Advances in resolvent-based modeling of turbulent flows

Aaron Towne

University of Michigan
Mechanical Engineering

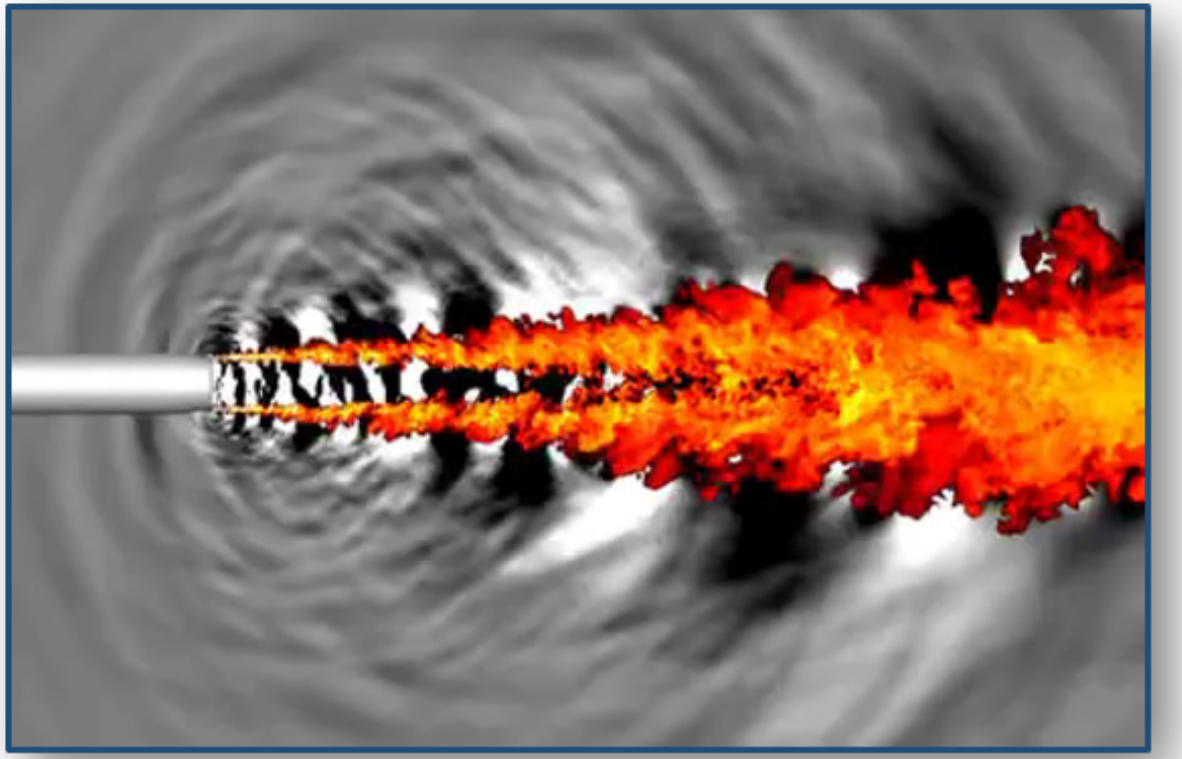
towne@umich.edu

Advanced Modeling & Simulation (AMS) Seminar Series
NASA Ames Research Center
October 5th, 2023

Reduced-complexity modeling of turbulent flows

- Turbulence is
  - Chaotic
  - Multi-scale
  - High-dimensional
- Simplified models can help us
  - Understand key physics
  - Obtain low-cost predictions
  - Develop control strategies
  - … and more …
- Often, models based on coherent structures

Turbulent jet

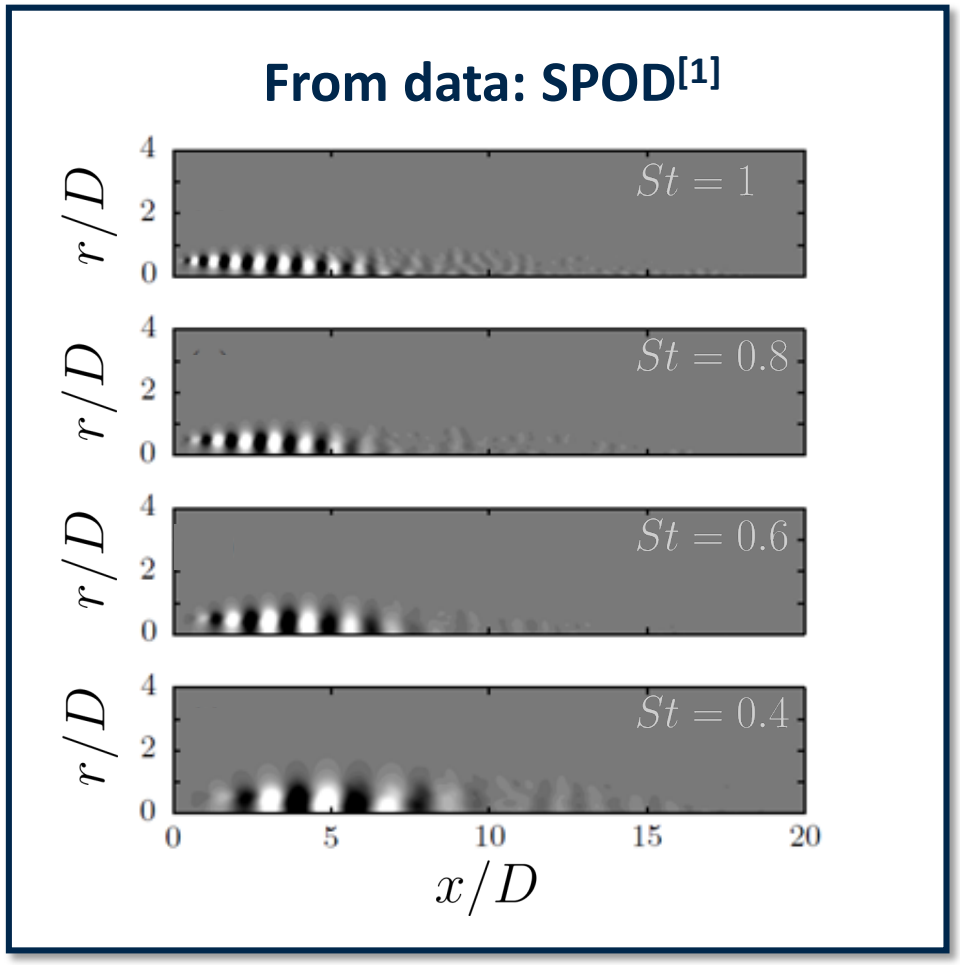


Temperature

Pressure

Plane cut through the jet centerline

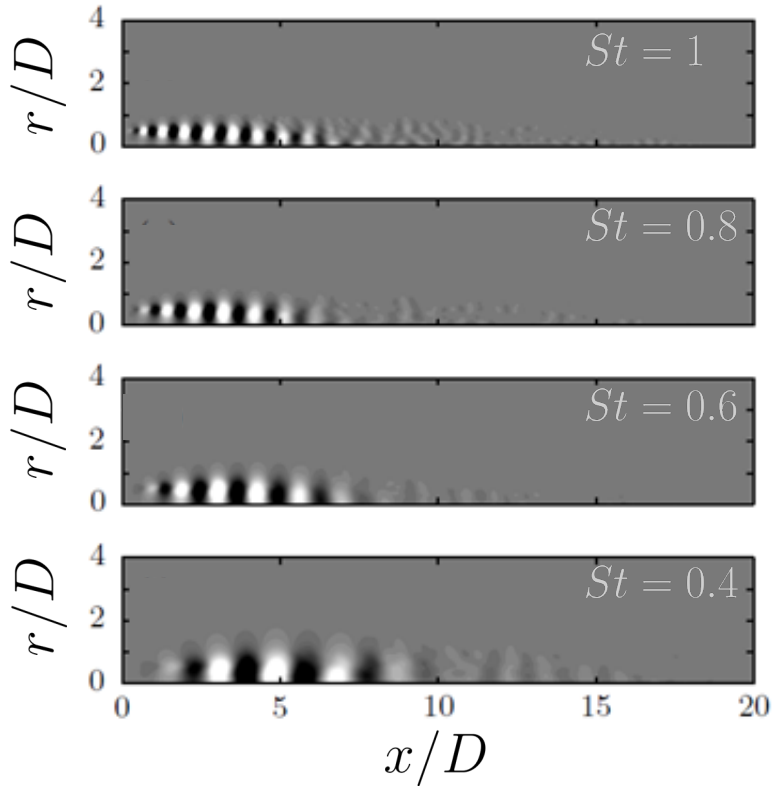
Extracting coherent structures from data



[1] Towne et al., *JFM*, 2018

Modeling coherent structures

From data: SPOD[1]



Model: Resolvent

Focus of this presentation

[1] Towne et al., *JFM*, 2018

Resolvent analysis formulation

$$\frac{\partial \boldsymbol{q}}{\partial t} - \boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{0} \xrightarrow{\boldsymbol{q}(\boldsymbol{x},t) = \bar{\boldsymbol{q}}(\boldsymbol{x}) + \boldsymbol{q}'(\boldsymbol{x},t)} \frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}' = \boldsymbol{f}$$

Linear Nonlinear

Resolvent analysis formulation

$$\frac{\partial \boldsymbol{q}}{\partial t} - \boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{0}$$

$$\boldsymbol{q}(\boldsymbol{x},t) = \bar{\boldsymbol{q}}(\boldsymbol{x}) + \boldsymbol{q}'(\boldsymbol{x},t)$$

$$\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}' = \boldsymbol{f}$$

Fourier Transform

$$\hat{\boldsymbol{q}} = \underbrace{(i\omega I - A)^{-1}}_{R(\omega)}\hat{\boldsymbol{f}}$$

Resolvent analysis formulation

$$\frac{\partial \boldsymbol{q}}{\partial t} - \boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{0}$$

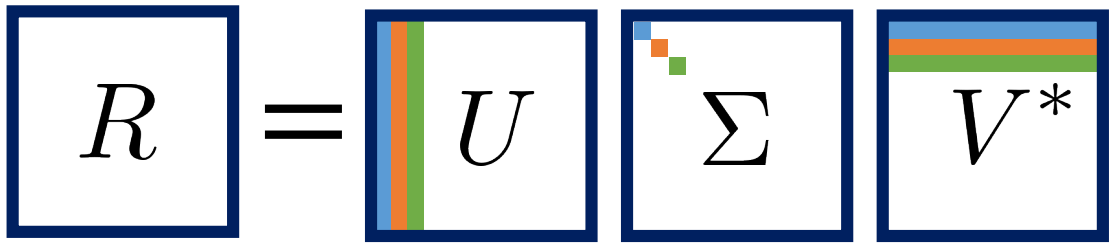
$\boldsymbol{q}(\boldsymbol{x},t) = \bar{\boldsymbol{q}}(\boldsymbol{x}) + \boldsymbol{q}'(\boldsymbol{x},t)$

$$\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}' = \boldsymbol{f}$$

Fourier Transform

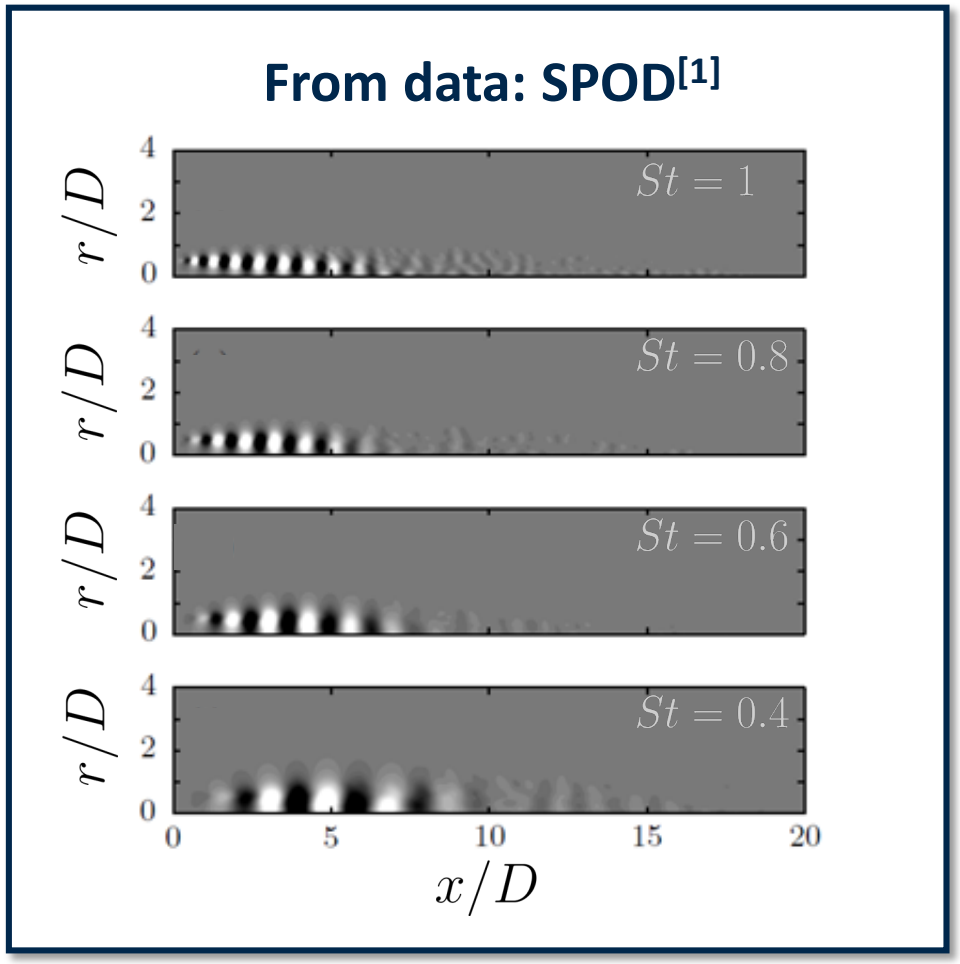
$$\hat{\boldsymbol{q}} = (i\omega I - A)^{-1}\,\hat{\boldsymbol{f}}$$

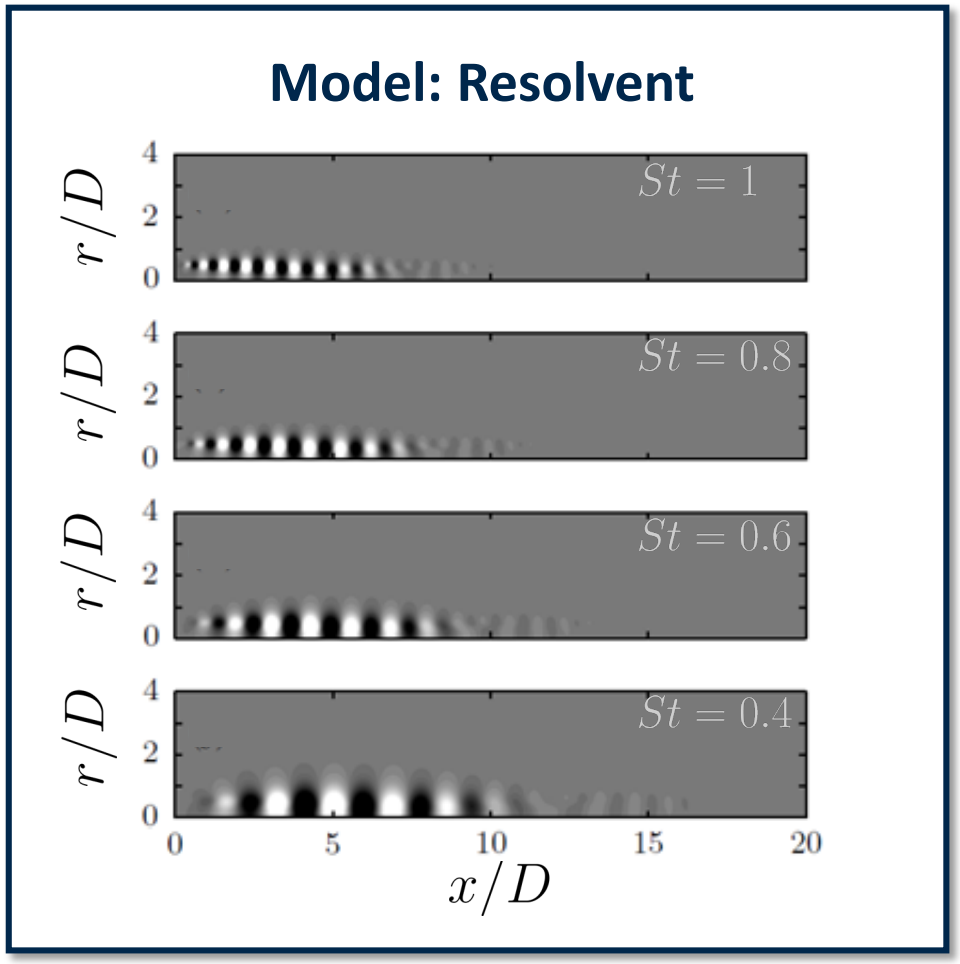
Singular value decomp.



Response ← Gain ← Forcing

Resolvent analysis models coherent structures



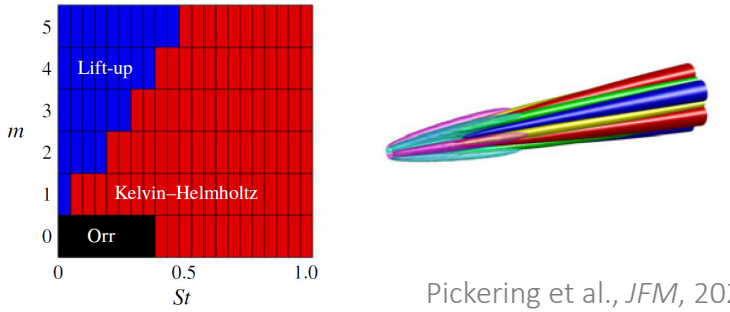


[1] Towne et al., *JFM*, 2018

Applications of resolvent analysis

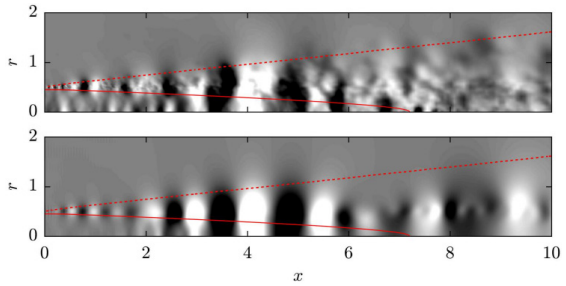
- Resolvent analysis can be used for …

Identifying physics



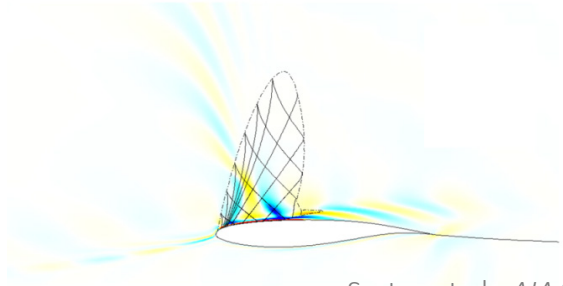
Pickering et al., *JFM*, 2020

Low-rank reconstruction



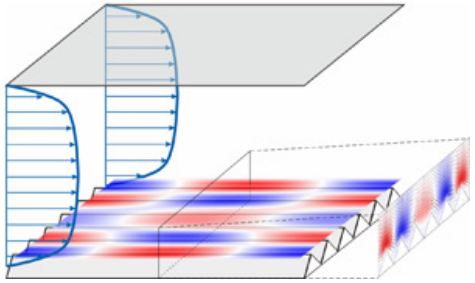
Schmidt et al., *AIAA*, 2017

Receptivity analysis



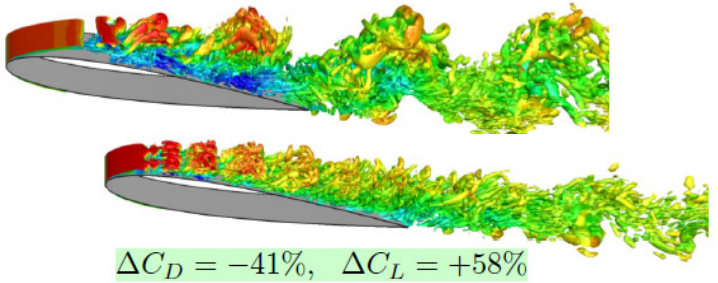
Sartor et al., *AIAAJ*, 2015

Design optimization



Ran et al., *JFM*, 2021

Open-loop control design



Yeh & Taira, *JFM*, 2019

… and more …

Limitations of resolvent analysis

Linear

$$\underbrace{\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}'}_{\text{Analysis focuses on linear part}} = \boldsymbol{f}$$

Think of nonlinear part as an external forcing

Limitations of resolvent analysis

Linear

$$\underbrace{\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}'}_{} = \boldsymbol{f}$$

Analysis focuses on linear part

Think of nonlinear part as an external forcing

Cost scales poorly

Computing resolvent modes becomes expensive for large systems



Still useful?

Limitations of resolvent analysis

Linear

$$\underbrace{\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}'}_{\text{Analysis focuses on linear part}} = \boldsymbol{f}$$

Think of nonlinear part as an external forcing

Cost scales poorly

Computing resolvent modes becomes expensive for large systems

Still useful?

Cost scaling: $O(N^p)$

N: Number of degrees of freedom

p: Scaling coefficient

Limitations of resolvent analysis

Linear

$$\underbrace{\frac{\partial \boldsymbol{q}'}{\partial t} - A\,\boldsymbol{q}'}_{\text{Analysis focuses on linear part}} = \boldsymbol{f}$$

Think of nonlinear part as an external forcing

Cost scales poorly

Computing resolvent modes becomes expensive for large systems

↓

Still useful?

Usually qualitative

Resolvent analysis

↓

Insight

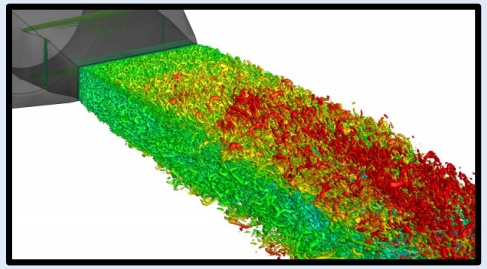
↓

Application

Outline

Efficient Computation

Motivation



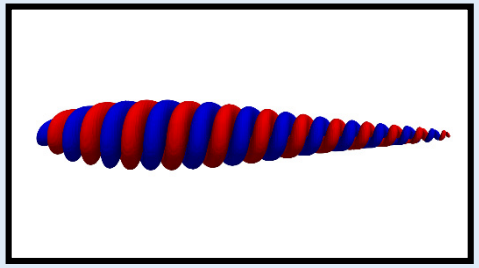
Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$



$$\hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

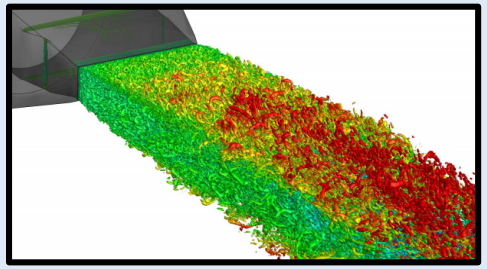
Demonstration



Outline

Efficient Computation

Motivation



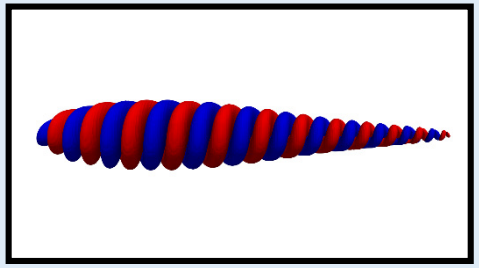
Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$



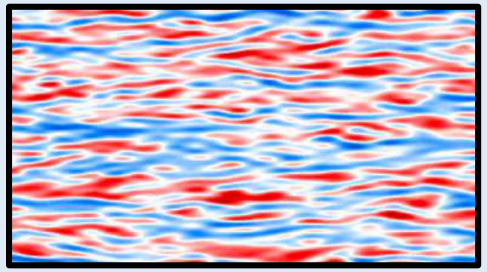
$$\hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

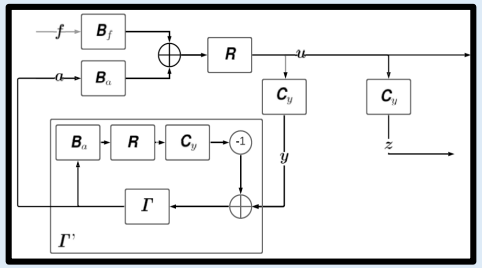


Estimation & Control

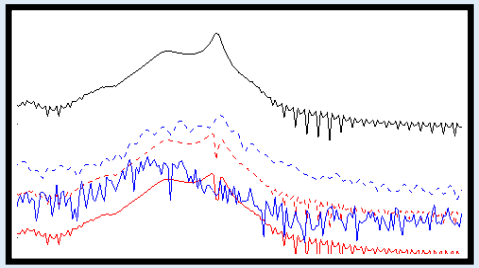
Flow estimation



Flow control



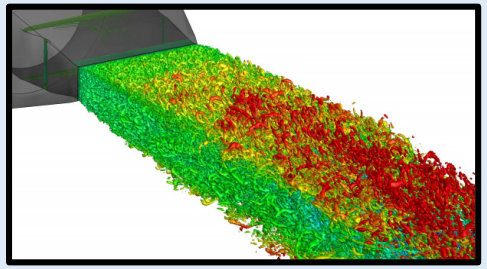
Features



Outline

Efficient Computation

Motivation

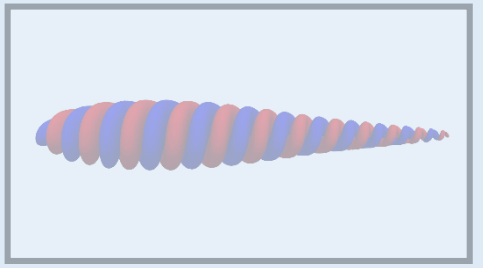


Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$

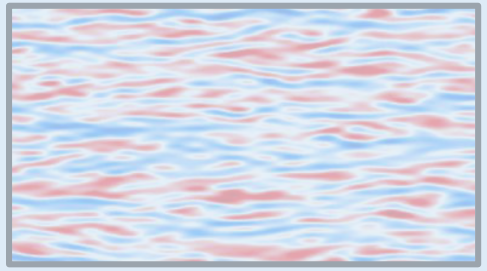
$$\rightarrow \quad \rightarrow \quad \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

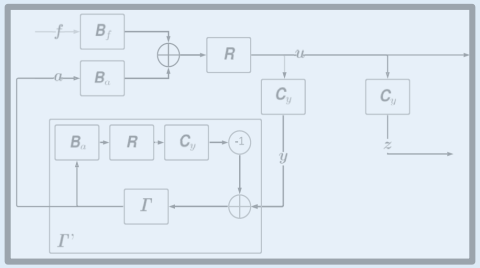


Estimation & Control

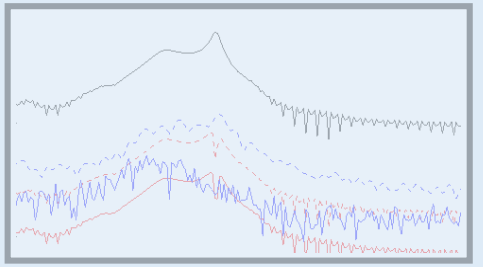
Flow estimation



Flow control



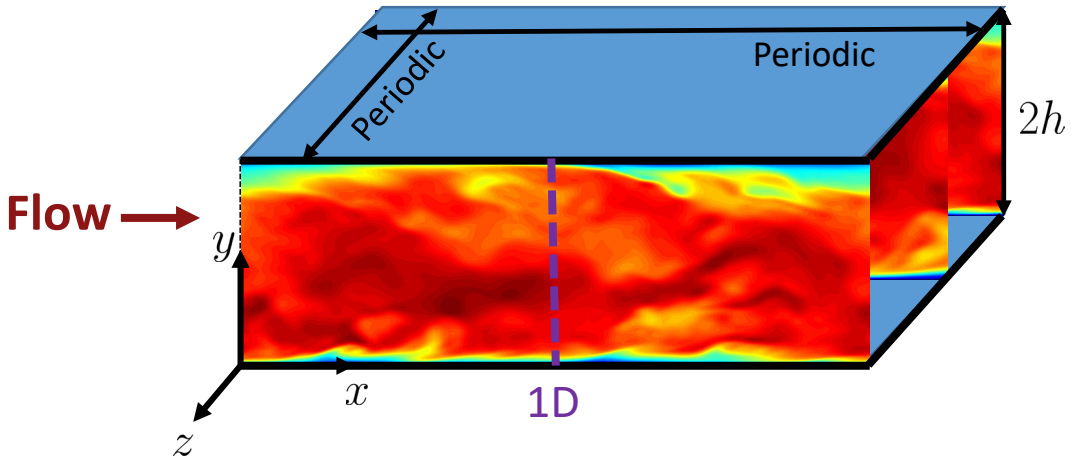
Features



Resolvent analysis cost

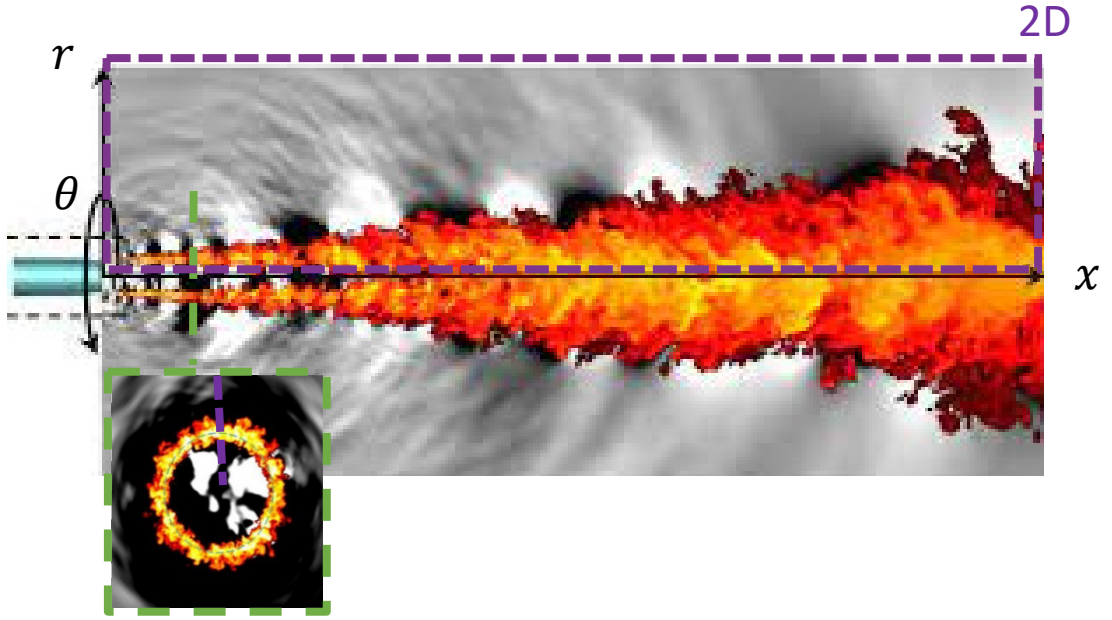
- Computational cost considerations have limited resolvent analysis to problems with symmetries that can be leveraged to reduce the problem dimension

Channel



3D → 1D

Round Jet

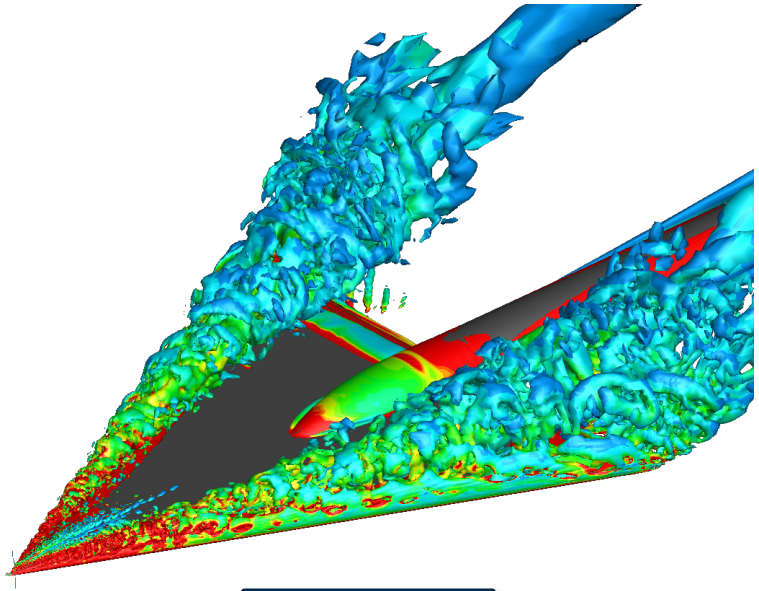


3D → 2D

Resolvent analysis cost

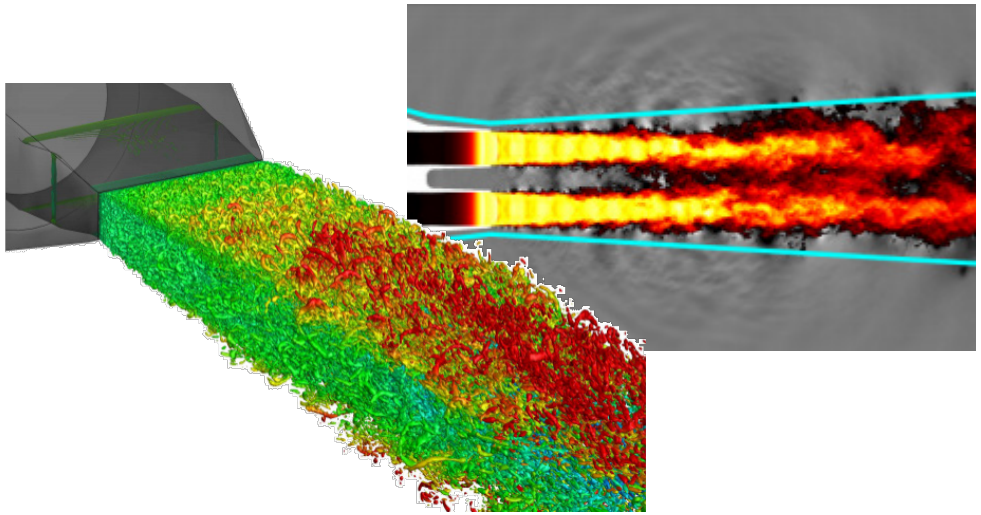
- Computational cost considerations have limited resolvent analysis to problems with symmetries that can be leveraged to reduce the problem dimension

3D Boundary Layers



3D

Complex Jets



3D

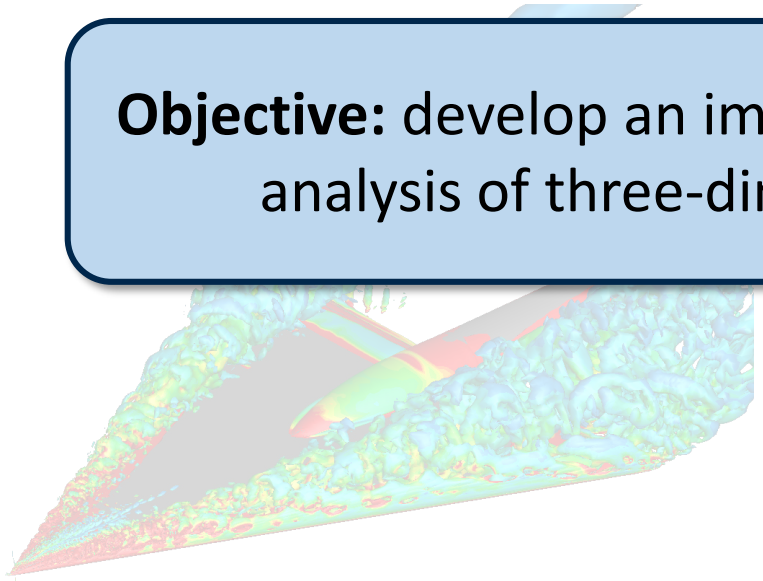
Resolvent analysis cost

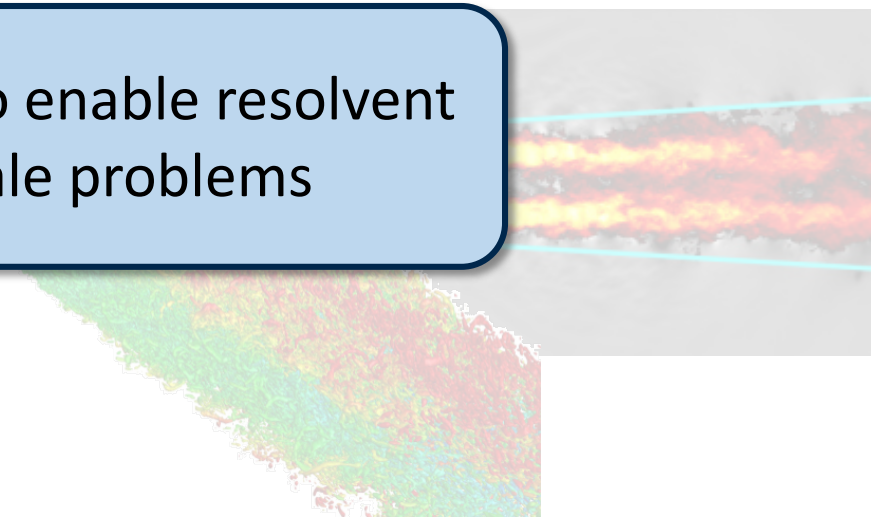
- Computational cost considerations have limited resolvent analysis to problems with symmetries that can be leveraged to reduce the problem dimension

3D Boundary Layer

Complex Jets

Objective: develop an improved algorithm to enable resolvent analysis of three-dimensional, large-scale problems





3D

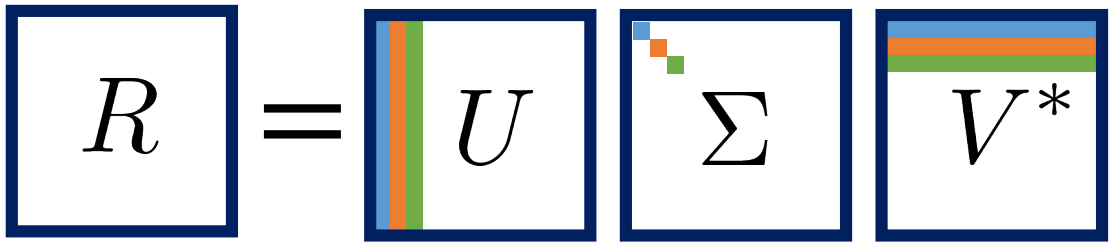
3D

Two potentially costly steps in resolvent analysis

1. Applying the action of the resolvent operator to a vector

$$\hat{\boldsymbol{q}} = \underbrace{(i\omega I - A)^{-1}}_{R}\hat{\boldsymbol{f}}$$

2. Computing the SVD



Computing resolvent modes

N = problem size

Naive approach

$O(N^3)$

1. Compute the inverse of $(i\omega I - A)$
2. Compute the full SVD

Standard approach[1,2]

$O(N^{1.5}) - O(N^2)$

1. Compute the LU of $(i\omega I - A)$
2. Compute the leading modes via Arnoldi

Optimal OWNS[3]

Improvement in x

1. Apply the resolvent operator via spatial marching
2. Adjoint upstream marching to find optimal

Randomized SVD[4,5]

$O(N^{1.5}) - O(N^2)$

1. Compute the LU of $(i\omega I - A)$
2. Compute the leading modes via RSVD

Time-stepping[6,7,8]

$O(N^?)$

1. Apply the resolvent operator via temporal marching
2. Compute the leading modes via power iteration or Arnoldi

[1] Jeun et al. *POF* 2016

[2] Schmidt et al. *JFM* 2018

[3] Towne et al. *JFM* 2022

[4] Moarref et al. 2013

[5] Ribeiro et al. *RRF* 2020

[6] Monokrousos et al. *JFM* 2010

[7] Martini et al. *JFM* 2021

[8] Farghadan et al. AIAA 2021

Computing resolvent modes

N = problem size

Naive approach

$O(N^3)$

1. Compute the inverse of $(i\omega I - A)$
2. Compute the full SVD

Randomized SVD[5]

$O(N^{1.5}) - O(N^2)$

1. Compute the LU of $(i\omega I - A)$
2. Compute the leading modes via RSVD

Standard approa

1. Compute the LU
2. Compute the lead

Optimal OWNS[3,4]

Improvement in x

1. Apply the resolvent operator via spatial marching
2. Adjoint upstream marching to find optimal

$O(N^?)$

via temporal

2. Compute the leading modes via power iteration or Arnoldi

Our approach: RSVD + careful time stepping

[1] Jeun et al. *POF* 2016

[2] Schmidt et al. *JFM* 2018

[3] Towne et al. *JFM* 2021

[4] Moerref et al. 2013

[5] Ribeiro et al. *RRF* 2020

[6] Monokrousos et al. *JFM* 2010

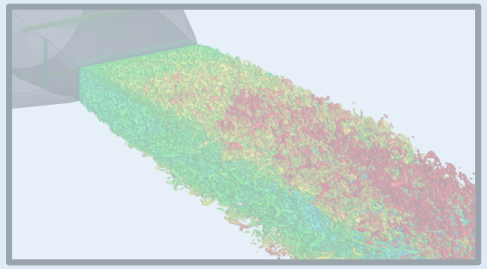
[7] Martini et al. *JFM* 2021

[8] Farghadan et al. AIAA 2021

Outline

Efficient Computation

Motivation



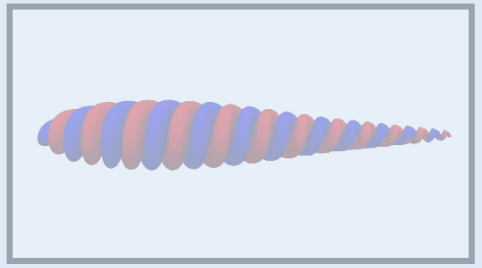
Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$



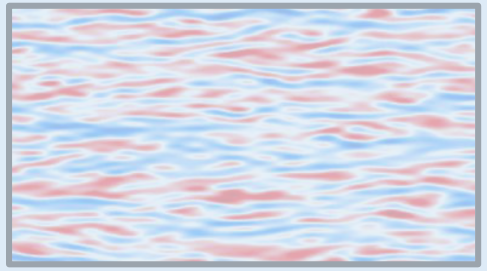
$$\hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

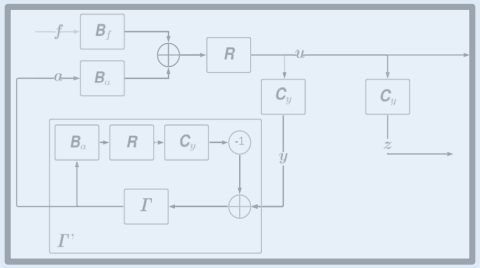


Estimation & Control

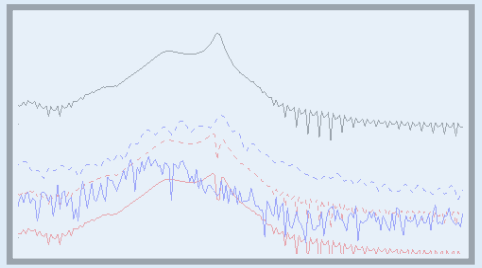
Flow estimation



Flow control



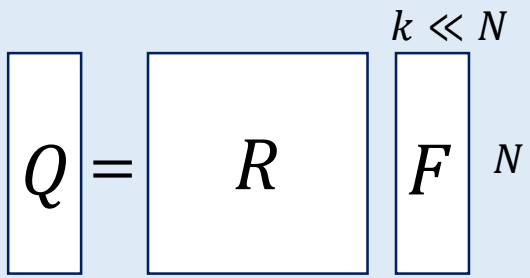
Features



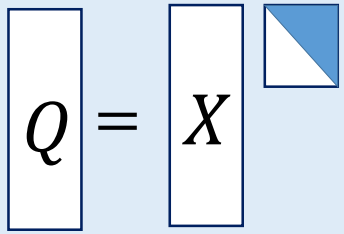
Randomized SVD

- **Idea:** sample the image and range of the resolvent operator → compute SVD of the reduced matrix

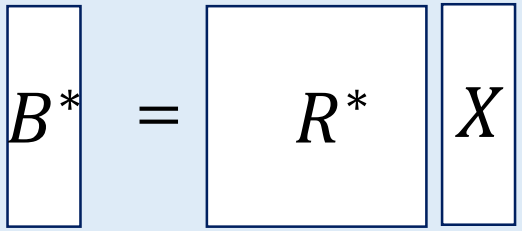
1. Sample range



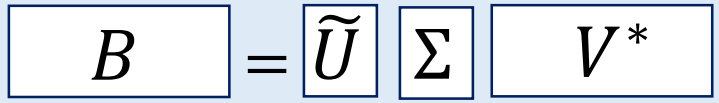
2. Orthogonalize



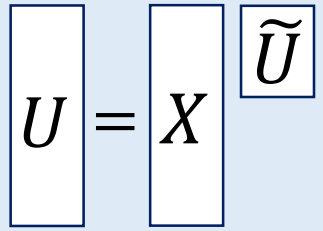
3. Sample image



4. SVD



5. Reconstruct U

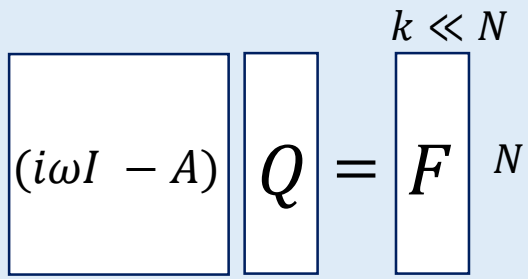


$$R \approx U \Sigma V^*$$

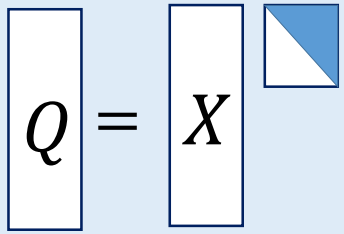
Randomized SVD

- **Idea:** sample the image and range of the resolvent operator → compute SVD of the reduced matrix

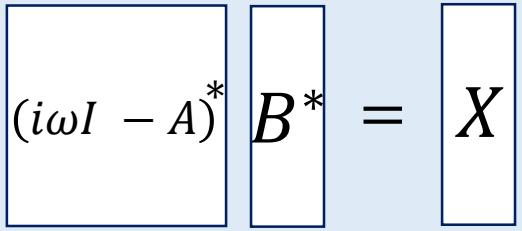
1. Sample range



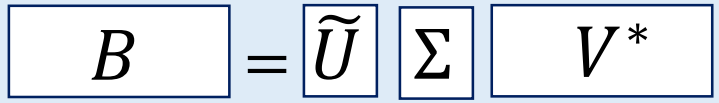
2. Orthogonalize



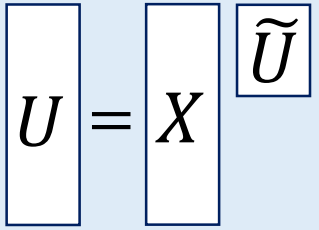
3. Sample image



4. SVD



5. Reconstruct U



$$R \approx U\Sigma V^*$$

[1] Ribeiro *et al.*, *PRF*, 2020

Randomized SVD

- **Idea:** sample the image and range of the resolvent operator → compute SVD of the reduced matrix

1. Sample range

Linear system

$$\frac{dq}{dt} - Aq = F(t)$$

2. Orthogonalize

$$Q = X$$



3. Sample image

Adjoint system

$$-\frac{db}{dt} - A^*b = q(t)$$

4. SVD

$$B = \tilde{U}\,\Sigma\,V^*$$

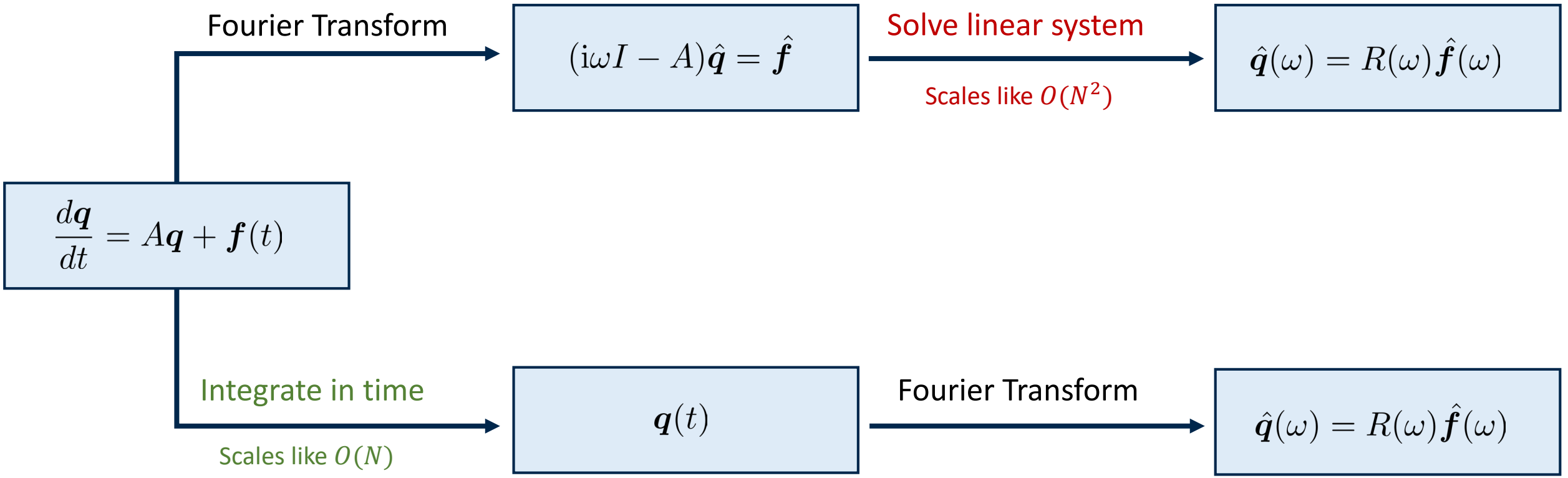
5. Reconstruct U

$$U = X\,\tilde{U}$$

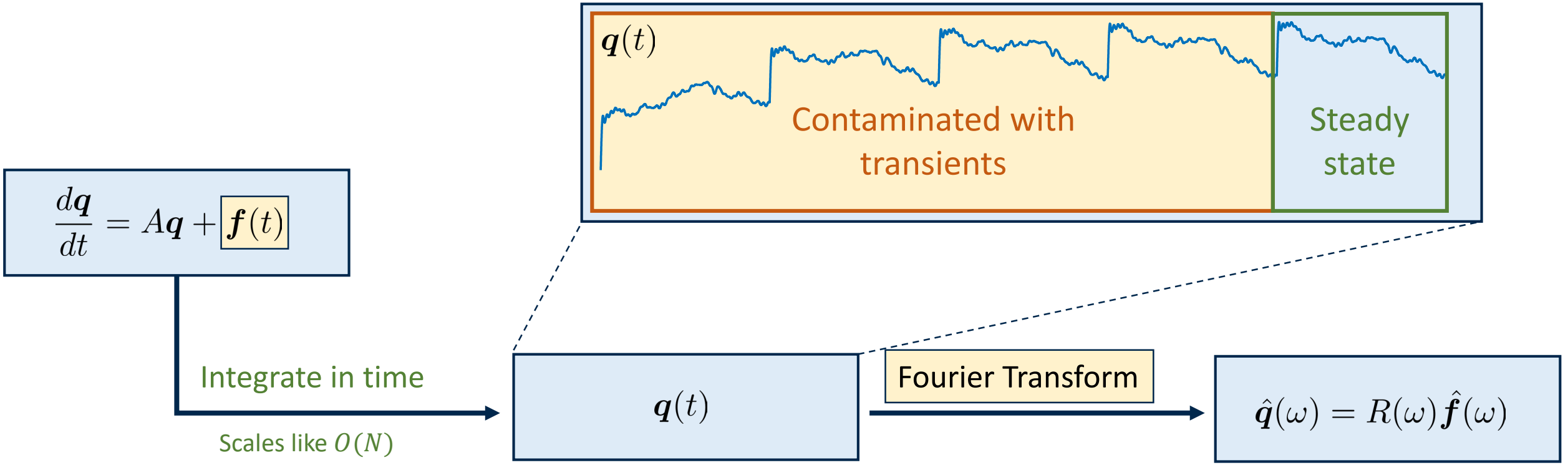
$$R \approx U\Sigma V^*$$

Action of the resolvent operator via time-stepping

- **Goal:** compute $\hat{\boldsymbol{q}}(\omega) = R(\omega)\hat{\boldsymbol{f}}(\omega)$



Action of the resolvent operator via time-stepping



How should we choose the forcing?

Single Frequency

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$
$$\boldsymbol{f}(t) = \hat{\boldsymbol{f}}_j e^{\mathrm{i}\omega_j t}$$

Monokrousos *et al. JFM* 2010

Integrate in time →

$$\boldsymbol{q}(t) = \hat{\boldsymbol{q}}_j e^{\mathrm{i}\omega_j t}$$

Fourier Transform →

$$\hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Sum of frequencies

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$
$$\boldsymbol{f}(t) = \sum_j \hat{\boldsymbol{f}}_j e^{\mathrm{i}\omega_j t}$$

Martini et al. *JFM* 2021

Integrate in time →

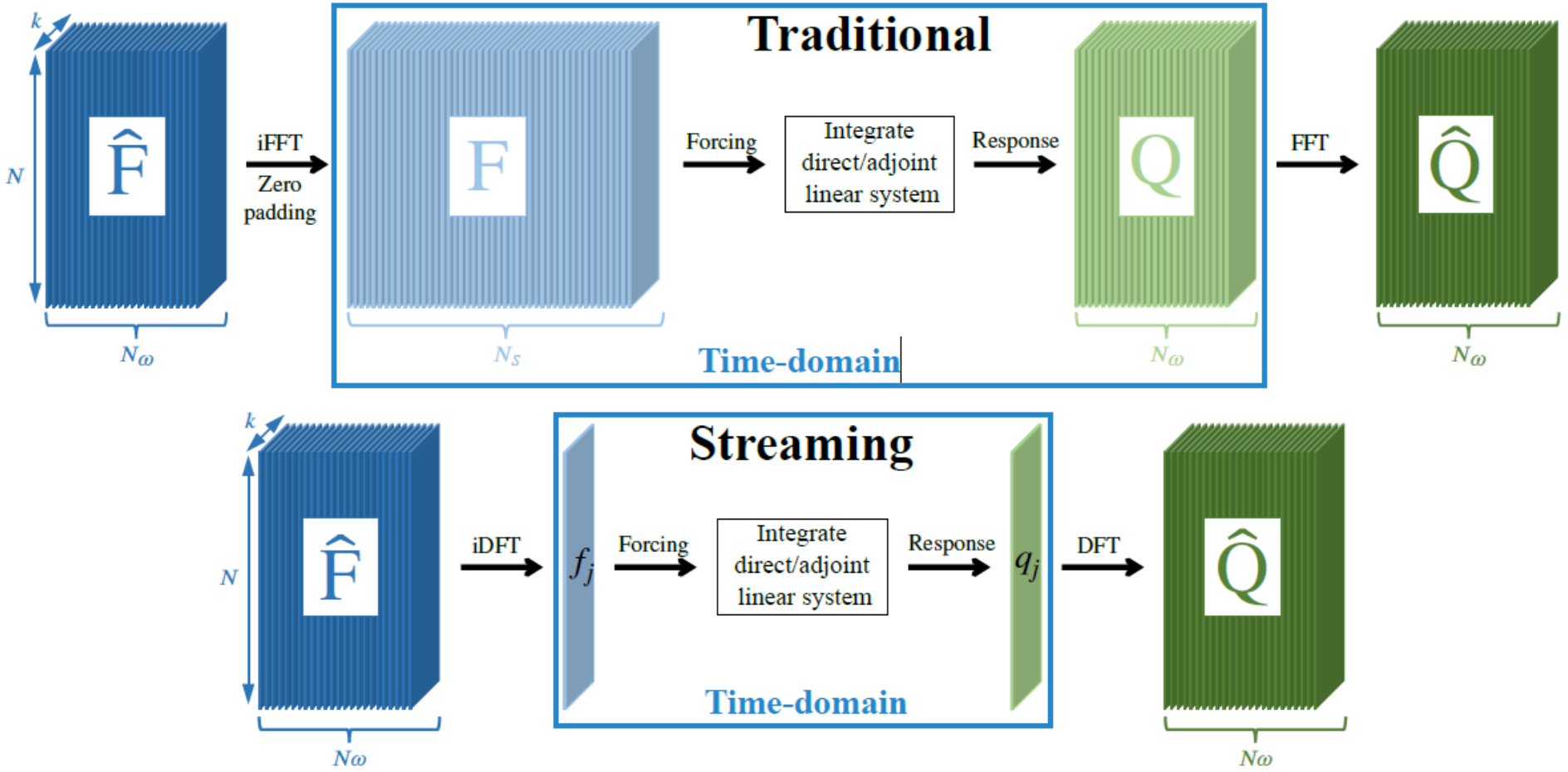
$$\boldsymbol{q}(t) = \sum_j \hat{\boldsymbol{q}}_j e^{\mathrm{i}\omega_j t}$$

Fourier Transform →

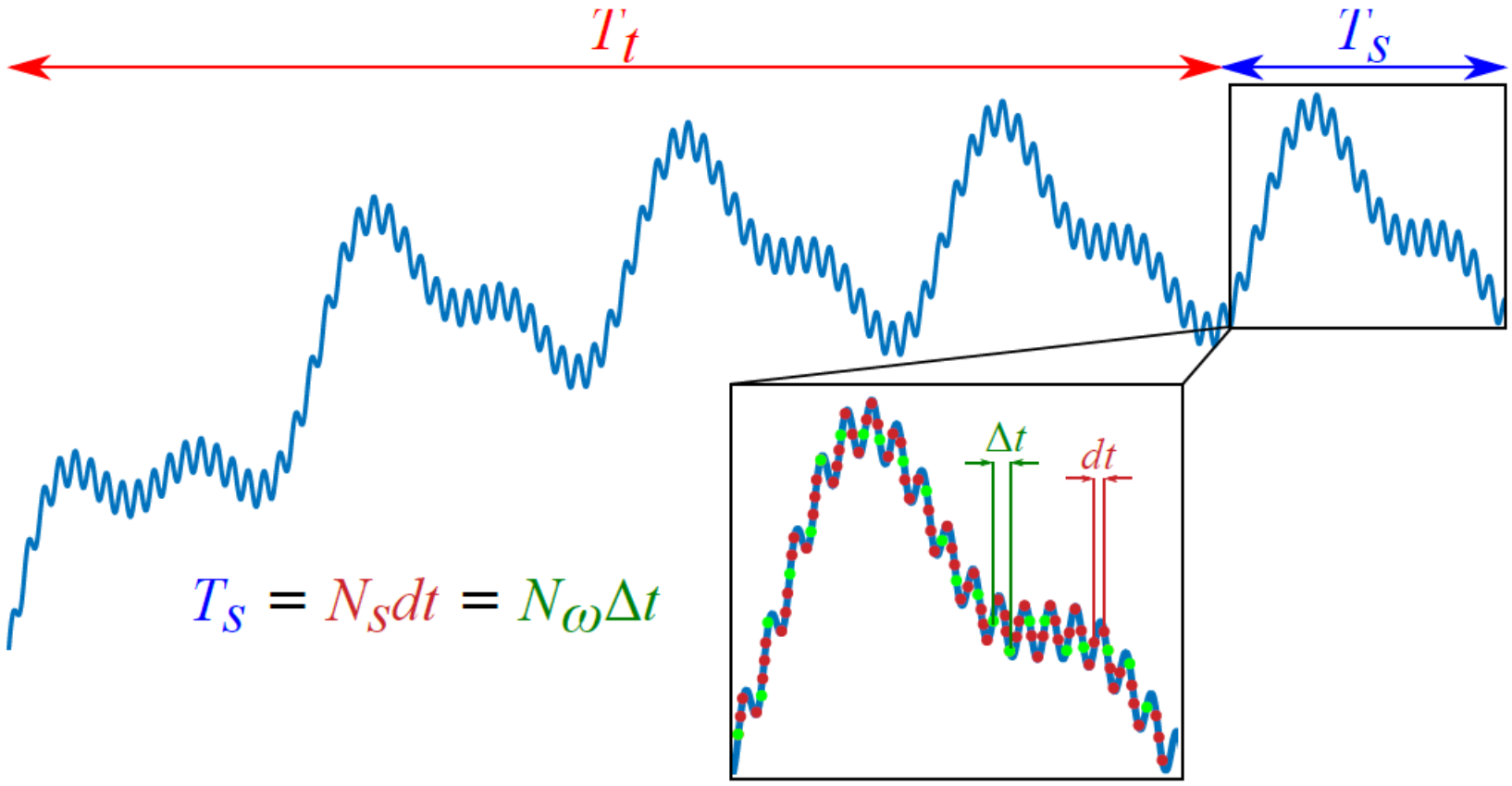
$$\hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j \quad \forall j$$

We get the action of the resolvent operator for all frequencies at once

Memory-efficient Fourier transforms

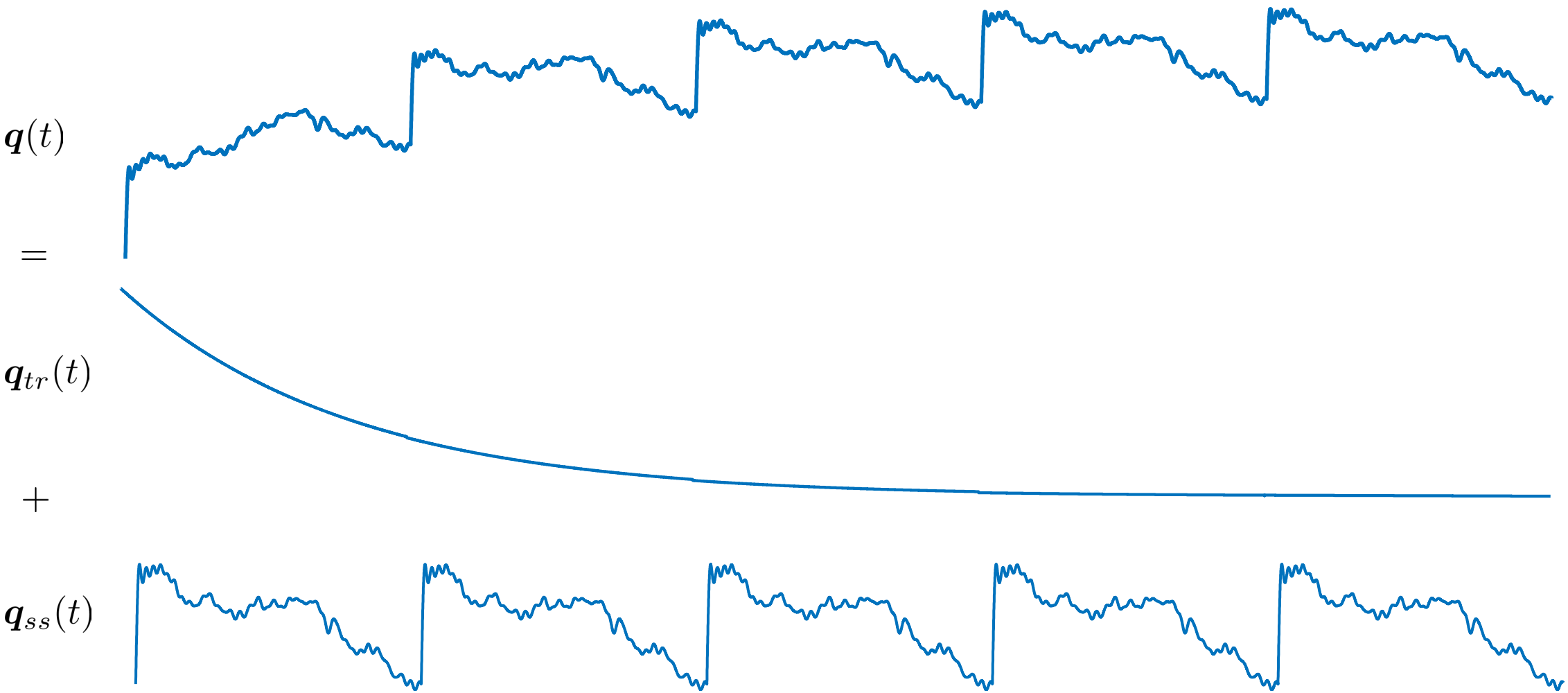


Memory-efficient Fourier transforms

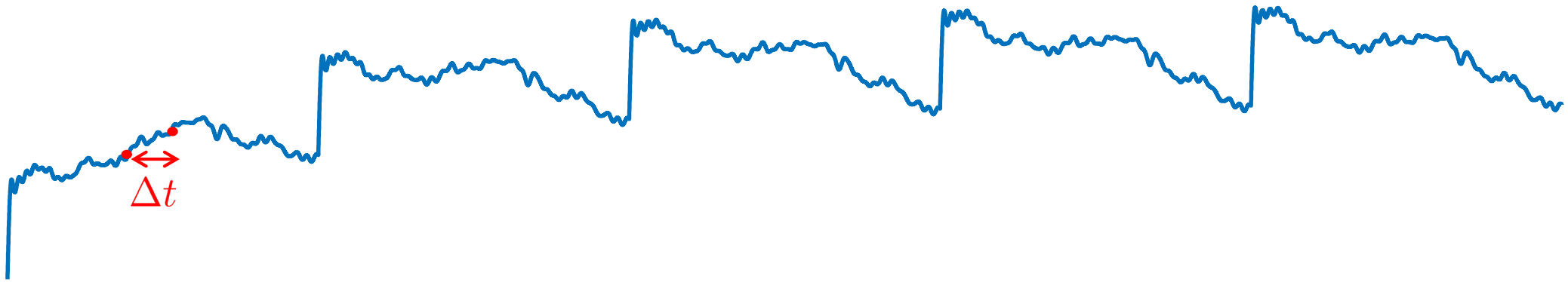


Since we control the frequency content via the forcing, we can aggressively downsample without aliasing

Eliminating the transients



Eliminating the transients



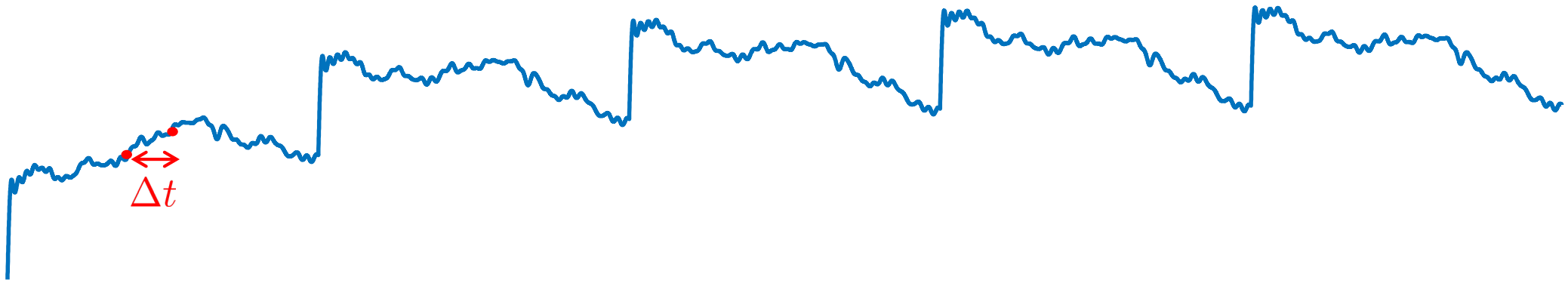
$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

Known

What we want

Eliminating the transients



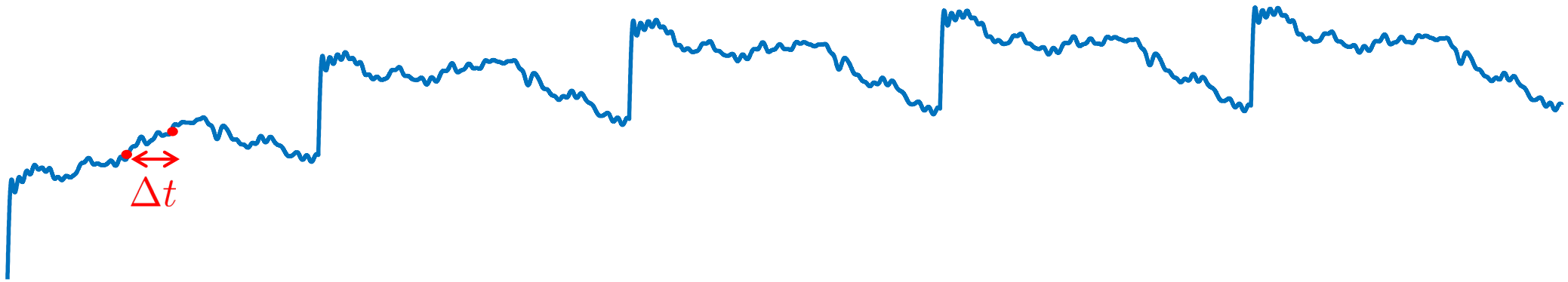
$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

$$\left.\begin{aligned} \boldsymbol{q}_{ss}(t+\Delta t) &= e^{\mathrm{i}\omega\Delta t}\boldsymbol{q}_{ss}(t) \\ \boldsymbol{q}_{tr}(t+\Delta t) &= e^{A\Delta t}\boldsymbol{q}_{tr}(t) \end{aligned}\right\}$$

Steady state and transient parts evolve differently

Eliminating the transients



$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

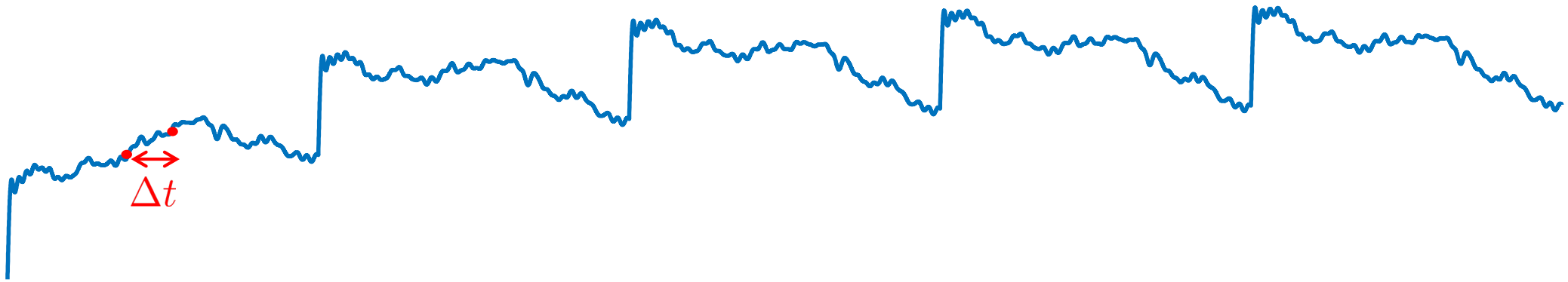
$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

$$\boldsymbol{q}_{ss}(t+\Delta t) = e^{\mathrm{i}\omega\Delta t}\boldsymbol{q}_{ss}(t)$$

$$\boldsymbol{q}_{tr}(t+\Delta t) = e^{A\Delta t}\boldsymbol{q}_{tr}(t)$$

Four equations &
Four unknowns

Eliminating the transients



$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

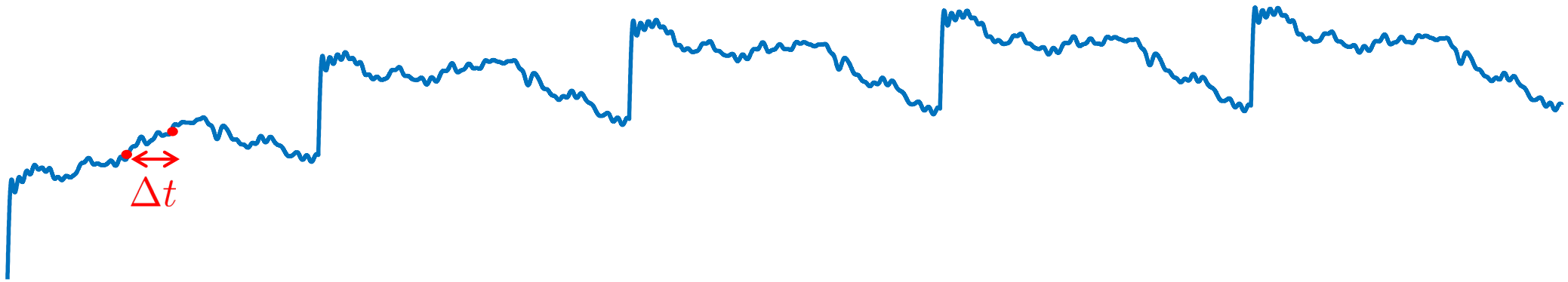
$$\boldsymbol{q}_{ss}(t+\Delta t) = e^{\mathrm{i}\omega\Delta t}\boldsymbol{q}_{ss}(t)$$

$$\boldsymbol{q}_{tr}(t+\Delta t) = e^{A\Delta t}\boldsymbol{q}_{tr}(t)$$

$$\left(I - e^{-\mathrm{i}\omega\Delta t}e^{A\Delta t}\right)\boldsymbol{q}_{tr}(t) = \boldsymbol{q}(t) - e^{-\mathrm{i}\omega\Delta t}\boldsymbol{q}(t+\Delta t)$$

Solve, then $\boldsymbol{q}_{ss}(t) = \boldsymbol{q}(t) - \boldsymbol{q}_{tr}(t)$

Eliminating the transients



$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

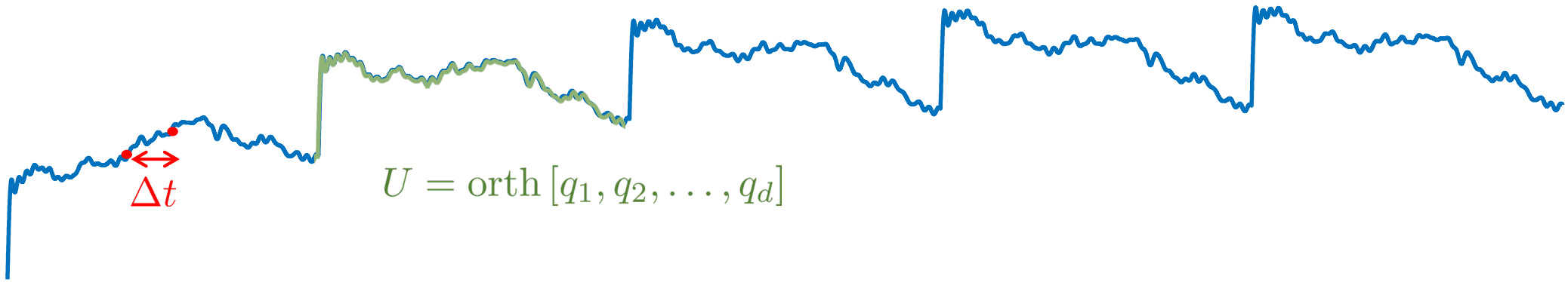
$$\boldsymbol{q}_{ss}(t+\Delta t) = e^{\mathrm{i}\omega\Delta t}\boldsymbol{q}_{ss}(t)$$

$$\boldsymbol{q}_{tr}(t+\Delta t) = e^{A\Delta t}\boldsymbol{q}_{tr}(t)$$

$$\left(I - e^{-\mathrm{i}\omega\Delta t}e^{A\Delta t}\right)\boldsymbol{q}_{tr}(t) = \boldsymbol{q}(t) - e^{-\mathrm{i}\omega\Delta t}\boldsymbol{q}(t+\Delta t)$$

Exact, but expensive to solve

Eliminating the transients



$$\boldsymbol{q}(t) = \boldsymbol{q}_{ss}(t) + \boldsymbol{q}_{tr}(t)$$

$$\boldsymbol{q}(t+\Delta t) = \boldsymbol{q}_{ss}(t+\Delta t) + \boldsymbol{q}_{tr}(t+\Delta t)$$

$$\boldsymbol{q}_{ss}(t+\Delta t) = e^{\mathrm{i}\omega\Delta t}\boldsymbol{q}_{ss}(t)$$

$$\boldsymbol{q}_{tr}(t+\Delta t) = e^{A\Delta t}\boldsymbol{q}_{tr}(t)$$

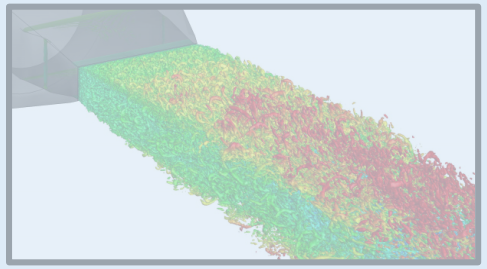
$$\left(I - e^{-\mathrm{i}\omega\Delta t}e^{A\Delta t}\right)\boldsymbol{q}_{tr}(t) = \boldsymbol{q}(t) - e^{-\mathrm{i}\omega\Delta t}\boldsymbol{q}(t+\Delta t)$$

Instead, obtain an approximation by projecting into a low-dimensional subspace U for which $\boldsymbol{q}_{tr} \approx UU^*\boldsymbol{q}_{tr}$

Outline

Efficient Computation

Motivation

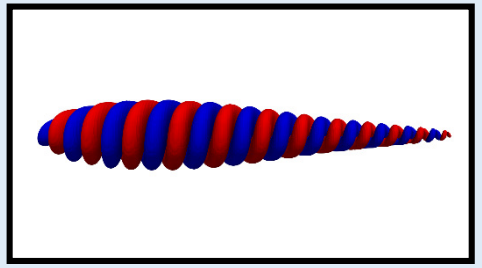


Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$

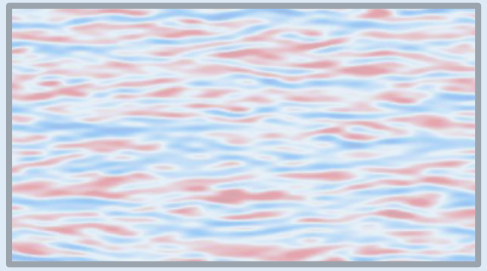
$$\rightarrow \quad \rightarrow \quad \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

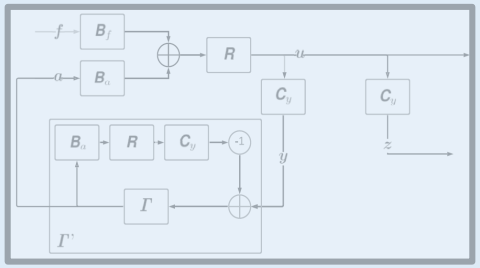


Estimation & Control

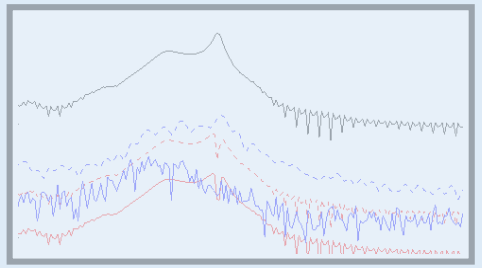
Flow estimation



Flow control

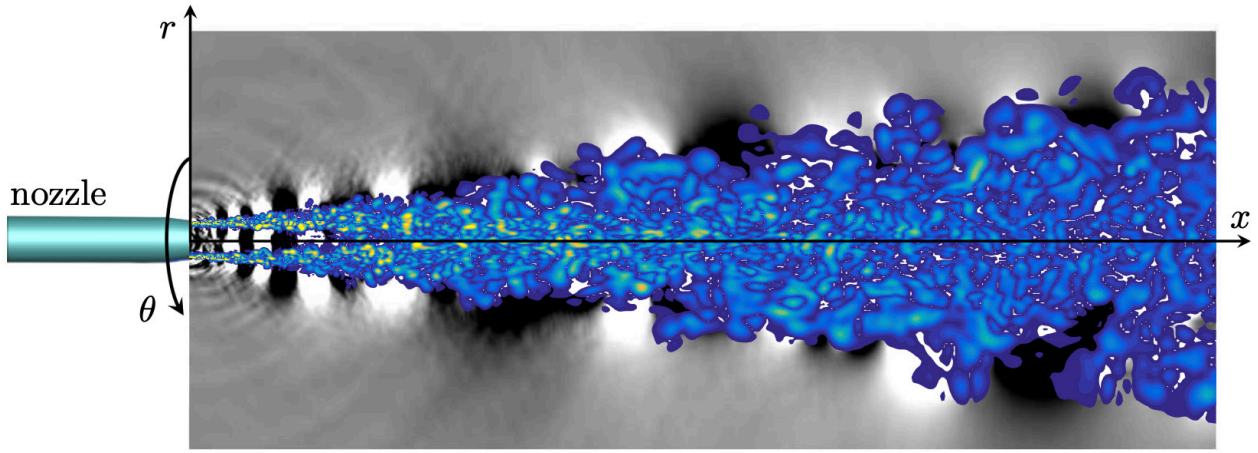


Features

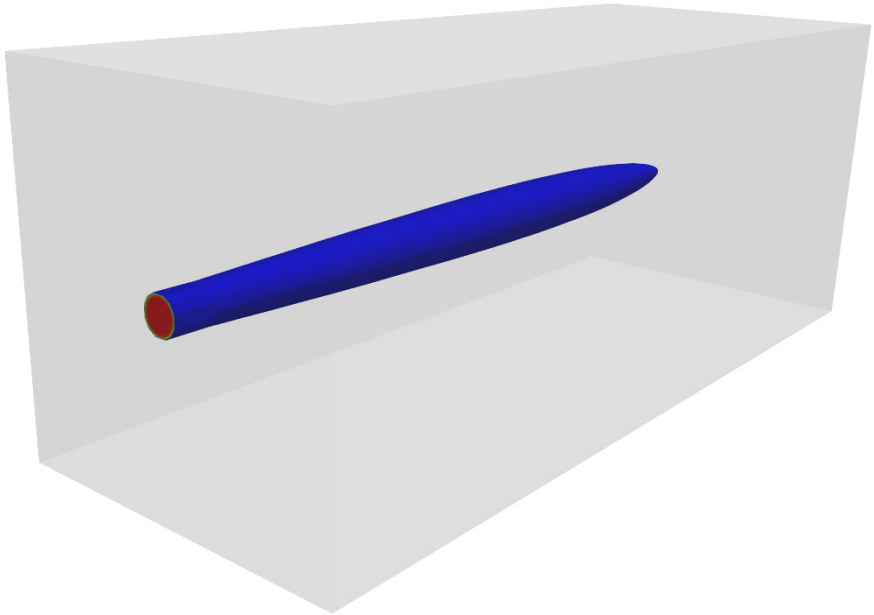


Example: nominally three-dimensional jet

Large eddy simulation[1]

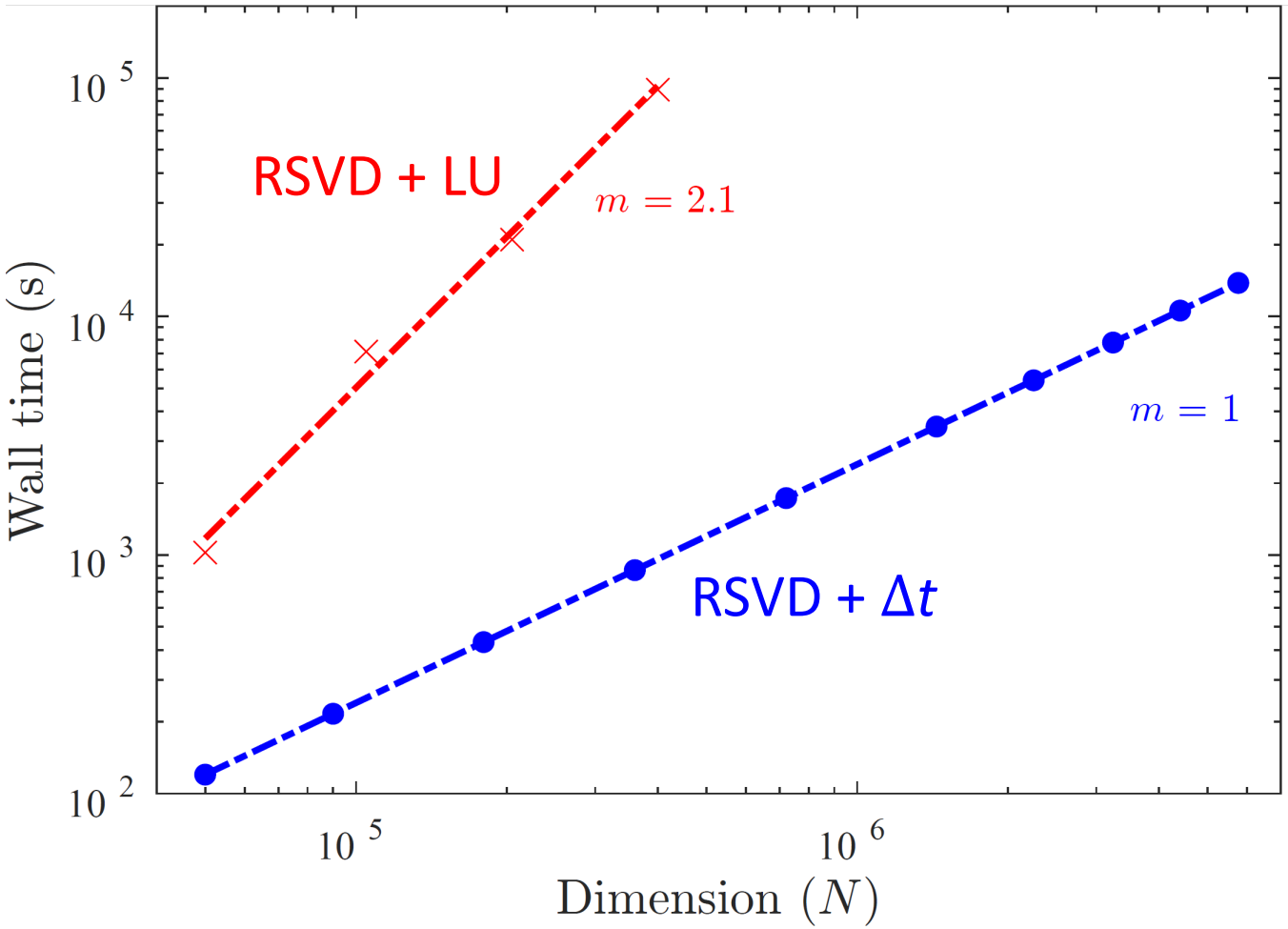


Three-dimensional mean



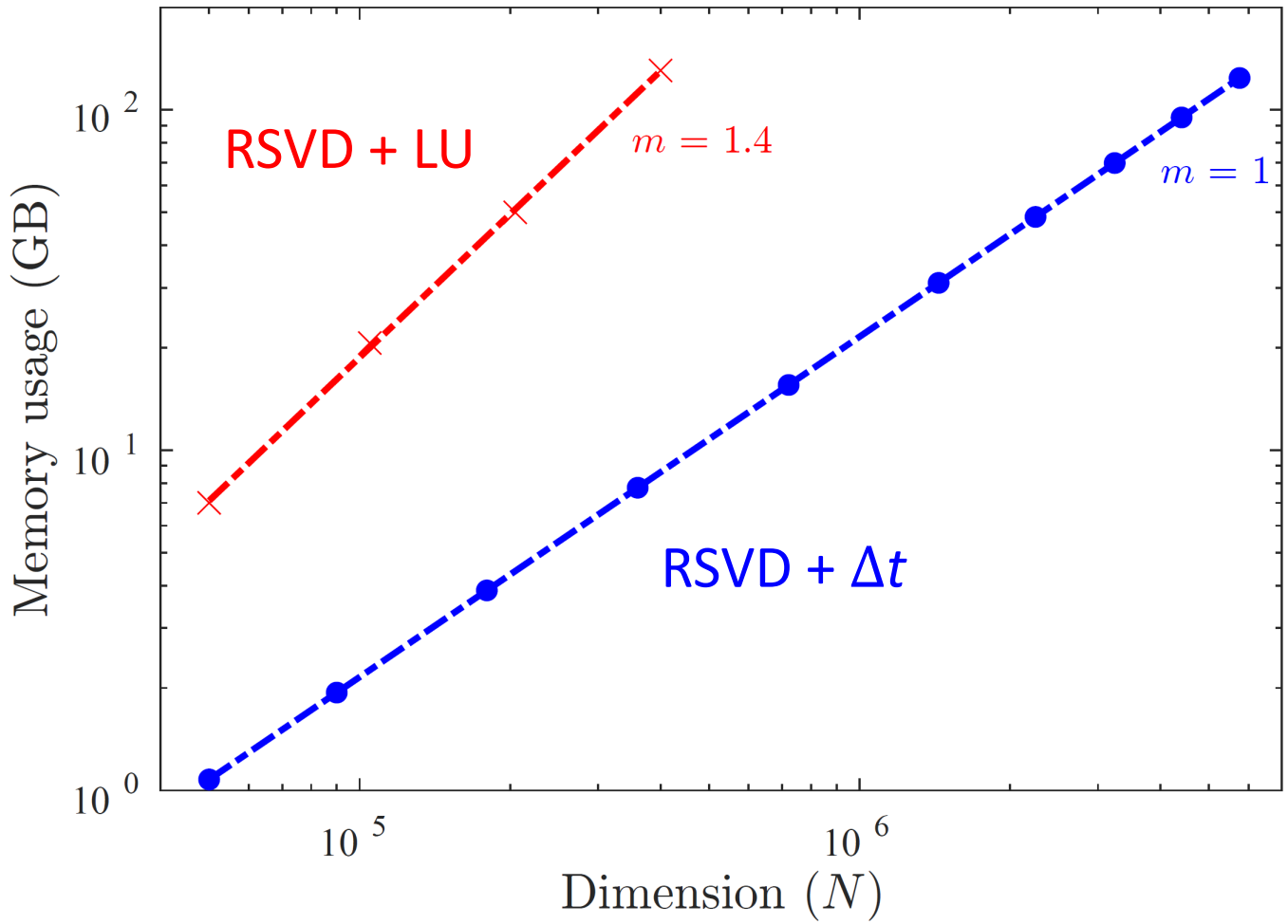
[1] Bres *et al.*, *JFM*, 2018

CPU time scaling



N = 40,000,000
14000x speedup

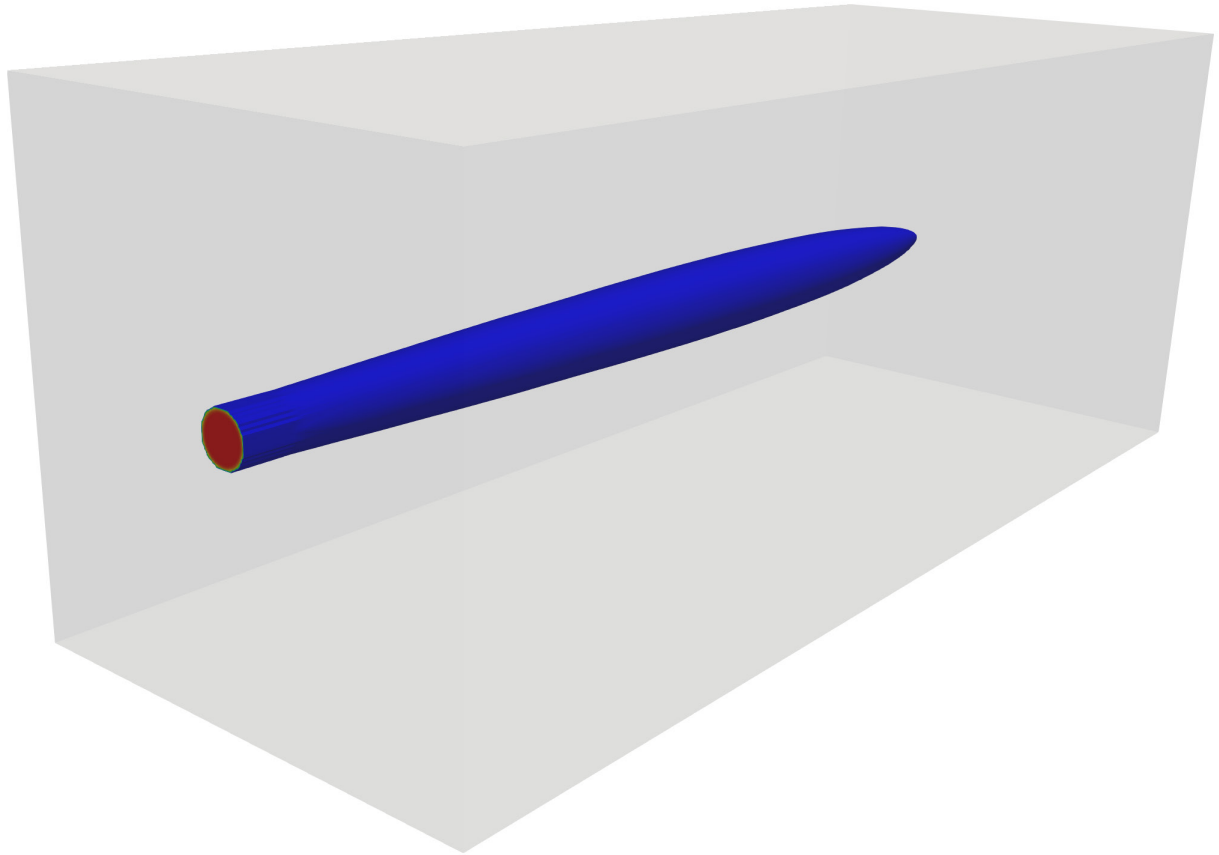
Peak memory usage scaling



N = 40,000,000
100x memory reduction

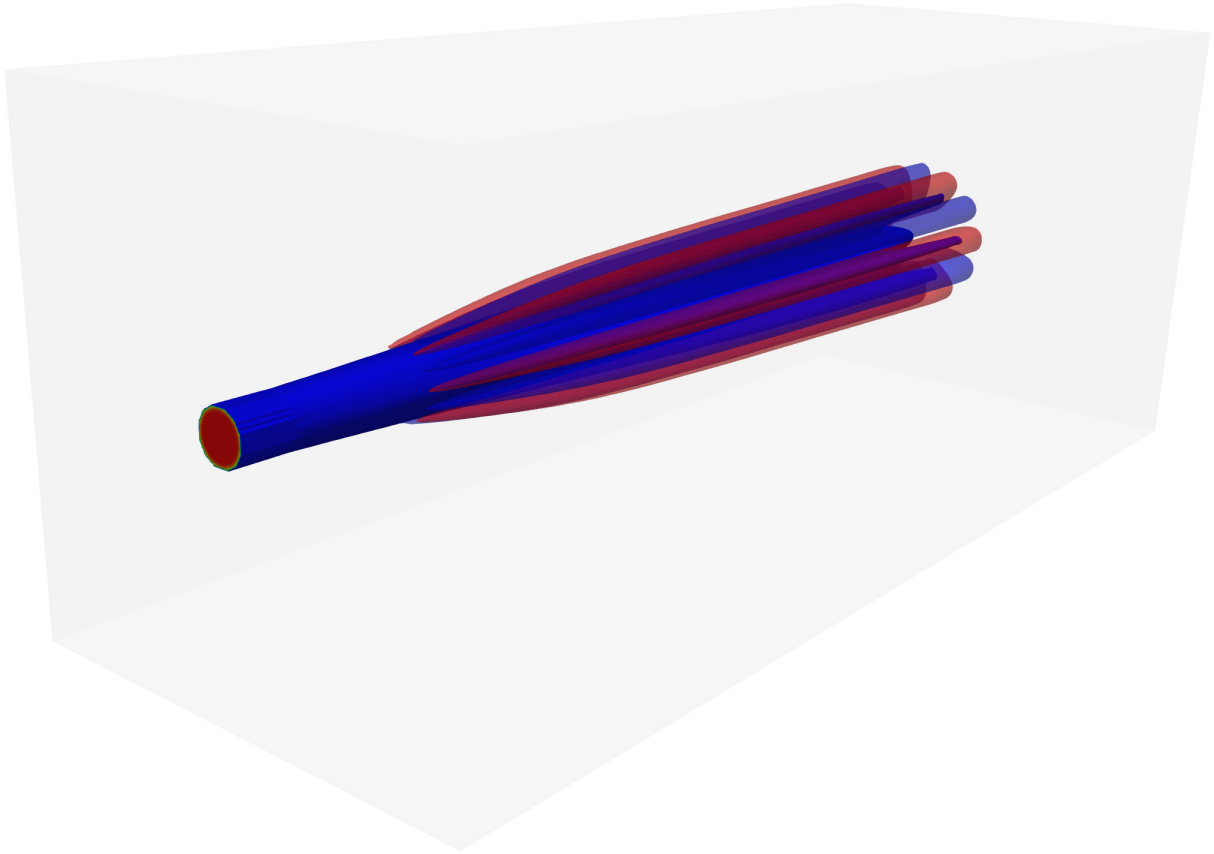
Maximum size for a desktop with 750 GB RAM:
RSVD + LU:
$N < 1{,}000{,}000$
RSVD + Δt:
$N < 40{,}000{,}000$

Example: jet with streaks



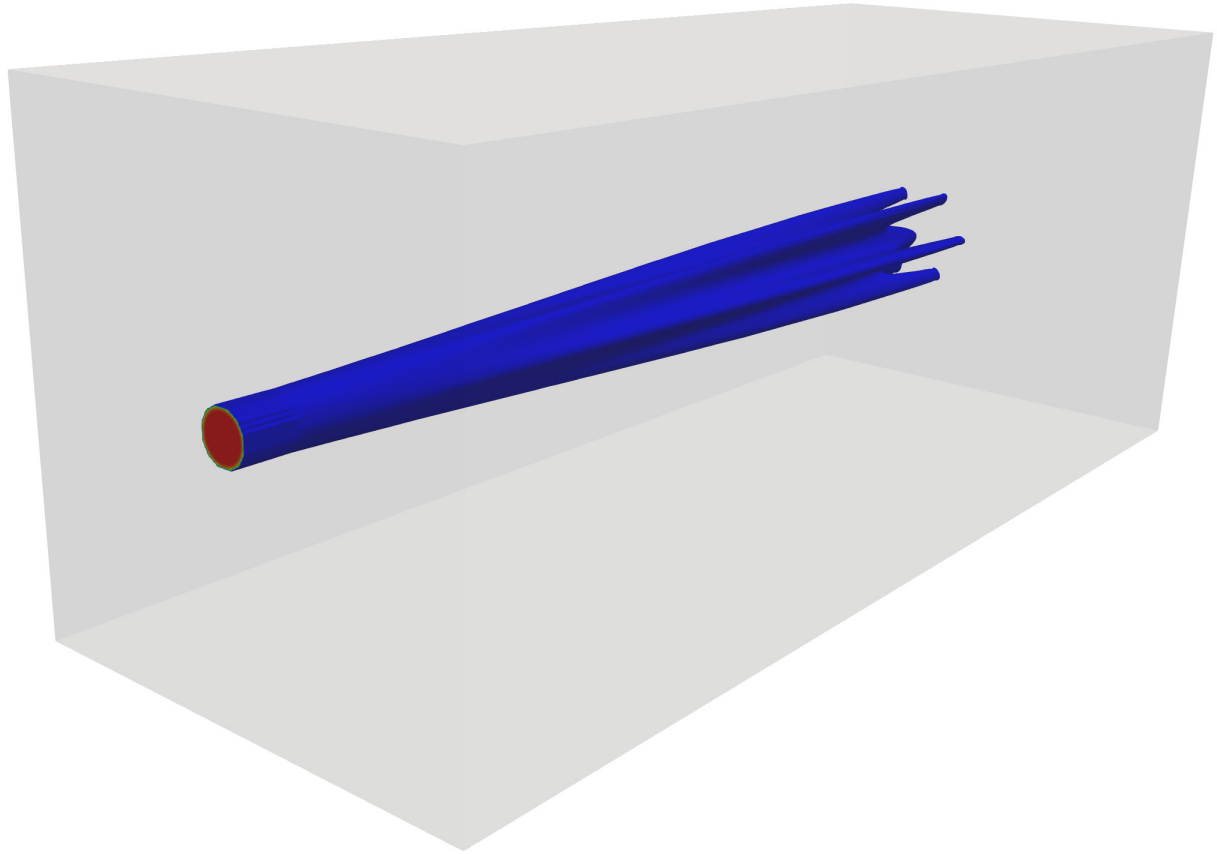
Example: jet with streaks

- Streaks with m=6 computed using axisymmetric code and added to mean flow



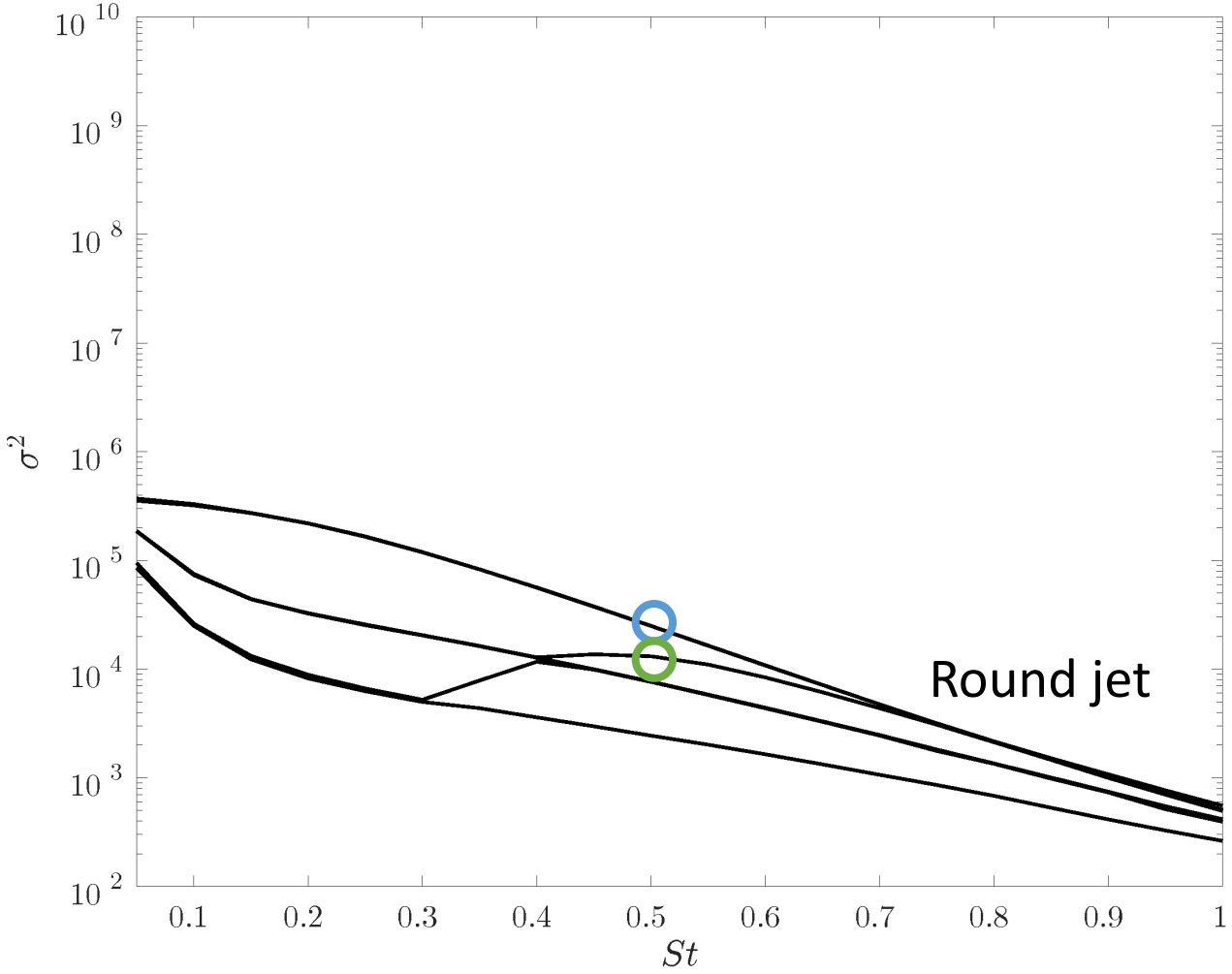
Example: jet with streaks

- Streaks with m=6 computed using axisymmetric code and added to mean flow

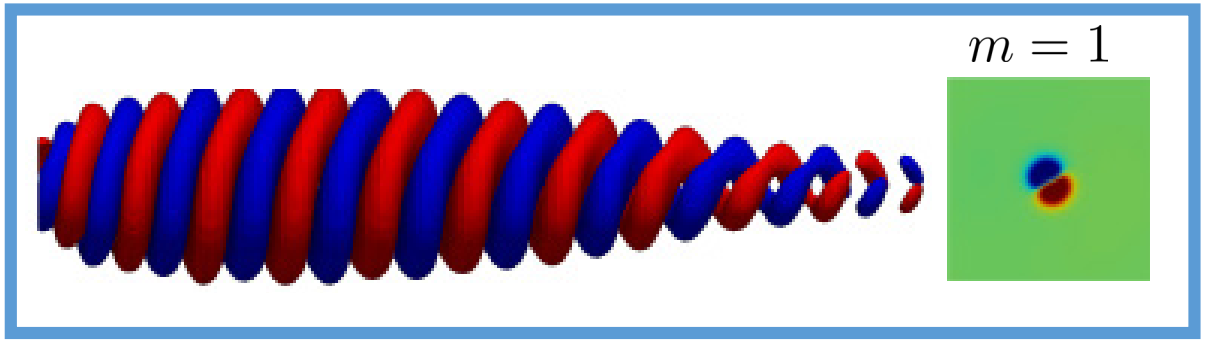


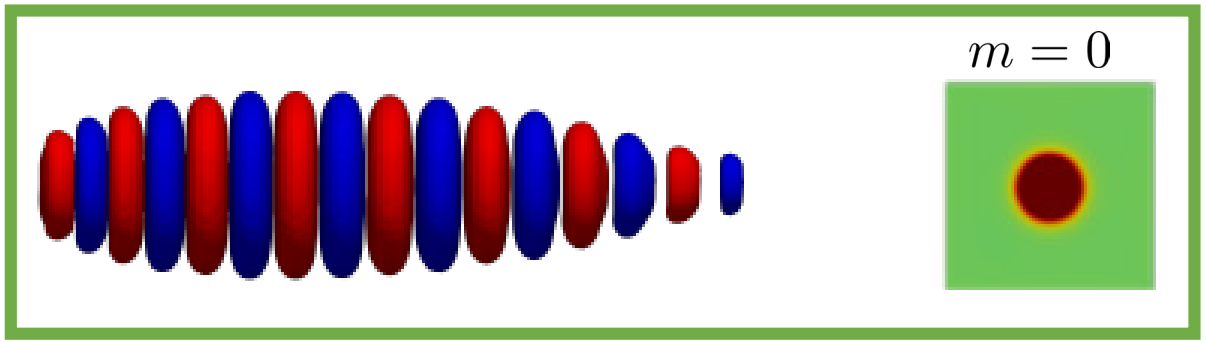
5% amplitude

Impact of streaks



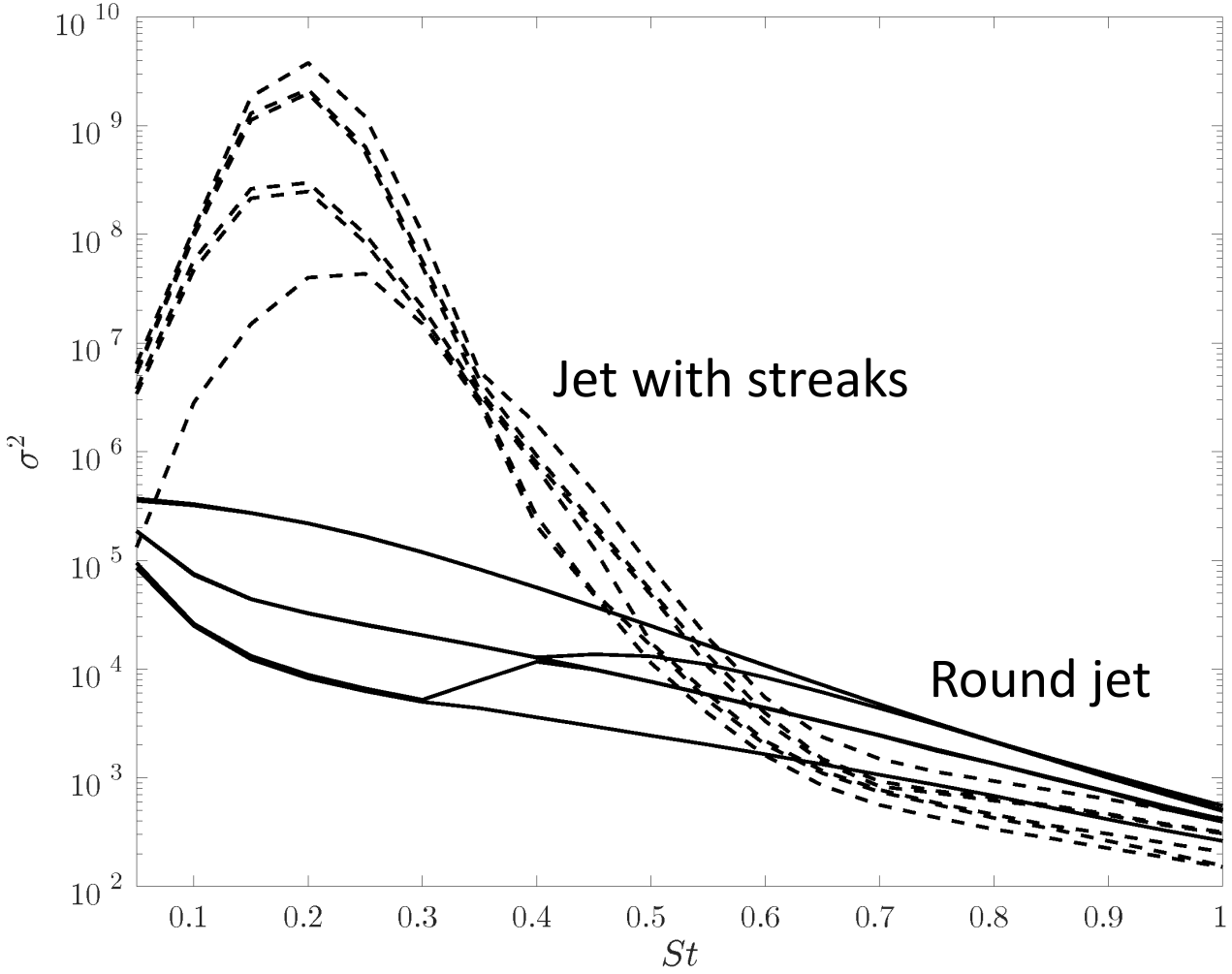
$$\boldsymbol{q} \sim e^{im\theta}$$



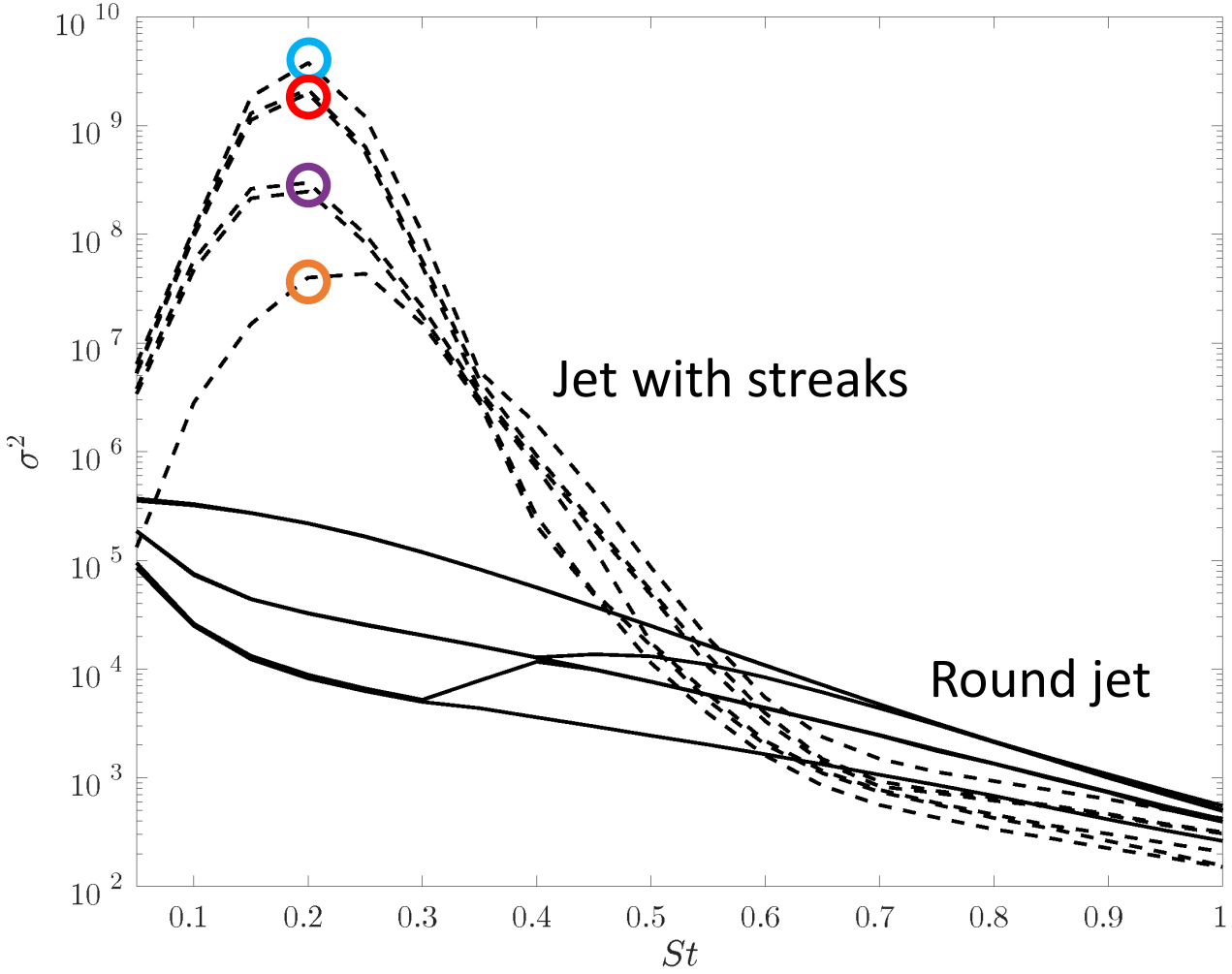


Impact of streaks

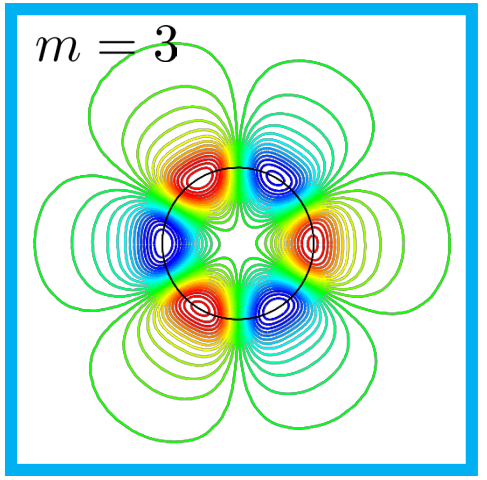
$$\boldsymbol{q} \sim e^{im\theta}$$

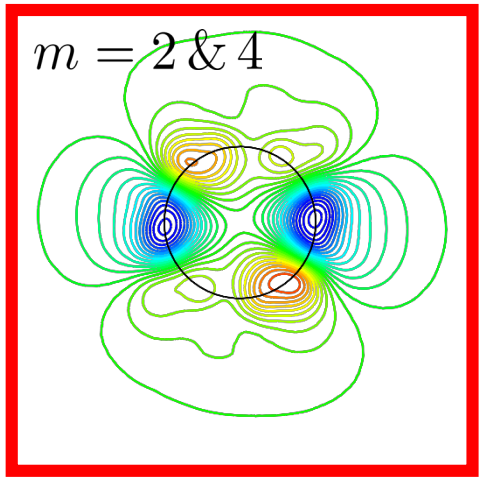


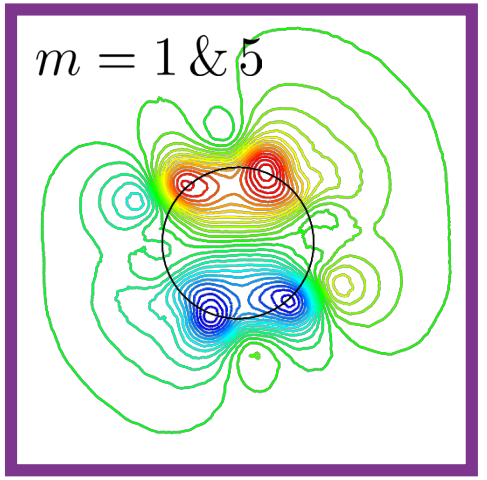
Impact of streaks

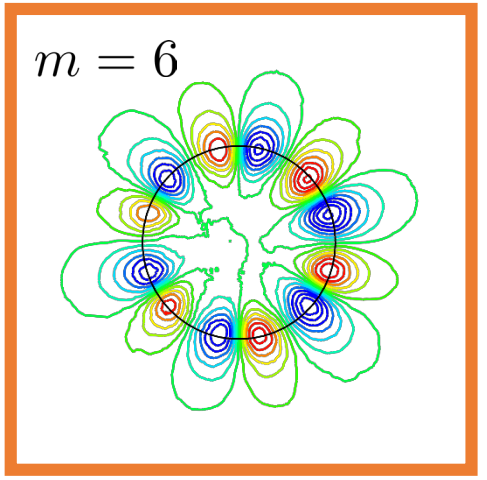


$$\boldsymbol{q} \sim e^{im\theta}$$









Cost comparison

$N = 39.2\,M$

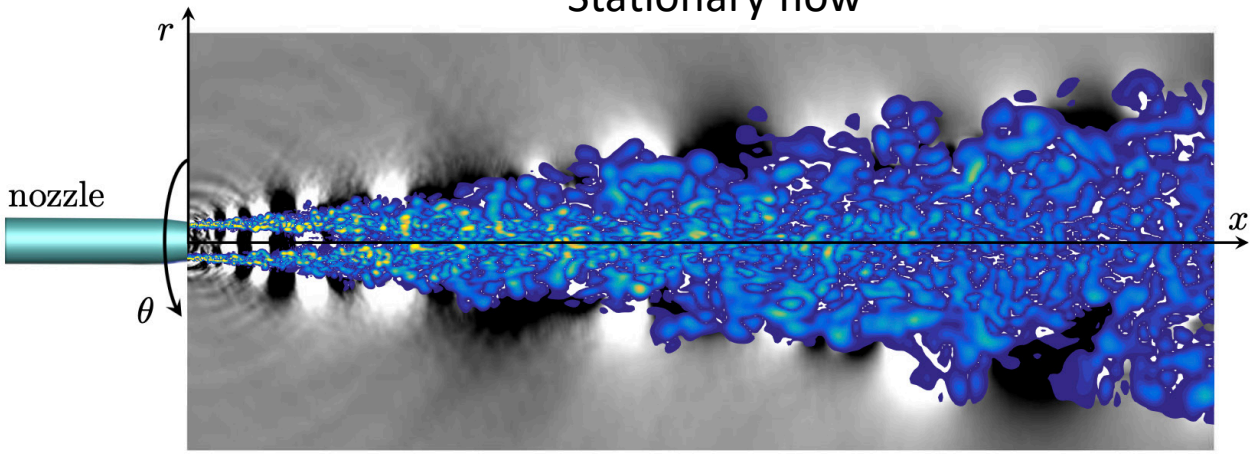
Algorithm	CPU time (hours)			Memory (GB)	
	Total	Action of $\boldsymbol{R}$ and $\boldsymbol{R}^*$	SVD/QR		
RSVD-LU	7.53×10^7	7.53×10^7	0.762	1.33×10^5	133 TB
RSVD-Δt	1.83×10^4	1.83×10^4	0.762	7.36×10^2	0.75 TB

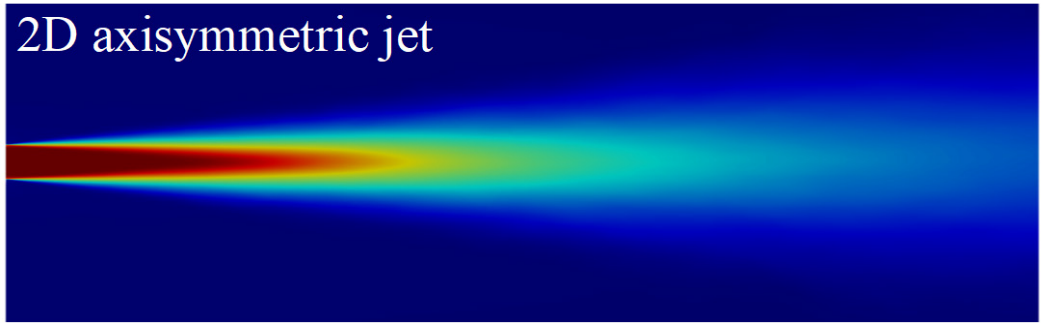
Using 300 cores: < 2 days using RSVD-Δt > 28 years using RSVD-LU

Extension to harmonic resolvent analysis

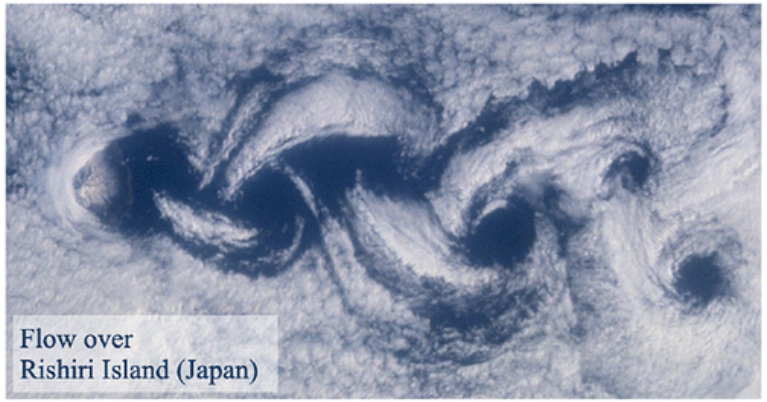
Harmonic resolvent analysis = Resolvent analysis for periodic flows

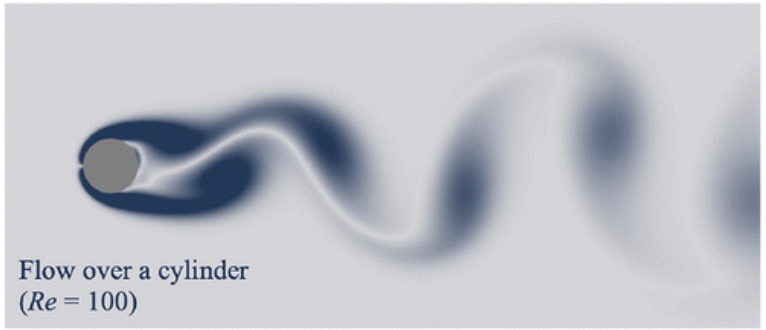
Stationary flow





Periodic flow





Harmonic resolvent analysis formulation

$$\frac{\partial \boldsymbol{q}}{\partial t} - \boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{0}$$

$$\boldsymbol{q}(\boldsymbol{x},t) = \bar{\boldsymbol{q}}(\boldsymbol{x},t) + \boldsymbol{q}'(\boldsymbol{x},t)$$

$$\frac{\partial \boldsymbol{q}'}{\partial t} - A(t)\,\boldsymbol{q}' = \boldsymbol{f} \qquad A(t) = A(t+T)$$

Fourier Transform

$$[T\hat{\boldsymbol{q}}]_{j\omega_f} = \mathrm{i}j\omega_f \hat{\boldsymbol{q}}_{j\omega_f} - \sum_{j=-\infty}^{\infty} \hat{\mathrm{A}}_{(k-j)\omega_f}\,\hat{\boldsymbol{q}}_{j\omega_f} = \hat{f}_{k\omega_f}$$

Solve and SVD

$$\hat{\boldsymbol{q}} = \underbrace{T^{-1}}_{H}\,\hat{\boldsymbol{f}}$$

Harmonic resolvent analysis formulation

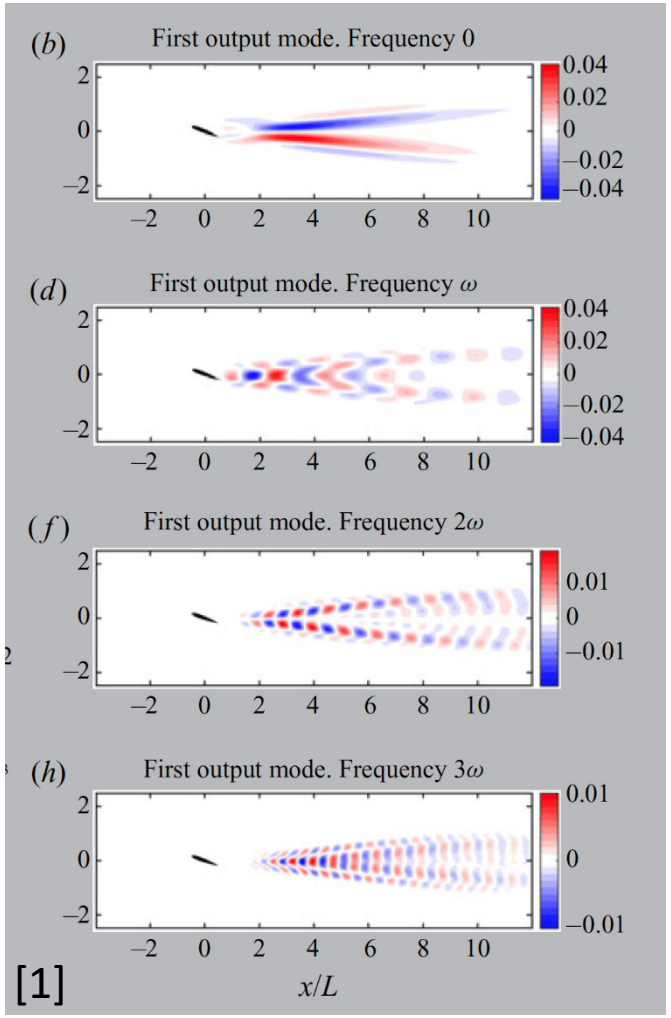
$$\frac{\partial \boldsymbol{q}}{\partial t} - \boldsymbol{N}(\boldsymbol{q}) = \boldsymbol{0} \xrightarrow{\boldsymbol{q}(\boldsymbol{x},t) = \bar{\boldsymbol{q}}(\boldsymbol{x},t) + \boldsymbol{q}'(\boldsymbol{x},t)} \frac{\partial \boldsymbol{q}'}{\partial t} - A(t)\,\boldsymbol{q}' = \boldsymbol{f}$$

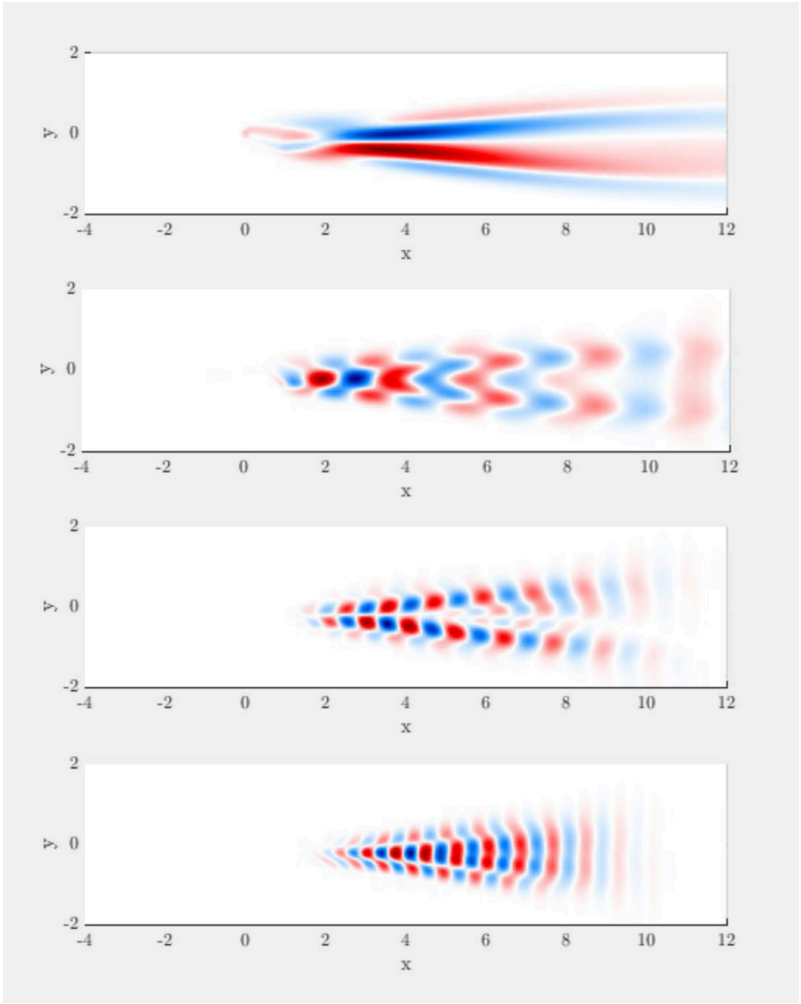
$$A(t) = A(t+T)$$

$$\mathbf{T} = \begin{bmatrix} \ddots & \vdots & \vdots & \vdots & \vdots & \iddots \\ \cdots & \mathbf{R}_{-\omega_f}^{-1} & \hat{\mathbf{A}}_{-\omega_f} & \hat{\mathbf{A}}_{-2\omega_f} & \hat{\mathbf{A}}_{-3\omega_f} & \cdots \\ \cdots & \hat{\mathbf{A}}_{\omega_f} & \mathbf{R}_{0}^{-1} & \hat{\mathbf{A}}_{-\omega_f} & \hat{\mathbf{A}}_{-2\omega_f} & \cdots \\ \cdots & \hat{\mathbf{A}}_{2\omega_f} & \hat{\mathbf{A}}_{\omega_f} & \mathbf{R}_{\omega_f}^{-1} & \hat{\mathbf{A}}_{-\omega_f} & \cdots \\ \cdots & \hat{\mathbf{A}}_{3\omega_f} & \hat{\mathbf{A}}_{2\omega_f} & \hat{\mathbf{A}}_{\omega_f} & \mathbf{R}_{2\omega_f}^{-1} & \cdots \\ \iddots & \vdots & \vdots & \vdots & \vdots & \ddots \end{bmatrix}$$

Example: harmonic resolvent modes for an airfoil

RSVD-LU





RSVD-Δt

[1] Padovan *et al.*, *JFM*, 2020

Summary: scalable resolvent analysis

We have developed a new algorithm for computing resolvent modes using RSVD + time stepping that

Scales linearly with problem dimension

Computes all desired frequencies simultaneously (via the choice of forcing)

Minimizes memory consumption (using streaming Fourier sums)

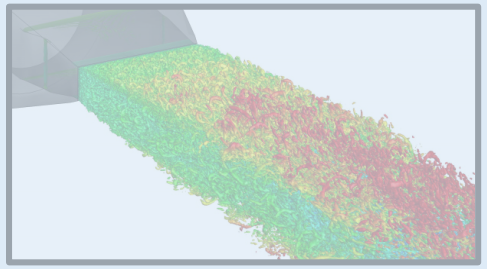
Minimizes the simulation length (explicitly removing transients)

Enabling resolvent analysis of previously intractable large-scale problems

Outline

Efficient Computation

Motivation

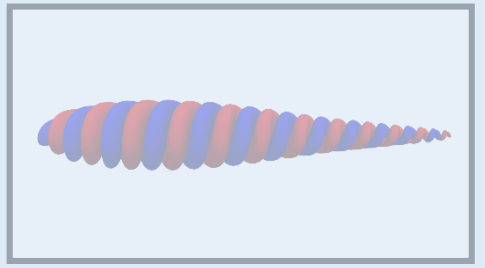


Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$

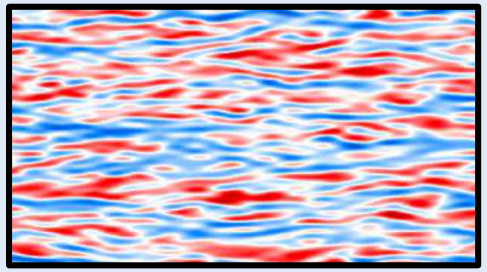
$$\rightarrow \quad \rightarrow \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

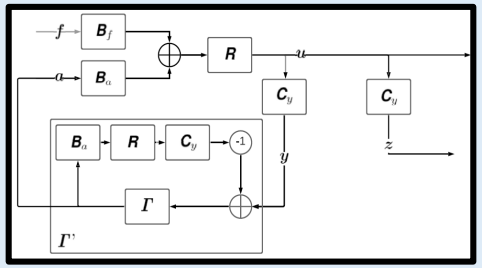


Estimation & Control

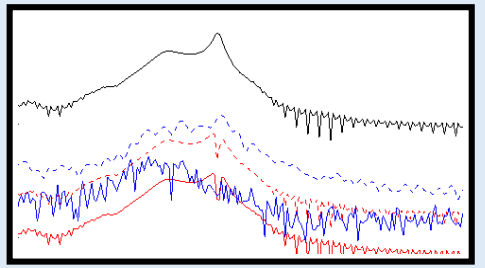
Flow estimation



Flow control

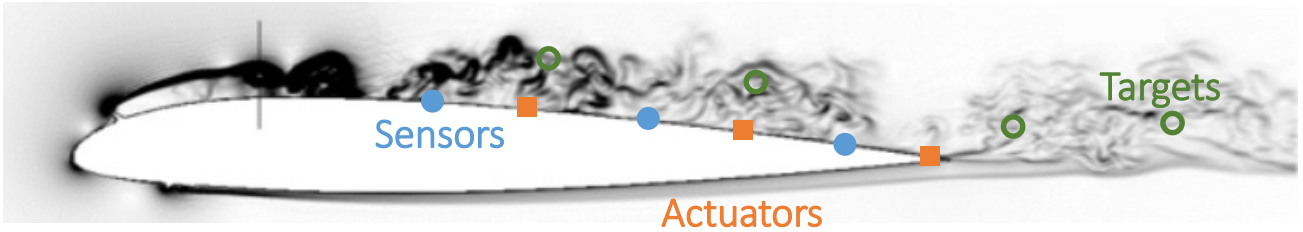


Features



Flow estimation & control

- **Flow estimation:** use limited measurements to approximate the state of the flow at other locations and/or for other quantities that are not measured
- **Flow control:** manipulate a flow through some form of actuation to produce a desired change in the flow behavior



Standard techniques are poorly suited for turbulence

- Turbulent flows are:

Nonlinear

Most methods assume a linear system forced by white noise

High dimensional

Kalman filters scale poorly with problem size

Chaotic

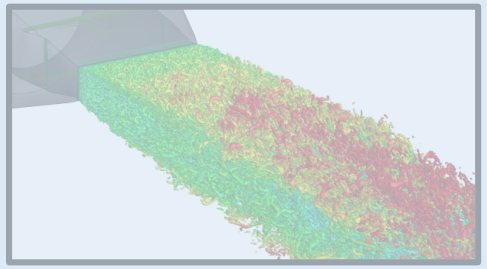
Do not explicitly leverage existence of coherent structures

Our objective: use resolvent analysis to obtain an improved flow estimation and control framework

Outline

Efficient Computation

Motivation



Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$



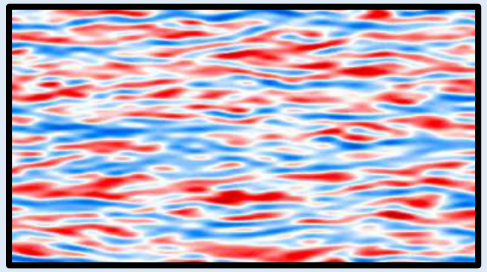
$$\rightarrow \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

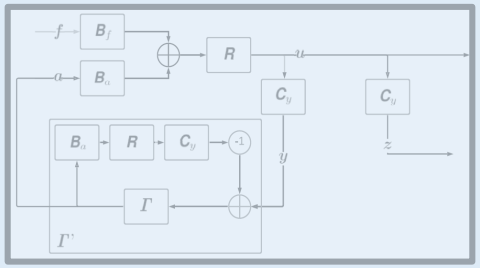


Estimation & Control

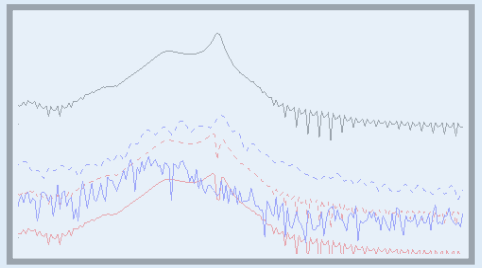
Flow estimation



Flow control



Features



Resolvent-based estimation

Linearized Navier-Stokes

$$\frac{d\boldsymbol{q}}{dt} - A\boldsymbol{q} = B\boldsymbol{f}$$

Measurement (with noise)

$$\boldsymbol{y} = C\boldsymbol{q} + \boldsymbol{n}$$

Cost function to minimize

$$J = \int_{-\infty}^{\infty} \left(\underbrace{E\{\boldsymbol{e}^*(t)\boldsymbol{e}(t)\}}_{\text{Minimize estimation error}} + \underbrace{\Lambda_-(t)T_{q,c}(t) + \Lambda_-^*(t)T_{q,c}^*(t)}_{\text{Enforce causality}} \right) dt$$

Lagrange multipliers: $\Lambda_-(t > 0) = 0$

Objective: minimize estimation error

$$\boldsymbol{e}(t) = \boldsymbol{q}(t) - \tilde{\boldsymbol{q}}(t)$$

Seek transfer function between the measurements and state

$$\tilde{\boldsymbol{q}}(t) = \int_{-\infty}^{\infty} T_q(t-\tau)\boldsymbol{y}(\tau)\, d\tau \xrightarrow{\text{FT}} \hat{\tilde{\boldsymbol{q}}}(\omega) = \hat{T}_q(\omega)\hat{\boldsymbol{y}}(\omega)$$

Find T_q that minimizes error

Martini et al., *JFM*, 2022

Resolvent-based estimation

Finding the fixed points of the cost function leads to:

$$\hat{T}_{q,c}(\omega)\hat{G}_l(\omega) + \hat{\Lambda}_-(\omega) = \hat{G}_r(\omega)$$
$$T_{q,c}(t<0) = 0$$
$$\Lambda_-(t>0) = 0$$

} Wiener-Hopf problem

Martini et al., *JFM*, 2022

Resolvent-based estimation

Finding the fixed points of the cost function leads to:

$$\hat{T}_{q,c}(\omega)\hat{G}_l(\omega) + \hat{\Lambda}_-(\omega) = \hat{G}_r(\omega)$$
$$T_{q,c}(t<0) = 0$$
$$\Lambda_-(t>0) = 0$$

} Wiener-Hopf problem

$$\hat{G}_l(\omega) = R_{yf}(\omega)\hat{F}(\omega)R^*_{yf}(\omega) + \hat{N}(\omega) \qquad \hat{G}_r(\omega) = R_{qf}(\omega)\hat{F}(\omega)R^*_{yf}(\omega)$$

$R_{qf} = (\mathrm{i}\omega I - A)^{-1}$

Standard resolvent operator

$R_{yf} = C\,(\mathrm{i}\omega I - A)^{-1}$

Modified resolvent operator

Resolvent-based estimation

Finding the fixed points of the cost function leads to:

$$\hat{T}_{q,c}(\omega)\hat{G}_l(\omega) + \hat{\Lambda}_-(\omega) = \hat{G}_r(\omega)$$
$$T_{q,c}(t<0) = 0$$
$$\Lambda_-(t>0) = 0$$

} Wiener-Hopf problem

$$\hat{G}_l(\omega) = R_{yf}(\omega)\hat{F}(\omega)R^*_{yf}(\omega) + \hat{N}(\omega) \qquad \hat{G}_r(\omega) = R_{qf}(\omega)\hat{F}(\omega)R^*_{yf}(\omega)$$

Space-time forcing statistics

Space-time noise statistics

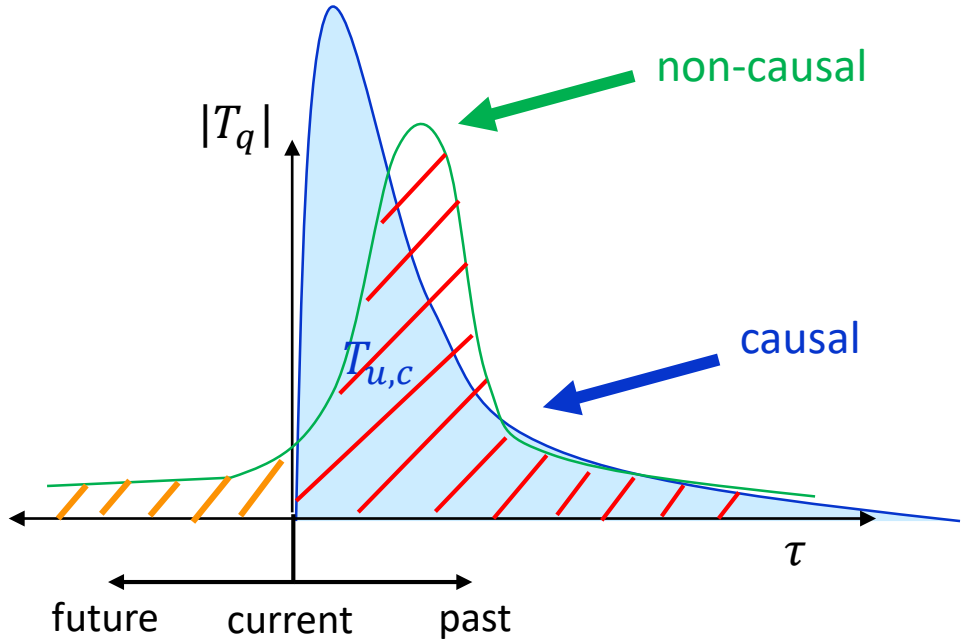
Resolvent-based estimation

- Transform back to the time domain and convolve with measurement

Causal: $$\tilde{\boldsymbol{q}}(t) = \int_{-\infty}^{0} T_{\boldsymbol{q}}(t-\tau)\boldsymbol{y}(\tau)\,d\tau$$

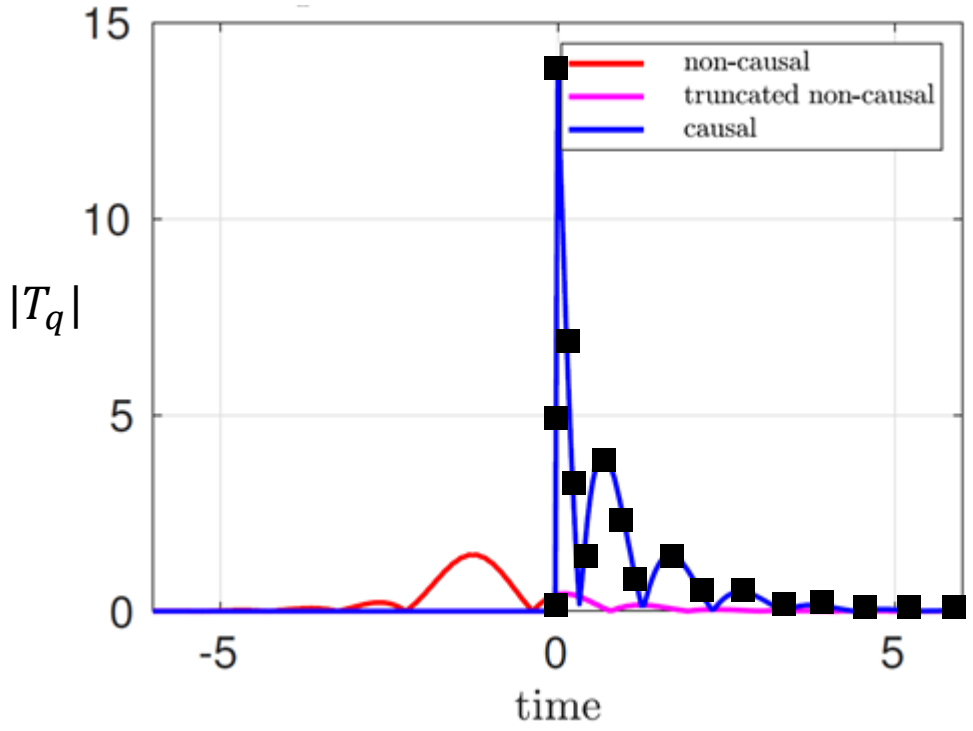
Noncausal: $$\tilde{\boldsymbol{q}}(t) = \int_{-\infty}^{\infty} T_{\boldsymbol{q}}(t-\tau)\boldsymbol{y}(\tau)\,d\tau$$

Truncated noncausal: $$\tilde{\boldsymbol{q}}(t) = \int_{-\infty}^{0} T_{\boldsymbol{q}}(t-\tau)\boldsymbol{y}(\tau)\,d\tau$$



Connection with Kalman filters

- ***But isn't Kalman filtering optimal?*** The Kalman filter is the optimal linear estimator under the assumption of white-in-time forcing
- The causal resolvent-based estimator reproduces the Kalman filter when equivalent assumptions are made



Connection with Kalman filters

- ***But isn't Kalman filtering optimal?*** The Kalman filter is the optimal linear estimator under the assumption of white-in-time forcing
- The causal resolvent-based estimator reproduces the Kalman filter when equivalent assumptions are made
- The resolvent-based method has two advantages
  - **Accounts for space-time forcing color**

Kalman

$$E\{\boldsymbol{f}(t)\boldsymbol{f}^*(t')\} = I\delta(t-t')$$
$$E\{\boldsymbol{n}(t)\boldsymbol{n}^*(t')\} = \epsilon I\delta(t-t')$$

Resolvent-based

$$E\{\boldsymbol{f}(t)\boldsymbol{f}^*(t')\} = F(t-t')$$
$$E\{\boldsymbol{n}(t)\boldsymbol{n}^*(t')\} = N(t-t')$$

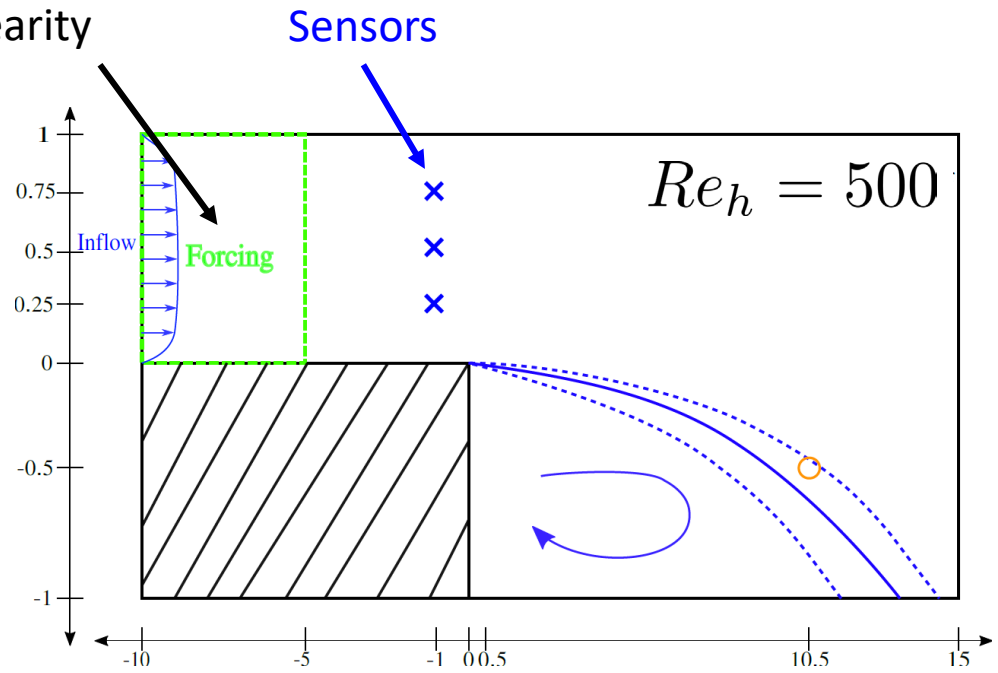
Connection with Kalman filters

- ***But isn't Kalman filtering optimal?*** The Kalman filter is the optimal linear estimator under the assumption of white-in-time forcing
- The causal resolvent-based estimator reproduces the Kalman filter when equivalent assumptions are made
- The resolvent-based method has two advantages
  - Accounts for space-time forcing color
  - **Fast computation without the need for a priori model reduction**

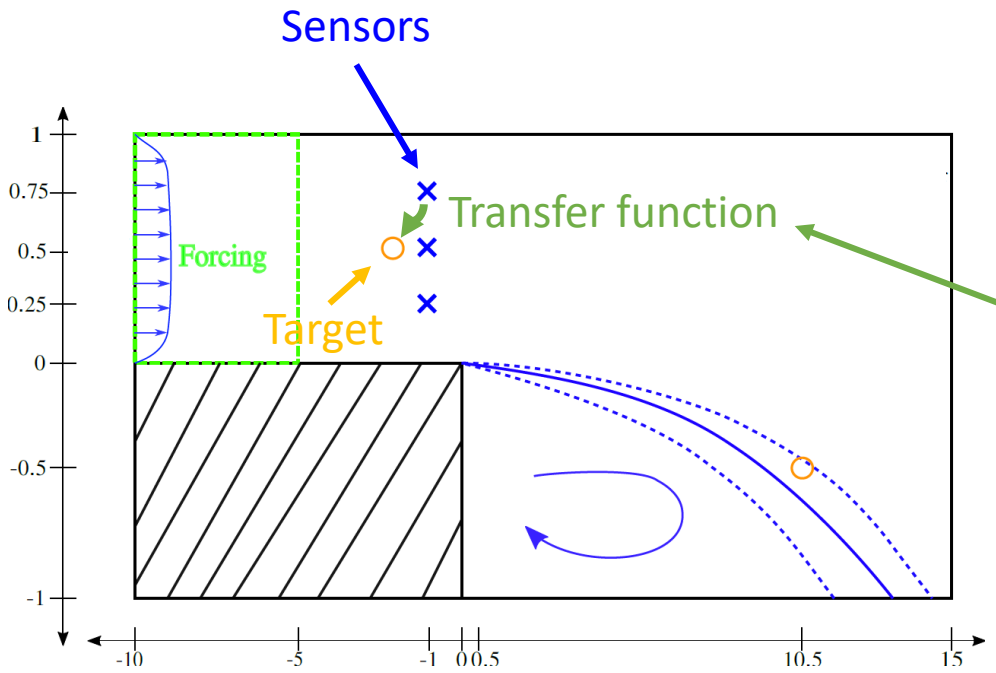
Kalman	**Resolvent-based**
Algebraic Riccati equation	Time-stepping approach
$\rightarrow O(N^3)$	$\rightarrow O(N)$

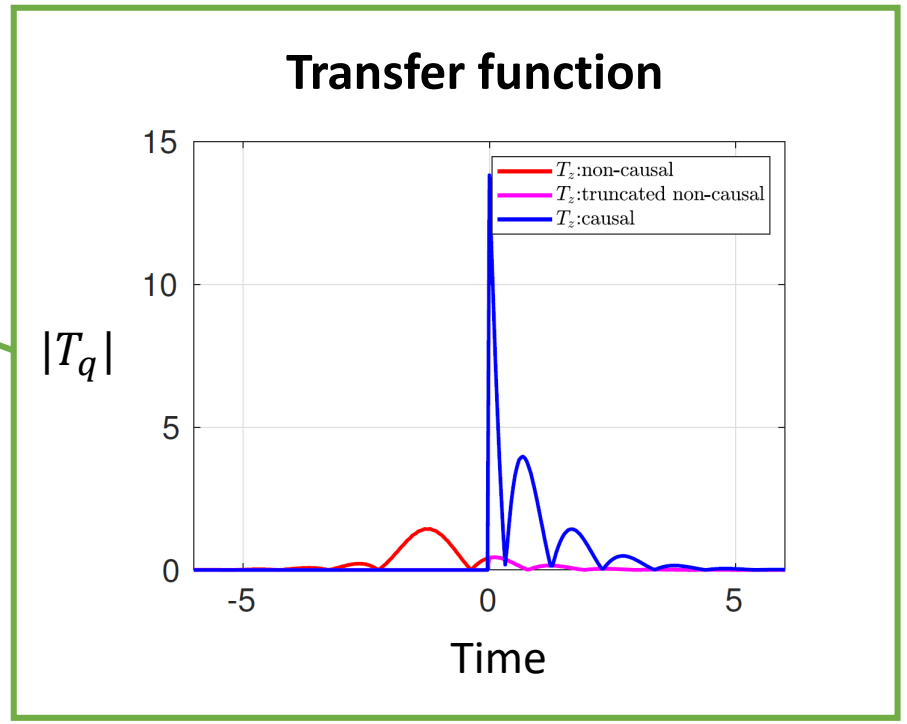
Example: backward facing step

Amplitude determines degree of nonlinearity

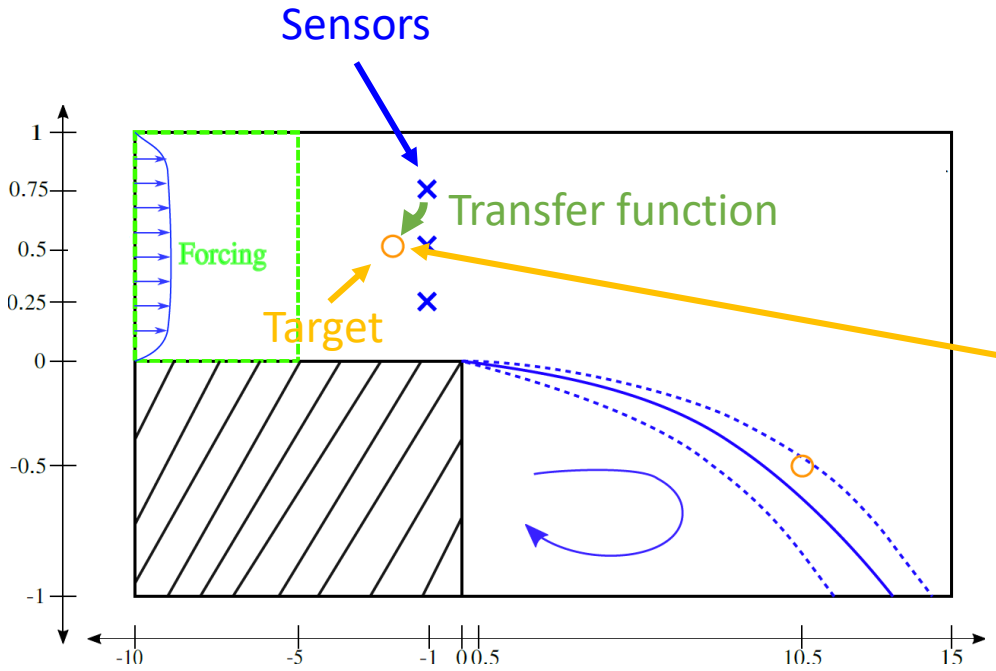


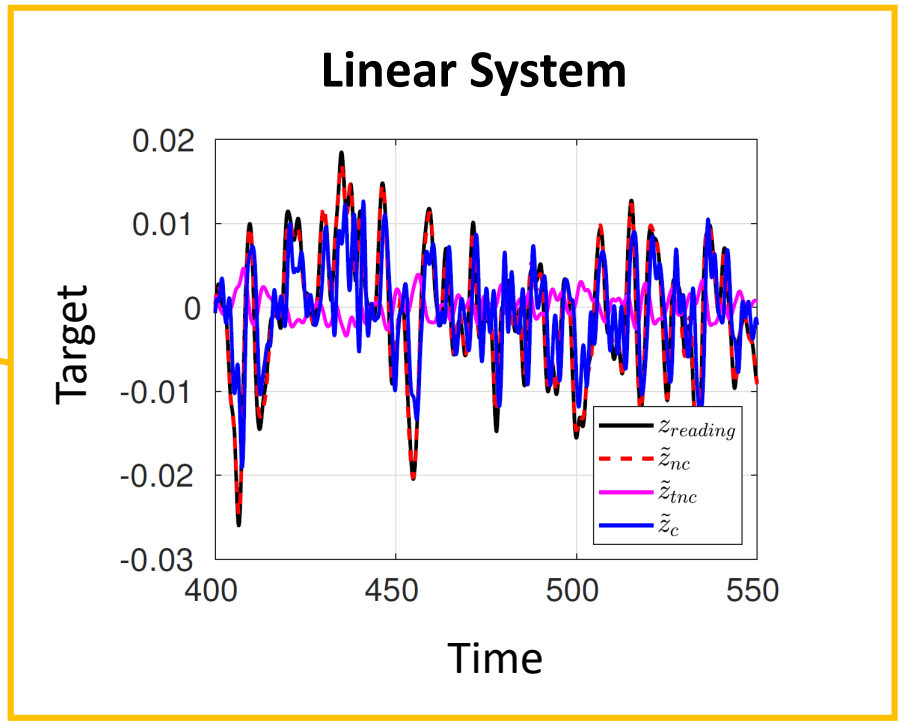
Example: backward facing step



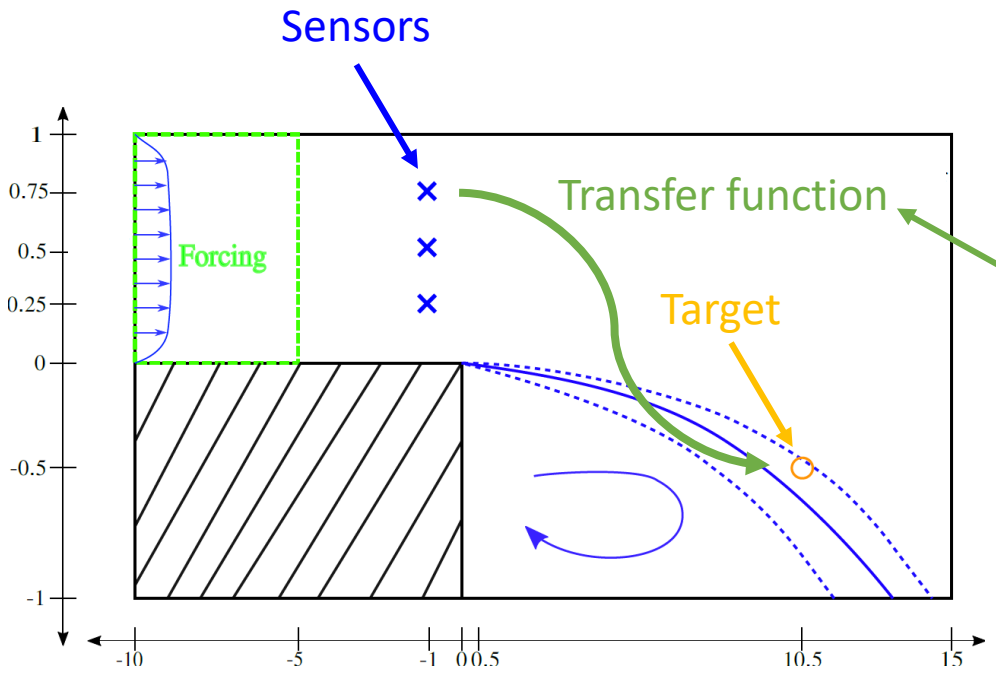


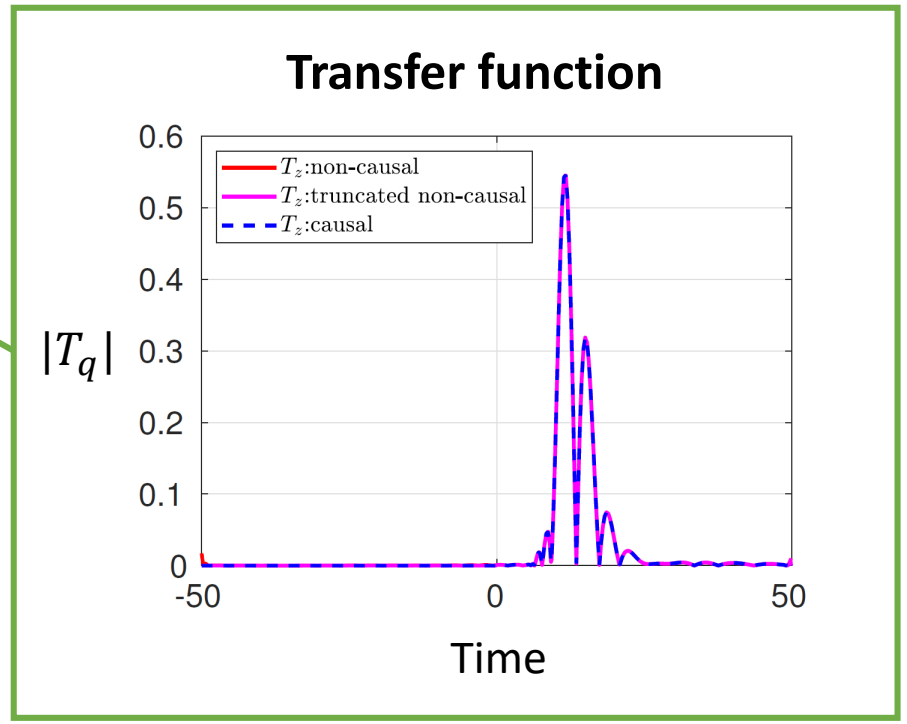
Example: backward facing step



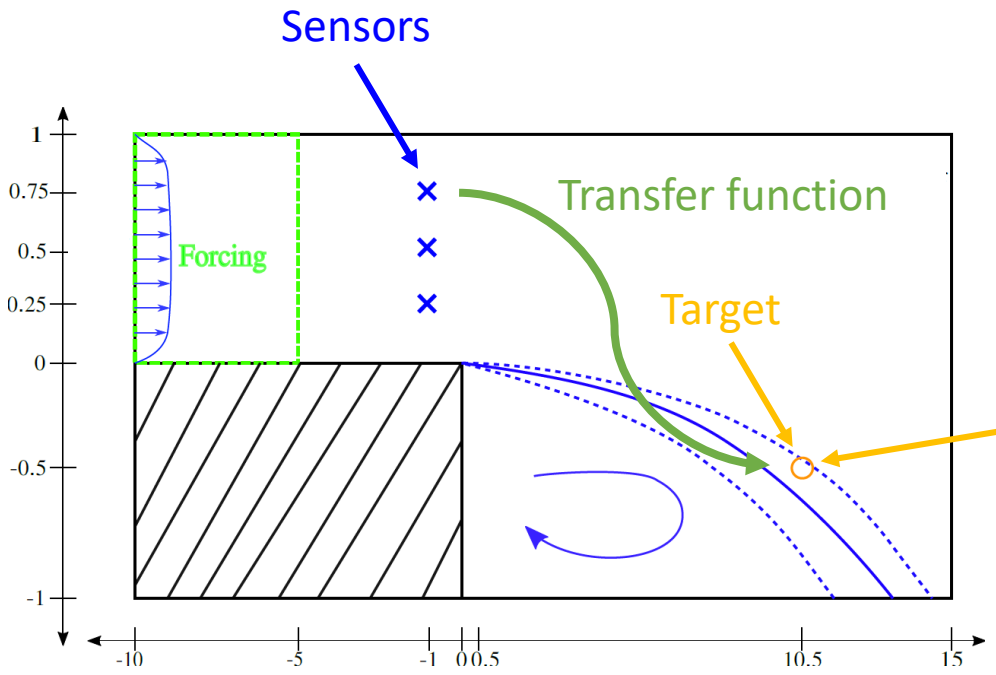


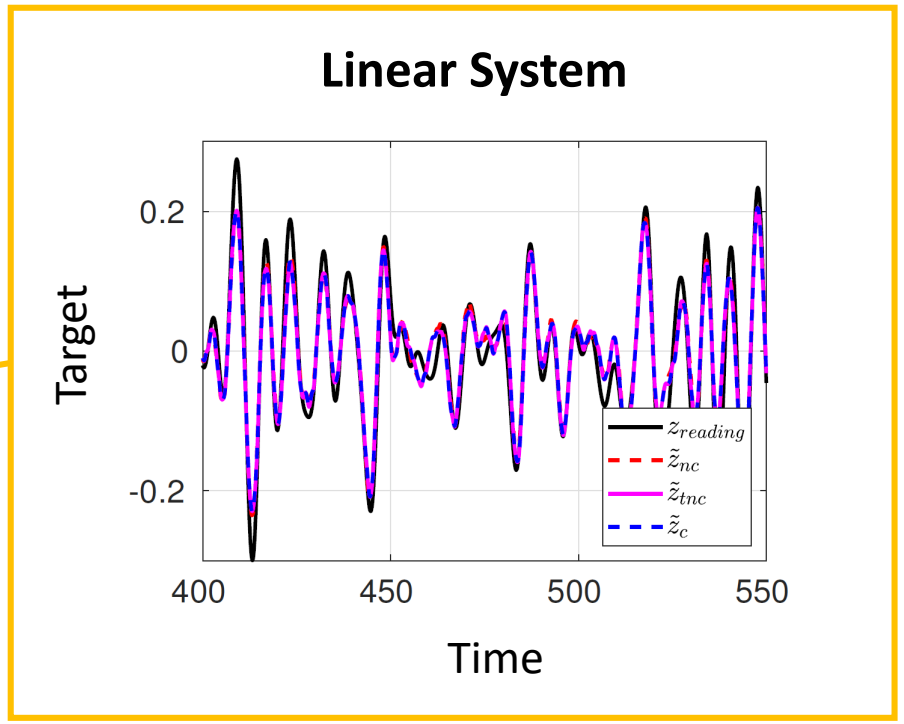
Example: backward facing step



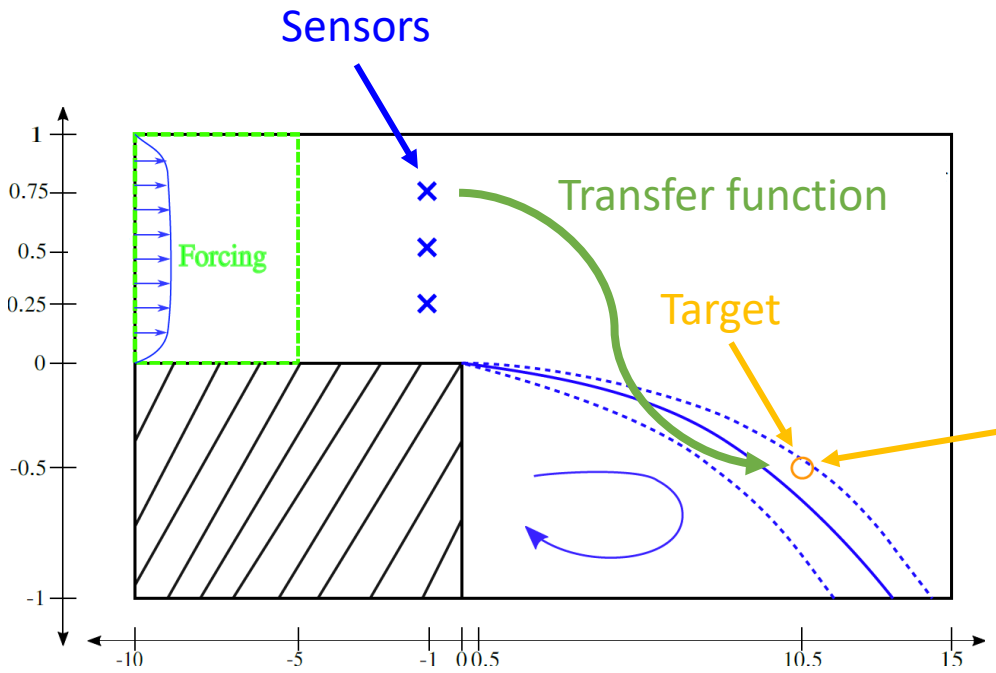


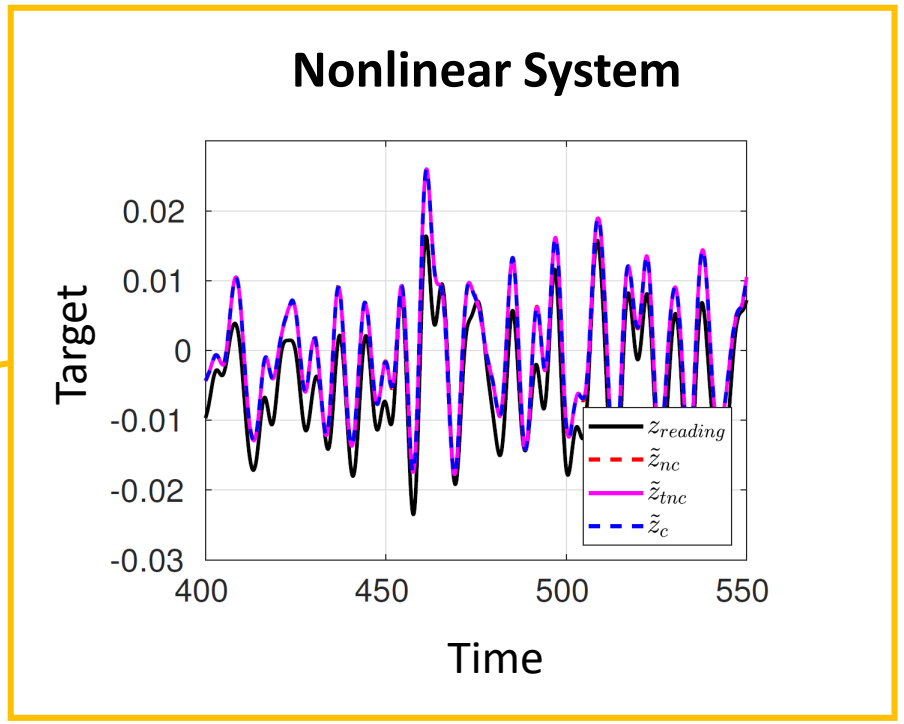
Example: backward facing step





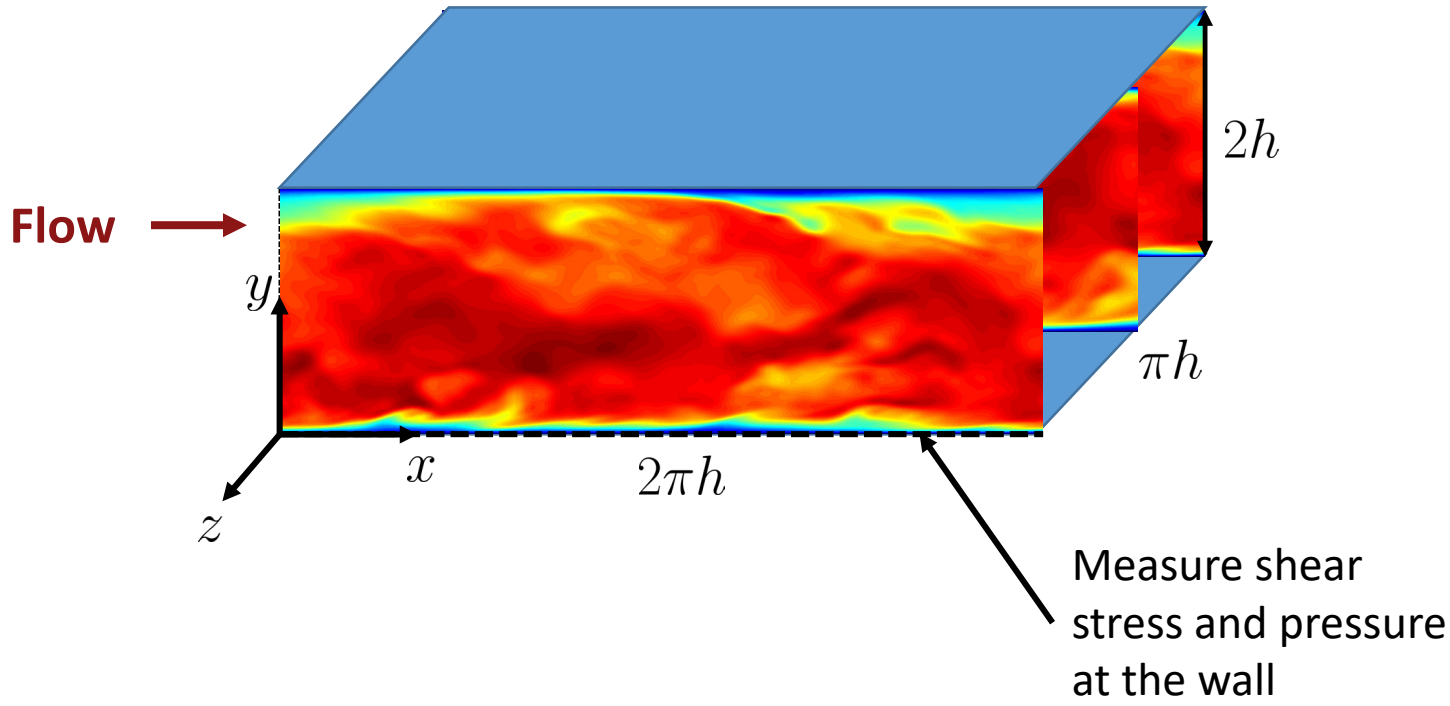
Example: backward facing step





Application: turbulent channel flow

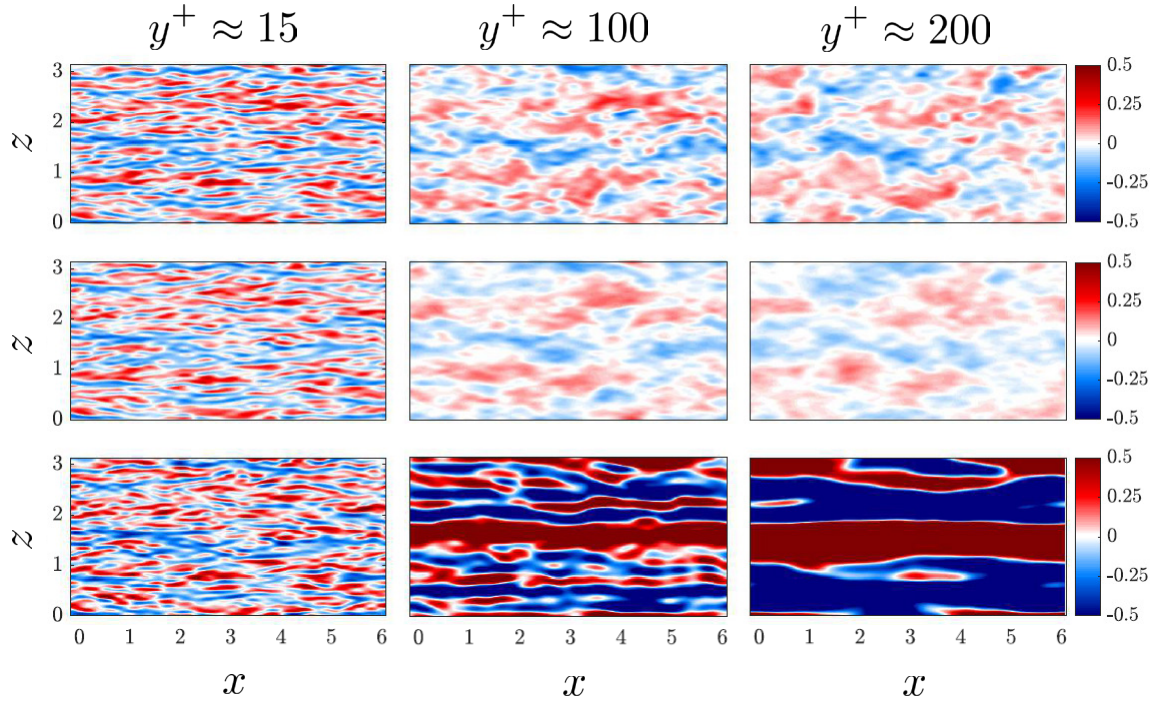
- Reynolds numbers: $Re_\tau \approx 180, \mathbf{550}, 1000$



Amaral et al., *JFM*, 2021

Application: turbulent channel flow

Instantaneous snapshot of u' for three different wall-normal distances and $Re_\tau = 550$



DNS

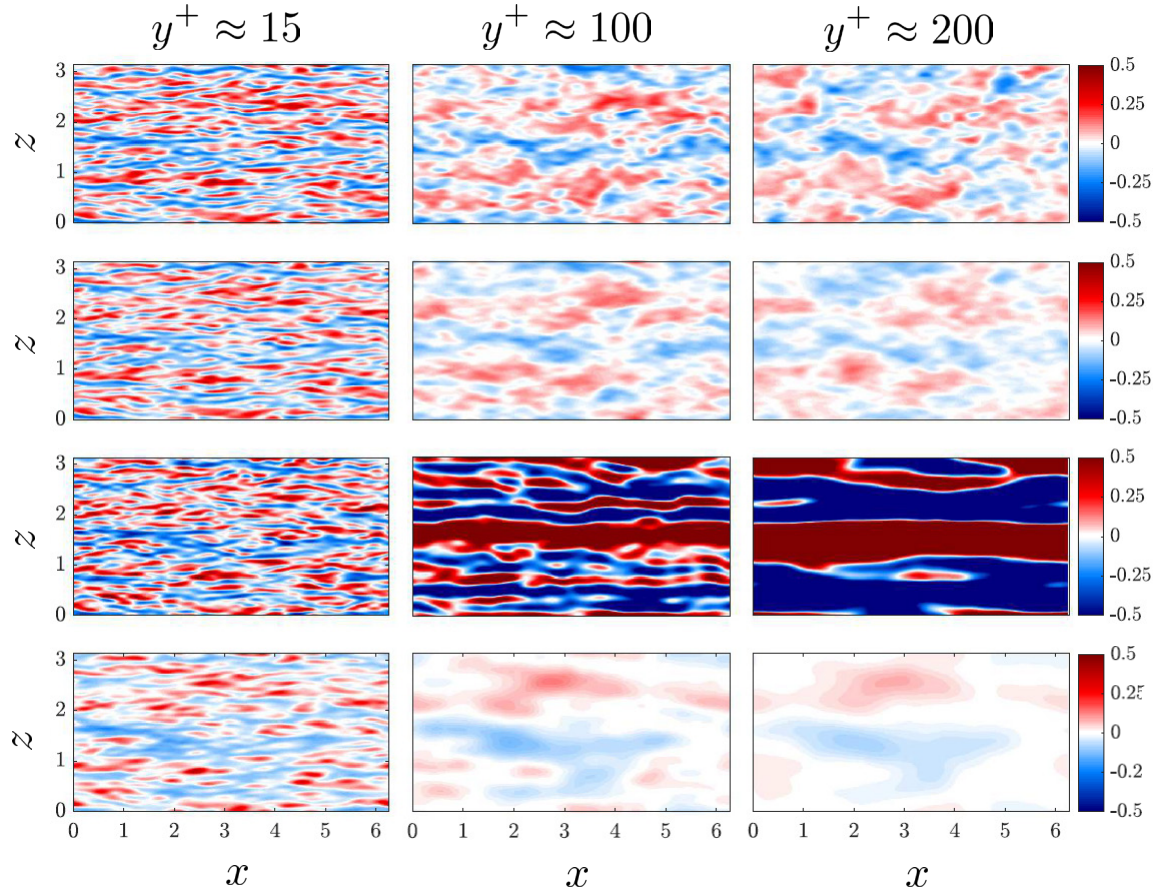
Resolvent: with forcing color

Resolvent: without forcing color (Kalman smoother)

Amaral et al., *JFM*, 2021

Application: turbulent channel flow

Instantaneous snapshot of u' for three different wall-normal distances and $Re_\tau = 550$



DNS

Resolvent: with forcing color

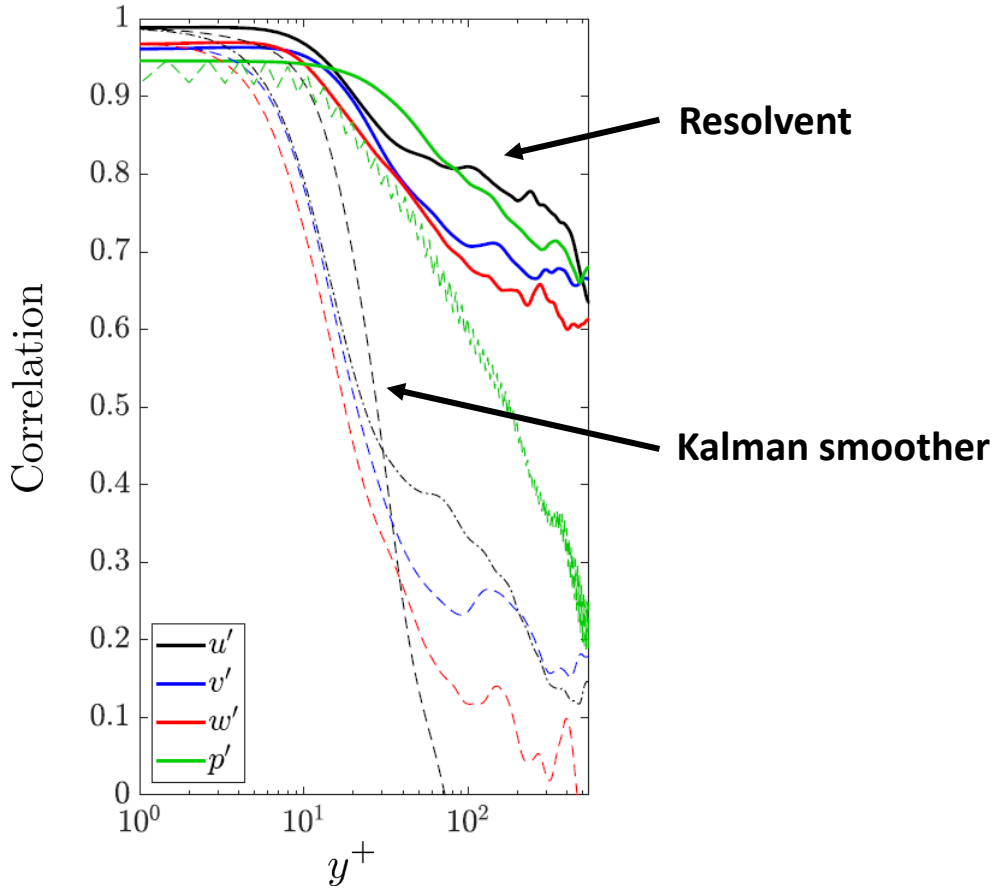
Resolvent: without forcing color (Kalman smoother)

Eddy viscosity

Amaral et al., *JFM*, 2021

Application: turbulent channel flow

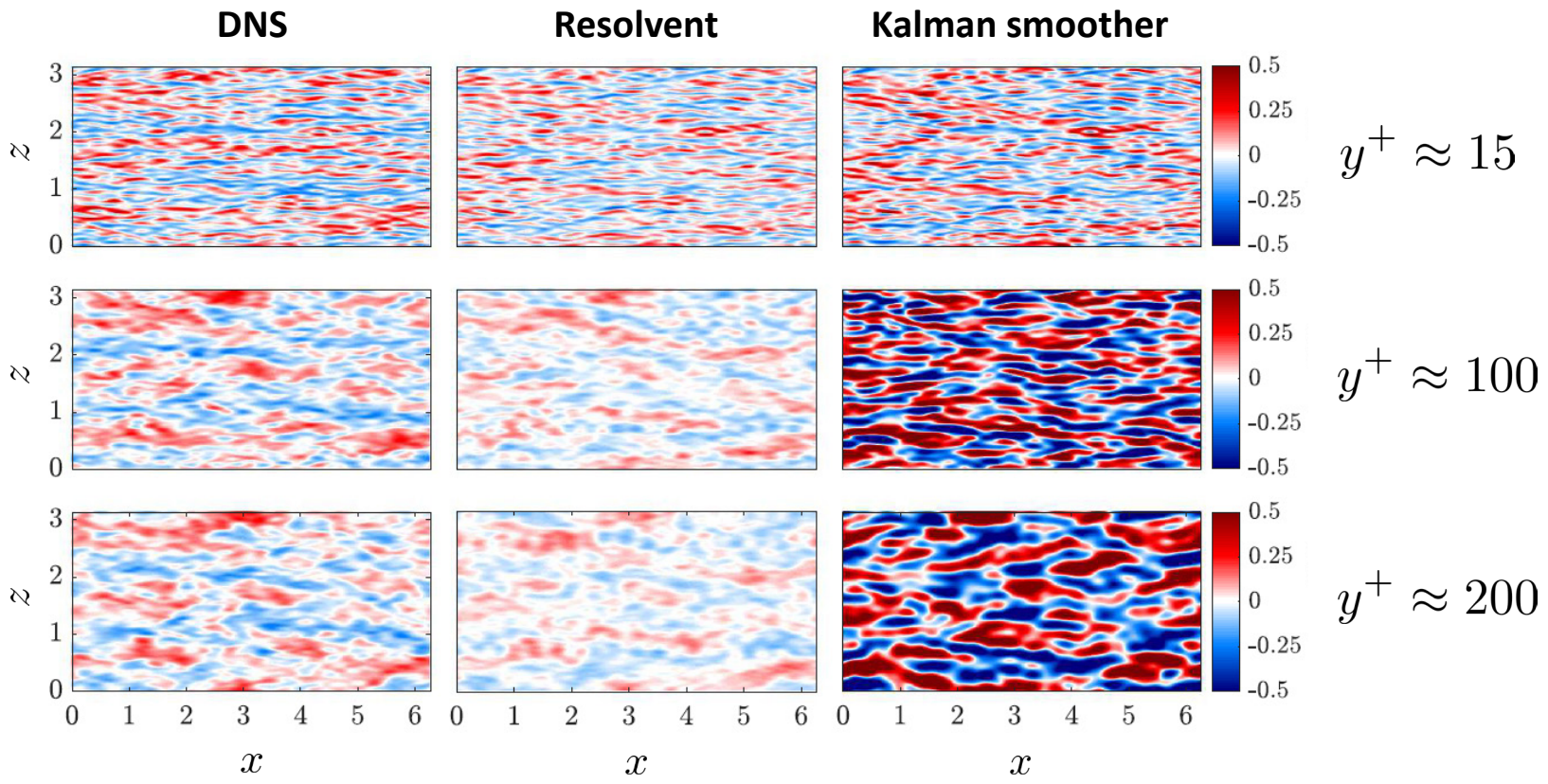
Correlation between the DNS and reconstructed fields as a function of y^+ for $Re_\tau = 550$



Amaral et al., *JFM*, 2021

Application: turbulent channel flow

Instantaneous snapshot of u' for three different wall-normal distances and $Re_\tau = 1000$

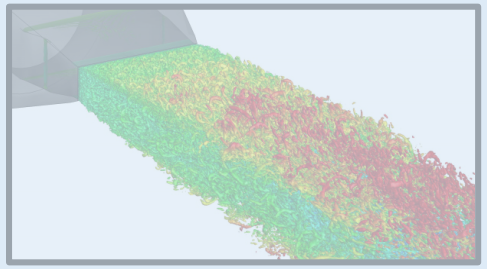


Amaral et al., *JFM*, 2021

Outline

Efficient Computation

Motivation

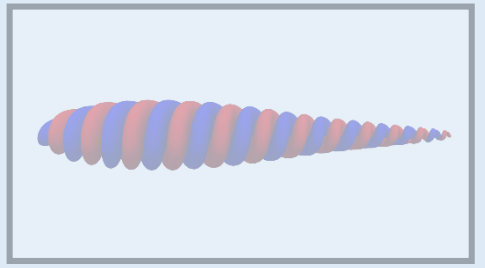


Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$

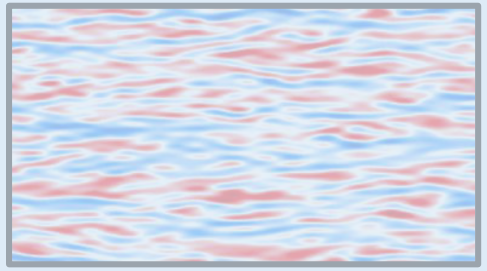
$$\rightarrow \quad \rightarrow \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

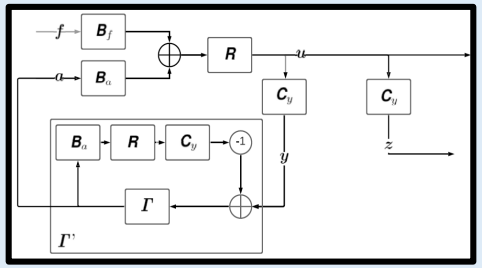


Estimation & Control

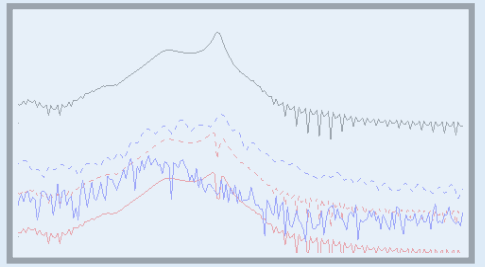
Flow estimation



Flow control



Features



Resolvent-based optimal control

- Find control law $\hat{a}_c = \Gamma' \hat{y}$ that minimizes the cost function

$$J' = \int_{-\infty}^{\infty} \Big(\underbrace{\boldsymbol{z}^*(t)\boldsymbol{z}(t)}_{\text{Minimize fluctuation energy at targets}} + \underbrace{\boldsymbol{a}_c^*(t) P \boldsymbol{a}_c(t)}_{\text{Penalize control effort}} + \underbrace{\Lambda_-(t)\boldsymbol{a}_c(t) + \Lambda_-^*(t)\boldsymbol{a}_c^*(t)}_{\text{Enforce causality}} \Big)\, dt$$

$$\boldsymbol{z} = C_z \boldsymbol{q}$$

- Leads to a Wiener-Hopf problem involving resolvent operators and forcing statistics

$$\hat{G}_l(\omega) = R_{zf}(\omega)\hat{F}(\omega)R_{yf}^*(\omega) + \hat{N}(\omega) \qquad \hat{H}_l(\omega) = R_{za}^*(\omega)R_{za}^*(\omega) + P$$

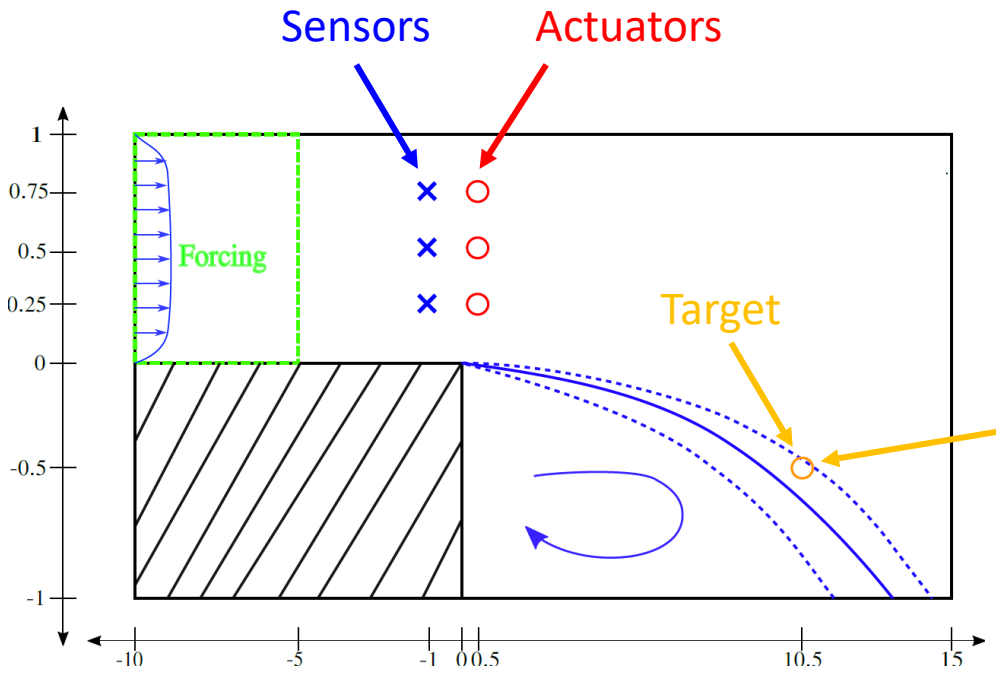
$$\hat{G}_r(\omega) = R_{zf}(\omega)\hat{F}(\omega)R_{yf}^*(\omega) \qquad \hat{H}_r(\omega) = -R_{za}^*(\omega)$$

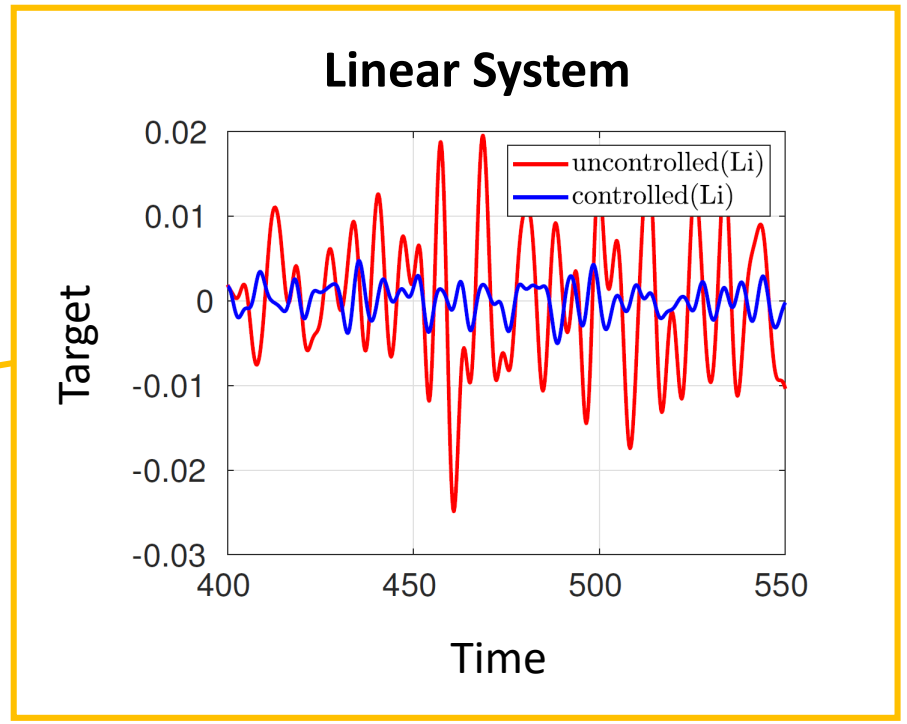
Resolvent-based optimal control

- Properties
  - Recovers LQG control for white-in-time forcing
  - Efficient computation via a time-marching method enables efficient application to large systems

Resolvent-based optimal control

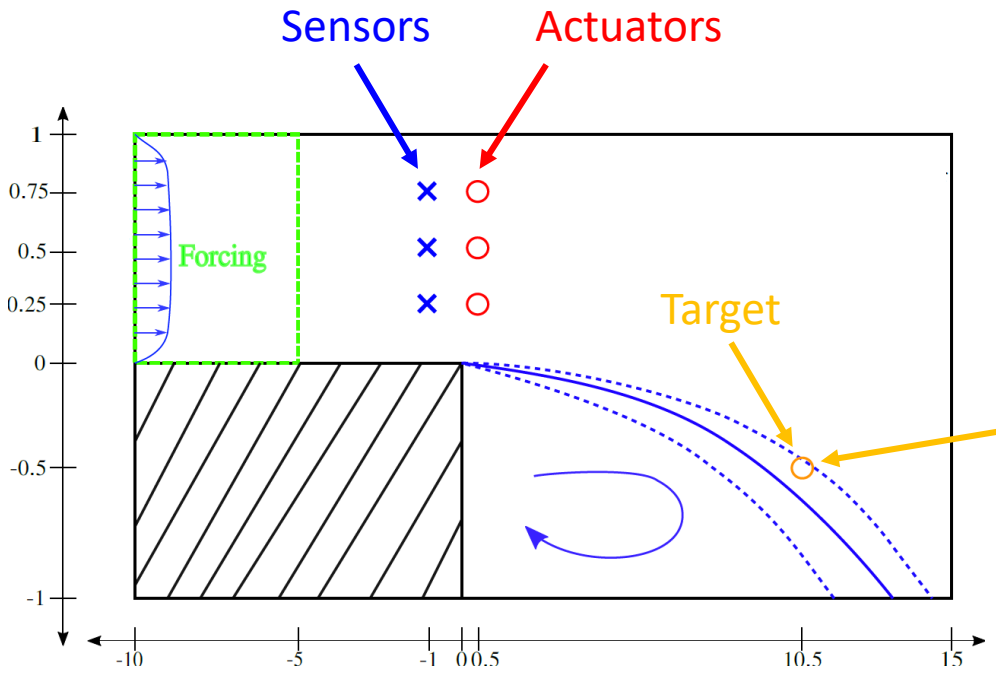
- Results for the backward facing step

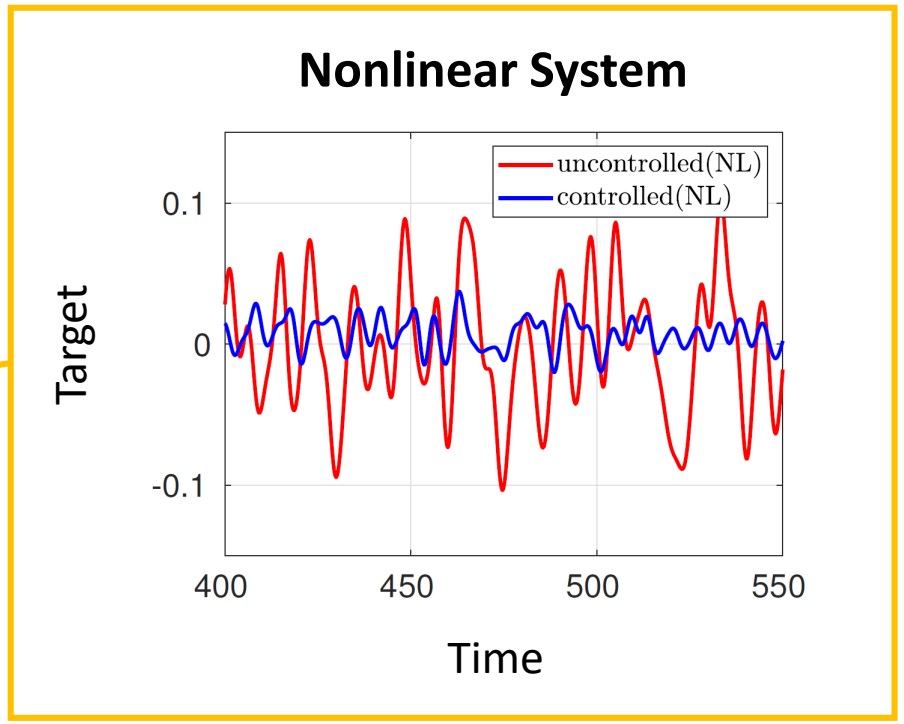




Resolvent-based optimal control

- Results for the backward facing step

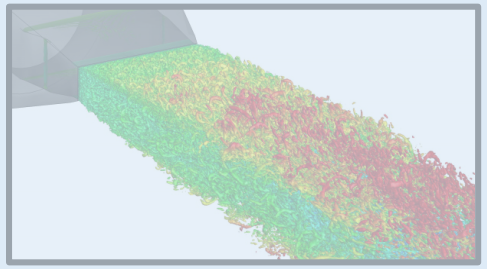




Outline

Efficient Computation

Motivation



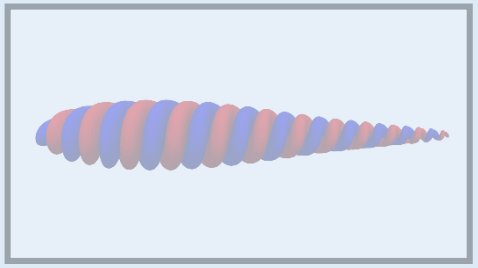
Algorithm

$$\frac{d\boldsymbol{q}}{dt} = A\boldsymbol{q} + \boldsymbol{f}(t)$$



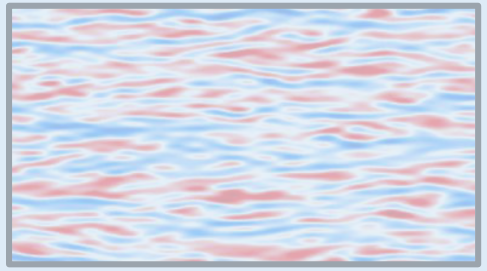
$$\rightarrow \hat{\boldsymbol{q}}_j = R(\omega_j)\hat{\boldsymbol{f}}_j$$

Demonstration

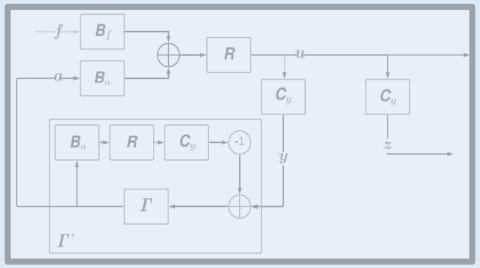


Estimation & Control

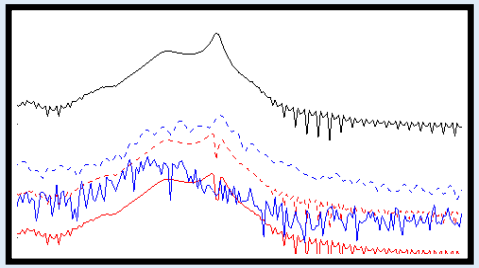
Flow estimation



Flow control



Features



Data-driven construction of control kernels

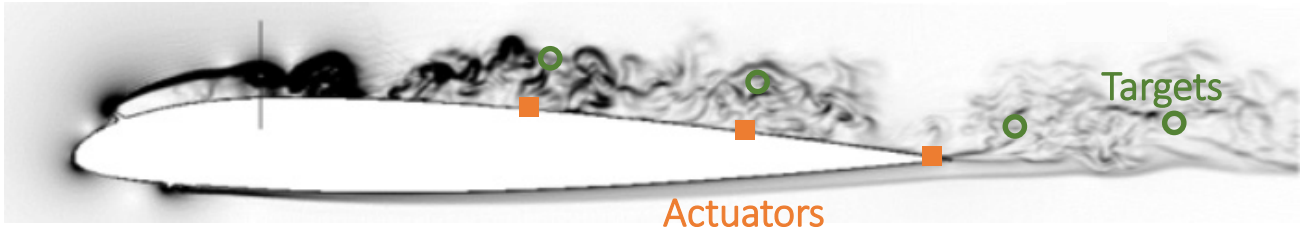
- The W-H coefficients can be constructed from data!

$\hat{G}_l(\omega)$, $\hat{G}_r(\omega)$:

Can be computed from sensor/target ***correlations*** of the ***uncontrolled*** system

$\hat{H}_l(\omega)$, $\hat{H}_r(\omega)$:

Can be computed from the ***impulse response*** between actuators & targets



Data-driven construction of control kernels

- This data-driven implementation broadens the applicability of the framework

Simulations

We can compute the control kernels ***without*** the need for a ***linearized code*** or ***adjoint capabilities***

Experiments

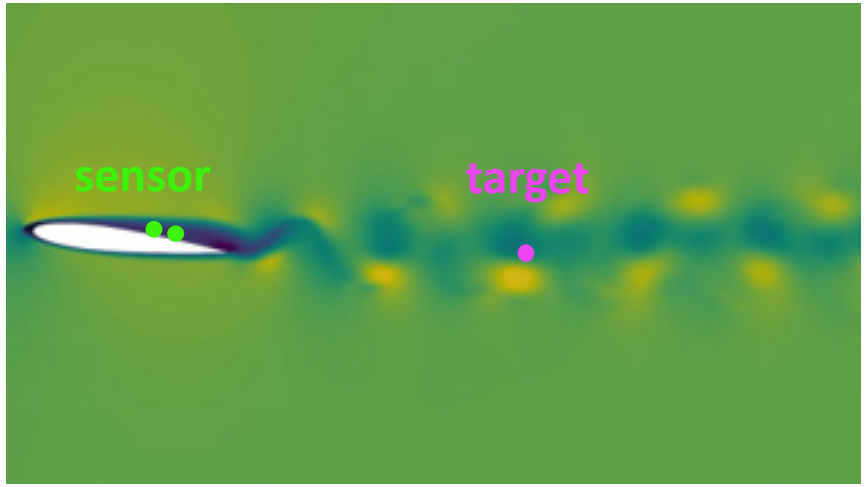
We can compute the control kernels by ***measuring correlations*** and performing ***impulse responses***

Example: laminar airfoil estimation

Truth

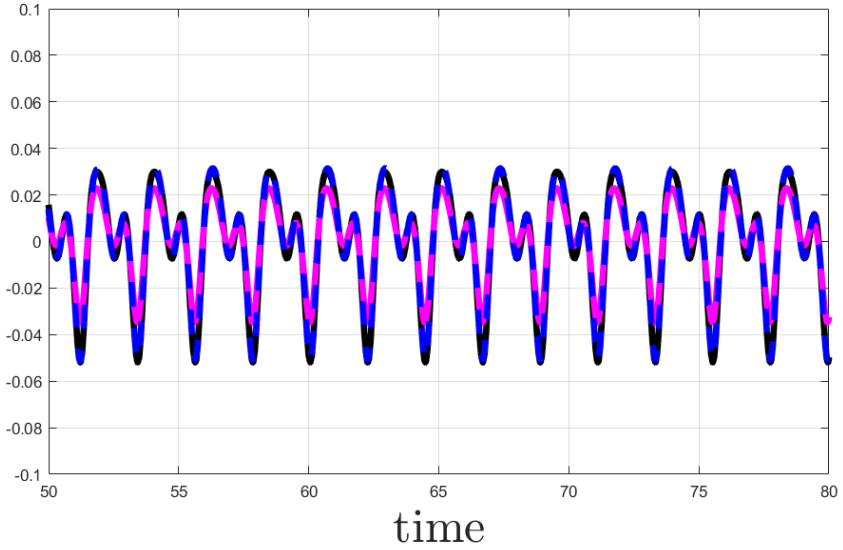
Causal

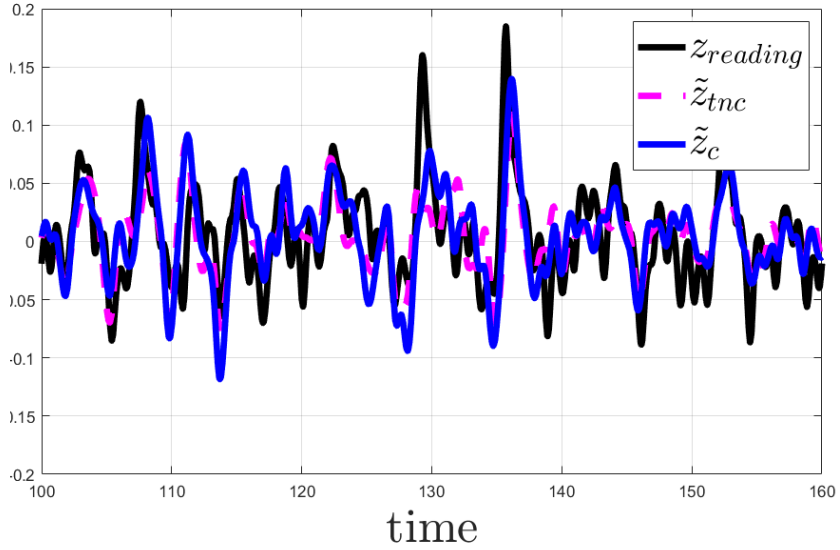
Trunc. non-causal



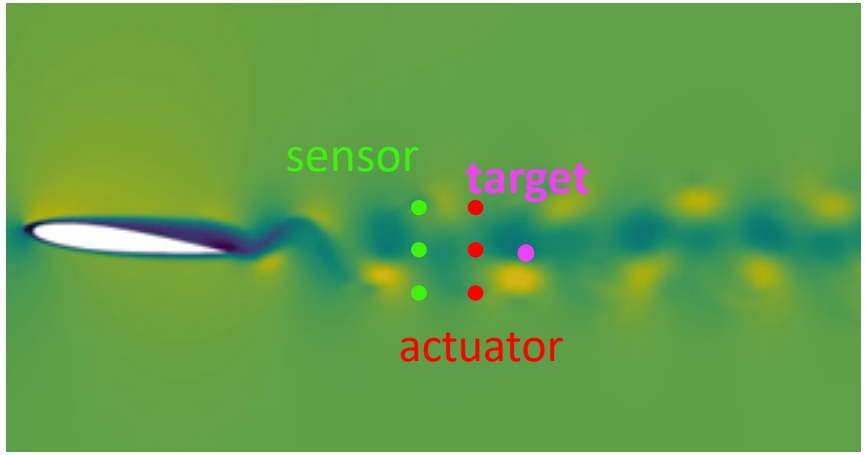
$Re = 5{,}000$
$Ma = 0.3$
$AoA = 6^{\circ}$

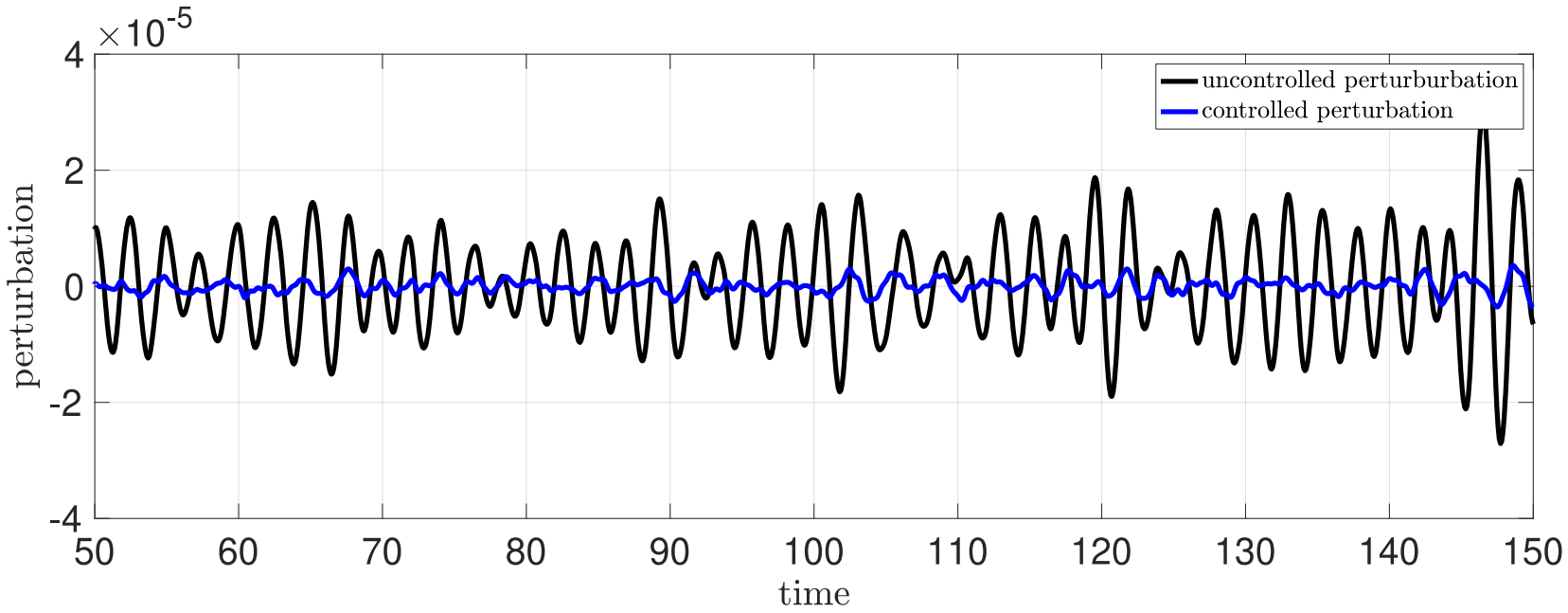
u'



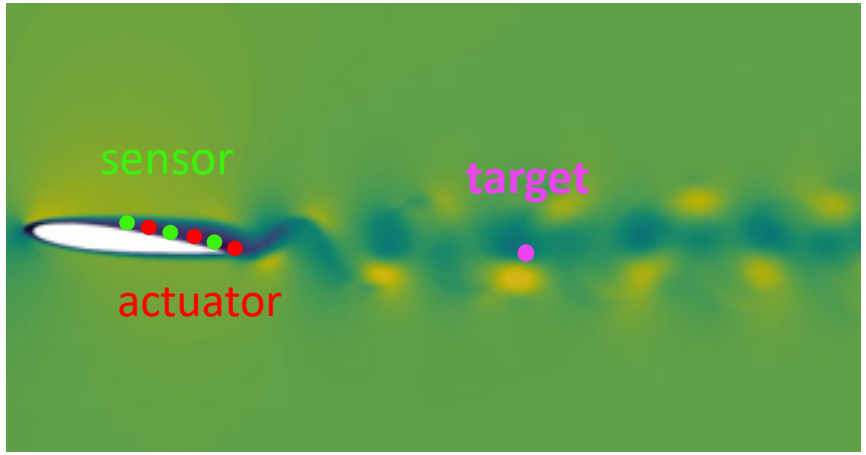


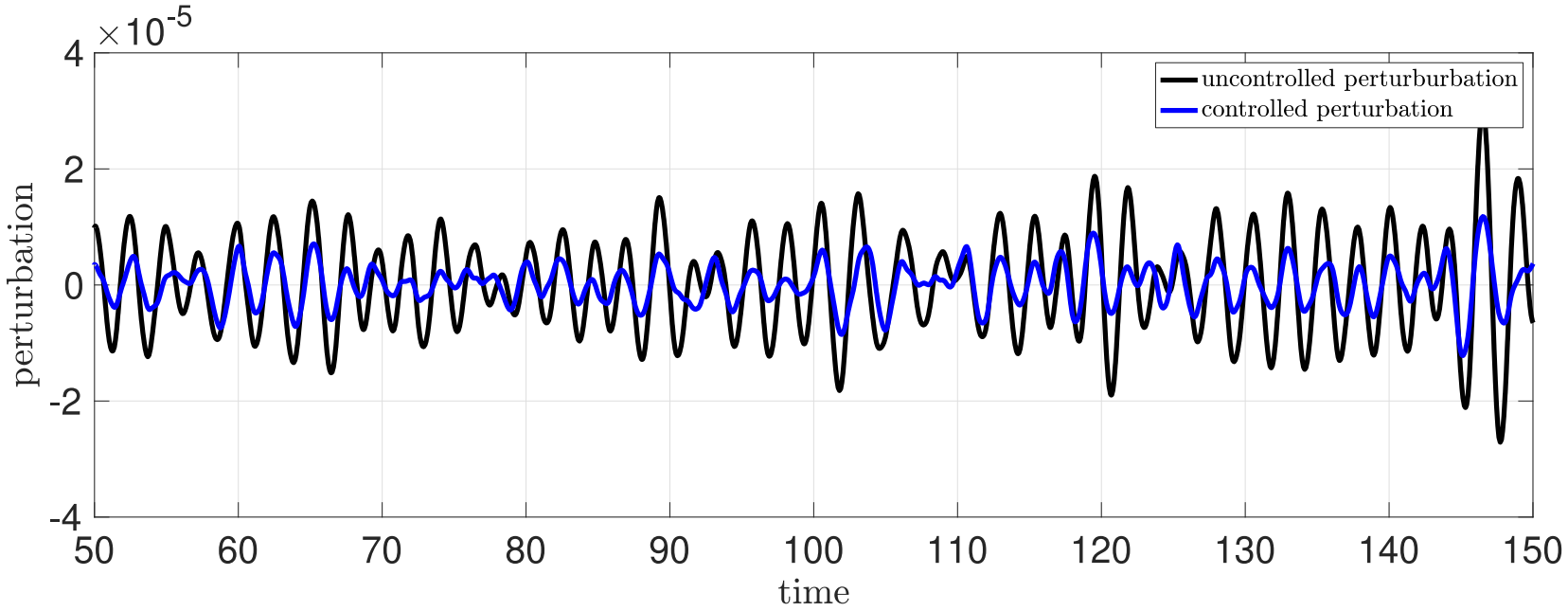
Example: laminar airfoil linear control



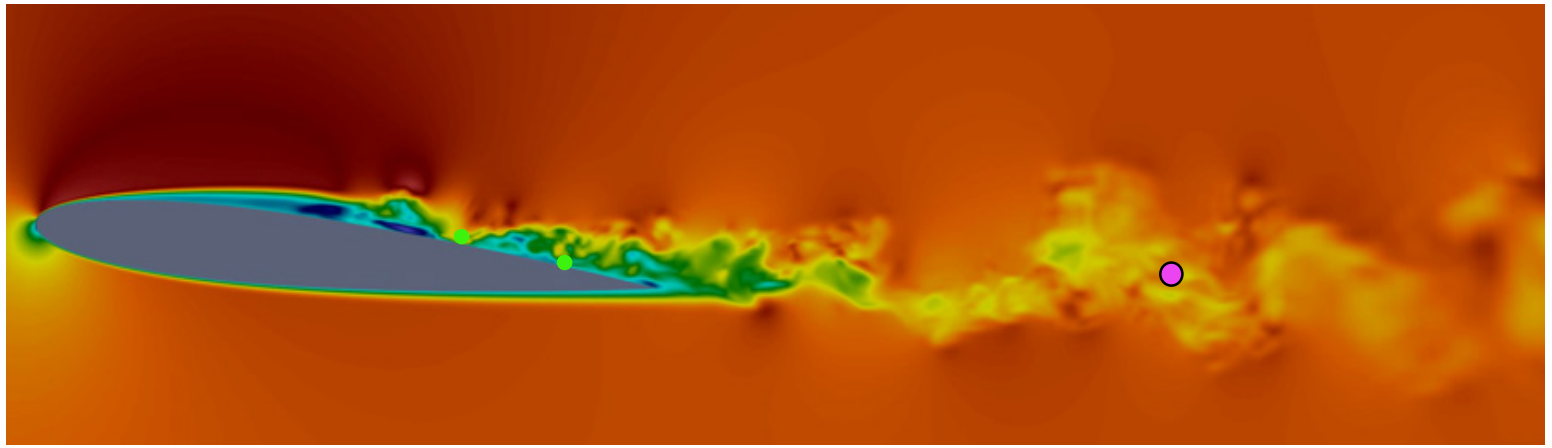


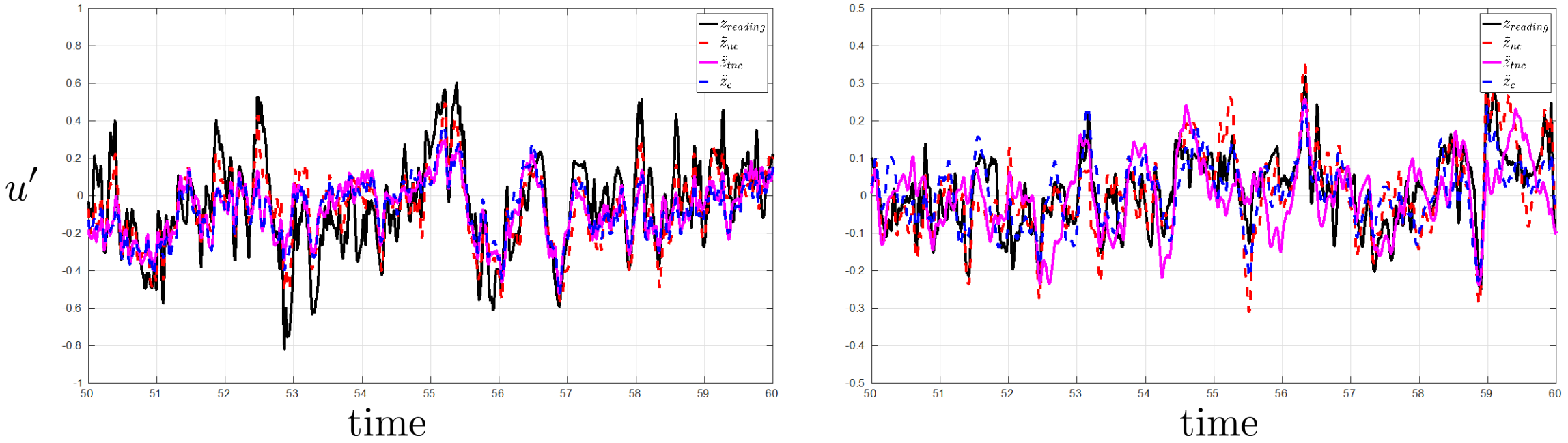
Example: laminar airfoil linear control





Example: turbulent airfoil estimation





Efficient exploration of sensor/actuator placement

- Requires the ability to:

    1. Efficiently construct many control/estimation kernels for different configurations

    2. Efficiently evaluate the performance of each configuration

Building multiple estimation/control kernels

Solve impulse responses

$$\frac{d\boldsymbol{q}_i}{dt} = A\boldsymbol{q}_i + B_f\hat{\boldsymbol{e}}_i\delta(t)$$

Majority of the cost

FT

Build resolvent operators

$$R_{zf} = C_z[\hat{\boldsymbol{q}}_1, \hat{\boldsymbol{q}}_2, \ldots, \hat{\boldsymbol{q}}_{n_f}]$$

$$R_{yf} = C_y[\hat{\boldsymbol{q}}_1, \hat{\boldsymbol{q}}_2, \ldots, \hat{\boldsymbol{q}}_{n_f}]$$

Very inexpensive

Solve W-H

Estimation & control kernels

$$T(\tau) \quad \Gamma(\tau)$$

Once we have the impulse response data $[\hat{q}_1, \hat{q}_2, \ldots, \hat{q}_{n_f}]$,

we can use it to form the target and sensor resolvent operators R_{zf} and R_{yf}

for any choice of target and sensor locations just by multiplying by C_z and C_y

at no additional cost

A priori performance evaluation

- The performance of the controller can be estimated without applying control

Target cross-spectral density of the controlled flow

$$S_{zz} = \hat{H}_r \hat{\Gamma} \hat{G}_l \hat{\Gamma}^* \hat{H}_r - \hat{H}_r^* \hat{\Gamma} \hat{G}_r C_z^* - C_z \hat{G}_r \hat{\Gamma}^* \hat{H}_r$$

Trace is quantity to be minimized

Estimation & control coefficients and kernels

"Estimate" because this expression assumes that the flow will respond linearly to the control

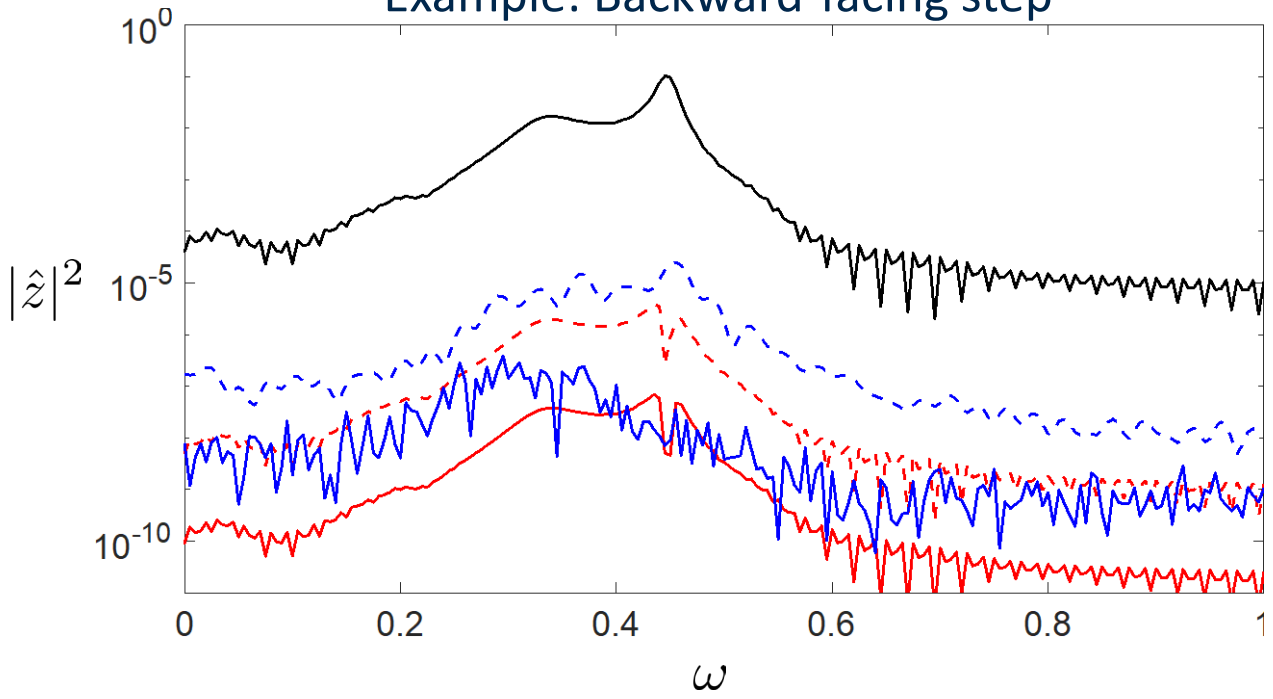
Efficient exploration of sensor/actuator placement

Inexpensive variation of kernels **+** a priori performance evaluation **=** ***Inexpensive exploration of sensor and actuator placement***, even for large systems

Efficient exploration of sensor/actuator placement

Inexpensive variation of kernels + a priori performance evaluation = ***Inexpensive exploration of sensor and actuator placement***, even for large systems

Example: Backward-facing step



Uncontrolled

Actual nonlinear performance of setup A

Actual nonlinear performance of setup B

Summary: resolvent-based estimation and control

We have developed optimal flow estimation and control methods, formulated in terms of resolvent operators, with

Reduced cost (using a matrix-free time-marching method)

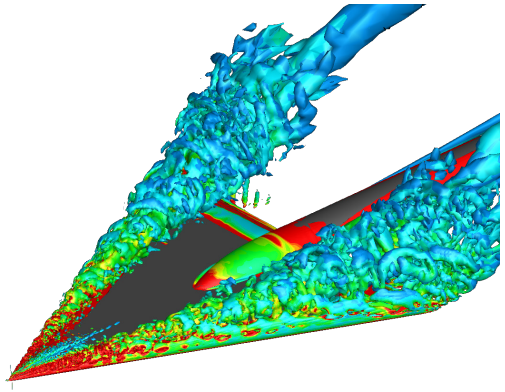
Improved accuracy (by accounting for space-time forcing color)

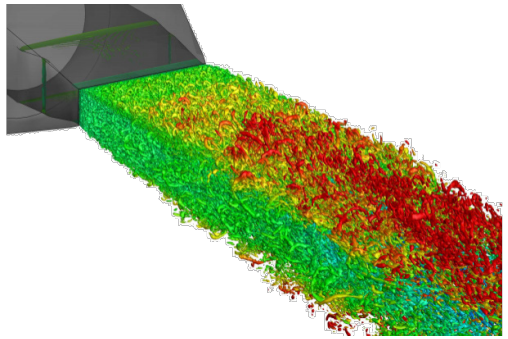
Data-driven implementation (using correlations and impulse response data)

Efficient sensor/actuator placement (enabled by a priori error estimates)

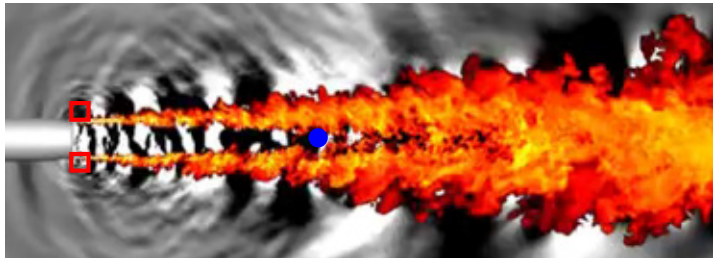
Looking forward

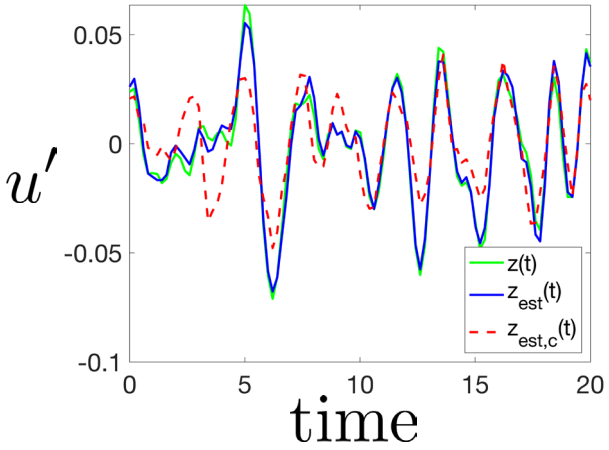
- Resolvent analysis of complex/three-dimensional flows





- Jet noise control





Acknowledgements

University of Michigan
Mechanical Engineering

Rutvij Baghwat



Junoh Jung



Ali Farghadan



Yuhao Zhou



INSTITUT P Université de Poitiers

Peter Jordan



ITA

André Cavalieri



Eduardo Martini



Filipe Amaral



Funding

AFOSR
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH

Office of Naval Research
USN ONR Science & Technology

MICDE

Thank you for your attention

Questions?

towne@umich.edu

Farghadan et al. (2021) A randomized time-domain algorithm for efficiently computing resolvent modes. *AIAA 2021-2896*.

Martini et al. (2021) Efficient computation of global resolvent modes. *J. Fluid Mech.*, 919, A3.

Farghadan et al. (2023) Scalable resolvent analysis for three-dimensional flows. *arXiv:2309.04617*

Towne et al. (2020) Resolvent-based estimation of space-time flow statistics. *J. Fluid Mech.*, 883, A17.

Amaral et al. (2021) Resolvent-based estimation of turbulent channel flow using wall measurements. *J. Fluid Mech.*, 927, A17.

Martini et al. (2022) Resolvent-based tools for optimal estimation and control via the Wiener–Hopf formalism. *J. Fluid Mech.*, 937, A19.

Jung et al. (2023) Optimal resolvent-based estimation and control of flow over a NACA0012 airfoil. *AIAA SciTech 2023*.