, which are defined as

$$C_D = \frac{F_x}{\frac{1}{2}\rho_\infty U_\infty^2 A}, \quad C_L = \frac{F_y}{\frac{1}{2}\rho_\infty U_\infty^2 A}, \quad C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty U_\infty^2}, \tag{10.1}$$

where F_x and F_y are the drag and lift forces on the airfoil, $A = L_c w$ is the projected area, w is the width of the airfoil, and p is the pressure at the airfoil surface. For validation of the numerical setup, we compare the time-averaged coefficient of pressure distribution ($\bar{C}_p$) for the 3D baseline separated flow with Kojima *et al.* (2013) in Fig. 34(c), where agreement can be seen. In particular, a strong negative pressure peak on the suction side near the leading edge is visible followed by a plateau in pressure distribution. This pressure plateau indicates the laminar-turbulent transition over the separation bubble (Benton & Visbal 2018). For 2D baseline simulation, mean drag $\bar{C}_D = 0.127$ and mean lift $\bar{C}_L = 0.818$. For 3D baseline simulation, $\bar{C}_D = 0.114$ and $\bar{C}_L = 0.557$. For performing flow control simulations, an actuator setup is required as discussed below.

10.2. Actuator setup

A blowing/suction actuator is placed upstream of the separation point of the baseline flow for flow control. The separation points for 2D and 3D baseline flow are located at $x_s/L_c = 0.032$ and 0.037, respectively. The center of the actuator is located at $x_a/L_c =$

0.03 in the streamwise direction on the suction side of the airfoil. The actuator width in the chordwise direction is $l_x/L_c = 0.02$. A wall-normal velocity component (u_{jet}) with a parabolic shape is prescribed at the actuator location leading to blowing/suction. For 3D flow control, two equally spaced slots are incorporated, similar to the work by Munday & Taira (2014). A hyperbolic tangent function is used for the spanwise jet velocity profile to smoothen the velocity discontinuity at the edge of the slots. As the spanwise width of the airfoil considered in this work is $w = 0.2$, the spanwise wavenumbers k_z^+ are integer multiples of 10π. The chosen slot configuration corresponds to $k_z^+ = 20\pi$. The actuator location is shown in red in Figure 34(a) (right). The wall-normal velocity is prescribed as

$$u_{jet} = \hat{b}\left[1 - \frac{(x - x_a)^2}{l_x^2}\right]\phi\left(\frac{z - z_a}{L_c}, \frac{l_z}{L_c}\right), \qquad \frac{|x - x_a|}{L_c} \leqslant \frac{l_x}{2L_c}, \tag{10.2}$$

where, $z_a/L_c = \pm 0.05$ are the center of the slots in the spanwise direction and $l_z/L_c = 0.025$ is the spanwise width of each slot. Here, $\hat{b}$ is the forcing amplitude, and ϕ is the hyperbolic tangent velocity profile. To determine the forcing amplitude $\hat{b}$, we follow a cluster-based control strategy discussed below.

10.3. Cluster-based discretization

Characterization of a feedback control law requires knowledge of the flow state at each instant in time. In a computational setting, a fluid flow state typically comprises millions of degrees of freedom equal to the number of grid points times the number of variables considered. Parametrization of an optimal controller can thus get prohibitively expensive, especially in a full-state model-free environment. Instead of the high-dimensional full state information, we consider a feature space of the baseline flow consisting of reduced observables of the fluid flow state.

The amplitude and phase dynamics in bluff body flows are captured well by the lift coefficient C_L and its time derivative $\dot{C}_L$, while the drag coefficient C_D captures the mean shift effects (Noack *et al.* 2003*a*; Taira & Nakao 2018). These reduced number of observables define a three-dimensional feature space denoted by $s = (C_L, \dot{C}_L, C_D)$. The subset (C_L, C_D) is also part of the control optimization procedure, as will be discussed in §10.4. This choice of observables is robust with respect to slow changes in the flow in contrast to a reduced-order description in terms of few coherent structures deduced by a modal decomposition, which may significantly change. A similar feature space was also utilized in the work by Loiseau *et al.* (2017) to deduce sparse, reduced-order, nonlinear models for a two-dimensional flow over a circular cylinder at $Re = 100$. Generally, as the complexity of the flow increases with increasing Reynolds number, the attractor becomes more and more high-dimensional and modeling the interaction terms becomes infeasible. In this work, instead of modeling the interaction terms, we pursue a partitioning of the feature space to identify coarse-grained phases of the flow, which define a model-free parametrization of the feedback control law, suitable for complex flows. This parametrization can be constrained to the actual attractor to avoid the curse of dimensionality, which motivates a data-driven clustering analysis.

The cluster analysis partitions a set of objects or observations with common characteristics into few distinct groups or *clusters*. There are many available clustering techniques including spectral clustering, hierarchical clustering, and subspace clustering, each with their individual pros and cons (Rokach & Maimon 2005). Cluster analysis is commonly used to find a natural grouping among data, in contrast, here we employ clustering to discretize or *coarse-grain* the control law defined in the feature space. This is critical to group flow states with similar aerodynamic characteristics, e.g., a high drag states and

DISTRIBUTION A: Distribution approved for public release.

low drag states of the flow. Centroid-based clustering algorithms aims to classify objects into K representative clusters $\{\mathcal{C}_k\}_{k=1}^K$ by minimizing the within-cluster variance while maximizing the between-cluster variance. Each set of observations belonging to a cluster $\mathcal{C}_k$ is represented by its corresponding cluster centroid c_k, which is computed as the mean over all observations belonging to this cluster. These cluster centroids represent the mean behavior of the clusters.

The most popular centroid-based clustering technique is the *k-means* algorithm (Lloyd 1982) which is an unsupervised classification algorithm where observations are partitioned into K clusters. Each observation of the baseline flow, measured by $s^b = (C_L, \dot{C_L}, C_D)$, is associated with a characteristic baseline velocity field $\boldsymbol{u}^b$. Here, the superscript b stands for baseline flow. Given an ensemble of observations in terms of a trajectory $s^b(t)$, the cluster-based centroids $\{c_k\}_{k=1}^K$ are obtained by solving an optimization problem minimizing the total within-cluster variance

$$\underset{\mathcal{C}}{\texttt{arg min}}\ J_w(c_1, ..., c_K) = \underset{\mathcal{C}}{\texttt{arg min}} \sum_{k=1}^{K} \sum_{s^b \in \mathcal{C}_k} ||s^b - c_k||^2. \tag{10.3}$$

This yields a set of K clusters, $\mathcal{C} = \{\mathcal{C}_1, ..., \mathcal{C}_K\}$, each with a centroidal representative state c_k. Here, the centroid $c_k = \frac{1}{N_k} \sum_{s^b \in \mathcal{C}_k} s^b$, where N_k is the number of measurements belonging to cluster k. In the work of Kaiser *et al.* (2014*a*), this has led to the cluster-based reduced-order model (CROM) by modeling the transitions as a Markov process (Norris 1998). There exist an important connection to the linear Liouville equation and the asscoiated Perron-Frobenius operator, which propagate an ensemble of trajectories in phase space. In particular, CROM represents a discretization of the Perron-Frobenius operator and yields a coarse-grained linear model in the probability space, represented by the Markov transition matrix. Here, we examine the Markov transitions in the baseline and controlled flows as a post-analysis tool to understand how the optimized control law modifies transition probabilities, i.e. redirects trajectories in phase space.

A tradeoff between complexity of the cluster-based representation and data compression determines an optimal choice of the number of clusters K (Chiang & Mirkin 2010). It should be noted that increasing K is associated with an increase in the computational cost of the overall approach. The choice of number of clusters can be justified using an elbow method or F-test (Lomax & Hahs-Vaughn 2013). The inter-cluster variance is given as

$$J_i = \frac{1}{N} \sum_{k=1}^{K} |N_k| ||c_k - \bar{c}||^2, \tag{10.4}$$

where $\bar{c} = \frac{1}{N} \sum_{k=1}^{K} |N_k| c_k$ is the centroid of the entire data set (Goutte *et al.* 1999) and N is the total number of measurements in this data set. The total variance is given by $J = J_w/N + J_i$. The F-test is the ratio of inter-cluster variance J_i to the total variance J, which is typically maximized. We chose $J_i/J \geqslant 0.9$ for our analysis which leads to $K = 10$. Thus, each dataset of trajectories for the 2D and 3D baseline flows are partitioned into $K = 10$ clusters. With this clustered feature space representation, we discuss our flow control design below.

10.4. Optimized feedback control design

Traditional approaches for flow control involve modeling and manipulating the interaction between few relevant coherent structures. Models are often constructed by decomposing the flow into a truncated series of coherent structures or spatial modes and modeling their dynamic interactions. However, these modes are generally limited to the

DISTRIBUTION A: Distribution approved for public release.

data they have been extracted from. By applying actuation, the behavior of the flow often changes and thus leads to a change in the modal representation, which is not captured by the identified model. Hence, modeling errors and changes in the flow with control due to nonlinearity severely hamper the performance of the control laws obtained.

As opposed to these model-based approaches, we seek a feedback control law determined in a model-free manner. In particular, the segmentation of the feature space using clustering yields in its simplest form a piecewise-constant control law, which is parametrized by the actuation value associated with each cluster. This is a distance-based, globally defined control law, i.e. the specific actuation value is associated with the nearest centroid, and does not neglect any dynamics. With feedback of the current state in the feature space, the central idea of the flow control strategy is to re-route the trajectories so as to maximize the performance objectives. Using an iterative optimization procedure, we can iteratively optimize the control law in an automated fashion. We now discuss the procedure for optimized feedback control design.

Each cluster $\mathcal{C}_k$ with its associated centroid c_k is assigned a constant control amplitude b_k. This amounts (depending on the sign) to a blowing/suction jet velocity for each cluster. Cluster control amplitudes are then smoothed using normalized radial basis kernel smoothing (Wand & Jones 1994). The current state of the observable $s(t)$ at time t is sensed for performing feedback control. The distance of the measurement $s(t)$ to each cluster centroid c_k in feature space, $d_k = ||s(t) - c_k||$, is then computed. The smaller the distance, the more the influence of the cluster centroid amplitude. With the cluster centroids, the current sensor measurement and the cluster control amplitudes, the forcing amplitude $\hat{b}$ as required in Eq. (10.2) can be defined as

$$\hat{b}(b_k, s(t)) = \beta \frac{\sum_{k=1}^{K} b_k e^{-||s(t)-c_k||^2/\sigma^2}}{\sum_{k=1}^{K} e^{-||s(t)-c_k||^2/\sigma^2}}, \tag{10.5}$$

where $\sigma^2 = J_i$, provided by Eq. (10.4). Here, β is the feedback gain which is set to unity, unless otherwise noted. We consider higher values of β for 3D flow control studies in §10.8. The flow control is implemented as a proportional feedback controller depending on the current state in the feature space.

Iterative optimization of the cluster control amplitudes $\{b_k\}_{k=1}^{K}$ is then performed to minimize a cost functional consisting of both state and control variables. For optimized control, we need to minimize both the aerodynamic power output P_a and the actuation power input P_b leading to an objective function $\mathcal{J} = P_a + P_b$. The aerodynamic power is the power required by the system to overcome drag. We can compute the aerodynamic power (also thrust power) as,

$$P_a = \frac{F_x U_\infty}{\frac{1}{2}\rho U_\infty^3 A} = \frac{\frac{1}{2}C_D \rho U_\infty^3 A}{\frac{1}{2}\rho U_\infty^3 A} = \frac{C_D}{U_\infty^3}\left(\frac{F_y}{\frac{1}{2}\rho C_L A}\right)^{3/2} = \gamma \frac{C_D}{C_L^{3/2}}, \tag{10.6}$$

where $\gamma = \left(\frac{F_y}{\frac{1}{2}\rho U_\infty^2 A}\right)^{3/2}$. At cruise condition, lift is equal to the weight of the flying vehicle W. Here, we set $\gamma = \bar{C}_L^{3/2}$, where $\bar{C}_L$ is the mean baseline lift. This choice also results in an appropriate characterization of the (unforced) baseline drag power. The unsteady actuation power is related to the momentum injected to the fluid as

$$P_b = \frac{\rho_\infty (u_{jet})^3 A_s}{\frac{1}{2}\rho_\infty U_\infty^3 A} = \frac{2}{T U_\infty^3 A}\int_0^T \int_{-w/2}^{w/2} \int_{x_a-\sigma/2}^{x_a+\sigma/2} |u_{jet}|^3 \mathrm{d}x\mathrm{d}z\mathrm{d}t, \tag{10.7}$$

DISTRIBUTION A: Distribution approved for public release.

where A_s is the area of actuation. The time-averaged momentum coefficient corresponding this actuation power is

$$\bar{C}_\mu = \frac{\rho_\infty |u_{jet}|^2 A_s}{\frac{1}{2}\rho_\infty U_\infty^2 A} = \frac{2}{TU_\infty^2 A}\int_0^T \int_{-w/2}^{w/2}\int_{x_a-\sigma/2}^{x_a+\sigma/2} |u_{jet}|^2 \mathrm{d}x\mathrm{d}z\mathrm{d}t.s \tag{10.8}$$

To optimize the cluster control amplitudes $\{b_k\}_{k=1}^K$, we utilize the simplex search algorithm (Nelder & Mead 1965). Simplex search is a gradient-free multidimensional unconstrained optimization technique. Variants of simplex search to add constraints (Luersen *et al.* 2004) and parallel implementations for alleviating computational cost (Lee & Wiswall 2007) are utilized for 3D flow control examinations in this work. The simplex method iteratively optimizes the cluster-based feedback control laws. To start the iterative optimization, an initial simplex is formed. As our goal is to determine K optimized cluster control amplitudes, $\{b_k\}_{k=1}^K$, we define an initial simplex of $N_b = K+1$ vertices. Each vertex of the simplex is evaluated with a controlled LES, each with a unique set of K cluster control amplitudes to define $\hat{b}$. We incorporate a Latin-hypercube sampling of the parametric space of $N_b \times K$ control amplitudes for the initial simplex, which leads to a near-random sample of parameter values from multidimensional distribution (McKay *et al.* 2000).

We simulate the N_b initial flow control cases over a finite time horizon T. This time horizon is typically chosen as multiples of characteristic shedding frequency $St = fL_c \sin(\alpha)/U_\infty$ of the flow. Once the initial N_b control cases are simulated and objective function $\mathcal{J}$ for each case are evaluated, the simplex search algorithm performs operations of reflection, expansion, contraction and shrinkage on the cluster control amplitudes to minimize the objective function $\mathcal{J}$. These operations quickly span the search space of cluster amplitudes to maximize performance objectives. Ultimately, a choice of cluster control amplitudes (b_k) results in a minimal $\mathcal{J}$, maximizing aerodynamic performance, which we deem as optimized.

As we have K cluster control amplitudes to optimize, visualization of the control landscape over all the clusters can be very insightful but also challenging. Multidimensional scaling (MDS)(Young & Householder 1938) aims to visualize the organization of high-dimensional objects by finding a low-dimensional subspace, which optimally preserves the distances between objects in the high-dimensional space. Here, we employ MDS to visualize the similarity between control laws by finding a low-dimensional embedding maintaining pairwise distances between them (Kaiser *et al.* 2017*b*,*a*). In addition to measuring the similarity or dissimilarity between the evaluated control laws, the visualization helps in tracking of the search directions tending towards the optimum of our objective. The pairwise distances in MDS are defined as

$$D_{ij} = \sqrt{\frac{1}{2}\sum_{t=1}^{T}[\hat{b}(b_k^i, s^i(t)) - \hat{b}(b_k^j, s^i(t))]^2 + \frac{1}{2}\sum_{t=1}^{T}[\hat{b}(b_k^i, s^j(t)) - \hat{b}(b_k^j, s^j(t))]^2}, \tag{10.9}$$

where $\hat{b}$ is defined by Eq. (10.5). The superscript i and j indicate the iteration number corresponding to the control cases in the optimization procedure. MDS then aims to find a set of points $\{\gamma^i\}_{i=1}^{N_c}$, where N_c is the total number of evaluated control laws, in a low-dimensional subspace such that $||\gamma^i - \gamma^j|| \approx D_{ij}$ is approximated in a least-squares sense. Here, we aim to find a two-dimensional subspace, $\gamma^i = \{\gamma_1, \gamma_2\}$, for visualization purposes. The proximity map can help evaluate the relative performance of each control case with iterative optimization based on the variation in γ_1 and γ_2 variables, which represent the two directions in the identified subspace. Moreover, these proximity maps

DISTRIBUTION A: Distribution approved for public release.

can aid to accelerate the optimization process by estimating the performance of untested control laws as proposed in Kaiser *et al.* (2017*a*).

Using the above approach, an optimized cluster-based feedback control law can be determined. We demonstrate the optimization of cluster control amplitudes for the 2D separated flow in the following section. We then extend the approach to find cluster-based control laws for 3D separated flows, for which the associated computational cost and complexity of flow control increases significantly.

10.5. Separation control of the flow around a two-dimensional airfoil

In this section, we demonstrate the cluster-based feedback control optimization for a two-dimensional flow around an airfoil to maximize performance. In particular, we first present the clustering results based on data from the baseline configuration in Sec. 10.6, that are employed to partition the feature space and thus the control law. Then, we demonstrate how the coarse-grained control law is optimized in Sec. 10.7. The iterative optimization procedure is further analyzed using using proximity maps; and the change of dynamics due to the effect of control is examined using Markov transition models.

10.6. Baseline feature space clustering

A cluster-based analysis of the 2D baseline separated flow over the airfoil is performed as described in §10.3. In post-stalled configurations, a strong adverse pressure gradient due to separation causes a large increase in the associated pressure drag, thereby increasing the pressure losses as well as the size of the wake. Using a cluster-based analysis, we aim to identify characteristic phases of the flow is associated with these losses. For the analysis, a time series of baseline trajectories $s^b(t)$ is collected. The constant time step $\triangle t$ is chosen so as to resolve the baseline vortex shedding frequency, $St = fL_c \sin(\alpha)/U_\infty = 0.081$ with at least 500 snapshots. Also, the collected data spans a total convective time of $tL_c/U_\infty = 110$. The data of the observable $s^b(t)$ is then partitioned into $K = 10$ clusters using the *k-means* algorithm as shown in Figure 35(a). The cluster discretization of the feature space separates characteristic phase regimes of the flow providing a representative phase map. Generally speaking, high lift states of the flow are accompanied by high drag for massively separated flows. The stochasticity in the dynamics can be observed from the trajectories in the feature space.

By analyzing the transitions between clusters, transition dynamics between characteristic phases of the flow can be deduced. These transition dynamics can be described by Markov chains and underlie the cluster-based reduced-order modeling (CROM) framework (Kaiser *et al.* 2014*a*). The Markov model associated with CROM provides a stochastic framework for transition dynamics and its analysis can be used to identify different, weakly connected dynamic regimes. Here, the model is used to extract corresponding transition dynamics associated with post-stalled flows. The resulting CROM model is a finite-rank approximation of the Perron-Frobenius operator associated with the Liouville equation, which describes the evolution of probability densities in state space (Lasota & Mackey 2013; Bollt & Santitissadeekorn 2013). The cluster-based model is represented by a probabilistic cluster transition matrix P. The elements of this matrix describe the probability of transition from cluster c_k to c_j in one forward time step $\triangle t$ and are given as

$$P_{jk} = \frac{N_{jk}}{N_k}, \tag{10.10}$$

where N_{jk} is the number of transitions from cluster c_k to c_j. The cluster transition matrix has a column sum of $\sum_j P_{jk} = 1$. Such a normalization is necessary so that total

DISTRIBUTION A: Distribution approved for public release.

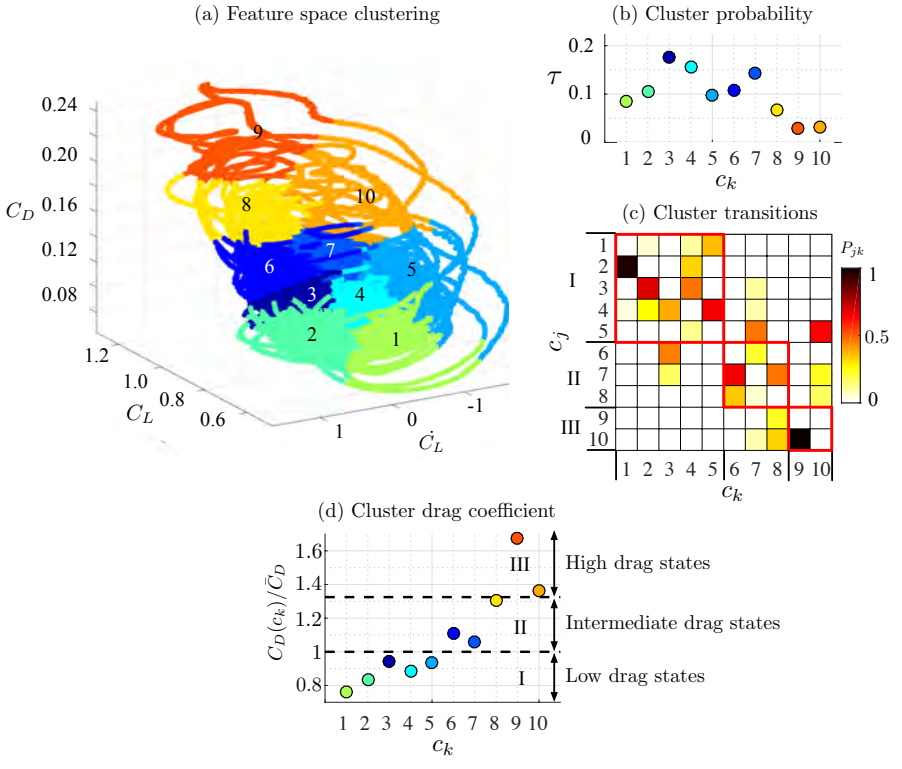


Figure 35: (a) Feature space clustering of the 2D baseline flow. (b) Cluster residence probabilities and (c) cluster transition probabilities. The red boxes indicate cluster subsets. (d) Normalized drag corresponding to cluster centroids.

probability for all transitions is unity. The diagonal elements of this matrix represent the residence likelihood within the same cluster and the off-diagonal entries represent the inter-cluster transitions.

We further define the probability of the flow to reside in one of the clusters, which also amounts to the average time the flow remains in any of the clusters, as $\tau_k = |N_k|/N$, also referred to as cluster probability.The cluster probabilities and the cluster transitions are shown in Figure 35(b) and (c), respectively. The cluster probability is highest for cluster 3 followed by clusters 4 and 7. The higher the cluster probability, the closer the cluster centroid is to the centroid of the entire trajectory $\bar{c}$. The flow states in cluster 1 and 9 always transition to cluster 2 and 10 respectively, resulting in a high probability of transition.

To further simplify these transitions, we can identify subsets, i.e. groups of clusters, in which transitions occur more frequently than between these. To identify the partition of these decomposable sub-Markov chains (Kontovasilis & Mitrou 1995), we use the directed modularity maximization algorithm (Leicht & Newman 2008). The algorithm detects modular subsets where the transitions between clusters within a subset are dense compared to the transition between clusters from different subsets. Here, it identifies three subsets: subset I:$\{1, 2, 3, 4, 5\}$, subset II:$\{6, 7, 8\}$, and subset III:$\{9, 10\}$. These subsets are displayed as red boxes in Figure 35(b). High probability inter-subset transitions are observed between cluster $3 \rightarrow 6$, $7 \rightarrow 5$ and $10 \rightarrow 5$. These transitions also mark the major paths of transition between subsets. Most of the remaining transitions are within

DISTRIBUTION A: Distribution approved for public release.

subsets. We further analyze the drag contribution of each cluster, which is obtained from the cluster centroids. We normalize the drag coefficient associated with each cluster centroid with the mean drag $\bar{C}_D$ as shown in Figure 35(d). The dashed lines separate the subsets. Interestingly, subset I corresponds to low drag states of the flow $C_D/\bar{C}_D < 1$, subset II corresponds to intermediate drag states $1 \leqslant C_D/\bar{C}_D \leqslant 1.325$ and subset III corresponds to the high drag states $C_D/\bar{C}_D > 1.325$. Thus, different levels of drag are all dynamically separated, which can significantly simplify the control design: It suggests steering the flow to the subset associated with the lowest drag and then keeping it in this dynamic regime may be sufficient.

An alternate view point of the Markov chain is a random walk on a directed graph with nodes being clusters and edges being the transition dynamics between them (Newman 2010). The edge weights correspond to the number of transitions between the respective clusters N_{jk} normalized by the maximum number of cluster transitions observed in the entire trajectory, $\texttt{max}(N_{jk})$. This normalization assigns higher weights to cluster transitions with more frequency. We ignore self-loops, i.e. transitions within the cluster and edges weights less that a threshold of 0.1. The directed graph representation of the cluster transitions is shown in Figure 36. The edge weights are indicated by the thickness of the lines for this graph representation. The segregation into subsets follows our earlier discussion. Here, we can clearly identify the paths of cluster transitions and visualize the phase evolution of the flow. One characteristic feature that stands out from this graph representation is the role of cluster 8. The flow state reaches cluster 8 from cluster 6. Only through this cluster 8, the flow can transition to the high drag state of cluster 9. The flow can also transition to the other high drag state of cluster 10 through cluster 8. We call cluster 8 the switching cluster. The significance of the cluster transition pathway $6 \to 8 \to 9$ is an important consideration for flow control. The elimination of this cluster transition pathway is key in enhancing aerodynamic performance, particularly for drag reduction.

The above discussion is limited to cluster transitions in the feature space. However, understanding the flow behavior in each cluster is critical in understanding changes in flow physics leading to transitions from low drag to high drag states and vice versa. Each observable state $s(t)$ has an associated flow field. We can average the flow snapshots within each cluster to obtain cluster representative flow fields. The cluster-averaged pressure and the instantaneous vorticity fields are shown in Figure 36. The flow fields associated with clusters 3 and 4 are similar to the time-averaged mean flow. This is expected due their close proximity to the centroid of the entire trajectory. The cluster flow fields in subset I give us a better understanding of the flow physics at the low-drag states of the flow. The different phases of the flow are clearly visible in the cluster transitions $3 \to 4 \to 5 \to 1 \to 2$ on observing the associated vortex dynamics. We see the initiation of shear-layer roll up in cluster 3 and the presence of leading edge vortices in cluster 4. These vortices start shedding mid-chord in cluster 5. In cluster 5, the maximum number of compactly arranged rolled-up structures can be observed above the airfoil surface. In cluster 1, we see the trailing edge roll-up followed by von Kármán vortex shedding in the wake in cluster 2. In these low drag states, the low pressure core above the suction side of the airfoil is closer to the leading edge, especially in clusters 1 and 2.

The cluster transition from 3 to 6 is characterized by strengthening of the low pressure core near the mid-chord of the airfoil. This results from elongating the vortex sheet on the airfoil surface. Following the graph representation, the flow at cluster 6 could transition either to cluster 7 or 8. Cluster 7 is characterized by the shear layer roll-up and intermittent shedding in the wake. The rolled-up vortices shed near the mid-chord

DISTRIBUTION A: Distribution approved for public release.

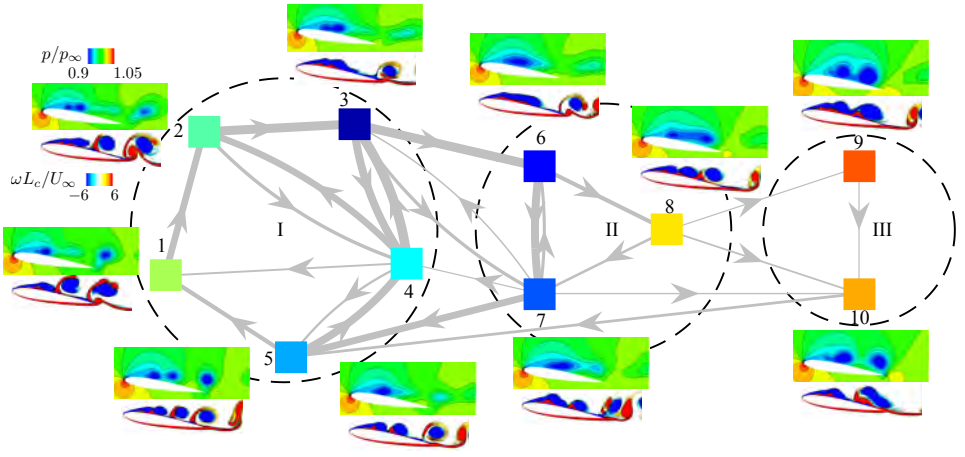


Figure 36: Graph of Markov chain highlighting transitions between clusters. 2D cluster-averaged pressure flow fields (p/p_∞) and instantaneous vorticity contours ($\omega L_c/U_\infty$) corresponding to each cluster are shown. The dashed circles indicate the cluster subsets.

of the airfoil resulting in a flow transition $7 \rightarrow 5$. The transition to cluster 8 results when the rolled-up vortices grow in size elongating the pressure core. As opposed to cluster 5, here the vortices do not detach from the airfoil surface but are arranged compactly on the surface. The flow from cluster 8 transitions to cluster 9, accompanied by a low pressure core spanning the entire airfoil with a large roll-up near the trailing edge of the airfoil. This results in a fully separated flow and correspondingly high drag. Following this state, the trailing edge vortex sheet rolls up and the vortices detach from the airfoil at the trailing edge leading to a subsequent transition to cluster 10 and then to 5. In cluster 10, the flow briefly reattaches and then separates near the trailing edge. Also, cluster 7 is characterized by high frequency shedding in the wake as opposed to the low frequency shedding in cluster 10.

The significance of the clusters can also be examined by looking at the cluster-averaged pressure distribution over the airfoil. We show the pressure distribution over the airfoil surface and streamwise velocity profile for one representative cluster in each cluster subset in Figure 37(a) and (b), respectively. On the suction side of the airfoil, we can see a favorable pressure gradient near the mid-chord for cluster 1 (low drag cluster) with a shorter flat pressure region indicating the shorter region of separation. This is also shown by the dashed lines corresponding to zero streamwise velocity contour, $u/U_\infty = 0$. Clusters 8 and 9 have larger regions of separation and are associated with an adverse pressure gradient near the trailing edge.

The above discussion of the cluster transitions and associated evolution of the flow are based on discretizing the attractor, given by observations in the feature space, into $K = 10$ clusters. The clustering analysis is based on the *k-means* algorithm which is primarily concerned with minimizing the total within-cluster variance. The current cluster analysis focuses on obtaining the best compressed coverage of trajectories in the feature space. Such a data-driven, state-space discretization enables us to obtain a coarse-grained global control law over the entire feature space. It must be noted that increasing the number of clusters may reveal additional flow paths and other clustering techniques may result in alternate state space discretizations. Although there may be variability in the phases of the flow characterized with alternate discretizations, the essence of the clusters transitions

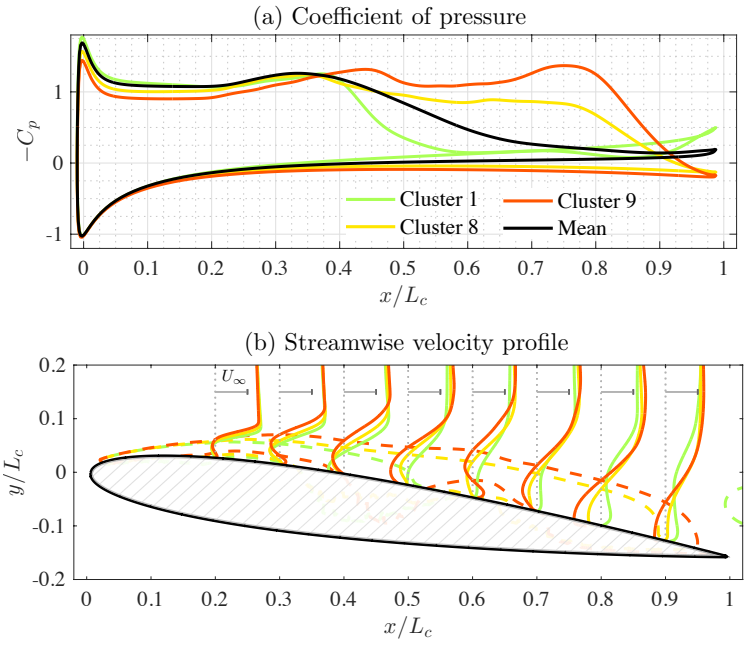


Figure 37: (a) Cluster-averaged pressure distribution over the suction and pressure surface of airfoil and (b) cluster-averaged streamwise velocity profiles for 2D baseline flow.

will remain. With the understanding of baseline transition dynamics, we now design cluster-based feedback control laws to control flow separation.

10.7. Optimized feedback control

Based on the coarse-graining of the feature space into clusters discussed in the previous section we design a cluster-based feedback control law. The actuation input $\hat{b}(t)$ in (10.5) at time t is determined with the feedback of the observable $s(t)$ in the feature space. The actuation input takes into account the control amplitudes at each cluster-discretized phase of the flow $\{b_k\}_{k=1}^K$ and the relative distance of the current state to the cluster centroids $\{c_k\}_{k=1}^K$ to obtain a smooth control law over all the clusters. Thus, actuation inputs associated with cluster in close proximity to the current measurement are weighed more strongly than those associated with clusters farther away. This defines a distance-based global control law, despite based on an initial clustering of the baseline configuration and which does not rely on the identification of characteristic coherent structures in the flow. The control amplitudes $\{b_k\}_{k=1}^K$ are then iteratively optimized using simplex search to achieve the desired objective as discussed in §10.4. We design an initial simplex using Latin hypercube sampling with zero-mean offset control amplitudes $\sum_k b_k = 0$. As the forcing amplitude, given by Eq. (10.5), is a weighted summation over cluster control amplitudes and depends on the distance from the cluster centroids, this zero-mean offset may not be fully realized with feedback control. Time averages in controlled flow are estimated over 12 periods of the baseline shedding frequency.

The optimization of the cost functional $\mathcal{J}$ is summarized in Figure 38(a). Following the initial simplex consisting of $K+1$ control cases (indicated by blue dots), cost functional is immediately minimized, typically within a few iterations. The square symbol denotes

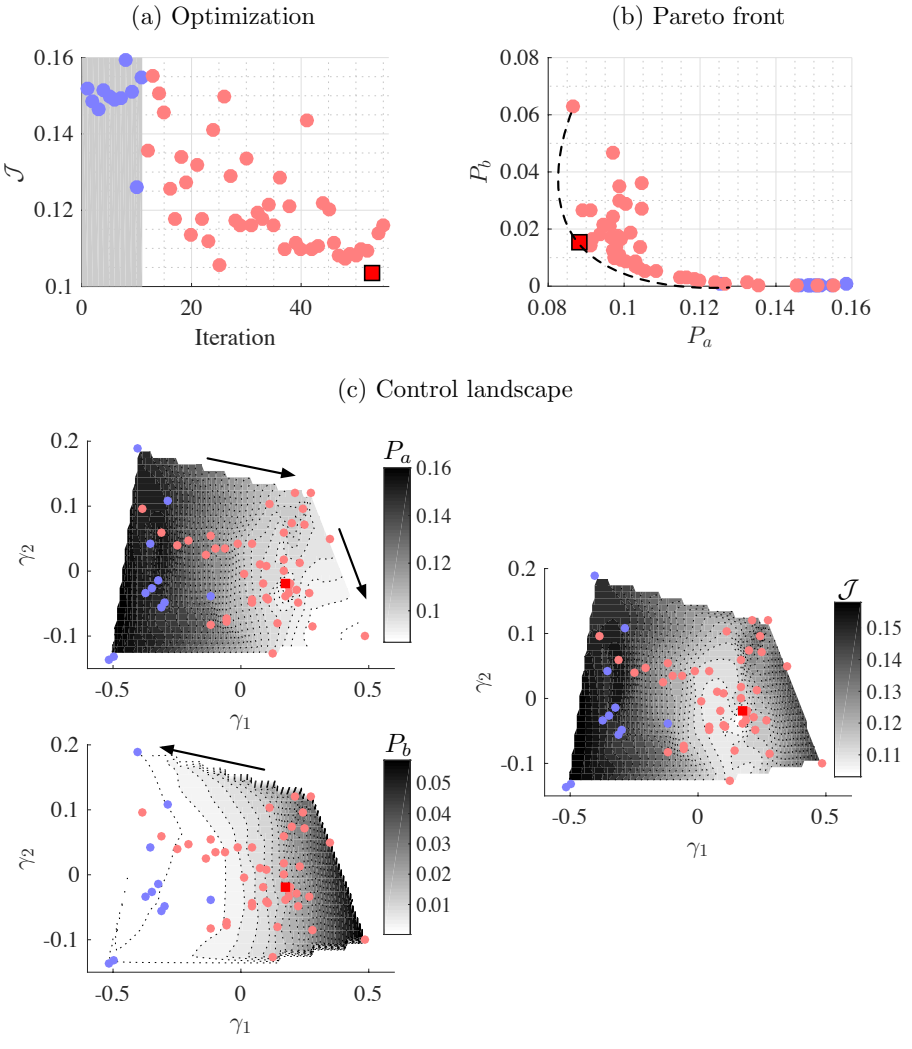


Figure 38: (a) Objective function minimization to determine the optimized control law and (b) Pareto front. The red square symbol denotes the optimized case. The dashed line in (b) indicates the Pareto front of control. (c) Control objective landscape, $P_a = P_a(\gamma_1, \gamma_2)$, $P_b = P_b(\gamma_1, \gamma_2)$, and $\mathcal{J} = \mathcal{J}(\gamma_1, \gamma_2)$ balancing P_a and P_b, all three determined using multidimensional scaling. The transparent blue dots indicate initial simplex control cases. The arrows indicate directions of minimization.

the optimized control case. The optimization over aerodynamic power P_a and actuation power P_b is shown in Figure 38(b). The dotted line highlights the Pareto front.

To visualize the control landscape and search space topology, MDS is performed as discussed in §10.4. MDS identifies a low-dimensional embedding of the control cases in terms of a proximity map, in which distances between points measure the similarity between control laws. Here, we extract a 2D proximity map over the (γ_1, γ_2) space. Each point in this proximity map stands for a control case or control law, respectively. Pairwise distances between the control cases over this map approximate their true distances estimated from the distance measure in Eq. (10.9). The similarity between control laws

in this map increases as the distance between them gets smaller. We fit surfaces of the form $P_a = P_a(\gamma_1, \gamma_2)$, $P_b = P_b(\gamma_1, \gamma_2)$ and $\mathcal{J} = \mathcal{J}(\gamma_1, \gamma_2)$ to all the evaluated control laws in Figure 38(c). The proximity maps provide information on the complexity of the objective functions, e.g. a single vs. multiple minima, and indicate optimization directions for minimizing the aerodynamic and actuation power. Minimizing aerodynamic power is associated with the direction of increasing γ_1, while minimizing actuation power is associated with direction of decreasing γ_1. As actuation power input increases (increasing γ_1), the aerodynamic performance improves. For $\gamma_1 > 0$, minimal aerodynamic power is obtained with decreasing γ_2. In summary, minimizing aerodynamic power involves maximizing γ_1 and minimizing γ_2, while minimizing actuation power involves minimizing γ_1. Balancing both power considerations, the control landscape shrinks at the optimal location for the cost functional $\mathcal{J}$. Beyond analysis of the optimization procedure, these proximity maps can help to accelerate the control law optimization by estimating the expected performance of control laws without evaluating them, which can then, e.g., be discarded if a control law with a similar performance has already been evaluated. A related control law performance predictor based on proximity maps has been previously proposed (Kaiser *et al.* 2017*a*).

We now investigate the optimized control case to gain insights on the control strategy uncovered by cluster-based control optimization. The baseline and optimally controlled trajectories in the lift-drag coefficient plane are shown in Figure 39(a). A 41% drag reduction is achieved with the optimized control law. In addition, the unsteadiness in the flow is reduced. In the inset of Figure 39(a), the time evolution of the drag coefficient with and without control is shown. In the optimally controlled flow, the trajectories are pushed away from the high drag states towards the low drag states of the flow. We compare the lift spectra $\hat{C}_L$ in baseline and control cases in Figure 39(b). A dominant peak for the optimal control is obtained corresponding to a forcing frequency of $St^+ = 0.243$, identified by the cluster-based optimization procedure. This frequency is thrice the dominant baseline shedding frequency. As the feature space is discretized based on the baseline trajectories, the cluster-based procedure identifies optimal flow actuation at harmonics of the baseline shedding frequency.

DISTRIBUTION A: Distribution approved for public release.

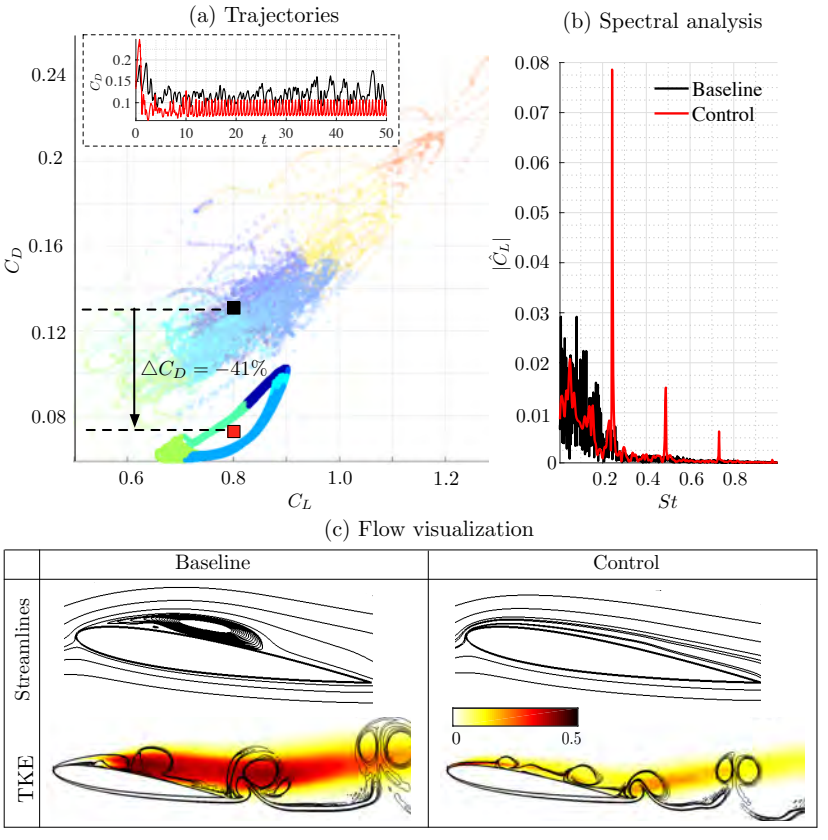


Figure 39: Comparison of baseline and optimized control case: (a) Trajectories, (b) spectral analysis of lift data, and (c) time-averaged streamlines and turbulent kinetic energy (TKE) fluctuations. Baseline trajectories are shown in transparent in (a). The contour lines in (c) (bottom) indicate instantaneous vorticity fields.

The time-averaged streamlines and TKE for both the baseline and controlled flows are shown in Figure 39(c). The contour lines in TKE correspond to the instantaneous spanwise vorticity. The streamlines indicate that the separation bubble is completely eliminated with control compared to the baseline resulting in a fully attached flow. Moerover, with control, the turbulent kinetic energy fluctuation decreases dramatically. The fluctuations in turbulent kinetic energy in the baseline are primarily a result of the unsteadiness in the flow which decreases with control. Further, the roll-up of the vortices is delayed. Moreover, the leading-edge vortices are receptive to the shear layer formed from the trailing edge resulting in a lock-on to the control frequency.

The cluster control amplitudes, i.e. jet velocity b_k, associated with the optimized control case are shown in Figure 40(a). Negative and positive amplitudes indicate suction and blowing, respectively. The cluster control amplitudes are negative for clusters $1-5$ and positive for $6-10$. This indicates that suction is performed in subset I (clusters $1-5$) where $C_D/\bar{C_D} < 1$. In the remaining clusters (subset II and III), blowing is performed. In the optimized control case, suction is performed in subset I to keep the trajectories in the low drag state. As discussed in §10.6, the transition to high drag cluster 9 occurs via the path $6 \to 8 \to 9$. Blowing is performed in these clusters to kick the trajectories

DISTRIBUTION A: Distribution approved for public release.

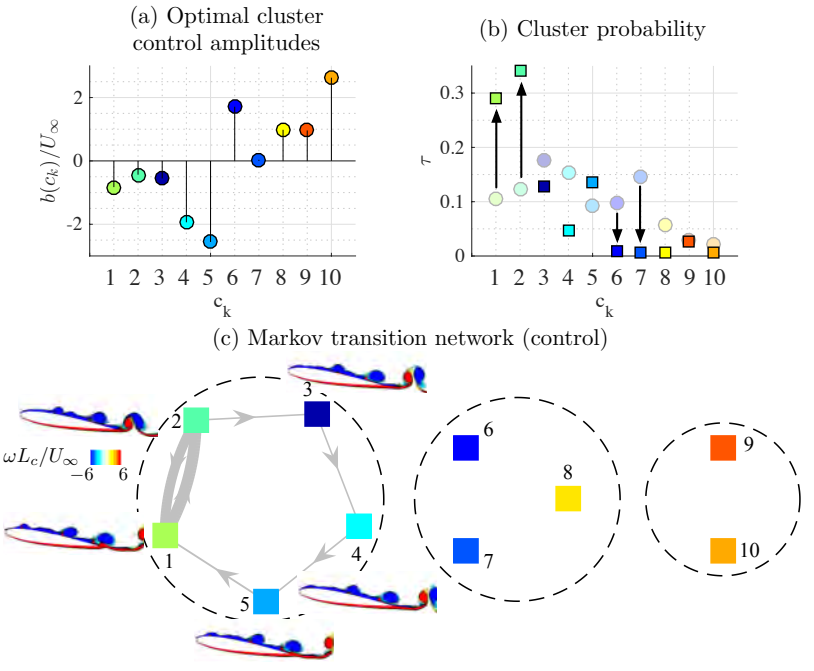


Figure 40: (a) Cluster jet velocity, (b) cluster residence probability for baseline (shown in transparent round symbols) and controlled flows (shown in bright square symbols) and (c) controlled Markov transition network with instantaneous vorticity contours ($\omega L_c/U_\infty$).

away from these high drag states. For these clusters, $C_D/\bar{C}_D > 1$. The flow states in cluster 9 always transition to cluster 10. The high blowing ratio in cluster 10 causes the flow to transition to cluster 5, where highest amount of suction is performed to keep the flow in the low drag states. At steady state, the mean cluster-based control amplitudes is $|\bar{\hat{b}}|/U_\infty = 0.78$. This leads to a momentum coefficient $\bar{C}_\mu = 0.016$.

We compute the cluster probability for controlled flow trajectories and compare these with the baseline flow as shown in Figure 40(b). The controlled flow spends a lot of time in the lowest states of cluster 1 and 2. The cluster probability associated with subset II clusters $6, 7, 8$ reduces considerably. The cluster transitions in the controlled flow are summarized in the controlled Markov transition network shown in Figure 40(c). The iterative optimization procedure coupled with cluster-based control laws results in a re-routing of the trajectories to reduce drag power associated with the flow. This minimizes the transition probability to clusters in subset II (intermediate drag states) which in turn avoids cluster transitions to clusters in subset III (high drag states). The controlled flow exhibits a limit cycle behavior, which is resolved by the low drag clusters in subset I.

Thus, the cluster-based control procedure identifies an optimized feedback control law to reduce flow separation. Suction is performed in the low drag clusters $C_D/\bar{C}_D < 1$, while blowing is performed in the remaining clusters. With control, the Markov transition network is modified so that the transition dynamics reside in the low drag states.

10.8. Separation control of the three-dimensional flow around an airfoil

In the previous section, the cluster based methodology was demonstrated for 2D flow control. In this section, we extend the approach to deduce global control laws

for 3D separated flows over an airfoil in a model-free manner. LES of 3D flows are computationally much more expensive than the 2D counterpart. The use of suction as demonstrated in the previous section may necessitate the use of additional devices like suction pump for practical applications, adding to weight constraints. Periodic excitation using blowing has shown to be effective for active flow control and can be realized by engine bleed in aerospace applications. Using the cluster-based strategy, we deduce an optimized global feedback control law for unsteady blowing to minimize actuation power input and correspondingly enhance aerodynamic performance.

10.9. Baseline feature space clustering

The 3D baseline separated flow is associated with von Kármán vortex shedding in the wake and Kelvin-Helmholtz instabilities in the shear layer. The dominant frequency associated with von-Kármán vortex shedding is $St = 0.0884$ and the dominant shear layer frequency associated with the Kelvin-Helmholtz instability is almost an order of magnitude larger, $St = 0.6952$. For the cluster-based analysis of the 3D baseline flow, a time series trajectory of the observables $s^b(t)$ is collected at a constant time step $\triangle t = 0.0036$. The time step is chosen so as to resolve each period corresponding to the dominant wake shedding frequency with 500 time instances. The feature space segmentation into $K = 10$ clusters is shown in Figure 41(a). The unsteadiness in the 3D baseline flow, observed from the range of fluctuation in $\dot{C_L}$, is less compared to the 2D baseline flow, which leads to more frequent cluster transitions. The discrepancy between the probability of the higher drag state of cluster 10 and lower drag state of cluster 1 are also correspondingly reduced. The cluster probabilities of both these clusters are comparable, as shown in Figure 41(b). As cluster 6 is positioned closest to the centroid of the full data set, $\bar{c}$, its cluster probability is the highest. The transition probabilities of the cluster transition matrix are shown in Figure 41(c). The flow states in cluster 1 and 9 always transition to cluster 2 and 10, respectively, resulting in a high probability of transition.

Performing directed modularity maximization, three subsets can be identified: subset I:$\{1, 2, 3, 4\}$, subset II:$\{5, 6, 7, 8\}$, and subset III:$\{9, 10\}$. These subsets are highlighted in red boxes in Figure 41(c). High probability inter-subset transitions originate from cluster 3 in subset I, 7 in subset II and 10 in subset III. The grouping of the clusters into subsets can also be observed over the cluster centroids of the drag co-ordinate, shown in Figure 41(d). Here, subset I corresponds to low drag states of the flow $C_D(c_k)/\bar{C}_D < 0.98$, subset II correspond to intermediate drag states $0.98 \leqslant C_D(c_k)/\bar{C}_D \leqslant 1.1$, and subset III corresponds to the high drag states $C_D(c_k)/\bar{C}_D > 1.1$.

Further clarity in the transitions can be obtained by looking at the Markov transition network shown in Figure 42. We see high volume of transitions in this 3D flow network compared to the 2D Markov transition network. High-probability cluster transitions involve cluster 6 due to its central position in feature space. The transition to high drag clusters 9 and 10 can only occur via cluster 7, which we refer to as the switching cluster in the 3D flow configuration. The midspan pressure flow fields associated with the cluster centroids are shown in Figure 42. We have also shown representative instantaneous flow fields in terms of isosurfaces of the Q-criterion, which is colored by pressure in each cluster. Coherent structures in the wake from von-Kármán vortex shedding in the low and intermediate drag clusters can be observed from the corresponding flow fields. The strengthening of the low pressure core near the mid-chord of the airfoil and elongation of the associated leading-edge vortex sheet in cluster 3 result in a transition to the switching cluster 7, from which the flow evolves to the high drag states. These high-drag states are accompanied by a spanning of the low pressure core across the entire airfoil

DISTRIBUTION A: Distribution approved for public release.

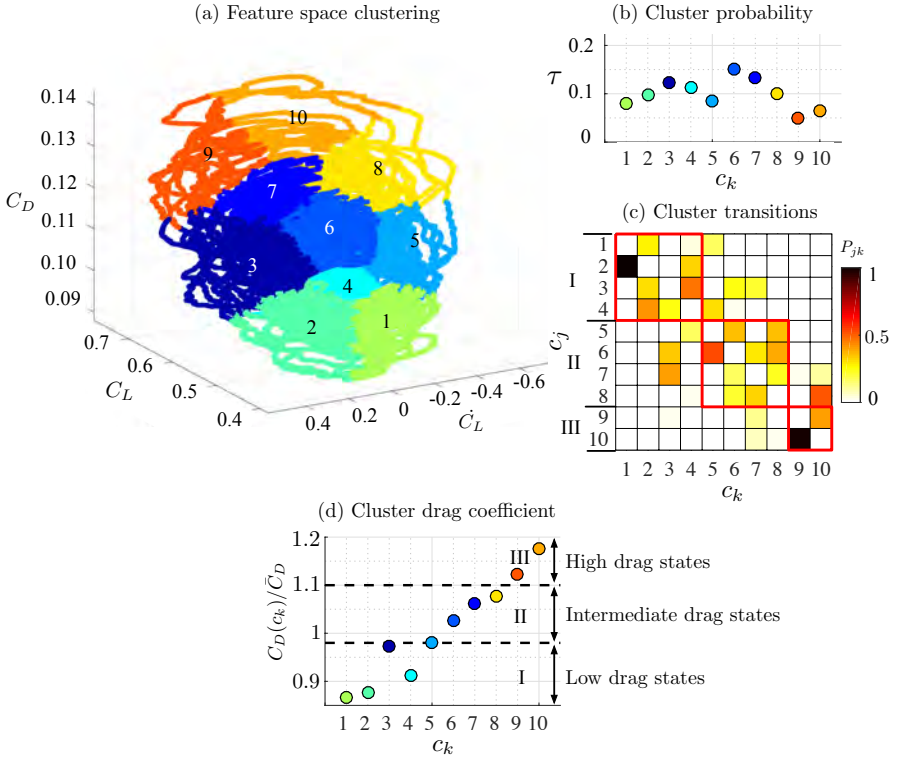


Figure 41: (a) Feature space clustering of 3D baseline flow data. (b) Cluster probabilities, (c) transitions probabilities and (d) normalized, average drag coefficient across clusters. The red boxes and the dashed circles indicate the cluster subsets.

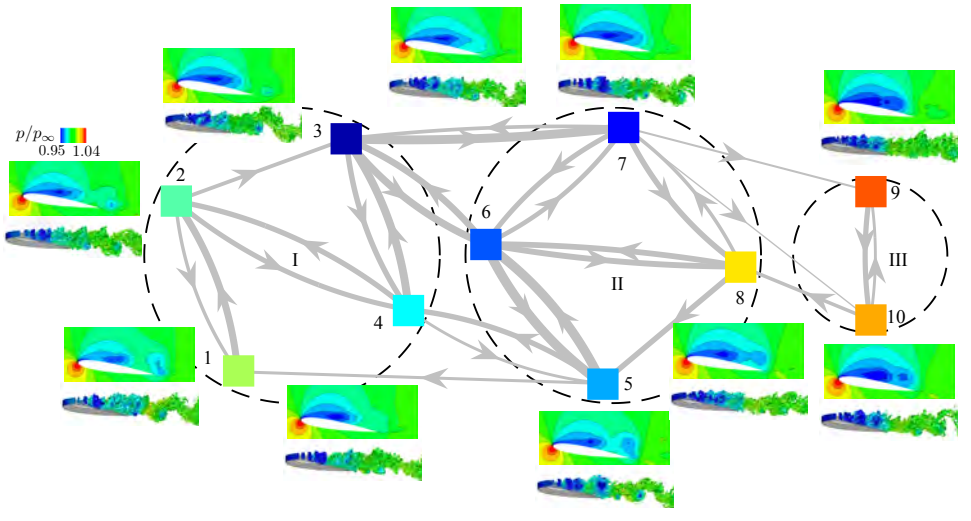


Figure 42: Graph of Markov chain highlighting transitions between clusters for the 3D baseline flow. The dashed circles indicate the cluster subsets.

DISTRIBUTION A: Distribution approved for public release.

surface leading to a complete flow separation. Here, large roll-up of vortical structures are observed near the trailing edge which do not detach. Alternatively, if the vortex sheet rolls up and detaches in cluster 3, the resulting transitions $3 \rightarrow 6 \rightarrow 5$ ultimately lead the trajectories to low-drag cluster states associated with 1 and 4. Thus, the modification of the cluster pathway $3 \rightarrow 7 \rightarrow 9 \rightarrow 10$ is an important consideration for flow control, particularly for drag reduction. In the next section, we use the cluster partition to design the flow control strategy.

10.10. Optimized feedback control

In the review by Greenblatt & Wygnanski (2000), it was shown that the excitation of Kelvin-Helmholtz instabilities in the shear layer is essential for suppression of flow separation. Knowledge of these instabilities can be leveraged to design flow control strategies. Here, a cluster-based control law is optimized without assuming any prior knowledge of instabilities.

We perform feedback control using the discretized clusters in 3D baseline flow. The objective is to deduce blowing amplitudes in characteristic phases of the flow in order to maximize the aerodynamic performance. This is achieved by iteratively optimizing the control amplitudes b_k in each cluster in an automated fashion to minimize the cost functional $\mathcal{J}$, involving aerodynamic and actuation power. We particularly seek a control law associated with minimum actuation power for feedback control. The value associated with γ in Eq. (10.6) influences the relative importance of actuation power (input cost) and aerodynamic power (state cost) in the objective function evaluation. Increasing γ lowers the relative importance of actuation power and may yield an optimized control law associated with higher C_μ. As mentioned in §10.4, we set $\gamma = \bar{C}_L^{3/2}$ to obtain the optimized cluster-based amplitudes. The feedback gain β in Eq. (10.5) associated with these optimized control amplitudes is subsequently increased to explore the flow control implications at higher C_μ.

To speed up the 3D computations, we employ a parallel simplex method as in the work by Lee & Wiswall (2007). The method is similar to the original simplex method, except that multiple control simulations can be run in parallel to accelerate the optimization process. In the 2D control effort, the cluster-based control optimization was unconstrained allowing for both blowing and suction jet velocities in the clusters. However, in the 3D control effort, a constraint is added to restrict the control amplitudes in the simplex search such that pure blowing is performed (Luersen *et al.* 2004). This requires $0 \leqslant b_k/U_\infty \leqslant 3.3$. The lower constraint ensures that only pure blowing is performed and the upper constraint limits the highest blowing ratio that can be achieved. As mentioned before, the addition of this constraint is motivated primarily to examine if the cluster-based strategy can take advantage of shear layer and wake instabilities to control flow separation.

Time averages in the controlled flow are estimated over eight periods of dominant wake shedding frequency. The cost functional associated with the control simulations is outlined in Figure 43(a). Following the initial simplex cases (shown in blue), the optimization procedure iteratively minimizes the desired objective function. The Pareto front of control is shown in Figure 43(b). Case A is the optimized control case deduced from the optimization procedure. We also highlight another control case B, whose objective function evaluation is close to case A. Control case A is associated with lower actuation power (P_b) and correspondingly lower $C_\mu = 0.0068$, while control case B is associated with higher $C_\mu = 0.016$. However, the aerodynamic power (P_a) is lower for B compared to A. In order to evaluate the effect of minimization of cost functional

DISTRIBUTION A: Distribution approved for public release.

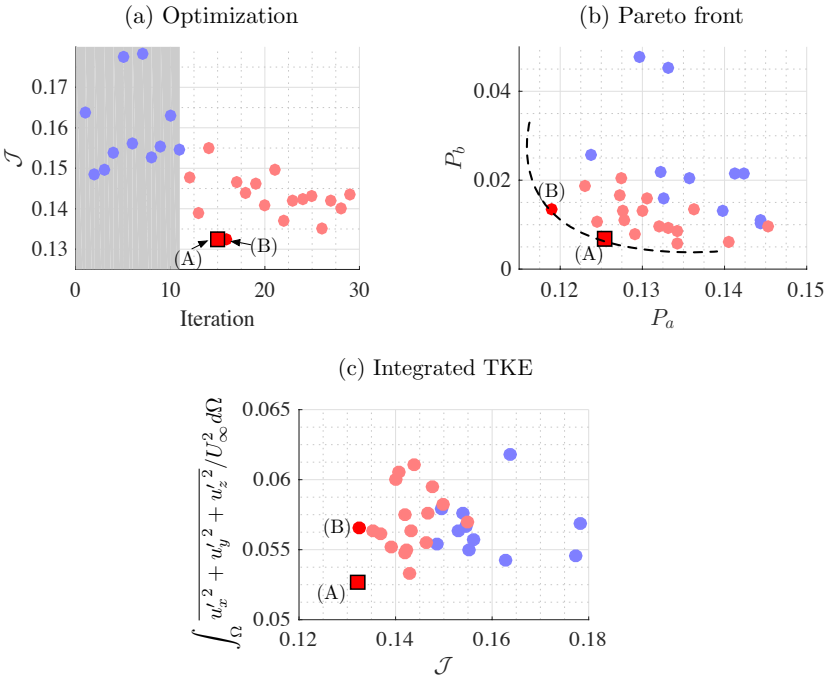


Figure 43: (a) Objective function $\mathcal{J}$ minimization to determine the optimized control case, and (b) Pareto front (P_a, P_b). The square symbol denotes the optimized case. (c) Integrated turbulent kinetic energy over the entire computational domain. The transparent blue dots indicate initial simplex control cases and the red square symbol denotes the optimized case.

to the overall flow physics, we integrate the TKE in the computational domain for all the control cases, which is shown in Figure 43(c). A minimum integrated TKE for the optimized control case A is obtained. Due to a lower actuation power input and lower velocity fluctuations in the streamwise wake associated with drag reduction, the TKE fluctuations are minimized with the optimized control law.

The control landscape over the optimization variables obtained using MDS is shown in Figure 44. Compared to the 2D control effort, the proximity map is much more complex for 3D flow control. Although the actuation power P_b increases with γ_1 for $\gamma_1 > 0$, the aerodynamic power P_a does not decrease correspondingly. The control landscape shrinks at the optimal location for the cost functional $\mathcal{J}$ balancing both power considerations. Control case A and B occupy different positions in the proximity map as shown in Figure 44. The cluster-based procedure is able to extract not only global minima but also local minima, highlighted from different regions of the proximity map. For further optimization of the control laws with lower C_μ, a simplex consisting of $K+1$ cases near the current optimized case A can be chosen. For further optimization of the control laws with higher C_μ and better aerodynamic performance, a simplex consisting of $K+1$ cases near the case B can be chosen. Thus, the gradient-free search for optimizing cluster control amplitudes explores different regions of the control landscape effectively. The proximity map can serve as a guide for tracking the performance of the flow control cases.

Using the same cluster control amplitudes $\{b_k\}_{k=1}^{K}$ obtained from the optimized case

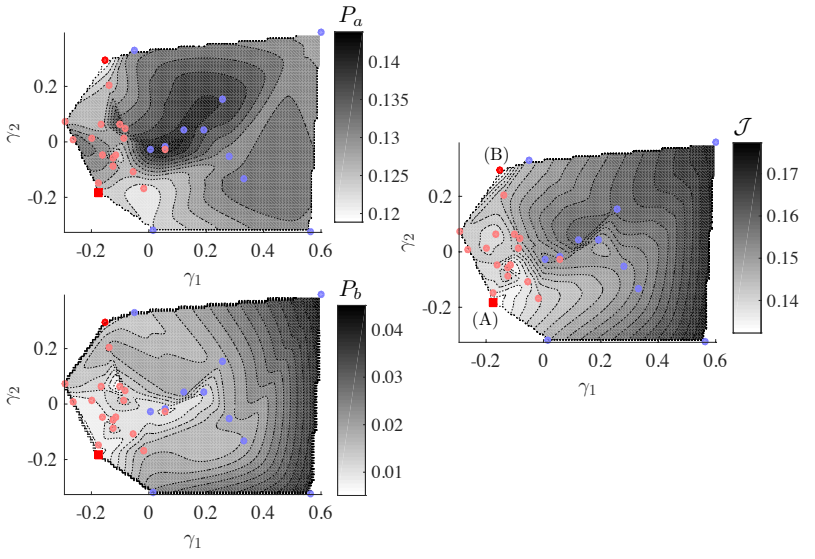


Figure 44: Control landscape, $P_a = P_a(\gamma_1, \gamma_2)$, $P_b = P_b(\gamma_1, \gamma_2)$, and $\mathcal{J} = \mathcal{J}(\gamma_1, \gamma_2)$, using multidimensional scaling. The transparent blue dots indicate initial simplex control cases and the red square symbol denotes the optimized case.

A, we increase the feedback gain $\beta = 1.6$ in Eq. (10.5) to evaluate the effect of increasing actuation input on control performance. This yields a correspondingly higher $C_\mu = 0.016$. This control case with higher feedback gain will be referred to as case C in the following discussion. Comparative examination of control cases A and C with the baseline flow is performed in Figure 45. The drag reduction obtained in both cases is shown in Figure 45(a). The black, red and blue dashed lines indicates the mean drag associated with baseline, case A, and case C, respectively. We obtain a 13% drag reduction for case A and 20% drag reduction for case C. A spectral analysis of the lift coefficient $|\hat{C}_L|$ is performed to highlight the associated amplitude and frequency range of forcing. These are shown in Figure 45(b). For the optimized control case A, the forcing operates in the range of wake frequencies with the peak at $St^+ = 0.0844$, close to the dominant shedding frequency. Periodic forcing near the wake frequency with commensurate C_μ yields a drag reduction of 11.5% (Amitay & Glezer 2002*b*). For control case C, along with the wake frequencies, the dominant shear layer frequency $St^+ = 0.695$ is triggered with the cluster-based feedback control. The increase in feedback gain associated with the optimized control amplitudes results in better aerodynamic performance, observed from the ensuing drag reduction in Figure 45(a). The spectral richness of the feedback forcing can be observed across both control simulations. The fact that the cluster-based control strategy in adaptively able to force the flow at characteristic frequencies corresponding to fundamental instabilities of the baseline flow purely from the feedback of select observables demonstrates the power of this data-based approach.

We also show the streamlines and instantaneous flow fields associated with the control cases A and C in Figure 45(c). The blue dashed lines indicate the contour line corresponding to time- and spanwise-averaged streamwise velocity, $\bar{u}_x/U_\infty = 0$. This characterizes the extent of the separation region in the flow. For both the control cases, the size of the separation bubble is reduced compared to the baseline flow. For case A, the flow reattaches near the mid-chord of the airfoil. However, in this case, vortices merge near

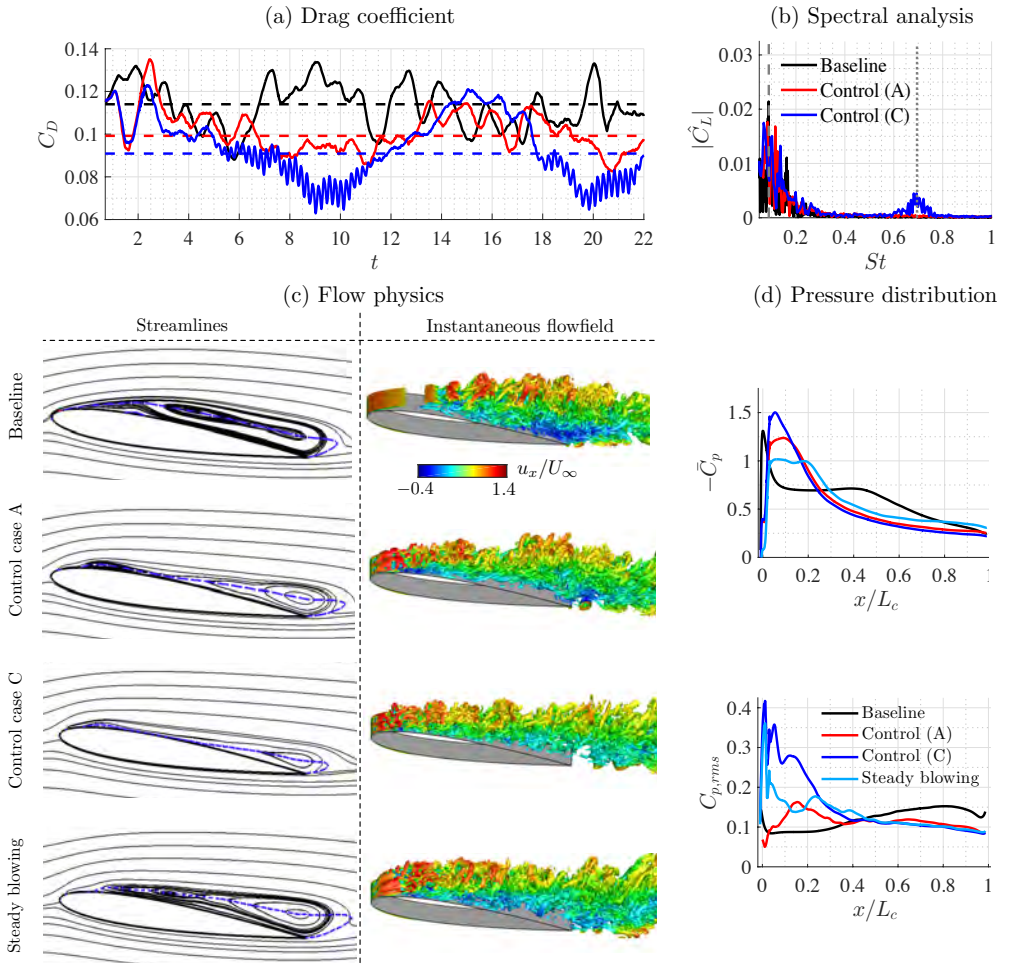


Figure 45: Comparison of baseline, optimized control case A and control case C with higher feedback gain: (a) Drag coefficient, (b) spectral analysis of lift data, (c) time-averaged streamlines and instantaneous flow field (highlighted by Q-criterion) colored by streamwise velocity, and (d) time-averaged and root-mean square fluctuation of coefficient of pressure distribution on suction side of the airfoil. The dashed lines in (a) indicate the mean drag. The dashed line in (b) corresponds to the dominant shedding frequency and dotted line corresponds to the shear layer frequency. The blue dashed line in (c) indicates the contour line of zero time- and spanwise-averaged streamwise velocity.

the trailing edge and the flow separates. A streamwise velocity deficit upstream translates to this trailing edge separation. In the control case C, the size of the separated region is considerably reduced. The vortical structures in the flow field are highlighted by the a level set of the Q-criterion, which is colored by the streamwise velocity component. Actuation-triggered shear layer frequencies result in a break-up of spanwise vortices. Suppression of separation due to a faster laminar-turbulent transition over the separation bubble is obtained, which results from the break-up of spanwise vortices. The coherence in the wake originating from von Kármán vortical structures is correspondingly lost leading to entrainment of free-stream momentum and mixing. This is consistent with

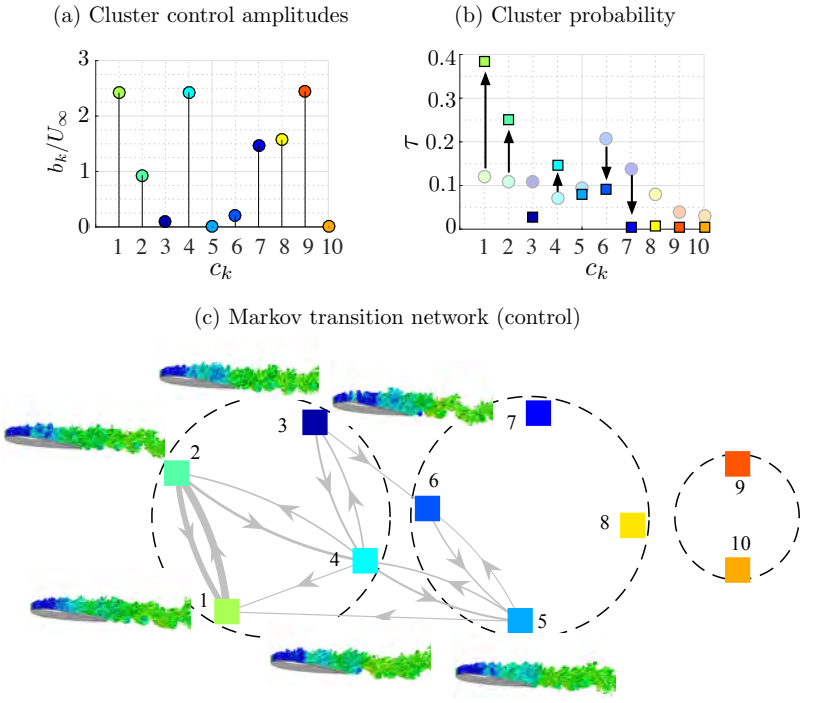


Figure 46: Comparison of optimized control case A with baseline: (a) Cluster jet velocities, (b) cluster probability for baseline (shown in transparent round symbols) and controlled flows (shown in square symbols) and (c) graph of optimally controlled Markov chain.

the conclusions in Greenblatt & Wygnanski (2000). To show the effectiveness of the cluster-based strategy, we compare control case C with steady blowing performed at the same $C_\mu = 0.016$. The time-averaged streamlines and instantaneous flow field is shown in Figure 45(c). Steady blowing does not suppress separation as well as control case C as it does not leverage fundamental instabilities in the flow.

In both control cases A and C, we observe a low frequency modulation in the drag coefficient. This modulation causes an increased unsteadiness in the drag force. These have been observed before in pulse-modulated actuated studies reported in Amitay & Glezer (2006*a*). Such modulation results from a sub-optimal pressure recovery closer to the trailing edge of the airfoil. Previous studies with a model-based feedback control have shown that suppressing this low frequency modulation yields additional performance benefits (Nair *et al.* 2018). There is scope for a model-free extension to suppress the low frequency modulation using the current feedback control strategy using additional clusters.

The average and r.m.s fluctuations of the pressure coefficient distribution over the suction surface of the airfoil are shown in 45(d). For control case C, the average suction pressure is highest at $x/L_c = 0.1$. Although the location of peak in airfoil suction pressure for control case A and case C is similar, the peak value for case A is lower owing to lower C_μ. For both control cases, the flat pressure region disappears indicating a faster laminar-turbulent transition. Also, the accelerated roll-up and transition can be observed by peaks in the r.m.s fluctuations of the pressure coefficient over the suction surface.

The cluster based control amplitudes b_k corresponding to the optimized control case

DISTRIBUTION A: Distribution approved for public release.

A is shown in Figure 46(a). Only positive amplitudes are present which indicate that only blowing is performed in all clusters as constrained in the optimization procedure. Clusters 1, 4, and 9 have maximum blowing amplitudes, followed by clusters 8, 7, and 2. The remaining clusters, $\{3, 5, 6, 10\}$ have considerably lower control amplitudes. The cluster probabilities for the controlled flow trajectories are compared with baseline flow as shown in Figure 46(b). The controlled flow spends long times in the low drag states of clusters 1, 2 and 4. The cluster probabilities associated with subset II clusters $6, 7, 8$ reduces considerably. The cluster transitions in the controlled flow are summarized in the controlled Markov transition network shown in Figure 46(c). Blowing in cluster 2 deters transitions to cluster 3. Also, blowing in cluster 7 deters the transition $3 \rightarrow 7$. This eliminates the transition to the high drag states of the flow leading to drag reduction and controlling flow separation. Natural baseline transitions are invoked in clusters 3, 5, and 6 eventually leading the flow to low drag cluster states. In addition to the above mentioned cluster transition dynamics, a cyclic transition between clusters 1 and 2 at shear layer frequency is obtained for case C.

Thus, we can achieve suppression of separation in 3D flow using a cluster-based control strategy. As demonstrated, the optimized cluster-based control amplitudes lead to a transformed Markov transition network, essentially re-routing trajectories to desirable state space regions. The model-free, iterative optimization of the global control law over the coarse-grained feature space leads to optimized cluster transitions and cluster probabilities that maximize aerodynamic performance. The approach can easily be extended to achieve other desired performance objectives. Also, only the information of the feature space trajectories are required for both control and optimization of the feedback control laws.

DISTRIBUTION A: Distribution approved for public release.

11. Phase-based control of periodic fluid flows

11.1. Introduction

Time-periodic fluid flows are ubiquitous with examples of pulsatile flow of blood, rotating flows in turbomachines and engines, and vortex shedding in the wake of transportation vehicles. Controlling fluid flows (Bewley 2001; Kim & Bewley 2007; Brunton & Noack 2015*a*), and periodic vortex shedding specifically (Colonius & Williams 2011), has remained a significant engineering challenge, with tremendous potential impact for industry and technology. Despite the challenges of flow control, researchers have long been inspired and encouraged by the exceptional performance of biological flyers and swimmers that take advantage of unsteady aerodynamics through flapping for phenomenal maneuverability in a range of flow conditions (Ellington *et al.* 1996; Dickinson *et al.* 1999; Shyy *et al.* 2008; Dabiri 2009; Wu 2011; Ramananarivo *et al.* 2011). To achieve such performance, even in high-amplitude gusting conditions, the animal must sense the flow and adjust its wings or fins in real-time (Shyy *et al.* 2016). For instance, Dickinson *et al.* (1999) suggested that regulating the phase of wing rotation effectively controls the aerodynamic forces during maneuvers. The wing motion is, in turn, regulated by modifying the activation phase of the associated muscles (Dickinson *et al.* 2000), which involves precise timing modulation (Sponberg & Daniel 2012). Inspired by these natural systems, there are ongoing efforts to develop simple yet robust control laws for engineering flows.

Most engineering systems rely on steady or quasi-steady analysis, for which well-established controllers can be designed and implemented within comfortably predefined ranges of uncertainty and perturbations (Kim & Bewley 2007). However, linearizing the flow about a steady state is often inadequate to precisely control large-amplitude flow oscillations and unsteadiness (Bewley 2001). At the moment, there are few studies that perform control of flows with large unsteady fluctuations, due to the challenges of high-dimensionality, strong nonlinearity, and time-delays (Brunton & Noack 2015*a*; Brunton *et al.* 2020*a*). One of the main approaches for analyzing perturbation dynamics about time-varying periodic base flows is Floquet analysis (Herbert *et al.* 1987; Barkley & Henderson 1996). The transient instabilities in these time-varying flows can also be described using optimally time-dependent modes (Babaee & Sapsis 2016), which can then be leveraged for flow control (Blanchard & Sapsis 2019).

The present work is motivated by the seminal work of Pastoor *et al.* (2008) where they designed a physically motivated phase controller for turbulent flow around a D-shaped body. In this work, the mechanism of phase control explicitly synchronizes the upper and lower shear layer development. By delaying the formation of vortex shedding at low control amplitudes, this phase control outperforms a number of other control strategies in terms of the drag reduction achieved. However, their phase control requires substantial effort in designing a nonlinear reduced-order vortex model for tracking and predicting the dependence of vortex shedding phase on control inputs. In this work, we describe these dependencies of vortex shedding phase on control inputs though a simple sensor-based phase-reduction framework.

Phase-reduction analysis provides a complementary approach to Floquet theory, enabling a description of the phase of a dynamical system comprised of interacting nonlinear oscillators (Nakao 2016; Kuramoto & Nakao 2019). Phase-reduction analysis has a rich history, especially in neuroscience (Brown *et al.* 2004; Schultheiss *et al.* 2011), dating back to the works of Malkin (1949), Kuramoto (1984) and Winfree (2001). The analysis identifies a set of points in phase space that result in oscillations having the same phase, called *isochrons* (Guckenheimer 1975). Similarly, amplitude reduction identifies a set of

DISTRIBUTION A: Distribution approved for public release.

points that converge to the same trajectory, called *isostables* (Wilson & Moehlis 2016). These isochrons and isostables are eigenfunctions of the infinite-dimensional Koopman operator (Koopman 1931; Mezić 2005; Mauroy *et al.* 2013). Techniques for amplitude-phase reduction can also be extended to analyze transient responses far from the stable limit-cycle dynamics (Shirasaka *et al.* 2017) and to build network models (Nair *et al.* 2018). Further, these amplitude-phase response models have been widely used for control in the context of neuronal dynamics and circadian rhythms (Efimov *et al.* 2009). One such effort develops a closed-loop approach to tune deep brain stimulation to suppress essential tremors (Holt *et al.* 2016). In this work, we develop a phase-response model to track transient responses in periodic fluid flows and subsequently alter the transient flow characteristics.

Transient flow control is a challenging problem due to the fundamental difficulty in (i) precisely characterizing the response to actuation due to the nonlinear, high-dimensional nature of the underlying physics, and (ii) developing control strategies over a time scale commensurate with variations in the flow. Phase reduction analysis alleviates the former concern as it characterizes the effect of perturbations on the phase dynamics of oscillatory systems. The analysis is especially suitable for high-dimensional systems as it can collapse the full-state dynamics to a phase-based representation in a single scalar variable. The collective phase description of spatially extended systems, such as oscillatory convection, was first introduced in the works of (Kawamura & Nakao 2013, 2015). More recently, Taira & Nakao (2018) used a phase-reduction approach to analyze synchronization of perturbation dynamics about time-varying base states for unsteady wake flows. Using this approach, the conditions for synchronization (lock-on) to external excitation were determined (Munday & Taira 2013). Moreover, phase-based characterization of the global flow field around a bluff body (Khodkar & Taira 2020) and sensitivity of phase dynamics to body motion (Khodkar *et al.* 2020). Such a phase-reduced representation of the flow physics forms the basis of the analysis in this work. Instead of determining the lock-on conditions, we deduce the optimal external excitation to achieve transient control of periodic fluid flows.

In this work, we demonstrate the ability to perform rapid, energy-efficient control of highly unsteady periodic flows using phase-sensitivity analysis. We leverage recent developments in variational methods to determine phase-based optimal control inputs that alter the transient behavior (synaptic activation time) of spiking neurons (Moehlis *et al.* 2006; Monga *et al.* 2019), extending them to enable high-dimensional transient flow control. We demonstrate our approach on several relevant flow systems, including the illustrative Stuart-Landau oscillator model, and the canonical flow examples of laminar flow past a cylinder and flow over an airfoil. In each example, we demonstrate the robustness of the proposed approach, as well as the ability to incorporate physically realistic measurements and actuation strategies. Based on the phase-reduction analysis, we introduce two control optimization strategies, based on the Euler-Lagrange equations and model predictive control. This formulation enables the real-time control of the phase of vortex shedding, including prescribing the desired time of vortex pinch-off. The ability to modify the phase dynamics of time-varying vortical flows accelerates or decelerates the associated vortex shedding process (Pastoor *et al.* 2008; Joe *et al.* 2011), which in turn affects the unsteady aerodynamic forces. Thus, we have provided a useful framework for transient control in oscillatory fluid flows, enabling the manipulation of aerodynamic characteristics in a time frame that is faster than the characteristic time of the flow oscillation.

DISTRIBUTION A: Distribution approved for public release.

11.2. Formulation

Our overarching goal is to characterize and control the phase dynamics of oscillatory fluid systems, which may be described as a dynamical system of the form

$$\dot{\boldsymbol{q}} = \boldsymbol{N}(\boldsymbol{q}) + \boldsymbol{f}, \tag{11.1}$$

where $\boldsymbol{q}$ is the oscillatory variable and $\boldsymbol{f}$ is the external forcing or control input. The fluid flows in this paper are governed by the non-dimensional, incompressible, forced Navier-Stokes equation where $\boldsymbol{q} = \boldsymbol{q}(\boldsymbol{x}, t)$ is the velocity field, defined over a spatial coordinate $\boldsymbol{x}$ and time t, and $\boldsymbol{f} = \boldsymbol{f}(\boldsymbol{x}, t)$ is the external forcing (actuation). It is common to represent the flow field discretely over a high-resolution spatial domain, resulting in a high-dimensional flow state $\boldsymbol{q}$ whose evolution is described by a system of coupled nonlinear ordinary differential equations. In these high-dimensional fluid flow systems, the oscillatory variable and forcing both depend on space and time.

A large class of high-dimensional unsteady fluid flows have time-periodic behavior

$$\boldsymbol{q}(\boldsymbol{x}, t) = \boldsymbol{q}(\boldsymbol{x}, t + T), \tag{11.2}$$

where T is the period of the flow $\boldsymbol{q}$ and $\omega_n = 2\pi/T$ is the corresponding natural frequency of oscillation. We focus on such time-periodic unsteady fluid flows in this work.

This work designs an energy-optimal forcing $\boldsymbol{f}(\boldsymbol{x}, t)$ to control the oscillatory phase of time-periodic fluid flows. Our work is inspired by two threads of prior work: phase-reduction analysis for fluid flows (Taira & Nakao 2018) and energy-optimal phase control for neuronal systems (Moehlis *et al.* 2006). Combining these approaches results in a strategy for energy-optimal phase control of unsteady fluid flows, summarized in Figure 68. The overall procedure is carried out in two stages as follows:

(i) **Phase-reduction analysis:** Following the work of Taira & Nakao (2018), we extend phase-reduction analysis (Nakao 2016) to describe the high-dimensional behavior of periodic wake flows. This stage requires (a) introducing perturbations to the flow at various phases of oscillation, (b) tracking the effect of these perturbations compared to baseline oscillations, and (c) constructing a reduced-order model described by the phase-sensitivity function. The phase-based reduced-order model describes the phase dynamics of oscillations in the unsteady fluid flow system. This modeling procedure is explained in more detail in §11.2.1.

(ii) **Optimal phase control:** In this stage, we extend the optimal control strategy developed by Moehlis *et al.* (2006) to unsteady fluid flows. In particular, we alter the characteristic phases of unsteady fluid flow systems on a timescale that is commensurate with the characteristic flow timescale. This characteristic time corresponds to the natural shedding frequency in unsteady bluff-body flows. This stage includes (d) an optimal open-loop control design strategy using the Euler–Lagrange equations to determine the control input required for desired phase alteration of the flow, (e) implementation of the control law in the unsteady fluid flow, and (f) evaluation of the transient and steady-state control performance along with the associated flow physics. The open-loop control forcing is designed offline using the phase-reduced model and is then introduced to the flow. It is also possible to use the reduced-order model for real-time feedback control with model predictive control. This optimal phase control procedure is described in detail in §11.2.2.

Before we describe the formulation for controlling oscillatory phase in detail, we outline some of the goals and objectives of this work. In terms of the methodology, we extend the optimal phase-based control formulation of Moehlis *et al.* (2006) to high-dimensional dynamical systems, in particular, for unsteady fluid flows. An alternative closed-loop

DISTRIBUTION A: Distribution approved for public release.

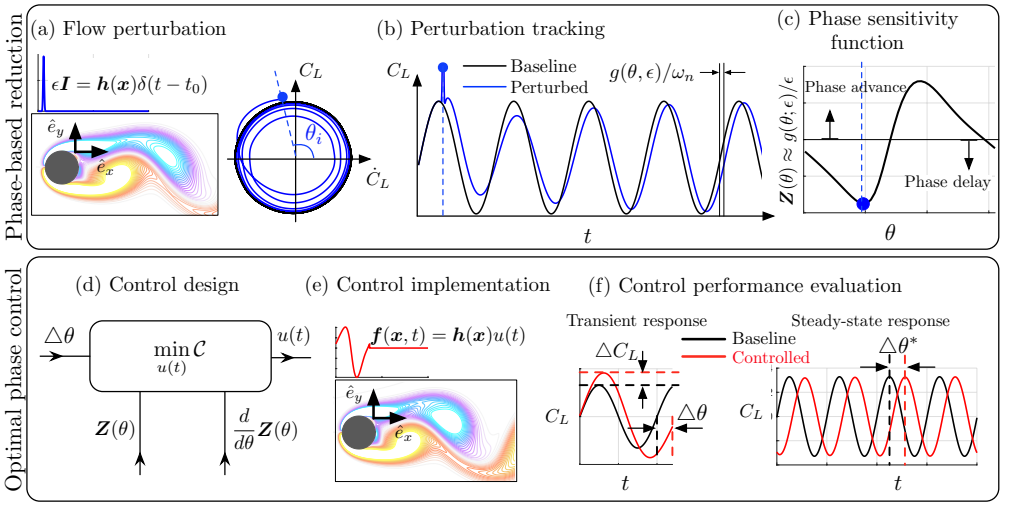


Figure 47: Overview of phase reduction analysis: (a) Introduction of impulse flow perturbation, (b) baseline and perturbed lift coefficient trajectories (left: phase portrait of oscillations in $\dot{C}_L$ - C_L plane, right: time history) and (c) phase sensitivity function $\boldsymbol{Z}(\theta)$. Overview of energy-optimal phase control: (d) Design of optimal control input $u(t)$ for desired phase shift $\triangle\theta$ and (e) controlled flow with the forcing $\boldsymbol{f}(t)$, (f) transient and steady-state response of lift coefficient on application of control.

model predictive control strategy is also proposed. Some of the specific physics-based objectives are to (i) achieve a transient phase shift in unsteady fluid flow oscillations by a desired amount after implementation of control, (ii) explore the changes in the maximum lift coefficient in the immediate transient after application of control for varying phase alterations, (iii) examine the asymptotic phase shift achieved in steady-state, long after the control is switched off, and (iv) investigate the implications of phase control on the vortex formation and shedding process.

11.2.1. Phase-reduction analysis

We first apply phase-reduction analysis (Kuramoto 1975; Kuramoto & Nakao 2019) to a general class of time-periodic dynamical systems, given by Eq. (11.1). The phase of the time-periodic solution $\boldsymbol{q}$ of the unforced system, with frequency ω_n, is given by

$$\dot{\theta}(t) = \omega_n, \quad \theta \in [-\pi, \pi], \tag{11.3}$$

where θ is the phase of the limit-cycle oscillation. The single scalar phase variable θ describes the oscillatory behavior of $\boldsymbol{q}$. Solutions $\boldsymbol{q}$ near the limit cycle will all have frequency ω_n, although they will have different phases, given by the scalar field $\Theta(\boldsymbol{q})$, with dynamics given by

$$\dot{\Theta}(\boldsymbol{q}) = \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} = \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q}) \cdot \boldsymbol{N}(\boldsymbol{q}) = \omega_n. \tag{11.4}$$

The level sets of the phase function $\Theta(\boldsymbol{q})$ constitute the isochron coordinates, which are directly perpendicular to $\nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q})$ (Guckenheimer 1975).

To deduce the phase-reduction of the full system given by Eq. (11.1), we consider the introduction of small perturbations to the system at characteristic phases θ such that $\boldsymbol{f}(\boldsymbol{x}, t) = \epsilon \boldsymbol{I}(\boldsymbol{x}, t) = \epsilon \boldsymbol{h}(\boldsymbol{x})\delta(t - t_0)$, where $\epsilon \ll 1$, $\boldsymbol{h}(\boldsymbol{x})$ is a spatial profile with $\|\boldsymbol{h}\| = 1$,

DISTRIBUTION A: Distribution approved for public release.

and δ is a unit-area delta function centered at the time $t_0 \in [0, T]$ when $\boldsymbol{q}$ is at phase θ. Ignoring the higher-order effects, we obtain

$$\dot{\Theta}(\boldsymbol{q}) = \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q}) \cdot \dot{\boldsymbol{q}} = \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q}) \cdot [\boldsymbol{N}(\boldsymbol{q}) + \boldsymbol{f}] = \omega_n + \underbrace{\epsilon \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q}) \cdot \boldsymbol{h}}_{g(\theta,\epsilon)} \delta(t - t_0). \tag{11.5}$$

We can measure the asymptotic phase shift along the limit cycle for each perturbed initial phase θ to compute the phase sensitivity function $\boldsymbol{Z}(\theta)$ as

$$\boldsymbol{Z}(\theta) = \lim_{\epsilon \to 0} \frac{g(\theta, \epsilon)}{\epsilon} = \nabla_{\boldsymbol{q}}\Theta(\boldsymbol{q})|_{\theta} \cdot \boldsymbol{h}. \tag{11.6}$$

Here, $g(\theta, \epsilon)$ is the difference in the phase of baseline and perturbed trajectories at steady-state. Note that there are alternative approaches to compute the phase sensitivity function $\boldsymbol{Z}(\theta)$. One approach is to solve an associated adjoint problem to Eq. (11.5) with Malkin's normalization condition (Brown *et al.* 2004) but requires the explicit formulation of an adjoint system of equations. Global isochrons can also be computed using the method of Fourier averages (Mauroy & Mezić 2012, 2018).

Once the phase sensitivity function is determined, the phase dynamics of the dynamical system can be approximated linearly through the phase-reduced model

$$\dot{\theta}(t) = \omega_n + \boldsymbol{Z}(\theta) \cdot \boldsymbol{f} = \omega_n + \epsilon \boldsymbol{Z}(\theta) \cdot \boldsymbol{h} \delta(t - t_0). \tag{11.7}$$

Now, we apply this phase-reduction analysis to high-dimensional dynamical systems governed by Eq. (11.1) (Taira & Nakao 2018). Using the phase-reduction analysis described above, we can simplify the high-dimensional dynamics associated with periodic flows by introducing a phase coordinate θ that parameterizes the periodic behavior of $\boldsymbol{q}(\boldsymbol{x}, t)$. Thus, the high-dimensional dynamics may be mapped to a low-dimensional representation of a phase coordinate θ via a phase function $\Theta(\boldsymbol{q})$. To evaluate $\boldsymbol{Z}(\theta)$ given an actuation strategy with spatial profile $\boldsymbol{h}(\boldsymbol{x})$, we introduce small impulsive perturbations to the flow of the form,

$$\boldsymbol{I}(\boldsymbol{x}, t) := \boldsymbol{h}(\boldsymbol{x})\delta(t - t_0) = \boldsymbol{h}(\boldsymbol{x}) \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2} \left(\frac{t - t_0}{\sigma} \right)^2 \right], \tag{11.8}$$

where $\boldsymbol{h}(\boldsymbol{x})$ is the spatial profile of perturbation, and t_0 corresponds to the time at a particular phase of the flow $\theta(t_0)$ when the impulse is introduced. For all cases considered, $\sigma = 10\triangle t$, where $\triangle t$ is the numerical time step. The spatial profile depend on the type of actuation considered.

Tracking these added perturbations to the high-dimensional flow system, we obtain the infinitesimal phase sensitivity function $\boldsymbol{Z}(\theta)$ in Eq. (11.6), evaluated at each perturbed initial phase. The phase sensitivity function characterizes the effect of external perturbations on the phase of the periodic limit-cycle orbit. Instead of tracking the entire high-dimensional flow field ($\boldsymbol{q}$), time-series data from sensor measurements can be used. In the case of unsteady time-periodic bluff body flows, we utilize the lift coefficient (C_L) to provide an accurate representation of the phase characteristics of the flow (Williams *et al.* 2009; Loiseau *et al.* 2018; Taira & Nakao 2018). The lift and drag coefficients are defined as

$$C_L = \frac{F_{\text{lift}}}{\frac{1}{2}\rho U_\infty^2 l}, \quad C_D = \frac{F_{\text{drag}}}{\frac{1}{2}\rho U_\infty^2 l} \tag{11.9}$$

where F_{lift} and F_{drag} are the lift and drag forces on the body, ρ is the density, U_∞ is the free stream velocity and l is the characteristic length scale. Once, the phase-sensitivity

DISTRIBUTION A: Distribution approved for public release.

function is obtained, we can formulate the reduced-order model in Eq. (11.7), which describes the phase evolution of infinitesimal perturbations on the flow.

A summary of the phase reduction analysis is outlined in Figure 68 (top panel). First, impulse response simulations are performed by introducing perturbations to the flow through our actuators. We explore several actuation strategies, including blowing/suction, rotation and pitching of the body. For illustration, we show one such strategy using blowing/suction actuation for the cylinder flow in Figure 68(a). Once the actuation strategy is determined, the evolution of the lift coefficient is tracked in the $\dot{C}_L$ - C_L plane, as in Figure 68(b). The phase coordinate θ can be easily defined in this plane. We also note that other sensor measurements that appropriately represent the oscillatory behavior of the system can also be used to define phase. We show the baseline limit cycle trajectories (in black) as well as the perturbed trajectories in blue. The impulsive perturbation is introduced at θ_i. We measure the asymptotic phase shift $g(\theta, \epsilon)$ by comparing the baseline and perturbed lift trajectories. Repeating the impulse response analysis for a number of θ, we can compute the phase sensitivity function using Eq. (11.6), as seen in Figure 68(c). As this method only requires the perturbed trajectories at steady-state compared to the baseline, the phase sensitivity function can be obtained in experiments as well (Williams *et al.* 2009). For $\boldsymbol{Z}(\theta) < 0$, phase advance in the perturbed trajectories is observed, i.e., the phase of the perturbed trajectories leading the baseline flow trajectories. For $\boldsymbol{Z}(\theta) > 0$, phase delay is observed with the perturbed trajectories lagging the baseline trajectories in phase.

11.2.2. Energy-optimal phase control

In the previous section, we derived a reduced-order model to describe the phase dynamics of high-dimensional unsteady fluid flows. In this section, we manipulate these phase dynamics to advance or delay the phase of flow oscillation. We emphasize that the present control strategy aims to rapidly change the phase of the flow oscillations within a time scale commensurate with the characteristic time of flow oscillation T, rather than asymptotic phase control.

Our specific goal is to determine the optimal forcing input $\boldsymbol{f}(\boldsymbol{x}, t)$ to shift the phase of oscillation by a desired amount $\triangle\theta$ within a single period of oscillation T. This objective may be formulated as modifying the period of the first oscillation to be $T^* = T + \triangle\theta/\omega_n$. After this first period, we turn off the control, and the flow returns to its original period, but with a fixed phase difference $\triangle\theta$. The phase shift is given by $\triangle\theta = \omega_n(T^* - T)$, so a phase advance corresponds to $T^* < T$ and a phase delay corresponds to $T^* > T$.

The control signal will use the same actuation strategy as the perturbations in Eq. (11.8), with spatial forcing $\boldsymbol{h}$, but with a non-impulsive control magnitude $u(t)$: The forcing input added to the unsteady fluid flow is given by

$$\boldsymbol{f}(\boldsymbol{x}, t) = \begin{cases} \boldsymbol{h}(\boldsymbol{x})u(t) & \text{if } 0 \leqslant t < T^* \\ 0 & \text{if } t \geqslant T^*. \end{cases} \tag{11.10}$$

Here, the control signal $u(t)$ can take arbitrary shape as opposed to the impulsive perturbation considered above. Moreover, we are interested in the control signal $u(t)$ that minimizes the overall energy input to the high-dimensional fluid flow system, while achieving the desired phase control. For a specified time T^*, and for all control inputs $u(t)$ which evolve from $\theta(0) = 0$ to $\theta(T^*) = 2\pi$, we seek the control signal that minimizes a cost function (Moehlis *et al.* 2006) of the form,

$$\mathcal{C}[u(t)] = \int_0^{T^*} \left\{ [u(t)]^2 + \lambda(t)[\dot{\theta}(t) - \omega_n - Z(\theta)u(t)] \right\} dt, \tag{11.11}$$

DISTRIBUTION A: Distribution approved for public release.

where $\lambda(t)$ is the Lagrange multiplier that enforces the system dynamics $\dot{\theta}(t) = \omega_n + Z(\theta)u(t)$. The phase delay $\triangle\theta$ is automatically enforced by the boundary conditions.

To minimize the cost function $\mathcal{C}[u(t)]$, we apply the calculus of variations (Moehlis *et al.* 2006; Monga & Moehlis 2019; Monga *et al.* 2019) and solve the associated Euler–Lagrange equations given by

$$u(t) = \frac{\lambda(t)Z(\theta(t))}{2}, \quad \dot{\theta} = \omega_n + \frac{\lambda(t)[Z(\theta)]^2}{2}, \quad \dot{\lambda} = -\frac{[\lambda(t)]^2 Z(\theta)}{2}\frac{d}{d\theta}Z(\theta) \tag{11.12}$$

subject to initial and final conditions for phase, i.e., $\theta(0) = 0$ and $\theta(T^*) = 2\pi$. This constitutes a two-point boundary value problem in the phase coordinate $\theta(t)$. Given the phase sensitivity function $Z(\theta)$ and its gradient $dZ(\theta)/d\theta$, Eq. (11.12) can be solved to deduce the optimal control input $u(t)$ for a desired phase shift $\triangle\theta$. If λ is approximately constant, the control strategy is directly proportional to the phase sensitivity function, which is related to opposition control (Siegel *et al.* 2003; Gerhard *et al.* 2003).

In addition to the initial condition for phase, we also need the initial condition for the Lagrange multiplier $\lambda(0)$ to solve Eq. (11.12). The appropriate initial condition can be found by a shooting method. Using the converged initial conditions, the optimal control signal $u(t)$ can be deduced. We can then compute the optimal forcing input given by Eq. (11.10) which is subsequently added to the Navier-Stokes equation in Eq. (11.1) to achieve the desired phase shift.

A summary of this phase control strategy is shown in Figure 68 (bottom panel). Here, for a desired phase shift $\triangle\theta$, the phase sensitivity function and its gradient are utilized to deduce the optimal control input $u(t)$ minimizing the cost function $\mathcal{C}[u(t)]$, as shown in Figure 68(d). The control forcing is then added to the Navier-Stokes equation as shown in Figure 68(e) after which the transient and steady-state control performance is evaluated, as in Figure 68(f).

An alternative to the phase control strategy described above is to use model predictive control (MPC) (Garcia *et al.* 1989; Morari & Lee 1999) based on the phase response model. Instead of solving the Euler–Lagrange equations, an optimal control problem can be solved in real-time over a receding horizon subject to the imposed system constraints. The procedure for MPC-based phase-optimal control is described in the appendix. Using a simple example, we aim to show the connections between the feedback law derived using MPC and that of the energy-optimal phase control strategy described earlier in this section. The energy-optimal control strategy provides us with an intuition regarding the nature of the optimal control input by directly relating it to the phase sensitivity function. MPC does not provide such an intuitive connection, and obtaining an effective control law requires tuning of state and input weights. However, noise robustness and the ability to include constraints make MPC an attractive alternative strategy for phase control of fluid flows. This robustness is especially important for systems that have stochastic disturbances and slowly varying parameters, as MPC is able to correct for model uncertainty through sensor feedback. Throughout this work, we primarily use the optimal control design based on the Euler–Lagrange equations, due to its simplicity and effectiveness. This approach is also compared to MPC on a simple example in the next section, showing that performance is comparable.

11.3. Results

We demonstrate the phase-based flow control strategy on three example systems as summarized in Figure 70. First, we investigate a Stuart–Landau oscillator, where the phase sensitivity function can be computed analytically. We then demonstrate the phase reduction analysis and optimal control on a canonical flow over a cylinder. Here,

DISTRIBUTION A: Distribution approved for public release.

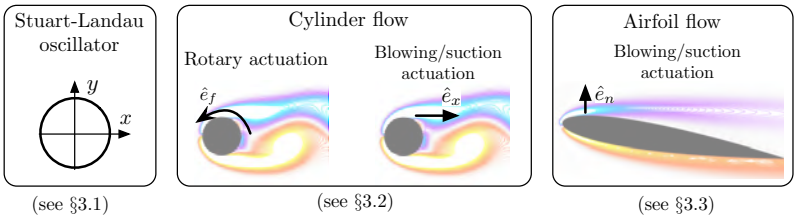


Figure 48: Summary of the examples considered in this work for optimal phase control: (a) Stuart-Landau oscillator, (b) unsteady flow over a cylinder at $Re = 100$ with rotary and blowing/suction (x-direction) actuation, (c) unsteady two-dimensional flow over an airfoil at $Re = 1000$ and angle of attack of $\alpha = 9^\circ$ with blowing/suction actuation.

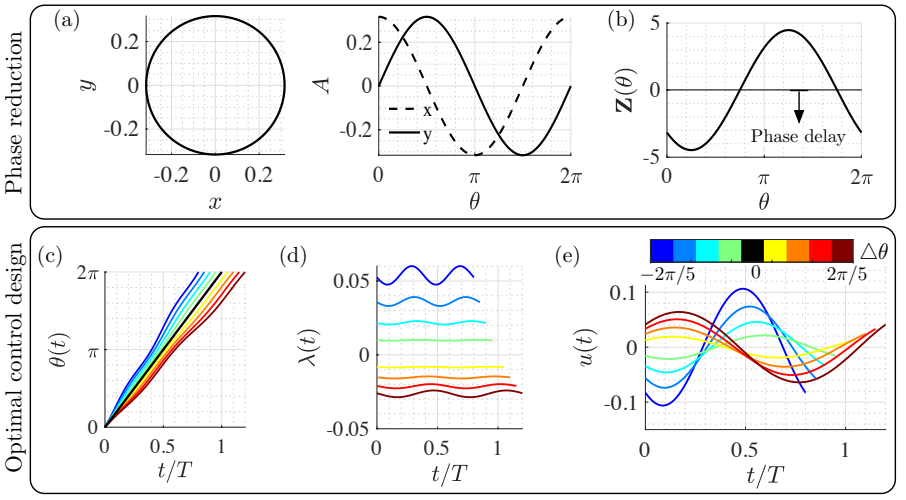


Figure 49: Overview of phase-reduction and optimal phase control for a Stuart-Landau oscillator: (a) Baseline trajectories; (b) analytical phase-sensitivity function and optimal control design for desired phase shifts of $-2\pi/5 \leqslant \triangle\theta \leqslant 2\pi/5$; (c) phase coordinate dynamics; (d) dynamics of the Lagrange multiplier; and (e) the optimal control input from the Euler–Lagrange equations.

we explore two actuation strategies, one using rotary actuation and the other using momentum injection. Finally, we implement optimal phase control on a two-dimensional flow past an airfoil using blowing/suction actuation.

11.3.1. Stuart–Landau equation

We begin with a simple motivating example, given by the Stuart-Landau (SL) oscillator

$$\frac{\mathrm{d}A}{\mathrm{d}t} = a_0 A - a_1 A|A|^2 + \sigma\xi(t) + \mathrm{Re}[f(t)], \tag{11.13}$$

where $A(t) = x(t) + iy(t)$ is the complex amplitude, $f(t)$ is the forcing, and $\xi(t) = \xi_x(t) + i\xi_y(t)$ is independent and identically distributed Gaussian white noise with unit variance and strength σ. For $\sigma > 0$, the SL equation exhibits stochastic dynamics. We use a stochastic Runga–Kutta solver to integrate the dynamics (Kasdin 1995).

The Stuart-Landau equation was initially used to study the nonlinear instability of plane Poiseuille flow (Stuart 1960) and was later extended for weakly nonlinear stability

DISTRIBUTION A: Distribution approved for public release.

analysis of cylinder wakes and open cavity flows (Sipp & Lebedev 2007). Bagheri (2013) showed that the leading Koopman eigenfunctions of the cylinder flow can be constructed using these coordinates.

In Cartesian coordinates, we obtain the well-known Hopf normal form, given by

$$\frac{dx}{dt} = \mu x - \omega y - (x^2 + y^2)(x - \beta y) + \sigma\xi_x(t) + f(t), \tag{11.14a}$$

$$\frac{dy}{dt} = \omega y + \mu x - (x^2 + y^2)(\beta x + y) + \sigma\xi_y(t), \tag{11.14b}$$

where, $a_0 = \mu + i\omega$ and $a_1 = 1 + i\beta$. Here, $f(t)$ is the forcing added in the x-direction (for phase control). We set $\mu = 0.1, \omega = 1$ and $\beta = 1$. The time step for numerical integration is $\triangle t = 0.01$. The radial amplitude is given by $r = \sqrt{x^2 + y^2}$. For the case with no noise, the radial amplitude is constant, $r(t) = r_0$, with a period of oscillation $T = 6.98$. The phase coordinate on the limit-cycle is $\theta = \tan^{-1}(y/x)$. The phase sensitivity function for forcing in the x-direction can be constructed analytically (Moehlis 2014) as

$$Z(\theta) = \frac{\beta}{\sqrt{\mu}}\cos\theta - \frac{1}{\sqrt{\mu}}\sin\theta. \tag{11.15}$$

This phase-sensitivity function only depends on the system parameters and the phase coordinate.

The baseline trajectories and the phase sensitivity function for this example are shown in Figure 71. A maximum phase delay is observed around $\theta = \pi/5$, while a maximum phase advance is observed around $\theta = 6\pi/5$. We can also deduce its gradient analytically from Eq. (11.15). With this information, we are able to design an optimal phase controller, as described in §11.2.2.

We consider a range of phase shifts so that $-2\pi/5 \leqslant \triangle\theta \leqslant 2\pi/5$. A phase advance of $\triangle\theta = -2\pi/5$ corresponds to $T^*/T = 0.8$ and a phase delay of $\triangle\theta = 2\pi/5$ corresponds to $T^*/T = 1.2$. The phase coordinate $\theta(t)$, corresponding Lagrange multiplier $\lambda(t)$, and the optimal control input $u(t)$ obtained by solving the associated Euler–Lagrange equations in Eq. (11.12) are shown in Figure 71(c), (d) and (e), respectively. The color of the lines correspond to the varying values of desired phase shift. As described before, the initial condition $\lambda(0)$ is obtained by performing a shooting method. The converged initial conditions fall in the range $-0.026 \leqslant \lambda(0) \leqslant 0.0525$. For phase advance (negative phase shifts), $\lambda(t) > 0$, while for phase delays (positive phase shift), $\lambda(t) < 0$. The baseline phase is shown by the solid black line.

We also notice the correspondence between the phase sensitivity function and the optimal control inputs $u(t)$ in Figure 71(b) and (e), respectively. The shape of the control input closely follows that of the phase sensitivity function, especially for phase advance (negative phase shifts). For phase delay (positive phase shift), the control input follows the opposite trend of the phase sensitivity function. This is intuitive as the control input opposes any phase advances in the model by adding an opposite control force.

We implement the optimal control law $f(t) = u(t)$ in the SL system in Eq. (11.14). Control simulations are performed with no noise ($\sigma = 0$) and with stochastic dynamics ($\sigma = 0.05$) for extreme phase shifts of $\triangle\theta = \pm 2\pi/5$. The control input $u(t)$, control trajectories $y(t)$, and the radial amplitude $r(t) = \sqrt{x^2 + y^2}$ for the baseline SL oscillator and the stochastic SL oscillator are shown in Figures 72(a) and (b), respectively. For the baseline SL oscillator, the prescribed phase shift is achieved in the immediate transient (within T^*) for both the control cases corresponding to $\triangle\theta = \pm 2\pi/5$. This phase shift is also observed in the asymptotic behavior. For the stochastic system, shown in Figure 72(b), the prescribed phase shifts are consistently achieved with control. This

DISTRIBUTION A: Distribution approved for public release.

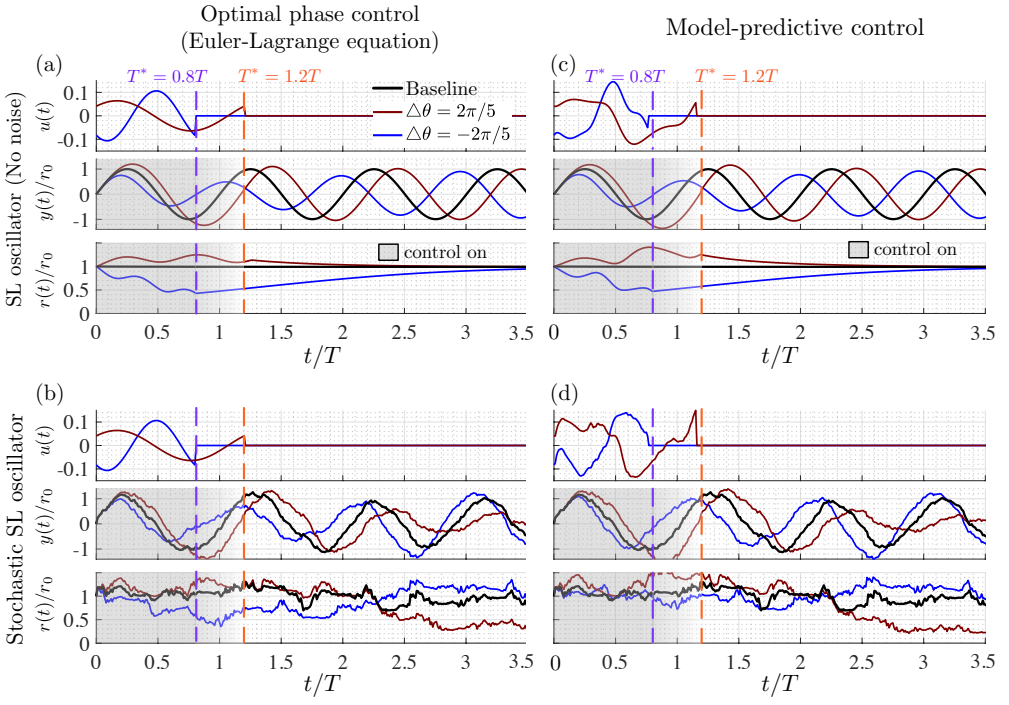


Figure 50: Optimal phase control (using Euler–Lagrange equation) of Stuart–Landau oscillator with (a) no noise ($\sigma = 0$) and (b) stochastic dynamics $\sigma = 0.05$. Model-predictive control of Stuart-Landau oscillator with (c) no noise ($\sigma = 0$) and (d) stochastic dynamics $\sigma = 0.05$. For each case, the control input $u(t)$, trajectory $y(t)$ and radial amplitude $r(t)$ are shown.

demonstrates the robustness of the control technique to stochastic disturbances. However, the phase shifts are not maintained asymptotically after the control is switched off. When the control is on, along with the phase shift, we also notice an amplitude modulation in the trajectories of $y(t)$. For phase delay, an increase in the amplitude is observed while for phase advance, a decrease in the amplitude is observed. These changes in the amplitude by performing optimal phase control are clearly visible in $r(t)$ as well.

Additionally, we perform real-time phase control using MPC, as discussed in the Appendix. The prediction and control horizons are fixed at $n_c = n_p = 8$, and we also fix $Q = 5, R = 1, R_{\triangle u} = 100$. The results using MPC are shown in Figure 72(c) and (d). The MPC and energy-optimal results are remarkably similar. The variation in the control inputs follow similar trends for both control cases and responses are almost identical.

The phase optimal control strategy not only helps to advance and delay phase, but it also appropriately modulates the amplitude characteristics. In the following examples, we will see the implications of optimal phase control applied to unsteady fluid flows.

11.3.2. **Laminar flow over a cylinder**

We now demonstrate our phase control approach on the canonical two-dimensional, incompressible flow past a circular cylinder (Noack *et al.* 2003*b*) at Reynolds number of $Re = U_\infty d/\nu = 100$, where U_∞ is the free stream velocity, d is the cylinder diameter and ν is the kinematic viscosity. Direct numerical simulations are performed using the immersed boundary projection method (Colonius & Taira 2008*a*; Kajishima & Taira 2017) with

a computational grid extending over $(x/d, y/d) \in [-31, 33] \times [-32, 32]$. A multi-domain technique is utilized with smallest spatial resolution of $\triangle x/d = \triangle y/d = 0.02$. The simulation time step is $\triangle t = 0.005$. Details of the simulation and validation can be found in Taira & Colonius (2007*a*).

The baseline flow exhibits periodic vortex shedding in the cylinder wake with shedding frequency of $\omega_n = 2\pi f_n U_\infty/d = 1.03$ (characteristic time $T = 2\pi/\omega_n = 6.1$). In this flow, we will explore two different actuation strategies, first using rotary actuation (§ 11.3.2) and then using momentum injection (§ 11.3.2). Before we introduce perturbations for phase-reduction analysis or forcing input for phase-based control, we need to develop the actuation strategies.

Rotary actuation

In this section, we achieve phase control of the unsteady cylinder flow by rotating the cylinder as actuation. Unlike the previous example of the SL oscillator, the phase-sensitivity function for this high-dimensional fluid flow cannot be deduced analytically. We follow the phase-reduction approach outlined in §11.2.1 to construct the phase sensitivity function. To introduce perturbations, we impulsively rotate the cylinder in the counter-clockwise direction, as in Eq. (11.8).

For reference, we show the baseline lift fluctuation over a period of oscillation in Figure 74(a), which resembles the baseline trajectory of the Stuart–Landau oscillator. Multiple direct numerical simulations are performed by introducing weak rotary perturbations ($\epsilon = 0.1$) at various phases θ. We then compute the asymptotic phase difference between the perturbed lift with respect to the baseline after 15 periods of oscillation, resulting in the phase sensitivity function $\boldsymbol{Z}(\theta)$ using Eq. (11.6). We show the phase sensitivity function for perturbation size of $\epsilon = 0.1$ in Figure 74(b). For verification, we perform a similar analysis with a perturbation size of $\epsilon = 0.2$, and both sensitivity functions are nearly identical. Generally, phase delay is observed for phases corresponding to $C_L > 0$ and phase advance is observed for $C_L < 0$. The gradient of phase sensitivity $d\boldsymbol{Z}(\theta)/d\theta$ is shown in Figure 74(c). Larger gradients are observed at zero crossings of the lift curve.

Using the phase sensitivity function and its gradient, we are equipped to compute the optimal control input $u(t)$ for phase alteration of unsteady flow oscillations. We consider a range of desired phase shifts $-2\pi/5 \leqslant \triangle\theta \leqslant 2\pi/5$, which correspond to $0.8T \leqslant T^* \leqslant 1.2T$. Solving the Euler-Lagrange equations in Eq. (11.12), the control inputs for this range of phase shifts are shown in Figure 74(d). As in the previous example, for phase advance (negative phase shift) the control input follows a similar trend to that of the phase sensitivity function, while for phase delay (positive phase shift) the control input follows the opposite trend. For positive phase shift, the control input opposes any phase advance in the sensitivity function. The inset in Figure 74(d) shows the cost function $\mathcal{C}[u(t)]$ from Eq. (11.11) corresponding to the different phase shifts. As expected, lower cost is associated with smaller phase shifts. The cost for negative phase shifts is higher than for positive phase shifts, as the magnitude of input is larger.

With the control input, we can determine the corresponding forcing to add to the Navier-Stokes equations, given by Eq. (11.10). We perform direct numerical simulations with the optimal forcing for each case, and the control forcing is switched off after $t = T^*$. The lift coefficients for the controlled simulations are shown in Figure 74(e). The baseline lift coefficient is shown in black. For all control cases, immediately after the application of control, the desired phase shift is achieved. Thus, the flow immediately locks on to the frequency prescribed within characteristic time (one period). In previous studies, the short-time Fourier transform and a Kalman filter were used to phase-lock the lift fluctuations with the control signal within 2-3 periods of oscillation (Joe *et al.* 2011), while

DISTRIBUTION A: Distribution approved for public release.

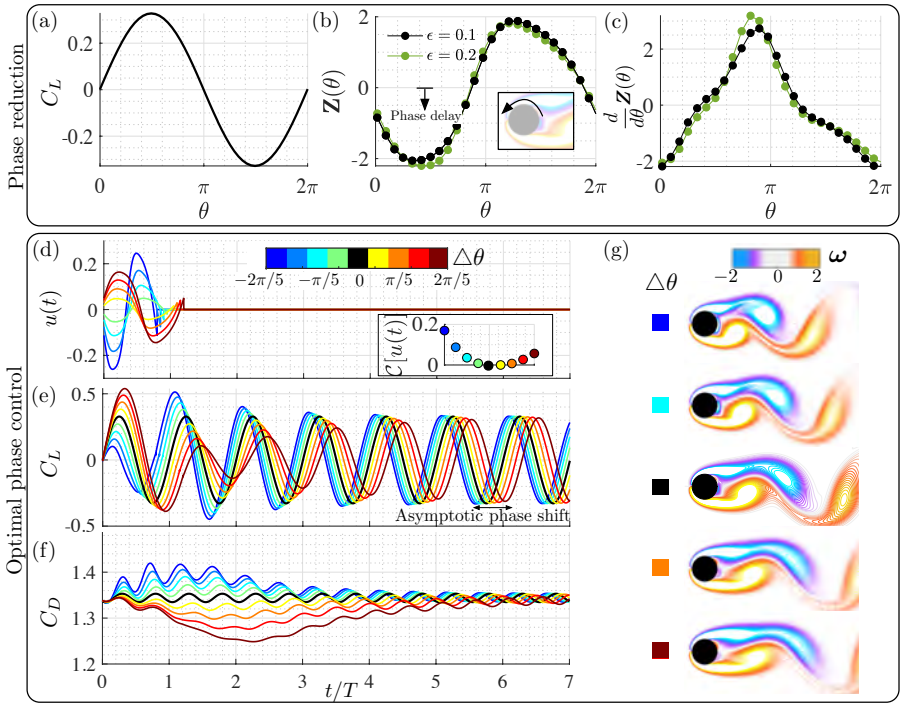


Figure 51: Overview of phase-reduction and optimal phase control for cylinder flow using **rotary actuation**: (a) Lift coefficient (C_L) of baseline cylinder flow with respect to the phase coordinate θ, (b) phase sensitivity function and (c) its gradient (right) for impulse perturbation introduced at an actuator located near the top separation point in the $\hat{e}_x$ direction, (d) energy-optimal control input needed to shift the phase of oscillation (with associated cost function in inset), (e) lift and (f) drag characteristics after application of forcing in direct numerical simulations, and (g) instantaneous flow fields immediately after control is turned off ($t = T^*$). The color of the lines correspond to the cases with desired phase shift $\triangle\theta$ with $\triangle\theta = 0$ indicates baseline.

our controller achieves immediate phase-locking. The desired phase shift is sustained asymptotically after the control is switched off.

An important consequence of the phase-optimal control strategy is the effect on the amplitude of lift fluctuations in the immediate transient. We see that for positive phase shifts, the amplitude of lift fluctuation increases immediately after application of control. For the negative phase shift, in the first oscillation cycle, the lift amplitude decreases with control. However, we observe a sharp rise in the lift fluctuation over the baseline in the second oscillation cycle. This increase in amplitude was not observed in the Stuart-Landau oscillator. Along with the lift coefficient, we also show the drag coefficient in Figure 74(f). The drag coefficient decreases with positive phase shifts while it increases with negative phase shifts in the immediate transient. For positive phase shifts, a maximum drag reduction is achieved after the control is switched off. This is evidence that transient phase control is an effective means for both increasing lift and decreasing drag (Amitay & Glezer 2002*a*). Though not shown, the transient phase control may be repeated after the system reaches steady state, about 6 oscillation cycles later.

To understand the flow physics associated with the transient phase control, we analyze

the instantaneous vorticity fields $\boldsymbol{\omega}(\boldsymbol{x}, t)$ at $t = T^*$ for the different phase shifts, shown in Figure 74(g). We can see that for negative phase shifts, the vortex shedding takes place earlier and the length of the shed vortex is shorter. The size of the shed vortex is proportional to the magnitude of the desired phase shift. Only once the vortex is shed, we see an increase in the lift fluctuation in the control cases of negative phase shifts (i.e. after $t = T^*$). Thus, one mechanism of enhancing lift is to shed the vortex earlier.

The other mechanism of increasing lift is to delay the vortex shedding as much as possible, which is seen in the positive phase shift control cases. In these cases, the vortex shedding is delayed, resulting in an elongated vortical structure This stabilizes the recirculation region behind the cylinder and leads to a drag reduction as well. Note that the cost associated with shedding a vortex earlier is slightly larger. We argue that for positive phase shifts, the control principle is similar to that of direct opposition control of vortices in the near wake, as discussed in the work of Siegel *et al.* (2003) and Gerhard *et al.* (2003). In these works, direct opposition control of wake structures with transversal oscillations of the cylinder enable the formation of elongated recirculation bubbles (Pastoor *et al.* 2008). Using a control law that opposes any phase advances in the phase-sensitivity function leads directly to control of vortices in the near wake.

Momentum injection

We now investigate the phase-optimal control technique for another actuation strategy based on momentum injection, where we add a localized force to the Navier-Stokes equation to mimic a blowing/suction type actuator. The spatial profile of the actuator is a Dirac–delta in space, approximated using the three-cell discrete delta function (Roma *et al.* 1999). The actuation is introduced at the average separation point $\boldsymbol{x}_s$ on the upper surface of the cylinder, at an angle of 58.4° from the aft separation point and positioned at $3\triangle x$ from the cylinder surface. The actuation is prescribed in the x-direction. While there is flexibility in choosing the actuation for the proposed control method, some actuation configurations may be more effective for controlling phase (Taira & Nakao 2018; Bhattacharjee *et al.* 2018).

For reference, the baseline lift is shown in Figure 75(a). Again, the phase sensitivity function $\boldsymbol{Z}(\theta)$ is obtained from impulsive perturbations, as shown in Figure 75(b). The analysis is performed for perturbation amplitudes of $\epsilon = 0.01$ and 0.05. The phase sensitivity functions are similar for different perturbation sizes, except near $\theta = 3\pi/5$. Compared to the rotary control case, the phase sensitivity function has a larger magnitude at most characteristic phases. This shows that the flow is more sensitive to momentum injection than rotary control. The gradient of the phase sensitivity function is highest around $\theta = \pi$, as seen in Figure 75(c).

The phase control inputs to achieve the desired phase shifts using momentum injection are shown in Figure 75(d). As with rotary actuation, the cost function is larger for negative phase shifts (shown in inset). However, the cost of momentum injection is an order of magnitude lower. This is expected because of the increased phase sensitivity in the momentum injection case. We can also compute the actuation cost with the steady and unsteady momentum coefficients given by

$$\bar{C}_\mu = \frac{\rho u_{\text{jet}}^2 \sigma}{\frac{1}{2}\rho U_\infty^2 d}, \qquad C'_\mu = \frac{\rho u_{\text{rms}}^2 \sigma}{\frac{1}{2}\rho U_\infty^2 d}, \tag{11.16}$$

respectively, where σ is the actuator width, given by the grid spacing $\sigma = \triangle x$. The characteristic velocities u_{jet} and u_{rms} are computed with companion simulations by setting the freestream velocity to be zero (Munday & Taira 2013). The steady and

DISTRIBUTION A: Distribution approved for public release.

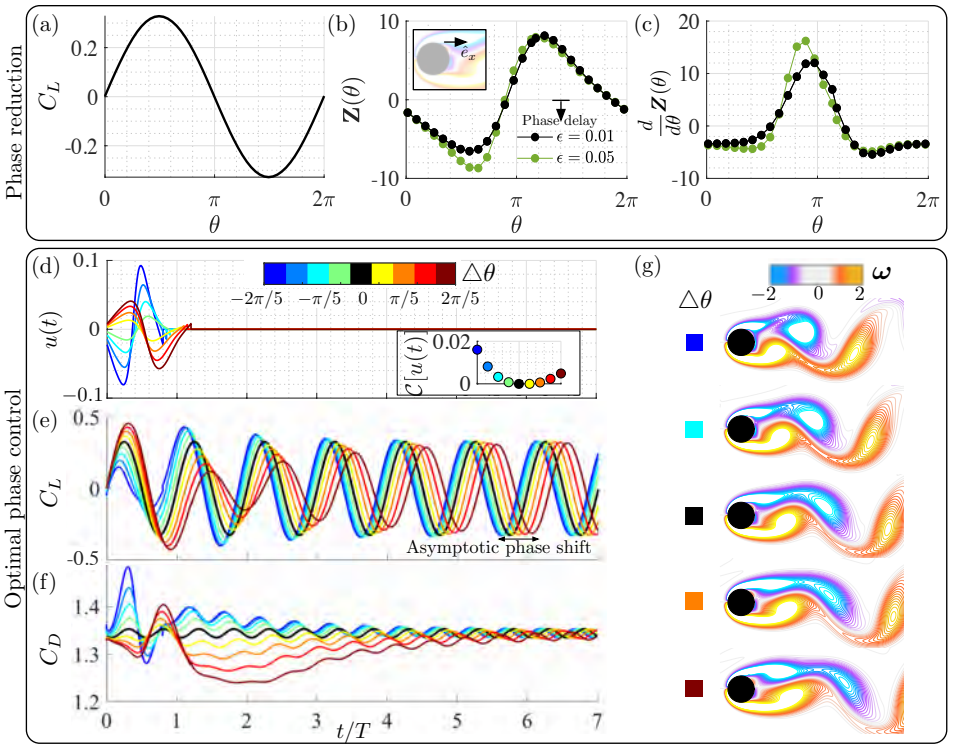


Figure 52: Overview of phase-reduction and optimal phase control for cylinder flow using **momentum injection**: (a) Lift coefficient (C_L) of baseline cylinder flow with respect to the phase coordinate θ, (b) phase sensitivity function and (c) its gradient (right) for impulse perturbation introduced at an actuator located near the top separation point in the $\hat{e}_x$ direction, (d) energy-optimal control input needed to shift the phase of oscillation (with associated cost function in inset), (e) lift and (f) drag characteristics after application of forcing in direct numerical simulations, and (g) instantaneous flow fields immediately after control is turned off ($t = T^*$). The color of the lines correspond to the cases with desired phase shift $\triangle\theta$ with $\triangle\theta = 0$ indicates baseline.

unsteady coefficient of momentum for $\triangle\theta = -2\pi/5$ are $\bar{C}_\mu = 0.0078$ and $C'_\mu = 0.0125$, respectively. For $\triangle\theta = 2\pi/5$, $\bar{C}_\mu = 0.006$ and $C'_\mu = 0.01$.

Incorporating the forcing input in the Navier–Stokes equations, we perform direct numerical simulations for the varying desired phase shifts. The time history of the lift and drag coefficients in the controlled simulations are shown in Figure 75(e) and (f), respectively. We see similar control performance with respect to the transient phase shift, lift fluctuation, and asymptotic phase shift compared to the rotary actuation case. However, the performance in drag differs slightly. When the control in on, the drag both increases and decreases with respect to the baseline. When the control is off, the drag coefficient decreases for positive phase shift and increases slightly for negative phase shifts. This is consistent with the results from rotary actuation. The (controlled) instantaneous vorticity fields at $t = T^*$ are shown in Figure 75(g). The characteristic flow physics for the control based on momentum injection is almost identical to that of rotary actuation. This shows the consistency of the optimal phase control approach in terms of altering the vortex shedding behavior for different actuation strategies.

We now compare key features of interest for both rotary actuation and momentum injection. In particular, we investigate the maximum lift coefficient, lift impulse, and the

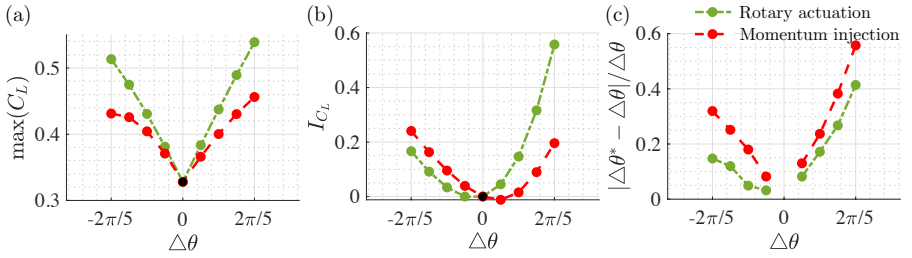


Figure 53: Comparison of control performance for rotary actuation and momentum injection: (a) Maximum lift coefficient, (b) lift impulse and (c) error in asymptotic phase shift for varying desired phase shifts.

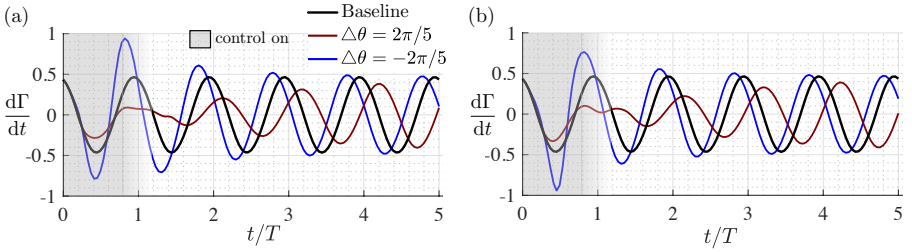


Figure 54: Vorticity flux comparison with respect to the baseline for (a) rotary control and (b) momentum injection.

error in asymptotic phase shift with phase optimal control, as shown in Figure 76(a), (b) and (c), respectively. Lift impulse is the area under the lift-time curve (Williams *et al.* 2009), demonstrating the overall benefit of control in terms of lift enhancement. With an increase in the phase shift, the maximum lift increases. With larger phase shifts, the benefit in lift enhancement increases dramatically. However, because of the considerable amplitude variation, the error in asymptotic phase shift increases with phase shift. This is expected as the optimal control input is based on infinitesimal phase sensitivity functions and larger control amplitudes introduce nonlinear effects. Note that there is negligible error in the immediate transient phase shift. The phase control performance of rotary control is superior to that of momentum injection from an aerodynamic view point. However, as mentioned before, the cost associated with rotary control is an order of magnitude higher.

Amitay & Glezer (2002*a*, 2006*b*) analyzed transients associated with flow reattachment and separation using pulsed amplitude modulation of a synthetic jet actuator. They observed that shedding of large vortices and momentary increase in circulation were crucial in terms of achieving the transient flow attachment. These enhancements were accentuated with actuation frequency close to the natural frequency of the flow (Amitay & Glezer 2006*b*; Colonius & Williams 2011). We similarly investigate flow transients in terms of the rate of change of circulation to characterize the effect of control given by

$$\frac{\mathrm{d}\Gamma}{\mathrm{d}t} = \int_{x/d=\mathrm{const}} \omega(\boldsymbol{x},t)\boldsymbol{q}(\boldsymbol{x},t)\cdot\hat{\boldsymbol{e}}_x \mathrm{d}y \tag{11.17}$$

where $\boldsymbol{q}$ is the velocity field and ω is the vorticity. We compute the vorticity flux at $x/d = 1.2$. The rate of change of circulation in time, for both rotary actuation and

momentum injection, are shown in Figure 77(a) and (b) for extreme phase shifts. The trends are similar to the variation in the lift coefficient. The instantaneous vorticity flux for positive phase shifts is slightly greater than that of the baseline in the initial oscillation leading to an immediate increase in lift coefficient. The time-rate of change of circulation increases dramatically towards the end of the control forcing for the negative phase shift. This time corresponds to the vortex pinch-off, i.e., when the vortex is shed. The increase in flux is larger for the rotary actuation, leading to higher control performance compared to that of momentum injection.

11.3.3. **Flow over an airfoil**

Finally, we demonstrate the optimal phase control approach on a two-dimensional, incompressible flow over a NACA0012 airfoil at an angle of attack $\alpha = 9^\circ$ with chord-based Reynolds number $Re = 1000$. This example is chosen to demonstrate the advanced capabilities of the phase-based control approach to enhance aerodynamic characteristics of wings in unsteady flow conditions.

The immersed-boundary projection method is used for direct numerical simulation (Colonius & Taira 2008*a*). Five nested domains are used, with an outermost domain of $-16 \leqslant x/c \leqslant 16, -16 \leqslant y/c \leqslant 16$ and an inner grid resolution of $\triangle x = 0.0055$, where c is the chord length. The integration time step is $\triangle t = 0.001$. The baseline lift coefficient fluctuation is shown in Figure 79(a). The mean lift force is $\bar{C}_L = 0.394$ and the oscillation frequency is $\omega_n = 2\pi f_n U/c = 7.17$, corresponding to a characteristic time of $T = 2\pi/\omega_n$. There are three fundamental differences in this high-dimensional flow example compared to the previous cylinder flow example: (i) the baseline mean lift force is nonzero, (ii) the harmonics of the natural shedding frequency are stronger and can be observed in the baseline lift oscillation, and (iii) the flow exhibits both leading-edge and trailing-edge separation.

For phase-reduction and phase optimal control, we use wall-normal momentum injection, modeled by a localized force added near the average separation point. We then perform the impulse-response analysis at 25 characteristic phases θ. The difference between the perturbed trajectories and the baseline fluctuations after 20 time periods of oscillation is assessed to compute the phase sensitivity function, as shown in Figure 79(b). After introducing the perturbation, the flow takes more oscillations to reach steady state compared to the cylinder flow. The amplitude of the perturbation is $\epsilon = 0.0005$. The magnitude of the phase sensitivity function is larger compared to both actuation cases for the cylinder flow example. The maximum phase advance sensitivity is observed at the phase corresponding to minimum lift coefficient. Phase delay sensitivity is largest at phases corresponding to larger lift coefficient. Surprisingly, the phase sensitivity function has a sinusoidal shape, even for this highly complex flow. The corresponding gradient is shown in Figure 79(c). The optimal control inputs for a desired phase shift $-2\pi/5 \leqslant \triangle\theta \leqslant 2\pi/5$ are shown in Figure 79(d). The associated cost of control (shown in inset) is lower than the corresponding cost of momentum injection for the cylinder flow example. This is expected as the magnitude of the phase sensitivity function for the airfoil flow is larger. The steady and unsteady coefficient of momentum for $\triangle\theta = -2\pi/5$ are $\bar{C}_\mu = 0.01$ and $C'_\mu = 0.0132$, respectively. For $\triangle\theta = 2\pi/5$, $\bar{C}_\mu = 0.01$ and $C'_\mu = 0.01$.

After applying the optimal forcing input to the Navier–Stokes equations and performing direct numerical simulations, we examine the fluctuations in the lift coefficient compared to the baseline for all cases in Figure 79(e). To highlight the variation with respect to the baseline, we separate the lift fluctuation for negative phase shifts and positive phase shift. We also color the increments in the fluctuation to further emphasize the changes with control. Similar to the cylinder flow example, we achieve an immediate

DISTRIBUTION A: Distribution approved for public release.

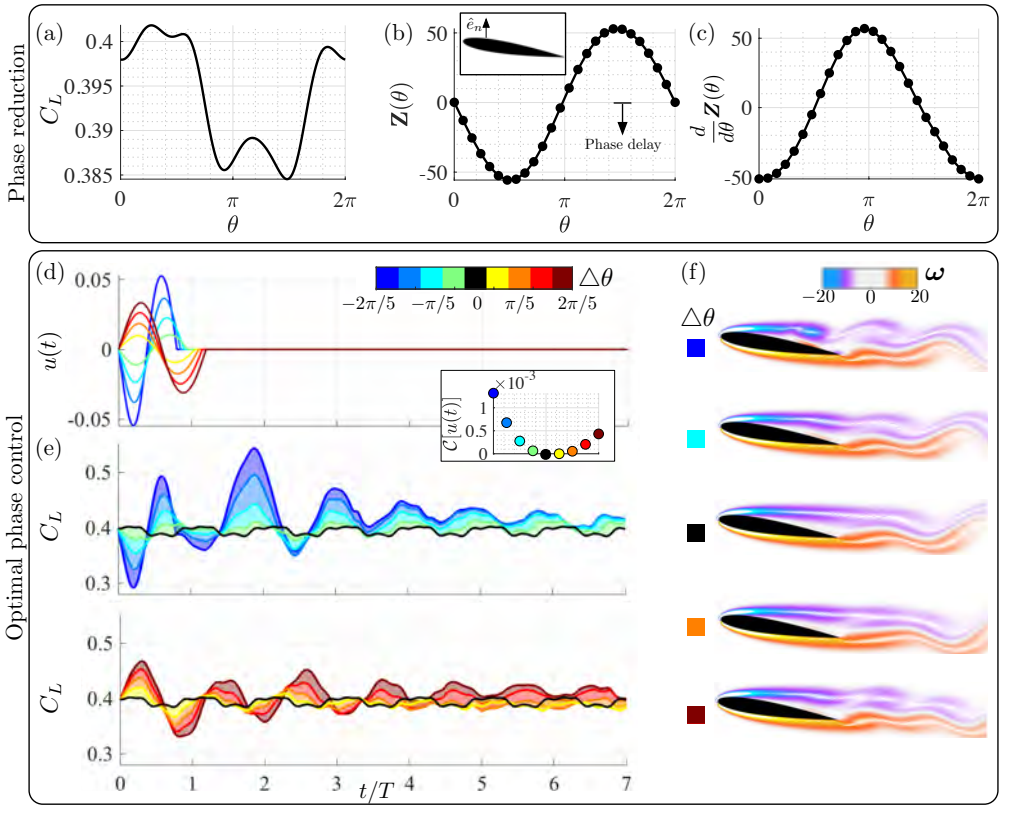


Figure 55: Overview of phase-reduction and optimal phase control for flow over an airfoil using momentum injection: (a) Lift coefficient (C_L) of baseline cylinder flow with respect to the phase coordinate θ, (b) phase sensitivity function and (c) its gradient (right) for impulse perturbation introduced at an actuator located near the top separation point in the $\hat{e}_x$ direction, (d) energy-optimal control input needed to shift the phase of oscillation (with associated cost function in inset), (e) lift characteristics after application of forcing in direct numerical simulations for negative and positive phase shifts (shown separately), and (f) instantaneous flow fields after control is turned off ($t = 2T^*$). The color of the lines correspond to the cases with desired phase shift $\triangle\theta$ with $\triangle\theta = 0$ indicates baseline. For visual clarity, the fluctuation in the controlled lift trajectories are shown as shaded areas relative to the baseline lift coefficient.

increase in lift for the positive phase shift cases. Immediately after control is switched off, we see a rapid increase in lift fluctuation for the negative phase shift cases. The transient phase shift is exactly as prescribed, indicating that immediate lock-on is obtained even for this complex flow. We also see that lift fluctuations are higher than the baseline for all cases for a number of oscillation cycles, even after the control is switched off. Note that although there are distinct harmonic oscillations in the baseline lift, when control is applied, these harmonics are suppressed until downstream. The work of Joe *et al.* (Joe *et al.* 2011) on an inclined flat plate showed that larger ranges of negative phase shift in the control signal contributed to lift enhancement, while sharp decrease in enhancement at positive phase shift regimes were observed. Although there is a preference for negative phase shifts for larger lift enhancement, the current phase control strategy enables lift enhancement over both positive and negative phase shifts. While not shown, the drag

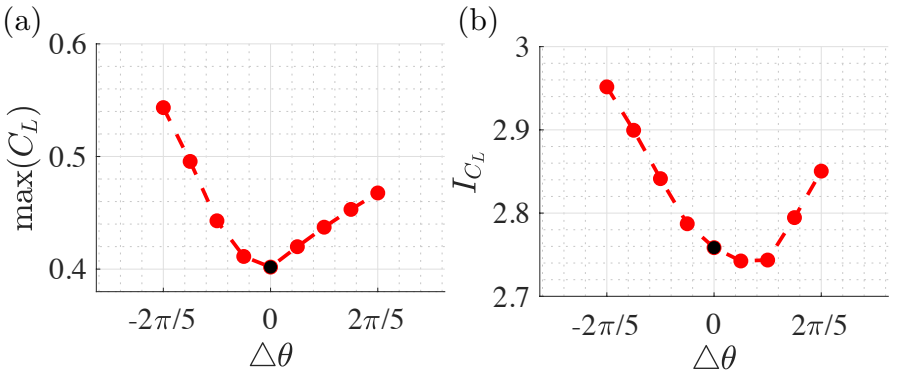


Figure 56: Evaluation of control performance: (a) Maximum lift coefficient and (b) lift impulse for varying phase shifts.

force decreases during control for both positive and negative phase shifts. When the control is switched off, the drag force increases due to the nonlinear interaction with the trailing edge vortices, breaking the flow stabilization in the wake.

On observing the characteristic vorticity flow fields associated with the control cases in Figure 79(f), we see that a leading-edge vortex (LEV) (Wang & Eldredge 2013; Eldredge & Jones 2019) forms earlier for the controlled flow with negative phase shifts. For positive phase shifts, the roll-up process of the leading edge vortex is delayed, resulting in a wake elongation. In the negative phase shift cases, the interaction of the shed LEV with the trailing edge vortex (TEV) results in the high lift after the control is switched off. These results are consistent with the findings of Darabi & Wygnanski (2004) and Colonius & Williams (2011). Thus, with optimal phase control, we can precisely control the timing of vortex shedding. We also highlight the maximum lift force and corresponding lift impulse in Figure 80(a) and (b), respectively. We can clearly see that negative phase shifts are preferred for maximum lift and lift impulse. A clear benefit in lift impulse for positive phase shifts are only observed for $\triangle\theta > \pi/10$. Thus, phase optimal control is an effective way of increasing transient aerodynamic performance.

11.4. Conclusion

We developed a simple yet robust control strategy to modify the phase of highly nonlinear, periodic fluid flows. Our control strategy combines phase-amplitude reduction and optimal phase-based control techniques for high-dimensional unsteady fluid flows. The phase-reduction technique used here provides a path to concisely describe the phase dynamics of unsteady fluid flows via a phase sensitivity function, complementing Floquet theory. Importantly, we show how this description can be used for effective flow control, demonstrating how it can be used to modify vortex dynamics in the near wake of bluff body flows, and on timescales that are commensurate with the period of vortex shedding. We consider several example systems, including the Stuart–Landau oscillator, the canonical fluid flow past a cylinder, and a highly complex fluid flow past an airfoil, illustrating the use of this approach for practical tailoring of aerodynamic responses. Our control strategy requires only measurements of physically attainable observables, such as lift and drag, and does not require high-dimensional data.

We demonstrate the performance of this phase-based control approach for two high-dimensional flow systems, the canonical flow over a cylinder and flow over an airfoil. In each system, our control law is only active for around one oscillation cycle, and is highly

DISTRIBUTION A: Distribution approved for public release.

effective in modifying the phase of the flow to a desired phase shift. We explore the phase-optimal control strategy for a range of prescribed phase shifts with respective to the baseline oscillation. Interestingly, two separate mechanisms are exploited to achieve lift enhancement for negative and positive phase shifts. For negative phase shifts, the vortex formation process and shedding is accelerated; once the vortex is shed, we see a transient increase in lift. In contrast, for positive phase shifts, vortex structures in the wake are elongated, causing an immediate increase in lift. As the wake is elongated and in turn stabilized, we also achieve a transient drag reduction. Investigating the relationship between the control forcing and the phase-sensitivity function, we observe that for positive phase shifts, the control principle is similar to performing direct opposition control to restrict the shedding of vortex structures.

We propose two control approaches to rapidly and efficiently alter the phase dynamics of unsteady fluid flows, and comparing these two approaches illuminates some interesting trade-offs. In the first approach, we solve an offline optimization problem using the Euler-Lagrange equations and then design the optimal forcing. Since this solution is tied directly to the phase sensitivity function, it is simple, effective, and has an intuitive interpretation. The second approach involves real-time feedback control using model-predictive control (MPC). For the simple Stuart-Landau oscillator, the two approaches are remarkably similar. However, we show that the MPC approach is especially successful in systems with stochastic disturbances and slowly varying parameters, where MPC produces effective control laws that correct for model uncertainty and are robust to noise.

This work is mainly focused on controlling the phase of periodic vortex shedding in unsteady fluid flows, and there is tremendous scope for future work to build on this analysis. Coupling the current framework with isostable coordinates (Wilson 2020), it will be possible to alter both phase and amplitude simultaneously for unsteady fluid flows. Further work needs to be done in representing the phase of more complex multi-modal observables, such as those found in fluid-structure interactions. A thorough assessment on the underlying influence of controllability and observability to the effectiveness of the flow control strategy is also needed. Extending this work for quasi-periodic and turbulent flows is an important future direction. The phase-amplitude reduction analysis could be combined with cluster-based approaches (Kaiser *et al.* 2014*a*; Nair *et al.* 2019) to improve its applicability to aperiodic and turbulent flows. Finally, it may be possible to extend the optimization procedure described above to a longer horizon, potentially uncovering more favorable control laws.

11.5. Appendix: Model-predictive control (MPC)

Here, we elaborate on model-predictive control (MPC) (Garcia *et al.* 1989) for optimal phase control. MPC can be used to perform real-time feedback control, can incorporate constraints, and has been shown to be robust to noisy observations. A schematic for MPC is shown in Figure 81 (a). For a problem of interest, there are three key components for MPC: (i) the dynamical system (plant) on which control is to be performed (in the case of fluids flows, this would be the Navier–Stokes equation, given by Eq. (11.1)), (ii) the model used for prediction of the state (or observable), and (iii) the control optimization algorithm used to deduce the real-time optimal control law. We also require knowledge of a desired reference state to be achieved with control. In the MPC framework, an optimal control problem is solved over a receding horizon to determine the next control input. The optimization is repeated at each time step and the control law is updated, as shown in Figure 81 (b).

In the work of Kaiser *et al.* (Kaiser *et al.* 2018), MPC was combined with several models based on dynamic mode decomposition (DMD) (Schmid 2010*b*; Proctor *et al.*

DISTRIBUTION A: Distribution approved for public release.

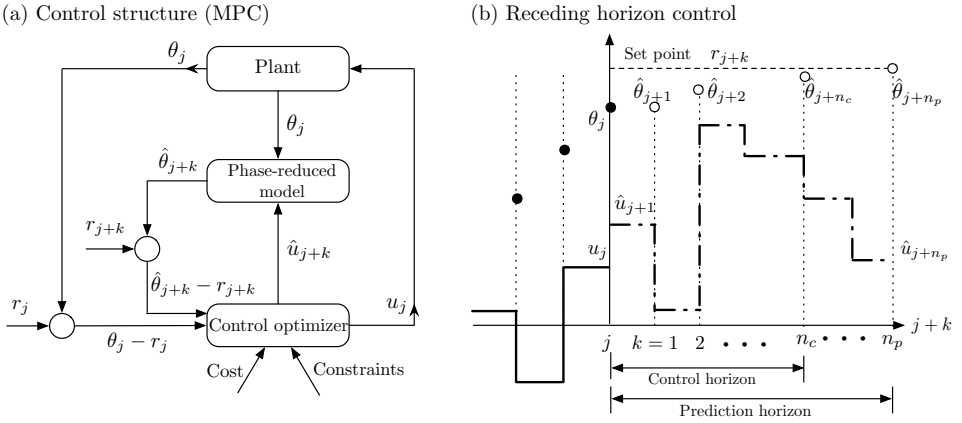


Figure 57: (a) Schematic of model-predictive control (MPC) structure and (b) receding horizon framework (adapted from Kaiser *et al.* (2018)).

2016*a*), neural networks (Hornik *et al.* 1989) and sparse identification of nonlinear dynamics (SINDy) (Brunton *et al.* 2016*b*). Here, we use the phase-reduced model, given by Eq. (11.7), where $\boldsymbol{f}(t)$ is the forcing input. For simplicity, at time t_j, r_j is the reference state, θ_j is the phase coordinate, and u_j is the control input. Note that for unsteady fluid flows, the forcing is described by Eq. (11.10). For phase-based control, the reference r should be T^* periodic for a desired phase shift of $\triangle\theta = \omega_n(T^* - T)$; thus $r_j = 2\pi t_j/T^*$. The output of the phase-reduced model is $\hat{\theta}$ and the control optimizer output is $\hat{u}$. As shown in Figure 81 (b), the prediction and optimization horizons are n_p and n_c time steps, respectively. At each step, the control optimization identifies a sequence of control inputs $\hat{u} = \{\hat{u}_{j+1}, \hat{u}_{j+2}, \cdots, \hat{u}_{j+n_c}\}$ based on predicted phases $\hat{\theta} = \{\hat{\theta}_{j+1}, \hat{\theta}_{j+2}, \cdots, \hat{\theta}_{j+n_p}\}$, after which $u_j = \hat{u}_{j+1}$ is applied to the plant, and then the procedure is repeated.

The MPC optimization (Kaiser *et al.* 2018) at each time step is given by

$$\min_{\hat{u}} \mathcal{J} = \min_{\hat{u}} \sum_{k=0}^{n_p} ||\hat{\theta}_{j+k} - r_{j+k}||_Q^2 + \sum_{k=1}^{n_c} (||\hat{u}_{j+k}||_{R_u}^2 + ||\triangle\hat{u}_{j+k}||_R^2) \tag{11.18}$$

subject to constraints $\hat{u}_{\min} \leqslant \hat{u}_k \leqslant \hat{u}_{\max}$ and $\triangle\hat{u}_{\min} \leqslant \triangle\hat{u}_k \leqslant \triangle\hat{u}_{\max}$. Here $\triangle\hat{u}_k = \hat{u}_k - \hat{u}_{k-1}$ is the input rate. We penalize the phase coordinate, control input and its rate. Also, for each term, a weighted norm is computed $||x||_Q = x^T Q x$. The weights are chosen to be positive semi-definite.

DISTRIBUTION A: Distribution approved for public release.

12. Randomized Dynamic Mode Decomposition

This section presents a randomized algorithm for computing the near-optimal low-rank dynamic mode decomposition (DMD). Randomized algorithms are emerging techniques to compute low-rank matrix approximations at a fraction of the cost of deterministic algorithms, easing the computational challenges arising in the area of 'big data'. The idea is to derive a small matrix from the high-dimensional data, which is then used to efficiently compute the dynamic modes and eigenvalues. The algorithm is presented in a modular probabilistic framework, and the approximation quality can be controlled via oversampling and power iterations. The effectiveness of the resulting randomized DMD (rDMD) algorithm is demonstrated on several benchmark examples of increasing complexity, providing an accurate and efficient approach to extract spatiotemporal coherent structures from big data in a framework that scales with the intrinsic rank of the data, rather than the ambient measurement dimension. For this work we assume that the dynamics of the problem under consideration is evolving on a low-dimensional subspace that is well characterized by a fast decaying singular value spectrum.

Extracting dominant coherent structures and modal expansions from high-dimensional data is a cornerstone of computational science and engineering (Taira *et al.* 2017*b*). The dynamic mode decomposition (DMD) is a leading data-driven algorithm to extract spatiotemporal coherent structures from high-dimensional data sets (Schmid 2010*b*; Tu *et al.* 2014; Kutz *et al.* 2016*a*). DMD originated in the fluid dynamics community (Schmid 2010*b*; Rowley *et al.* 2009*b*), where the identification of coherent structures, or *modes*, is often an important step in building models for prediction, estimation, and control (Brunton & Noack 2015*a*; Taira *et al.* 2017*b*). In contrast to the classic proper orthogonal decomposition (POD) (Berkooz *et al.* 1993*b*; Holmes *et al.* 2012*b*), which orders modes based on how much energy or variance of the flow they capture, DMD identifies spatially correlated modes that oscillate at a fixed frequency in time, possibly with an exponentially growing or decaying envelope. Thus, DMD combines the advantageous features of POD in space and the Fourier transform in time (Kutz *et al.* 2016*a*).

With rapidly increasing volumes of measurement data from simulations and experiments, modal extraction algorithms such as DMD may become prohibitively expensive, especially for online or real-time analysis. Even though DMD is based on an efficient singular value decomposition (SVD), computations scale with the dimension of the measurements, rather than with the intrinsic dimension of the data. With increasingly vast measurements, it is often the case that the intrinsic rank of the data does not increase appreciably, even as the dimension of the ambient measurements grows. Indeed, many high-dimensional systems exhibit low-dimensional patterns and coherent structures, which is one of the driving perspectives in both POD and DMD analysis. In this work, we develop a computationally effective strategy to compute the DMD using randomized linear algebra, which scales with the intrinsic rank of the dynamics, rather than with the measurement dimension. More concretely, we embed the DMD in a probabilistic framework by following the seminal work of Halko et al. (Halko *et al.* 2011). The main concept is depicted in Figure 58. Numerical experiments show that our randomized algorithm achieves considerable speedups over previously proposed algorithms for computing the DMD such as the compressed DMD algorithm (Brunton *et al.* 2015). Further, the approximation error can be controlled via oversampling and additional power iterations. This allows the user to choose the optimal trade-off between computational time and accuracy. Importantly, we also demonstrate the ability to handle data which are too big to fit into fast memory by using a blocked matrix scheme.

DISTRIBUTION A: Distribution approved for public release.

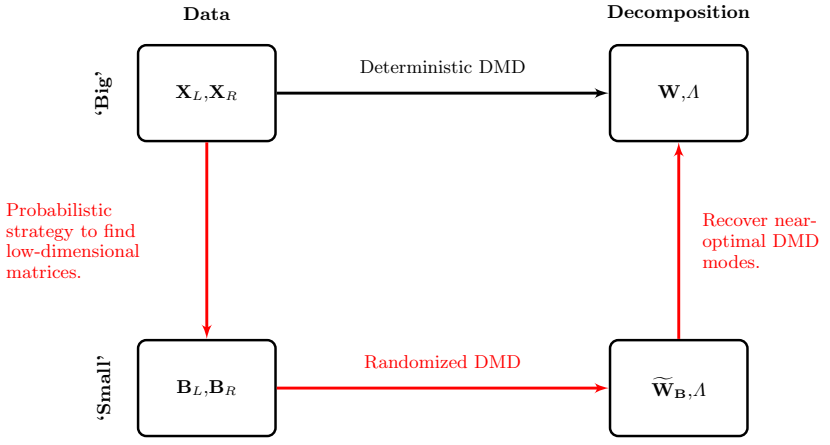


Figure 58: Conceptual architecture of the randomized dynamic mode decomposition (rDMD). First, small matrices are derived from the high-dimensional input data. The low-dimensional snapshot matrices $\mathbf{B}_L$ and $\mathbf{B}_R$ are then used to compute the approximate dynamic modes $\widetilde{\mathbf{W}}_\mathbf{B}$, and eigenvalues Λ. Finally, the near-optimal modes $\mathbf{W}$ may be recovered.

12.1. Related Work

Efforts to address the computational challenges of DMD can be traced back to the original paper by Schmid (Schmid 2010*b*). It was recognized that DMD could be computed in a lower dimensional space and then used to reconstruct the dominant modes and dynamics in the original high-dimensional space. This philosophy has been adopted in several strategies to determine the low-dimensional subspace, as shown in Fig. 59. Since then, efficient parallelized algorithms have been proposed for computations at scale (Sayadi & Schmid 2016).

In (Schmid 2010*b*), the high-dimensional data are projected onto the linear span of the associated POD modes, enabling an inexpensive low-dimensional DMD. The high-dimensional DMD modes have the same temporal dynamics as the low-dimensional modes and are obtained via lifting with the POD modes. However, computing the POD modes may be expensive and strategies have been developed to alleviate this cost. An algorithm utilizing the randomized singular value decomposition (rSVD) has been presented by (Erichson & Donovan 2016) and (Bistrian & Navon 2016, 2018). While this approach is reliable and robust to noise, only the computation of the SVD is accelerated, so that subsequent computational steps involved in the DMD algorithm remain expensive. The paper by Bistrian and Navon (Bistrian & Navon 2016) has other advantages, including identifying the most influential DMD modes and guarantees that the low-order model satisfies the boundary conditions of the full model.

As an alternative to using POD modes, a low-dimensional approximation of the data can be obtained by random projections (Brunton *et al.* 2015). This is justified by the Johnson-Lindenstrauss lemma (Johnson & Lindenstrauss 1984; Fowler 2009) which provides statistical guarantees on the preservation of the distances between the data when projected into the low-dimensional space. This approach avoids the cost of computing POD modes and has favorable statistical error bounds.

In contrast with these projection-based approaches, alternative strategies for reducing the dimension of the data involve computations with subsampled snapshots. In this case, the reduction of the computational cost comes from the fact that only a subset of the

DISTRIBUTION A: Distribution approved for public release.

data is used to derive the DMD and the full data is never processed by the algorithm. DMD is then determined from a *sketch-sampled* database. Variants of this approach differ in the sampling strategy; for example, (Brunton *et al.* 2015) and (Erichson *et al.* 2016*a*), develop the compressed dynamic mode decomposition, which involves forming a small data matrix by randomly projecting the high-dimensional row-space of a larger data matrix. This approach has been successful in computational fluid dynamics and video processing applications, where the data have relatively low noise. However, this is a suboptimal strategy, which leads to a large variance in the performance due to poor statistical guarantees in the resulting sketched data matrix. Clustering techniques can be employed to select a relevant subset of the data (Guéniat *et al.* 2015). Further alternatives derive from algebraic considerations, such as the maximization of the volume of the submatrix extracted from the data (e.g., Q-DEIM (Drmač & Gugercin 2016; Manohar *et al.* 2018)) or rely on energy-based arguments such as leverage-scores and length-squared sampling (Frieze *et al.* 2004; Mahoney 2011; Drineas & Mahoney 2016).

12.2. Assumptions, Limitations, Applications, and Extensions

Within a short time, DMD has become a workhorse algorithm for extracting dominant low-dimensional oscillating patterns from high-dimensional data. One key benefit of DMD is that it is equally valid for experimental and numerical data, as it does not rely on knowledge of the governing equations. Although it may not be stated explicitly, it is often assumed that the data is periodic or quasi-periodic in nature. DMD typically fails for data that is non-stationary or that exhibits broadband or intermittent phenomena. DMD is also quite sensitive to noise (Duke *et al.* 2012; Dawson *et al.* 2016), although several algorithms exist to de-bias DMD results in the presence of noisy data (Dawson *et al.* 2016; Hemati *et al.* 2017). Although not a limitation, most applications of DMD involve high-dimensional systems that evolve on a low-dimensional attractor, in which case the singular value decomposition is used to determine the subspace. Despite the assumptions and limitations above, DMD has been widely applied on a variety of systems in fluid mechanics (Rowley *et al.* 2009*b*; Schmid *et al.* 2011; Lusseyran *et al.* 2011; Seena & Sung 2011; Basley *et al.* 2013), epidemiology (Proctor & Eckhoff 2015), neuroscience (Brunton *et al.* 2016*a*), video modeling (Erichson *et al.* 2016*b*; Erichson & Donovan 2016; Kutz *et al.* 2015), robotics (Berger *et al.* 2015), and plasma physics (Taylor *et al.* 2017). Much of the success of DMD relates to its simple formulation in terms of linear algebra. The simplicity of DMD has enabled several extensions, including for control (Proctor *et al.* 2016*b*), multiresolution analysis (Kutz *et al.* 2016*b*), recursive orthogonalization of modes for Galerkin projection (Noack *et al.* 2016*b*), the use of time-delay coordinates (Tu *et al.* 2014; Arbabi & Mezić 2017; Das & Giannakis 2017), sparse identification of dynamic regimes (Kramer *et al.* 2015), Bayesian formulations (Takeishi *et al.* 2017).

Many of the applications and extensions above fundamentally rely on the dynamics evolving on a low-dimensional subspace that is well characterized by a fast decaying singular value spectrum. Although this is not a fundamental limitation of DMD, it is often a basic assumption, and we rely on this low-rank structure here.

Contribution of the Present Work

This work presents a randomized DMD algorithm that enables the accurate and efficient extraction of spatiotemporal coherent structures from high-dimensional data. The algorithm scales with the intrinsic rank of the dynamics, which are assumed to be low-dimensional, rather than the ambient measurement dimension. We demonstrate the effectiveness of this algorithm on two data sets of increasing complexity, namely the canonical fluid flow past a circular cylinder and the high-dimensional sea-surface

DISTRIBUTION A: Distribution approved for public release.

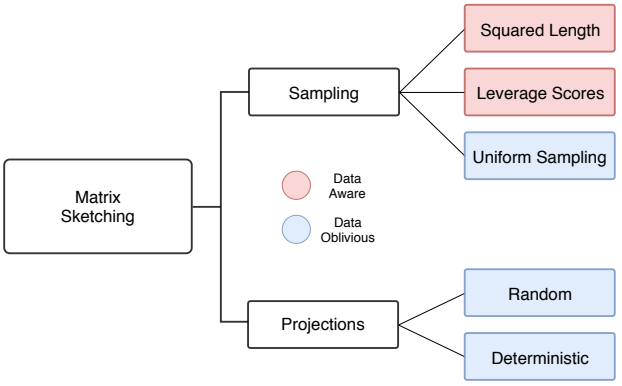


Figure 59: Sketching techniques for high-dimensional data; this work uses random projections.

temperature dataset. Moreover, we show that is possible to control the accuracy of the algorithm via oversampling and power iterations. In order to promote reproducible research, our open-source code is available at https://github.com/erichson/ristretto.

12.3. Technical Preliminaries

We now establish notation and provide a brief overview of the singular value decomposition (SVD) and the dynamic mode decomposition (DMD).

12.3.1. Notation

Vectors in $\mathbb{R}^n$ and $\mathbb{C}^n$ are denoted as bold lower case letters $\mathbf{x} = [x_1, x_2, ..., x_n]$. Both real $\mathbb{R}^{n\times m}$ and complex $\mathbb{C}^{n\times m}$ matrices are denoted by bold capitals $\mathbf{X}$, and its entry at the i-th row and j-th column is denoted as $\mathbf{X}(i, j)$. The Hermitian transpose of a matrix is denoted as $\mathbf{X}^*$. The spectral or operator norm of a matrix is defined as the largest singular value $\sigma_{\max}(\mathbf{X})$ of the matrix $\mathbf{X}$, i.e., the square root of the largest eigenvalue $\lambda_{\max}$ of the positive-semidefinite matrix $\mathbf{X}^*\mathbf{X}$:

$$\|\mathbf{X}\|_2 = \sqrt{\lambda_{\max}(\mathbf{X}^*\mathbf{X})} = \sigma_{\max}(\mathbf{X}).$$

The Frobenius norm of a matrix $\mathbf{X}$ is the positive square root of the sum of the absolute squares of its elements, which is equal to the positive square root of the trace of $\mathbf{X}^*\mathbf{X}$

$$\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{m} |\mathbf{X}(i,j)|^2} = \sqrt{\text{trace}(\mathbf{X}^*\mathbf{X})}.$$

The relative reconstruction error is $\|\mathbf{X} - \widehat{\mathbf{X}}\|_F / \|\mathbf{X}\|_F$, where $\widehat{\mathbf{X}}$ is an approximation to the matrix $\mathbf{X}$. The column space (range) of $\mathbf{X}$ is denoted $\text{col}(\mathbf{X})$ and the row space is $\text{row}(\mathbf{X})$.

12.4. The Singular Value Decomposition

Given an $n \times m$ matrix $\mathbf{X}$, the 'economic' singular value decomposition (SVD) produces the factorization

$$\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^* = \sum_{i}^{r} \mathbf{u}_i \sigma_i \mathbf{v}_i^*, \tag{12.1}$$

where $\mathbf{U} = [\mathbf{u}_1, ..., \mathbf{u}_r] \in \mathbb{R}^{n\times r}$ and $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_r] \in \mathbb{R}^{m\times r}$ are orthonormal, $\Sigma \in \mathbb{R}^{r\times r}$ is diagonal, and $r = \min(m, n)$. The left singular vectors in $\mathbf{U}$ provide a basis for the

column space of $\mathbf{X}$, and the right singular vectors in $\mathbf{V}$ form a basis for the row space of $\mathbf{X}$. Σ contains the corresponding non-negative singular values $\sigma_1 \geqslant ... \geqslant \sigma_r \geqslant 0$. Often, only the k dominant singular vectors and values are of interest, resulting in the low-rank SVD:

$$\mathbf{X}_k = \mathbf{U}_k \Sigma_k \mathbf{V}_k^* = [\mathbf{u}_1, \ldots, \mathbf{u}_k]\mathrm{diag}(\sigma_1, \ldots, \sigma_k)[\mathbf{v}_1, \ldots, \mathbf{v}_k]^* = \sum_i^k \mathbf{u}_i \sigma_i \mathbf{v}_i^*.$$

[The Moore-Penrose Pseudoinverse]The Moore-Penrose Pseudoinverse Given the singular value decomposition $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^*$, the pseudoinverse $\mathbf{X}^\dagger$ is computed as

$$\mathbf{X}^\dagger := \mathbf{V}\Sigma^\dagger\mathbf{U}^* = \sum_i^r \mathbf{v}_i \sigma_i^\dagger \mathbf{u}_i^*, \tag{12.2}$$

where $\Sigma^\dagger$ is here understood as

$$\Sigma^\dagger = \mathrm{diag}\left(\sigma_i^\dagger\right), \qquad \sigma_i^\dagger = \begin{cases} \sigma_i^{-1} & \text{if } \sigma_i > 0, \\ 0 & \text{otherwise,} \end{cases} \qquad \forall\, i.$$

We use the Moore-Penrose pseudoinverse in the following to provide a least squares solution to a system of linear equations.

12.5. Dynamic Mode Decomposition

DMD is a dimensionality reduction technique, originally introduced in the field of fluid dynamics (Schmid 2010*b*; Rowley *et al.* 2009*b*; Kutz *et al.* 2016*a*). The method extracts spatiotemporal coherent structures from an ordered time series of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$, separated in time by a constant step Δt. Specifically, the aim is to find the eigenvectors and eigenvalues of the time-independent linear operator $\mathbf{A} : \mathbb{R}^n \to \mathbb{R}^n$ that best approximates the map from a given snapshot $\mathbf{x}_j$ to the subsequent snapshot $\mathbf{x}_{j+1}$ as

$$\mathbf{x}_{j+1} \approx \mathbf{A}\mathbf{x}_j. \tag{12.3}$$

Following (Tu *et al.* 2014), the deterministic DMD algorithm proceeds by first separating the snapshot sequence $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m$ into two overlapping sets of data

$$\mathbf{X}_L = \begin{bmatrix} | & | & & | \\ \mathbf{x}_0 & \mathbf{x}_1 & \cdots & \mathbf{x}_{m-1} \\ | & | & & | \end{bmatrix}, \quad \mathbf{X}_R = \begin{bmatrix} | & | & & | \\ \mathbf{x}_1 & \mathbf{x}_2 & \cdots & \mathbf{x}_m \\ | & | & & | \end{bmatrix}. \tag{12.4}$$

$\mathbf{X}_L \in \mathbb{R}^{n \times m}$ and $\mathbf{X}_R \in \mathbb{R}^{n \times m}$ are called the left and right snapshot sequences. Equation (12.3) can then be reformulated in matrix notation as

$$\mathbf{X}_R \approx \mathbf{A}\mathbf{X}_L. \tag{12.5}$$

Many variants of the DMD have been proposed since its introduction, as discussed in (Kutz *et al.* 2016*a*). Here, we discuss the *exact DMD* formulation of Tu et al. (Tu *et al.* 2014).

12.5.1. Non-Projected DMD

In order to find an estimate for the linear map $\mathbf{A}$, the following least-squares problem can be formulated

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \|\mathbf{X}_R - \mathbf{A}\mathbf{X}_L\|_F^2. \tag{12.6}$$

DISTRIBUTION A: Distribution approved for public release.

The estimator for the best-fit linear map is given in closed-form as

$$\widehat{\mathbf{A}} := \mathbf{X}_R \mathbf{X}_L^{\dagger}, \tag{12.7}$$

where $\mathbf{X}^{\dagger}$ denotes the pseudoinverse from Eq. (12.2). The DMD modes are the eigenvectors of $\widehat{\mathbf{A}} \in \mathbb{R}^{n \times n}$.

12.5.2. Projected DMD

If the data is high-dimensional (i.e., n is large), the linear map $\widehat{\mathbf{A}}$ in Eq. (12.7) may be intractable to evaluate and analyze directly. Instead, a rank-reduced approximation of the linear map is determined by projecting it onto a k-dimensional subspace, $k \leqslant \min(n, m)$. We denote $\mathbf{X}_L = \mathbf{U}\boldsymbol{\Sigma}\mathbf{V}^*$ the singular value decomposition of $\mathbf{X}_L$. The projected map onto the class of rank-k linear operators can be obtained by pre- and post-multiplying Eq. (12.7) with $\mathbf{P}_k \in \mathbb{R}^{n \times k}$ and $\mathbf{T}_k \in \mathbb{R}^{n \times k}$:

$$\widetilde{\mathbf{A}} := \mathbf{P}_k^* \widehat{\mathbf{A}} \mathbf{T}_k = \mathbf{P}_k^* \mathbf{X}_R \mathbf{X}_L^{\dagger} \mathbf{T}_k = \mathbf{P}_k^* \mathbf{X}_R \mathbf{V} \boldsymbol{\Sigma}^{\dagger} \mathbf{U}^* \mathbf{T}_k, \tag{12.8}$$

where $\widetilde{\mathbf{A}} \in \mathbb{R}^{k \times k}$ is the projected map, i.e., a low-dimensional system matrix.

The *projected DMD* (Schmid 2010*b*) relies on the hypothesis that the columns of $\mathbf{A}\mathbf{X}_L$ are in the linear span of $\mathbf{X}_L$. Within this approximation, projection matrices $\mathbf{P}$ and $\mathbf{T}$ are chosen such that they span a subspace of the column space of $\mathbf{X}_L$. A convenient choice is given by the dominant (i.e., associated with the largest singular values) left singular vectors $\mathbf{U}_k$ of $\mathbf{X}_L$ (i.e., POD modes) so that $\mathbf{P}_k = \mathbf{T}_k = \mathbf{U}_k$, $\mathbf{U}_k^* \mathbf{U}_k = \mathbf{I}_k$, with $k \leqslant \operatorname{rank}(\mathbf{X}_L)$, and

$$\widetilde{\mathbf{A}} = \mathbf{U}_k^* \mathbf{X}_R \mathbf{V}_k \boldsymbol{\Sigma}_k^{-1}, \tag{12.9}$$

since $\mathbf{U}^* \mathbf{U}_k = \begin{bmatrix} \mathbf{I}_k \\ \mathbf{0} \end{bmatrix}$.

Note that when we use the POD modes $\mathbf{U}$, one can interpret $\widehat{\mathbf{A}}\mathbf{U}_k$ as the shifted spatial POD modes over one time step Δt. Now, by premultiplying $\widehat{\mathbf{A}}\mathbf{U}_k$ with $\mathbf{U}_k^*$, one obtains a projected map $\widetilde{\mathbf{A}} = \mathbf{U}_k^* \widehat{\mathbf{A}} \mathbf{U}_k$ which encodes the cross-correlation of the POD modes $\mathbf{U}_k$ with the time-shifted spatial POD modes. It thus becomes obvious that the DMD encodes more information about the temporal evolution of the system under consideration than the time-averaged POD modes (Schmid 2010*b*).

Low-dimensional projection is a form of spectral filtering which has the positive effect of dampening the influence of noise (Hansen *et al.* 2006; Gorodnitsky & Rao 1994). This effect is illustrated in Figure 60, which shows the non-projected linear map in absence and presence of white noise. The projected linear map avoids the amplification of noise and acts as a hard-threshold regularizer. The difficulty is to choose the rank providing a good trade-off between suppressing the noise and retaining the useful information. In practice, the optimal hard-threshold (Gavish & Donoho 2014) provides a good heuristic to determine the rank k.

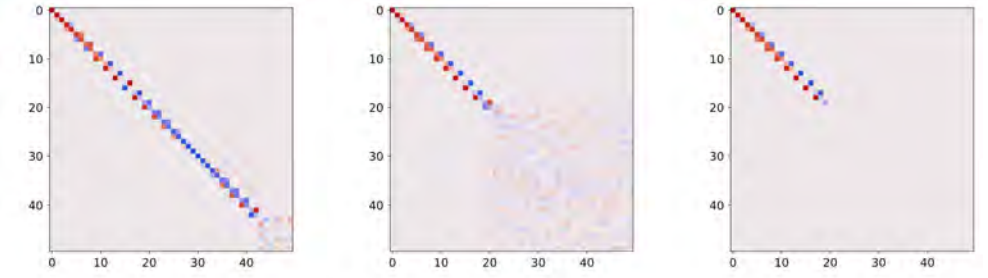
Once the linear map $\widetilde{\mathbf{A}}$ is approximated, its eigendecomposition is computed

$$\widetilde{\mathbf{A}} \widetilde{\mathbf{W}} = \widetilde{\mathbf{W}} \boldsymbol{\Lambda}, \tag{12.10}$$

where the columns of $\widetilde{\mathbf{W}} \in \mathbb{C}^{k \times k}$ are eigenvectors of $\widetilde{\mathbf{A}}$, and $\boldsymbol{\Lambda} \in \mathbb{C}^{k \times k}$ is a diagonal matrix containing the corresponding eigenvalues λ_j. Eigenvalues of $\widetilde{\mathbf{A}}$ are eigenvalues of $\widehat{\mathbf{A}}$ and one may recover DMD modes as the columns of $\mathbf{W}_k$, (Tu *et al.* 2014):

$$\mathbf{W}_k := \mathbf{X}_R \mathbf{V}_k \boldsymbol{\Sigma}_k^{-1} \widetilde{\mathbf{W}}. \tag{12.11}$$

DISTRIBUTION A: Distribution approved for public release.



(a) Non-projected linear map in the absence of white noise.

(b) Non-projected linear map in the presence of white noise.

(c) Projected linear map in the presence of white noise.

Figure 60: Illustration of the non-projected and projected map for toy data in the absence and presence of white noise. The regularization effect of the projected map reduces the influence of noise, while preserving the dominant information.

In both the projected and the non-projected formulation, a singular value decomposition of $\mathbf{X}_L$ is involved. This is computationally demanding, and the resources required to compute DMD can be tremendous for large data matrices.

12.5.3. A Note on Regularization

The projected algorithm described above relies on the truncated singular value decomposition (TSVD), also referred to as pseudo-inverse filter, to solve the unconstrained least-squares problem in Eq. (12.6). Indeed, Figure 60 illustrates that the low-rank approximation introduces an effective regularization effect. This warrants an extended discussion on regularization, following the work by Hansen (Hansen 1987, 1990; Hansen *et al.* 2006, 1992).

In DMD, we often experience a linear system of equations $\mathbf{X}_R = \mathbf{A}\mathbf{X}_L$ where $\mathbf{X}_L$ or $\mathbf{X}_R$ are ill-conditioned, i.e., the ratio between the largest and smallest singular value is large, so that the pseudo-inverse may magnify small errors. Regularization becomes crucial in this situation to compute an accurate estimate of $\mathbf{A}$, since a small perturbation in $\mathbf{X}_R$ or $\mathbf{X}_L$ may result in a large perturbation in the solution. Tikhonov-Phillips regularization (Tikhonov 1963; Phillips 1962), also known as ridge regression in the statistical literature, is one of the most popular regularization techniques. The regularized least squares problem becomes

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \left\|\mathbf{X}_R - \mathbf{A}\mathbf{X}_L\right\|_F^2 + \lambda^2 \left\|\mathbf{A}\right\|_F^2 . \tag{12.12}$$

More concisely, this can be expressed as the following augmented least-squares problem

$$\widehat{\mathbf{A}} = \arg\min_{\mathbf{A}} \left\| \begin{bmatrix} \mathbf{X}_R \\ \mathbf{0} \end{bmatrix} - \mathbf{A} \begin{bmatrix} \mathbf{X}_L \\ \lambda \mathbf{I} \end{bmatrix} \right\|_F^2 . \tag{12.13}$$

The estimator for the map $\mathbf{A}$ takes advantage of the regularized inverse

$$\mathbf{X}_\lambda^\dagger := [\mathbf{X}^*\mathbf{X} + \lambda^2 \mathbf{I}]^{-1}\mathbf{X}^* = \mathbf{V}\boldsymbol{\Sigma}_\lambda^+ \mathbf{U}^* \quad \text{with} \quad \boldsymbol{\Sigma}_\lambda^+ := \operatorname{diag}\left(\frac{\sigma_1}{\sigma_1^2 + \lambda^2}, ..., \frac{\sigma_r}{\sigma_r^2 + \lambda^2} \right) \tag{12.14}$$

where the additional regularization (controlled via the tuning parameter λ) improves the conditioning of the problem. This form of regularization can also be seen as a smooth filter that attenuates the parts of the solution corresponding to the small singular values.

DISTRIBUTION A: Distribution approved for public release.

Specifically, the filter f takes the form (Hansen 1987)

$$f_i = \frac{\sigma_i^2}{\sigma_i^2 + \lambda^2}, \qquad i = 1, 2, ..., r. \tag{12.15}$$

The Tikhonov-Phillips regularization scheme is closely related to the TSVD and the Wiener filter. Indeed, the TSVD is known to be a method for regularizing ill-posed linear least-squares problems, which often produces very similar results (Varah 1979). More concretely, regularization via the TSVD can be seen as a hard-threshold filter, which takes the form

$$f_i = \begin{cases} 1, & \sigma_i \geqslant \sigma_k \\ 0, & \sigma_i < \sigma_k \end{cases}. \tag{12.16}$$

Here, k controls how much smoothing (or low-pass filtering) is introduced, i.e., increasing k includes more terms in the SVD expansion; thus components with higher frequencies are included. See (Hansen *et al.* 2006) for further details. As a consequence of the regularization effect introduced by TSVD, the DMD algorithm requires a careful choice of the target-rank k to compute a meaningful low-rank DMD approximation. Indeed, the solutions (i.e., the computed DMD modes and eigenvalues) may be sensitive to the amount of regularization which is controlled via k. In practice, once can treat the parameter k as a tuning-parameter and find the optimal value via cross-validation.

12.6. Randomized Methods for Linear Algebra

With rapidly increasing volumes of measurement data from simulations and experiments, deterministic modal extraction algorithms may become prohibitively expensive. Fortunately, a wide range of applications produce data which feature low-rank structure, i.e., the rank of the data matrix, given by the number of independent columns or rows, is much smaller than the ambient dimension of the data space. In other words, the data feature a large amount of redundant information. In this case, randomized methods allow one to efficiently produce familiar decompositions from linear algebra. These so-called *randomized* numerical methods have the potential to transform computational linear algebra, providing accurate matrix decompositions at a fraction of the cost of deterministic methods. Indeed, over the past two decades, probabilistic algorithms have been prominent for computing low-rank matrix approximations, as described in a number of excellent surveys (Halko *et al.* 2011; Mahoney 2011; Drineas & Mahoney 2016; Brunton & Kutz 2018). The idea is to use randomness as a computational strategy to find a smaller representation, often denoted as *sketch*. This smaller matrix sketch can be used to compute an approximate low-rank factorization for the high-dimensional data matrix $\mathbf{X} \in \mathbb{R}^{n \times m}$.

Broadly, these techniques can be divided into random sampling and random projection-based approaches. The former class of techniques carefully samples and rescales some 'interesting' rows or columns to form a sketch. For instance, we can sample rows by relying on energy-based arguments such as leverage-scores and length-squared sampling (Frieze *et al.* 2004; Mahoney 2011; Drineas & Mahoney 2016). The second class of random projection techniques relies on favorable properties of random matrices and forms the sketch as a randomly weighted linear combination of the columns or rows. Computationally efficient strategies to form such a sketch include the subsampled randomized Hadamard transform (SRHT) (Sarlos 2006; Tropp 2011) and the CountSketch (Woodruff *et al.* 2014). Choosing between the different sampling and random projection strategies depends on the particular application. In general, sampling is more computational efficient, while random projections preserve more of the information in the data.

DISTRIBUTION A: Distribution approved for public release.

12.6.1. Probabilistic Framework

One of the most effective off-the-shelf methods to form a sketch is the probabilistic framework proposed in the seminal work by Halko et al. (Halko *et al.* 2011):

- **Stage A:** Given a desired target-rank k, find a near-optimal orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ for the range of the input matrix $\mathbf{X}$.
- **Stage B:** Given the near-optimal basis $\mathbf{Q}$, project the input matrix onto the low-dimensional space, resulting in $\mathbf{B} \in \mathbb{R}^{k\times m}$. This smaller matrix can then be used to compute a near-optimal low-rank approximation.

12.6.2. Stage A: Computing a Near-Optimal Basis

The first stage is used to approximate the range of the input matrix. Given a target rank $k \ll \min(m, n)$, the aim is to compute a near-optimal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ for the input matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ such that

$$\mathbf{X} \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}. \tag{12.17}$$

Specifically, the range of the high-dimensional input matrix is sampled using the concept of random projections. Thus a basis is efficiently computed as

$$\mathbf{Y} = \mathbf{X}\Omega, \tag{12.18}$$

where $\Omega \in \mathbb{R}^{m\times k}$ denotes a random test matrix drawn from the normal Gaussian distribution. However, the cost of dense matrix multiplications can be prohibitive. The time complexity is order $\mathcal{O}(nmk)$. To improve this scaling, more sophisticated random test matrices, such as the subsampled randomized Hadamard transform (SRHT), have been proposed (Sarlos 2006; Tropp 2011). Indeed, the time complexity can be reduced to $\mathcal{O}(nm \cdot \log(k))$ by using a structured random test matrix to sample the range of the input matrix.

The orthonormal basis $\mathbf{Q} \in \mathbb{R}^{n\times k}$ is obtained via the QR-decomposition $\mathbf{Y} = \mathbf{QR}$.

Stage B: Computing a Low-Dimensional Matrix

We now derive a smaller matrix $\mathbf{B}$ from the high-dimensional input matrix $\mathbf{X}$. Specifically, given the near-optimal basis $\mathbf{Q}$, the matrix $\mathbf{X}$ is projected onto the low-dimensional space

$$\mathbf{B} = \mathbf{Q}^*\mathbf{X}, \tag{12.19}$$

which yields the smaller matrix $\mathbf{B} \in \mathbb{R}^{k\times m}$. This process preserves the geometric structure in a Euclidean sense, so that angles between vectors and their lengths are preserved. It follows that

$$\mathbf{X} \approx \mathbf{QB}. \tag{12.20}$$

12.6.3. Oversampling

In theory, if the matrix $\mathbf{X}$ has exact rank k, the sampled matrix $\mathbf{Y}$ spans a basis for the column space, with high probability. In practice, however, it is common that the truncated singular values $\{\sigma_i\}_{i\geqslant k+1}$ are non-zero. Thus, it helps to construct a slightly larger test matrix in order to obtain an improved basis. Thus, we compute $\mathbf{Y}$ using an $\Omega \in \mathbb{R}^{m\times l}$ test matrix instead, where $l = k + p$. Thus, the oversampling parameter p denotes the number of additional samples. In most situations small values $p = \{5, 10\}$ are sufficient to obtain a good basis that is comparable to the best possible basis (Halko *et al.* 2011).

DISTRIBUTION A: Distribution approved for public release.

(1)	$l = k + p$	Slight oversampling.
(2)	$\Omega = \texttt{rand}(m, l)$	Generate random test matrix.
(3)	$\mathbf{Y} = \mathbf{X}\Omega$	Compute sampling matrix.
(4)	**for** $j = 1, \ldots, q$	Power iterations (optional).
(5)	$[\mathbf{Q}, \sim] = \texttt{qr}(\mathbf{Y})$	
(6)	$[\mathbf{Z}, \sim] = \texttt{qr}(\mathbf{X}^*\mathbf{Q})$	
(7)	$\mathbf{Y} = \mathbf{X}\mathbf{Z}$	
(8)	**end for**	
(9)	$[\mathbf{Q}, \sim] = \texttt{qr}(\mathbf{Y})$	Orthonormalize sampling matrix.
(10)	$\mathbf{B} = \mathbf{Q}^*\mathbf{X}$	Project input matrix to smaller space.

Algorithm 1: Randomized QB decomposition. Given a matrix $\mathbf{X} \in \mathbb{R}^{n\times m}$ and a target rank $k \ll \min(m, n)$, the basis matrix $\mathbf{Q} \in \mathbb{R}^{n\times k}$ with orthonormal columns and the smaller matrix $\mathbf{B} \in \mathbb{R}^{k\times m}$ are computed. The approximation quality can be controlled via oversampling p and the computation of q power iterations.

12.6.4. Power Iteration Scheme

A second strategy to improve the performance is the concept of power iterations (Rokhlin *et al.* 2009; Gu 2015). In particular, a slowly decaying singular value spectrum of the input matrix can seriously affect the quality of the approximated basis matrix $\mathbf{Q}$. Thus, the method of power iterations is used to pre-process the input matrix in order to promote a faster decaying spectrum. The sampling matrix $\mathbf{Y}$ is obtained as

$$\mathbf{Y} = \big((\mathbf{X}\mathbf{X}^*)^q\mathbf{X}\big)\Omega \tag{12.21}$$

where q is an integer specifying the number of power iterations. With $\mathbf{X} = \mathbf{U}\Sigma\mathbf{V}^*$, one has $\mathbf{X}^{(q)} := (\mathbf{X}\mathbf{X}^*)^q\mathbf{X} = \mathbf{U}\Sigma^{2q+1}\mathbf{V}^*$. Hence, for $q > 0$, the pre-processed matrix $\mathbf{X}^{(q)}$ has a relatively fast decay of singular values compared to the input matrix $\mathbf{X}$. The drawback of this method is that additional passes over the input matrix are required. However, as few as $q = \{1, 2\}$ power iterations can considerably improve the approximation quality, even when the singular values of the input matrix decay slowly.

Algorithm 1 describes this overall procedure for the randomized QB decomposition.

A direct implementation of the power iteration scheme, as outlined in Section 12.6.4, is numerically unstable due to round-off errors. Instead, the sampling matrix $\mathbf{Y}$ is orthogonalized between each computational step to improve the stability (Halko *et al.* 2011).

As default values for the oversampling and power iteration parameter, we suggest $p = 10$, and $q = 2$.

12.6.5. Theoretical Performance

Both the concept of oversampling and the power iteration scheme provide control over the quality of the low-rank approximation. The average-case error behavior is described in the probabilistic framework as (Martinsson 2016):

$$\mathbb{E}_{\mu_\mathbf{Q}}\|\mathbf{X} - \mathbf{Q}\mathbf{Q}^*\mathbf{X}\|_F \leqslant \left[1 + \sqrt{\frac{k}{p-1}} + \frac{e\sqrt{l}}{p} \cdot \sqrt{\min(m, n) - k}\right]^{\frac{1}{2q+1}} \sigma_{k+1}(\mathbf{X}),$$

with $e \equiv \exp(1)$, $\mathbb{E}_{\mu_\mathbf{Q}}$ the expectation operator over the probability measure of $\mathbf{Q}$ and $\sigma_{k+1}(\mathbf{X})$ the $(k+1)$-th element of the decreasing sequence of singular values of $\mathbf{X}$. Here it is assumed that $p \geqslant 2$. Thus, both oversampling and the computation of additional power iterations drive the approximation error down.

DISTRIBUTION A: Distribution approved for public release.

Computational Considerations

The steps involved in computing the approximate basis $\mathbf{Q}$ and the low-dimensional matrix $\mathbf{B}$ are simple to implement, and embarrassingly parallelizable. Thus, randomized algorithms can benefit from modern computational architectures, and they are particularly suitable for GPU-accelerated computing. Another favorable computational aspect is that only two passes over the input matrix are required in order to obtain the low-dimensional matrix. Pass-efficiency is a crucial aspect when dealing with massive data matrices which are too large to fit into fast memory, since reading data from the hard disk is prohibitively slow and often constitutes the actual bottleneck.

Blocked Randomized Algorithm

When dealing with massive fluid flows that are too large to read into fast memory, the extension to sequential, distributed, and parallel computing might be inevitable (Sayadi & Schmid 2016). In particular, it might be necessary to distribute the data across processors which have no access to a shared memory to exchange information. To address this limitation, Martinsson and Voronin (Martinsson & Voronin 2016) proposed a blocked scheme to compute the QB decomposition on smaller blocks of the data. The basic idea is that a given high-dimensional sequence of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$ is subdivided into b smaller blocks along the rows. The submatrices can then be processed in b independent streams of calculations. Here, b is assumed to be a power of two, and zero padding can be used in order to divide the data into blocks of the same size. This scheme constructs the smaller matrix $\mathbf{B}$ in a hierarchical fashion, for more details see (Voronin & Martinsson 2015).

In the following, we describe a modified and simple scheme that is well-suited for practical applications such as computing the dynamic mode decomposition for large-scale fluid flows. Unlike (Martinsson & Voronin 2016; Voronin & Martinsson 2015), which discuss a fixed-precision approximation, we focus on a fixed-rank approximation problem. Further, our motivation is that it is often unnecessary to use the full hierarchical scheme if the desired target rank k is relatively small. By small we mean that we can fit a matrix of dimension $(b \cdot k) \times m$ into fast memory. To be more precise, suppose again that we are given an input matrix $\mathbf{X}$ with n rows and m columns. We partition $\mathbf{X}$ along the rows into b blocks $\mathbf{X}_i$ of dimension $n/b \times m$. To illustrate the scheme we set $b = 4$ so that

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_1 \\ \mathbf{X}_2 \\ \mathbf{X}_3 \\ \mathbf{X}_4 \end{bmatrix}. \tag{12.22}$$

Next, we approximate each block $\mathbf{X}_i$ by a fix rank-k approximation, using the QB decomposition described in Algorithm 1, which yields

$$\mathbf{X} \approx \begin{bmatrix} \mathbf{Q}_1\mathbf{B}_1 \\ \mathbf{Q}_2\mathbf{B}_2 \\ \mathbf{Q}_3\mathbf{B}_3 \\ \mathbf{Q}_4\mathbf{B}_4 \end{bmatrix} = \operatorname{diag}(\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \mathbf{Q}_4) \begin{bmatrix} \mathbf{B}_1 \\ \mathbf{B}_2 \\ \mathbf{B}_3 \\ \mathbf{B}_4 \end{bmatrix}. \tag{12.23}$$

We can then collect all matrices $\mathbf{B}_i \in \mathbb{R}^{k \times m}$ and stack them together as

$$\mathbf{K} = \begin{bmatrix} \mathbf{B}_1^* & \mathbf{B}_2^* & \mathbf{B}_3^* & \mathbf{B}_4^* \end{bmatrix}^*. \tag{12.24}$$

Subsequently, we compute the QB decomposition of $\mathbf{K} \in \mathbb{R}^{(b \cdot k) \times m}$ and obtain

$$\mathbf{K} \approx \widehat{\mathbf{Q}}\mathbf{B}. \tag{12.25}$$

DISTRIBUTION A: Distribution approved for public release.

The small matrix $\mathbf{B} \in \mathbb{R}^{k\times m}$ can then be used to compute the randomized dynamic mode decomposition as described below. The basis matrix $\mathbf{Q} \in \mathbb{R}^{n\times k}$ can be formed as

$$\mathbf{Q} = \operatorname{diag}(\mathbf{Q}_1, \mathbf{Q}_2, \mathbf{Q}_3, \mathbf{Q}_4)\,\widehat{\mathbf{Q}}. \tag{12.26}$$

In practice, we choose the target-rank l for the approximation slightly larger than k, i.e., $l = k + p$.

12.7. Randomized Dynamic Mode Decomposition

In many cases, even with increased measurement resolution, the data may have dominant coherent structures that define a low-dimensional attractor (Holmes *et al.* 2012*b*); in fact, the presence of these structures was the original motivation for methods such as DMD. Hence, it seems natural to use randomized methods for linear algebra to accelerate the computation of the approximate low-rank DMD.

12.7.1. Problem Formulation Revisited

We start our discussion by formulating the following least-squares problem to find an estimate $\widehat{\mathbf{A}}_{\mathbf{B}}$ for the projected linear map $\widetilde{\mathbf{A}} \in \mathbb{R}^{l\times l}$ in terms of the projected snapshot sequences $\mathbf{B}_L := \mathbf{P}^*\mathbf{X}_L \in \mathbb{R}^{l\times m}$ and $\mathbf{B}_R := \mathbf{P}^*\mathbf{X}_R \in \mathbb{R}^{l\times m}$:

$$\widehat{\mathbf{A}}_{\mathbf{B}} := \arg\min_{\mathbf{A}_{\mathbf{B}}} \|\mathbf{B}_R - \mathbf{A}_{\mathbf{B}}\mathbf{B}_L\|_F^2, \tag{12.27}$$

where $\mathbf{P} \in \mathbb{R}^{n\times l}$ is a projection matrix discussed below. Recall that $l = k + p$, where k denotes the desired target rank, and p is the oversampling parameter. Here we make the assumption that $\operatorname{col}(\mathbf{P}) \approx \operatorname{col}(\mathbf{X}_L)$ as well as $\operatorname{col}(\mathbf{P}) \approx \operatorname{col}(\mathbf{X}_R)$.

The question remains how to construct $\mathbf{P}$ in order to quickly compute $\mathbf{B}_L$ and $\mathbf{B}_R$. The dominant left singular vectors of $\mathbf{X}_R$ provide a good choice; however, the computational demands to compute the deterministic SVD may be prohibitive for large data sets. Next, we discuss randomized methods to compute DMD.

12.7.2. Compressed Dynamic Mode Decomposition

Brunton et al. (Brunton *et al.* 2015) proposed one of the first algorithms using the idea of matrix sketching to find small matrices $\mathbf{B}_L$ and $\mathbf{B}_R$. The algorithm proceeds by forming a small number of random linear combinations of the rows of the left and right snapshot matrices to form the following representation

$$\mathbf{B}_L = \mathbf{S}\mathbf{X}_L, \quad \mathbf{B}_R = \mathbf{S}\mathbf{X}_R. \tag{12.28}$$

with $\mathbf{S} \in \mathbb{R}^{l\times n}$ a random test matrix. As discussed in Section **??**, $\mathbf{S}$ can be constructed by drawing its entries from the standard normal distribution. While using a Gaussian random test matrix has beneficial theoretical properties, the dense matrix multiplication $\mathbf{S}\mathbf{X}_L$ can be expensive for large dense matrices. Alternatively, $\mathbf{S}$ can be chosen to be *a random and rescaled subset* of the identity matrix, i.e., $\mathbf{B}$ is formed by sampling and rescaling l rows from $\mathbf{X}$ with probability p_i. Thus, $\mathbf{S}$ is very sparse and is not required to be explicitly constructed and stored. More concretely, we sample the ith row of $\mathbf{X}$ with probability p_i and, if sampled, the row is rescaled by the factor $1/\sqrt{l \cdot p_i}$ in order to yield an unbiased estimator (Drineas & Mahoney 2017). A naive approach is to sample rows with uniform probability $p_i = 1/n$. Uniform random sampling may work if the information is evenly distributed across the input data, so that dominant structures are not spatially localized. Indeed, it was demonstrated in (Brunton *et al.* 2015) that the dominant DMD modes and eigenvalues for some low-rank fluid flows can

DISTRIBUTION A: Distribution approved for public release.

be obtained from a massively under-sampled sketch. However, the variance can be large and the performance may be poor in the presence of white noise. While not explored by (Brunton *et al.* 2015), the performance can be improved using more sophisticated sampling techniques, such as leverage score sampling (Drineas *et al.* 2012; Ma *et al.* 2015; Drineas & Mahoney 2017). Better performance is expected when using structured random test matrices such as the subsampled randomized Hadamard transform (SRHT) (Sarlos 2006; Tropp 2011) or the CountSketch (Woodruff *et al.* 2014). Both of these methods enable efficient matrix multiplications, yet they are more computationally demanding than random sampling.

The shortcoming of the algorithm in (Brunton *et al.* 2015) is that l depends on the ambient dimension of the measurement space of the input matrix. Thus, even if the data matrix is low-rank, a large l may be required to compute the approximate DMD. Next, we discuss a randomized algorithm which depends on the intrinsic rank of the input matrix.

An Improved Randomized Scheme

In the following, we present a novel algorithm to compute the dynamic mode decomposition using the probabilistic framework above. The proposed algorithm is simple to implement and can fully benefit from modern computational architectures, e.g., parallelization and multi-threading.

Given a sequence of snapshots $\mathbf{x}_0, \mathbf{x}_1, ..., \mathbf{x}_m \in \mathbb{R}^n$, we first compute the near-optimal basis $\mathbf{Q} \in \mathbb{R}^{n \times l}$ using the randomized methods which we have discussed in Section **??**. Then, we project the data onto the low-dimensional space, so that we obtain the low-dimensional sequence of snapshots

$$\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_m := \mathbf{Q}^* \mathbf{x}_0, \mathbf{Q}^* \mathbf{x}_1, ..., \mathbf{Q}^* \mathbf{x}_m \in \mathbb{R}^l.$$

More concisely, we can express this as

$$\mathbf{B} := \mathbf{Q}^* \mathbf{X}.$$

Next, we separate the sequence $\mathbf{b}_0, \mathbf{b}_1, ..., \mathbf{b}_m$ into two overlapping matrices $\mathbf{B}_L \in \mathbb{R}^{l \times m}$ and $\mathbf{B}_R \in \mathbb{R}^{l \times m}$

$$\mathbf{B}_L = \begin{bmatrix} | & | & & | \\ \mathbf{b}_0 & \mathbf{b}_1 & \cdots & \mathbf{b}_{m-1} \\ | & | & & | \end{bmatrix}, \quad \mathbf{B}_R = \begin{bmatrix} | & | & & | \\ \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_m \\ | & | & & | \end{bmatrix}. \tag{12.29}$$

This leads to the following projected least-squares problem

$$\widehat{\mathbf{A}}_{\mathbf{B}} = \underset{\mathbf{A}_{\mathbf{B}}}{\arg\min} \|\mathbf{B}_R - \mathbf{A}_{\mathbf{B}} \mathbf{B}_L\|_F^2. \tag{12.30}$$

Using the pseudoinverse, the estimator for the linear map $\widehat{\mathbf{A}}_{\mathbf{B}} \in \mathbb{R}^{l \times l}$ is defined as

$$\widehat{\mathbf{A}}_{\mathbf{B}} := \mathbf{B}_R \mathbf{B}_L^\dagger = \mathbf{B}_R \mathbf{V} \Sigma^{-1} \widetilde{\mathbf{U}}^*, \tag{12.31}$$

where $\widetilde{\mathbf{U}} \in \mathbb{R}^{l \times k}$ and $\mathbf{V} \in \mathbb{R}^{m \times k}$ are the truncated left and right singular vectors of $\mathbf{B}_L$. The diagonal matrix $\Sigma \in \mathbb{R}^{k \times k}$ contains the corresponding singular values. If $p = 0$, then $l = k$, and no truncation is needed. Next, $\widehat{\mathbf{A}}_{\mathbf{B}}$ is projected onto the left singular vectors

$$\widetilde{\mathbf{A}}_{\mathbf{B}} = \widetilde{\mathbf{U}}^* \widehat{\mathbf{A}}_{\mathbf{B}} \widetilde{\mathbf{U}} \tag{12.32a}$$

$$= \widetilde{\mathbf{U}}^* \mathbf{B}_R \mathbf{V} \Sigma^{-1}. \tag{12.32b}$$

DISTRIBUTION A: Distribution approved for public release.

The DMD modes, containing the spatial information, are then obtained by computing the eigendecomposition of $\widetilde{\mathbf{A}}_\mathbf{B} \in \mathbb{R}^{k\times k}$

$$\widetilde{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{W}}_\mathbf{B} = \widetilde{\mathbf{W}}_\mathbf{B}\boldsymbol{\Lambda}, \tag{12.33}$$

where the columns of $\widetilde{\mathbf{W}}_\mathbf{B} \in \mathbb{C}^{k\times k}$ are eigenvectors $\widetilde{\mathbf{w}}_{\mathbf{B},j}$, and $\boldsymbol{\Lambda} \in \mathbb{C}^{k\times k}$ is a diagonal matrix containing the corresponding eigenvalues λ_j. The high-dimensional DMD modes $\mathbf{W} \in \mathbb{C}^{n\times k}$ may be recovered as

$$\mathbf{W} = \mathbf{Q}\mathbf{B}_R\mathbf{V}\Sigma^{-1}\widetilde{\mathbf{W}}_\mathbf{B}. \tag{12.34}$$

The computational steps for a practical implementation are sketched in Algorithm 2.

Derivation

In the following we outline the derivation of Eq. (12.34). Recalling Eq. (12.7), the estimated transfer operator $\widehat{\mathbf{A}}$ is defined as

$$\widehat{\mathbf{A}} := \mathbf{X}_R\mathbf{X}_L^\dagger. \tag{12.35}$$

Random projections of the data matrix $\mathbf{X}$ result in an approximate orthonormal basis $\mathbf{Q}$ for the range of $\widehat{\mathbf{A}}$, as described in Sec. **??**. The sampling strategy is assumed to be efficient enough so that $\mathbf{X}_L \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}_L$ and $\mathbf{X}_R \approx \mathbf{Q}\mathbf{Q}^*\mathbf{X}_R$. Equation (12.35) then becomes

$$\widehat{\mathbf{A}} \approx \left(\mathbf{Q}\mathbf{Q}^*\mathbf{X}_R\right)\left(\mathbf{Q}\mathbf{Q}^*\mathbf{X}_L\right)^\dagger. \tag{12.36}$$

Letting $\mathbf{B}_L := \mathbf{Q}^*\mathbf{X}_L$ and $\mathbf{B}_R := \mathbf{Q}^*\mathbf{X}_R$, the projected transfer operator estimate is defined as in Eq. (12.31), $\widehat{\mathbf{A}}_\mathbf{B} := \mathbf{B}_R\mathbf{B}_L^\dagger$. Substituting this into Eq. (12.36) leads to

$$\widehat{\mathbf{A}} \approx \mathbf{Q}\widehat{\mathbf{A}}_\mathbf{B}\mathbf{Q}^*. \tag{12.37}$$

Letting $\mathbf{B}_L = \widetilde{\mathbf{U}}\boldsymbol{\Sigma}\mathbf{V}^*$ be the SVD of $\mathbf{B}_L$, and left- and right-projecting $\widehat{\mathbf{A}}_\mathbf{B}$ onto the dominant k left singular vectors $\widetilde{\mathbf{U}}$, yields

$$\widetilde{\mathbf{A}}_\mathbf{B} := \widetilde{\mathbf{U}}^*\widehat{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{U}}. \tag{12.38}$$

The eigendecomposition is given by $\widetilde{\mathbf{A}}_\mathbf{B}\widetilde{\mathbf{W}}_\mathbf{B} = \widetilde{\mathbf{W}}_\mathbf{B}\boldsymbol{\Lambda}$, as in Eq. (12.33).

(1)	$[\mathbf{Q}, \mathbf{B}] = \texttt{rqb}(\mathbf{X}, p, q)$	Randomized QB decomposition (Alg. 1).
(2)	$\mathbf{B} \to \{\mathbf{B}_L, \mathbf{B}_R\}$	Left/right low-dimensional snapshot matrix.
(3)	$[\widetilde{\mathbf{U}}, \Sigma, \mathbf{V}] = \texttt{svd}(\mathbf{B}_L, k)$	Truncated SVD.
(4)	$\widetilde{\mathbf{A}}_\mathbf{B} = \widetilde{\mathbf{U}}^*\mathbf{B}_R\mathbf{V}\Sigma^{-1}$	Least squares fit.
(5)	$[\widetilde{\mathbf{W}}_\mathbf{B}, \Lambda] = \texttt{eig}(\widetilde{\mathbf{A}}_\mathbf{B})$	Eigenvalue decomposition.
(6)	$\mathbf{W} \leftarrow \mathbf{Q}\mathbf{B}_R\mathbf{V}\Sigma^{-1}\widetilde{\mathbf{W}}_\mathbf{B}$	Recover high-dimensional DMD modes $\mathbf{W}$.

Algorithm 2: Randomized Dynamic Mode Decomposition (rDMD). Given a snapshot matrix $\mathbf{X}$ and a target rank k, the near-optimal dominant dynamic modes $\mathbf{W}$ and eigenvalues Λ are computed. The approximation quality can be controlled via oversampling p and the computation of power iterations q. An implementation in Python is available via the GIT repository https://github.com/erichson/ristretto.

DISTRIBUTION A: Distribution approved for public release.

Let $\widehat{\mathbf{W}}_\mathbf{B} := \mathbf{B}_R \mathbf{V}\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}$. It is simple to verify that this is an eigenvector of $\widehat{\mathbf{A}}_\mathbf{B}$:

$$\begin{aligned}\widehat{\mathbf{A}}_\mathbf{B}\widehat{\mathbf{W}}_\mathbf{B} &= \mathbf{B}_R \mathbf{B}_L^\dagger \mathbf{B}_R \mathbf{V}\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B},\\ &= \mathbf{B}_R \mathbf{V}\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{A}}_\mathbf{B}\widehat{\mathbf{W}}_\mathbf{B},\\ &= \mathbf{B}_R \mathbf{V}\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}\boldsymbol{\Lambda},\\ &= \widehat{\mathbf{W}}_\mathbf{B}\boldsymbol{\Lambda}.\end{aligned}$$

Substituting the eigendecomposition of $\widehat{\mathbf{A}}_\mathbf{B}$ in Eq. (12.37) and right-multiplying by $\mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B}$ leads to

$$\widehat{\mathbf{A}}\mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B} \approx \mathbf{Q}\widehat{\mathbf{W}}_\mathbf{B}\boldsymbol{\Lambda}, \tag{12.39}$$

so that identification with $\widehat{\mathbf{A}}\mathbf{W} = \mathbf{W}\widetilde{\boldsymbol{\Lambda}}$ verifies the claim in Eq. (12.34) on the eigendecomposition of $\widehat{\mathbf{A}}$:

$$\mathbf{W} \approx \mathbf{Q}\mathbf{B}_R \mathbf{V}\boldsymbol{\Sigma}^{-1}\widetilde{\mathbf{W}}_\mathbf{B}, \qquad \widetilde{\boldsymbol{\Lambda}} \approx \boldsymbol{\Lambda}.$$

12.8. Numerical Results

In the following we present numerical results demonstrating the performance of the randomized dynamic mode decomposition (rDMD). First, we provide some visual results for a flow behind a cylinder and climate data. Then, we evaluate and compare the computational time and accuracy of randomized algorithm to the deterministic algorithm.

All computations are performed using Amazon Web Services (AWS). We use a G3 instance with 16 Xeon E5-2686 CPUs and 1 NVIDIA Tesla M60 GPU. The underlying numerical linear algebra routines are accelerated using the Intel Math Kernel Library (MKL).

12.8.1. Fluid Flow Behind a Cylinder

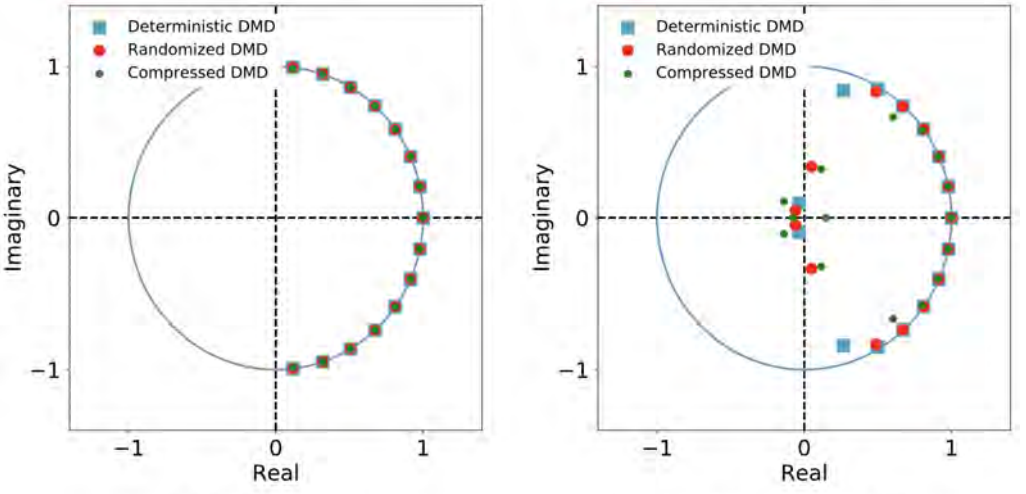
As a canonical example, we consider the fluid flow behind a cylinder at Reynolds number $Re = 100$. The data consist of a sequence of 151 snapshots of fluid vorticity fields on a 449×199 grid†, computed using the immersed boundary projection method (Taira & Colonius 2007*b*; Colonius & Taira 2008*b*). The flow features a periodically shedding wake structure and the resulting dataset is low-rank. While this data set poses no computational challenge, it demonstrates the accuracy and quality of the randomized approximation on an example that builds intuition. Flattening and concatenating the snapshots horizontally yields a matrix of dimension $\mathbf{X} \in \mathbb{R}^{89,351\times 151}$, i.e., the columns are the flattened snapshots $\mathbf{x} \in \mathbb{R}^{449\times 199}$.

We compute the low-rank DMD approximation using $k = 15$ as the desired target rank. Figure 61(a) shows the DMD eigenvalues. The proposed randomized DMD algorithm (with $p = 10$ and $q = 0$), and the compressed DMD algorithm (with $l = 1000$) faithfully capture the eigenvalues. Overall, the randomized algorithm leads to a 6 fold speedup compared to the deterministic DMD algorithm. Further, if the singular value spectrum is slowly decaying, the approximation accuracy can be improved by computing additional power iterations q. To further contextualize the results, Fig. 62 shows the leading six DMD modes in absence of noise. The randomized algorithm faithfully reveals the coherent structures, while requiring considerably less computational resources.

Next, the analysis is repeated in presence of additive white noise with a signal-to-noise (SNR) ratio of 10. Figure 61(b) shows distinct performance of the different algorithms. The deterministic algorithm performs most accurate, capturing the first

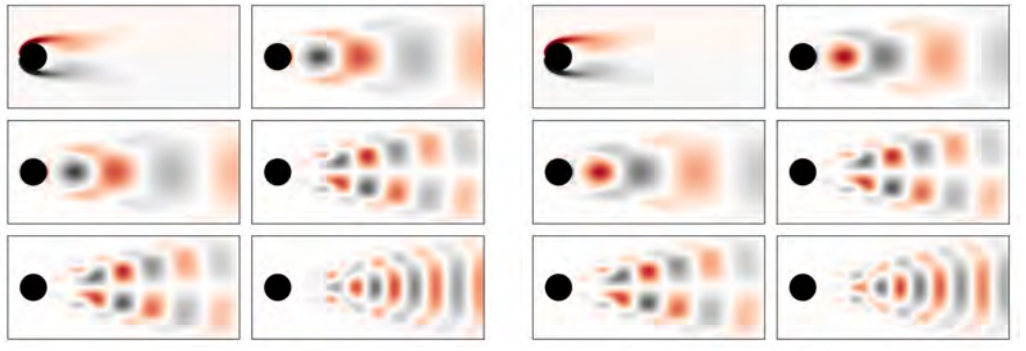
† Data for this example may be downloaded at dmdbook.com/DATA.zip.

DISTRIBUTION A: Distribution approved for public release.



(a) No noise. (b) Noisy with SNR of 10.

Figure 61: DMD eigenvalues for fluid flow behind a cylinder. Both the compressed and randomized DMD algorithms capture the eigenvalues in the absence of noise (a). In the presence of white noise with SNR of 10, rDMD performs better than sampling rows (b).

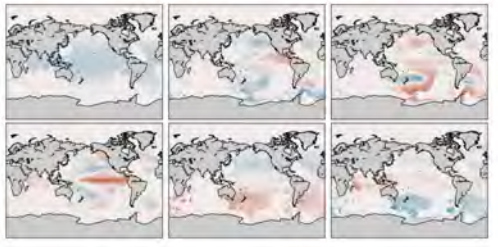


(a) Deterministic DMD modes. (b) Randomized DMD modes.

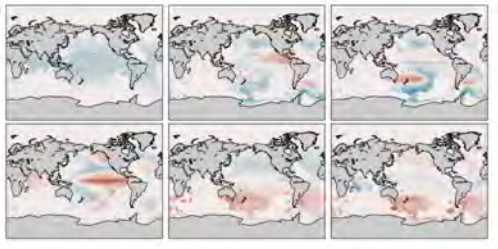
Figure 62: Leading dynamic modes extracted from the fluid flow behind a cylinder.

eleven eigenvalues. The randomized DMD algorithm reliably captures the first nine eigenvalues, while the compressed algorithm only accurately captures seven of the eigenvalues. This is, despite the fact that the compressed DMD algorithm uses a large sketched snapshot sequence of dimension 1000×151, i.e., $1,000$ randomly selected rows out of the $89,351$ rows of the flow dataset are used. For comparison the randomized DMD algorithm provides a more satisfactory approximation using $l = 25$ and 2 additional power iterations, i.e., the sketched snapshot sequence is only of dimension 25×151. The results show that the randomized DMD algorithm yields a more accurate approximation, while allowing for higher compression rates. This is because the approximation quality of the randomized DMD algorithm depends on the intrinsic rank of the data and not on the ambient dimensions on the measurement space.

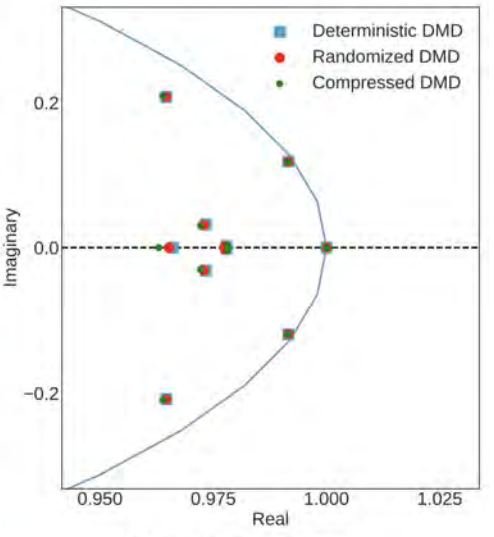
DISTRIBUTION A: Distribution approved for public release.



(a) Deterministic DMD modes.



(b) Blocked Randomized DMD modes.



(c) DMD eigenvalues.

Figure 63: Leading dynamic modes extracted from the aggregated high-resolution sea surface dataset. No distinct differences can be obtained. Further, the randomized DMD algorithm captures faithfully the eigenvalues. In contrast, the compressed DMD algorithm provides only a crude approximation for the eigenvalues.

12.8.2. Sea Surface Data

We now compute the dynamic mode decomposition on the high-resolution sea surface temperature (SST) data. The SST data are widely studied in climate science for climate monitoring and prediction, providing an improved understanding of the interactions between the ocean and the atmosphere (Reynolds *et al.* 2007, 2002; Smith & Reynolds 2005). Specifically, the daily SST measurements are constructed by combining infrared satellite data with observations provided by ships and buoys. In order to account and compensate for platform differences and sensor errors, a bias-adjusted methodology is used to combine the measurements from the different sources. Finally, the spatially complete SST map is produced via interpolation. A comprehensive discussion of the data is provided in (Reynolds *et al.* 2002).

The data are provided by the National Oceanic and Atmospheric Administration (NOAA) via their web site at `https://www.esrl.noaa.gov/psd/`. Data are available for the years from 1981 to 2018 with a temporal resolution of 1 day and a spatial grid resolution of 0.25°. In total, the data consists of $m = 13,149$ temporal snapshots which measure the daily temperature at $1440 \times 720 = 1,036,800$ spatial grid points. Since we omit data over land, the ambient dimension reduces to $n = 691,150$ spatial measurements in our analysis. Concatenating the reduced data yield a 36GB data matrix of dimension $\mathbf{X} \in \mathbb{R}^{691,150 \times 13,149}$, which is sufficiently large to test scaling.

12.8.3. Aggregated (Weekly Mean) Data

The full data set outstrips the available fast memory required to compute the deterministic DMD. Thus, we perform the analysis on an aggregated data set first, in order to compare the randomized and deterministic DMD algorithms. Therefore, we compute the weekly mean temperature, which reduces the number of temporal snapshots

DISTRIBUTION A: Distribution approved for public release.

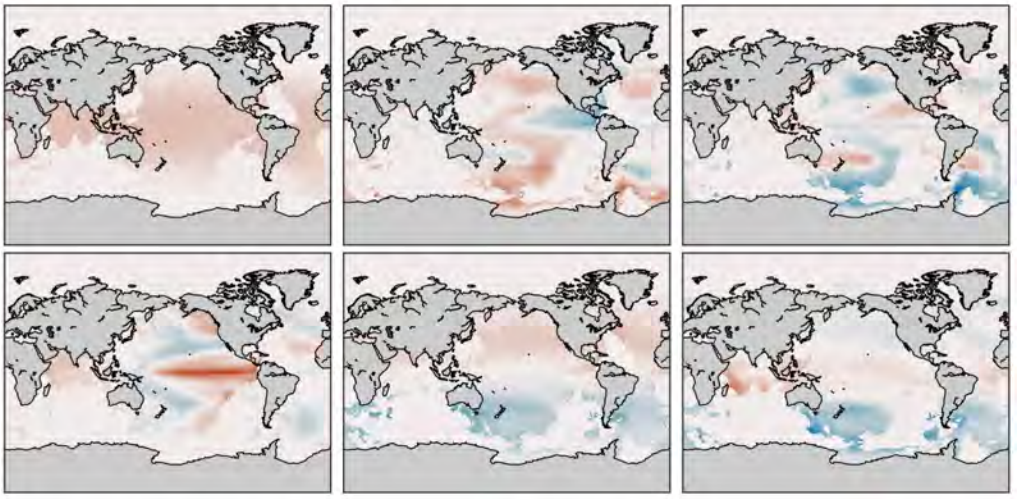


Figure 64: Leading dynamic modes of the full high-resolution SST dataset.

to $m = 1,878$. Figure 63(c) shows the corresponding eigenvalues. Unlike the eigenvalues obtained via the compressed DMD algorithm, the randomized DMD algorithm provides an accurate approximation for the dominant eigenvalues. Next, Fig. 63(a,b) show the extracted DMD modes. Indeed, the randomized modes faithfully capture the dominant coherent structure in the data.

12.8.4. **Full Data**

The blocked randomized DMD algorithm allows us to extract the DMD modes from the high-dimensional data set. While the full data-set does not fit into fast memory, it is only required that we can access some of the rows in a sequential manner. Computing the $k = 15$ approximation using $b = 4$ blocks takes about 150 seconds. The resulting modes are shown in Fig. 64. Here, the leading modes are similar to the modes extracted from the aggregated data set. However, subsequent modes may provide further insights which are not revealed by the aggregated data set.

12.8.5. **Computational Performance**

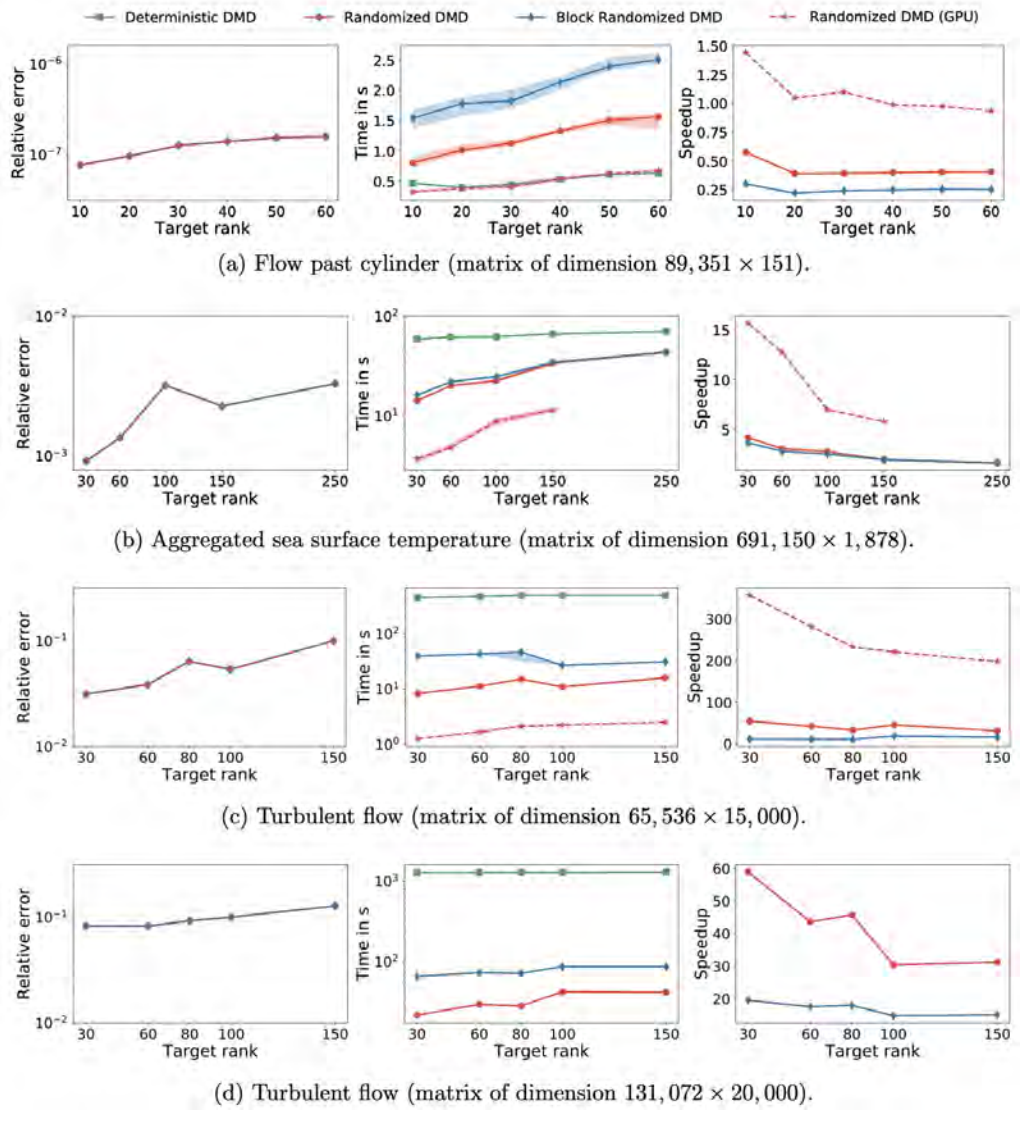
Next, we evaluate the computational performance of the randomized algorithms in terms of both time and accuracy. We measure the accuracy by computing the relative error of the randomized DMD compared to the approximation produced by the deterministic DMD algorithm

$$\rho(\widehat{\mathbf{X}}_{(\mathrm{DMD})}, \widehat{\mathbf{X}}_{(\mathrm{rDMD})}) = \frac{\|\widehat{\mathbf{X}}_{(\mathrm{DMD})} - \widehat{\mathbf{X}}_{(\mathrm{rDMD})}\|_F}{\|\widehat{\mathbf{X}}_{(\mathrm{DMD})}\|_F}, \tag{12.40}$$

where $\widehat{\mathbf{X}}$ denotes the reconstructed snapshot matrix using either the deterministic or randomized DMD algorithm. Here, we use a (high-performance) partial SVD algorithm to compute the k deterministic modes (Lehoucq *et al.* 1997).

Figure 65 summarizes the computational performance, for three different examples, and for varying target ranks. First, Figure 65 (a) shows the performance for the flow past cylinder. The snapshot matrix is tall and skinny. In this setting there is no computational advantage of the randomized algorithm over the deterministic algorithm. This is because the data matrix is overall very small and the (deterministic) partial SVD is highly-efficient

DISTRIBUTION A: Distribution approved for public release.



(a) Flow past cylinder (matrix of dimension $89,351 \times 151$).

(b) Aggregated sea surface temperature (matrix of dimension $691,150 \times 1,878$).

(c) Turbulent flow (matrix of dimension $65,536 \times 15,000$).

(d) Turbulent flow (matrix of dimension $131,072 \times 20,000$).

Figure 65: Average runtimes and errors, over 20 runs, for varying target ranks. Further, the gained speedup compared to the deterministic algorithm (baseline) is shown.

for tall and skinny matrices. Importantly, we stress that the randomized DMD algorithm provides a very accurate approximation for the flow past cylinder. That is, because this fluid flow example has a fast decaying singular value spectrum.

Second, Figure 65 (b) shows the performance for the aggregated SST data. This snapshot matrix is also tall and skinny, yet its dimensions are substantially larger than those of the previous example. Here, we start to see the computational benefits of the randomized DMD algorithm. We gain about a speedup factor of 3-5 for computing the low-rank DMD, while maintaining a good accuracy. Note that this problem has a slower decaying singular value spectrum and thus the accuracy is poorer than in the previous examples. This example also demonstrates the performance boost by using a GPU accelerated implementation of the randomized DMD. Clearly, we can see a significant

DISTRIBUTION A: Distribution approved for public release.

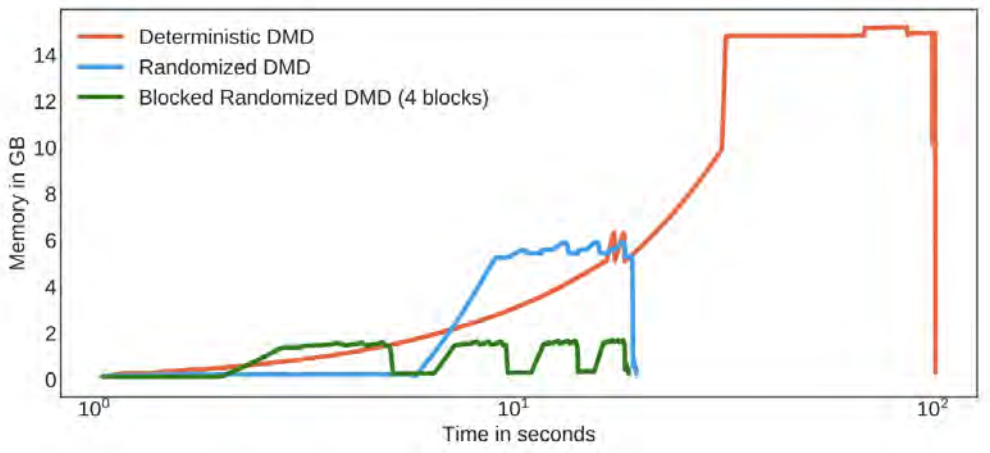


Figure 66: A profile of the memory usage vs runtime of the different DMD algorithms.

speedup by a factor of about 10-15 for computing the top 30 to 60 dynamic modes. Note that we run out of memory for target ranks larger than $k > 150$.

Third, Figure 65 (c) and (d) shows the performance for a turbulent flow dataset. The fluid flow was obtained with the model implementation of (Chirila 2018), which is based on the algorithm presented by (Wang *et al.* 2013). This dataset shows the advantages and disadvantages of the randomized algorithm. Indeed, we see some substantial gains in terms of the computational time, i.e., the GPU-accelerated algorithms achieve speedups of factors > 300. However, the accuracy is less satisfactory due to the slow decaying singular value spectrum of the data. This example illustrates the limits of the randomized algorithm, i.e., we need to assume that the data set features low-rank structure and has a reasonably fast decaying singular value spectrum.

The reader might have noticed that the blocked randomized scheme (using 2 blocks) requires more computational time to compute the approximate DMD. This is because the data matrices considered in the above examples fit into the fast memory. Hence, there is little advantage in using the blocked scheme here. However, the memory requirements are often more important than the absolute computational time. Figure 66 shows a profile of the memory usage vs runtime. The blocked randomized DMD algorithm requires only a fraction of the memory, compared to the deterministic algorithm, to compute the approximate low-rank dynamic mode decomposition. This can be crucial to carry-out the computations on mobile platforms, on GPUs (as seen in Figure 65 (c)), or to scale the computations to very large applications.

12.9. Conclusion

Randomness as a computational strategy has recently been shown capable of efficiently solving many standard problems in linear algebra. The need for highly efficient algorithms becomes increasingly important in the area of 'big data'. Here, we have proposed a novel randomized algorithm for computing the low-rank dynamic mode decomposition. Specifically, we have shown that DMD can be embedded in the probabilistic framework formulated by (Halko *et al.* 2011). This framework not only enables computations at scales far larger than what was previously possible, but it is also modular, flexible, and the error can be controlled. Hence, it can be also utilized as a framework to embed other innovations around the dynamic mode decomposition, for instance, see (Kutz *et al.*

DISTRIBUTION A: Distribution approved for public release.

2016*a*) for an overview.

The numerical results show that randomized dynamic mode decomposition (rDMD) has computational advantages over previously suggested probabilistic algorithms for computing the dominant dynamic modes and eigenvalues. More importantly, we showed that the algorithm can be executed using a blocked scheme which is memory efficient. This aspect is crucial in order to efficiently deal with massive data which are too big to fit into fast memory. Thus, we believe that the randomized DMD framework will provide a powerful and scalable architecture for extracting dominant spatiotemporal coherent structures and dynamics from increasingly large-scale data, for example from epidemiology, neuroscience, and fluid mechanics.

DISTRIBUTION A: Distribution approved for public release.

13. Supervised Machine Learning for Flow Regression

13.1. Introduction

In recent years, machine learning approaches have been successfully applied to a range of nonlinear and chaotic fluid flow problems. These applications have leveraged the powerful capability of supervised machine learning to serve as an universal approximator for the nonlinear relationship between input and output data (Kutz 2017; Brunton *et al.* 2020*b*; Brenner *et al.* 2019; Kreinovich 1991; Hornik 1991; Cybenko 1989; Baral *et al.* 2018). Within a short time frame, these efforts have produced a vast collection of studies with a wide selection of machine learning techniques. For an aspiring fluid mechanician, a naive dive into the ocean of machine learning models may be not only a bit intimidating but also nauseating. The objective of this paper is to present some of the notable techniques and provide guidelines on their uses for canonical regression problems for laminar and turbulent flows.

Machine learning has been applied to problems, including turbulence modeling for Reynolds Averaged Navier–Stokes (Ling *et al.* 2016; Duraisamy *et al.* 2019) and large-eddy simulations (Gamahara & Hattori 2017; Maulik & San 2017), modeling the dynamics of fluid flows (San & Maulik 2018; Lui & Wolf 2019; Srinivasan *et al.* 2019; Murata *et al.* 2020), flow estimation (Salehipour & Peltier 2019; Leoni *et al.* 2018) and reconstruction (Fukami *et al.* 2019*a*,*b*; Deng *et al.* 2019; Erichson *et al.* 2020), and optimization of flow control designs (Duriez *et al.* 2017; Rabault *et al.* 2019). These studies have shown that machine learning techniques can accurately model the complex input-output relationship within nonlinear fluid flows. In essence, these studies benefit from the ability of machine learning techniques to extract the nonlinear input-output mapping from data. Although great strides have been made to date, the field of fluid mechanics has not seen concrete presentations of benchmarks and guidelines to construct machine learning models. As such, we can benefit from a systematic procedure to develop machine learning models appropriate for fluid dynamics. In the present paper, we discuss how one may select a suitable approach from a large family of machine learning models developed in the computer science field.

We consider a systematic procedure to develop supervised machine learning models and assess them on canonical fluid flow problems. With supervised machine learning, training data comprised of input $\boldsymbol{x}$ and output (solution) $\boldsymbol{y}$ data are needed. The machine learning models are then trained using the given data sets to seek a nonlinear mapping $\boldsymbol{y} \approx \mathcal{F}(\boldsymbol{x}; \boldsymbol{w})$. This can be mathematically posed as a problem to find the optimal weights $\boldsymbol{w}$. Here, we consider four machine learning architectures to assess their characteristics, accuracy, computational cost, and robustness for laminar flow problems. Furthermore, we also examine the use of convolutional neural network (CNN) for super-resolution analysis. An overview of the current study is presented in figure 67 with illustrations of the example problems.

In section 13.2, we introduce five machine learning models that have been considered for fluid dynamics problems: (1) multi-layer perceptron, (2) random forest, (3) support vector regression, (4) extreme learning machine, and (5) CNN. These models are applied to regression problems in section 13.3. Namely, these problems are (1) the estimation of drag and lift coefficients for laminar flows, (2) the estimation of laminar wake flow from limited measurements, and (3) the super-resolution analysis for coarse flow data. We present the procedures to develop machine learning models and guidelines to ensure the models are accurate and robust. At last in section 13.4, we offer concluding remarks and possible extensions to currently available machine learning models.

DISTRIBUTION A: Distribution approved for public release.

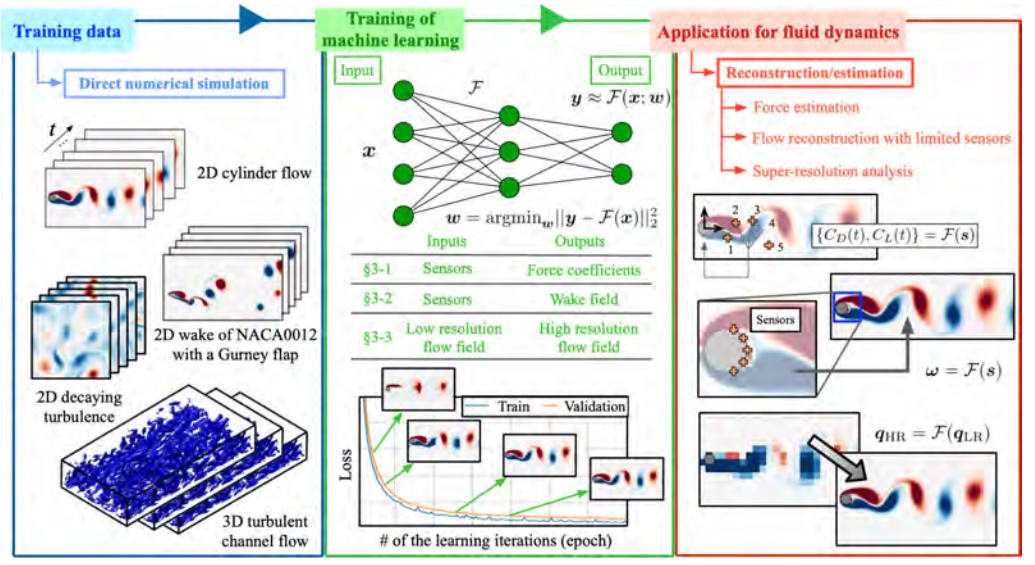


Figure 67: Overview of supervised machine learning problems for fluid flows discussed in this paper. We focus on reconstruction and estimation tasks for direct numerical simulation data.

Table 1: Strengths and weaknesses of the machine learning models covered in this paper.

ML models	Strengths	Weaknesses	Examples
MLP (§2-1-1)	Compare easily (Commonly used)	Curse of dimensionality	C_D,C_L estimation (§3-1) Wake reconstruction (§3-2)
RF (§2-1-2)	Interpretability (Feature importance)	Curse of dimensionality	C_D,C_L estimation (§3-1) Wake reconstruction (§3-2)
SVR (§2-1-3)	Avoids overfitting (Max margin problem)	Curse of dimensionality	C_D,C_L estimation (§3-1) Wake reconstruction (§3-2)
ELM (§2-1-4)	Fast computation	Noise robustness	C_D,C_L estimation (§3-1) Wake reconstruction (§3-2)
CNN (§2-1-5)	Low computational cost Handles big data	Limited for uniform grid	Super resolution (§3-3)

13.2. Machine learning methods

Constructing a supervised machine learning model requires a few steps. First, we must prepare the training data comprised of the input $\boldsymbol{x}$ and output $\boldsymbol{y}$. Second, a model $\mathcal{F}$ itself must be chosen based on its properties. Given the model and the available data set, we can then optimize the weights $\boldsymbol{w}$ to train the model, as illustrated in the green box of

DISTRIBUTION A: Distribution approved for public release.

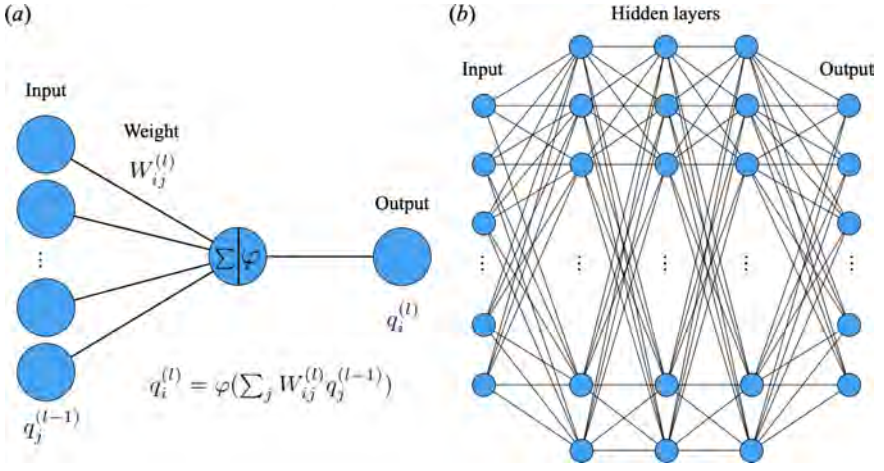


Figure 68: (*a*) A perceptron. (b) Multi-layer perceptron is an aggregate of perceptrons.

figure 67. In this paper, we consider five machine learning models $\mathcal{F}$ as presented in section 13.2.1. We also discuss the choice of norms for the loss function needed to optimize the model weights $\boldsymbol{w}$ in section 13.2.7 and emphasize the need for cross validation to prevent overfitting in section 13.2.8.

13.2.1. Models

For the example problems that will be considered in this paper, we can select a machine learning model from a very large number of available models (Brunton & Kutz 2019). In this section, we examine five of the representative machine learning models. Their strengths and weaknesses are summarized in table 1. We choose each of these machine learning models based on their strengths for each of the examples studied later. Here, we start our presentation of the models with the *multi-layer perceptron*, which has been widely used in fluid dynamics, yielding a large number of references. Next, we discuss an interpretable machine learning model, *random forest* that can reveal the feature importance (importance of the input influence). Also considered is the *support vector regression*, which is robust against overfitting. Highlighting the computational aspect, we cover the *extreme learning machine* founded on the shallow network and the least squares formulations. In this paper, these four machine learning models are compared with each other in terms of accuracy, computational cost, and robustness for canonical laminar flow problems. For fairness in comparison, parameters of each machine learning model are optimized as explained later. We also present the *convolutional neural network* which reveals its strength in handling high-dimensional input and output data. Below, let us briefly discuss each aforementioned machine learning model.

13.2.2. Multi-layer perceptron

A multi-layer perceptron (MLP) developed by Rumelhart *et al.* (Rumelhart *et al.* 1986) is pervasive in a wide range of sciences. The basic concept of MLP is motivated by the structure of biological neural circuits. An MLP has been successfully applied to diverse problems, including regression, classification, and speech recognition (Domingos 2012). We have also seen the use of MLP for turbulence modeling, reduced order modeling, and flow reconstruction in fluid dynamics (Ling *et al.* 2016; Lui & Wolf 2019; Maulik *et al.* 2019; Yu & Hesthaven 2019).

DISTRIBUTION A: Distribution approved for public release.

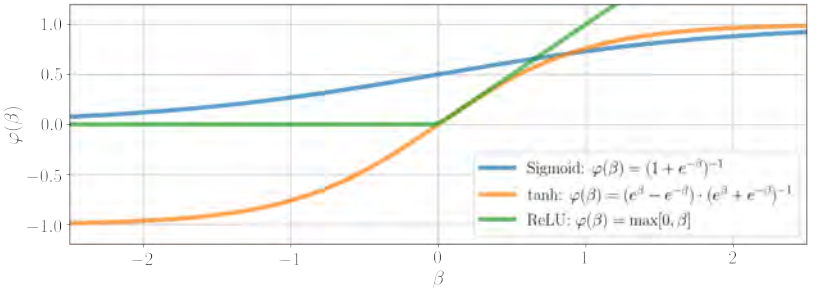


Figure 69: An example of activation functions $\varphi(\beta)$. In the present papar, we use ReLU $\varphi(\beta) = \max[0, \beta]$.

A minimum unit is called a perceptron as shown in figure 68(a). The input data from the $(l-1)$th layer are multiplied a weight $\boldsymbol{W}$. These inputs are linearly combined and passed through a nonlinear activation function φ:

$$q_i^{(l)} = \varphi(\sum_j W_{ij}^{(l)} q_j^{(l-1)}). \tag{13.1}$$

An MLP is constructed by connecting these perceptrons in layers, as illustrated in figure 68(b). Weights for all links W_{ij} are optimized to minimize a positive-definite loss (cost) function E by back propagation (Kingma & Ba 2014) such that $\boldsymbol{w} = \mathrm{argmin}_{\boldsymbol{w}}[E(\boldsymbol{y}, \mathcal{F}(\boldsymbol{x}; \boldsymbol{w}))]$.

The activation function φ is generally chosen to be monotonically increasing nonlinear functions. Some of the widely used functions of sigmoid $\varphi(\beta) = (1 + e^{-\beta})^{-1}$, hypabolic tangent (tanh) $\varphi(\beta) = (e^{\beta} - e^{-\beta}) \cdot (e^{\beta} + e^{-\beta})^{-1}$, and ReLU $\varphi(\beta) = \max[0, \beta]$, are shown in figure 69. All of these activation functions seem to perform well for shallow networks. For deep networks, ReLU is known to be a good candidate (Nair & Hinton 2010). Details on the specific choices and their influence on networks can be found in Haykin (Haykin 1998) and Nair and Hinton (Nair & Hinton 2010). In the MLP formulation, we use ReLU activation function (we also use ReLU later in the extreme learning machine and convolutional neural network formulations).

Let us present the steps to optimize the MLP parameters (numbers of units and layers) for actual examples of force estimation and wake reconstruction based on wake measurements, as discussed later in sections 3.1 and 3.2. As mentioned above, the aim here is to find the optimal numbers of units and layers for the multi-layer perceptron. In this case, the following procedure can be used:

(i) We fix the number of layers n_{layer} and find the number of units n^*_{unit} that minimizes the loss function. Multiple cross validations should be used.

(ii) We then fix the number of units n^*_{unit} and find the number of layers n^*_{layer} that minimizes the loss function. Multiple cross validations should be used.

(iii) The found parameter set $(n_{\mathrm{unit,opt}}, n_{\mathrm{layer,opt}}) = (n^*_{\mathrm{unit}}, n^*_{\mathrm{layer}})$ are used as the optimized parameters for the machine learning model. Depending on the problem, one can iterate over steps 1 and 2.

We also apply the same procedure to other machine learning models in this paper, as explained later.

The optimized parameters for each machine learning model of the present study are summarized in table 2. The details on the choice of each parameter are discussed later. Note that the proposed order here, e.g., units first and layer next of the MLP, is just

DISTRIBUTION A: Distribution approved for public release.

Table 2: Examples of optimized parameters of the machine learning model used in this paper.

ML models	1st parameter	2nd parameter	3rd parameter
MLP	Unit	Layer	—
RF	Depth	# of decision trees N_{Tree}	—
SVR	Epsilon tube e	Coefficient of error term C	—
ELM	Unit	—	—
CNN	Layer	# of filters	Filter size

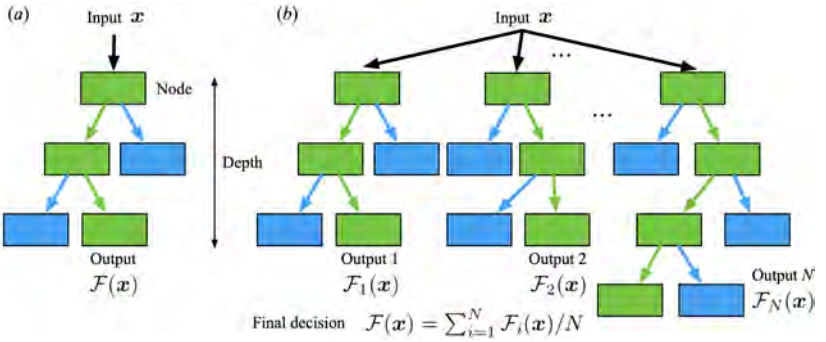


Figure 70: (a) Decision tree. (b) Random forest, which can be regarded as ensemble average of decision trees.

an example order taken in the present study. Users are able to change the order within the optimization process. As the other methods for hyper parameter decision, users can also consider the use of theoretical optimization methods, e.g., hyperopt (Bergstra *et al.* 2013) and Baysian optimization (Brochu *et al.* 2009; Maulik *et al.* 2020*b*).

13.2.3. Random forest

Next, let us consider a decision tree (Quinlan 1986) which is the basis for a random forest (RF). A decision tree has a flow-chart structure as shown in figure 70(a). The boxes in figure 70(a) are called nodes and represent the group of features and data. The learning objective of a decision tree is to seek *if/then/else* rules for obtaining the desired output $\boldsymbol{y}$, as branches (the allows in figure 70(a)). Let us present a simple example for understanding the construction of a decision tree: a model to estimate the amount of the water a person drinks when the temperature is T. If a person drinks a lot of water when the temperature T is more than the threshold T_{th}, the rule for water consumption would depend on $T \geqslant T_{\mathrm{th}}$ and $T < T_{\mathrm{th}}$. These rules are illustrated by the green and blue lines of

DISTRIBUTION A: Distribution approved for public release.

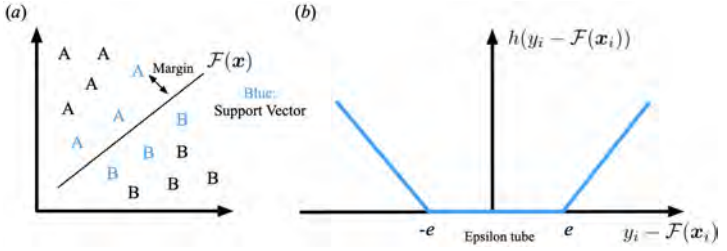


Figure 71: (a) Support vector machine with two-group classification. (b) The concept of epsilon tube.

figure 70(a). In this example, humidity is also another factor that influences the amount of water consumption. We can consider an increased number of influential variables so that the accuracy of the model is improved. To incorporate the complexity, the depth of a decision tree can be increased as shown in figure 70(a). In summary, the boxes and lines of figure 70(a) can be employed to construct rules for obtaining the output of the machine learning model $\mathcal{F}(\boldsymbol{x})$.

The decision tree can be explainable because the internal structure is comprised of simple *if/then/else* rules. However, it is known that the learning process of a decision tree is unstable and prone to overfitting due to the vulnerability with respect to small variations in the data (Dwyer & Holte 2007). To address these issues, an ensemble learning approach (Opitz & Maclin 1999) is often adopted. *Random forest* (RF) (Breiman 2001) is one of these ensemble learning techniques for a decision tree. The output value of RF is given an average score of the predictions of all decision trees, as shown in figure 70(b). As the optimized parameters, we focus on the depth and number of decision trees N_{Tree} (vertical and horizontal direction in figure 70(b)) as summarized in table 2, respectively. We note in passing that RF contains considerable hyper parameters, e.g., maximum depths of the tree and minimum number of samples required to be at the leaf node. Users can also consider these parameters for implementation.

13.2.4. Support vector regression

Support vector regression (SVR) (Smola & Schölkopf 2004) is an extension of support vector machine used for classification (Vapnik & Lerner 1963). This method has been also utilized for various problems that are similarly tackled by MLPs and RFs (Bishop 2006). In the field of fluid dynamics, the support vector machine has been applied to turbulence modeling (Ling & Templeton 2015) and reduced-order modeling (Carlberg *et al.* 2019).

Here, let us consider a two-dimensional classification problem for demonstration, as shown in figure 71(a). To classify data into two groups of A and B, we utilize only the components around the borders represented as blue components called the support vector. Next, to avoid misclassification, the distance between the border and neighboring components should be maximized. This problem is called the *maximum margin problem*. We can also use the same concepts for regression problems. Mathematically speaking, the objective of the learning process is to establish the function $\mathcal{F}(\boldsymbol{x}) = \boldsymbol{W}^T\phi(\boldsymbol{x})$ between input and output data by mapping training data into the high-dimensional feature space for regression. Here, the $\boldsymbol{W}$ and ϕ are the high-dimensional weight and map function, respectively. Seeking the border in figure 71(a) corresponds to finding the function $\mathcal{F}$. Note that we need to make this border in high-dimensional feature space (although the present example in figure 71(a) is an illustration for a low-dimensional classification

DISTRIBUTION A: Distribution approved for public release.

problem). This mapping is determined by solving an optimization problem of

$$\boldsymbol{W} = \mathrm{argmin}_{\boldsymbol{W}} \left[\frac{1}{2} ||\boldsymbol{W}||^2 + C \sum_{i=1}^{N} h(y_i - \mathcal{F}(\boldsymbol{x}_i; \boldsymbol{W})) \right] \tag{13.2}$$

$$\text{subject to} \quad h(y_i - \mathcal{F}(\boldsymbol{x}_i)) = \max(0, |y_i - \mathcal{F}(\boldsymbol{x}_i)| - e), \tag{13.3}$$

where h is the e-insensitive loss function and C is a coefficient of the error term, respectively. We are able to change the importance of each term by tuning the coefficient C. The schematic of function h is shown in figure 71(b). As shown here, the error $y - \mathcal{F}(\boldsymbol{x})$ (horizontal axis) within the constraint called epsilon tube is judged to achieve an error of $h(y - \mathcal{F}(\boldsymbol{x})) = 0$ (vertical axis). For instance, with the case that the output of machine learning model $\mathcal{F}(\boldsymbol{x})$ shows slight difference from y because of a noisy input, the error $h(y - \mathcal{F}(\boldsymbol{x}))$ is converted to zero if the error value is within the constraint by the epsilon tube. Thus, the epsilon tube can be regarded as the allowance against a noisy input. We can tune the allowance (tolerance) against noise through the loss function by incorporating e in the loss function (3). Here, we optimize C and e through the optimization procedure outlined in section 2.1.1.

13.2.5. Extreme learning machine

Various machine learning models with reduced computational requirements have been proposed. Such techniques are often simplified against the original formulations by sparsification and randomization. These ideas are particularly attractive for fluid dynamics since the amount of training data is enormous in size. In this paper, we consider the extreme learning machine (ELM) (Huang *et al.* 2004), which is a simplified version of the MLP. Its computational cost is significantly lower than the original MLP due to its simple construction. The basic structure of ELM is similar to MLP with only a single hidden layer. First, weights matrix between input and hidden layers with n nodes $\boldsymbol{W}_1 = [\boldsymbol{w}_1^1, \boldsymbol{w}_2^1, \cdots, \boldsymbol{w}_n^1]^{\mathrm{T}}$ are initialized by non-zero random values. This weights matrix and activation function φ are multiplied for input data matrix $\boldsymbol{q}_{\mathrm{in}}$ in a manner similar to the MLP.

In ELM, the output data matrix of the first layer $\boldsymbol{q}_h$ is expressed as

$$\boldsymbol{q}_h = \varphi(\boldsymbol{W}_1 \boldsymbol{q}_{\mathrm{in}}). \tag{13.4}$$

The weights matrix between hidden and output layers $\boldsymbol{W}_2 = [\boldsymbol{w}_1^2, \boldsymbol{w}_2^2, \cdots, \boldsymbol{w}_n^2]^{\mathrm{T}}$ are determined through the least-square formulation (pseudoinverse) such that

$$\boldsymbol{W}_2 = \boldsymbol{q}_h^* \boldsymbol{q}_{\mathrm{out}}^T, \tag{13.5}$$

where $*$ denotes the Moore–Penrose pseudoinverse (Albert 1972; Serre 2002) and $\boldsymbol{q}_{\mathrm{out}}$ is the output data matrix of the last layer. Only the number of units is optimized in this paper. We refer readers to Maulik and San (Maulik & San 2017) for additional details.

13.2.6. Convolutional neural network

In recent years, convolutional neural networks (CNN) (LeCun *et al.* 1998) have become heavily utilized, especially for image recognition tasks. One of the attractive functions of CNNs is their ability to handle high-dimensional data without encountering the curse of dimensionality, by incorporating filtering operations. Such capability has led to the increased uses of CNN, including the field of fluid dynamics (Salehipour & Peltier 2019; Fukami *et al.* 2019a,b).

The schematic of two-dimensional CNN is illustrated in figure 72. Now, let us consider an input data $\boldsymbol{x} = q^{(0)}$ which has $L \times L$ positions. The process of CNN can be expressed

DISTRIBUTION A: Distribution approved for public release.

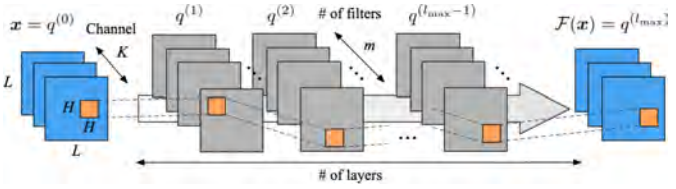


Figure 72: Two-dimensional convolutional neural network.

Table 3: Commonly used loss functions for regression tasks.

Error	L_1 error	L_2 error	L_2 logarithmic error
Definition	$\|\|\boldsymbol{q}_{\mathrm{Ref}} - \boldsymbol{q}_{\mathrm{ML}}\|\|$	$\|\|\boldsymbol{q}_{\mathrm{Ref}} - \boldsymbol{q}_{\mathrm{ML}}\|\|_2$	$\|\|\log(1+\boldsymbol{q}_{\mathrm{Ref}}) - \log(1+\boldsymbol{q}_{\mathrm{ML}})\|\|_2$
Note	Not sensitive for an outlier	Strict for an outlier	Tends to overestimate the value

in a procedural manner from $q^{(l-1)}$ to $q^{(l)}$, where l $(0 \leqslant l \leqslant l_{\max})$ represents the index of the network layer. The mathematical expression for $q^{(l)}$ is given as

$$q_{ijm}^{(l)} = \varphi\left(\sum_{k=0}^{K-1}\sum_{p=0}^{L-1}\sum_{s=0}^{L-1} h_{pskm}^{(l)} q_{i+p,j+s,k}^{(l-1)}\right), \tag{13.6}$$

where $q^{(l_{\max})} = \mathcal{F}(\boldsymbol{x})$ and K is the number of variables per each position of data. The orange $H \times H$ square in the schematic represents the filter h. The CNN can be considered as a framework to optimize the filtering coefficients in the above equation, which is analogous to the weights of MLPs as formulated as (13.1). The concept of filters in CNN is called weight sharing because the filter of size $H \times H$ is shared for whole image field of size $L \times L$. By using this weight sharing instead of the fully-connected weights in the MLP, we can handle big data with significantly lower computational costs. In the current work, three-dimensional CNN (with a three-dimensional filter of size $H \times H \times H$) is also utilized for the example of super-resolution reconstruction. Moreover, we utilize a customized CNN called hybrid downsampled skip-connection and multi-scale (DSC/MS) model for super-resolution analysis, which will be described in detail later.

13.2.7. Choice of norms in loss function

Let us discuss briefly the influence of loss functions on the performance of machine learning models. During the training process, supervised machine learning models aim to minimize the error between their output $\mathcal{F}(\boldsymbol{x})$ and an answer $\boldsymbol{y}$. Therefore, the choice of norm influences the updating procedure for the weights of machine learning models. We list commonly used loss functions for regression problems in table 3. In the present study, we use the L_2 norm error. Due to its definition including squared procedure, squared error is the strict measurement including outliers. On the other hand, the L_1 norm error is not as sensitive for outliers in the data set. The squared logarithmic error is generally utilized for cases where underestimation should be avoided (e.g., prediction of the number of customers for a restaurant, in which case overestimation is preferred). Alternatively, we

DISTRIBUTION A: Distribution approved for public release.

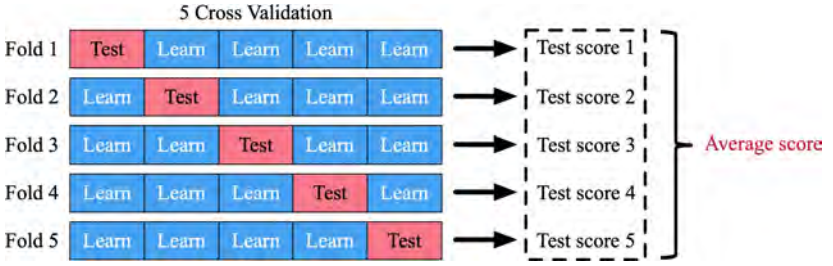


Figure 73: An example of five-fold cross validation. The blue boxes indicate training data used for deriving machine learning model. The red boxes indicate test data used to obtain test scores. An average score of five test scores are used for the optimization procedure and assessment.

can also incorporate the regularization term into a loss function to take a constraint for weights (Bishop 2006). It is widely known that the use of the regularization for weights helps machine learning models attain robustness for overfitting and have weights to be a sparse. Depending on the problem setting and characteristics of the provided data sets, one can choose the appropriate loss function.

13.2.8. Cross validation and overfitting

Presented in figure 73 is the five-fold cross validation used in the present study. In this example, the data set is divided to five subsets. To maintain reproducibility, we should use different boxes for each cross validation process as the test data set colored by red in figure 73. We then obtain an average score from five for assessment. Supervised machine learning model should be able to provide good predictions for unseen data from this training process, thus avoiding overfitting of data. This process is critically important as the quote by Brunton and Kutz (Brunton & Kutz 2019) suggests: "*if you don't cross-validate, you is dumb.*"

Furthermore, we can use various functions to prevent overfitting, i.e., dropout (Srivastava *et al.* 2014), L_1 or L_2 regularizations of loss function (Bishop 2006). However, we do not explore these options in the present study for brevity.

13.3. Examples

For demonstration, we consider three regression problems in fluid dynamics (as summarized in figure 67):

(i) The estimation of drag and lift coefficients from limited sensors in laminar flows;
(ii) The estimation of laminar wake flow from limited sensors; and
(iii) Super-resolution analysis of coarse images for laminar and turbulent flows

which are listed in the order of complexity. To tackle these problems, the following four machine learning models are tested:

(i) Multi-layer perceptron (MLP) (section 2.1.1)
(ii) Random forest (RF) (section 2.1.2)
(iii) Support vector regression (SVR) (section 2.1.3)
(iv) Extreme learning machine (ELM) (section 2.1.4)

for Problems 1 and 2. For Problem 3, we use:

(i) Two- and three-dimensional convolutional neural networks, and
(ii) Two- and three-dimensional DSC/MS model.

DISTRIBUTION A: Distribution approved for public release.

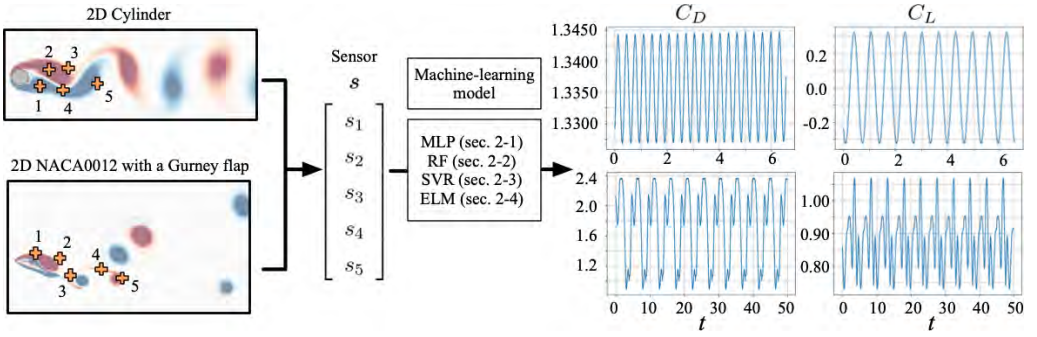


Figure 74: Estimation of C_D and C_L values with laminar flow. Models for estimating C_D and C_L are constructed individually.

We note that normalization and standardization are not applied to the current examples. Readers can also consider the use of these scaling methods to improve the quality of approximations (Shanker *et al.* 1996).

13.3.1. Estimation of drag and lift coefficients for laminar wakes

Various machine learning methods have been used to develop a suitable meta-modeling architecture for various flows. In earlier studies, there were efforts to estimate drag or lift coefficient from flow field data. Zhang *et al.* (Zhang *et al.* 2018) tried to estimate the lift coefficient from flow characteristics and geometry of NACA632615 airfoil. One of the interesting observations of their study is that the test accuracy can be improved by using an artificial image as the input, which has a color corresponding to the input parameters. The estimation method for drag and lift coefficient of various bluff body shapes were proposed by Miyanawala and Jaiman (Miyanawala & Jaiman 2018). They achieved 80% accuracy for test data by converting the regression problem into a classification problem. Here, we focus on the drag and lift coefficient estimations for the wakes behind a two-dimensional cylinder and NACA0012 airfoil with a Gurney flap. In the present setting, we only use five sensor measurements in the wake, as illustrated in figure 74. We examine this simple problem to assess the performance of the four machine learning models.

Let us consider the two-dimensional cylinder wake flow. The flow fields are prepared from a two-dimensional direct numerical simulation (Taira & Colonius 2007*c*; Colonius & Taira 2008*c*). The governing equations are the incompressible Navier–Stokes equations:

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0, \tag{13.7}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u} \cdot \boldsymbol{\nabla} \boldsymbol{u} = -\boldsymbol{\nabla} p + \frac{1}{Re_D} \nabla^2 \boldsymbol{u}, \tag{13.8}$$

where $\boldsymbol{u}$ and p are the non-dimensional velocity and pressure variables. The Reynolds number $Re_D = 100$ is based on the diameter and the freestream velocity. Five nested levels of multi-domains are used with the finest level being $(x, y)/D = [-1, 15] \times [-8, 18]$ and the largest domain being $(x, y)/D = [-5, 75] \times [-40, 40]$ in size. The time step is set to $\Delta t = 2.50 \times 10^{-3}$. In example 3.1, we utilize the data around the cylinder body: the size of domain of interest and the number of grid points set to $[-0.7, 15] \times [-5, 5]$ and $(N_x^*, N_y^*) = (192, 112)$, respectively.

As shown in figure 74, we consider readings from the five sensor measurements $\boldsymbol{s} = \{s_1, s_2, s_3, s_4, s_5\}$ in the wake as the input data to the machine learning models in order

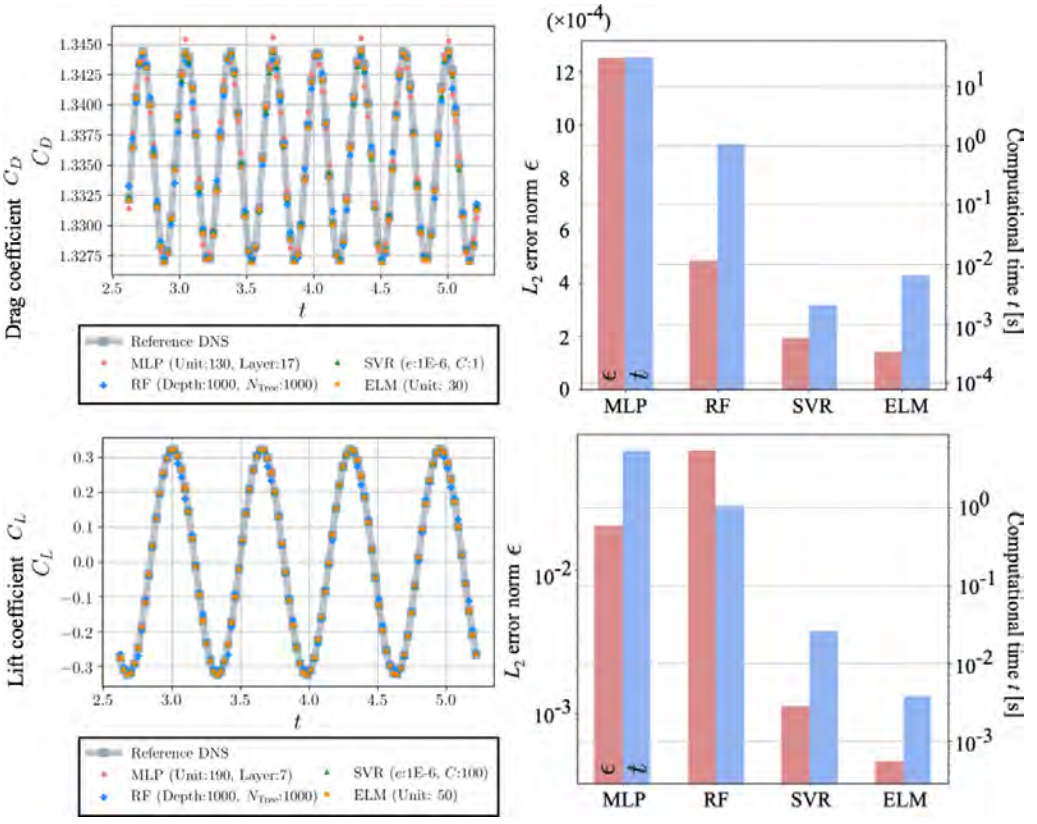


Figure 75: Summary of C_D and C_L estimations for laminar cylinder wake at $Re_D = 100$. In the top and bottom, C_D and C_L estimations, respectively, are presented. The left side shows temporal variations of force coefficients. The optimized parameters are provided in the graph legends. The color bars on the right show the relationship between L_2 error norm ϵ of each coefficient (red) and computational time (blue) for each machine learning model.

to estimate the force coefficients through

$$C_D = \mathcal{F}_D(\boldsymbol{s}) \quad \text{and} \quad C_L = \mathcal{F}_L(\boldsymbol{s}).$$

Here, the vorticity value ω is used as the input attribute. Note here that we make a machine learning model for each coefficient separately to avoid the influence of the magnitude differences on the model. The comparison of the results from the four machine learning models are summarized in figure 75. We use 100 snapshots corresponding to 8 periods in time as the training data for the baseline model. The top and bottom subplots represent the C_D and C_L estimations, respectively. The plots in the left side present the time history of force coefficients. As mentioned above, the parameters of each machine learning model are optimized for fair comparison and shown in the graph legend. For both C_D and C_L estimations with all machine learning models, the results well with the reference data. The assessment of L_2 error norm $\epsilon = ||f_{\text{True}} - f_{\text{Pred}}||_2 / ||f_{\text{True}}||_2$ and the computational time of the learning process on the NVIDIA Tesla K40 graphics processing unit (GPU) for the four machine learning models are shown on the right side of figure 75. The errors are approximately 0.2% for C_D and 1% for C_L for the all machine-learned

models, respectively. Note that these errors and computational costs are assessed with five-fold cross validation. With both force estimations, the ELM has advantage over other machine learning models in terms of the accuracy and computational cost. Note that although the ELM shows advantage in terms of both accuracy and computational cost, this method is not robust against noisy inputs due to its simple structure, as discussed later. Also, the comparison here is not straightforward for the computational costs since the parameters of each model is optimized for accuracy. Overall, all machine learning models work as expected since this problem is simple due to the periodic nature of the cylinder wake.

Next, let us consider another example of complex two-dimensional wake behind a NACA0012 airfoil with a Gurney flap as shown in figure 74. Although the flow field is periodic in time, the problem setting here can be regarded as tougher than the cylinder problem because the wake behind an airfoil with a Gurney flap has various frequency contents as explained later. The training data set is obtained by two-dimensional direct numerical simulation at $Re_c = 1000$, defined based on the chord length c (Meena *et al.* 2018*b*). Various types of wakes can be generated depending on the angle of attack α and the Gurney-flap height h/c. In what follows, we focus on the case of $h/c = 0.1$ and $\alpha = 20^\circ$ which produces a complex unsteady wake comprised of time-periodic shedding of two individual vortex pairs, as shown in figure 74. As a signature of two vortex pairs shedding from the airfoil, we observe two dominant frequencies (Meena *et al.* 2018*b*). For this problem, we use the five nested levels of multi domains. The finest domain range is $(x/c, y/c) = [-1, 1] \times [-1, 1]$ and the largest domain is $(x/c, y/c) = [-16, 16] \times [-16, 16]$. The time step is set to $\Delta t = 10^{-3}$. In example 3.1, we use the data around the airfoil: the size of utilized domain and the number of grid points are $[-0.5, 7] \times [-2.5, 2.5]$ and $(N_x^*, N_y^*) = (352, 240)$, respectively. Analogous to the cylinder example, we use the vorticity field ω as the input attribute to the machine learning models.

The summary of C_D and C_L estimations for flows over a NACA0012 airfoil with a Gurney flap is shown in figure 76. We use 100 snapshots corresponding to 8 periods in time as the training data following the cylinder problem. While this problem setting is somewhat more complicated than the cylinder example, the force estimations from MLP, SVR, and RF are in agreement with the references solution. However, the error from the ELM is notably high for this problem setting, although this model exhibits great performance for the cylinder problem. It suggests that an ELM is not necessarily robust for complex problems. Note that the ELM is able to accurately estimate the forces well when a large number of snapshots are available for training, as explained later with figure 77. When complexity is added to the flow problems, the accuracy of the models is compromised especially for those with lower computational costs. This point should be carefully considered when one chooses a machine learning model. Additional computational cost is likely necessary for robustness, as this example suggests.

The cylinder wake and NACA0012 flows are periodic in time. The accuracy of machine-learned models depends on the number of snapshots per cycle (temporal period) contained in the training data. The relationship between the training data density η and L_2 error norm ϵ of C_D estimation with laminar problems are summarized in figure 77. Here, the training data density η is defined as $\eta \equiv n_{\text{snapshot}}/n_{\text{cycle}}$, where n_{snapshot} is the number of snapshots and n_{cycle} is the number of cycles in time, respectively. The tuning parameters of the machine learning models are the same as those in the above discussions (obtained from $n_{\text{snapshot}} = 100$ and $n_{\text{cycle}} = 8$). In general, higher density leads to enhanced estimation. Noteworthy here is that increased number of cycles for the training data improves model accuracy. Although repeating data sets would not add merits to the training process, slight offset in phase can provide additional merits, which

DISTRIBUTION A: Distribution approved for public release.

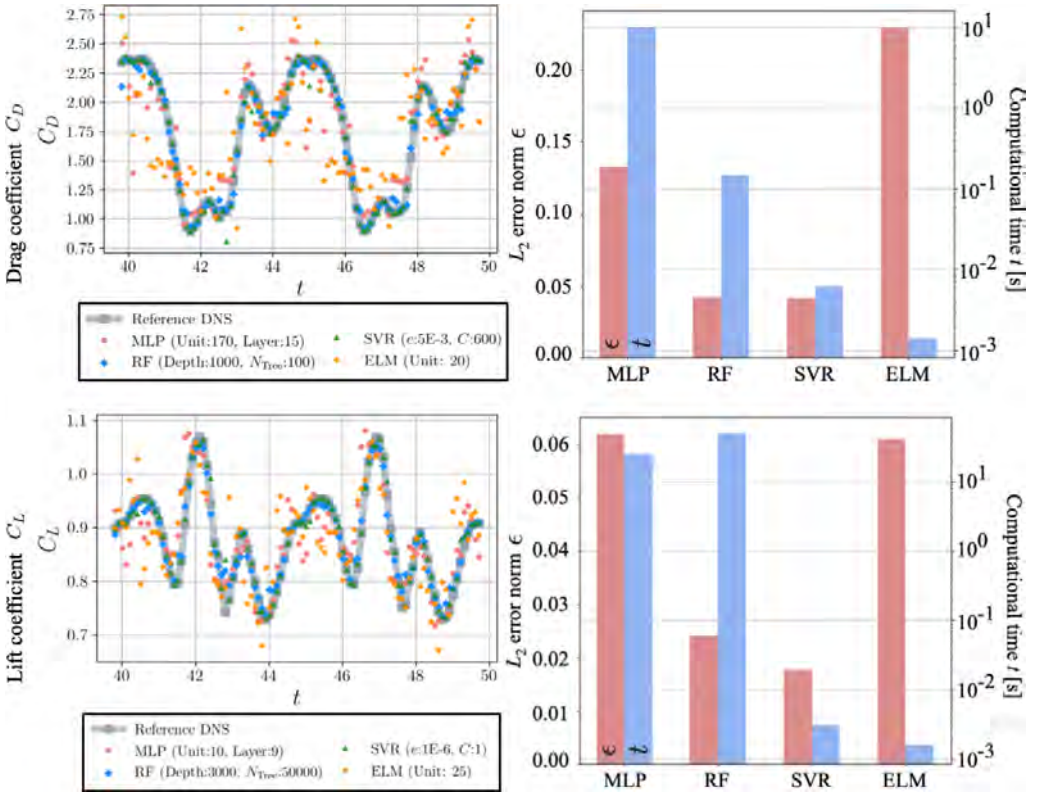


Figure 76: Summary of C_D and C_L estimations for wakes of NACA0012 airfoil with a Gurney flap at $Re_c = 1000$. The top left shows C_D estimations and bottom left presents C_L estimations. The optimized parameters are provided in the graph legends. The color bars located on the right show the relationship between L_2 error norm ϵ of each coefficient (red) and computational time (blue) for each machine learning model.

likely happened here. The other notable observation is that the ELM is not robust for short training data for the airfoil problem. Although the computational time to establish the ELM is much shorter compared to other models, it requires larger amount of data for reconstructing problems with added complexity. We note in passing that similar trends are observed with C_L estimations.

In the above discussions, the five sensor measurements are used as the input for the machine-learning models. Next, let us investigate the influence of the number of sensors n_{sensor} for the force estimations. We examine $n_{\text{sensor}} = \{1, 2, 3, 4, 5\}$. The number of sensors are reduced based on the magnitude of correlation coefficients with other sensors and force coefficients, as explained in detail later. The normalized correlation coefficient is given as

$$R_{ij} = \frac{C_{ij}}{\sqrt{C_{ii} \cdot C_{jj}}}, \tag{13.9}$$

where C_{ij}, C_{ii}, and C_{jj} denote the covariance matrices corresponding to that of a single sensor and its counterpart, the single sensor itself, and its counterparts themselves, respectively. The dependence on n_{sensor} with the covariance matrices for both C_D and C_L estimations are shown in figure 78. Since the correlation magnitude of s_2 is the lowest

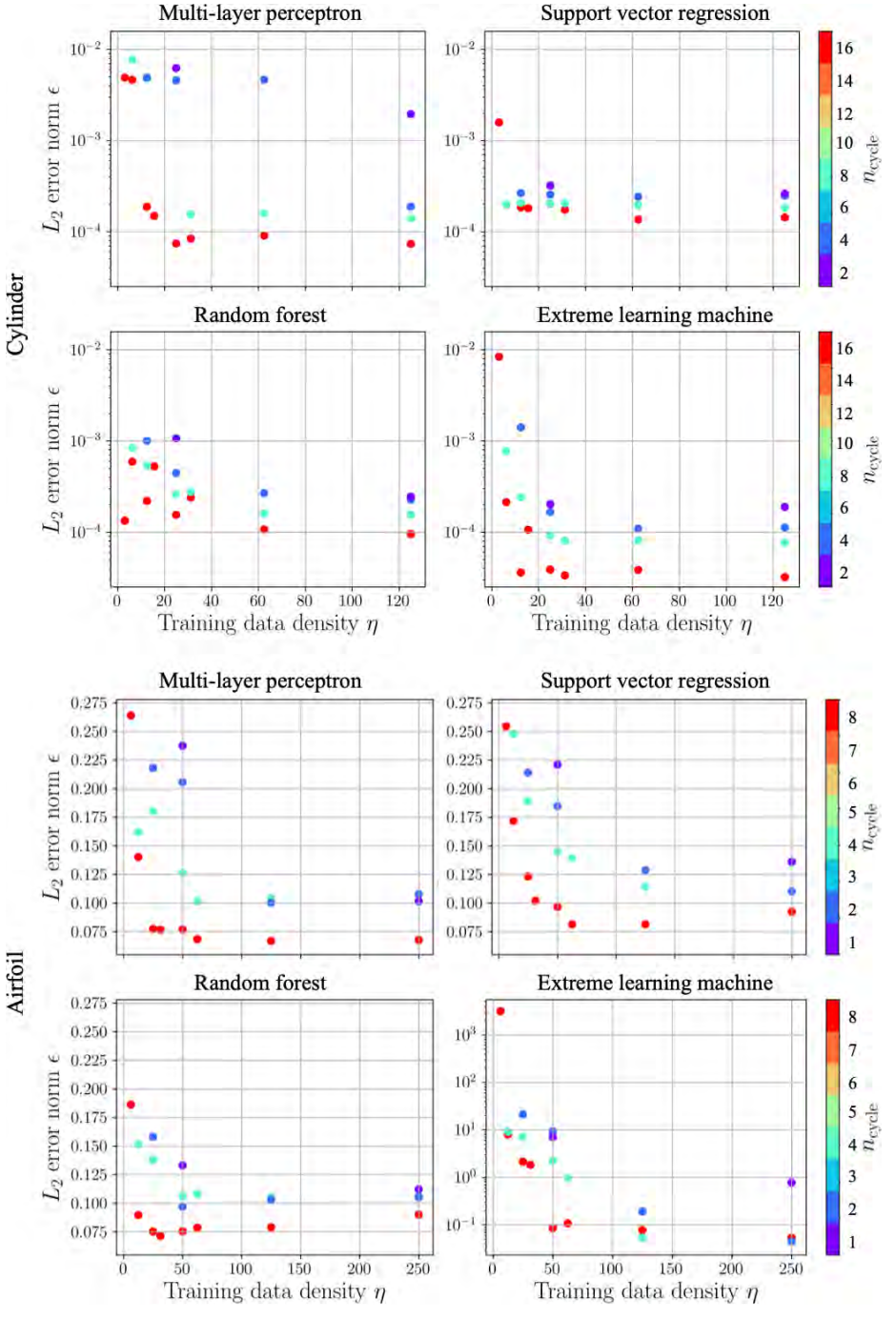


Figure 77: The relationship between the training data density η and L_2 error norm ϵ of C_D prediction for four machine learning models. Note that the range of y-axis with extreme learning machine for the airfoil problem is different from the others.

DISTRIBUTION A: Distribution approved for public release.

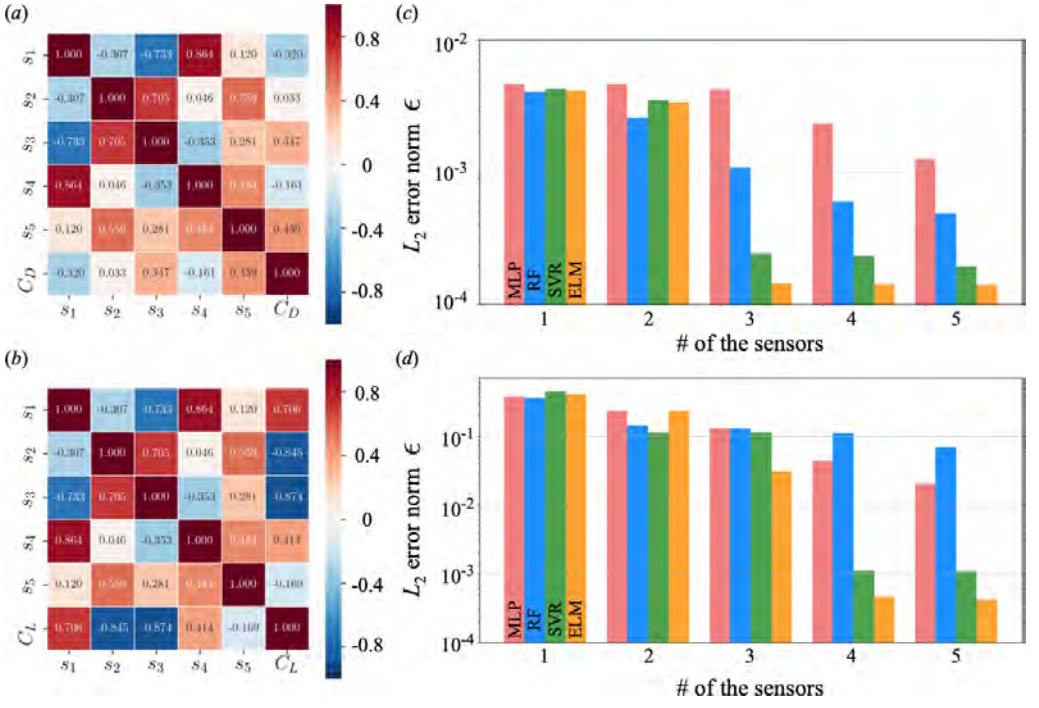


Figure 78: The influence of the number of sensors n_{sensor} on the accuracy of the force estimations. Correlation matrix of (*a*) C_D and (*b*) C_L estimations. The relationship between n_{sensor} and L_2 error norm of (*c*) C_D and (*d*) C_L estimations.

for C_D, we use $\boldsymbol{s}_{\text{reduced}} = \{s_1, s_3, s_4, s_5\}$ with $n_{\text{sensor}} = 4$. In other words, the sensor with the lowest correlation magnitude is discarded with decreasing the number of used sensors as the input. Note that the optimized parameter at $n_{\text{sensor}} = 5$ is used for construction of machine learning models with other number of the sensors. With both force estimations, the errors are dramatically increased when the high-correlation sensors are removed, i.e., $n_{\text{sensor}} = 2$ of C_D estimation (when s_1 is removed) and $n_{\text{sensor}} = 3$ of C_L estimation (when s_4 is removed). Although we apply the correlation analysis as this example, K-means clustering (MacQueen 1967) can be considered alternatively for the initial problem setting.

13.3.2. Estimation of laminar wakes from limited sensor measurements

Next, let us significantly increase the dimension and complexity of the problem. Here, we consider reconstructing the wake from limited measurements, as illustrated in figure 79. This problem settings for machine learning assessments are inspired by Erichson *et al.* (Erichson *et al.* 2020). They reconstructed the flow field from limited measurements using MLP referred to as the *shallow decoder*. We perform the assessments for cylinder flows and the NACA0012 wakes. In both cases, we use 100 snapshots corresponding to 8 periods as the training data. To focus on the wake region, we extract the flow field from the original DNS data $(N_x^*, N_y^*) = (192, 70)$ with $(x^*, y^*)/D = [-0.7, 15] \times [-3.3, 3.3]$ and $(N_x^*, N_y^*) = (352, 192)$ with $(x^*, y^*)/c = [-0.5, 7] \times [-2, 2]$ respectively for the cylinder and NACA0012 flows.

The results of the cylinder wake reconstruction are summarized in the top of figure 80. For the cylinder case, the machine learning models $\mathcal{F}$ are the mapping functions from

DISTRIBUTION A: Distribution approved for public release.

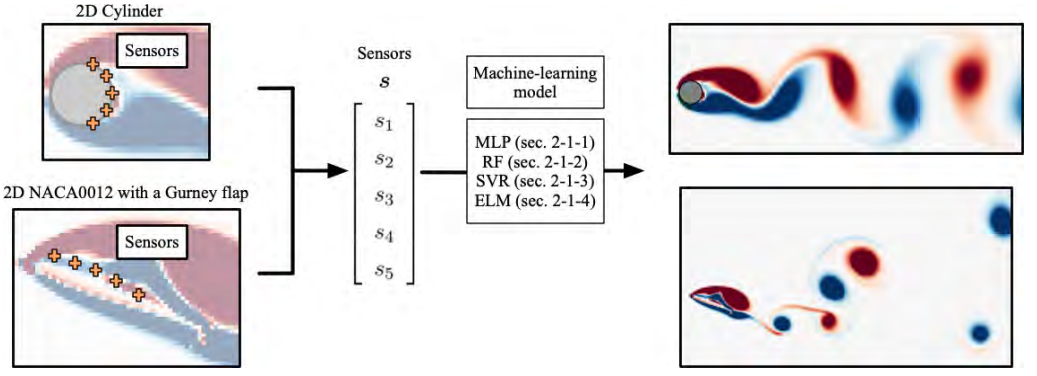


Figure 79: Wake reconstruction using machine learning with five surface-sensor measurements.

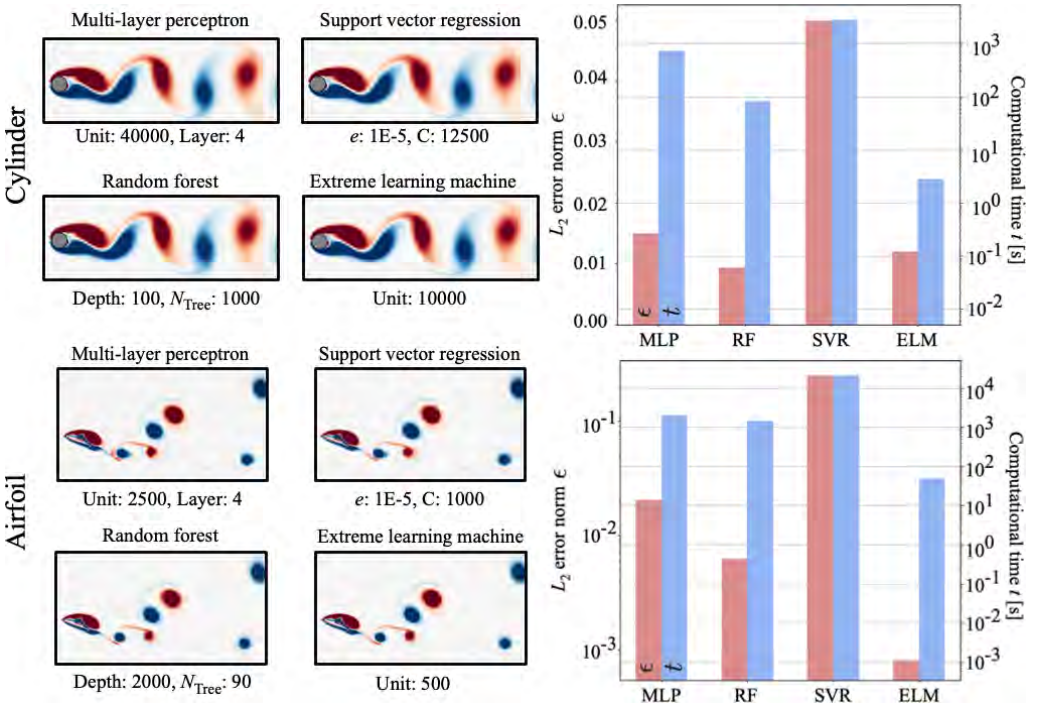


Figure 80: Summary of wake reconstruction for cylinder and NACA0012 with Gurney flap wakes. The left side shows the compilation of vorticity fields reconstructed by the four machine learning models. Optimal parameters are listed below the vorticity fields. The color bars located on the right side show the relationship between L_2 error norm ϵ (red) and computational time (blue) for the selected machine learning models.

five sensor measurements $\boldsymbol{s} \in \mathbb{R}^5$ to the whole wake flow field $\mathcal{F}(\boldsymbol{s}) \in \mathbb{R}^{13440}$ such that $\boldsymbol{\omega} \approx \mathcal{F}(\boldsymbol{s})$. The vorticity field ω is used as the input and output attributes. The flow field can be reconstructed well with all machine learning models as shown by the contour plots in figure 80. The relationship between L_2 error norm ϵ and computational costs are also evaluated. It can be seen that the SVR is not suitable for this high-dimensional output problem because the error levels are higher than those of the other models despite its

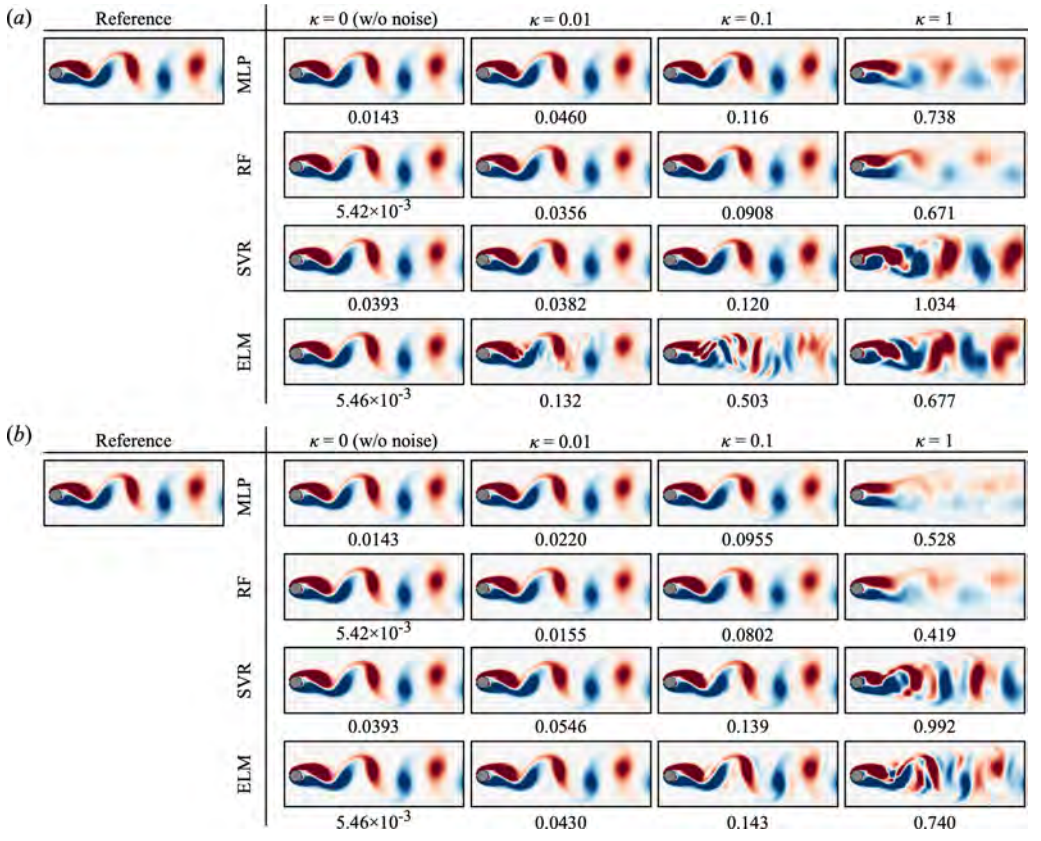


Figure 81: The reconstructed vorticity fields of cylinder wake from five noisy surface-sensor measurements ($\boldsymbol{s}+\kappa\boldsymbol{n}$). The values listed underneath each flow field represent the L_2 error norms. (a) Without noise addition and (b) with noise addition in training data.

higher computational costs.

We also consider the reconstruction of the wake of a NACA0012 airfoil with a Gurney flap, as summarized in the bottom of figure 80. With the airfoil wake, the machine learning models attempt to reconstruct the vorticity field ($\in \mathbb{R}^{67584}$) from five sensor measurements located on the upper surface. The reconstructed flow fields show reasonable agreement with the reference DNS data as shown in figures 79 and 80, although the L_2 error norms are slightly larger than those from the cylinder example except for ELM due to the increased complexity of the wake. In terms of the errors and computational costs, the trends for this problem follow the cylinder wake example.

Next, let us consider the robustness of these machine learning models against a noisy input by using the example of the wake reconstruction from the sensor measurements for each machine learning model. Here, we consider two cases to examine the robustness as follows:

(i) The machine-learned models pre-trained without noise are used as they are, while noise is added to the input data. The L_2 error norm of noisy input case is assessed as $\epsilon_{\text{noisy}} = ||\boldsymbol{\omega}_{\text{True}} - \mathcal{F}(\boldsymbol{s}+\kappa\boldsymbol{n})||_2/||\boldsymbol{\omega}_{\text{True}}||_2$, where $\boldsymbol{n}$ is the Gaussian noise and κ represents the magnitude of the noise.

(ii) Machine learning models are newly trained with training data containing noise such that $\epsilon^*_{\text{noisy}} = ||\boldsymbol{\omega}_{\text{True}} - \mathcal{F}^*(\boldsymbol{s}+\kappa\boldsymbol{n})||_2/||\boldsymbol{\omega}_{\text{True}}||_2$, where $\mathcal{F}^*$ is machine learning

DISTRIBUTION A: Distribution approved for public release.

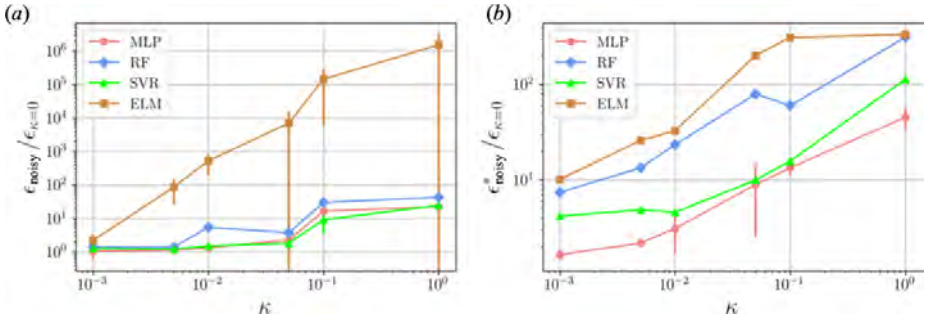


Figure 82: The L_2 error norms of the four machine-learned models over varied noise level κ. The error levels are normalized by the error level without noise added to training data. (a) Without noise addition and (b) with noise addition in training data.

models trained by using the training data with added noise.
The reconstructed vorticity fields of the cylinder wake from five noisy sensor measurements are shown in figure 81. The value shown below each contour is the L_2 error norm in the respective case. The relationship between the intensity of noisy inputs and L_2 error norms are also summarized in figure 82. In the cylinder flow case with MLP, RF, and SVR, the reconstructed flow fields show reasonable agreement with the reference DNS data up to $\kappa = 0.1$. Once κ reaches $\kappa = 1$, these three machine-learned models cannot reconstruct the wake well and L_2 errors become larger than 40%, as shown in figure 81. On the other hand, the vorticity field obtained by the ELM cannot be reconstructed well even $\kappa = 0.01$, although the accuracy is recovered by adding the noise into the training data as shown in figure 82(b). This trends can be observed especially for $\kappa = 0.1$ in figure 81. ELM lacks robustness against for noisy input in both cases due to its simplicity. Of course, this simple network structure has advantage in terms of low computational cost as mentioned above. Note that similar trends are observed for the NACA0012 wake case. Users are cautioned to choose the appropriate machine learning model based on the objective, robustness requirement, and available computational resources.

The influence of the number of sensors n_{sensor} on the accuracy is investigated for the wake reconstruction using the cylinder example. Instead of the force coefficients of R_{ij} in section 3.1, we use a new sensor s_6 as the representation sensor of wake region as shown in figure 83(a). A covariance matrix is then constructed using $\boldsymbol{s} = \{s_1, s_2, s_3, s_4, s_5, s_6\}$. We examine $n_{\text{sensor}} = 1, 2, 3, 4$, and 5. Similar to the discussion of the influence on the number of sensors with the force estimation in section 3.1, the sensor with the low correlation magnitude is discarded with decreasing the number of used sensors as the input. Note that the optimized parameter at $n_{\text{sensor}} = 5$ is used for construction of machine learning models with other number of the sensors. As presented in figure 83(c), the error increases with the reduction of the number of sensors. The reconstructed wakes corresponding to each number of sensors with all machine-learned models are summarized in figure 84. With $n_{\text{sensor}} = 3, 4$, and 5, the wakes are reconstructed well for all machine-learning models, although the errors increase slightly with lower n_{sensor}. The ELM and SVR can maintain the accuracy of reconstruction with $n_{\text{sensor}} = 2$. On the other hand, the reconstructed fields with the MLP and RF do not capture the vortex shedding correctly. For $n_{\text{sensor}} = 1$, all models are unable to reconstruct the wake with the present parameter settings of the machine learning models. It implies that MLP and RF are more sensitive than SVR and ELM for the wake reconstruction problem in terms of parameter tuning. Note here that the wake may be reconstructed with a small number of sensors and

DISTRIBUTION A: Distribution approved for public release.

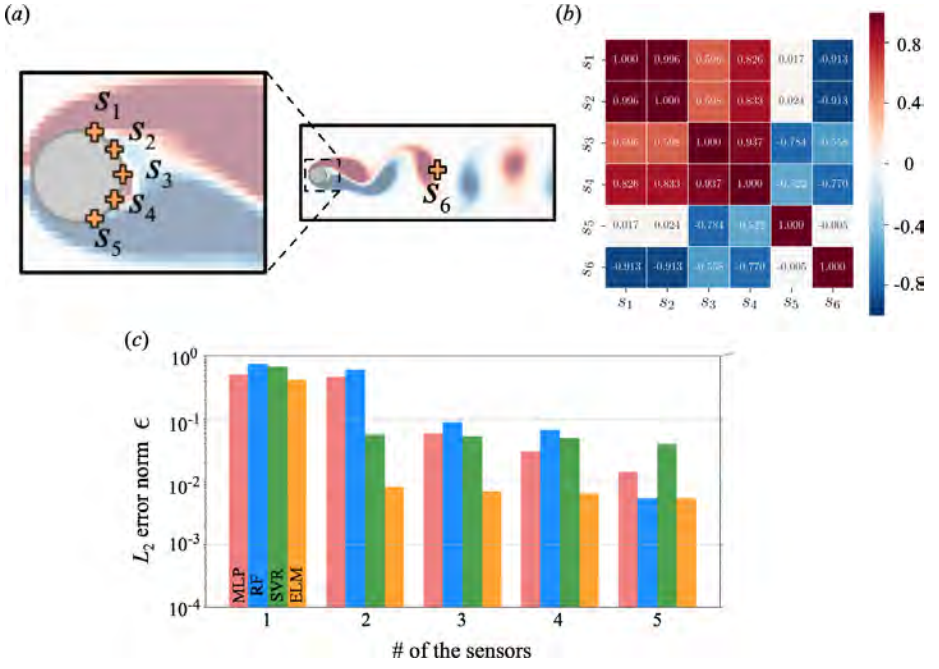


Figure 83: Influence of the number of sensors on the cylinder wake reconstruction. (*a*) Positions of each sensor. (*b*) Covariance matrix. (*c*) Relationship between the number of sensors and L_2 error norm.

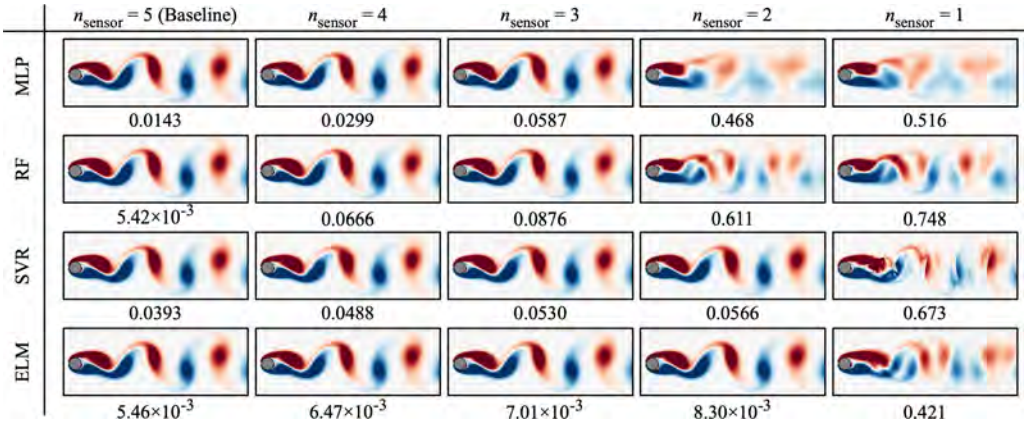


Figure 84: The reconstructed vorticity fields of cylinder wake with $n_{\text{sensor}} = 1, 2, 3, 4$, and 5.

appropriate parameters, although we use the same parameters tuned for $n_{\text{sensor}} = 5$ in the present study.

13.3.3. Super-resolution analysis of flow image data

The curse of dimensionality is known as one of the biggest issues for machine learning in dealing with large data sets, such as images or videos (Domingos 2012). For instance, applications of MLP is strongly restricted by this issue because of the fully-connected structure of MLP. In fact, four machine learning models discussed in section 3.2 cannot be

DISTRIBUTION A: Distribution approved for public release.

easily applied to much higher dimensional problems as of this moment, due to the curse of dimensionality and the computational burden. To overcome this issue, the convolutional neural network (CNN) was developed in computer science (LeCun *et al.* 1998). The key feature of a CNN is called weight sharing which considers the weight as the filter operation as mentioned in section 2.1.5. The basic assumption of CNN is that the pixels of that are far apart in images have no strong relationship. In fact, this assumption can be appropriate in many cases, allowing us to reconsider fully-connected structure. This is a very attractive idea for high-dimensional problems. Recently, there have been increased attention on the use of weight sharing with fluid dynamics.

One of the fields where this benefit is exploited is the example-based super resolution (Bannore 2009; Salvador 2016) in computer science. Through the use of super-resolution reconstruction, a high-resolution signal can be reconstructed from a given low-resolution input. Although the concept here has been explored with various techniques including interpolations and high-frequency transfer, supervised machine learning has recently made tremendous strides with the super-resolution analysis due to its ability to extract key features from training data. Here, let us present the super-resolution analysis as an example for extracting insights from two- and three-dimensional fluid flow data via the CNN.

13.3.4. Two-dimensional flows

We introduce the concept of super-resolution analysis using two-dimensional flows following Fukami *et al.* (Fukami *et al.* 2019*a*). The same data set of cylinder wake from the previous example is considered as a laminar flow example. We further consider the two-dimensional decaying homogeneous isotropic turbulence to examine the validity of the method for chaotic turbulent flows.

The turbulent data sets are obtained from direct numerical simulation over a bi-periodic domain (Taira *et al.* 2016*a*). The governing equation is the two-dimensional vorticity transport equation

$$\frac{\partial \omega}{\partial t} + \boldsymbol{u} \cdot \nabla \omega = \frac{1}{Re} \nabla^2 \omega, \tag{13.10}$$

where $\boldsymbol{u} = (u, v)$ and ω are the velocity and vorticity variables, respectively. The size of computational domain and the number of grid points are $(L_x, L_y) = (1, 1)$ and $(N_x, N_y) = (128, 128)$, respectively. The Reynolds number $Re \equiv u^* l^* / \nu$, where u^* is the characteristic velocity based on the square root of the spatially averaged initial kinetic energy, l^* is the initial integral length, and ν is the kinematic viscosity. The initial Reynolds number $Re(t_0) = 74.6$ and $\Delta t = 1.950 \times 10^{-4}$ are used for the present study. Here, we prepare 10000 snapshots, 70% of which are utilized for training and remaining 30 % are utilized for validation.

For the purpose of this study, we prepare, for the input coarse images, low-resolution data by max or average pooling operations which are common in image processing. These operations with two-dimensional image are derived as

$$q_{ij}^{\mathrm{LR}} = \left[\frac{1}{M^2} \sum_{p,s \in \mathcal{B}_{i,j}} \left(q_{ps}^{\mathrm{HR}} \right)^{\mathcal{B}} \right]^{\frac{1}{\mathcal{B}}}, \tag{13.11}$$

where $\mathcal{B} = 1$ and ∞ indicate average and max pooling, respectively. Max pooling is able to enhance the color effects and brightness. With average pooling, we can extract the average value from an arbitrary area. The original image of $P \times Q$ pixels can be reduced to $(P/R) \times (Q/S)$ pixels. In the two-dimensional flow examples, $R = S = 8$ with max and average pooling are adopted for the cylinder wake and decaying turbulence, respectively.

DISTRIBUTION A: Distribution approved for public release.

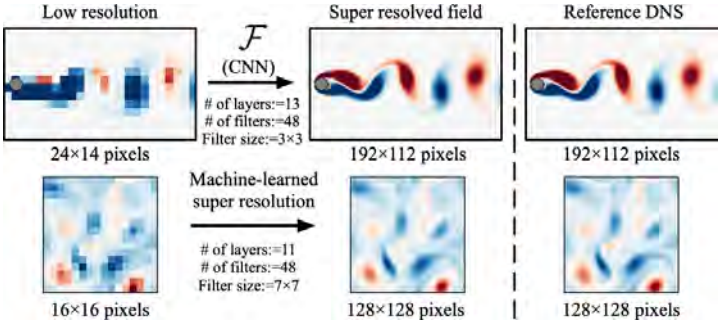


Figure 85: Super-resolution reconstruction with machine learning. The vorticity field ω are used as input and output attributes.

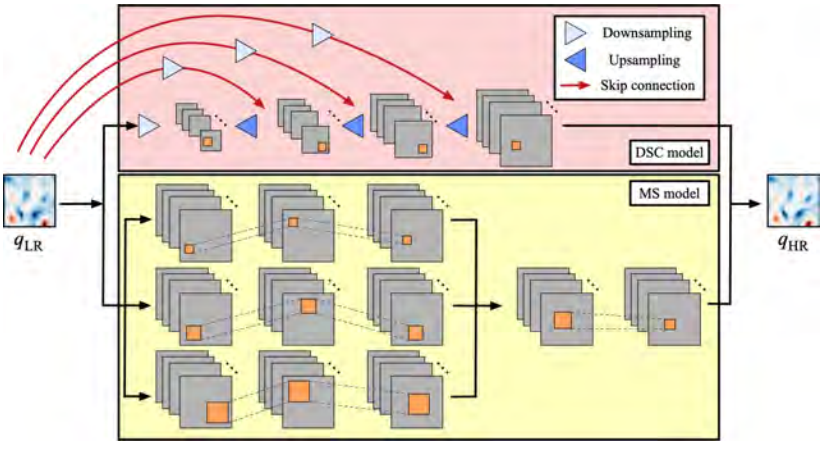


Figure 86: The two-dimensional hybrid downsampled skip-connection/multi-scale (DSC/MS) model.

The flow chart and results of machine-learned super-resolution reconstruction are illustrated in figure 85. For the two-dimensional turbulence case, the test data is prepared with the initial Reynolds number of $Re(t_0) = 87.7$, excluding the training process. The details can be found in (Fukami *et al.* 2019*a*). The aforementioned coarse input data are fed into a machine learning model. In this example, we use a convolutional neural network for $\mathcal{F}$. The optimized parameters are also shown in this schematic. In summary, the machine-learned super-resolution problem can be written as $\boldsymbol{q}_{\mathrm{HR}} = \mathcal{F}(\boldsymbol{q}_{\mathrm{LR}})$, where $\boldsymbol{q}$ is the feature vector. As the $\boldsymbol{q}$ of the two-dimensional setup, we use the vorticity field ω.

As shown in figure 85, the reconstructed flow fields show excellent agreement with the reference DNS data in both laminar and turbulence flow examples. Note that if the input data is further coarse such that $R = S > 8$, we need to modify the machine learning model to incorporate the physical characteristics.

Here, let us present the modified CNN called the hybrid downsampled skip-connection/multi-scale (DSC/MS) model developed by Fukami *et al.* (Fukami *et al.* 2019*a*) in figure 86. As shown, the model is constructed from two parts: namely, the downsampled skip-connection (DSC) model illustrated in red, and the multi-scale (MS) model illustrated in yellow. The DSC model retains a robustness against rotation and translation of flow images by incorporating downsampling and skip connections into its

DISTRIBUTION A: Distribution approved for public release.

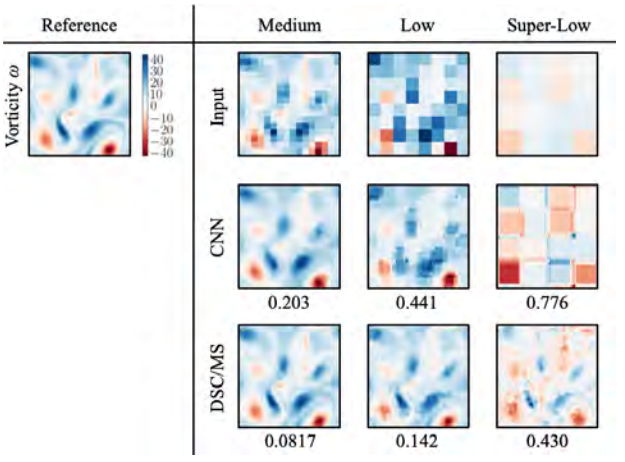


Figure 87: Super-resolution reconstruction with machine learning of two-dimensional turbulence. Reprinted with permission from Cambridge University Press.

architecture. On the other hand, the MS model inspired by Du *et al.* (Du *et al.* 2018) is utilized to take the length scale of flows into account for the flow structures through a number of different filter sizes. We refer readers to Fukami *et al.* (Fukami *et al.* 2019*a*) for details.

As the example of the super-resolution analysis, we consider cases with $R = S > 8$ for the aforementioned machine learning models. We summarize the vorticity fields of two-dimensional decaying turbulence with L_2 error norm listed below the contours in figure 87 (Fukami *et al.* 2019*a*). The flow fields reconstructed by the machine-learned models are in great agreement with the reference DNS data. Note that the conventional CNN does not do a good job. It is striking that the two-dimensional turbulent flow field of 128×128 pixels can be reconstructed from as little as 4×4 pixels by using the proposed model. Moreover, the machine-learned super-resolution analysis is able to increase the maximum wavenumber of the resolved flow field by approximately five fold.

13.3.5. Three-dimensional turbulent channel flow

As the demonstration to handle large three-dimensional data with the CNN, we consider the three-dimensional turbulent channel flow. The data sets are obtained by direct numerical simulation (Fukagata *et al.* 2006), for which the governing equations are the incompressible Navier–Stokes equations,

$$\boldsymbol{\nabla} \cdot \boldsymbol{u} = 0 \tag{13.12}$$

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{\nabla} \cdot (\boldsymbol{u}\boldsymbol{u}) = -\boldsymbol{\nabla} p + \frac{1}{Re_\tau} \nabla^2 \boldsymbol{u}, \tag{13.13}$$

where $\boldsymbol{u} = [u\ v\ w]^{\mathrm{T}}$ represents the velocity with u, v and w being the streamwise (x), wall-normal (y) and spanwise (z) components. Here, p is pressure, t is time, and $Re_\tau = u_\tau \delta / \nu$ is the friction Reynolds number. The quantities are non-dimensionalized using the channel half-width δ and the friction velocity u_τ. The size of the computational domain and the number of grid points are $(L_x, L_y, L_z) = (4\pi\delta, 2\delta, 2\pi\delta)$ and $(N_x, N_y, N_z) = (256, 96, 256)$, respectively. The grid is taken to be uniform in all directions. The no-slip boundary condition is imposed on the walls and the periodic boundary condition is applied in x and z directions. These computations are conduced under a constant pressure gradient at $Re_\tau = 180$.

DISTRIBUTION A: Distribution approved for public release.

As the training data, we take a subdomain of the whole computational domain: the extracted domain size and number of the grid points are ($L_x^* = 2\pi\delta, x \in [\pi\delta, 3\pi\delta]$, $L_y^* = \delta, y \in [0, \delta]$, $L_z^* = \pi\delta, z \in [0.5\pi\delta, 1.5\pi\delta]$) and $(N_x^*, N_y^*, N_z^*) = (128, 48, 128)$. Since turbulence statistics should be symmetric in y direction and homogeneous in the x and z directions, this subdomain retains the turbulent characteristics of the channel flow. Similar to the two-dimensional examples, we adopt the average pooling operation to obtain the coarse input data. The original data of $128 \times 48 \times 128$ grids are reduced to $16 \times 6 \times 16$ grids (medium resolution) and $8 \times 3 \times 8$ grids (low resolution). For training the machine learning model, 100 snapshots are employed. The time step between snapshots is $\Delta t^+ = 13.5$ in viscous time unit. We extend the hybrid DSC/MS model for the three-dimensional super-resolution reconstruction. Wider range of spatial scales are present in this three-dimensional high-Reynolds number turbulent channel flow, making the present problem more challenging than the above two-dimensional flows. The basic parameters are same with the two-dimensional models developed by Fukami *et al.* (Fukami *et al.* 2019*a*).

Let us present the Q-criteria isosurface contours $Q^+ = 0.07$ of the reference DNS data and reconstructed field in figure 88(a). The input coarse data contains barely any information on the vortex cores in streamwise direction. Almost all features are lost by the pooling operation even with medium resolution input. Note that the medium and low coarse input data also contain the information of vortex core. We can find the vortex-like structures with lower threshold, e.g., $Q^+ = 0.005$, as shown in the upper left of figure 88(a). In spite of this grossly coarse input data, the flow field can be reconstructed by the machine learning models, even for the low-resolution input. The velocity contours of $y - z$ section at $x^+ = 1127$ are shown in figure 88(b). The values listed below the contour plots are the L_2 error norms $\epsilon'_f = ||f'_{\mathrm{DNS}} - f'_{\mathrm{ML}}||_2 / ||f'_{\mathrm{DNS}}||_2$, where f' is the fluctuation component of f. The reconstructed flow fields show excellent agreement with the reference DNS data for all velocity components. In terms of the L_2 error norm, the accuracy of streamwise velocity is higher than other two components because of the strength of the feature in channel flow.

In summary, the high-resolution three-dimensional turbulent flow fields can be reconstructed from coarse flow field data through the supervised machine learning technique. The CNN is known to handle an input-output analysis with high-dimensional data set due to its use of filter operations. These results show that machine learning algorithms hold great potentials to serve as powerful analysis tools if used correctly.

13.4. Concluding remarks

We assessed the performance of a number of supervised machine learning models for a range of representative regression problems for fluid flows. Four machine learning algorithms of multi-layer perceptron, random forest, support vector regression, and extreme learning machine were considered for laminar flow problems. As shown in sections 3.1 and 3.2, users can select a network from a zoo of machine-learning models (Brunton & Kutz 2019), taking into account the computational cost, accuracy, robustness, and training data density.

Furthermore, we considered problems that require the handling of two- or three-dimensional fluid flow data. In particular, we considered the super-resolution analysis with convolutional neural network. We found that convolutional neural network is a powerful tool to handle big data, which capitalizes on the concept of filter operation to beat the curse of dimensionality. An adjustable filter for non-uniform grids including body-fitted grids used in high-Re airfoil flow may be desirable for future applications in fluid flows.

DISTRIBUTION A: Distribution approved for public release.

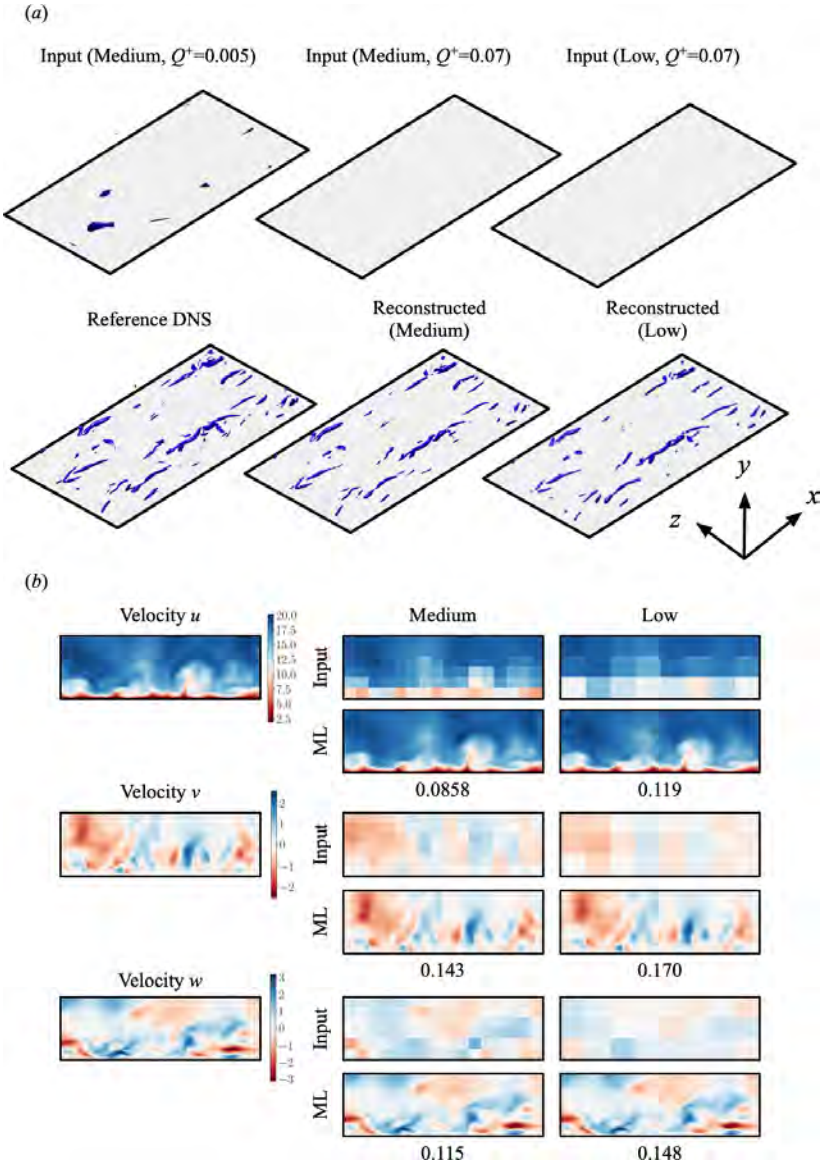


Figure 88: (a) The Q-criterion ($Q^+ = 0.07$) contour plots of (a) input coarse data, (b) reference DNS data, and machine-learned super-resolved data from (c) medium- and (d) low-resolution inputs. (b) The y-z section contours of velocity u, v, and w are for $x^+ = 1127$.

The field of supervised machine learning for fluid dynamics can be combined with other *data-oriented* architecture including linear-based methods (Taira *et al.* 2017*c*, 2020*a*), network science (Meena *et al.* 2018*a*), and artificial intelligence. Prior to employing machine learning, users should care what type of fluid flow problems can really benefit from supervised machine learning with fluid dynamics. Although neural networks are known as an "universal approximator"(Kreinovich 1991; Hornik 1991; Cybenko 1989; Baral *et al.* 2018), the supervised machine learning models are not suitable for extrapolation problem

DISTRIBUTION A: Distribution approved for public release.

in terms of training data (Taira 2019). While it is presently difficult to define the border of interpolation and extrapolation for fluid flow applications, future efforts may enable us to establish machine-learning models for fluid flows that can warn us if a model attempts to step outside of the demarcation of training data boundary.

DISTRIBUTION A: Distribution approved for public release.

14. Concluding Remarks

14.1. Conclusions

We have developed a network-theoretic and data-based framework for analyzing and controlling unsteady fluid flows. Using this framework, we are able to capture and modify the underlying structure of vortical and modal interactions present in unsteady fluid flows. We are also able to deduce feedback control laws directly using data trajectories from complex flows. The presented formulation opens up a promising avenue of interaction-based and model-free studies of a wide variety of fluid flows complementing classical analysis techniques.

For capturing vortical interactions, vortical elements in the form of discrete point vortices were considered as nodes for graph representations. The edge weights of the graph were related to the importance of the ties between the vortices, which were quantified based on the strength of the vortices and the relative distance between them. We sparsify the connections between these vortices based on spectral graph theory not only to cut some of these ties but also to redistribute their weights to preserve the dynamics of the vortices. We have demonstrated that an effective resistance approach to graph sparsification proves extremely useful in constructing sparsified-dynamics models. Moreover, sparsification reduces the number of connections to $\mathcal{O}(N \log N/\epsilon^2)$ based on approximation order ϵ. We also observe an increase in the effectiveness of sparsification with increasing number of point vortices.

In spite of the nonlinear nature of the dynamics of these point vortices, the sparse representations are capable of identifying the core vortex structures as well as tracking the bulk motion of these structures in an effective manner. We have referred to the dynamical model that uses the sparsified vortex network as the sparsified-dynamics model. In addition, the sparsified models preserve spectral properties of the original setup. Resparsification is utilized to enhance the prediction of the motion of the vortices over time. We observe that sparsification conserves the invariants of discrete vortices dynamics such as the Hamiltonian, linear and angular impulse, and circulation. The objectives of sparsified-dynamics model are aligned with those of reduced-order models but these goals are achieved by sparsifying the connections rather than deriving a model equation with state variable of reduced dimension. The results of graph-theoretic approach to discrete vortex dynamics point to promising research in its applicability to a variety of problems in fluid mechanics.

We have identified that the vortical interactions in two-dimensional isotropic turbulence have a scale-free network structure (Taira *et al.* 2016*b*). We have been able to reveal the structure by taking a continuous representation of the flow field and quantifying the vortical network using a Cartesian discretization. For two-dimensional isotropic turbulence, the node strength distribution was uncovered to be $P(s) \sim s^{-\gamma}$, where $\gamma = 2.7 \pm 0.5$. Furthermore, we have found that the unforced turbulent flow field possesses an underlying scale-free network structure until the circulation of vortices with characteristic velocity and length scales reach $(u^* l^*)_{\mathrm{cr}} \approx 0.063$.

By noticing that the turbulence network has scale-free characteristics, we were able to systematically show that the turbulence network is resilient against random perturbations but vulnerable against coordinated forcing on the hub vortices. It should be noted that estimating and controlling each and every vortical structure in a turbulent flow is most likely improbable and impractical. Instead, network analysis may provide a refreshing view point on how one can predict and modify the collective dynamics of vortices in the turbulent flow fields. We believe that the network-based analysis and control (Mesbahi & Egerstedt 2010; Liu *et al.* 2011; Cornelius *et al.* 2013; Kaiser *et al.* 2014*b*; Yan *et al.*

DISTRIBUTION A: Distribution approved for public release.

2015) will provide novel mathematical fabric for paving the path towards network-based modeling and control of turbulent flows, which can potentially impact a wide spectrum of problems.

Complementing our efforts in characterization of vortical interactions, we also developed a networked oscillator framework for describing modal interactions in unsteady fluid flows. The nodes of the network were considered as modal oscillators which constituted of POD mode pairs describing individual coherent structures in time-periodic flows. The interactions between the oscillators formed edges of the network which were characterized using a perturbation response based approach. Amplitude and phase perturbations introduced in the Navier–Stokes equations were tracked using regression techniques in the networked oscillator framework.

Using a canonical example of flow over a cylinder, the energy transfer dynamics were highlighted and excellent agreement of the model with DNS were observed for both amplitude and phase perturbation cases. An aggregate network model capturing global oscillator interactions in the flow was obtained using combined regression techniques over a multitude of such perturbation cases. The aggregate network model was attributed with the least error in terms of predicting the temporal dynamics of the modal oscillators.

With the knowledge of network interactions between oscillators, an optimal control strategy was designed for suppressing oscillator fluctuations to the natural limit cycle faster along with suppressing the limit cycle itself. Implementing control on the limit cycle as well as perturbations resulted in not only suppressing the modal oscillations but also resulted in a mean shift towards the unstable steady state exhibited by the shift mode. The mean shift corresponds to a reduction in drag and lift force on the cylinder reducing the unsteady forces therein. The interaction-based approach leverages the knowledge of modal interactions providing insights beyond traditional approaches for flow control of unsteady fluid flows.

We also proposed an alternative model-free feedback control strategy leveraging data-based clustering and optimization. The main goal of this approach is a model-free identification and optimization of a global feedback control law, particularly for drag reduction and controlling flow separation. The approach involved an unsupervised clustering of a feature space spanned by the aerodynamic forces, $s^b = (C_L, \dot{C}_L, C_D)$, and identified from data collected from the baseline (unforced) large eddy simulations (LES). These clusters partitioned the control laws, defined in the feature space, into few discrete states. In particular, a centroid-based clustering analysis, performed using the *k-means* algorithm, is employed, which segregates the characteristic phase regimes of the flow. A statistical analysis of the transitions between these clusters yields linear Markov models, which characterize the coarse-grained, probabilistic dynamics of post-stalled flows. The different phases of vortex shedding can be observed from analyzing characteristic flow fields associated with each cluster. Using a directed modularity maximization algorithm, groups of clusters (subsets) are extracted. These subsets divide the baseline trajectory into low, intermediate and high drag states. For the Markov chain graphs associated with the 2D and 3D baseline feature space clustering, switching clusters are associated with key transitions from low to high drag states. The flow physics associated with the switching clusters is characterized by a growth of the rolled-up vortical structures, which do not detach the surface of the airfoil along with an elongation of the low pressure core on the suction side. A modification of fundamental cluster transitions in a model-free manner was sought, which can be interpreted as a re-routing of trajectories associated with the baseline configuration to maximize desired performance objectives.

The feedback control law to be optimized is represented by a smoothened version of a piece-wise constant law, where each cluster is associated with a constant blowing or

DISTRIBUTION A: Distribution approved for public release.

suction jet velocity. The smoothing is deduced from feedback of the current position of the trajectory in the low-dimensional feature space relative to the cluster centroids. The control parameters in each cluster are then iteratively optimized to suppress flow separation. At each iteration step, the control law with the updated actuation parameters is evaluated in the simulation and the associated value of the cost functional penalizing drag power and actuation power is determined. The optimization procedure yields a set of a control laws, which progressively minimize the cost functional. For 3D flow control, additional constraints were added to the optimization procedure to deduce unsteady feedback control laws for pure blowing.

In 2D flow control effort, a 41% drag reduction is obtained with the optimized feedback control law along with complete flow reattachment. The optimized cluster-based control law involves suction at the low drag clusters and blowing at the intermediate and high drag clusters. Taking advantage of the fundamental wake instabilities, the optimized control law in 3D flow control yields a 13% drag reduction. Drag reduction was accompanied by a decrease in the turbulent kinetic energy (TKE) fluctuation in the flow. This results in the stabilization of the wake for 3D controlled flows. On increasing the feedback gain associated with the optimized cluster amplitudes, shear layer instabilities are trigerred in addition to the wake instabilities. Although the actuation power increases with this feedback gain, a faster break-up of the spanwise vortical structures is obtained yielding 20% drag reduction. For both 2D and 3D optimized control cases, the cluster residence probabilities, i.e. the probability of the flow to remain in a particular cluster, of the low drag states are increased with control and the probabilities associated with the high drag states are decreased. As a post-analysis, the Markov transition networks of the baseline and controlled configurations are analyzed and compared. The baseline Markov transition network is optimally modified to allow transitions that result in separation control and drag reduction.

In summary, a feedback control design using data-driven clustering provided a general, model-free formulation for flow control. While we have demonstrated its applicability and power to extract optimized feedback control laws for separated flow, the formulation is independent of the specific configuration and applicable to a variety of fluid flows. Of crucial importance is the formulation of the cost functional, as in any optimal control strategy. The associated control law is deduced and optimized in a model-free and automated manner based on consecutive evaluations of the simulation. The cluster-based approach combines the modern day computing capabilities with data-driven techniques and can be transformative in driving future flow control technology.

14.2. Future work

The analysis of fluid flows described in this final report have a vast range of possible applicabilities which have not been fully explored. Some avenues of application of network and data-based analysis in fluid flows can be explored as below.

- *Extend vortical and modal network analysis to more complex flows.*

Vortical network analysis has been applied to discrete point vortices and two-dimensional decaying isotropic turbulence. Modal network analysis has been applied to two-dimensional incompressible flow over a circular cylinder. The current vortical and modal network analysis will be extended to complex three-dimensional flows. For the modal networks, the effect of spanwise perturbations in the fluid flow will be examined.

- *Develop rigorous control framework for vortical and modal network models.*

Multi-agent based feedback control strategies will be integrated with the vortical and modal network models providing a framework for leader-follower based control of complex fluid flows. Efforts will be directed towards combining machine learning strategies with

DISTRIBUTION A: Distribution approved for public release.

the network models. Currently, networked oscillator model provides linear approximation to interactions between modes in unsteady fluid flow by introducing perturbations to the flow. Incorporating machine learning control with these modal network modes, transient perturbation dynamics of the modes will be studied.

- *Control of unsteady vortical flows.*

The theoretical and numerical foundations established in this work pave the way to enable control of unsteady flows that possess large coherent structures. This can propel active flow control techniques to modify the dynamics of unsteady vortical flows caused by gusts and unsteady body motion (resulting in unsteady base states). Such endeavor will require computationally tractable methods and robust controller designs. Our upcoming efforts will focus on taming highly unsteady flows.

DISTRIBUTION A: Distribution approved for public release.

15. Acknowledgements

We thank the US Air Force Office of Scientific Research (Grant number: FA9550-16-1-0650, Program Managers: Gregg Abate and Douglas Smith) for supporting this work. We also thank the International Student Exchange Program (Program Manager: Gary Strack) that enabled valuable international student visits to broaden the horizon of this research effort. Some of the computations were supported by HPCMP at the Department of Defense. We also acknowledge fruitful collaborations and enlightening discussions with Benjamin Erichson, Koji Fukagata, Kai Fukami, Nathan Kutz, Romit Maulik, Muralikrishnan Gopalakrishnan Meena, Bernd Noack, Nesar Ramachandra, and Peter Schmid.

DISTRIBUTION A: Distribution approved for public release.

References

Ahuja, S. & Rowley, C. W. 2010 Feedback control of unstable steady states of flow past a flat plate using reduced-order estimators. *J. Fluid Mech.* **645**, 447–478.

Albert, A. 1972 Regression and the moore-penrose pseudoinverse .

Albert, R. & Barabási, A.-L. 2002 Statistical mechanics of complex networks. *Rev. Mod. Phys.* **74** (1), 47–97.

Albert, R., Jeong, H. & Barabási, A.-L. 2000 Error and attack tolerance of complex networks. *Nature* **406**, 378–382.

Amitay, M. & Glezer, A. 2002*a* Controlled transients of flow reattachment over stalled airfoils. *International Journal of Heat and Fluid Flow* **23** (5), 690–699.

Amitay, Michael & Glezer, Ari 2002*b* Role of actuation frequency in controlled flow reattachment over a stalled airfoil. *AIAA journal* **40** (2), 209–216.

Amitay, Michael & Glezer, Ari 2006*a* Aerodynamic flow control using synthetic jet actuators. In *Control of Fluid Flow*, pp. 45–73. Springer.

Amitay, M. & Glezer, A. 2006*b* Flow transients induced on a 2D airfoil by pulse-modulated actuation. *Experiments in Fluids* **40** (2), 329–331.

Anderson, John David 1999 *Aircraft performance and design*. McGraw-Hill.

Arbabi, Hassan & Mezić, Igor 2017 Ergodic theory, dynamic mode decomposition and computation of spectral properties of the koopman operator. *SIAM J. Appl. Dyn. Syst.* **16** (4), 2096–2126.

Aubry, N., Holmes, P., Lumley, J. L. & Stone, E. 1988 The dynamics of coherent structures in the wall region of a turbulent boundary layer. *J. Fluid Mech.* **192**, 115–173.

Babaee, H & Sapsis, TP 2016 A minimization principle for the description of modes associated with finite-time instabilities. *Proceedings of the Royal Society A* **472** (2186), 20150779.

Bagheri, Shervin 2013 Koopman-mode decomposition of the cylinder wake. *Journal of Fluid Mechanics* **726**, 596–623.

Bagheri, S., Hœpffner, J., Schmid, P. & Henningson, D. 2009 Input-output analysis and control design applied to a linear model of spatially developing flows. *Applied Mechanics Reviews* **62** (2).

Bai, Zhe, Erichson, N. Benjamin, Gopalakrishnan Meena, Muralikrishnan, Taira, Kunihiko & Brunton, Steven L. 2019 Randomized methods to characterize large-scale vortical flow networks. *PLOS One* **14** (11).

Bannore, V. 2009 *Iterative-interpolation super-resolution image reconstruction: a computationally efficient technique*, , vol. 195. Springer Science & Business Media.

Barabási, A.-L. & Albert, R. 1999 Emergence of scaling in random networks. *Science* **286**, 509–512.

Baral, C., Fuentes, O. & Kreinovich, V. 2018 Why deep neural networks: A possible theoretical explanation, In Constraint Programming and Decision Making: Theory and Applications. *Springer* pp. 1–5.

Barkley, D. & Henderson, R. D. 1996 Three-dimensional Floquet stability analysis of the wake of a circular cylinder. *Journal of Fluid Mechanics* **322**, 215–241.

Barrat, A., Barthélemy, M., Pastor-Satorras, R. & Vespignani, A. 2004*a* The architecture of complex weighted networks. *PNAS* **101** (11), 3747–3752.

Barrat, A., Barthélemy, M. & Vespignani, A. 2004*b* Weighted evolving networks: coupling topology and weighted dynamics. *Phys. Rev. Let.* **92** (22), 228701.

Basley, J., Pastur, L. R., Delprat, N. & Lusseyran, F. 2013 Space-time aspects of a three-dimensional multi-modulated open cavity flow. *Physics of Fluids (1994-present)* **25** (6), 064105.

Batchelor, G. 2000 *An introduction to fluid dynamics*. Cambridge university press.

Benczúr, A. & Karger, D. R. 1996 Approximating $s-t$ minimum cuts in $\mathcal{O}(n^2)$ time. In *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing*, pp. 47–55. ACM.

Benton, Stuart I & Visbal, Miguel R 2018 High-frequency forcing to mitigate unsteady separation from a bursting separation bubble. *Physical Review Fluids* **3** (1), 013907.

Benzi, R. & Colella, M. 1992 A simple point vortex model for two-dimensional decaying turbulence. *Phys. Fluids A* **4** (5), 1036–1039.

DISTRIBUTION A: Distribution approved for public release.

Benzi, R., Paladin, G. & Vulpiani, A. 1990 Power spectra in two-dimensional turbulence. *Phys. Rev. A* **42** (6), 3654–3656.

Berger, E., Sastuba, M., Vogt, D., Jung, B. & Amor, H. B. 2015 Estimation of perturbations in robotic behavior using dynamic mode decomposition. *Journal of Advanced Robotics* **29** (5), 331–343.

Bergstra, J., Yamins, D. & Cox, D. 2013 Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. *International Conference on Machine Learning* pp. 115–123.

Berkooz, G., Holmes, P. & Lumley, J. 1993*a* The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics* **25** (1), 539–575.

Berkooz, G., Holmes, P. & Lumley, J. L. 1993*b* The proper orthogonal decomposition in the analysis of turbulent flows. *Annual Review of Fluid Mechanics* **23**, 539–575.

Bewley, T. R. 2001 Flow control: New challenges for a new renaissance. *Progress in Aerospace sciences* **37** (1), 21–58.

Bhattacharjee, D., Hemati, M., Klose, B. & Jacobs, G. 2018 Optimal actuator selection for airfoil separation control. In *2018 Flow Control Conference*, p. 3692.

Bishop, C. M. 2006 *Pattern recognition and machine learning*. springer.

Bistrian, D.A. & Navon, I.M. 2016 Randomized dynamic mode decomposition for non-intrusive reduced order modelling. *International Journal for Numerical Methods in Engineering* pp. 1–22.

Bistrian, DA & Navon, IM 2018 Efficiency of randomised dynamic mode decomposition for reduced order modelling. *International Journal of Computational Fluid Dynamics* **32** (2-3), 88–103.

Blanchard, A. & Sapsis, T. P. 2019 Stabilization of unsteady flows by reduced-order control with optimally time-dependent modes. *Physical Review Fluids* **4** (5), 053902.

Boffetta, G. & Ecke, R. E. 2012 Two-dimensional turbulence. *Annu. Rev. Fluid Mech.* **44**, 427–451.

Bollobás, B. 1998 *Modern graph theory*. Springer.

Bollt, Erik M & Santitissadeekorn, Naratip 2013 *Applied and Computational Measurable Dynamics*. SIAM.

Breiman, L. 2001 Random forests. *Mach. Learn.* **45** (1), 5–32.

Brenner, M. P., Eldredge, J. D. & Freund, J. B. 2019 Perspective on machine learning for advancing fluid mechanics. *Phys. Rev. Fluids* **4**, 100501.

Brès, Guillaume A, Ham, Frank E, Nichols, Joseph W & Lele, Sanjiva K 2017 Unstructured large-eddy simulations of supersonic jets. *AIAA Journal* pp. 1164–1184.

Brochu, E., Cora, V. & de N. Freitas 2009 A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *Technical Report TR-2009-023, University of British Columbia* .

Brown, E., Moehlis, J. & Holmes, P. 2004 On the phase reduction and response dynamics of neural oscillator populations. *Neural computation* **16** (4), 673–715.

Brunton, B. W., Johnson, L. A., Ojemann, J. G. & Kutz, J. N. 2016*a* Extracting spatial–temporal coherent patterns in large-scale neural recordings using dynamic mode decomposition. *Journal of Neuroscience Methods* **258**, 1–15.

Brunton, S. L. & Kutz, J. N. 2018 *Data-Driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*. Cambridge University Press.

Brunton, S. L. & Kutz, J. N. 2019 *Data-driven science and engineering: Machine learning, dynamical systems, and control*. Cambridge University Press.

Brunton, S. L. & Noack, B. R. 2015*a* Closed-loop turbulence control: Progress and challenges. *Applied Mechanics Reviews* **67**, 050801–1–050801–48.

Brunton, S. L. & Noack, B. R. 2015*b* Closed-loop turbulence control: progress and challenges. *App. Mech. Rev.* **67** (5), 050801.

Brunton, S. L., Noack, B. R. & Koumoutsakos, P. 2020*a* Machine learning for fluid mechanics. *Annual Review of Fluid Mechanics* **52**, 477–508.

Brunton, S. L., Noack, B. R. & Koumoutsakos, P. 2020*b* Machine learning for fluid mechanics. *Annu. Rev. Fluid Mech.* **52**, 477–508.

Brunton, S. L., Proctor, J. L & Kutz, J. N. 2016*b* Discovering governing equations from

DISTRIBUTION A: Distribution approved for public release.

data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences* **113** (15), 3932–3937.

Brunton, S. L., Proctor, J. L., Tu, J. H. & Kutz, J. N. 2015 Compressed sensing and dynamic mode decomposition. *Journal of Computational Dynamics* **2** (2), 165–191.

Caldarelli, G. 2007 *Scale-free networks*. Oxford Univ. Press.

Canuto, C., Hussaini, M. Y., Quarteroni, A. & Zang, T. A. 1988 *Spectral methods in fluid dynamics*. New York: Springer-Verlag.

Carlberg, K. T., Jameson, A., Kochenderfer, M. J., Morton, J., Peng, L. & Witherden, F. D. 2019 Recovering missing CFD data for high-order discretizations using deep neural networks and dynamics learning. *J. Comput. Phys.* **395**, 105–124.

Cauchemeza, S., Bhattarai, A., Marchbanks, T. L., Fagan, R. P., S. Ostroff, N. M. Ferguson, Swerdlow, D. & working group, Pennsylvania H1N1 2011 Role of social networks in shaping disease transmission during a community outbreak of 2009 H1N1 pandemic influenza. *Proceedings of the National Academy of Sciences* **108** (7), 2825–2830.

Chen, K. K., Tu, J. H. & Rowley, C. W. 2012 Variants of dynamic mode decomposition: boundary condition, Koopman, and Fourier analyses. *Journal of Nonlinear Science* **22** (6), 887–915.

Chen, W. K. 2004 *The Electrical Engineering Handbook*. Elsevier Science.

Chiang, Mark Ming-Tso & Mirkin, Boris 2010 Intelligent choice of the number of clusters in k-means clustering: an experimental study with different cluster spreads. *Journal of classification* **27** (1), 3–40.

Chirila, Dragos B. 2018 Towards lattice boltzmann models for climate sciences: The gelb programming language with applications. PhD thesis, University of Bremen.

Choi, Haecheon, Jeon, Woo-Pyung & Kim, Jinsung 2008 Control of flow over a bluff body. *Annu. Rev. Fluid Mech.* **40**, 113–139.

Chung, F. 1997 *Spectral Graph Theory*. American Mathematical Society.

Cohen, R. & Havlin, S. 2010 *Complex Networks: Structure, Robustness and Function*. Cambridge Univ. Press.

Colonius, T. & Taira, K. 2008*a* A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. *Comput. Methods Appl. Mech. Engrg.* **197**, 2131–2146.

Colonius, Tim & Taira, Kunihiko 2008*b* A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. *Comp. Meth. App. Mech. Eng.* **197**, 2131–2146.

Colonius, T. & Taira, K. 2008*c* A fast immersed boundary method using a nullspace approach and multi-domain far-field boundary conditions. *Comput. Methods Appl. Mech. Eng.* **197**, 2131–2146.

Colonius, Tim & Williams, David R 2011 Control of vortex shedding on two-and three-dimensional aerofoils. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **369** (1940), 1525–1539.

Cornelius, S. P., Kath, W. K. & Motter, A. E. 2013 Realistic control of network dynamics. *Nature Comm.* .

Cottet, G.-H. & Koumoutsakos, P. D. 2000 *Vortex methods: theory and practice*. Cambridge Univ. Press.

Cybenko, G. 1989 Approximation by superpositions of a sigmoidal function. *Math. Control Signals Syst.* **2**, 303–314.

Dabiri, J. O. 2009 Optimal vortex formation as a unifying principle in biological propulsion. *Annual Review of Fluid Mechanics* **41**, 17–33.

Darabi, A & Wygnanski, I 2004 Active management of naturally separated flow over a solid surface. part 2. the separation process. *Journal of Fluid Mechanics* **510**, 131–144.

Das, Suddhasattwa & Giannakis, Dimitrios 2017 Delay-coordinate maps and the spectra of koopman operators. *arXiv preprint arXiv:1706.08544* .

Davidson, P. A. 2004 *Turbulence: an introduction for scientists and engineers*. Oxford University Press.

Dawson, Scott TM, Hemati, Maziar S, Williams, Matthew O & Rowley, Clarence W

DISTRIBUTION A: Distribution approved for public release.

2016 Characterizing and correcting for the effect of sensor noise in the dynamic mode decomposition. *Experiments in Fluids* **57** (3), 1–19.

Deng, Z., Chen, Y., Liu, Y. & Kim, K. C. 2019 Time-resolved turbulent velocity field reconstruction using a long short-term memory (lstm)-based artificial intelligence framework. *Phys. Fluids* **31** (7), 075108.

Dickinson, M. H., Farley, C. T., Full, R. J., Koehl, MAR, Kram, R. & Lehman, S. 2000 How animals move: An integrative view. *Science* **288** (5463), 100–106.

Dickinson, M. H., Lehmann, F. & Sane, S. P. 1999 Wing rotation and the aerodynamic basis of insect flight. *Science* **284** (5422), 1954–1960.

Domingos, P. 2012 A few useful things to know about machine learning. *Communications of the ACM* **55** (10), 78–87.

Drineas, Petros, Magdon-Ismail, Malik, Mahoney, Michael W & Woodruff, David P 2012 Fast approximation of matrix coherence and statistical leverage. *Journal of Machine Learning Research* **13** (Dec), 3475–3506.

Drineas, Petros & Mahoney, Michael W. 2016 Randnla: Randomized numerical linear algebra. *Communications of the ACM* **59** (6), 80–90.

Drineas, Petros & Mahoney, Michael W 2017 Lectures on randomized numerical linear algebra. *arXiv preprint arXiv:1712.08880* .

Drmač, Zlatko & Gugercin, Serkan 2016 A new selection operator for the discrete empirical interpolation method—improved a priori error bound and extensions. *SIAM J. Sci. Comput.* **38** (2), A631–A648.

Du, X., Qu, X., He, Y. & Guo, D. 2018 Single image super-resolution based on multi-scale competitive convolutional neural network. *Sensors* **789** (18), 1–17.

Duarte-Cavajalino, J. M., Jahanshad, N., Lenglet, C., McMahon, K. L., de Zubicaray, G. .I., Martin, N. G., Wright, M. J., Thompson, P. M. & Sapiro, G. 2012 Hierarchical topological network analysis of anatomical human brain connectivity and differences related to sex and kinship. *NeuroImage* **59**, 3784–3804.

Duke, Daniel, Soria, Julio & Honnery, Damon 2012 An error analysis of the dynamic mode decomposition. *Experiments in fluids* **52** (2), 529–542.

Duraisamy, K., Iaccarino, G. & Xiao, H. 2019 Turbulence modeling in the age of data. *Annu. Rev. Fluid. Mech.* **51**, 357–377.

Duriez, T., Brunton, S. L. & Noack, B. R. 2017 *Machine learning control-taming nonlinear dynamics and turbulence*. Springer.

Dwyer, K. & Holte, R.t 2007 Decision tree instability and active learning. In *European conference on machine learning*, pp. 128–139. Springer.

Efimov, D., Sacré, P. & Sepulchre, R. 2009 Controlling the phase of an oscillator: a phase response curve approach. In *Proceedings of the 48h IEEE conference on decision and control (CDC)*, pp. 7692–7697.

Eldredge, Jeff D & Jones, Anya R 2019 Leading-edge vortices: mechanics and modeling. *Annual Review of Fluid Mechanics* **51**, 75–104.

Ellens, W., Spieksma, F. M., Van Mieghem, P., Jamakovic, A. & Kooij, R. E. 2011 Effective graph resistance. *Linear algebra and its applications* **435** (10), 2491–2506.

Ellington, C. P., van den Berg, C., Willmott, A. P. & Thomas, A. L. R. 1996 Leading-edge vortices in insect flight. *Nature* **384** (626-630).

Erichson, N. Benjamin, Brunton, Steven L. & Kutz, J. Nathan 2016*a* Compressed dynamic mode decomposition for background modeling. *Journal of Real-Time Image Processing* pp. 1–14.

Erichson, N. B., Brunton, S. L. & Kutz, J. N. 2016*b* Compressed dynamic mode decomposition for real-time object detection. *Journal of Real-Time Image Processing* .

Erichson, N. Benjamin & Donovan, Carl 2016 Randomized low-rank dynamic mode decomposition for motion detection. *Computer Vision and Image Understanding* **146**, 40–50.

Erichson, N. B., Mathelin, L., Yao, Z., Brunton, S. L., Mahoney, M. W. & Kutz, J. N. 2020 Shallow neural networks for fluid flow reconstruction with limited sensors. *Proc. Royal Soc. A* **476** (2238), 20200097.

Farazmand, M. M., Kevlahan, N. K.-R. & Protas, B. 2011 Controlling the dual cascade of two-dimensional turbulence. *J. Fluid Mech.* **668**, 202–222.

DISTRIBUTION A: Distribution approved for public release.

Floyd, R. W. 1962 Algorithm 97: Shortest path. *Comm. ACM* **5** (6), 345.

Fowler, J. E. 2009 Compressive-projection principal component analysis. *IEEE Transactions on Image Processing* **18** (10), 2230–2242.

Frieze, Alan, Kannan, Ravi & Vempala, Santosh 2004 Fast Monte-Carlo algorithms for finding low-rank approximations. *Journal of the ACM* **51** (6), 1025–1041.

Frisch, U. 1995 *Turbulence*. Cambridge Univ. Press.

Fukagata, K., Kasagi, N. & Koumoutsakos, P. 2006 A theoretical prediction of friction drag reduction in turbulent flow by superhydrophobic surfaces. *Phys. Fluids* **18**, 051703.

Fukami, Kai, Fukagata, Koji & Taira, Kunihiko 2018 Super-resolution reconstruction of turbulent flows with machine learning. *arXiv preprint arXiv:1811.11328* .

Fukami, K., Fukagata, K. & Taira, K. 2019*a* Super-resolution reconstruction of turbulent flows with machine learning. *J. Fluid Mech.* **870**, 106–120.

Fukami, Kai, Fukagata, Koji & Taira, Kunihiko 2020 Assessment of supervised machine learning methods for fluid flows. *Theoretical and Computational Fluid Dynamics* **34** (4), 497–519.

Fukami, K., Fukagata, K. & Taira, K. 2021*a* Machine-learning-based spatio-temporal super resolution reconstruction of turbulent flows. *J. Fluid Mech.* **909**, A9.

Fukami, Kai, Maulik, Romit, Ramachandra, Nesar, Fukagata, Koji & Taira, Kunihiko 2021*b* Global field reconstruction from sparse sensors with voronoi tessellation-assisted deep learning. *arXiv preprint arXiv:2101.00554* .

Fukami, K., Nabae, Y., Kawai, K. & Fukagata, K. 2019*b* Synthetic turbulent inflow generator using machine learning. *Phys. Rev. Fluids* **4**, 064603.

Gad-el-Hak, M. 2000 *Flow control: passive, active, and reactive flow management*. Cambridge Univ. Press.

Gamahara, M. & Hattori, Y. 2017 Searching for turbulence models by artificial neural network. *Phys. Rev. Fluids* **2** (5), 054604.

Garcia, C. E., Prett, D. M. & Morari, M. 1989 Model predictive control: theory and practice—a survey. *Automatica* **25** (3), 335–348.

Gavish, Matan & Donoho, David L 2014 The optimal hard threshold for singular values is $4\sqrt{3}$. *IEEE Transactions on Information Theory* **60** (8), 5040–5053.

Gerhard, J., Pastoor, M.and King, R., Noack, B., Dillmann, A., Morzynski, M. & Tadmor, G. 2003 Model-based control of vortex shedding using low-dimensional Galerkin models. In *33rd AIAA Fluid Dynamics Conference and Exhibit*, p. 4262.

Glass, R. J., Glass, L. M., Beyeler, W. E. & Min, H. J. 2006 Targeted social distancing design for pandemic influenza. *Emerg. Infect. Diseases* **12** (11), 1671–1681.

Gopalakrishnan Meena, Muralikrishnan, Nair, Aditya G. & Taira, Kunihiko 2018 Network community-based model reduction for vortical flows. *Physical Review E* **97**, 063103.

Gorodnitsky, Irina F & Rao, Bhaskar D 1994 Analysis of error produced by truncated SVD and Tikhonov regularization methods. In *Proceedings of 1994 28th Asilomar Conference on Signals, Systems and Computers*, pp. 25–29. IEEE.

Goutte, Cyril, Toft, Peter, Rostrup, Egill, Nielsen, Finn Å & Hansen, Lars Kai 1999 On clustering fMRI time series. *NeuroImage* **9** (3), 298–310.

Greenblatt, D. & Wygnanski, I. J. 2000 The control of flow separation by periodic excitation. *Prog. Aero. Sci.* **36**, 487–545.

Greengard, L. & Rokhlin, V. 1987 A fast algorithm for particle summations. *J. Comput. Phys.* **73**, 325–348.

Gu, M. 2015 Subspace iteration randomization and singular value problems. *SIAM Journal on Scientific Computing* **37** (3), A1139–A1173.

Guckenheimer, J. 1975 Isochrons and phaseless sets. *Journal of Mathematical Biology* **1** (3), 259–273.

Guéniat, F., Mathelin, L. & Pastur, L. 2015 A dynamic mode decomposition approach for large and arbitrarily sampled systems. *Physics of Fluids* **27** (2), 025113.

Halko, N., Martinsson, P. G. & Tropp, J. A. 2011 Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. *SIAM Review* **53** (2), 217–288.

DISTRIBUTION A: Distribution approved for public release.

Hansen, Per Christian 1987 The truncatedsvd as a method for regularization. *BIT Numerical Mathematics* **27** (4), 534–553.

Hansen, Per Christian 1990 Truncated singular value decomposition solutions to discrete ill-posed problems with ill-determined numerical rank. *SIAM Journal on Scientific and Statistical Computing* **11** (3), 503–518.

Hansen, Per Christian, Nagy, James G & O'leary, Dianne P 2006 *Deblurring images: matrices, spectra, and filtering*, , vol. 3. Siam.

Hansen, Per Christian, Sekii, Takashi & Shibahashi, Hiromoto 1992 The modified truncated svd method for regularization in general form. *SIAM Journal on Scientific and Statistical Computing* **13** (5), 1142–1150.

Haykin, S. 1998 A comprehensive foundation .

Hemati, M., Eldredge, J. D. & Speyer, J. L. 2014 Improving vortex models via optimal control theory. *Journal of Fluids and Structures* .

Hemati, Maziar S, Rowley, Clarence W, Deem, Eric A. & Cattafesta, Louis N. 2017 De-biasing the dynamic mode decomposition for applied Koopman spectral analysis. *Theoretical and Computational Fluid Dynamics* **31** (4), 349–368.

Herbert, T., Bertolotti, F. P. & Santos, G. R. 1987 Floquet analysis of secondary instability in shear flows. In *Stability of time dependent and spatially varying flows*, pp. 43–57. Springer.

Hinze, J. O. 1975 *Turbulence*. New York: McGraw-Hill.

Holmes, P., Lumley, J. L. & Berkooz, G. 1996 *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Univ. Press.

Holmes, P., Lumley, J. L. & Berkooz, G. 2012*a* *Turbulence, coherent structures, dynamical systems and symmetry*. Cambridge Univ. Press.

Holmes, P. J., Lumley, J. L., Berkooz, G. & Rowley, C. W. 2012*b* *Turbulence, coherent structures, dynamical systems and symmetry*, 2nd edn. Cambridge, England: Cambridge University Press.

Holt, A. B., Wilson, D., Shinn, M., Moehlis, J. & Netoff, T. I. 2016 Phasic burst stimulation: a closed-loop approach to tuning deep brain stimulation parameters for Parkinson's disease. *PLoS Computational Biology* **12** (7), e1005011.

Hornik, K. 1991 Approximation capabilities of multilayer feedforward networks. *Neural Netw.* **4**, 251–257.

Hornik, Kurt, Stinchcombe, Maxwell & White, Halbert 1989 Multilayer feedforward networks are universal approximators. *Neural Networks* **2** (5), 359–366.

Huang, G., Zhu, Q. & Siew, C. 2004 Extreme learning machine: a new learning scheme of feedforward neural networks. In *IEEE International Joint Conference on Neural Networks* pp. 985–990.

Hunt, Julian CR, Wray, Alan A & Moin, Parviz 1998 Eddies, streams, and convergence zones in turbulent flows. *Center for Turbulence Research Report CTR-S88* p. 193.

Joe, W. T., Colonius, T. & MacMynowski, D. G. 2011 Feedback control of vortex shedding from an inclined flat plate. *Theoretical and Computational Fluid Dynamics* **25** (1-4), 221–232.

Johnson, W. B & Lindenstrauss, J. 1984 Extensions of Lipschitz mappings into a Hilbert space. *Contemporary mathematics* **26**, 189–206.

Joslin, R. D. & Miller, D. N., ed. 2009 *Fundamentals and applications of modern flow control*. AIAA.

Kaiser, E., Kutz, J. N. & Brunton, S. L. 2018 Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A* **474** (2219), 20180335.

Kaiser, Eurika, Li, Ruiying & Noack, Bernd R 2017*a* On the control landscape topology. In *The 20th World Congress of the International Federation of Automatic Control (IFAC)*, pp. 1–5.

Kaiser, E., Noack, B. R., Cordier, L., Spohn, A., Segond, M., Abel, M., Daviller, G. & Niven, R. K. 2014*a* Cluster-based reduced-order modelling of a mixing layer. *Journal of Fluid Mechanics* **754**, 365–414.

Kaiser, E., Noack, B. R., Cordier, L., Spohn, A., Segond, M., Abel, M., Daviller, G.,

DISTRIBUTION A: Distribution approved for public release.

Osth, J., Krajnović, S. & Niven, R. K. 2014*b* Cluster-based reduced-order modelling of a mixing layer. *J. Fluid Mech.* **754**, 365–414.

Kaiser, Eurika, Noack, Bernd R, Spohn, Andreas, Cattafesta, Louis N & Morzyński, Marek 2017*b* Cluster-based control of a separating flow over a smoothly contoured ramp. *Theoretical and Computational Fluid Dynamics* pp. 1–15.

Kajishima, T. & Taira, K. 2017 *Computational Fluid Dynamics: Incompressible Turbulent Flows*. Springer.

Kasdin, N Jeremy 1995 Runge-kutta algorithm for the numerical integration of stochastic differential equations. *Journal of Guidance, Control, and Dynamics* **18** (1), 114–120.

Kawamura, Y. & Nakao, H. 2013 Collective phase description of oscillatory convection. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **23** (4), 043129.

Kawamura, Y. & Nakao, H. 2015 Phase description of oscillatory convection with a spatially translational mode. *Physica D: Nonlinear Phenomena* **295**, 11–29.

Kelner, J. & Levin, A. 2011 Spectral sparsification in the semi-streaming setting. *Leibniz International Proceedings in Informatics (LIPIcs) series* **9**, 440–451.

Khodkar, MA, Klamo, Joseph T & Taira, Kunihiko 2020 Phase-locking of laminar wake to periodic vibrations of a circular cylinder. *arXiv preprint arXiv:2011.10149* .

Khodkar, Mohammad Amin & Taira, Kunihiko 2020 Phase-synchronization properties of laminar cylinder wake for periodic external forcings. *Journal of Fluid Mechanics* **904**.

Kim, J. & Bewley, T. R. 2007 A linear systems approach to flow control. *Annual Review of Fluid Mechanics* **39**, 383–417.

Kingma, D. P. & Ba, J. 2014 Adam: A method for stochastic optimization. arXiv:1412.6980.

Klein, Douglas J & Randić, M 1993 Resistance distance. *Journal of Mathematical Chemistry* **12** (1), 81–95.

Kojima, R., Nonomura, T., Oyama, A. & Fujii, K. 2013 Large eddy simulation of low-Reynolds-number flow over thick and think NACA airfoils. *J. Aircraft* **50** (1), 187–196.

Kontovasilis, Kimon P & Mitrou, Nikolas M 1995 Markov-modulated traffic with nearly complete decomposability characteristics and associated fluid queueing models. *Advances in applied probability* **27** (4), 1144–1185.

Koopman, B. O. 1931 Hamiltonian systems and transformation in Hilbert space. *Proceedings of the National Academy of Sciences* **17** (5), 315–318, arXiv: http://www.pnas.org/content/17/5/315.full.pdf+html.

Kraichnan, R. H. & Montgomery, D. 1980 Two-dimensional turbulence. *Rep. Prog. Phys.* **43**, 547–619.

Kramer, Boris, Grover, Piyush, Boufounos, Petros, Benosman, Mouhacine & Nabi, Saleh 2015 Sparse sensing and dmd based identification of flow regimes and bifurcations in complex flows. *arXiv preprint arXiv:1510.02831* .

Kreinovich, V. Y. 1991 Arbitrary nonlinearity is sufficient to represent all functions by neural networks: a theorem. *Neural Netw.* **4**, 381–383.

Kuramoto, Y. 1975 Self-entrainment of a population of coupled non-linear oscillators. In *International symposium on mathematical problems in theoretical physics*, pp. 420–422. Springer.

Kuramoto, Y. 1984 *Chemical oscillations, waves, and turbulence*. Springer.

Kuramoto, Y. & Nakao, H. 2019 On the concept of dynamical reduction: the case of coupled oscillators. *Philosophical Transactions of the Royal Society A* **377** (2160), 20190041.

Kutz, J. N. 2017 Deep learning in fluid dynamics. *J. Fluid Mech.* **814**, 1–4.

Kutz, J. N., Brunton, S. L., Brunton, B. W. & Proctor, J. L. 2016*a* *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems*. SIAM.

Kutz, J. N., Fu, X. & Brunton, S. L. 2016*b* Multi-resolution dynamic mode decomposition. *SIAM Journal on Applied Dynamical Systems* **15** (2), 713–735, preprint. Available: arXiv:1506.00564.

Kutz, J Nathan, Fu, Xing, Brunton, Steve L & Erichson, N Benjamin 2015 Multi-resolution dynamic mode decomposition for foreground/background separation and object tracking pp. 921–929.

Lasota, Andrzej & Mackey, Michael C 2013 *Chaos, fractals, and noise: stochastic aspects of dynamics*, , vol. 97. Springer Science & Business Media.

Lassila, T., Manzoni, A., Quarteroni, A. & Rozza, G. 2014 Model order reduction in

DISTRIBUTION A: Distribution approved for public release.

fluid dynamics: challenges and perspectives. In *Reduced Order Methods for modeling and computational reduction*, pp. 235–273. Springer.

LeCun, Y., Bottou, L., Bengio, Y. & Haffner, P. 1998 Gradient-based learning applied to document recognition. *Proc. IEEE* **86** (11), 2278–2324.

Lee, Donghoon & Wiswall, Matthew 2007 A parallel implementation of the simplex function minimization routine. *Computational Economics* **30** (2), 17–187.

Lehoucq, RB, Sorensen, DC & Yang, C 1997 Arpack users' guide: Solution of large scale eigenvalue problems with implicitly restarted arnoldi methods. *Software Environ. Tools* **6**.

Leicht, Elizabeth A & Newman, Mark EJ 2008 Community structure in directed networks. *Physical review letters* **100** (11), 118703.

Leonard, Anthony 1980 Vortex methods for flow simulation. *Journal of Computational Physics* **37** (3), 289–335.

Leoni, P., C. D., Mazzino, A. & Biferale, L. 2018 Inferring flow parameters and turbulent configuration with physics-informed data assimilation and spectral nudging. *Phys. Rev. Fluids* **3** (10), 104604.

Lesieur, M. 2008 *Turbulence in fluids*, 4th edn. Springer.

Ling, J., Kurzawski, A. & Templeton, J 2016 Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *J. Fluid Mech.* **807**, 155–166.

Ling, J. & Templeton, J. 2015 Evaluation of machine learning algorithms for prediction of regions of high Reynolds averaged Navier Stokes uncertainty. *Phys. Fluids* **27** (8), 085103.

Liu, C, Zheng, X & Sung, CH 1998 Preconditioned multigrid methods for unsteady incompressible flows. *Journal of Computational Physics* **139** (1), 35–57.

Liu, Yang-Yu, Slotine, Jean-Jacques & Barabási, Albert-László 2011 Controllability of complex networks. *Nature* **473** (7346), 167–173.

Lloyd, Stuart 1982 Least squares quantization in PCM. *IEEE transactions on information theory* **28** (2), 129–137.

Lloyd-Smith, J., Schreiber, S., Kopp, P. & Getz, W. 2005 Superspreading and the effect of individual variation on disease emergence. *Nature* **438**, 355–359.

Loiseau, Jean-Christophe, Noack, Bernd R & Brunton, Steven L 2017 Sparse reduced-order modeling: Sensor-based dynamics to full-state estimation. *arXiv preprint arXiv:1706.03531* .

Loiseau, J.-C., Noack, B. R. & Brunton, S. L. 2018 Sparse reduced-order modelling: sensor-based dynamics to full-state estimation. *Journal of Fluid Mechanics* **844**, 459–490.

Lomax, Richard G & Hahs-Vaughn, Debbie L 2013 *Statistical concepts: A second course*. Routledge.

Luchtenburg, D. M., Günther, B., Noack, B. R., King, R. & Tadmor, G. 2009 A generalized mean-field model of the natural and high-frequency actuated flow around a high-lift configuration. *J. Fluid Mech.* **623**, 283–316.

Luersen, MA, Le Riche, Rodolphe & Guyon, F 2004 A constrained, globalized, and bounded nelder–mead method for engineering optimization. *Structural and Multidisciplinary Optimization* **27** (1-2), 43–54.

Lui, H. F. S. & Wolf, W. R. 2019 Construction of reduced-order models for fluid flows using deep feedforward neural networks. *J. Fluid Mech.* **872**, 963–994.

Lusseyran, F., Gueniat, F., Basley, J., Douay, C. L., Pastur, L. R., Faure, T. M. & Schmid, P. J. 2011 Flow coherent structures and frequency signature: application of the dynamic modes decomposition to open cavity flow. In *Journal of Physics: Conference Series*, , vol. 318, p. 042036. IOP Publishing.

Ma, Ping, Mahoney, Michael W & Yu, Bin 2015 A statistical perspective on algorithmic leveraging. *The Journal of Machine Learning Research* **16** (1), 861–911.

Ma, Z., Ahuja, S. & Rowley, C. W. 2009 Reduced order models for control of fluids using the eigensystem realization algorithm. *Theo. Comp. Fluid Dyn.* **25** (1), 233–247.

MacQueen, J. 1967 Some methods for classification and analysis of multivariate observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, , vol. 1, pp. 281–297. Oakland, CA, USA.

Mahoney, Michael W. 2011 Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning* **3** (2), 123–224.

DISTRIBUTION A: Distribution approved for public release.

Malkin, IG 1949 Methods of Poincare and Lyapunov in theory of nonlinear oscillations. *Gostexizdat, Moscow* .

Manohar, Krithika, Brunton, Bingni W., Kutz, J. Nathan & Brunton, Steven L. 2018 Data-driven sparse sensor placement. *IEEE Control Systems Magazine* **38** (3), 63–86.

Mao, X, Blackburn, HM & Sherwin, SJ 2015 Nonlinear optimal suppression of vortex shedding from a circular cylinder. *J. Fluid Mech.* **775**, 241–265.

Martinsson, Per-Gunnar 2016 Randomized methods for matrix computations. *arXiv preprint arXiv:1607.01649* .

Martinsson, Per-Gunnar & Voronin, Sergey 2016 A randomized blocked algorithm for efficiently computing rank-revealing factorizations of matrices. *SIAM Journal on Scientific Computing* **38** (5), S485–S507.

Maulik, R., Fukami, K., Ramachandra, N., Fukagata, K. & Taira, K. 2020*a* Probabilistic neural networks for fluid flow surrogate modeling and data recovery. *Phys. Rev. Fluids* **5**, 104401.

Maulik, R., Mohan, A., Lusch, B., Madireddy, S. & Balaprakash, P. 2020*b* Time-series learning of latent-space dynamics for reduced-order model closure. *Physica D: Nonlinear Phenomena* **405**, 132368.

Maulik, R. & San, O. 2017 A neural network approach for the blind deconvolution of turbulent flows. *J. Fluid Mech.* **831**, 151–181.

Maulik, R., San, O., Rasheed, A. & Vedula, P. 2019 Sub-grid modelling for two-dimensional turbulence using neural networks. *J. Fluid Mech.* **858**, 122–144.

Mauroy, A. & Mezić, I. 2012 On the use of Fourier averages to compute the global isochrons of (quasi) periodic dynamics. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **22** (3), 033112.

Mauroy, A. & Mezić, I. 2018 Global computation of phase-amplitude reduction for limit-cycle dynamics. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **28** (7), 073108.

Mauroy, A., Mezić, I. & Moehlis, J. 2013 Isostables, isochrons, and Koopman spectrum for the action–angle representation of stable fixed point dynamics. *Physica D: Nonlinear Phenomena* **261**, 19–30.

McKay, Michael D, Beckman, Richard J & Conover, William J 2000 A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics* **42** (1), 55–61.

McWilliams, J. C. 1984 The emergence of isolated coherent vortices in turbulent flow. *J. Fluid Mech.* **146**, 21–43.

Meena, M. Gopalakrishnan, Nair, A. G. & Taira, K. 2018*a* Network community-based model reduction for vortical flows. *Phys. Rev. E* **97** (6), 063103.

Meena, M. Gopalakrishnan, Taira, K. & Asai, K. 2018*b* Airfoil wake modification with gurney flap at low Reynolds number. *AIAA J.* **56** (4), 1348–1359.

Mesbahi, M. & Egerstedt, M. 2010 *Graph theoretic methods in multiagent networks*. Princeton Univ. Press.

Mezić, Igor 2005 Spectral properties of dynamical systems, model reduction and decompositions. *Nonlinear Dynamics* **41** (1-3), 309–325.

Mezic, Igor 2013 Analysis of fluid flows via spectral properties of the Koopman operator. *Annu. Rev. Fluid Mech.* **45**, 357–378.

Mieghem, P. V. 2011 *Graph spectra for complex networks*. Cambridge University Press.

Miyanawala, T. P. & Jaiman, R. K. 2018 A novel deep learning method for the predictions of current forces on bluff bodies. In *ASME 2018 37th International Conference on Ocean, Offshore and Arctic Engineering*. American Society of Mechanical Engineers Digital Collection.

Moehlis, J. 2014 Improving the precision of noisy oscillators. *Physica D: Nonlinear Phenomena* **272**.

Moehlis, J., Shea-Brown, E. & Rabitz, H. 2006 Optimal inputs for phase models of spiking neurons. *Journal of Computational and Nonlinear Dynamics* **1** (4), 358–367.

Mohar, B. 1991 The Laplacian spectrum of graphs. In *Graph theory, combinatorics, and applications* (ed. Y. Alavi, G. Chartrand, O. Ollermann & A. Schwenk), pp. 871–898. New York: Wiley.

DISTRIBUTION A: Distribution approved for public release.

Monga, B. & Moehlis, J. 2019 Optimal phase control of biological oscillators using augmented phase reduction. *Biological Cybernetics* **113** (1-2), 161–178.

Monga, B., Wilson, D., Matchen, T. & Moehlis, J. 2019 Phase reduction and phase-based optimal control for biological systems: a tutorial. *Biological Cybernetics* **113** (1-2), 11–46.

Morari, M. & Lee, J. H. 1999 Model predictive control: past, present and future. *Computers & Chemical Engineering* **23** (4-5), 667–682.

Morris, M. 1993 Epidemiology and social networks - modeling structured diffusion. *Sociol. Method Res.* **22**, 99–126.

Munday, P. M. & Taira, K. 2013 On the lock-on of vortex shedding to oscillatory actuation around a circular cylinder. *Phys. Fluids* **25**, 013601.

Munday, Phillip M & Taira, Kunihiko 2014 Separation control on NACA 0012 airfoil using momentum and wall-normal vorticity injection. In *32nd AIAA Applied Aerodynamics Conference*.

Murata, T., Fukami, K. & Fukagata, K. 2020 Nonlinear mode decomposition with convolutional neural networks for fluid dynamics. *J. Fluid Mech.* **882**, A13.

Nair, Aditya G, Brunton, Steven L & Taira, Kunihiko 2018 Networked-oscillator-based modeling and control of unsteady wake flows. *Physical Review E* **97** (6), 063107.

Nair, A. G. & Taira, K. 2015 Network-theoretic approach to sparsified discrete vortex dynamics. *J. Fluid Mech.* **768**, 549–571.

Nair, Aditya G., Yeh, Chi-An, Kaiser, Eurika, Noack, Bernd R., Brunton, Steven L. & Taira, Kunihiko 2019 Cluster-based feedback control of turbulent post-stall separated flows. *Journal of Fluid Mechanics* **875**, 345–375.

Nair, V. & Hinton, G. E. 2010 Rectified linear units improve restricted boltzmann machines. *Proc. Int. Conf. Mach. Learn.* pp. 807–814.

Nakao, H. 2016 Phase reduction approach to synchronisation of nonlinear oscillators. *Contemporary Physics* **57** (2), 188–214.

Nelder, John A & Mead, Roger 1965 A simplex method for function minimization. *The computer journal* **7** (4), 308–313.

Newman, Mark 2010 *Networks: an introduction*. Oxford University Press.

Newman, M. E. J. 2003 The structural and function of complex networks. *SIAM Rev.* **45** (2), 167–256.

Newman, M. E. J. 2004 Fast algorithm for detecting community structure in networks. *Physical review E* **69**, 066133.

Newton, P. K. 2001 *The N-vortex problem: analytical techniques*, *Applied Mathematical Sciences*, vol. 145. Springer.

Noack, Bernd R, Afanasiev, Konstantin, Morzyński, Marek, Tadmor, Gilead & Thiele, Frank 2003*a* A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *Journal of Fluid Mechanics* **497**, 335–363.

Noack, B. R., Afanasiev, K., Morzynski, M., Tadmor, G. & Thiele, F. 2003*b* A hierarchy of low-dimensional models for the transient and post-transient cylinder wake. *J. Fluid Mech.* **497**, 335–363.

Noack, B. R., Morzynski, M. & Tadmor, G. 2011 *Reduced-order modelling for flow control*. Springer-Verlag.

Noack, B. R., Papas, P. & Monkewitz, P. A. 2005 The need for a pressure-term representation in empirical Galerkin models of incompressible shear flows. *J. Fluid Mech.* **523**, 339–365.

Noack, Bernd R, Schlegel, Michael, Ahlborn, Boye, Mutschke, Gerd, Morzyński, Marek & Comte, Pierre 2008 A finite-time thermodynamics of unsteady fluid flows. *Journal of Non-Equilibrium Thermodynamics* **33** (2), 103–148.

Noack, B. R., Stankiewicz, W., Morzynski, M. & Schmid, P. J. 2016*a* Recursive dynamic mode decomposition of a transient cylinder wake. *J. Fluid Mech.* .

Noack, Bernd R, Stankiewicz, Witold, Morzynski, Marek & Schmid, Peter J 2016*b* Recursive dynamic mode decomposition of a transient cylinder wake. *Journal of Fluid Mechanics* **809**, 843–872.

Norris, James R 1998 *Markov chains*. Cambridge Univ. Press.

Opitz, D. & Maclin, R. 1999 Popular ensemble methods: An empirical study. *J. Artif. Intell. Res.* **11**, 169–198.

DISTRIBUTION A: Distribution approved for public release.

Owen, J. P., Li, Y.-O., Ziv, E., Strominger, Z., Gold, J., Bukhpun, P., Wakahiro, M., Friedman, E. J., Sherr, E. H. & Mukherjee, P. 2013 The structural connectome of the human brain in agenesis of the corpus callosum. *NeuroImage* **70**, 340–355.

Pastoor, M., Henning, L., Noack, B. R., King, R. & Tadmor, G. 2008 Feedback shear layer control for bluff body drag reduction. *J. Fluid Mech.* **608**, 161–196.

Peleg, D. & Ullman, J. 1989 An optimal synchronizer for the hypercube. *SIAM Journal on computing* **18** (4), 740–747.

Phillips, David L 1962 A technique for the numerical solution of certain integral equations of the first kind. *Journal of the ACM (JACM)* **9** (1), 84–97.

Pope, S. B. 2000 *Turbulent flows*. Cambridge Univ. Press.

Porter, M. A., Mucha, P. J., Newman, M. E. J. & Warmbrand, C. M. 2005 A network analysis of committees in the U.S. House of Representatives. *Proc. Nat. Acad. Sci.* **102** (20), 7057–7062.

Proctor, J. L., Brunton, S. L. & Kutz, J. N. 2016*a* Dynamic mode decomposition with control. *SIAM J. Appl. Dyn. Syst.* **15** (1), 142–161.

Proctor, Joshua L, Brunton, Steven L & Kutz, J Nathan 2016*b* Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems* **15** (1), 142–161.

Proctor, Joshua L & Eckhoff, Philip A 2015 Discovering dynamic patterns from infectious disease data using dynamic mode decomposition. *International health* **7** (2), 139–145.

Quinlan, J. R. 1986 Induction of decision trees. *Mach. Learn.* **1** (1), 81–106.

Rabault, J., Kuchta, M., Jensen, A., Réglade, U. & Cerardi, N. 2019 Artificial neural networks trained through deep reinforcement learning discover control strategies for active flow control. *J. Fluid Mech.* **865**, 281–302.

Ramananarivo, S., Godoy-Diana, R. & Thiria, B. 2011 Rather than resonance, flapping wing flyers may play on aerodynamics to improve performance. *Proceedings of the National Academy of Sciences* **108** (15), 5964–5969.

Rempfer, D 2000 On low-dimensional galerkin models for fluid flow. *Theoretical and Computational Fluid Dynamics* **14** (2), 75–88.

Reynolds, Richard W, Rayner, Nick A, Smith, Thomas M, Stokes, Diane C & Wang, Wanqiu 2002 An improved in situ and satellite SST analysis for climate. *Journal of climate* **15** (13), 1609–1625.

Reynolds, Richard W, Smith, Thomas M, Liu, Chunying, Chelton, Dudley B, Casey, Kenneth S & Schlax, Michael G 2007 Daily high-resolution-blended analyses for sea surface temperature. *Journal of Climate* **20** (22), 5473–5496.

Robinson, K., Cohen, T. & Colijn, C. 2012 The dynamics of sexual contact networks: effects on disease spread and control. *Theo. Popul. Bio.* **81**, 89–96.

Rokach, Lior & Maimon, Oded 2005 Clustering methods. In *Data mining and knowledge discovery handbook*. Springer.

Rokhlin, Vladimir, Szlam, Arthur & Tygert, Mark 2009 A randomized algorithm for principal component analysis. *SIAM Journal on Matrix Analysis and Applications* **31** (3), 1100–1124.

Roma, A. M., Peskin, C. S. & Berger, M. J. 1999 An adaptive version of the immersed boundary method. *J. Comput. Phys.* **153**, 509–534.

Roshko, A 1954 On the drag and shedding frequency of two-dimensional bluff bodies. *National Advisory Committee on Aeronautic* .

Roshko, A. 1976 Structure of turbulent shear flows: a new look. *AIAA J.* **14** (10), 1349–1357.

Rowley, C., Colonius, T. & Murray, R. 2004 Model reduction for compressible flows using POD and Galerkin projection. *Physica D: Nonlinear Phenomena* **189** (1), 115–129.

Rowley, C. W. 2005 Model reduction for fluids, using balanced proper orthogonal decomposition. *Int. J. Bif. Chaos* **15** (3), 997–1013.

Rowley, C. W., Mezić, I., Bagheri, S. & Henningson, D. S. 2009*a* Spectral analysis of nonlinear flows. *J. Fluid Mech.* **641**, 115–127.

Rowley, C. W., Mezic, I., Bagheri, S., Schlatter, P. & Henningson, D.S. 2009*b* Spectral analysis of nonlinear flows. *J. Fluid Mech.* **645**, 115–127.

Rubinov, M. & Sporns, O. 2010 Complex network measures of brain connectivity: Uses and interpretations. *NeuroImage* **52**, 1059–1069.

DISTRIBUTION A: Distribution approved for public release.

Rumelhart, D. E., Hinton, G. E. & Williams, R. J. 1986 Learning representations by back-propagation errors. *Nature* **322**, 533—536.

Saffman, P. G. 1992 *Vortex dynamics*. Cambridge Univ. Press.

Saffman, P. G. & Baker, G. R. 1979 Vortex interactions. *Annu. Rev. Fluid Mech.* **11** (1), 95–121.

Salathé, M. & Jones, J. H. 2010 Dynamics and control of diseases in networks with community structure. *PLoS Comput. Bio.* **6** (4), e1000736.

Salehipour, H. & Peltier, W. R. 2019 Deep learning of mixing by two 'atoms' of stratified turbulence. *J. Fluid Mech.* **861**, R4.

Salvador, J. 2016 *Example-based super resolution*. Elsevier Academic Press.

San, O. & Maulik, R. 2018 Extreme learning machine for reduced order modeling of turbulent geophysical flows. *Phys. Rev. E* **97**, 04322.

Sarlos, Tamas 2006 Improved approximation algorithms for large matrices via random projections. In *Foundations of Computer Science. 47th Annual IEEE Symposium on*, pp. 143–152.

Sarpkaya, T. 2004 A critical review of the intrinsic nature of vortex-induced vibrations. *Journal of Fluids and Structures* **19** (4), 289–447.

Sayadi, Taraneh & Schmid, Peter J 2016 Parallel data-driven decomposition algorithm for large-scale datasets: with application to transitional boundary layers. *Theoretical and Computational Fluid Dynamics* pp. 1–14.

Schmid, PJ, Li, L, Juniper, MP & Pust, O 2011 Applications of the dynamic mode decomposition. *Theor. Comp. Fluid Dyn.* **25** (1-4), 249–259.

Schmid, P. J. 2010*a* Dynamic mode decomposition of numerical and experimental data. *J. Fluid Mech.* **656**, 5–28.

Schmid, P. J. 2010*b* Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics* **656**, 5–28.

Schultheiss, N. W., Prinz, A. A. & Butera, R. J. 2011 *Phase response curves in neuroscience: theory, experiment, and analysis*. Springer.

Seena, Abu & Sung, Hyung Jin 2011 Dynamic mode decomposition of turbulent cavity flows for self-sustained oscillations. *Int. J. Heat Fluid Fl.* **32** (6), 1098–1110.

Serre, D. 2002 Matrices: Theory and applications .

Shanker, M., Hu, M. Y. & Hung, M. S. 1996 Effect of data standardization on neural network training. *Omega* **24** (4), 385–397.

Shirasaka, S., Kurebayashi, W. & Nakao, H. 2017 Phase-amplitude reduction of transient dynamics far from attractors for limit-cycling systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **27** (2), 023119.

Shyy, W., Kang, C., Chirarattananon, P., Ravi, S. & Liu, H. 2016 Aerodynamics, sensing and control of insect-scale flapping-wing flight. *Proceedings of the Royal Society A* **472** (2186), 20150712.

Shyy, W., Lian, Y., Tang, J., Viieru, D. & Liu, H. 2008 *Aerodynamics of low Reynolds number flyers*. Cambridge Univ. Press.

Sieber, M., Paschereit, C. O. & Oberleithner, K. 2016 Spectral proper orthogonal decomposition. *J. Fluid Mech.* **792**, 798–828.

Siegel, S., Cohen, K. & McLaughlin, T. 2003 Feedback control of a circular cylinder wake in experiment and simulation. In *33rd AIAA Fluid Dynamics Conference and Exhibit*.

Sipp, D. & Lebedev, A. 2007 Global stability of base and mean flows: A general approach and its applications to cylinder and open cavity flows. *Journal of Fluid Mechanics* **593**, 333–358.

Sirovich, L. 1987 Turbulence and the dynamics of coherent structures, Parts I-III. *Q. Appl. Math.* **XLV**, 561–590.

Smith, Thomas M & Reynolds, Richard W 2005 A global merged land–air–sea surface temperature reconstruction based on historical observations (1880–1997). *Journal of Climate* **18** (12), 2021–2036.

Smola, A. J. & Schölkopf, B. 2004 A tutorial on support vector regression. *Stat. Comput.* **14** (3), 199–222.

Spielman, D. A. & Srivastava, N. 2011 Graph sparsification by effective resistances. *SIAM J. Comput.* **40** (6), 1913–1926.

DISTRIBUTION A: Distribution approved for public release.

Spielman, D. A. & Teng, S.-H. 2011 Spectral sparsification of graphs. *SIAM J. Comput.* **40** (4), 981–1025.

Sponberg, S & Daniel, TL 2012 Abdicating power for control: A precision timing strategy to modulate function of flight power muscles. *Proceedings of the Royal Society B* **279** (1744), 3958–3966.

Srinivasan, P. A., Guastoni, L., Azizpour, H., Schlatter, P. & Vinuesa, R 2019 Predictions of turbulent shear flows using deep neural networks. *Phys. Rev. Fluids* **4**, 054603.

Srivastava, N. 2010 Spectral sparsification and restricted invertibility. PhD thesis, Yale University.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. 2014 Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research* **15** (1), 1929–1958.

Strogatz, S. H. 2014 *Nonlinear dynamics and chaos: with applications to physics, biology, chemistry, and engineering*. Westview press.

Stuart, JT 1960 On the non-linear mechanics of wave disturbances in stable and unstable parallel flows part 1. the basic behaviour in plane poiseuille flow. *Journal of Fluid Mechanics* **9** (3), 353–370.

Taira, K. 2019 Revealing essential dynamics from high-dimensional fluid flow data and operators. arXiv:1903.01913 .

Taira, K., Brunton, S. L., Dawson, S., Rowley, C. W., Colonius, T., McKeon, B. J., Schmidt, O. T., Gordeyev, S., Theofilis, V. & Ukeiley, L. S. 2017*a* Modal analysis of fluid flows: An overview. *in review, AIAA Journal* .

Taira, Kunihiko, Brunton, Steven L, Dawson, Scott, Rowley, Clarence W, Colonius, Tim, McKeon, Beverley J, Schmidt, Oliver T, Gordeyev, Stanislav, Theofilis, Vassilios & Ukeiley, Lawrence S 2017*b* Modal analysis of fluid flows: An overview. *AIAA Journal* **55** (12), 4013–4041.

Taira, K., Brunton, S. L., Dawson, S. T. M., Rowley, C. W., Colonius, T., McKeon, B. J., Schmidt, O. T., Gordeyev, S., Theofilis, V. & Ukeiley, L. S. 2017*c* Modal analysis of fluid flows: An overview. *AIAA J.* **55** (12), 4013–4041.

Taira, K. & Colonius, T. 2007*a* The immersed boundary method: a projection approach. *J. Comput. Phys.* **225**, 2118–2137.

Taira, Kunihiko & Colonius, Tim 2007*b* The immersed boundary method: A projection approach. *J. Comp. Phys.* **225**, 2118–2137.

Taira, K. & Colonius, T. 2007*c* The immersed boundary method: A projection approach. *J. Comput. Phys.* **225** (2), 2118–2137.

Taira, K., Hemati, M. S., Brunton, S. L., Sun, Y., Duraisamy, K., Bagheri, S., Dawson, S. & Yeh, C.-A 2020*a* Modal analysis of fluid flows: Applications and outlook. *AIAA J.* **58** (3), 998–1022.

Taira, Kunihiko, Hemati, Maziar S., Brunton, Steven L., Sun, Yiyang, Duraisamy, Karthik, Bagheri, Shervin amd Dawson, Scott T. M. & Yeh, Chi-An 2020*b* Modal analysis of fluid flows: applications and outlook. *AIAA Journal* pp. 998–1022.

Taira, K., Nair, A. G. & Brunton, S. L. 2016*a* Network structure of two-dimensional decaying isotropic turbulence. *J. Fluid Mech.* **795**, R2.

Taira, K., Nair, A. G. & Brunton, S. L. 2016*b* Network structure of two-dimensional decaying isotropic turbulence. *J. Fluid Mech.* **795**.

Taira, Kunihiko & Nakao, Hiroya 2018 Phase-response analysis of synchronization for periodic flows. *Journal of Fluid Mechanics* **846**.

Takeishi, Naoya, Kawahara, Yoshinobu, Tabei, Yasuo & Yairi, Takehisa 2017 Bayesian dynamic mode decomposition. *Twenty-Sixth International Joint Conference on Artificial Intelligence* .

Taylor, G. I. 1918 On the dissipation of eddies. Reports and Memoranda 598. Aero. Res. Comm.

Taylor, Roy, Kutz, J Nathan, Morgan, Kyle & Nelson, Brian 2017 Dynamic mode decomposition for plasma diagnostics and validation. *arXiv preprint arXiv:1702.06871* .

Tennekes, H. & Lumley, J. L. 1972 *A first course in turbulence*. Cambridge: MIT Press.

Thyng, K. M., Greene, C. A., Hetland, R. D., Zimmerle, H. M. & DiMarco, S. F. 2016

DISTRIBUTION A: Distribution approved for public release.

True colors of oceanography: Guidelines for effective and accurate colormap selection. *Oceanography* **29** (3), 9–13.

Tikhonov, Andrei Nikolajevits 1963 Solution of incorrectly formulated problems and the regularization method. *Soviet Math.* **4**, 1035–1038.

Tropp, Joel A 2011 Improved analysis of the subsampled randomized hadamard transform. *Advances in Adaptive Data Analysis* **3** (01n02), 115–126.

Tu, J. H., Rowley, C. W., Luchtenburg, D. M., Brunton, S. L. & Kutz, J. N. 2014 On dynamic mode decomposition: theory and applications. *Journal of Computational Dynamics* **1** (2), 391–421.

Vapnik, V. & Lerner, A. 1963 Pattern recognition using generalized portrait method. *Automation and remote control* **24**, 774–780.

Varah, JM 1979 A practical examination of some numerical methods for linear discrete ill-posed problems. *Siam Review* **21** (1), 100–111.

Voronin, Sergey & Martinsson, Per-Gunnar 2015 RSVDPACK: Subroutines for computing partial singular value decompositions via randomized sampling on single core, multi core, and GPU architectures, arXiv: 1502.05366.

Vreman, A. W. 2004 An eddy-viscosity subgrid-scale model for turbulent shear flow: Algebraic theory and applications. *Phys. Fluids* **16** (10), 3670–3681.

Wand, Matt P & Jones, M Chris 1994 *Kernel smoothing*. CRC Press.

Wang, C & Eldredge, J 2013 Low-order phenomenological modeling of leading-edge vortex formation. *Theoretical and Computational Fluid Dynamics* **27** (5), 577–598.

Wang, Jia, Wang, Donghai, Lallemand, Pierre & Luo, Li-Shi Shi 2013 Lattice Boltzmann simulations of thermal convective flows in two dimensions. *Computers and Mathematics with Applications* **65** (2), 262–286.

Weiss, J. B., Provenzale, A. & McWilliams, J. C. 1998 Lagrangian dynamics in high-dimensional point-vortex systems. *Phys. Fluids* **10** (8), 1929–1941.

Williams, D. R., Tadmor, G., Colonius, T., Kerstens, W., Quach, V. & Buntain, S. 2009 Lift response of a stalled wing to pulsatile disturbances. *AIAA Journal* **47** (12), 3031–3037.

Williamson, C. H. K. & Govardhan, R. 2004 Vortex-induced vibrations. *Annu. Rev. Fluid Mech.* **36**, 413–455.

Wilson, Dan 2020 A data-driven phase and isostable reduced modeling framework for oscillatory dynamical systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science* **30** (1), 013121.

Wilson, D. & Moehlis, J. 2016 Isostable reduction of periodic orbits. *Physical Review E* **94** (5), 052213.

Winfree, A. T. 2001 *The geometry of biological time*. Springer.

Woodruff, David P & others 2014 Sketching as a tool for numerical linear algebra. *Foundations and Trends in Theoretical Computer Science* **10** (1–2), 1–157.

Wu, T. Y. 2011 Fish swimming and bird/insect flight. *Annual Review of Fluid Mechanics* **43**, 25–58.

Yan, G., Tsekenis, G., Barzel, B., Slotine, J.-J., Liu, Y.-Y. & Barabási, A.-L. 2015 Spectrum of controlling and observing complex networks. *Nature Physics* .

Yeh, Chi-An, Gopalakrishnan Meena, Muralikrishnan & Taira, Kunihiko 2021 Network broadcast analysis and control of turbulent flows. *Journal of Fluid Mechanics* **910** (A15).

Yeh, Chi-An, Munday, Phillip & Taira, Kunihiko 2017 Use of local periodic heating for separation control on a NACA 0012 airfoil. *AIAA Paper* p. 1451.

Young, Gale & Householder, Aiston S 1938 Discussion of a set of points in terms of their mutual distances. *Psychometrika* **3** (1), 19–22.

Yu, J. & Hesthaven, J. S. 2019 Flowfield reconstruction method using artificial neural network. *AIAA J.* **57(2)**, 482–498.

Zhang, Y., Sung, W. J. & Mavris, D. N. 2018 Application of convolutional neural network to predict airfoil lift coefficient. In *AIAA paper*, pp. 2018–1903.

DISTRIBUTION A: Distribution approved for public release.