Inverse kernel decomposition (IKD)

Chengrui Li, Anqi Wu

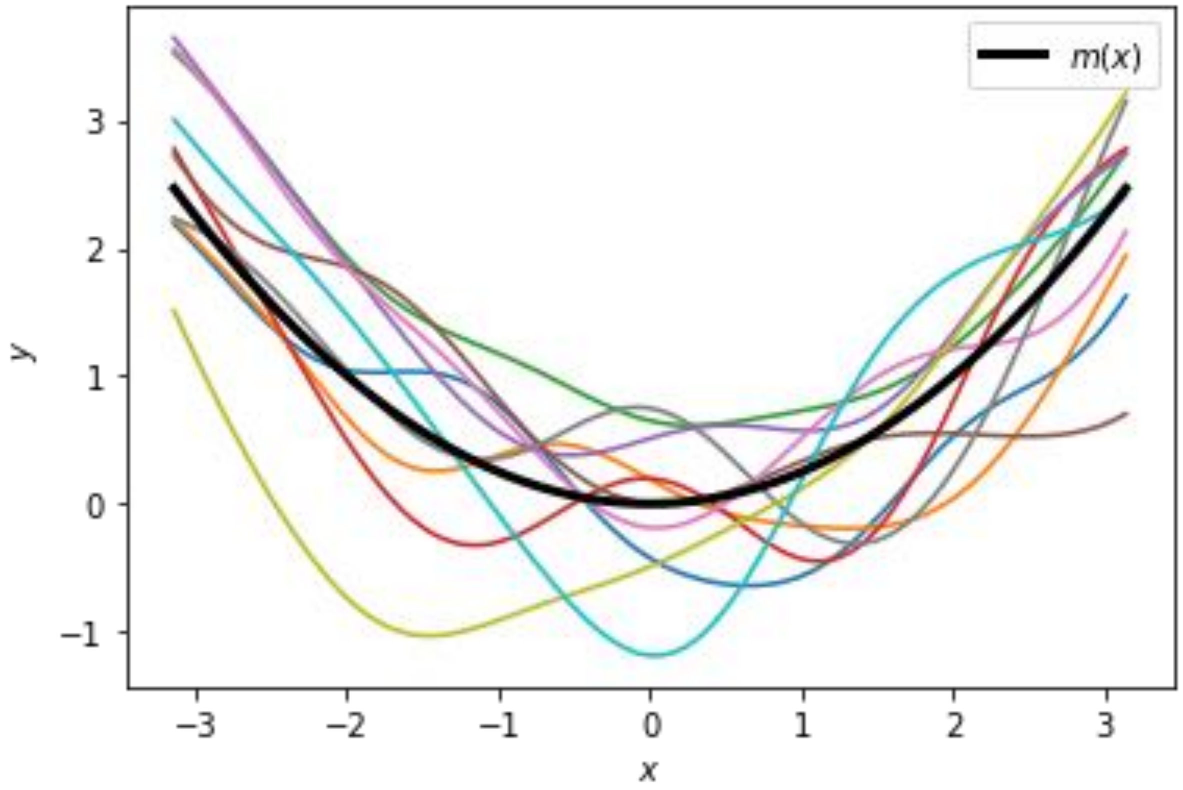
TMLR

1 GP and GPLVM background

Gaussian process (GP)

- $f(\boldsymbol{x})$ is a stochastic function from GP
$$f \sim \mathcal{GP}\left(m(\boldsymbol{x}), k(\boldsymbol{x}, \boldsymbol{x}')\right)$$
where $m(\boldsymbol{x})$ is the average output. The **kernel function** $k(\boldsymbol{x}, \boldsymbol{x}')$ makes the closer the inputs $(\boldsymbol{x}, \boldsymbol{x}')$ are, the higher the correlation of the outputs is, so that the function is smooth
- Method of sampling discretized GP outputs $\boldsymbol{y} = [y_1, \cdots, y_N]^{\mathrm{T}}$ on discretized inputs $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_N]^{\mathrm{T}}$:
$$\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K})$$
where $\boldsymbol{\mu} = [m(\boldsymbol{x}_1), \cdots, m(\boldsymbol{x}_N)]^{\mathrm{T}}$, $\boldsymbol{K} = \left(k(\boldsymbol{x}_i, \boldsymbol{x}_j)\right)_{N \times N}$

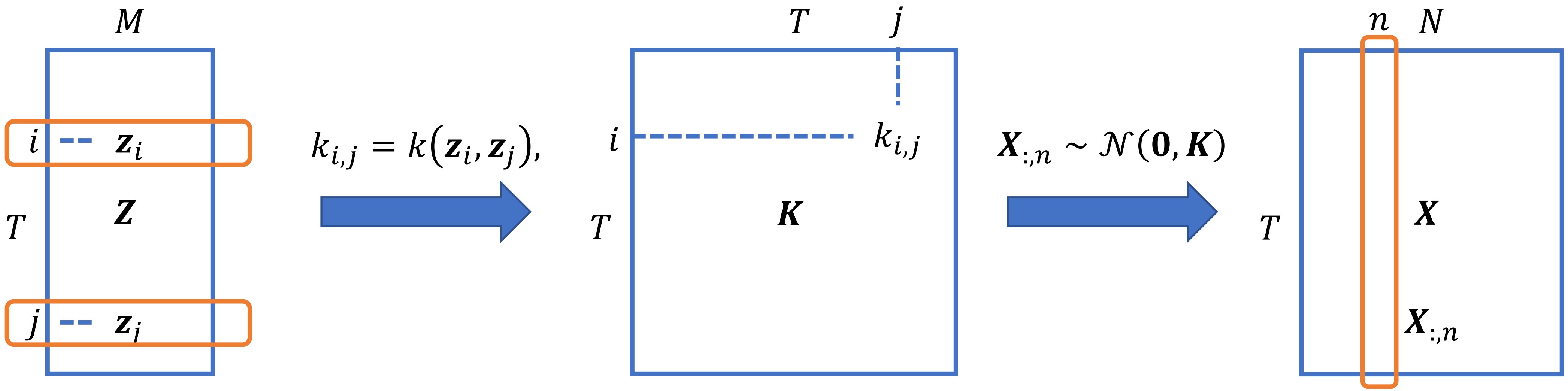
Gaussian process (GP)



$$m(x) = \frac{1}{4}x^2, k(x, x') = \left(\frac{1}{\sqrt{2}}\right)^2 \exp\left(-\frac{1}{2}(x - x')^2\right)$$

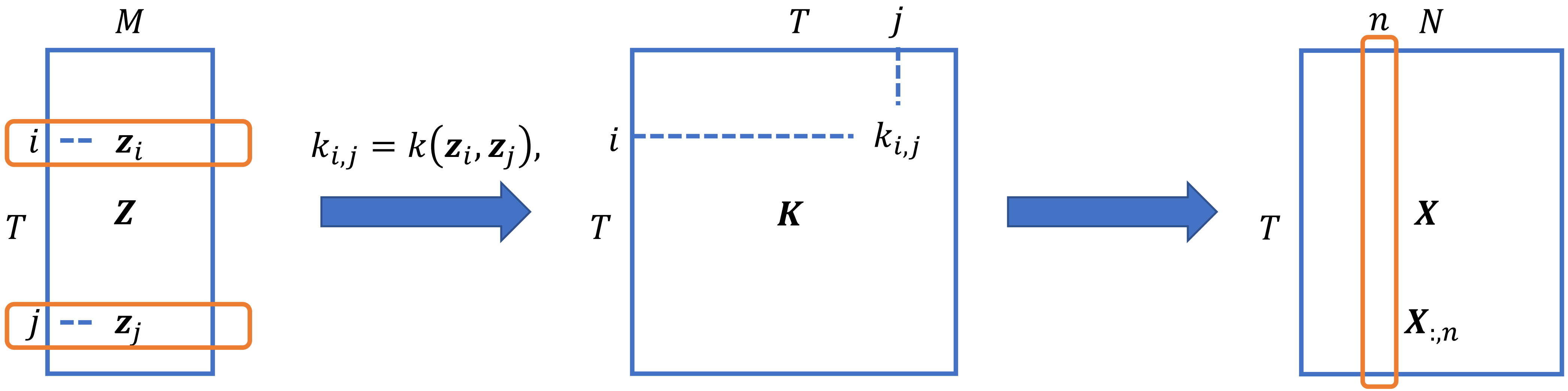
Gaussian process latent variable model (GPLVM)

- Observation: $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_T]^{\mathrm{T}} \in \mathbb{R}^{T \times N}$, T data points, N observation dimensions
- Latent variables: $\boldsymbol{Z} = [\boldsymbol{z}_1, \cdots, \boldsymbol{z}_T]^{\mathrm{T}} \in \mathbb{R}^{T \times M}$, M latent dimensions
- Each observed dimension is sampled from GP independently



Gaussian process latent variable model (GPLVM)

- Dimensionality reduction problem: when given $\boldsymbol{X}$, find the optimal $\boldsymbol{Z}$ under the GPLVM assumption
- Method 1: Optimization-based traditional GPVLM solver
- Method 2: Our newly proposed **inverse kernel decomposition (IKD)**



Derivation of IKD, from $\boldsymbol{k_{i,j}}$ to $\boldsymbol{d_{i,j}}$

- Use the squared exponential (SE) kernel for example,
$$k_{i,j} = k(\boldsymbol{z}_i, \boldsymbol{z}_j) = \sigma^2 \exp\left(-\frac{\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|^2}{2l^2}\right)$$
where σ^2 is the marginal variance and l is the length-scale
- Denote the squared distance $d_{i,j} \coloneqq \frac{\left\|\boldsymbol{z}_i - \boldsymbol{z}_j\right\|^2}{l^2}$, and let f be the mapping rule of SE, then we can write
$$k_{i,j} = \sigma^2 f(d_{i,j}) = \sigma^2 \exp\left(-\frac{d_{i,j}}{2}\right)$$
- Since f is strictly monotonic, we can write the inverse relationship as
$$d_{i,j} = f^{-1}\left(\frac{k_{i,j}}{\sigma^2}\right) = -2\ln\frac{k_{i,j}}{\sigma^2}$$

2 Method—IKD

Derivation of IKD, from $\boldsymbol{D} = \left(d_{i,j}\right)_{T\times T}$ to $\boldsymbol{Z}$

- Denote $\tilde{\boldsymbol{z}} = \frac{\boldsymbol{z}-\boldsymbol{z}_1}{l}$ with $\tilde{\boldsymbol{z}}_1 = \boldsymbol{0}$, we have

$$d_{i,j} = \frac{1}{l^2}\left(\boldsymbol{z}_i - \boldsymbol{z}_j\right)^{\mathrm{T}}\left(\boldsymbol{z}_i - \boldsymbol{z}_j\right) = \tilde{\boldsymbol{z}}_i^{\mathrm{T}}\tilde{\boldsymbol{z}}_i + \tilde{\boldsymbol{z}}_j^{\mathrm{T}}\tilde{\boldsymbol{z}}_j - 2\tilde{\boldsymbol{z}}_i^{\mathrm{T}}\tilde{\boldsymbol{z}}_j$$

- Making use of $\tilde{\boldsymbol{z}}_1 = \boldsymbol{0}$, we can get $d_{1,j} = \tilde{\boldsymbol{z}}_j^{\mathrm{T}}\tilde{\boldsymbol{z}}_j$, and finally obtains

$$\tilde{\boldsymbol{Z}}\tilde{\boldsymbol{Z}}^{\mathsf{T}} = \left[\begin{array}{c|cccc} 0 & 0 & 0 & \cdots & 0 \\ \hline 0 & d_{2,1} & \frac{1}{2}(d_{2,1}+d_{1,3}-d_{2,3}) & \cdots & \frac{1}{2}(d_{2,1}+d_{1,T}-d_{2,T}) \\ 0 & \frac{1}{2}(d_{3,1}+d_{1,2}-d_{3,2}) & d_{3,1} & \cdots & \frac{1}{2}(d_{3,1}+d_{1,T}-d_{3,T}) \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & \frac{1}{2}(d_{T,1}+d_{1,2}-d_{T,2}) & \frac{1}{2}(d_{T,1}+d_{1,3}-d_{T,3}) & \cdots & d_{T,1} \end{array}\right]$$

Denote it as $g(\boldsymbol{D})$

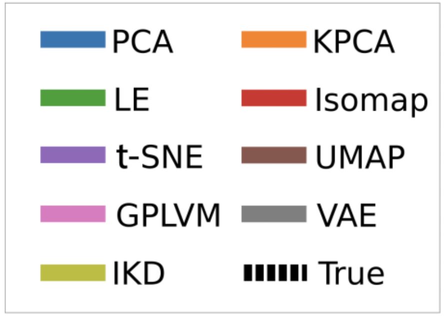
Derivation of IKD, algorithm

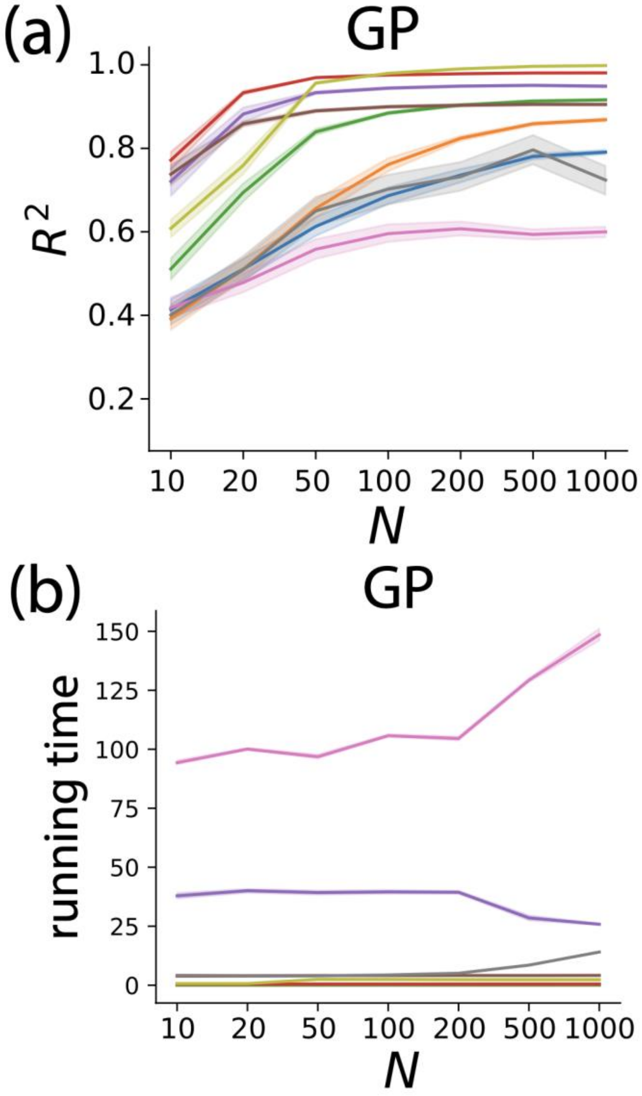
- Compute the $T \times T$ correlation matrix $\boldsymbol{S}$ of $\boldsymbol{X}$
- $\hat{d}_{i,j} = f^{-1}(s_{i,j})$ serves as an estimation of $d_{i,j}$
- $\boldsymbol{U}, \boldsymbol{\Lambda} \leftarrow$ eigen-decomposition of $g(\widetilde{\boldsymbol{D}})$
- $\widetilde{\boldsymbol{U}} = \left(\sqrt{\lambda_1}\boldsymbol{U}_{:,1}, \cdots, \sqrt{\lambda_M}\boldsymbol{U}_{:,M}\right)$ is the optimal rank-M positive definite approximation of $\boldsymbol{Z}$, where $\lambda_1, \cdots, \lambda_M$ are the first M largest (algebraically) positive eigenvalues of $g(\widetilde{\boldsymbol{D}})$ and $\boldsymbol{U}_{:,1}, \cdots, \boldsymbol{U}_{:,M}$ are the corresponding eigenvectors.

IKD with general stationary kernels

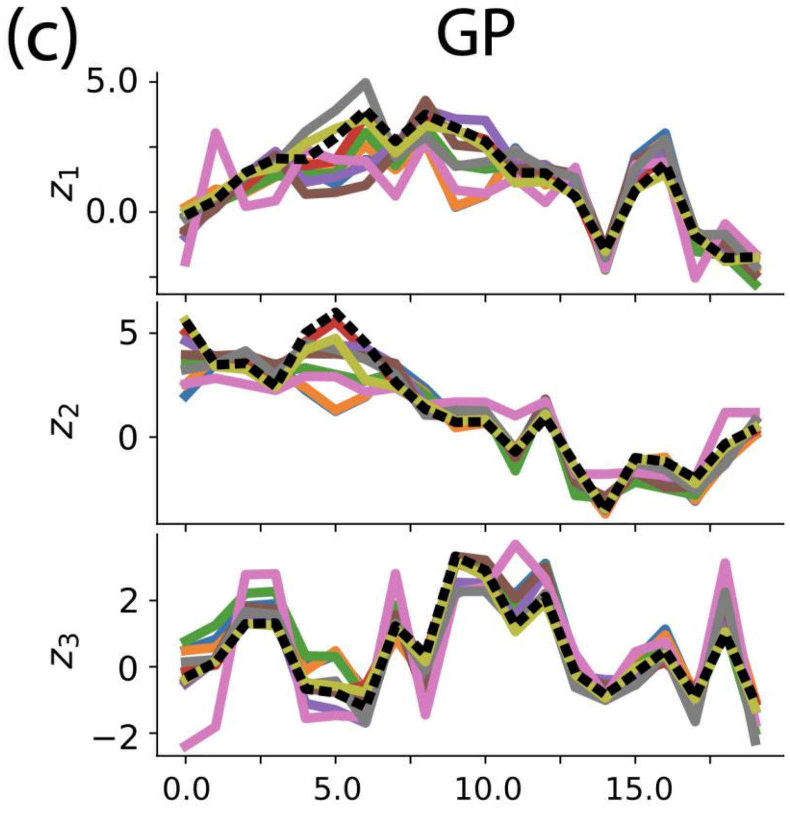
- Squared exponential: $f(d) = \exp\left(-\frac{d}{2}\right)$
  - Generalize to ARD kernel: $k(\boldsymbol{z}_i, \boldsymbol{z}_j) = \sigma^2 \exp\left(-\frac{1}{2}\sum_{m=1}^{M}\frac{1}{l_m^2}\left(z_{i,m} - z_{j,m}\right)^2\right)$
  - Generalize to Gaussian kernel: $k(\boldsymbol{z}_i, \boldsymbol{z}_j) = \sigma^2 \exp\left(-\frac{1}{2}\left(\boldsymbol{z}_i - \boldsymbol{z}_j\right)^{\mathrm{T}}\boldsymbol{L}^{-1}(\boldsymbol{z}_i - \boldsymbol{z}_j)\right)$
- Rational quadratic: $f(d) = \left(1 + \frac{d}{2\alpha}\right)^{-\alpha}$
- γ-exponential: $f(d) = \exp\left(-d^{\frac{\gamma}{2}}\right)$
- Matérn: $f(d) = \frac{2^{1-\nu}}{\Gamma(\nu)}\left(\sqrt{2\nu}\sqrt{d}\right)^{\nu} K_{\nu}\left(\sqrt{2\nu}\sqrt{d}\right)$
  - No closed-form inverse, but it is solvable with root-finding algorithm

Dimensionality reduction on synthetic dataset from GP





Visualization of the estimated latent from different methods

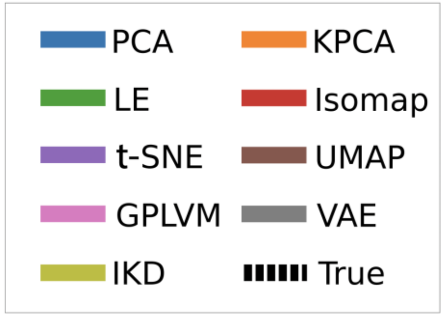


- $\boldsymbol{Z} \in \mathbb{R}^{T\times 3} \xrightarrow{\mathcal{GP}} \boldsymbol{X} \in \mathbb{R}^{T\times N} \xrightarrow{\text{IKD}} \widetilde{\boldsymbol{Z}}$

- Isomap is the best when $N < 50$
- IKD is the best when $N > 50$
- IKD is time efficient compared with optimization-based methods
- IKD captures the detail of the latent very well

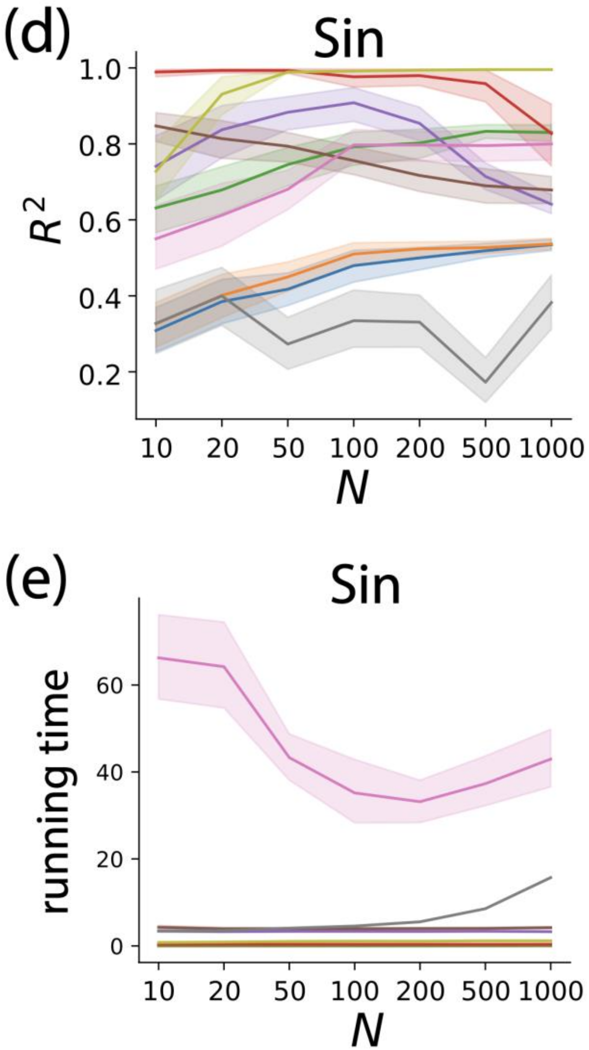
3 Experiments

Dimensionality reduction on synthetic dataset from sine function

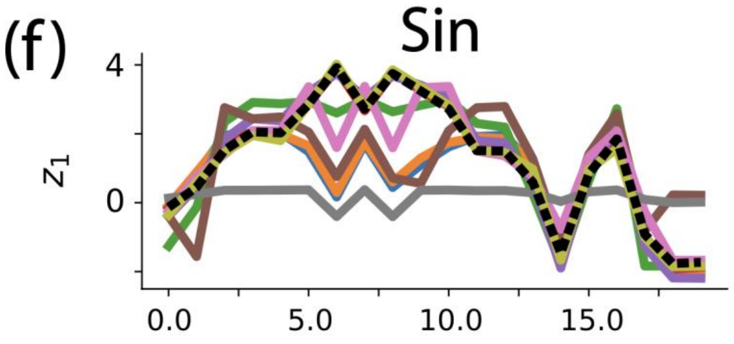


- $\boldsymbol{Z} \in \mathbb{R}^{T\times 1} \xrightarrow{\sin\cdot} \boldsymbol{X} \in \mathbb{R}^{T\times N} \xrightarrow{\text{IKD}} \widetilde{\boldsymbol{Z}}$
- $\boldsymbol{x}_t = \sin(\boldsymbol{\Omega} \boldsymbol{z}_t + \boldsymbol{\varphi}) + \boldsymbol{\varepsilon}_t$, where $\boldsymbol{\varepsilon}_t$ are Gaussian noises

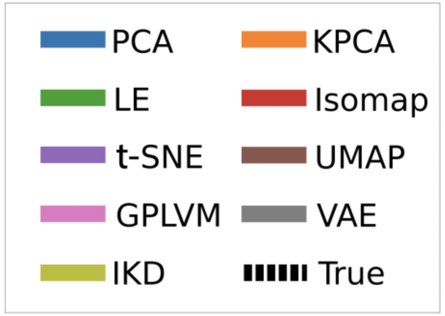
- Isomap is the best when $N < 50$
- IKD is the best when $N > 50$
- IKD is time efficient compared with optimization-based methods
- IKD captures the detail of the latent very well



Visualization of the estimated latent from different methods

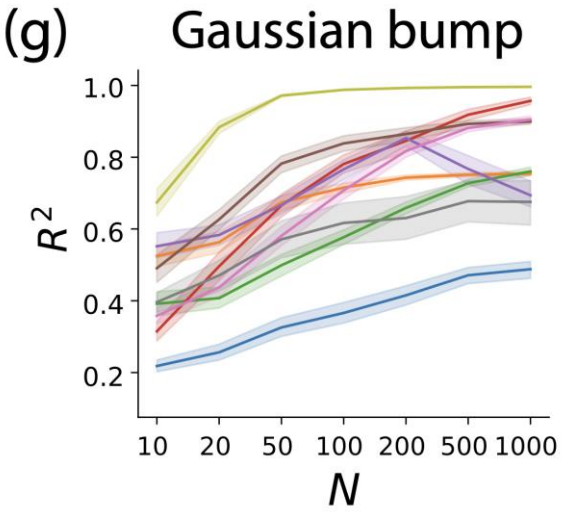


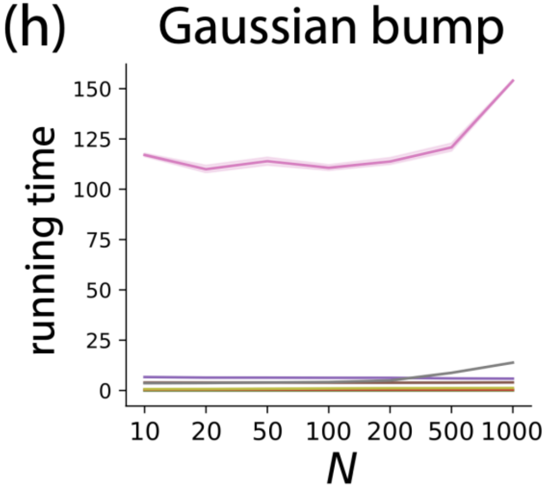
Dimensionality reduction on synthetic dataset from Gaussian Bump function



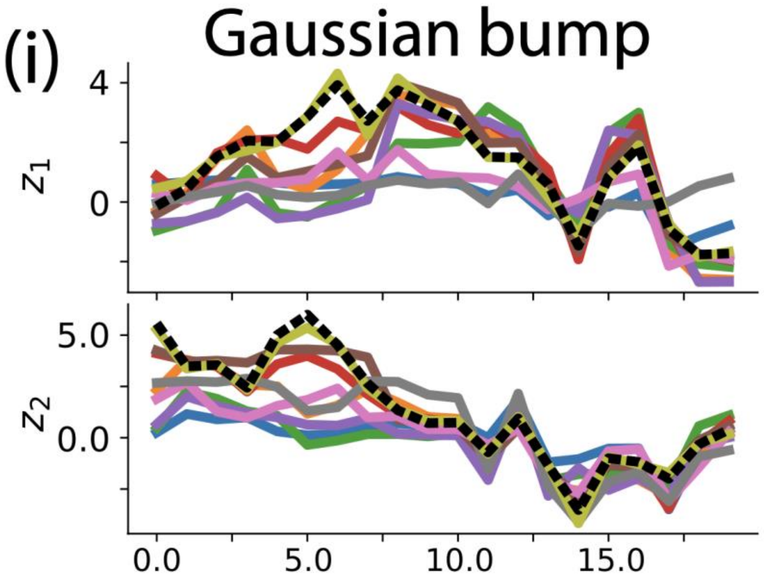
- $\boldsymbol{Z} \in \mathbb{R}^{T\times 3} \xrightarrow{\text{Gaussian Bump}} \boldsymbol{X} \in \mathbb{R}^{T\times N} \xrightarrow{\text{IKD}} \widetilde{\boldsymbol{Z}}$
- $x_{t,n} = 20\exp(-\|\boldsymbol{z}_t - \boldsymbol{c}_t\|_2^2) + \varepsilon_{t,n}$, where $\varepsilon_{t,n}$ are Gaussian noises

- IKD is the best one among all methods for all observation dimensionality N
- IKD is time efficient compared with optimization-based methods
- IKD captures the detail of the latent very well



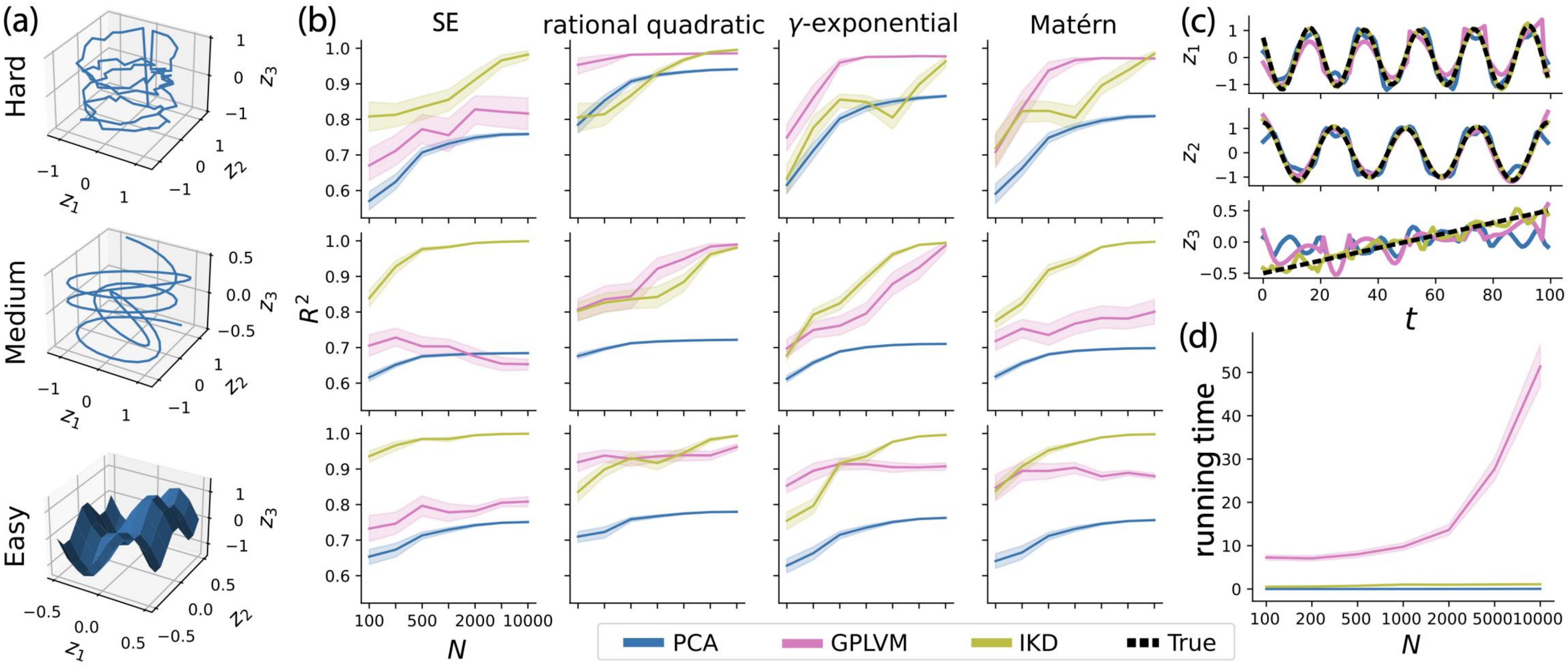


Visualization of the estimated latent from different methods



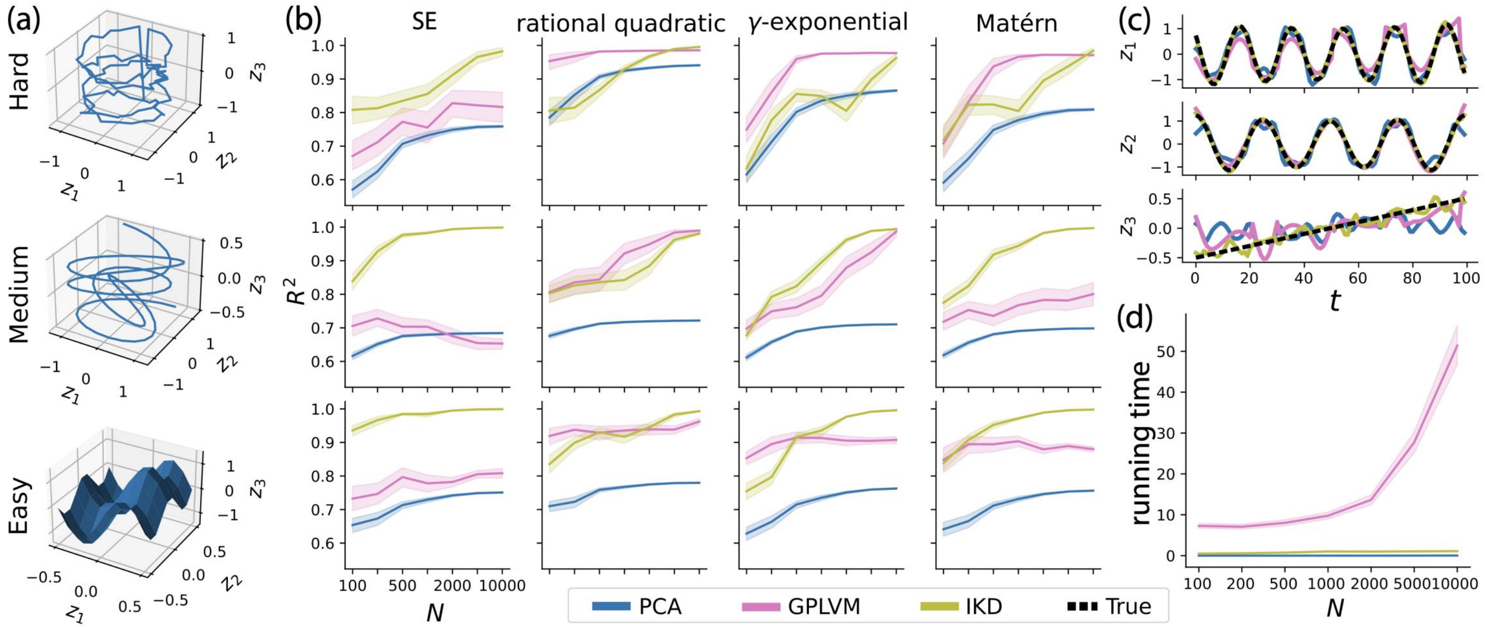
Ablation study

Three 3D latents of different difficulty levels, four different kernels, GP mapping function



Ablation study

- IKD is always the best for the most commonly used SE kernel
- IKD is competitive when observation dimensionality is high
- IKD is time efficient compared with the traditional optimization-based GPLVM solver

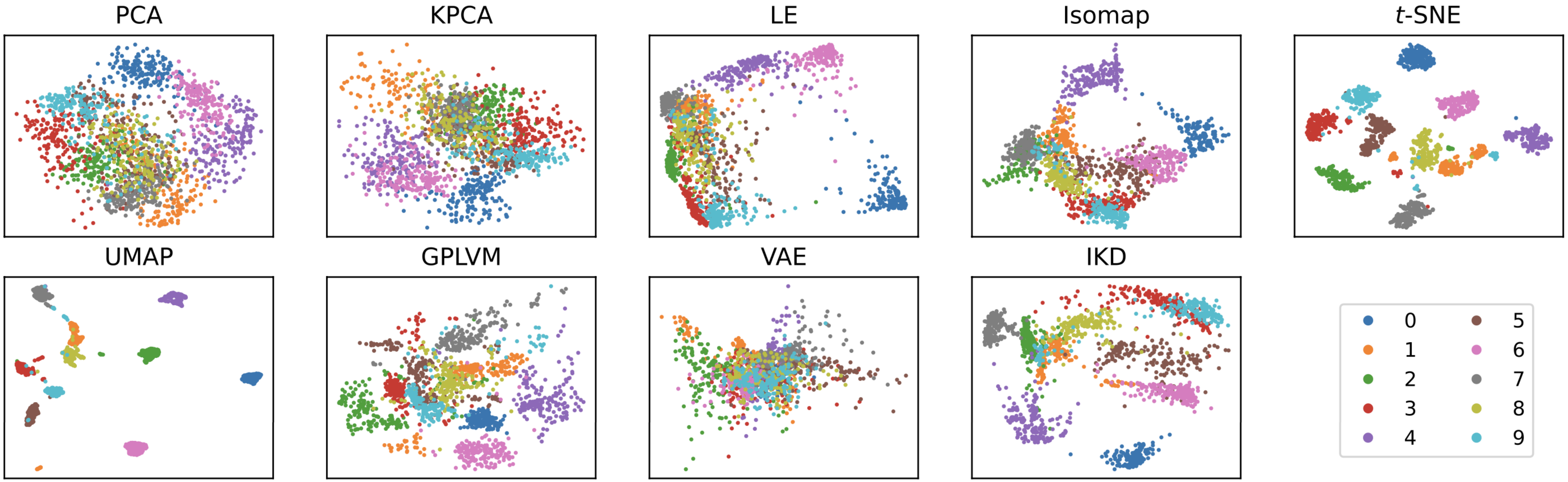


Real-world data

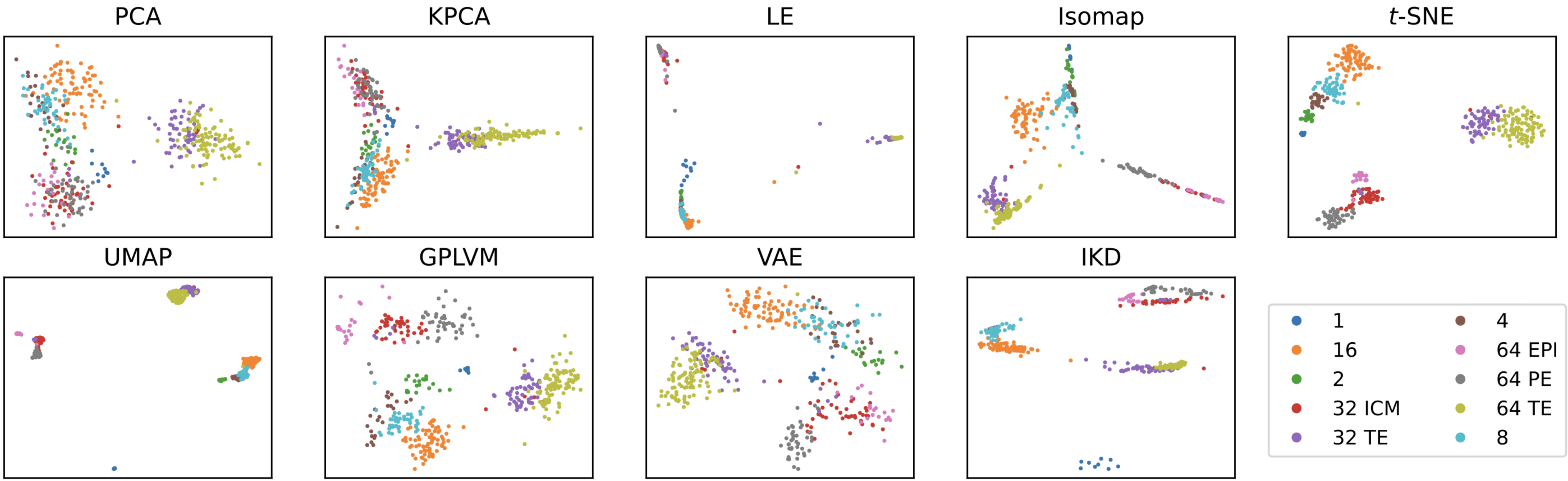
- Single-cell qPCR (PRC): Normalized measurements of 48 genes of a single cell at 10 different stages. There are 437 data points in total, resulting in $\boldsymbol{X} \in \mathbb{R}^{437 \times 84}$
- Handwritten digits (digits): It consists 1797 grayscale images of handwritten digits. Each one is an 8×8 image, resulting in $\boldsymbol{X} \in \mathbb{R}^{1797 \times 64}$
- COIL-20: It consists 1440 grayscale photos. For each one of the 20 objects in total, 72 photos were taken from different angles. Each one is a 128×128 image, resulting in $\boldsymbol{X} \in \mathbb{R}^{1440 \times 16384}$
- Fashion MNIST (F-MNIST) : It consists 70000 grayscale images of 10 fashion items (clothing, bags, etc.). We use a subset of it, resulting in $\boldsymbol{X} \in \mathbb{R}^{3000 \times 784}$

Digits dataset, $\boldsymbol{X} \in \mathbb{R}^{1797 \times 64}$

- Visually, t-SNE and UMAP are the best, then IKD, GPLVM, and Isomap

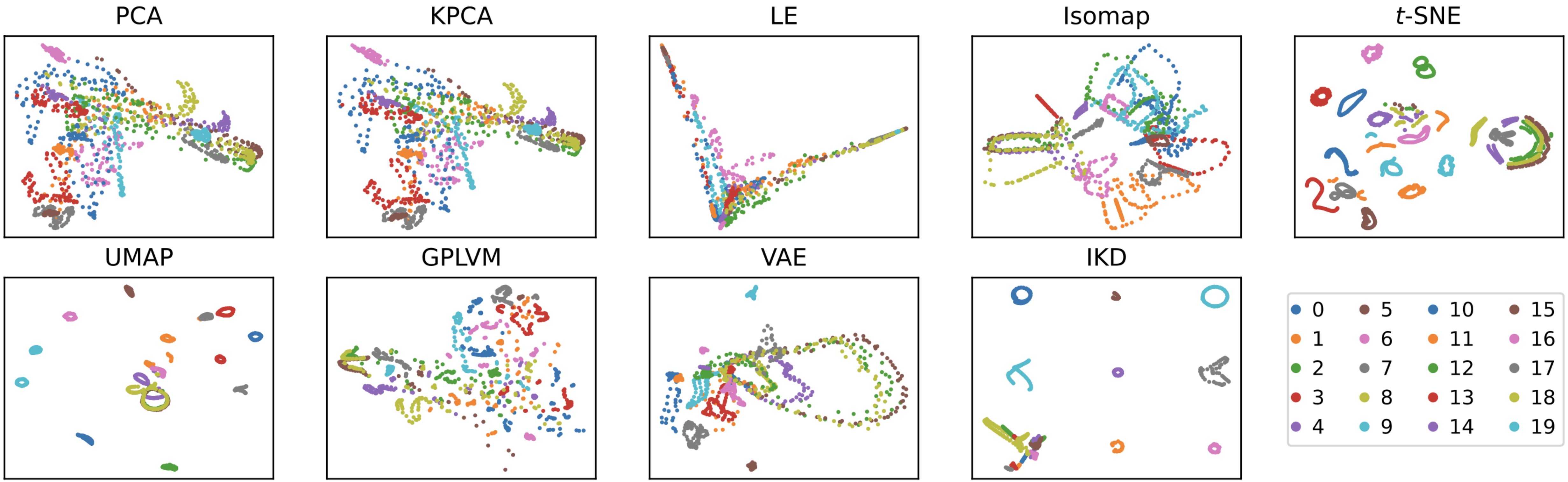


PCR dataset, $\boldsymbol{X} \in \mathbb{R}^{437\times 84}$



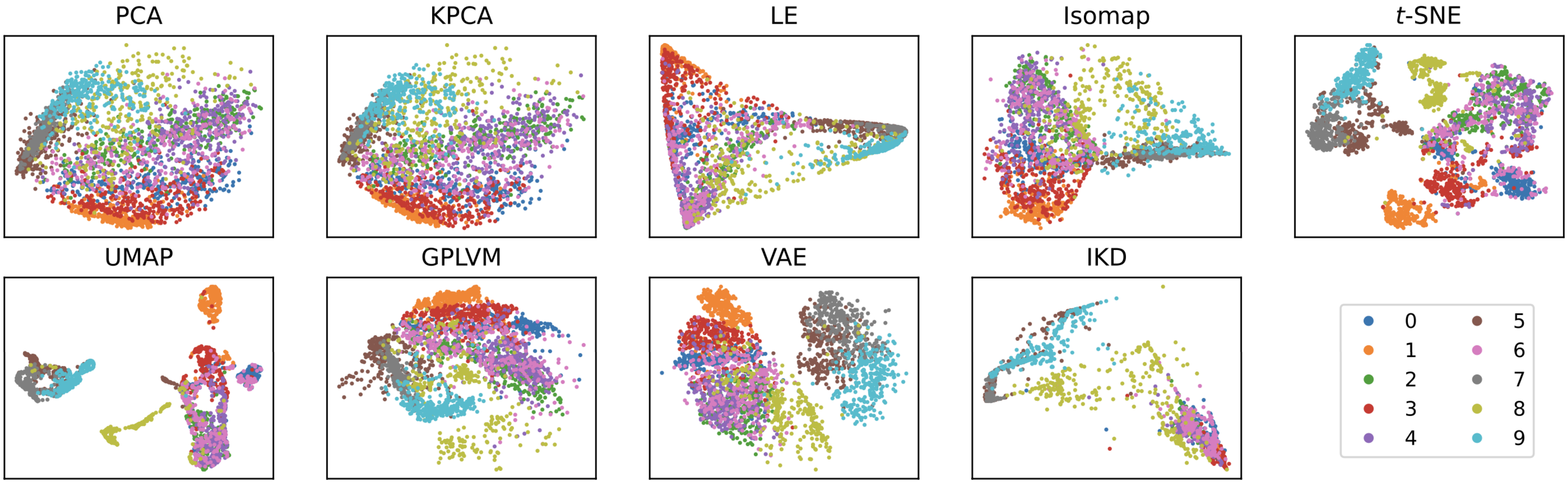
COIL-20 dataset, $\boldsymbol{X} \in \mathbb{R}^{1440 \times 16384}$

- Note that connected subgraphs are detected by IKD
- IKD should be the best since the observation dimensionality in this dataset is very high



F-MNIST dataset, $\boldsymbol{X} \in \mathbb{R}^{3000\times784}$

- The most difficult dataset
- Optimization-based methods are better than non-optimization-based methods

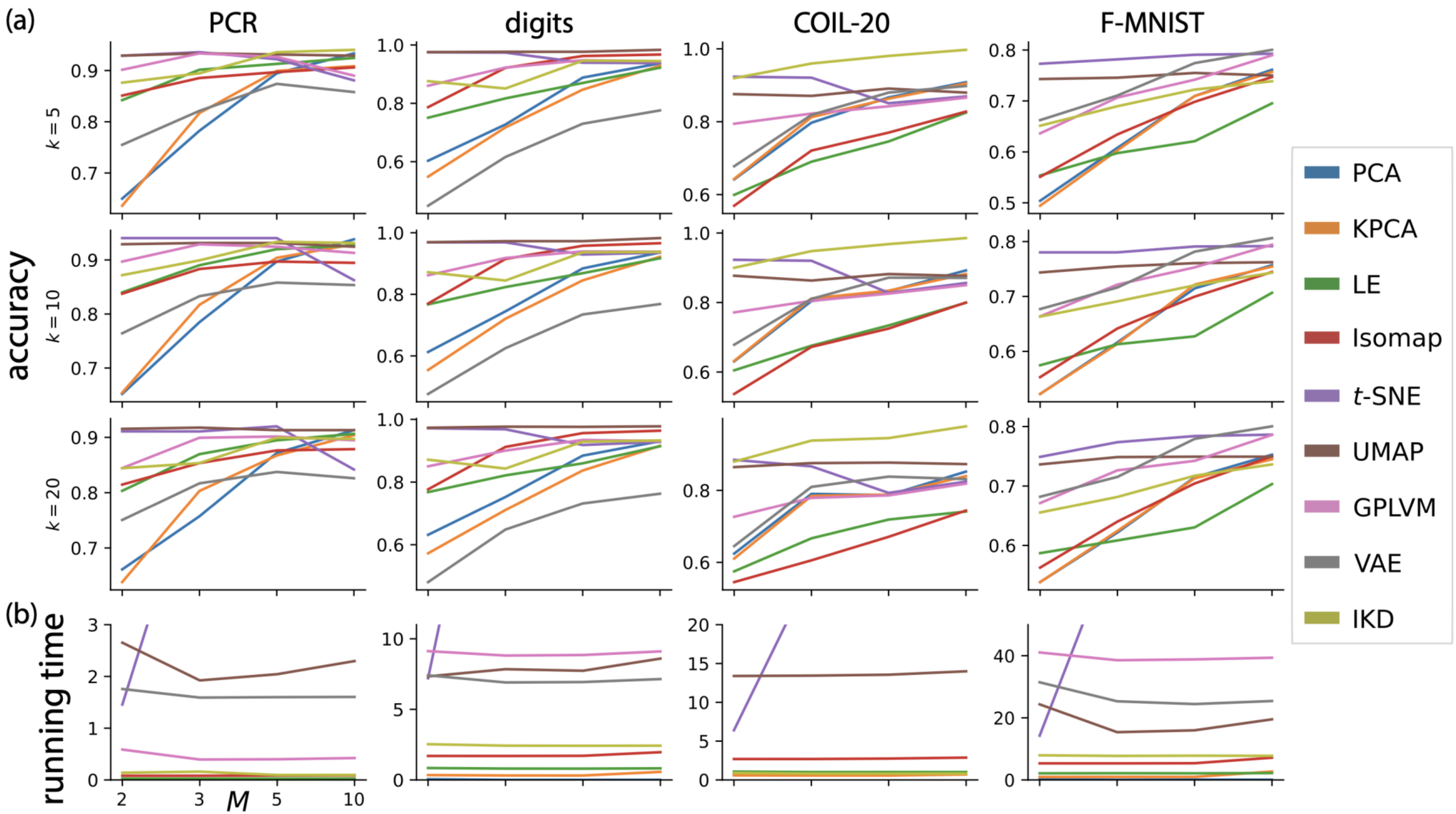


Quantitative comparison on real-world dataset

- Reduce the dimensionality to $\{2,3,5,10\}$ dimensional latent
- Use 5-fold cross-validation k-NN ($k \in \{5,10,20\}$) to evaluate the quality of the estimated low-dimensional latent
- Record the running time of each method

Quantitative comparison on real-world dataset

- IKD is faster than optimization-based methods
- IKD is one of the best among eigen-decomposition-based methods
- IKD is the most effective method for high-dimensional data



Quantitative comparison on real-world dataset

- IKD, as an eigen-decomposition-based method, consumes short running time, but is able to obtain dimensionality reduction results better than other eigen-decomposition-based methods
- When facing high-dimensional observation data, IKD can perform significantly better than all other methods in a very short time
- In terms of running time, IKD is on par with Isomap, and these eigen-decomposition-based methods are significantly faster than those four optimization-based methods

Thanks! Questions…