Discovering Hidden Physics Behind Transport Dynamics
Supplementary Material

This supplementary material contains proofs for our representation theorems and additional implementation details for `YETI`. Supp. A and Supp. B give the proofs for Theorem 1 and Theorem 2, respectively. Supp. C introduces our advection-diffusion PDE toolkit in `PyTorch` and discusses numerical discretization and stability conditions. Supp. D provides detailed descriptions of our brain advection-diffusion simulation dataset, including how we construct the velocity and diffusion fields, and how we simulate the brain advection-diffusion time-series.

A. Theorem 1: Divergence-free Vector Representation by the Curl of Potentials

For any vector field $\mathbf{V} \in L^p(\Omega)^d$ *on a bounded domain* $\Omega \subset \mathbb{R}^d$ *with smooth boundary* $\partial\Omega$. *If* $\mathbf{V}$ *satisfies* $\nabla \cdot \mathbf{V} = 0$, *there exists a potential* Ψ *in* $L^p(\Omega)^\alpha$ *such that (*$\alpha = 1$ *when* $d = 2$, $\alpha = 3$ *when* $d = 3$*)*

$$\mathbf{V} = \nabla \times \Psi, \quad \Psi \cdot \mathbf{n}\big|_{\partial\Omega} = 0, \Psi \in L^p(\Omega)^\alpha. \tag{19}$$

Conversely, for any $\Psi \in L^p(\Omega)^\alpha$, $\nabla \cdot \mathbf{V} = \nabla \cdot (\nabla \times \Psi) = 0$.
(Here, L^p refers to the space of measurable functions for which the p-th power of the function absolute value is Lebesgue integrable. Specifically, let $1 \leq p < \infty$ and (Ω, Σ, μ) be a measure space. $L^p(\Omega)$ space is the set of all measurable functions whose absolute value raised to the p-th power has a finite integral, i.e., $\|f\|_p \equiv (\int_\Omega |f|^p \, \mathrm{d}\mu)^{1/p} < \infty$.)

Proof. We give a brief proof in this supplementary material to make it self-contained. Please refer to [13, 1, 33, 2] for additional discussions regarding alternative boundary conditions for the potentials and domains with complex geometry.

Definition A.1 (The space of curl of potentials).

$$\mathscr{H}_{curl}(\Omega) \equiv \{\nabla \times \Psi \,\big|\, \Psi \in L^p(\Omega)^\alpha, \Omega \in \mathbb{R}^d\}, \tag{20}$$

where $\alpha = 1$ *when* $d = 2$, $\alpha = 3$ *when* $d = 3$.

Definition A.2 (The space of divergence-free velocity fields).

$$\mathscr{H}_{div}(\Omega) \equiv \{\mathbf{V} \in L^p(\Omega)^d \,\big|\, \nabla \cdot \mathbf{V} = 0, \Omega \in \mathbb{R}^d\}, \tag{21}$$

where $\alpha = 1$ *when* $d = 2$, $\alpha = 3$ *when* $d = 3$.

- Clearly, for $\forall \mathbf{V} \in \mathscr{H}_{\text{curl}}(\Omega)$, $\nabla \cdot \mathbf{V} = 0$.

  Specifically, when $d = 3$, the curl of a vector field $\Psi = \Psi_x \mathbf{i} + \Psi_y \mathbf{j} + \Psi_z \mathbf{k}$ is computed by

$$\begin{aligned} \nabla \times \Psi &= \begin{vmatrix} \mathbf{i} & \mathbf{j} & \mathbf{k} \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ \Psi_x & \Psi_y & \Psi_z \end{vmatrix} = \left(\frac{\partial \Psi_z}{\partial y} - \frac{\partial \Psi_y}{\partial z}\right)\mathbf{i} + \left(\frac{\partial \Psi_x}{\partial z} - \frac{\partial \Psi_z}{\partial x}\right)\mathbf{j} + \left(\frac{\partial \Psi_y}{\partial x} - \frac{\partial \Psi_x}{\partial y}\right)\mathbf{k} \\ &= \left[\frac{\partial \Psi_z}{\partial y} - \frac{\partial \Psi_y}{\partial z}, \frac{\partial \Psi_x}{\partial z} - \frac{\partial \Psi_z}{\partial x}, \frac{\partial \Psi_y}{\partial x} - \frac{\partial \Psi_x}{\partial y}\right]^T. \end{aligned} \tag{22}$$

  Therefore,

$$\nabla \cdot (\nabla \times \Psi) = \frac{\partial}{\partial x}\left(\frac{\partial \Psi_z}{\partial y} - \frac{\partial \Psi_y}{\partial z}\right) + \frac{\partial}{\partial y}\left(\frac{\partial \Psi_x}{\partial z} - \frac{\partial \Psi_z}{\partial x}\right) + \frac{\partial}{\partial z}\left(\frac{\partial \Psi_y}{\partial x} - \frac{\partial \Psi_x}{\partial y}\right) = 0. \tag{23}$$

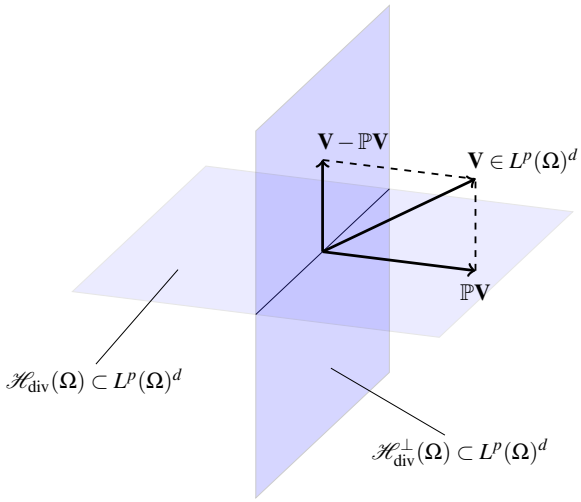

Figure A.10: Decompose a vector into divergence-free part and its orthogonal complement.

When $d=2$, we can express Ψ as $\Psi=\Psi_x\mathbf{i}+\Psi_y\mathbf{j}\,(+0\,\mathbf{k})$. Likewise,

$$\nabla\times\Psi=\begin{vmatrix}\mathbf{i} & \mathbf{j} & \mathbf{k}\\ \dfrac{\partial}{\partial x} & \dfrac{\partial}{\partial y} & \dfrac{\partial}{\partial z}\\ \Psi_x & \Psi_y & 0\end{vmatrix}=\left[0,\,0,\,\frac{\partial\Psi_y}{\partial x}-\frac{\partial\Psi_x}{\partial y}\right]^T, \tag{24}$$

as such,

$$\nabla\cdot(\nabla\times\Psi)=0+0+\frac{\partial}{\partial z}\left(\frac{\partial\Psi_y}{\partial x}-\frac{\partial\Psi_x}{\partial y}\right)=0. \tag{25}$$

- Starting from sufficiently smooth vectors in $L^p(\Omega)^d$, the Helmholtz-Hodge decomposition theorem [33] ensures the existence of its divergence-free part.

Theorem A.1 (Helmholtz-Hodge Decomposition (HHD)). *A sufficiently smooth vector field $\mathbf{V}\in L^p(\Omega)^d$ on a domain $\Omega\subset\mathbb{R}^d$ with smooth boundary $\partial\Omega$, can be uniquely decomposed into a pure gradient field and a divergence-free vector field*

$$\mathbf{V}=\nabla p+\mathbf{u},\quad p\in L^p(\Omega),\,\mathbf{u}\in\mathscr{H}_{div}(\Omega). \tag{26}$$

Theorem A.1 can be interpreted as a mapping $\mathbb{P}$, which uniquely projects the smooth vector field $\mathbf{V}$ to its divergence-free part $\mathbf{u}$ by $\mathbb{P}\mathbf{V}=\mathbf{u}$, and representing $\mathbf{V}$ as $\mathbf{u}$ plus its orthogonal complement (∇p) (Fig. A.10). Therefore, the following conclusion is natural:

Corollary A.1. *$\mathbb{P}$ is an identity mapping on $H_{div}(\Omega)$. I.e, for $\forall\mathbf{V}\in\mathscr{H}_{div}(\Omega)$, $\mathbb{P}\mathbf{V}=\mathbf{V}$.*

Further, for the special cases $\Omega\in\mathbb{R}^d$, $d=2,3$, $\mathscr{H}_{\text{div}}(\Omega)$ is equivalent to the space of the curl of potentials with vanishing boundary condition [13]:

$$\mathscr{H}_{\text{div}}(\Omega)\equiv\left\{\Psi\in\mathscr{H}_{\text{curl}}(\Omega)\,\middle|\,\Psi\cdot\mathbf{n}\big|_{\partial\Omega}=0,\,\Omega\in\mathbb{R}^d\right\}. \tag{27}$$

Combined with Corollary A.1, we have for $\forall\mathbf{V}\in L^p(\Omega)^d$, if $\mathbf{V}$ is divergence-free, there exists a potential $\Psi\in\mathscr{H}_{\text{curl}}$ with vanishing boundary condition, which satisfies $\mathbf{V}=\nabla\times\Psi$.

□

B. Theorem 2: Symmetric PSD Tensor Representation by Spectral Decomposition

For any tensor $\mathbf{D} \in PSD(n)$*, there exists an upper triangular matrix with zero diagonal entries,* $\mathbf{B} \in \mathbb{R}^{\frac{n(n-1)}{2}}$*, and a diagonal matrix with non-negative diagonal entries,* $\Lambda \in SD(n)$*, satisfying:*

$$\mathbf{D} = \mathbf{U}\Lambda\mathbf{U}^T, \quad \mathbf{U} = exp(\mathbf{B} - \mathbf{B}^T) \in SO(n). \tag{28}$$

Conversely, for any upper triangular matrix with zero diagonal entries, $\mathbf{B} \in \mathbb{R}^{\frac{n(n-1)}{2}}$*, and any diagonal matrix with non-negative diagonal entries,* $\Lambda \in SD(n)$*, Eq. (28) computes a symmetric PSD tensor,* $\mathbf{D} \in PSD(n)$*.*

Proof. First of all, let us define the following three special groups of interest:

Definition B.1 (The $n \times n$ symmetric PSD tensor group).

$$PSD(n) \equiv \{\mathbf{D} \in \mathbb{R}^{n\times n} \,\big|\, \forall \mathbf{x} \in \mathbb{R}^n : \mathbf{x}^T\mathbf{D}\mathbf{x} \geq 0\}. \tag{29}$$

Definition B.2 (The special group of real orthogonal matrices).

$$SO(n) \equiv \{\mathbf{U} \in \mathbb{R}^{n\times n} \,\big|\, \mathbf{U}^T\mathbf{U} = \mathbf{I}, det(\mathbf{U}) = 1\}. \tag{30}$$

Definition B.3 (The special group of real diagonal matrices with all non-negative entries).

$$SD(n) \equiv \{diag(\lambda_1, ...\lambda_n) \in \mathbb{R}^{n\times n} \,\big|\, \lambda_1, ..., \lambda_n \geq 0\}. \tag{31}$$

It is natural to consider the spectral decomposition form for a tensor field $\mathbf{D} \in PSD(n)$:

Theorem B.1 (Spectral Decomposition for Symmetric Positive Semi-definite Matrix). *Let* $\mathbf{D}$ *be a* $n \times n$ *real symmetric matrix, which is positive semi-definite. Then* $\mathbf{D}$ *can be factorized as:*

$$\mathbf{D} = \mathbf{U}\Lambda\mathbf{U}^T, \quad \mathbf{U} \in SO(n), \Lambda \in SD(n), \tag{32}$$

where columns of $\mathbf{U}$ *are the eigenvectors of* $\mathbf{D}$*, the diagonal entries of* Λ *are the corresponding non-negative eigenvalues.*

As such, any symmetric PSD tensor can be represented via its orthogonal eigenvectors and corresponding eigenvalues. Intuitively, we can represent a tensor $\mathbf{D} \in PSD(n)$ by representing its eigenvectors $\mathbf{U} \in SO(n)$ and eigenvalues $\Lambda \in SD(n)$.

- It is straightforward to represent a diagonal matrix $\Lambda \in SD(n)$ with non-negative entries by simply imposing non-negativity in the implementation (For example via a `ReLU`). Representing an orthogonal matrix $\mathbf{U}$ needs more parametrization tricks.

  In order to obtain an orthogonal matrix $\mathbf{U}$ by construction, we resort to the surjective Lie exponential mapping on $SO(n)$, which is in fact a compact and connected Lie group [27]. When seen as a sub-manifold of $\mathbb{R}^{n\times n}$ equipped with the metric induced from the ambient space $\langle \mathbf{X}, \mathbf{Y} \rangle = tr(\mathbf{X}^T\mathbf{Y})$, $SO(n)$ inherits a bi-invariant metric (i.e., the metric is invariant with respect to left and right multiplication by matrices of the group). The tangent space at the identity element of $SO(n)$ is a Lie algebra of $SO(n)$, which is essentially the group of skew-symmetric matrices,

  $$\mathfrak{so}(n) = \{\mathbf{A} \in \mathbb{R}^{n\times n} \,\big|\, \mathbf{A} + \mathbf{A}^T = 0\}. \tag{33}$$

  The Lie exponential map is a map from the Lie algebra $\mathfrak{g}$ of a Lie group G to the group itself, which is an important tool for studying local structure of Lie groups. Specifically, it is written as:

  $$\begin{aligned} exp : \mathfrak{g} &\to G \\ \mathbf{A} &\mapsto exp(\mathbf{A}) := \mathbf{I} + \mathbf{A} + \frac{1}{2}\mathbf{A}^2 + \cdots. \end{aligned} \tag{34}$$

  For the special group $SO(n)$, which is connected and compact [27], the Lie exponential map from $\mathfrak{so}(n)$ to $SO(n)$ is surjective, which means $\forall \mathbf{U} \in SO(n)$, $\exists \mathbf{A} \in \mathfrak{so}(n), exp(\mathbf{A}) = \mathbf{U}$. Follow [27], an optimization problem from $SO(n)$ is equivalent to its corresponding optimization problem in Euclidean space, via Lie exponential mapping. In other words,

  $$\min_{\mathbf{U}\in SO(n)} f(\mathbf{U}) \quad \Longleftrightarrow \quad \min_{\mathbf{A}\in\mathfrak{so}(n)} f(exp(\mathbf{A})). \tag{35}$$

Furthermore, combining with the isomorphic mapping from vector space to $\mathfrak{so}(n)$ [26],

$$\begin{aligned} \alpha : \mathbb{R}^{\frac{n(n-1)}{2}} &\to \mathfrak{so}(n) \\ \mathbf{B} &\mapsto \mathbf{B} - \mathbf{B}^T, \end{aligned} \tag{36}$$

where $\mathbb{R}^{\frac{n(n-1)}{2}}$ refers to the space of upper triangular matrices with zero diagonal entries, we reach the following equivalent optimization problem on the Euclidean space $\mathbb{R}^{\frac{n(n-1)}{2}}$.

$$\min_{\mathbf{U}\in SO(n)} f(\mathbf{U}) \Leftrightarrow \min_{\mathbf{A}\in\mathfrak{so}(n)} f(exp(\mathbf{A})) \Leftrightarrow \min_{\mathbf{B}\in\mathbb{R}^{\frac{n(n-1)}{2}}} f(exp(\mathbf{B}-\mathbf{B}^T)). \tag{37}$$

In implementation, solving the exponential mapping is computationally expensive, especially for large scale matrices. Therefore, we use the *Cayley* retraction, a first order approximation for the exponential mapping [26]:

$$exp(\mathbf{A}) \approx \phi(\mathbf{A}) = (\mathbf{I} + \frac{1}{2}\mathbf{A})(\mathbf{I} - \frac{1}{2}\mathbf{A})^{-1}. \tag{38}$$

Therefore, given any tensor $\mathbf{D} \in PSD(n)$, according to the existence of its spectral decomposition via Eq. (32) and the two surjective mappings of Eqs. (34, 36), there exists an upper triangular matrix with zero diagonal entries, $\mathbf{B} \in \mathbb{R}^{\frac{n(n-1)}{2}}$, and a diagonal matrix with non-negative diagonal entries, $\Lambda \in SD(n)$, that satisfy Eq. (28).

- Conversely, given any upper triangular matrix with zero diagonal entries, $\mathbf{B} \in \mathbb{R}^{\frac{n(n-1)}{2}}$, and any diagonal matrix with non-negative diagonal entries, $\Lambda \in SD(n)$, Eq. (28) computes a symmetric PSD tensor, $\mathbf{D} \in PSD(n)$. Here, we give brief illustrations for the cases of $n = 2, 3$, corresponding to diffusion tensors on 2D and 3D domains in our work respectively.

  – *2D tensors*. Denote $\mathbf{B} = \begin{bmatrix} 0 & s \\ 0 & 0 \end{bmatrix} \in \mathbb{R}^{\frac{2(2-1)}{2}}$, applying Eq. (36) we have

  $$\mathbf{B} \mapsto \mathbf{A} := \alpha(\mathbf{B}) = \begin{bmatrix} 0 & s \\ -s & 0 \end{bmatrix} \in \mathfrak{so}(2), \tag{39}$$

  applying Cayley retraction in Eq. (38) we have

  $$\mathbf{A} \mapsto \mathbf{U} := exp(\mathbf{A}) \approx \begin{bmatrix} 1 & \frac{s}{2} \\ -\frac{s}{2} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{s}{2} \\ \frac{s}{2} & 1 \end{bmatrix}^{-1} = \frac{1}{s^2+4} \begin{bmatrix} 4-s^2 & 4s \\ -4s & 4-s^2 \end{bmatrix} \in SO(2), \tag{40}$$

  Combining with any given diagonal matrix with non-negative diagonal entries, $\Lambda \in SD(2)$, we obtain $\mathbf{D} = \mathbf{U}\Lambda\mathbf{U}^T \in PSD(2)$ by definition.

  – *3D tensors*. Denote $\mathbf{B} = \begin{bmatrix} 0 & s_1 & s_2 \\ 0 & 0 & s_3 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{\frac{3(3-1)}{2}}$, applying Eq. (36) we have

  $$\mathbf{B} \mapsto \mathbf{A} := \alpha(\mathbf{B}) = \begin{bmatrix} 0 & s_1 & s_2 \\ -s_1 & 0 & s_3 \\ -s_2 & -s_3 & 0 \end{bmatrix} \in \mathfrak{so}(3), \tag{41}$$

  applying Cayley retraction in Eq. (38) we have

  $$\begin{aligned} \mathbf{A} \mapsto \mathbf{U} :&= exp(\mathbf{A}) \approx \begin{bmatrix} 1 & \frac{s_1}{2} & \frac{s_2}{2} \\ -\frac{s_1}{2} & 1 & \frac{s_3}{2} \\ -\frac{s_2}{2} & -\frac{s_3}{2} & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{s_1}{2} & -\frac{s_2}{2} \\ \frac{s_1}{2} & 1 & -\frac{s_3}{2} \\ \frac{s_2}{2} & \frac{s_3}{2} & 1 \end{bmatrix}^{-1} \\ &= \frac{1}{s_1^2+s_2^2+s_3^2+4} \begin{bmatrix} s_1^2+s_2^2+4 & s_2s_3 & -s_1s_3 \\ s_2s_3 & s_1^2+s_3^2+4 & s_1s_2 \\ -s_1s_3 & s_1s_2 & s_2^2+s_3^2+4 \end{bmatrix} \in SO(3), \end{aligned} \tag{42}$$

  Combining with any given diagonal matrix with non-negative diagonal entries, $\Lambda \in SD(3)$, we obtain $\mathbf{D} = \mathbf{U}\Lambda\mathbf{U}^T \in PSD(3)$ by definition.

□

C. PyTorch Advection-Diffusion PDE Toolkit: Brief Manual on Numerical Derivations

Our advection-diffusion PDE toolkit is designed to solve, separately or jointly, advection and diffusion PDEs (1/2/3D).

$$\frac{\partial C(\mathbf{x},t)}{\partial t} = \left.\frac{\partial C(\mathbf{x},t)}{\partial t}\right|_{\text{adv}} + \left.\frac{\partial C(\mathbf{x},t)}{\partial t}\right|_{\text{diff}} = \underbrace{-\nabla\left(\mathbf{V}(\mathbf{x})\cdot C(\mathbf{x},t)\right)}_{\text{Fluid flow}} + \underbrace{\nabla\cdot\left(\mathbf{D}(\mathbf{x})\nabla C(\mathbf{x},t)\right)}_{\text{Diffusion}}$$
$$(\nabla\cdot\mathbf{V}=0) = \underbrace{-\mathbf{V}(\mathbf{x})\cdot\nabla C(\mathbf{x},t)}_{\text{Incompressible flow}} + \underbrace{\nabla\cdot\left(\mathbf{D}(\mathbf{x})\nabla C(\mathbf{x},t)\right)}_{\text{Diffusion}}. \tag{43}$$

One can choose to model the velocity field as a constant, a vector field, or a divergence-free vector field (for impressible flow). Furthermore, the diffusion field can be modeled as a constant, a non-negative scalar field, or a symmetric positive semi-definite (PSD) tensor field. The toolkit is implemented as a custom `torch.nn.Module` subclass, such that one can directly use it as an advection-diffusion PDE solver for data simulation or easily wrap it into DNNs or numerical optimization frameworks for inverse PDE problems, i.e., parameters estimation.

C.1. Computing Advection

Given a certain velocity field $\mathbf{V}$, we have the advection equation generally written as:

$$\left.\frac{\partial C}{\partial t}\right|_{\text{adv}} = -\nabla\cdot(\mathbf{V}C), \quad \mathbf{V}\in\mathbb{R}^d(\Omega), \tag{44}$$

where d refers to the dimension of the domain Ω, $C=C(\mathbf{x},t)$ denotes the mass concentration at location $\mathbf{x}\in\Omega$ and time t. For 1D domains, or the special case where the velocity is constant (V) over space, Eq. (44) is simplified as $\left.\frac{\partial C}{\partial t}\right|_{\text{adv}} = -V\cdot\nabla C$.

Advection with Incompressible Flow Assume the the velocity field $\mathbf{V}$ is divergence free, i.e., $\nabla\cdot\mathbf{V}=0$, we have

$$\left.\frac{\partial C}{\partial t}\right|_{\text{adv}} = -\nabla\cdot(\mathbf{V}C) = -\nabla\cdot\mathbf{V}C - \mathbf{V}\cdot\nabla C = -\mathbf{V}\cdot\nabla C. \tag{45}$$

First-order Upwind Scheme Given a 3D volumetric concentration image $C=\left(C_{i,j,k}\right)_{N_x\times N_y\times N_z}$ with grid sizes Δx, Δy, Δz, we approximate the partial differential derivatives in the advection equation (the right hand side of Eq. (44)) using upwind differences. We apply a first-order upwind scheme along each direction x, y, z, based on the corresponding velocity components V^x, V^y, V^z of $\mathbf{V}$. Specifically, $\frac{\partial C}{\partial x}$ at (i,j,k) is determined as:

$$\left.\frac{\partial C}{\partial x}\right|_{i,j,k} = \begin{cases} \frac{C_{i,j,k}-C_{i-1,j,k}}{\Delta x}, & V^x_{i,j,k}\geq 0 \\ \frac{C_{i+1,j,k}-C_{i,j,k}}{\Delta x}, & V^x_{i,j,k}<0 \end{cases}, \quad \left.\frac{\partial C}{\partial y}\right|_{i,j,k} = \begin{cases} \frac{C_{i,j,k}-C_{i,j-1,k}}{\Delta y}, & V^y_{i,j,k}\geq 0 \\ \frac{C_{i,j+1,k}-C_{i,j,k}}{\Delta y}, & V^y_{i,j,k}<0 \end{cases}, \quad \left.\frac{\partial C}{\partial z}\right|_{i,j,k} = \begin{cases} \frac{C_{i,j,k}-C_{i,j,k-1}}{\Delta z}, & V^z_{i,j,k}\geq 0 \\ \frac{C_{i,j,k+1}-C_{i,j,k}}{\Delta z}, & V^z_{i,j,k}<0 \end{cases}. \tag{46}$$

Courant-Friedrichs-Lewy Condition for Advection To ensure that the above upwind scheme is numerically stable when integrating Eq. (44), the following Courant-Friedrichs-Lewy condition (CFL) should be satisfied:

$$c = \sum_{ax\in\{x,y,z\}} \frac{V^{ax}\Delta t}{\Delta ax} \leq c_{\max}, \tag{47}$$

where $c_{\max}$ approximately equals to 1 when applying explicit methods for ordinary differential equations (ODEs) [19, 25].

C.2. Computing Diffusion

Given a certain diffusion field $\mathbf{D}$, the diffusion equation is written as:

$$\left.\frac{\partial C}{\partial t}\right|_{\text{diff}} = \nabla\cdot(\mathbf{D}\nabla C), \quad \mathbf{D}\in\mathbb{R}^{n\times n}(\Omega). \tag{48}$$

where $C=C(\mathbf{x},t)$ denotes the mass concentration at location $\mathbf{x}\in\Omega$ and time t. For 1D domains, or the special case where the diffusion is a constant or a scalar field (D), Eq. (48) can be simplified to $\left.\frac{\partial C}{\partial t}\right|_{\text{diff}} = D\cdot\Delta C$.

Nested Central-Forward-Backward Differences In order to obtain a more stable numerical discretization scheme for the diffusion part in Eq. (48), in practice, we explicitly compute its expanded formula. Here, we give the brief illustration of our discretization scheme for $n = 2, n = 3$, corresponding to the diffusion tensors on 2D and 3D domains, respectively. ($\mathbf{D}$ are represented via Eq. (40) (2D) or Eq. (42) (3D) when the diffusion fields are assumed to be symmetric PSD.)

- *2D tensors.* For symmetric PSD tensor field $\mathbf{D} = \begin{bmatrix} Dxx & Dxy \\ Dxy & Dyy \end{bmatrix} \in \mathbb{R}^{2\times 2}(\Omega)$:

$$\begin{aligned} \left.\frac{\partial C}{\partial t}\right|_{\text{diff}} &= \nabla \cdot \left(\begin{bmatrix} Dxx & Dxy \\ Dxy & Dyy \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial C}{\partial x} \\ \frac{\partial C}{\partial y} \end{bmatrix} \right) = \frac{\partial}{\partial x}\left(Dxx\frac{\partial C}{\partial x} + Dxy\frac{\partial C}{\partial y} \right) + \frac{\partial}{\partial y}\left(Dxy\frac{\partial C}{\partial x} + Dyy\frac{\partial C}{\partial y} \right) \\ &= \underbrace{\left(\frac{\partial Dxx}{\partial x}\frac{\partial C}{\partial x} + \frac{\partial Dxy}{\partial x}\frac{\partial C}{\partial y} + \frac{\partial Dxy}{\partial y}\frac{\partial C}{\partial x} + \frac{\partial Dyy}{\partial y}\frac{\partial C}{\partial y} \right)}_{\text{(a)}} + \underbrace{\left(Dxx\frac{\partial^2 C}{\partial x^2} + 2Dxy\frac{\partial^2 C}{\partial x \partial y} + Dyy\frac{\partial^2 C}{\partial y^2} \right)}_{\text{(b)}}. \end{aligned} \tag{49}$$

- For symmetric positive semi-definite tensor $\mathbf{D} = \begin{bmatrix} Dxx & Dxy & Dxz \\ Dxy & Dyy & Dyz \\ Dxz & Dyz & Dzz \end{bmatrix} \in \mathbb{R}^{3\times 3}(\Omega)$:

$$\begin{aligned} \left.\frac{\partial C}{\partial t}\right|_{\text{diff}} &= \nabla \cdot \left(\begin{bmatrix} Dxx & Dxy & Dxz \\ Dxy & Dyy & Dyz \\ Dxz & Dyz & Dzz \end{bmatrix} \cdot \begin{bmatrix} \frac{\partial C}{\partial x} \\ \frac{\partial C}{\partial y} \\ \frac{\partial C}{\partial z} \end{bmatrix} \right) \\ &= \frac{\partial}{\partial x}\left(Dxx\frac{\partial C}{\partial x} + Dxy\frac{\partial C}{\partial y} + Dxz\frac{\partial C}{\partial z} \right) + \frac{\partial}{\partial y}\left(Dxy\frac{\partial C}{\partial x} + Dyy\frac{\partial C}{\partial y} + Dyz\frac{\partial C}{\partial z} \right) + \frac{\partial}{\partial z}\left(Dxz\frac{\partial C}{\partial x} + Dyz\frac{\partial C}{\partial y} + Dzz\frac{\partial C}{\partial z} \right) \\ &= \underbrace{\left(\left(\frac{\partial Dxx}{\partial x} + \frac{\partial Dxy}{\partial y} + \frac{\partial Dxz}{\partial z} \right)\frac{\partial C}{\partial x} + \left(\frac{\partial Dxy}{\partial x} + \frac{\partial Dyy}{\partial y} + \frac{\partial Dyz}{\partial z} \right)\frac{\partial C}{\partial y} + \left(\frac{\partial Dxz}{\partial x} + \frac{\partial Dyz}{\partial y} + \frac{\partial Dzz}{\partial z} \right)\frac{\partial C}{\partial z} \right)}_{\text{(a)}} \\ &\quad + \underbrace{\left(\left(\frac{\partial Dxx}{\partial x} + \frac{\partial Dxy}{\partial y} + \frac{\partial Dxz}{\partial z} \right)\frac{\partial C}{\partial x} + \left(\frac{\partial Dxy}{\partial x} + \frac{\partial Dyy}{\partial y} + \frac{\partial Dyz}{\partial z} \right)\frac{\partial C}{\partial y} + \left(\frac{\partial Dxz}{\partial x} + \frac{\partial Dyz}{\partial y} + \frac{\partial Dzz}{\partial z} \right)\frac{\partial C}{\partial z} \right)}_{\text{(a)}} \end{aligned} \tag{50}$$

Specifically, we use the central difference scheme for discretizing all first-order spatial derivatives in (a), and nested forward-backward differences for all second-order operators in (b).

Mesh Fourier Number for Diffusion To ensure that the above finite difference scheme for diffusion is numerically stable when integrating forward in time, the following condition should be satisfied[4]:

$$F = \sum_{ax\in\{x,y,z\}} \frac{D^{ax}\Delta t}{\Delta ax^2} \leq \frac{1}{2}, \tag{51}$$

where $D^{ax}, ax \in \{x, y, z\}$ refer to the diagonal entries Dxx, Dyy, Dzz of the diffusion tensor $\mathbf{D}$.

C.3. Numerical Solution

As introduced above, after discretizing all the spatial derivatives on the right side of Eq. (43), we obtain a system of ordinary differential equations (ODEs), which can be solved by numerical integration. We then use the RK45 method to advance in time (δt) to predict $\widehat{C}^{t+\delta t}$. Note when the input mass transport time-series has relatively large temporal resolution (Δt), the chosen δt should be smaller than Δt to satisfy the stability conditions discussed in Supp. C.1-C.2, thereby ensuring stable numerical integration.

[4] See https://hplgit.github.io/fdm-book/doc/pub/book/sphinx/._book011.html.

D. Brain Advection-Diffusion Dataset

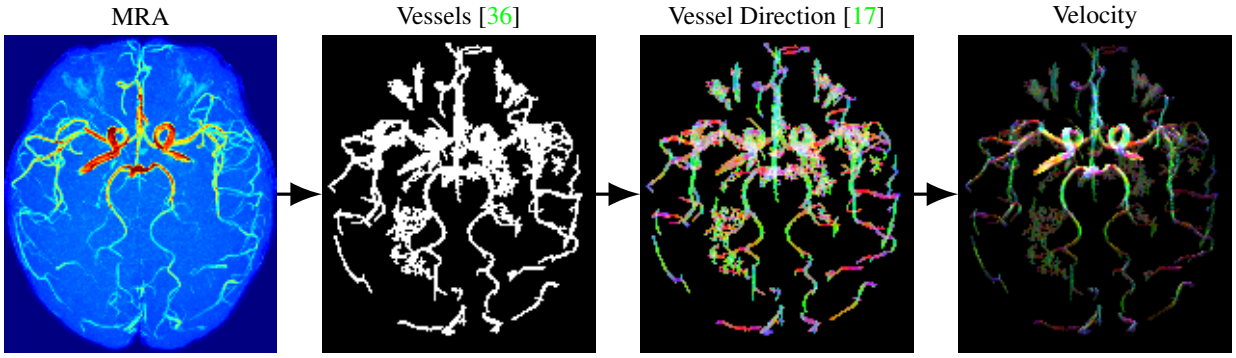

Figure D.11: Velocity simulation workflow (images shown in maximum intensity projection (MIP)), the last two vector fields are displayed in RGB maps (red - sagittal; green - coronal; blue - axial).

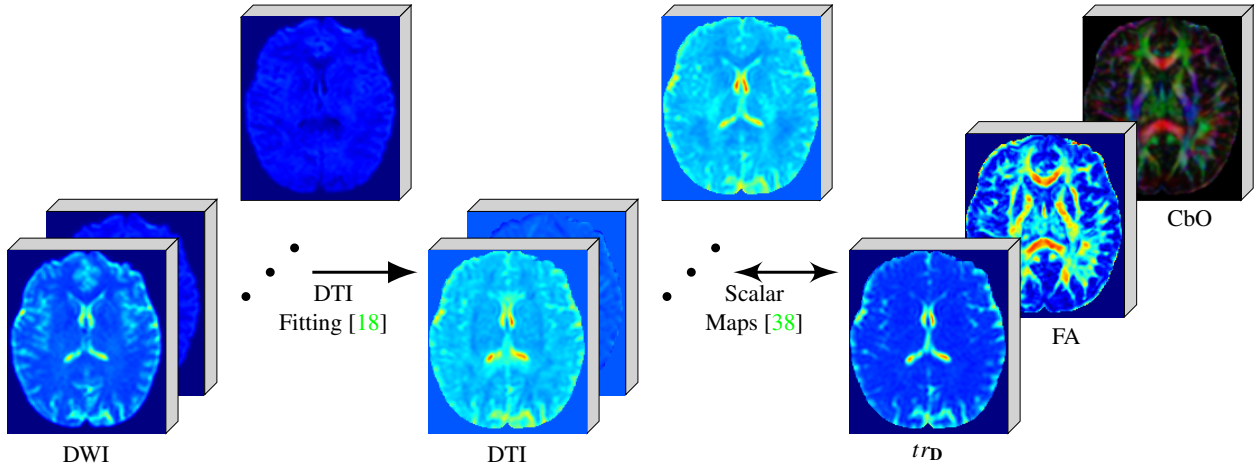

Figure D.12: Diffusion simulation workflow. DTI is estimated from DWIs using Dipy [18]. The scalar maps of DTI are then computed to describe the diffusion tensor structure.

Our brain advection-diffusion simulation dataset is based on the public `IXI` brain dataset[5], from which we use 200 patients with complete collections of T1-/T2-weighted images, magnetic resonance angiography (MRA) image, and diffusion-weighted images (DWI) with 15 directions for the simulation of 3D divergence-free velocity vector and symmetric PSD diffusion tensor fields. All above images are resampled to isotropic spacing ($1\,mm$), rigidly registered intra-subject, and skull-stripped using `ITK`[6].

D.1. Divergence-free Velocity Vector Field Simulation (Fig. D.11)

Blood Vessel Segmentation Brain blood vessels are segmented by `ITK-TubeTK` using T1-/T2-weighted and MRA images, where T1-/T2-weighted images are used to obtain more robust segmentation results[7].

[5] Available for download: http://brain-development.org/ixi-dataset/.
[6] Code in https://github.com/InsightSoftwareConsortium/ITK.
[7] Code in https://github.com/InsightSoftwareConsortium/ITKTubeTK/tree/master/examples/MRA-Head.

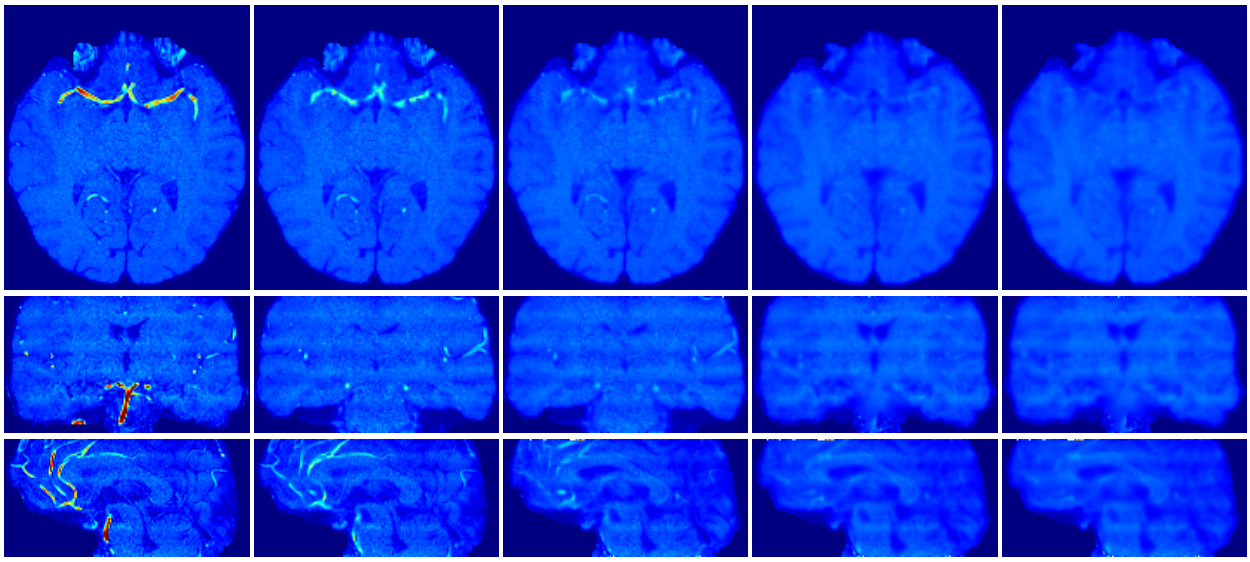

Figure D.13: Time-series of one brain advection-diffusion sample. Top: on one axial slice; Middle: on one coronal slice; Bottom: on one sagittal slice.

Principal Vessel Direction Inference We infer a vessel's directions by analyzing the spectral decomposition of the second order derivatives (Hessian, $\mathscr{H}_I$) of the vessel segmentation image I:

$$\mathscr{H}_I = \mathbf{U}_I \Lambda_I \mathbf{U}_I^T = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \mathbf{u}_3^T \end{bmatrix}. \tag{52}$$

where the eigenvalues are sorted by their absolute values as $|\lambda_1| \leq |\lambda_2| \leq |\lambda_3|$. Following Frangi et al. [17], the eigenvector $\mathbf{u}_1$ corresponding to λ_1 with the smallest absolute value, indicates the direction along the vessel, while the remaining eigenvectors $\mathbf{u}_2, \mathbf{u}_3$ form a base for the orthogonal plane crossing the vessel. We therefore take $\mathbf{u}_1$ as the vessel direction maps.

In order to reduce the effect from the sign ambiguity of eigenvector directions and thus obtain more consistent flow velocity directions, we first extract the centerline of the vessel map and determine the flow directions along the centerline. Specifically, we start at the point with the largest MRA intensity along the centerline, from which we set the flow direction to its nearest centerline point. Likewise, we traverse all points along the centerline and successively assign directions for the nearest point along the centerline. Afterwards, velocity directions of the remaining voxels within the vessels are set to be the same as its nearest centerline point. I.e., flip the sign of the velocity component if it is not the same with that of its nearest centerline point. In this way, we ensure a velocity field with consistent flow directions.

Divergence-free Velocity Simulation The velocity field $\mathbf{V}$ is obtained, by multiplying the vessel direction maps with the MRA intensities, to reflect the local velocity magnitudes. We scale the value range of the velocity field to $[-1, 1]$, and additionally compute velocity fields with 50% of the original velocity values.

Divergence-free velocity fields $\mathbf{V}_{\text{div}}$ are then estimated from $\mathbf{V}$ via Eq. (19) using numerical optimization, by encouraging $\mathbf{V}_{\text{div}} := \nabla \times \Psi$ close to $\mathbf{V}$:

$$\min_{\Psi} \left\| \nabla \times \Psi - \mathbf{V} \right\|_2^2, \quad \Psi \cdot \mathbf{n} \big|_{\partial\Omega} = 0, \Psi \in \mathbb{R}^3(\Omega). \tag{53}$$

D.2. Symmetric PSD Diffusion Tensor Field Simulation (Fig. D.12)

Diffusion Tensor Estimation Diffusion tensors are reconstructed from the DWIs using `Dipy`[8] [18]. Specifically, the diffusion tensors are estimated based on the following equation:

$$\frac{S(\mathbf{g}, b)}{S_0} = e^{-b\mathbf{g}^T \mathbf{D}\mathbf{g}}, \tag{54}$$

where $\mathbf{g}$ is a unit vector indicating the direction of measurement and b are the parameters of measurement, e.g., incorporating the strength and duration of the diffusion-weighting gradient. $S(\mathbf{g}, b)$ is the diffusion-weighted signal measured and S_0 is the signal measured with no diffusion weighting. $\mathbf{D}$ is the symmetric PSD tensor which contains six free parameters to be fit:

$$\mathbf{D} = \begin{bmatrix} D_{xx} & D_{xy} & D_{xz} \\ D_{xy} & D_{yy} & D_{yz} \\ D_{xz} & D_{yz} & D_{zz} \end{bmatrix}. \tag{55}$$

We scale the value range of the diffusion tensor fields for each subject sample to $[-0.2, 0.2]$, and additionally compute diffusion fields with 50% of the original diffusion values.

Diffusion Scalar Maps Upon obtaining the diffusion tensor $\mathbf{D}$, we decompose it into eigenvectors ($\mathbf{U}$) and eigenvalues (Λ):

$$\mathbf{D} = \mathbf{U}\Lambda\mathbf{U}^T = \begin{bmatrix} \mathbf{u}_1 & \mathbf{u}_2 & \mathbf{u}_3 \end{bmatrix} \begin{bmatrix} \lambda_1 & & \\ & \lambda_2 & \\ & & \lambda_3 \end{bmatrix} \begin{bmatrix} \mathbf{u}_1^T \\ \mathbf{u}_2^T \\ \mathbf{u}_3^T, \end{bmatrix}. \tag{56}$$

from which we can further visualize the diffusion scalar maps [38], which describe tensor structure. In this work, we consider (1) Trace ($tr_{\mathbf{D}}$),

$$tr_{\mathbf{D}} = \lambda_1 + \lambda_2 + \lambda_3; \tag{57}$$

(2) Fractional anisotropy (FA),

$$\mathrm{FA} = \sqrt{\frac{1}{2}} \sqrt{\frac{(\lambda_1 - \lambda_2)^2 + (\lambda_2 - \lambda_3)^2 + (\lambda_3 - \lambda_1)^2}{\lambda_1^2 + \lambda_2^2 + \lambda_3^2}}; \tag{58}$$

(3) Principal diffusion orientation ($\mathbf{U}_{\text{prin}}$), the eigenvector $\mathbf{U}_{\text{prin}} = [u_x, u_y, u_z]^T$ corresponding to the largest eigenvalue.

Further, we also visualize the color-by-orientation (CbO) map, by assigning colors to voxels based on a combination of FA and $\mathbf{U}_{\text{prin}}$, i.e.,

$$\text{Red} = \mathrm{FA} \cdot u_x; \ \text{Green} = \mathrm{FA} \cdot u_y; \ \text{Blue} = \mathrm{FA} \cdot u_z. \tag{59}$$

D.3. Brain Advection-diffusion Time-series Simulation (Fig. D.13)

For each brain advection-diffusion sample, the initial concentration state is assumed to be given by the MRA image with intensity ranges rescaled to $[0, 1]$.Time-series (length $N_T = 40$, interval $\Delta t = 0.1\,s$) are then simulated given the computed divergence-free velocity fields (Supp. D.1) and symmetric PSD diffusion tensor fields (Supp. D.2) by our advection-diffusion PDE solver (Supp. C). Thus the simulated dataset includes 800 brain advection-diffusion time-series (4 time-series for each of the 200 subjects, based on the four combinations of the simulated two velocity fields and two diffusion fields).

[8] Code in https://github.com/dipy/dipy.