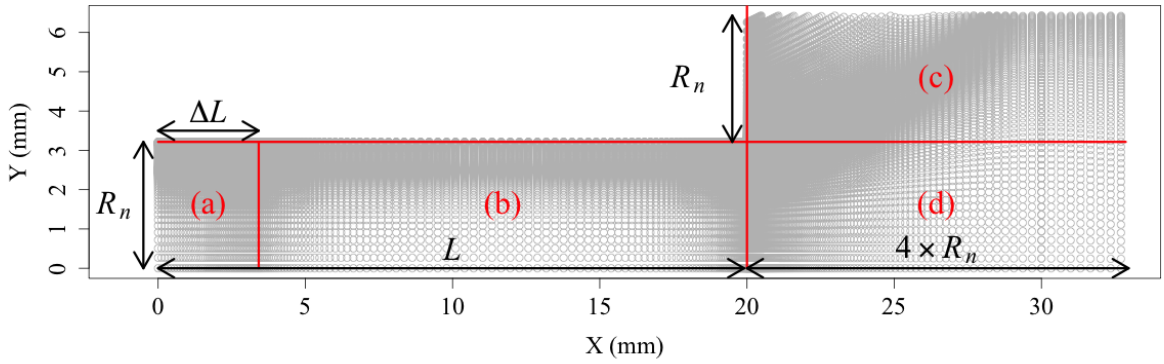

Figure A.1: *Partition of the spatial grid for the first simulation case.*

Appendices

A Computing the CPOD expansion

The driving idea behind CPOD is that a common spatial domain is needed to extract common instabilities over multiple injector geometries, since each simulation run has different geometries and varying grid points. We first describe a physically justifiable method for obtaining such a common domain, and then use this to compute the CPOD expansion.

A.1 Common grid

1. Identify the densest grid (i.e., with the most grid points) among the n simulation runs, and set this as the common reference grid.

2. For each simulation, partition the grid into the following four parts: (a) from injector head-end to the inlet, (b) from the inlet to the nozzle exit, (c) the top portion of the downstream region and (d) the bottom portion of the downstream region (see Figure A.1 for an illustration). This splits the flow in such a way that the linearity assumption can be physically justified.

3. Linearly rescale each part of the partition to the common grid by the corresponding geometry parameters L, R_n and ΔL (see Figure A.1).

4. For each simulation, interpolate the original flow data onto the spatial grid of the common geometry. This step ensures the flow is realized over a common set of grid points for all n simulations. In our implementation, the *inverse distance weighting* interpolation method (Shepard, 1968) is used with 10 nearest neighbours.

A.2 POD expansion

After flows from each simulation have been rescaled onto the common grid, the original POD expansion can be used to extract common flow instabilities. Let $\{\mathbf{x}_j\}_{j=1}^{J}$ and $\{t_m\}_{m=1}^{T}$ denote the set of common grid points and simulated time-steps, respectively, and let $\tilde{Y}(\mathbf{x}, t; \mathbf{c}_i)$ be an interpolated flow variable for geometric setting $\mathbf{c}_i$, $i = 1, \cdots, n$ (for brevity, assume a single flow variable, e.g., x-velocity, for the exposition below). The CPOD expansion can be computed using the following three steps.

1. For notational convenience, we combine all combinations of geometries and time-steps into a single index. Set $N = nT$ and let $l = 1, \cdots, N$ index all combinations of n design settings and T time-steps, and let $\tilde{Y}_l(\mathbf{x}) \equiv \tilde{Y}(\mathbf{x}, (t, \mathbf{c})_l)$. Define $\mathbf{Q} \in \mathbb{R}^{N \times N}$ as the following inner-product matrix:

$$\mathbf{Q}_{l,m} = \sum_{j=1}^{J} \tilde{Y}_l(\mathbf{x}_j)\tilde{Y}_m(\mathbf{x}_j).$$

   Such an inner-product is possible because all n simulated flows are observed on a set of *common* gridpoints set.

   First, compute the eigenvectors $\mathbf{a}_k \in \mathbb{R}^N$ satisfying:

$$\mathbf{Q}\mathbf{a}_k = \lambda_k \mathbf{a}_k,$$

where λ_k is the k-th largest eigenvalue of $\mathbf{Q}$. Since a full eigendecomposition requires $O(N^3)$ work, this step may be intractible to perform when the temporal resolution is dense. To this end, we employed a variant of the implicitly restarted Arnoldi method (Lehoucq et al., 1998), which can efficiently approximate leading eigenvalues and eigenvectors.

2. Compute the k-th mode $\phi_k(\mathbf{x})$ as:

$$\begin{bmatrix} \phi_k(\mathbf{x}_1) \\ \phi_k(\mathbf{x}_2) \\ \vdots \\ \phi_k(\mathbf{x}_J) \end{bmatrix} = \begin{pmatrix} \tilde{Y}_1(\mathbf{x}_1) & \cdots & \tilde{Y}_N(\mathbf{x}_1) \\ \vdots & \ddots & \vdots \\ \tilde{Y}_1(\mathbf{x}_J) & \cdots & \tilde{Y}_N(\mathbf{x}_J) \end{pmatrix} \mathbf{a}_k.$$

To ensure orthonormality, apply the following normalization:

$$\phi_k(\mathbf{x}_j) := \frac{\phi_k(\mathbf{x}_j)}{\|\phi_k(\mathbf{x})\|}, \quad \|\phi_k(\mathbf{x})\| = \sqrt{\sum_{j=1}^{J} \phi_k(\mathbf{x}_j)^2}$$

3. Lastly, derive the CPOD coefficients $(\beta_{l,1}, \cdots, \beta_{l,N})^T$ for the snapshot at index l (i.e., with design setting and time-step $(\mathbf{c}, t)_l$) as:

$$\begin{bmatrix} \beta_{l,1} \\ \beta_{l,2} \\ \vdots \\ \beta_{l,N} \end{bmatrix} = \begin{pmatrix} \phi_1(\mathbf{x}_1) & \cdots & \phi_1(\mathbf{x}_J) \\ \vdots & \ddots & \vdots \\ \phi_N(\mathbf{x}_1) & \cdots & \phi_N(\mathbf{x}_J) \end{pmatrix} \begin{bmatrix} \tilde{Y}_l(\mathbf{x}_1) \\ \tilde{Y}_l(\mathbf{x}_2) \\ \vdots \\ \tilde{Y}_l(\mathbf{x}_J) \end{bmatrix}.$$

Using these coefficients and a truncation at $K_r < N$ modes, it is easy to show the following decomposition of the flow at the design setting $\mathbf{c}_i$ and time-step t_m indexed

by l:

$$Y(\mathbf{x}_j, t_m; \mathbf{c}_i) \approx \sum_{k=1}^{K_r} \beta_{l,k} \mathcal{M}_i\{\phi_k(\mathbf{x}_j)\}, \quad j = 1, \cdots, J,$$

as asserted in (3).

B Proof of Theorem 2

Define the map $A : \mathbb{R}^K \times \mathbb{R}^{K\times K} \times \mathbb{R}^p \to \mathbb{R}^K \times \mathbb{R}^{K\times K} \times \mathbb{R}^p$ as a single-loop of the graphical LASSO operator for optimizing $\mathbf{T}$ with $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ fixed, and define $B : \mathbb{R}^K \times \mathbb{R}^{K\times K} \times \mathbb{R}^p \to \mathbb{R}^K \times \mathbb{R}^{K\times K} \times \mathbb{R}^p$ as the L-BFGS map for a single line-search when optimizing $\boldsymbol{\mu}$ and $\boldsymbol{\tau}$ with $\mathbf{T}$ fixed. Each BCD cycle in Algorithm 1 then follows the map composition $S = A^M \circ B^N$, where $M < \infty$ and $N < \infty$ are the iteration count for the graphical LASSO operator and number of line-searches, respectively. The parameter estimates at iteration m of the BCD cycle can then be given by:

$$\Theta_{m+1} = S(\Theta_m), \quad \text{where } \Theta_m = (\boldsymbol{\mu}_m, \mathbf{T}_m, \boldsymbol{\tau}_m).$$

Define the set of stationary solutions as $\Gamma = \{\Theta \ : \ \nabla l_\lambda(\Theta) = \mathbf{0}\}$, where ∇l_λ is the gradient of the negative log-likelihood l_λ. Using the Global Convergence Theorem (see Section 7.7 of Luenberger and Ye, 2008), we can prove stationary convergence:

$$\lim_{m\to\infty} \Theta_m = \Theta^* \in \Gamma,$$

if the following three conditions hold:

(i) $\{\Theta_m\}_{m=1}^{\infty}$ is contained within a compact subset of $\mathbb{R}^K \times \mathbb{R}^{K\times K} \times \mathbb{R}^p$,

(ii) l_λ is a continuous descent function on Γ under map S,

(iii) S is closed for points outside of Γ.

We will verify these conditions below.

(i) This is easily verified by the fact that $|\boldsymbol{\mu}_m| \leq \left(\max_{i,r,k} |\beta_k^{(r)}(\mathbf{c}_i)|\right) \mathbf{1}_K$, $\mathbf{0} \preceq \mathbf{T}_m \preceq \left(\max_{k,r} s^2\{\beta_k^{(r)}(\mathbf{c}_i)\}_{i=1}^n\right) \mathbf{I}_K$ and $\boldsymbol{\tau}_m \in [0,1]^p$, where $s^2\{\cdot\}$ returns the sample standard deviation for a set of scalars.

(ii) To prove that S is a descent function, we need to show that if $\Theta \in \Gamma$, then $l_\lambda\{S(\Theta)\} = l_\lambda\{\Theta\}$, and if $\Theta \notin \Gamma$, then $l_\lambda\{S(\Theta)\} < l_\lambda\{\Theta\}$. The first condition is trivial, since $M = 0$ and $N = 0$ when Θ is stationary. The second condition follows from the fact that the maps A and B incur a strict decrease in l_λ whenever $\mathbf{T}$ and $(\boldsymbol{\mu}, \boldsymbol{\tau})$ are non-stationary, respectively.

(iii) Note that A^M is a continuous map (since the graphical LASSO map is a continuous operator) and the line-search map B^N is also continuous. Since $S = A^M \circ B^N$, it must be continuous as well, from which the closedness of S follows.

C Proof of Theorem 3

Fix some spatial coordinate $\mathbf{x}$ and time-step t, and let:

$$\mathbf{y} = (Y^{(u)}(\mathbf{x}, t; \mathbf{c}_{new}), Y^{(v)}(\mathbf{x}, t; \mathbf{c}_{new}), Y^{(w)}(\mathbf{x}, t; \mathbf{c}_{new}))^T$$

be the true simulated flows for x-, y- and circumferential velocities at the new setting $\mathbf{c}_{new}$,

$$\hat{\mathbf{y}} = (\hat{Y}^{(u)}(\mathbf{x}, t; \mathbf{c}_{new}), \hat{Y}^{(v)}(\mathbf{x}, t; \mathbf{c}_{new}), \hat{Y}^{(w)}(\mathbf{x}, t; \mathbf{c}_{new}))^T$$

be its corresponding prediction from (9), and

$$\bar{\mathbf{y}} = (\bar{Y}^{(u)}(\mathbf{x}; \mathbf{c}_{new}), \bar{Y}^{(v)}(\mathbf{x}; \mathbf{c}_{new}), \bar{Y}^{(w)}(\mathbf{x}; \mathbf{c}_{new}))^T$$

be its time-averaged flow. It is easy to verify that, given the simulation data $\mathcal{D} = \{Y^{(r)}(\mathbf{x}, t; \mathbf{c}_i)\}$, the conditional distribution of $\mathbf{y}|\mathcal{D}$ is $\mathcal{N}(\hat{\mathbf{y}}, \Phi(\mathbf{x}, t))$, where:

$$\Phi(\mathbf{x}, t) \equiv \begin{bmatrix} \mathbf{m}^{(u)} & 0 & 0 \\ 0 & \mathbf{m}^{(v)} & 0 \\ 0 & 0 & \mathbf{m}^{(w)} \end{bmatrix} [\mathbb{V}\{\boldsymbol{\beta}(t; \mathbf{c}_{new})|\{\boldsymbol{\beta}(t; \mathbf{c}_i)\}_{i=1}^n\}]_{uvw} \begin{bmatrix} \mathbf{m}^{(u)} & 0 & 0 \\ 0 & \mathbf{m}^{(v)} & 0 \\ 0 & 0 & \mathbf{m}^{(w)} \end{bmatrix}^T, \tag{C.1}$$

with:

$$\mathbf{m}^{(r)} = \left[\begin{array}{cccc} \mathcal{M}_{new}\{\phi_1^{(r)}(\mathbf{x})\}, & \mathcal{M}_{new}\{\phi_2^{(r)}(\mathbf{x})\}, & \cdots & \mathcal{M}_{new}\{\phi_{K_r}^{(r)}(\mathbf{x})\} \end{array} \right], \quad r = u, v, w.$$

Letting $\Phi(t) = \mathbf{U}\Lambda\mathbf{U}^T$ be the eigendecomposition of $\Phi(t)$, with $\Lambda = \text{diag}\{\lambda_j\}$, it follows that $\Lambda^{-1/2}\mathbf{U}^T(\mathbf{y} - \bar{\mathbf{y}})|\mathcal{D} \stackrel{d}{=} \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_K)$, where $\boldsymbol{\mu} = \Lambda^{-1/2}\mathbf{U}^T(\hat{\mathbf{y}} - \bar{\mathbf{y}})$ and $K = K_u + K_v + K_w$. Denoting $\mathbf{a} = \Lambda^{-1/2}\mathbf{U}^T(\mathbf{y} - \bar{\mathbf{y}})$, the TKE expression in (13) can be rewritten as:

$$\begin{aligned} \kappa(\mathbf{x}, t) &= \frac{1}{2}(\mathbf{y} - \bar{\mathbf{y}})^T(\mathbf{y} - \bar{\mathbf{y}}) = \frac{1}{2}(\mathbf{U}\Lambda^{1/2}\mathbf{a})^T(\mathbf{U}\Lambda^{1/2}\mathbf{a}) \\ &= \frac{1}{2}(\mathbf{a}^T\Lambda^{1/2}\mathbf{U}^T\mathbf{U}\Lambda^{1/2}\mathbf{a}) \\ &= \frac{1}{2}\mathbf{a}^T\Lambda\mathbf{a} = \frac{1}{2}\sum_{j=1}^K \lambda_j a_j^2. \end{aligned} \tag{C.2}$$

Since $\mathbf{a} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{I}_K)$, a_j^2 has a non-central chi-square distribution with one degree-of-freedom and non-centrality parameter μ_j^2 (we denote this as $\chi_1^2(\mu_j^2)$). $\kappa(\mathbf{x}, t)$ then becomes:

$$\sum_{j=1}^K \frac{\lambda_j}{2}\chi_1^2(\mu_j^2), \tag{C.3}$$

which is a sum of weighted non-central chi-squared distributions. The computation of the distribution function for such a random variable has been studied extensively, see, e.g., Imhof (1961), Davies (1973, 1980), Castaño-Martínez and López-Blázquez (2005), and

Liu et al. (2009), and we appeal to these methods for computing the pointwise confidence interval of $\kappa(\mathbf{x}, t)$ in Section 4. Specifically, we employ the method of Liu et al. (2009) through the R (R Core Team, 2015) package `CompQuadForm` (Duchesne and de Micheaux, 2010).

References

Castaño-Martínez, A. and López-Blázquez, F. (2005). Distribution of a sum of weighted noncentral chi-square variables. *TEST*, 14(2):397–415.

Davies, R. B. (1973). Numerical inversion of a characteristic function. *Biometrika*, 60(2):415–417.

Davies, R. B. (1980). Algorithm AS 155: The distribution of a linear combination of χ^2 random variables. *Journal of the Royal Statistical Society. Series C*, 29(3):323–333.

Duchesne, P. and de Micheaux, P. L. (2010). Computing the distribution of quadratic forms: Further comparisons between the Liu-Tang-Zhang approximation and exact methods. *Computational Statistics and Data Analysis*, 54(4):858–862.

Imhof, J. P. (1961). Computing the distribution of quadratic forms in normal variables. *Biometrika*, 48(3/4):419–426.

Lehoucq, R. B., Sorensen, D. C., and Yang, C. (1998). *ARPACK Users' Guide: Solution of Large-Scale Eigenvalue Problems with Implicitly Restarted Arnoldi Methods*, volume 6. SIAM, Philadelphia.

Liu, H., Tang, Y., and Zhang, H. H. (2009). A new chi-square approximation to the distribution of non-negative definite quadratic forms in non-central normal variables. *Computational Statistics and Data Analysis*, 53(4):853–856.

Luenberger, D. G. and Ye, Y. (2008). *Linear and Nonlinear Programming*, volume 116. Springer, US.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Shepard, D. (1968). A two-dimensional interpolation function for irregularly-spaced data. *Proceedings of the 23rd ACM National Conference*. 517–524.