Notes on POD

Chenyu Zhang

May 20, 2021

1 Covariance POD

1.1 A simple 1D case

Suppose that there are two points a,b in the system, and we are interested in the correlations of velocities at these two points. We can construct a snapshot of matrix:

$$S = \begin{bmatrix} U_a(t_1) & U_b(t_1) \\ U_a(t_2) & U_b(t_2) \\ \vdots & \vdots \\ U_a(t_m) & U_b(t_m) \end{bmatrix} \tag{1}$$

Remove the temporal average of the two velocities $\bar{U}_a$ and $\bar{U}_b$, defining the fluctuation $u'_a(t) = U_a(t) - \bar{U}_a$ and $u'_b(t) = U_b(t) - \bar{U}_b$, we have a new matrix U

$$U = \begin{bmatrix} u'_a(t_1) & u'_b(t_1) \\ u'_a(t_2) & u'_b(t_2) \\ \vdots & \vdots \\ u'_a(t_m) & u'_b(t_m) \end{bmatrix} \equiv \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ \vdots & \vdots \\ u_{m1} & u_{m2} \end{bmatrix} \tag{2}$$

We want to know if there is any correlation between the velocities at the two points, so we calculate the covariance matrix C:

$$C = \frac{1}{m-1} U^T U = \frac{1}{m-1} \begin{bmatrix} \sum_{i=1}^m u_a'^2(t_i) & \sum_{i=1}^m u'_a(t_i) u'_b(t_i) \\ \sum_{i=1}^m u'_b(t_i) u'_a(t_i) & \sum_{i=1}^m u_b'^2(t_i) \end{bmatrix} \tag{3}$$

If the covariance matrix C is diagonal, then it means that u'_a and u'_b are perfectly uncorrelated. If C is not diagonal, it means that there is correlation between the two velocities. It would be nice to come up with velocity modes that are decoupled. This means that we want to diagonalize the covariance matrix and find the corresponding basis. The basis axes are called *the principle axes or the proper orthogonal modes* of the dataset. Formally, the covariance matrix C is diagonalized as

$$C = \Phi \lambda \Phi^T = \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \phi_{11} & \phi_{21} \\ \phi_{12} & \phi_{22} \end{bmatrix} \tag{4}$$

Keep in mind that because the matrix C is symmetric, the eigenvalues should be real. Also, a covariance matrix is always semi positive definite, which means that the real eigenvalues should always be non-negative. To prove this, first notice that

$$C = \frac{1}{m-1} U^T U \tag{5}$$

So for a nonzero vector $y \in \mathbb{R}^k$, we have

$$\begin{aligned} y^T C y &= y^T (\frac{1}{m-1} U^T U) y \\ &= \frac{1}{m-1} (y^T U^T U y) \\ &= \frac{1}{m-1} (Uy)^2 \geq 0 \end{aligned} \tag{6}$$

Therefore, the covariance matrix is always semi positive definite. We can project the data U onto these eigenvectors to compute the variance along each principle axis:

$$A = U\Phi = \begin{bmatrix} u_{11} & u_{12} \\ u_{21} & u_{22} \\ \vdots & \vdots \\ u_{m1} & u_{m2} \end{bmatrix} \begin{bmatrix} \phi_{11} & \phi_{12} \\ \phi_{21} & \phi_{22} \end{bmatrix} \tag{7}$$

The hope is that each eigenmode can be linked to an independent coherent structure responsible for the velocity fluctuations. Since $A = U\Phi$ by equation (5), $U = A\Phi^{-1} = A\Phi^T$. So

$$U = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ \vdots & \vdots \\ a_{m1} & a_{m2} \end{bmatrix} \begin{bmatrix} \phi_{11} & \phi_{21} \\ \phi_{12} & \phi_{22} \end{bmatrix} = \begin{bmatrix} a_{11} \\ a_{21} \\ \vdots \\ a_{m1} \end{bmatrix} \begin{bmatrix} \phi_{11} & \phi_{21} \end{bmatrix} + \begin{bmatrix} a_{12} \\ a_{22} \\ \vdots \\ a_{m2} \end{bmatrix} \begin{bmatrix} \phi_{12} & \phi_{22} \end{bmatrix} = \tilde{U}^1 + \tilde{U}^2 \tag{8}$$

Suppose $\lambda_1 \gg \lambda_2$, then most variance happens along the first principle axis, which means that $\tilde{U}^1$ is playing the dominating role in the fluctuations. In this case, we can neglect $\tilde{U}^2$ and focus on $\tilde{U}^1$. This is called the *dimensionality reduction.*

1.2 N-D case

Suppose the system has a size of $n = N_x \times N_y = 5805$. Suppose there are $m = N_t$ temporal points. Then the new fluctuation matrix U would be of shape $m \times n$.

$$U = \begin{bmatrix} u_{11} & \dots & u_{1n} \\ u_{21} & \dots & u_{2n} \\ \vdots & \ddots & \vdots \\ u_{m1} & \dots & u_{mn} \end{bmatrix} = \begin{bmatrix} u'(x_1, y_1, t_1) & \dots & u'(x_{N_x}, y_{N_y}, t_1) \\ u'(x_1, y_1, t_1) & \dots & u'(x_{N_x}, y_{N_y}, t_2) \\ \vdots & \ddots & \vdots \\ u'(x_1, y_1, t_m) & \dots & u'(x_{N_x}, y_{N_y}, t_m) \end{bmatrix} \tag{9}$$

where each column represents the time evolution at each point, and each row represents the snapshot at each time. The covariance matrix ($n \times n$) can be constructed as U^TU, and the eigenvalues can be written as $\lambda_1, \lambda_2, ..., \lambda_n$, the eigenvectors can be written as:

$$\Phi = \begin{bmatrix} \phi_{11} & \dots & \phi_{1n} \\ \phi_{21} & \dots & \phi_{2n} \\ \vdots & \ddots & \vdots \\ \phi_{n1} & \dots & \phi_{nn} \end{bmatrix} \tag{10}$$

Eigenmodes constructed this way are *spatial modes.* To construct temporal modes, form UU^T. For our project, the matrices will be too large if we don't downsample them. The spatial mode Φ matrix will be of shape and size $96^2 \times 128 \times 512^2 \approx 3.09 \times 10^{11}$, assuming our simulation resolution is $(512, 128, 96)$. If we choose to analyse all four fields in one matrix, it is going to be 16 times bigger (for details, please see the section 1.3). Just like in the 1D example, the matrix U can be decomposed as

$$U = \sum_{k=1}^{n} \tilde{U}^k \tag{11}$$

and based on the eigenvalues, we can choose to only focus on a few of the $\tilde{U}^k$s.

1.3 Procedure

Let components (v_r, v_z, v_θ, P) of the field be denoted as q_i with $i = (v_r, v_z, v_\theta, P)$. Let the number of modes in the (r, θ, z) directions be N_r, N_θ, and N_z, respectively. For now, we stick to a fixed time, t. Then $q_i(r, z, \theta, t)$ is a 4D array of dimension $4 \times N_r \times N_z \times N_\theta$. We average over the azimuthal angle θ. Following Skitka *et al.* define the mean coordinate as $\overline{\theta} \equiv (\theta_1 + \theta_2)/2$ and the deviation from the coordinate to be $\delta\theta \equiv \theta_1 - \theta_2$. Then

$$\begin{aligned} c_{ij}(r_1, z_1, r_2, z_2, \theta_1 - \theta_2) &= \overline{q_i(r_1, \theta_1, z_1, t)\ q_j(r_2, \theta_2, z_2, t)} \\ &= \frac{1}{2\pi} \int_0^{2\pi} d\overline{\theta}\ q_i(r_1, \overline{\theta} + \frac{\delta\theta}{2}, z_1, t)\ q_j(r_2, \overline{\theta} - \frac{\delta\theta}{2}, z_2, t) \end{aligned} \tag{12}$$

With a Fourier transform over the θ direction:

$$\begin{aligned} q_i(r_1, \theta_1, z_1, t) &= \sum_{m_1=0}^{N_\theta} q_{im_1}(r_1, z_1, t) e^{im_1\theta_1} \\ q_j(r_2, \theta_2, z_2, t) &= \sum_{m_2=0}^{N_\theta} q_{jm_2}(r_2, z_2, t) e^{im_2\theta_2} \end{aligned} \tag{13}$$

the two-point correlation becomes:

$$\begin{aligned} c_{ij}(r_1, z_1; r_2, z_2, \theta_1 - \theta_2, t) &= \frac{1}{(2\pi)} \int d\bar{\theta} \sum_{m_1,m_2=0}^{N_\theta} q_{im_1}(r_1, z_1, t)\ q_{jm_2}(r_2, z_2, t) e^{im_1(\bar{\theta}+\frac{\delta\theta}{2})} e^{im_2(\bar{\theta}-\frac{\delta\theta}{2})} \\ &= \sum_{m_1,m_2=0}^{N_\theta} q_{im_1}(r_1, z_1, t)\ q_{jm_2}(r_2, z_2, t)\ \delta_{m_1,-m_2}\ e^{i(m_1-m_2)\frac{\delta\theta}{2}} \\ &= \sum_{m=0}^{N_\theta} q_{im}(r_1, z_1, t)\ q_{jm}(r_2, z_2, t)^* e^{im\delta\theta} \end{aligned} \tag{14}$$

The Fourier transform can be done with np.fft. To assemble equation (14), we can take the m-th slice of each scalar field matrix, denoted as $q_i(m)$, and do

$$\frac{1}{N_\theta} \sum_m q_{im}(r_1, z_1, t)\ q_{jm}(r_2, z_2, t)^* e^{im\delta\theta} \tag{15}$$

The result of the above calculation for each $\delta\theta$ is the column of the covariance matrix that varies over $\delta\theta$.

It is simplest to ignore the sum over m and analyze separately the contributions to the covariance matrix at each wavenumber m.

$$c_{ijm}(r_1, z_1; r_2, z_2, t) = q_{im}(r_1, z_1, t)\ q_{jm}(r_2, z_2, t)^* \tag{16}$$

The result of this will be an array of dimension $(4N_rN_z) \times (4N_rN_z)$ at each time t and for each zonal wavenumber m. The covariance matrix is Hermitian. To see this is true note that

$$\begin{aligned} c_{ijm}(r_1, z_1; r_2, z_2; t) &= q_{im}(r_1, z_1, t)\ q_{jm}(r_2, z_2, t)^* \\ &= c_{jim}(r_2, z_2; r_1, z_1; t)^* \end{aligned} \tag{17}$$

The covariance matrix is also semi-positive definite, so all the eigenvalues must be real and non-negative. This fact can be used as a sanity check on code.

1.4 Numerical implementation

There are basically three steps:

1. Forming the U matrix. This takes $O(mn)$ steps and $O(mn)$ memory.

2. Forming the covariance matrix C. This takes $O(n^2)$ steps and $O(n^2)$ memory, for spatial modes.

3. Finding the eigenvalues and eigenvectors. There are several ways:

   - The QR method. This method is iterative. Before doing the QR decomposition, first take the matrix to Hessenberg form using Amoldi iteration. Since the covariance matrix C is symmetric, the Hessenberg form will be tridiagonal, which makes QR relatively easy. This is an expansive algorithm since the QR decomposition can be $O(n^3)$, and convergence can be slow. With the Hessenberg form, the QR decomposition can be $O(n^2)$.
   - There are many more algorithms. I found this book Numerical Methods for Large Eigenvalue Problems which talks about a lot of algorithms for the eigenvalue problem. I'm also curious about what Altan used for his paper.

- There is a Fortran package called LAPACK, in which there is a method called divide-and-conconqor (DGESDD) that is currently the best method in solving for SVD for dense matrices. The fastest way to implement LAPACK is through Scipy (which is super convinent).

In conclusion, for solving for SVD, we just need to implement LAPACK in Scipy. The time it takes would just be minutes.