SPD-DDPM: Denoising Diffusion Probabilistic Models in the Symmetric Positive Definite Space

Yunchen Li[1*], Zhou Yu[1,2*], Gaoqi He[1], Yunhang Shen[3], Ke Li[3], Xing Sun[3], Shaohui Lin[1,2†]

[1]East China Normal University
[2]Key Laboratory of Advanced Theory and Application in Statistics and Data Science, Ministry of Education, China
[3]Youtu Lab, Tencent, Shanghai, China
52284404001@stu.ecnu.edu.cn, zyu@stat.ecnu.edu.cn, gqhe@cs.ecnu.edu.cn,
{shenyunhang01, tristanli.sh, winfred.sun, shaohuilin007}@gmail.com

Abstract

Symmetric positive definite (SPD) matrices have shown important value and applications in statistics and machine learning, such as FMRI analysis and traffic prediction. Previous works on SPD matrices mostly focus on discriminative models, where predictions are made directly on $E(X|y)$, where y is a vector and X is an SPD matrix. However, these methods are challenging to handle for large-scale data. In this paper, inspired by denoising diffusion probabilistic model (DDPM), we propose a novel generative model, termed SPD-DDPM, by introducing Gaussian distribution in the SPD space to estimate $E(X|y)$. Moreover, our model can estimate $p(X)$ unconditionally and flexibly without giving y. On the one hand, the model conditionally learns $p(X|y)$ and utilizes the mean of samples to obtain $E(X|y)$ as a prediction. On the other hand, the model unconditionally learns the probability distribution of the data $p(X)$ and generates samples that conform to this distribution. Furthermore, we propose a new SPD net which is much deeper than the previous networks and allows for the inclusion of conditional factors. Experiment results on toy data and real taxi data demonstrate that our models effectively fit the data distribution both unconditionally and conditionally.

Introduction

Symmetric positive definite (SPD) matrices hold significant importance within the domains of machine learning and multivariate statistical analysis. They play a pervasive role across a diverse range of applications, including FMRI analysis (Petersen and Müller 2019), hand gesture recognition (Nguyen et al. 2019), facial recognition (Otberdout et al. 2019), traffic prediction (Tucker, Wu, and Müller 2023), brain-computer interfaces (Barachant et al. 2013), video classification (Wang et al. 2023a; Huang and Van Gool 2017; Wang et al. 2022), interaction recognition (Nguyen 2021), and many more.

There are two main tasks for SPD matrices: *classification* and *SPD matrix estimation*. The first one involves using SPD matrices as inputs to predict the classes of matrices. They often involve transforming SPD matrices into Euclidean space vectors via a Log mapping and then employ SPDNet accompanied by convolution with activation (Higham 2008) and batch normalization (Brooks et al. 2019) for classification. Recently, several works have been proposed to modify SPD Net via multi-scale features, UNet structure, and local convolution (Chen et al. 2023; Wang et al. 2022; Zhang et al. 2020). The second one aims to estimate the SPD matrix as the predicted variable X. Previous methods often use traditional statistical modeling frameworks as discriminative models by estimating $E(X|y)$ given the conditional variables y. For example, they define an additive model in a reproducing kernel Hilbert space using kernel functions (Lin, Müller, and Park 2023) or define a regression model in a metric space upon Fréchet mean (Petersen and Müller 2019; Qiu, Yu, and Zhu 2022). However, these models often face challenges when dealing with high-dimensional SPD matrices or predictor y. In addition, these methods primarily focus on predicting $E(X|y)$ and currently lack approaches for estimating the probability density in the SPD space. This motivates the construction of generative models for the SPD space. Such models not only benefit from predicting $p(X)$ but also utilize maximum likelihood estimation to make predictions for X given conditional variables y.

Denoising diffusion probabilistic model (DDPM) (Ho, Jain, and Abbeel 2020) is one of the most popular and effective generative models. Several researchers have extended DDPM to Riemannian manifolds, including SO(3) (Leach et al. 2022), SE(3) (Yim et al. 2023), spheres (Huang et al. 2022), and more (De Bortoli et al. 2022). These extensions have proven to be of practical value in fields such as protein generation (Corso et al. 2022), molecular docking (Corso et al. 2022), and others. However, few works have discussed and studied the SPD space. Meanwhile, the classical DDPM has poor performance when estimating probability distribution in the SPD space. When real data resides on a low-dimensional manifold, estimating in a high-dimensional Euclidean space is inefficient, and the distribution on the manifold may not exhibit favorable characteristics in the Euclidean space where it is embedded.

To address the SPD space distribution estimation prob-

[*]These authors contributed equally.
[†]Corresponding author

lem, we propose a novel denoising diffusion probabilistic model in the SPD space, termed SPD-DDPM. The key components of SPD-DDPM lie in the equation of the backward process and the network to generate SPD matrices. Firstly, by introducing Gaussian distribution, addition and multiplication operations in the SPD space, we extend the DDPM in the Eucidean space to the SPD space, including the forward and backward process. Secondly, we introduce a novel SPD net that takes an SPD matrix as input using double convolutions and allowing for the incorporation of conditions y into the network.

Our contributions are summarized as follows:

- We propose a novel SPD-DDPM to accurately estimate the SPD matrix X, which provides two versions: conditional and unconditional generation.
- We introduce the Gaussian distribution in the SPD space, the addition and multiplication operations in the SPD space to extend the DDPM theory.
- A new SPD U-Net is proposed to effectively incorporate the conditions during the generative process.

Experiment results on toy data and real taxi data demonstrate that our models effectively fit the data distribution both unconditionally and conditionally and provide accurate predictions.

Preliminary

Denoising Diffusion Probabilistic Model (DDPM)

Assume the source data X is a distribution $X \sim q(X)$. DDPM defines a forward process of T steps that gradually adds noise to transform the source data into a standard Gaussian distribution by:

$$q(X_t|X_{t-1}) = N(\sqrt{1-\beta_t}X_{t-1}, \beta_t I), \tag{1}$$

where X_t is the forward output at the t-th step, and β_t is a hyperparameter. Eq. 1 can be reparameterized as:

$$X_t = \sqrt{1-\beta_t}X_{t-1} + \sqrt{\beta_t}\epsilon_t,\ \epsilon_t \sim N(0, I). \tag{2}$$

After that, we build the distribution of $q(X_t|X_0)$ as:

$$\begin{aligned} q(X_t|X_0) &= N(\sqrt{\bar{\alpha}_t}X_0, (1-\sqrt{\bar{\alpha}_t})I), \\ \alpha_t &= 1-\beta_t,\ \bar{\alpha}_t = \prod_{i=0}^{t}\alpha_i. \end{aligned} \tag{3}$$

Note that the forward process does not require training. After T forward steps, the source data X, *a.k.a.* X_0, is transformed into a normal distribution.

In the backward process, DDPM attempts to solve $q(X_{t-1}|X_t, X_0)$ based on Bayes theorem:

$$\begin{aligned} q(X_{t-1}|X_t, X_0) &= q(X_t|X_{t-1})\frac{q(X_{t-1}|X_0)}{q(X_t|X_0)} \\ &= N(\mu_t(X_t, X_0), \sigma_t^2 I), \\ \mu_t(X_t, X_0) = \frac{1}{\sqrt{\alpha_t}}&(X_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\epsilon),\ \sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t. \end{aligned} \tag{4}$$

where $\epsilon \sim N(0, I)$ represents the noise added to X_t, but we do not known its exact value when given X_t. Thus, a neural network ϵ_θ is used to approximate it. The loss function can be formulated as:

$$\|\epsilon - \epsilon_\theta(X_t, t)\|^2 \tag{5}$$

According to Eq. 4, the reverse process is constructed to obtain $p(X_{t-1}|X_t)$:

$$p(X_{t-1}|X_t) = N\big(\frac{1}{\sqrt{\alpha_t}}X_t - \frac{1-\alpha_t}{\sqrt{\alpha_t(1-\bar{\alpha}_t)}}\epsilon_\theta, \Sigma_t\big), \tag{6}$$

where $\Sigma_t = \sigma_t^2 I$.

During inference, the data X_T is sampled from the standard Gaussian distribution and then employ Eq. 6 to gradually remove noise to generate the desired output of $\hat{X}$.

Symmetric Positive Definite (SPD) Space

Various real data (*e.g.*, FRMI) are satisfied with the SPD condition to form SPD matrices, which is significantly different from the data in the Euclidean space. In the SPD space (denoted by Sym^+), symmetry and positive definiteness are held:

$$\begin{aligned} &Symmetry: \forall X \in Sym^+, X^{'} = X, \\ &Positive: \forall X \in Sym^+, \forall a \in R^m, a^{'}Xa > 0. \end{aligned} \tag{7}$$

Following (Said et al. 2017), Gaussian distribution in the SPD space is defined as:

$$p(X|\bar{X}, \sigma^2) = \frac{1}{\zeta(\sigma)}\exp[-\frac{d(X, \bar{X})^2}{2\sigma^2}], \tag{8}$$

where $\zeta(\sigma)$ is the regularization coefficient. The distance d in Eq. 8 is affine-invariant metric (Moakher 2005), which is defined as:

$$d(X_1, X_2)^2 = tr[\log(X_1^{-0.5}X_2X_1^{-0.5})]^2. \tag{9}$$

Based on the above definition of affine-invariant metric, the exponent and logarithm mappings are formulated by following the work (Higham 2008):

$$\begin{aligned} \mathrm{Exp}_{X_1}(X_2) &= X_1^{\frac{1}{2}}\exp(X_1^{-\frac{1}{2}}X_2X_1^{-\frac{1}{2}})X_1^{\frac{1}{2}}, \\ \mathrm{Log}_{X_1}(X_2) &= X_1^{\frac{1}{2}}\log(X_1^{-\frac{1}{2}}X_2X_1^{-\frac{1}{2}})X_1^{\frac{1}{2}}. \end{aligned} \tag{10}$$

In the SPD space, we can employ the exponent and logarithm mappings in Eq. 10 to further define the addition and multiplication operations in SPD space for computation:

In the SPD space, the base matrix we select is identity matrix I, and the addition operation $\oplus$ and multiplication operation $\odot$ can be defined as:

Definition 1

$$\begin{aligned} X_1 \oplus X_2 &= \mathrm{Exp}_I(\mathrm{Log}_I(X_1) + \mathrm{Log}_I(X_2)). \\ r \odot X &= \mathrm{Exp}_I(r \cdot \mathrm{Log}_I(X)). \end{aligned} \tag{11}$$

Since the base matrix I used in Eq. 10, the exponent and logarithm mappings are equal to $\exp$ and $\log$ function. Thus, addition and multiplication operations in **Definition** 1 are reformulated as:

$$\begin{aligned} X_1 \oplus X_2 &= \exp(\log(X_1) + \log(X_2)) \\ &= \exp(\frac{1}{2}\log(X_1) + \log(X_2) + \frac{1}{2}\log(X_1)) \\ &= \exp(\log(X_1^{\frac{1}{2}}) + \log(X_2) + \log(X_1^{\frac{1}{2}})) \\ &= \exp(\log(X_1^{\frac{1}{2}}X_2X_1^{\frac{1}{2}})) = X_1^{\frac{1}{2}}X_2X_1^{\frac{1}{2}}, \end{aligned} \tag{12}$$

$$r \odot X = \exp(r \cdot \log(X)) = \exp(\log(X^r)) = X^r. \tag{13}$$

In the work (Terras 2012), SPD space with affine-invariant metric is a homogeneous space under the action of the linear group $GL(m)$, where the group action is defined as:

$$(X, A) \to X \cdot A \overset{def}{=} A^{'}XA. \tag{14}$$

Therefore, the affine-invariant metric remains invariant under group actions:

$$d(X_1, X_2)^2 = d(X_1 \cdot A, X_2 \cdot A)^2. \tag{15}$$

Based on Eqs. 15, 8, and 9, we have the following proposition:

Proposition 1 $X \sim G(\bar{X}, \sigma^2) \implies X \cdot A \sim G(\bar{X} \cdot A, \sigma^2)$.

Proof:

For any function $f : P_m \to R$ (P_m is SPD space), and any $A \in GL(m)$, according to Eq.14, we have:

$$\int_{P_m} f(X)dv(X) = \int_{P_m} f(X \cdot A)dv(X).$$

where $dv(X)$ is the Riemannian volume element defined as:

$$dv(X) = det(X)^{-\frac{m+1}{2}} \prod_{i \leq j} dX_{ij}.$$

Intuitively, $dv(X)$ is the Riemannian volume element of $ds^2(X)$, and $ds^2(X)$ is invariant under congruence transformations and inversion:

$$ds^2(X) = tr[X^{-1}dX]^2.$$

Let X be a random variable in P_m, let $\phi : P_m \to R$ be a test function. If $X \sim G(\bar{X}, \sigma^2)$ and $Z = X \cdot A$, then the expection of $\phi(Z)$is given by:

$$\begin{aligned}
&\int_{P_m} \phi(X \cdot A)p(X|\bar{X}, \sigma^2)dv(X) \\
&= \int_{P_m} \phi(Z)p(Z \cdot A^{-1}|\bar{X}, \sigma^2)dv(Z) \\
&= \int_{P_m} \phi(Z)p(Z|\bar{X} \cdot A, \sigma^2)dv(Z).
\end{aligned}$$

Method

Symmetric Positive Definite Denoising Diffusion Probabilistic Model (SDP-DDPM)

Different from DDPM, which is computed in Euclidean space, we propose SDP-DDPM based on the operational rules (*i.e.*, Eq. 12 and Eq. 13) and Proposition 1 in the SPD space.

In the forward process, following DDPM, we can formulate $q(X_t|X_{t-1})$ in the SPD space as:

$$q(X_t|X_{t-1}) \sim G(\alpha_t \odot X_{t-1}, \beta_t^2), \tag{16}$$

where $\alpha_t^2 + \beta_t^2 = 1$, and $G(\cdot, \cdot)$ is gaussian distribution in the SPD space defined in Eq. 8. By using Eq. 12, Eq. 13) and Proposition 1, we have:

$$\begin{aligned}
X_t &= \alpha_t \odot X_{t-1} \oplus \beta_t \odot \epsilon_t, \quad \epsilon_t \sim G(I, 1) \\
&= X_{t-1}^{\frac{\alpha_t}{2}} \, \epsilon_t^{\beta_t} \, X_{t-1}^{\frac{\alpha_t}{2}} \sim G(\alpha_t \odot X_{t-1}, \beta_t^2).
\end{aligned} \tag{17}$$

By using Eq. 17, we can obtain the relationship between X_t and X_0 as below:

$$\begin{aligned}
X_t &= \alpha_t \odot X_{t-1} \oplus \beta_t \odot \epsilon_t \\
&= \alpha_t \odot (\alpha_{t-1} \odot X_{t-2} \oplus \beta_{t-1} \odot \epsilon_{t-1}) \oplus \beta_t \odot \epsilon_t \\
&= \alpha_t \alpha_{t-1} \odot X_{t-2} \oplus \alpha_t \beta_{t-1} \odot \epsilon_{t-1} \oplus \beta_t \odot \epsilon_t \\
&= (\alpha_t \dots \alpha_1) \odot X_0 \oplus (\alpha_t \dots \alpha_2)\beta_1 \epsilon_1 \oplus \dots \oplus \beta_t \epsilon_t \\
&= \bar{\alpha}_t \odot X_0 \oplus \sqrt{1 - \bar{\alpha}_t^2} \odot \epsilon, \ \epsilon \sim G(I, 1) \\
&\sim G(\bar{\alpha}_t \odot X_0, 1 - \bar{\alpha}_t^2), \ \bar{\alpha}_t = \prod_{i=0}^{t} \alpha_i.
\end{aligned} \tag{18}$$

By defining $\bar{\beta}_t$ to be $\sqrt{1 - \bar{\alpha}_t^2}$, X_t can be further formulated as:

$$X_t = \bar{\alpha}_t \odot X_0 \oplus \bar{\beta}_t \odot \epsilon. \tag{19}$$

Based on Eq. 19, we only set $\bar{\alpha}_t$ to approach 0, X_t will be transformed to the standard Gaussian distribution in the SPD space. Empirically, α_t is set to $\sqrt{1 - \frac{0.08t}{T}}$ in our experiments, such that $\bar{\alpha}_t \to 0$ when $t \to \infty$. Therefore, we can randomly sample the noise from the standard Gaussian distribution in the SPD space, and use the following backward process to generate the desired output.

In the backward process, we also compute $q(X_{t-1}|X_t, X_0)$ using the Bayesian technique as:

$$\begin{aligned}
&q(X_{t-1}|X_t, X_0) = q(X_t|X_{t-1}) \frac{q(X_{t-1}|X_0)}{q(X_t|X_0)} \\
&\propto \exp\left(-\frac{d(X_t, X_{t-1}^{\alpha_t})^2}{2\beta_t^2} - \frac{d(X_{t-1}, X_0^{\bar{\alpha}_{t-1}})}{2\bar{\beta}_{t-1}^2}\right) \\
&\propto \exp\left[-\frac{tr(\bar{\beta}_{t-1}^2 \alpha_t^2 + \beta_t^2)(\log \tilde{\mu}^{-\frac{1}{2}} \, X_0 \, \tilde{\mu}^{-\frac{1}{2}})^2}{2\bar{\beta}_{t-1}^2 \beta_t^2}\right] \\
&\sim G(\tilde{\mu}(X_0, \epsilon), \tilde{\sigma}_t^2), \\
&\text{with} \quad \tilde{\mu}(x_0, \epsilon) = \frac{1}{\alpha_t} \odot X_t \ominus \frac{\beta_t^2}{\alpha_t \bar{\beta}_t} \odot \epsilon, \tilde{\sigma}_t = \frac{\bar{\beta}_{t-1}\beta_t}{\bar{\beta}_t}.
\end{aligned} \tag{20}$$

Note that the final computation expression of $q(X_{t-1}|X_t, X_0)$ is similar to DDPM (*i.e.*, Eq. 4), while their difference is that our addition and scalar multiplication are defined through exponential and logarithmic mappings. In Euclidean space, the exponential and logarithmic mappings can be regarded as addition and subtraction operations.

Further, we also formulate $p(X_{t-1}|X_t)$ as:

$$p(X_{t-1}|X_t) \sim G(\mu_\theta(X_t, t), \Sigma_t^2), \tag{21}$$

where $\Sigma_t^2 = \beta_t^2$. $\mu_\theta(X_t, t)$ can be computed according to $q(X_{t-1}|X_t, X_0)$:

$$\mu_\theta(X_t, t) = \frac{1}{\alpha_t} \odot X_t \ominus \frac{\beta_t^2}{\alpha_t \bar{\beta}_t} \odot \epsilon_\theta. \tag{22}$$

During training, we aim to minimize the KL divergence between $p(X_{t-1}|X_t)$ and $q(X_{t-1}|X_t, X_0)$. We also found that $\mathrm{KL}(q(X_{t-1}|X_t, X_0) \ \| \ p(X_{t-1}|X_t))$ is bounded by

$d(\mu_{X_0,\epsilon},\mu_\theta)^2$, *i.e.*,

$$\begin{aligned}
&KL(q(X_{t-1}|X_t,X_0)\parallel p(X_{t-1}|X_t))\\
&\propto \frac{2\beta_t^2 d(\mu_\theta,X)^2-2\tilde{\sigma}_t^2 d(\mu_{X_0,\epsilon},X)}{2\beta_t^2\tilde{\sigma}_t^2}\\
&\propto \frac{2\beta_t^2[d(\mu_\theta,X)^2-d(\mu_{x_0,\epsilon},X)^2]}{2\beta_t^2\tilde{\sigma}_t^2}\\
&\leq \kappa\, d(\mu_{X_0,\epsilon},\mu_\theta)^2,
\end{aligned} \tag{23}$$

where κ is a constant, and $d(\mu_{X_0,\epsilon},\mu_\theta)^2$ is further simplified as:

$$\begin{aligned}
&d(\mu_{X_0,\epsilon},\mu_\theta)^2\\
&=d(\frac{1}{\alpha_t}\odot X_t\ominus\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}\odot\epsilon,\ \frac{1}{\alpha_t}\odot X_t\ominus\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}\odot\epsilon_\theta)^2\\
&=d(X_t^{-\frac{1}{2\alpha_t}}\epsilon^{\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}}X_t^{-\frac{1}{2\alpha_t}},X_t^{-\frac{1}{2\alpha_t}}\epsilon_\theta^{\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}}X_t^{-\frac{1}{2\alpha_t}})^2\\
&=d(\epsilon^{\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}},\epsilon_\theta^{\frac{\beta_t^2}{\alpha_t\bar{\beta}_t}})^2\\
&=(\frac{\beta_t^2}{\alpha_t\bar{\beta}_t})^2 d(\epsilon,\epsilon_\theta)^2.
\end{aligned} \tag{24}$$

For computational simplicity, we use $d(\epsilon,\epsilon_\theta)^2$ as the training objective function. If we degenerate it to the Euclidean space, it will be L_2 distance to train DDPM. The detailed training process of SPD-DDPM is presented in Alg. 1.

For inference, we start from the standard Gaussian distribution in SPD space and gradually denoise the sample to generate $\hat{X}_0$, whose process is presented in Alg. 2. γ is used to balance the diversity and robustness. A smaller γ leads to greater diversity in the generated matrix, but it may result in low quality. On the other hand, a larger γ produces higher quality outputs, but with reduced diversity.

Alg. 1, and 2 present the unconditional training of SPD-DDPM, while our SPD-DDPM also works well under the conditional generation. In the following, we will first describe the Symmetric Positive Definite U-Net to fit ϵ and then propose the conditional process of SPD-DDPM.

Algorithm 1: Unconditional Training of SPD-DDPM.

1: **repeat**
2: $t\sim$ Uniform$(1,\cdots,T)$
3: $X_0\sim q(X_0)$
4: $\epsilon\sim G(I,1)$
5: $X_t=\bar{\alpha}_t\odot X_0\oplus\bar{\beta}_t\odot\epsilon$
6: Take gradient descent step on $\nabla_\theta\parallel\epsilon-\epsilon_\theta(X_t,t)\parallel$
7: **until** convergence

Algorithm 2: Unconditional Sampling of SPD-DDPM.

1: $X_T\sim G(I,1)$
2: **for** $t=T,\dots 1$ **do**
3: $z\sim G(I,1)$
4: $X_{t-1}=\frac{1}{\alpha_t}\odot(X_t\ominus\frac{\beta_t^2}{\bar{\beta}_t}\epsilon_\theta)\oplus\tilde{\sigma_t}/\gamma\odot z$, where $\ominus$ is defined by using $-$ to replace $+$ in Eq. 1.
5: **end for**
6: **return** X_0

The detailed proofs presented in this section can be found in https://arxiv.org/abs/2312.08200.

Symmetric Positive Definite U-Net

In SPD-DDPM, a neural network ϵ_θ takes an SPD matrix and time t as the inputs to output an SPD matrix. Traditional U-Net in DDPM can not guarantee the symmetry and positive definiteness of the output.Huang *et al.* (Huang and Van Gool 2017) proposed a simple SPD Net by defining the convolution and activation layers. However, the objective of this network framework is to transform the SPD matrix into a vector in Euclidean space, and this network has a limited capacity to fit ϵ. Moreover, the above SPD Net can not incorporate the conditional factors t. It fails to adapt to the different distributions of X_t at different moments.

In this paper, we propose a novel high-capacity SPD U-Net framework to estimate the SPD matrix. We first follow the convolution and activation layer defined in (Huang and Van Gool 2017) as our SPD U-Net computation operations, which are defined by:

$$X_k=f_b^{(k)}(X_{k-1};W_k)=W_kX_{k-1}W_k^{'}, \tag{25}$$

$$X_k=f_r^{(k)}(X_{k-1})=U_{k-1}\max(\epsilon I,\Sigma_{k-1})U_{k-1}^{'}. \tag{26}$$

Furthermore, we add the time factor t by first putting it through an embedding layer following a shallow neural network to generate a matrix E_t. As shown in Fig. 1, we incorporate the time factor t into the neural network using the following equation:

$$X_k=E_tX_{k-1}E_t^{'}. \tag{27}$$

To better estimate the distribution of source data, we increase the network depth by adding double convolution as shown in Fig. 1. The SPD matrix passes through a convolution layer using 25 and then adds condition t by 27. After that, we do activation using 26. We repeat the above computation operation once more as a basic unit, termed double convolution, which increases the network depth and enhances its fitting capability.

The overall framework of our network is shown in Fig. 2. The down convolution is defined as $W_k\in R^{d_k\times d_{k-1}}(d_k<d_{k-1})$. The up convolution is defined as filling the diagonal of the input matrix with 1 to match the output size, and then performing convolution (25). The concatenation layer is defined as computing the mean of two layers.

Different from Wang *et al.* (Wang et al. 2023a), our neural network is deeper and allows for conditional additions. Additionally, the objective of our network is not to reconstruct the input SPD matrix but to fit other SPD matrices instead. We will conduct ablation experiments to evaluate the effectiveness of each component.

Conditional Symmetric Positive Definite Denoising Diffusion Probabilistic Model

We employ the classifier-free framework in the conditional generation, which additionally adds the conditional vector y in the SPD net. Similar to the incorporation process of the time factor t, the conditional vector t is transformed into a matrix via an embedding layer with a shallow network. Thus, the training loss function in the conditional

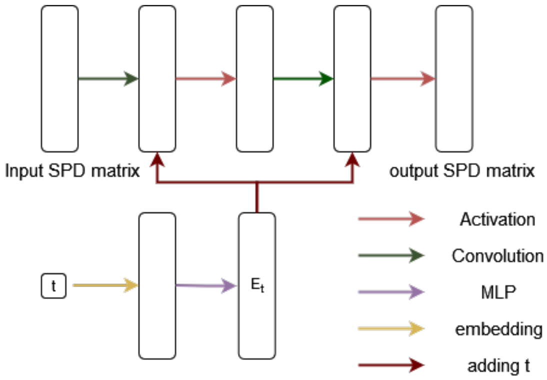

Figure 1: Double Convolution in SPD U-Net. Input matrix passes through convolution layers, followed by adding the condition of time factor t, and then activation layers. Repeat the above process making the double convolution block.

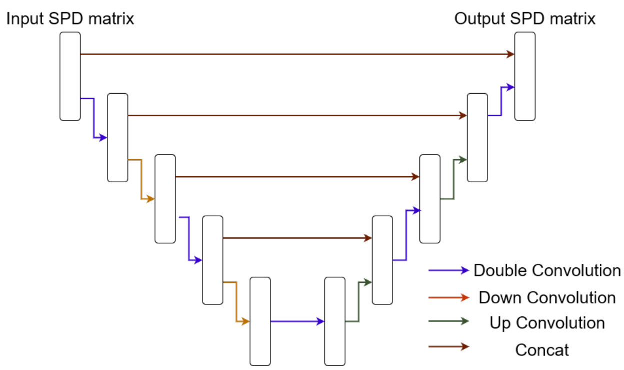

Figure 2: Architecture of SPD U-Net. SPD U-Net contains five double convolutions, two down convolution and two up convolution layers.

SPD-DDPM can be formulated as:

$$\parallel \epsilon - \epsilon_\theta(X_t, t, y) \parallel . \tag{28}$$

As presented in Alg. 3, the training process is similar to the unconditional SPD-DDPM.

In conditional generation, we aim to incorporate conditional generation as a novel approach for regression modeling. In regression models, it is common to assume that X given y follows a certain distribution and use $E(X|y)$ as the prediction of X given y. In this work, we assume $X|y \sim G(\mu_y, \sigma^2)$ with μ_y being $E(X|y)$. Said *et al.* (Said et al. 2017) have deduced that the maximum likelihood estimation of $E(X|y)$ is the empirical Riemannian centre of samples from $P(X|y)$, where the Riemannian centre $\hat{X}_N$ is defined as:

$$\hat{X}_N = \arg\min_X \frac{1}{N} \sum_{n=1}^{N} d(X, X^n)^2. \tag{29}$$

$\hat{X}_N$ under the affine-invariant metric in Eq. 29 can be obtained using a gradient descent algorithm (Dryden, Koloydenko, and Zhou 2009; Pennec 1999; Pennec, Fillard, and Ayache 2006). Therefore, we train the model to fit $p(X|y)$.

Algorithm 3: Conditional Training of SDP-DDPM.

1: **repeat**
2: $\quad t \sim \text{Uniform}(1, \cdots, T)$
3: $\quad X_0 \sim q(X_0)$
4: $\quad \epsilon \sim G(I, 1)$
5: $\quad X_t = \bar{\alpha}_t \odot X_0 \oplus \bar{\beta}_t \odot \epsilon$
6: $\quad$ Take gradient descent step on
$\quad \nabla_\theta \parallel \epsilon - \epsilon_\theta(X_t, t, y) \parallel$
7: **until** convergence

Algorithm 4: Conditional Inference of SDP-DDPM.

1: **for** $i = 1, \ldots, N$ **do**
2: $\quad X_T^i \sim G(I, 1)$
3: $\quad$ **for** $t = T, \ldots 1$ **do**
4: $\quad\quad z \sim G(I, 1)$
5: $\quad\quad X_{t-1}^i = \frac{1}{\alpha_t} \odot (X_t^i \ominus \frac{\beta_t^2}{\bar{\beta}_t} \epsilon_\theta) \oplus \tilde{\sigma}_t / \gamma \odot z$
6: $\quad$ **end for**
7: **end for**
8: $\hat{X} = \arg\min_X \frac{1}{N} \sum_{n=1}^{N} d^2(X, X^n)$
9: **return** $\hat{X}$

During inference, given a specific y, we first generate N samples from $p(X|y)$. We then compute $\hat{X}_N$ by minimizing Eq. (29) as the prediction of $E(X|y)$. The detailed inference process is presented in Alg. 4.

Experiments

Experimental Setups

Datasets. We use the simulation or toy data to test the performance of unconditional SPD-DDPM. We first randomly select a matrix A as the center point of the distribution, and then sample total $15,000$ SPD matrices from $G(A, \sigma^2)$ as the training data.

In the conditional generation, taxi data from the New York City Taxi and Limousine Commission is selected, which is available in [1] Data preprocessing is following (Tucker, Wu, and Müller 2023). New York City is divided into 10 boroughs. Each element of the SPD matrix X_{ij} represents the passenger flow between the i-th borough and the j-th borough. We obtain $8,700$ weighted adjacency matrices by collecting data from 2019.1 to 2019.12 and also collect the 13 predictors following (Tucker, Wu, and Müller 2023). We select $7,600$ samples as the training set and $1,100$ samples as the testing set.

Evaluation metric. In the unconditional generation, the Affine-invariant metric is widely used to test performance:

$$d(A, B)^2 = tr[\log(A^{-0.5} B A^{-0.5})]^2.$$

In the condition generation, there is no previous method that optimizes under the Affine-invariant metric, so we select both Frobenius and Affine-invariant metric as the evaluation

[1] https://www.nyc.gov/site/tlc/about/tlc-trip-record-data.page.

Method	Mean Distance↓
DDPM	813.75± 187.92
SPD-DDPM	23.17± 3.00

Table 1: Mean distance between the generated samples and the real distribution in the unconditional generation. $a \pm b$ means the mean and std in all tables by running the model 5 times, respectively.

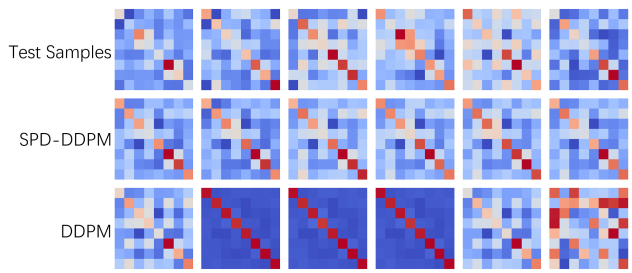

Figure 3: Visualization of SPD samples. We randomly select six samples from test samples, generated by SPD-DDPM and DDPM. Each column presents one sample.

protocol, where Frobenius distance is defined as:

$$d(A, B)^2 = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{m}(A_{ij} - B_{ij})^2}.$$

Implementation details. We implement our method using PyTorch 2.0.1 (Paszke et al. 2017) with one NVIDIA A6000 GPU. The models are trained using Adam optimizer (Kingma and Ba 2014) with default parameters to DDPM. The learning rate is initialized by 0.0015 decaying by each iteration with the cosine function. We set α_t and T to $\sqrt{1 - \frac{0.08t}{T}}$ and 200, respectively. γ in Alg. 2 and 4 is both default set to 10 unless the specific discuss. Code is available at https://github.com/li-yun-chen/SPD-DDPM.

Unconditional Generation

We generate 300 SPD matrices and use a heatmap to visualize the training samples and generated samples. Each heatplot shows the value of one SPD matrix. To evaluate the effectiveness of our unconditional SPD-DDPM, we compare our model with DDPM.

We first calculate the mean distance between the generated samples and the real distribution as a measure of evaluating the quality of generation. The results are summarized in Tab. 1. compared to DDPM, our unconditional SPD-DDPM achieves a lower mean distance. Furthermore, we visualize the generated samples in Fig. 3. Obviously, our models are better to fit the data, compared to DDPM. In this experiment, our data in the SPD space is a low-dimensional manifold, directly using DDPM to generate samples cannot be effective for the generation. In contrast, our unconditional SPD-DDPM fully considers the SPD information for better generation.

Method	Frobenius ↓	Affine-invariant ↓
Frechet Regression	6.6448	8.0099
DDPM	62.97	1203.10
SPD-DDPM	6.4684	0.6123

Table 2: Comparsion among Frechet Regression, DDPM and SPD-DDPM for conditional generation.

Conditional Generation

Our goal is to use predictor variables y to generate responding SPD matrices, achieving generative predictive modeling. Practically, we generate $n = 20$ samples for each predictor y. We compare our method with the discriminative method Frechet Regression (Petersen and Müller 2019) and DDPM. Our method is optimized under the Affine-invariant metric, while Frechet Regression is optimized under the Forbinus metric. The results are summarized in Tab. 2. Our method outperforms Frechet Regression in both Frobenius and Affine-invariant metrics. In the Frobenius metric, our method slightly outperforms Frechet Regression because it is optimized under the Frobenius norm. In the Affine-invariant metric, our method significantly outperforms Frechet Regression. Our conditional SPD-DDPM also achieves significantly lower Frobenius metric, compared to DDPM.

To further evaluate the effectiveness of our method, we visualize the SPD matrices in the Manhattan map Fig. 4. The left visualization represents the Frechet Regression, followed by the visualizations for our method and real data. We found that the predictions of Frechet Regression are overestimated and fail to capture the differences between each edge effectively. While our method shows a closer approximation to the real matrix.

Actually, discriminative methods require inputting all the data into the model simultaneously, rather than training in batches. When dealing with large-scale data, it will exceed memory limitations. Furthermore, their approach lacks the capability to retain intermediate model parameters, necessitating retraining the model when new samples need estimation. Our approach is based on the neural network, allowing us to circumvent the issues mentioned above.

Ablation Study

In this ablation study, we mainly evaluate the effects of hyperparameter γ and SPD U-Net, which are very important to SPD-DDPM.

Effect of γ. We evaluate the effect of γ in unconditional generation. As shown in Tab. 3, larger γ generates a lower mean distance to better estimate the inputs. In addition, the std is also reduced by increasing γ.

Effect of SPD U-Net. In this paper, we propose a high-capacity SPD U-Net. Compared to the traditional SPD net, the network proposed in this paper is deeper and incorporates a double convolution, down and up convolution, and concatenation structure. We further demonstrate the effectiveness of these modifications through ablation experiments in unconditional generation by comparing the mean distance

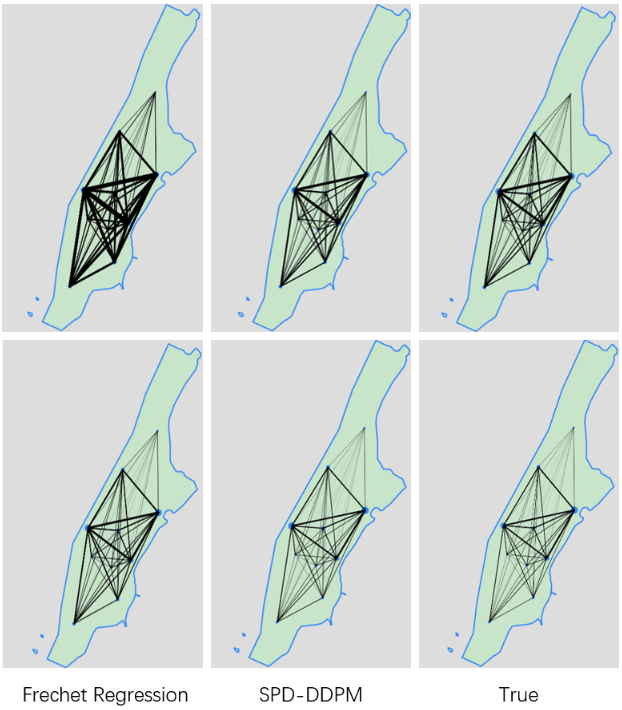

Figure 4: Visualization of taxi data. On the left is the data predicted by Frechet Regression, in the middle is the data predicted by our method, and on the right is the real data. Each node represents a region, and the thickness of the line segments represents the number of taxis.

Method	Mean Distance↓
SPD-DDPM(γ = 2)	65.81± 17.86
SPD-DDPM(γ = 4)	35.99± 7.58
SPD-DDPM(γ = 10)	23.17± 3.00

Table 3: Effect of γ for unconditional generation.

between samples and the distribution mean under $\gamma = 10$.

If we replace the double convolution with a single layer, reducing the number of convolutional layers by half, the performance of model generation decreases 3 mean distance. It indicates the effectiveness of double convolution for network's fitting capability. The generation will fails when removing the concatenation structure. Thus, concatenation operations play a vital role in our SPD U-Net. Removing the down and up convolutions, each layer uses the matrices with the same dimensions, and the distance will increase to 42.75, which significantly affects the model's generalization ability.

Related Work

Denoising Diffusion Probabilistic Model

DDPM is a generative model proposed by Ho *et al.* (Ho, Jain, and Abbeel 2020). Nichol *et al.* (Nichol and Dhariwal 2021) and Song *et al.* (Song, Meng, and Ermon 2020) further made improvements to it. The diffusion model is first applied in computer vision and found numerous applications in conditional image generation (Nichol et al. 2021; Rombach et al. 2022), image restoration (Wang, Yu, and Zhang 2022; Lugmayr et al. 2022), 3D image generation (Wang et al. 2023b), and other fields. Song *et al.* (Song et al. 2020) introduced the score-based generation method to accelerate the generation speed of the diffusion model from the perspective of stochastic differential equations (Lu et al. 2022). Different with these methods, our SPD-DDPM is constructed on the SPD space, rather than Euclidean space. By introducing Gaussian distribution, the addition and multiplication operations in the SPD space, we effectively generate the forward and backward processes in the SPD space for better estimating the SPD matrix unconditionally and conditionally.

Method	Mean distance↓
Without double convolution	26.17± 3.75
Without concatenation	–
Without down&up convolution	42.75± 3.25
SPD-DDPM with SPD U-Net	23.17± 3.00

Table 4: Ablation Study of SPD U-Net.

Symmetric Positive Definite Metric

SPD space is a nonlinear metric space. Depending on the metric, SPD forms a Riemannian manifold. Different matrices have been proposed, such as affine-invariant metric (Moakher 2005; Pennec 2006; Fletcher and Joshi 2007), log-Euclidean metric (Arsigny et al. 2007), log-Cholesky metric (Lin 2019) and scaling rotation distance (Jung, Schwartzman, and Groisser 2015). Some work studied regression with SPD matrix valued responses. Using Frechet mean (Fréchet 1948), Petersen *et al.* (Petersen and Müller 2019) proposed Frechet Regression under different matric. Based on this, Tucker *et al.* (Tucker, Wu, and Müller 2023) proposed a method for variable selection. Lin *et al.* (Lin, Müller, and Park 2023) proposed an adaptive model in SPD space. Qiu *et al.* (Qiu, Yu, and Zhu 2022) propose random forest with SPD matrix response. Different from these methods, our SPD-DDPM is the first generation model in SPD space, which can fit probability distribution in SPD space.

Conclusion

This paper proposes a novel denoising diffusion probabilistic model in the SPD space, termed SDP-DDPM. The proposed SDP-DDPM is applied for both unconditional and conditional generation. In the unconditional version, SDP-DDPM fits the probability distribution and generates samples in SPD space. In the conditional version, the model generates the distribution of SPD given a specific condition and calculates the expectation $E(X|y)$ as the prediction. A high-capacity SPD U-Net structure is further introduced to improve the data fitting performance. Experimental results demonstrate the strong capability of the proposed method in the fitting probability distributions.

Acknowledgements

This paper is sponsored by the National Natural Science Foundation of China (NO. 62102151, NO. 12371289), the Shanghai Sailing Program (21YF1411200), Basic Research Project of Shanghai Science and Technology Commission(NO. 22JC1400800), CCF-Tencent Rhino-Bird Open Research Fund, the Open Research Fund of Key Laboratory of Advanced Theory and Application in Statistics and Data Science, Ministry of Education (KLATASDS2305), and the Fundamental Research Funds for the Central Universities.

References

Arsigny, V.; Fillard, P.; Pennec, X.; and Ayache, N. 2007. Geometric means in a novel vector space structure on symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 29(1): 328–347.

Barachant, A.; Bonnet, S.; Congedo, M.; and Jutten, C. 2013. Classification of covariance matrices using a Riemannian-based kernel for BCI applications. *Neurocomputing*, 112: 172–178.

Brooks, D.; Schwander, O.; Barbaresco, F.; Schneider, J.-Y.; and Cord, M. 2019. Riemannian batch normalization for SPD neural networks. *Advances in Neural Information Processing Systems*, 32.

Chen, Z.; Xu, T.; Wu, X.-J.; Wang, R.; Huang, Z.; and Kittler, J. 2023. Riemannian Local Mechanism for SPD Neural Networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, 7104–7112.

Corso, G.; Stärk, H.; Jing, B.; Barzilay, R.; and Jaakkola, T. 2022. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*.

De Bortoli, V.; Mathieu, E.; Hutchinson, M.; Thornton, J.; Teh, Y. W.; and Doucet, A. 2022. Riemannian score-based generative modelling. *Advances in Neural Information Processing Systems*, 35: 2406–2422.

Dryden, I. L.; Koloydenko, A.; and Zhou, D. 2009. Non-Euclidean statistics for covariance matrices, with applications to diffusion tensor imaging.

Fletcher, P. T.; and Joshi, S. 2007. Riemannian geometry for the statistical analysis of diffusion tensor data. *Signal Processing*, 87(2): 250–262.

Fréchet, M. 1948. Les éléments aléatoires de nature quelconque dans un espace distancié. In *Annales de l'institut Henri Poincaré*, volume 10, 215–310.

Higham, N. J. 2008. *Functions of matrices: theory and computation*. SIAM.

Ho, J.; Jain, A.; and Abbeel, P. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33: 6840–6851.

Huang, C.-W.; Aghajohari, M.; Bose, J.; Panangaden, P.; and Courville, A. C. 2022. Riemannian diffusion models. *Advances in Neural Information Processing Systems*, 35: 2750–2761.

Huang, Z.; and Van Gool, L. 2017. A riemannian network for spd matrix learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 31.

Jung, S.; Schwartzman, A.; and Groisser, D. 2015. Scaling-rotation distance and interpolation of symmetric positive-definite matrices. *SIAM Journal on Matrix Analysis and Applications*, 36(3): 1180–1201.

Kingma, D. P.; and Ba, J. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Leach, A.; Schmon, S. M.; Degiacomi, M. T.; and Willcocks, C. G. 2022. Denoising diffusion probabilistic models on so (3) for rotational alignment. In *ICLR 2022 Workshop on Geometrical and Topological Representation Learning*.

Lin, Z. 2019. Riemannian geometry of symmetric positive definite matrices via Cholesky decomposition. *SIAM Journal on Matrix Analysis and Applications*, 40(4): 1353–1370.

Lin, Z.; Müller, H.-G.; and Park, B. 2023. Additive models for symmetric positive-definite matrices and Lie groups. *Biometrika*, 110(2): 361–379.

Lu, C.; Zhou, Y.; Bao, F.; Chen, J.; Li, C.; and Zhu, J. 2022. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35: 5775–5787.

Lugmayr, A.; Danelljan, M.; Romero, A.; Yu, F.; Timofte, R.; and Van Gool, L. 2022. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11461–11471.

Moakher, M. 2005. A differential geometric approach to the geometric mean of symmetric positive-definite matrices. *SIAM journal on matrix analysis and applications*, 26(3): 735–747.

Nguyen, X. S. 2021. Geomnet: A neural network based on riemannian geometries of spd matrix space and cholesky space for 3d skeleton-based interaction recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13379–13389.

Nguyen, X. S.; Brun, L.; Lézoray, O.; and Bougleux, S. 2019. A neural network based on SPD manifold learning for skeleton-based hand gesture recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12036–12045.

Nichol, A.; Dhariwal, P.; Ramesh, A.; Shyam, P.; Mishkin, P.; McGrew, B.; Sutskever, I.; and Chen, M. 2021. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*.

Nichol, A. Q.; and Dhariwal, P. 2021. Improved denoising diffusion probabilistic models. In *International Conference on Machine Learning*, 8162–8171. PMLR.

Otberdout, N.; Kacem, A.; Daoudi, M.; Ballihi, L.; and Berretti, S. 2019. Automatic analysis of facial expressions based on deep covariance trajectories. *IEEE transactions on neural networks and learning systems*, 31(10): 3892–3905.

Paszke, A.; Gross, S.; Chintala, S.; Chanan, G.; Yang, E.; DeVito, Z.; Lin, Z.; Desmaison, A.; Antiga, L.; and Lerer, A. 2017. Automatic differentiation in pytorch.

Pennec, X. 1999. Probabilities and statistics on Riemannian manifolds: Basic tools for geometric measurements. In *NSIP*, volume 3, 194–198.

Pennec, X. 2006. Intrinsic statistics on Riemannian manifolds: Basic tools for geometric measurements. *Journal of Mathematical Imaging and Vision*, 25: 127–154.

Pennec, X.; Fillard, P.; and Ayache, N. 2006. A Riemannian framework for tensor computing. *International Journal of computer vision*, 66: 41–66.

Petersen, A.; and Müller, H.-G. 2019. Fréchet regression for random objects with Euclidean predictors.

Qiu, R.; Yu, Z.; and Zhu, R. 2022. Random Forests Weighted Local Fr\'echet Regression with Theoretical Guarantee. *arXiv preprint arXiv:2202.04912*.

Rombach, R.; Blattmann, A.; Lorenz, D.; Esser, P.; and Ommer, B. 2022. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 10684–10695.

Said, S.; Bombrun, L.; Berthoumieu, Y.; and Manton, J. H. 2017. Riemannian Gaussian distributions on the space of symmetric positive definite matrices. *IEEE Transactions on Information Theory*, 63(4): 2153–2170.

Song, J.; Meng, C.; and Ermon, S. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502*.

Song, Y.; Sohl-Dickstein, J.; Kingma, D. P.; Kumar, A.; Ermon, S.; and Poole, B. 2020. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*.

Terras, A. 2012. *Harmonic analysis on symmetric spaces and applications II*. Springer Science & Business Media.

Tucker, D. C.; Wu, Y.; and Müller, H.-G. 2023. Variable selection for global Fréchet regression. *Journal of the American Statistical Association*, 118(542): 1023–1037.

Wang, R.; Wu, X.-J.; Chen, Z.; Xu, T.; and Kittler, J. 2022. DreamNet: A Deep Riemannian Manifold Network for SPD Matrix Learning. In *Proceedings of the Asian Conference on Computer Vision*, 3241–3257.

Wang, R.; Wu, X.-J.; Xu, T.; Hu, C.; and Kittler, J. 2023a. U-SPDNet: An SPD manifold learning-based neural network for visual classification. *Neural networks*, 161: 382–396.

Wang, Y.; Yu, J.; and Zhang, J. 2022. Zero-shot image restoration using denoising diffusion null-space model. *arXiv preprint arXiv:2212.00490*.

Wang, Z.; Lu, C.; Wang, Y.; Bao, F.; Li, C.; Su, H.; and Zhu, J. 2023b. ProlificDreamer: High-Fidelity and Diverse Text-to-3D Generation with Variational Score Distillation. *arXiv preprint arXiv:2305.16213*.

Yim, J.; Trippe, B. L.; De Bortoli, V.; Mathieu, E.; Doucet, A.; Barzilay, R.; and Jaakkola, T. 2023. SE (3) diffusion model with application to protein backbone generation. *arXiv preprint arXiv:2302.02277*.

Zhang, T.; Zheng, W.; Cui, Z.; Zong, Y.; Li, C.; Zhou, X.; and Yang, J. 2020. Deep manifold-to-manifold transforming network for skeleton-based action recognition. *IEEE Transactions on Multimedia*, 22(11): 2926–2937.