A. How Jacobian Matrix Forms a Constantly Scaled Orthonormal System

In this appendix, we derive equations corresponding to Eqs. (13) and (14) for the case of $M > N$. The guiding principle of derivation is the same as in Section 4.2: examining the condition to minimize the expected loss. As in Section 4.2, we assume that the encoder and the decoder are trained enough in terms of reconstruction error so that $\boldsymbol{x} \simeq \hat{\boldsymbol{x}}$ holds and the second term $\lambda_1 h\left(D\left(\boldsymbol{x}, \hat{\boldsymbol{x}}\right)\right)$ in Eq. (7) can be ignored.

We assume that the Jacobian matrix $\boldsymbol{J}(\boldsymbol{z}) = \partial \boldsymbol{x}/\partial \boldsymbol{z} = \partial g_\phi(\boldsymbol{z})/\partial \boldsymbol{z} \in \mathbb{R}^{M\times N}$ is full-rank at every point $\boldsymbol{z} \in \mathbb{R}^N$ as in Section 4.2. Based on Eq. (6), Eq. (4) and Taylor expansion, the difference $\breve{\boldsymbol{x}} - \hat{\boldsymbol{x}}$ can be approximated by $\boldsymbol{\epsilon} = \sum_{i=1}^N \epsilon_i (\partial \boldsymbol{x}/\partial z_i) \in \mathbb{R}^M$. As in Section 4.2, the expectation of the third term in Eq. (7) is re-written as follows:

$$\underset{\boldsymbol{\epsilon}\sim P_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon})}{E}\left[\boldsymbol{\epsilon}^\top \boldsymbol{A}(\boldsymbol{x})\boldsymbol{\epsilon}\right] = \sigma^2 \sum_{j=1}^N \left(\frac{\partial \boldsymbol{x}}{\partial z_j}\right)^\top \boldsymbol{A}(\boldsymbol{x}) \left(\frac{\partial \boldsymbol{x}}{\partial z_j}\right). \tag{20}$$

This equation has the same form as Eq. (10) except the differences in dimensions: $\partial \boldsymbol{x}/\partial z_j \in \mathbb{R}^M$ and $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{M\times M}$ in Eq. (20) while $\partial \boldsymbol{x}/\partial z_j \in \mathbb{R}^N$ and $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{N\times N}$ in Eq. (10). We have essentially no difference from Section 4.2 so far.

However, from this point, we cannot follow the same way we used in Section 4.2 to derive the equation corresponding to Eq. (13), due to the mismatch of M and N. Yet, as we show below, step-by-step modifications lead us to the same conclusion.

Firstly, note that we can always regard g_ϕ as a composition function by inserting a smooth invertible function $\rho : \mathbb{R}^N \to \mathbb{R}^N$ and its inverse as follows:

$$g_\phi(\boldsymbol{z}) = g_\phi(\rho^{-1}(\rho(\boldsymbol{z}))) = \tilde{g}_\phi(\rho(\boldsymbol{z})). \tag{21}$$

Let $\boldsymbol{y} \in \mathbb{R}^N$ be an auxiliary variable defined by $\boldsymbol{y} = \rho(\boldsymbol{z})$. Due to the chain rule of differentiation, $\partial \boldsymbol{x}/\partial \boldsymbol{z}$ can be represented as

$$\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{z}} = \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{y}}\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}} = \boldsymbol{G}\boldsymbol{B}, \tag{22}$$

where we define $\boldsymbol{G}$ and $\boldsymbol{B}$ as $\boldsymbol{G} = \partial \boldsymbol{x}/\partial \boldsymbol{y} \in \mathbb{R}^{M\times N}, \boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_N) = \partial \boldsymbol{z}/\partial \boldsymbol{y} \in \mathbb{R}^{N\times N}$.

Since $\boldsymbol{z}$ and $\boldsymbol{y}$ have the same dimension N, the relationship between $P_{\boldsymbol{z}}(\boldsymbol{z})$ and $P_{\boldsymbol{y}}(\boldsymbol{y})$ is described by $|\det(\boldsymbol{B})|$ (the absolute value of Jacobian determinant), which corresponds to the volume change under the function ρ, as follows:

$$P_{\boldsymbol{z}}(\boldsymbol{z}) = |\det(\boldsymbol{B})| P_{\boldsymbol{y}}(\boldsymbol{y}). \tag{23}$$

Thus the expectation of L in Eq. (7) can be approximated as follows:

$$\underset{\boldsymbol{\epsilon}\sim P_{\boldsymbol{\epsilon}}(\boldsymbol{\epsilon})}{E}[L] \simeq -\log(|\det(\boldsymbol{B})|) - \log(P_{\boldsymbol{y}}(\boldsymbol{y}))) + \lambda_2 \sigma^2 \left(\sum_{j=1}^N \left(\boldsymbol{G}\boldsymbol{b}_j\right)^\top \boldsymbol{A}(\boldsymbol{x}) \left(\boldsymbol{G}\boldsymbol{b}_j\right)\right). \tag{24}$$

To derive the minimization condition of the expected loss, we need further preparations. Let $\tilde{b}_{ij}$ denote the cofactor of matrix $\boldsymbol{B}$ with regard to the element b_{ij}. We define a vector $\tilde{\boldsymbol{b}}_j$ as follows:

$$\tilde{\boldsymbol{b}}_j = \begin{pmatrix} \tilde{b}_{1j} \\ \tilde{b}_{2j} \\ \vdots \\ \tilde{b}_{Nj} \end{pmatrix}. \tag{25}$$

The following equation is a property of the cofactor (Strang, 2006):

$$\boldsymbol{b}_i^\top \tilde{\boldsymbol{b}}_j = \sum_{k=1}^N b_{ki}\tilde{b}_{kj} = \delta_{ij}\det(\boldsymbol{B}). \tag{26}$$

In addition, since $|\det(\boldsymbol{B})| = (\det(\boldsymbol{B})^2)^{\frac{1}{2}} = ((\sum_{k=1}^N b_{kj}\tilde{b}_{kj})^2)^{\frac{1}{2}}$, we have the following result:

$$\frac{\partial|\det(\boldsymbol{B})|}{\partial b_{ij}} = \frac{1}{2}((\sum_{k=1}^N b_{kj}\tilde{b}_{kj})^2)^{-\frac{1}{2}} \cdot 2(\sum_{k=1}^N b_{kj}\tilde{b}_{kj})\tilde{b}_{ij} = \frac{\det(\boldsymbol{B})}{|\det(\boldsymbol{B})|}\tilde{b}_{ij}. \tag{27}$$

Therefore, the following equations hold:

$$\frac{\partial \log(|\det(\boldsymbol{B})|)}{\partial b_{ij}} = \frac{1}{|\det(\boldsymbol{B})|}\frac{\partial|\det(\boldsymbol{B})|}{\partial b_{ij}} = \frac{1}{|\det(\boldsymbol{B})|}\frac{\det(\boldsymbol{B})}{|\det(\boldsymbol{B})|}\tilde{b}_{ij} = \frac{\det(\boldsymbol{B})}{\det(\boldsymbol{B})^2}\tilde{b}_{ij} = \frac{1}{\det(\boldsymbol{B})}\tilde{b}_{ij}, \tag{28}$$

$$\frac{\partial \log(|\det(\boldsymbol{B})|)}{\partial \boldsymbol{b}_j} = \frac{1}{\det(\boldsymbol{B})}\tilde{\boldsymbol{b}}_j. \tag{29}$$

By differentiating the right hand side of Eq. (24) by $\boldsymbol{b}_j$ and setting the result to zero, the following equation is derived as a condition to minimize the expected loss:

$$2\lambda_2\sigma^2\boldsymbol{G}^\top\boldsymbol{A}(\boldsymbol{x})\boldsymbol{G}\boldsymbol{b}_j = \frac{1}{\det(\boldsymbol{B})}\tilde{\boldsymbol{b}}_j. \tag{30}$$

Here we used Eq. (29). By multiplying $\boldsymbol{b}_i^\top$ to this equation from the left and dividing the result by $2\lambda_2\sigma^2$, we have

$$(\boldsymbol{G}\,\boldsymbol{b}_i)^\top\,\boldsymbol{A}(\boldsymbol{x})(\boldsymbol{G}\,\boldsymbol{b}_j) = \frac{1}{2\lambda_2\sigma^2}\frac{1}{\det(\boldsymbol{B})}\boldsymbol{b}_i^\top\tilde{\boldsymbol{b}}_j \tag{31}$$

$$= \frac{1}{2\lambda_2\sigma^2}\delta_{ij}, \tag{32}$$

where the second line follows from Eq. (26). Since $\boldsymbol{G}\boldsymbol{b}_i = (\partial\boldsymbol{x}/\partial\boldsymbol{y})(\partial\boldsymbol{y}/\partial z_i) = \partial\boldsymbol{x}/\partial z_i$ and $\boldsymbol{G}\boldsymbol{b}_j = (\partial\boldsymbol{x}/\partial\boldsymbol{y})(\partial\boldsymbol{y}/\partial z_j) = \partial\boldsymbol{x}/\partial z_j$, we can come back to the expression with the original variables $\boldsymbol{x}$ and $\boldsymbol{z}$ and reach the following conclusion:

$$\left(\frac{\partial\boldsymbol{x}}{\partial z_i}\right)^\top\boldsymbol{A}(\boldsymbol{x})\left(\frac{\partial\boldsymbol{x}}{\partial z_j}\right) = \frac{1}{2\lambda_2\sigma^2}\delta_{ij}. \tag{33}$$

Here the dimensions are different from Eq. (13) ($\partial\boldsymbol{x}/\partial z_i, \partial\boldsymbol{x}/\partial z_j \in \mathbb{R}^M$ and $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{M\times M}$) but the meaning is same: the columns of the Jacobian matrix of two spaces $\partial\boldsymbol{x}/\partial z_1, \ldots, \partial\boldsymbol{x}/\partial z_N$ form a constantly-scaled orthonormal system with respect to the inner product defined by $\boldsymbol{A}(\boldsymbol{x})$ at every point.

Now we can derive the second conclusion in the exactly same manner as in Section 4.2, although the dimensions are different ($\boldsymbol{v}_{\boldsymbol{x}}, \boldsymbol{w}_{\boldsymbol{x}} \in \mathbb{R}^M$, $\boldsymbol{A}(\boldsymbol{x}) \in \mathbb{R}^{M\times M}$ and $\boldsymbol{v}_{\boldsymbol{z}}, \boldsymbol{w}_{\boldsymbol{z}} \in \mathbb{R}^N$):

$$\begin{aligned}\boldsymbol{v}_x\boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_x &= \sum_{i=0}^N\sum_{j=0}^N\left(\frac{\partial\boldsymbol{x}}{\partial z_i}v_{zi}\right)^\top\boldsymbol{A}(\boldsymbol{x})\left(\frac{\partial\boldsymbol{x}}{\partial z_j}w_{zj}\right)\\ &= \frac{1}{2\lambda_2\sigma^2}\sum_{i=0}^N v_{z_i}w_{z_i} = \frac{1}{2\lambda_2\sigma^2}\boldsymbol{v}_z\cdot\boldsymbol{w}_z,\end{aligned} \tag{34}$$

which means the map is isometric in the sense of Eq. (2).

B. Product of Singular Values as a Generalization of the Absolute Value of Jacobian Determinant

In this appendix, we show the following two arguments we stated in the last part of Section 4.2: i) when a region in $\mathbb{R}^N$ is mapped to $\mathbb{R}^M$ by the decoder function, the ratio of the volume of the original region and its corresponding value is equal to the product of singular values of Jacobian matrix, ii) this quantity can be expressed by $\boldsymbol{A}(\boldsymbol{x})$ under a certain condition. The Jacobian matrix $\boldsymbol{J}(\boldsymbol{z}) = \partial\boldsymbol{x}/\partial\boldsymbol{z} = \partial g_\phi(\boldsymbol{z})/\partial\boldsymbol{z} \in \mathbb{R}^{M\times N}$ is assumed to be full-rank as in Section 4.2 and Appendix A.

Let's consider the singular value decomposition $\boldsymbol{J}(\boldsymbol{z}) = \boldsymbol{U}(\boldsymbol{z})\boldsymbol{\Sigma}(\boldsymbol{z})\boldsymbol{V}(\boldsymbol{z})^\top$, where $\boldsymbol{U}(\boldsymbol{z}) \in \mathbb{R}^{M\times M}, \boldsymbol{\Sigma}(\boldsymbol{z}) \in \mathbb{R}^{M\times N}, \boldsymbol{V}(\boldsymbol{z}) \in \mathbb{R}^{N\times N}$. Note that $\{\boldsymbol{V}_{:,j}(\boldsymbol{z})\}_{j=1}^N$ is an orthonormal basis of $\mathbb{R}^N$ and $\{\boldsymbol{U}_{:,j}(\boldsymbol{z})\}_{j=1}^M$ is an orthonormal basis of $\mathbb{R}^M$ with respect to the standard inner product.

Consider an N-dimensional hypercube $\mathbf{c}$ specified by $\{\varepsilon \boldsymbol{V}_{:,j}(\boldsymbol{z})\}_{j=1}^{N}$ ($\varepsilon > 0$) attached to $\boldsymbol{z} \in \mathbb{R}^N$. When ε is small, the effect of the decoder function on $\{\varepsilon \boldsymbol{V}_{:,j}(\boldsymbol{z})\}_{j=1}^{N}$ is approximated by $\boldsymbol{J}(\boldsymbol{z}) = \boldsymbol{U}(\boldsymbol{z})\boldsymbol{\Sigma}(\boldsymbol{z})\boldsymbol{V}(\boldsymbol{z})^\top$ and thus the mapped region of $\mathbf{c}$ in $\mathbb{R}^M$ is approximated by a region $\tilde{\mathbf{c}}$ specified by $\{\varepsilon \boldsymbol{J}(\boldsymbol{z})\boldsymbol{V}_{:,j}(\boldsymbol{z})\}_{j=1}^{N} = \{\epsilon s_j(\boldsymbol{z})\boldsymbol{U}_{:,j}(\boldsymbol{z})\}_{j=1}^{N}$, where $s_1(\boldsymbol{z}) \geq \cdots \geq s_N(\boldsymbol{z}) > 0$ are the singular values of $\boldsymbol{J}(\boldsymbol{z})$ (remember full-rank assumption we posed).

Therefore, while the volume of the original hypercube $\mathbf{c}$ is ε^N, the corresponding value of $\tilde{\mathbf{c}} \in \mathbb{R}^M$ is $\varepsilon^N J_{sv}(\boldsymbol{z})$, where we define $J_{sv}(\boldsymbol{z})$ as $J_{sv}(\boldsymbol{z}) = s_1(\boldsymbol{z}) \cdots s_N(\boldsymbol{z})$, that is, the product of the singular values of the Jacobian matrix $\boldsymbol{J}(\boldsymbol{z})$. This relationship holds for any $\boldsymbol{z} \in \mathbb{R}^N$ and we can take arbitrary small ε. Thus, the ratio of the volume of an arbitrary region in $\mathbb{R}^N$ and its corresponding value in $\mathbb{R}^M$ is also $J_{sv}(\boldsymbol{z})$*.

Let's move to the second argument. Note that Eq. (33) can be rewritten in the following form since $\boldsymbol{J}(\boldsymbol{z}) = (\partial \boldsymbol{x}/\partial z_1, \ldots, \partial \boldsymbol{x}/\partial z_N)$:

$$\boldsymbol{J}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{J}(\boldsymbol{z}) = \frac{1}{2\lambda_2 \sigma^2} \boldsymbol{I}_N. \tag{35}$$

Let $0 < \alpha_1(\boldsymbol{A}(\boldsymbol{x})) \leq \cdots \leq \alpha_N(\boldsymbol{A}(\boldsymbol{x})) \leq \cdots \leq \alpha_M(\boldsymbol{A}(\boldsymbol{x}))$ be the eigenvalues of $\boldsymbol{A}(\boldsymbol{x})$. If the condition

$$[\boldsymbol{O}_{(M-N)\times N}\ \boldsymbol{I}_N] \boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z}) \begin{bmatrix} \boldsymbol{I}_N \\ \boldsymbol{O}_{(M-N)\times N} \end{bmatrix} = \boldsymbol{O}_{(M-N)\times N} \tag{36}$$

holds for all $\boldsymbol{z} \in \mathbb{R}^N$, the following relation holds for $J_{sv}(\boldsymbol{z})$:

$$J_{sv}(\boldsymbol{z}) = \left(\frac{1}{2\lambda_2 \sigma^2}\right)^{\frac{N}{2}} \left(\alpha_1(\boldsymbol{A}(\boldsymbol{x})) \cdots \alpha_N(\boldsymbol{A}(\boldsymbol{x}))\right)^{-\frac{1}{2}}. \tag{37}$$

Here $\boldsymbol{O}_{(M-N)\times N} \in \mathbb{R}^{(M-N)\times N}$ denotes the matrix consisting of zeros.

To see this, let us first define $\boldsymbol{S}(\boldsymbol{z}) \in \mathbb{R}^{N\times N}$ as $\boldsymbol{S}(\boldsymbol{z}) = \mathrm{diag}(s_1(\boldsymbol{z}), \ldots, s_N(\boldsymbol{z}))$. Then $\boldsymbol{J}(\boldsymbol{z}) = \boldsymbol{U}(\boldsymbol{z})^\top [\boldsymbol{S}(\boldsymbol{z})\ \boldsymbol{O}_{(M-N)\times N}]^\top \boldsymbol{V}(\boldsymbol{z})$. We obtain the following equation by substituting this expression of $\boldsymbol{J}(\boldsymbol{z})$ to Eq. (35):

$$\boldsymbol{V}(\boldsymbol{z})[\boldsymbol{S}(\boldsymbol{z})\ \boldsymbol{O}_{(M-N)\times N}] \boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z}) \begin{bmatrix} \boldsymbol{S}(\boldsymbol{z}) \\ \boldsymbol{O}_{(M-N)\times N} \end{bmatrix} \boldsymbol{V}(\boldsymbol{z})^\top = \frac{1}{2\lambda_2 \sigma^2} \boldsymbol{I}_N. \tag{38}$$

Furthermore, we get the following equation by multiplying Eq. (38) by $\boldsymbol{S}(\boldsymbol{z})^{-1}\boldsymbol{V}(\boldsymbol{z})^\top$ from the left and $\boldsymbol{V}(\boldsymbol{z})\boldsymbol{S}(\boldsymbol{z})^{-1}$ from the right:

$$[\boldsymbol{I}_N\ \boldsymbol{O}_{(M-N)\times N}] \boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z}) \begin{bmatrix} \boldsymbol{I}_N \\ \boldsymbol{O}_{(M-N)\times N} \end{bmatrix} = \frac{1}{2\lambda_2 \sigma^2} \boldsymbol{S}(\boldsymbol{z})^{-2} \tag{39}$$

This means $\boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z})$ has the following form:

$$\boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z}) = \begin{bmatrix} \frac{1}{2\lambda_2 \sigma^2} \boldsymbol{S}(\boldsymbol{z})^{-2} & \boldsymbol{C} \\ \boldsymbol{C}^\top & \boldsymbol{D} \end{bmatrix}, \tag{40}$$

where $\boldsymbol{C} \in \mathbb{R}^{N\times(M-N)}$ and $\boldsymbol{D} \in \mathbb{R}^{(M-N)\times(M-N)}$. Note that the standard basis vectors of $\mathbb{R}^N$, namely, $\boldsymbol{e}^{(1)} = [1 \cdots 0]^\top, \ldots, \boldsymbol{e}^{(N)} = [0 \cdots 1]^\top \in \mathbb{R}^N$, are the eigenvectors of $\frac{1}{2\lambda_2 \sigma^2} \boldsymbol{S}(\boldsymbol{z})^{-2}$ and corresponding eigenvalues are $\frac{1}{2\lambda_2 \sigma^2 s_1(\boldsymbol{z})^2} < \cdots < \frac{1}{2\lambda_2 \sigma^2 s_N(\boldsymbol{z})^2}$. According to the expression (40), the condition (36) means $\boldsymbol{C}^\top = \boldsymbol{O}_{(M-N)\times N}$, and thus $\boldsymbol{C}^\top \boldsymbol{e}^{(j)} = \boldsymbol{0}$ for all $j = 1, \ldots N$ in this situation. Note also that the eigenvalues of $\boldsymbol{U}(\boldsymbol{z})^\top \boldsymbol{A}(\boldsymbol{x}) \boldsymbol{U}(\boldsymbol{z})$ coincide with those of $\boldsymbol{A}(\boldsymbol{x})$. Therefore, if Eq. (36) holds, we have

$$\alpha_1(\boldsymbol{A}(\boldsymbol{x})) = \frac{1}{2\lambda_2 \sigma^2 s_1(\boldsymbol{z})^2}, \ldots, \alpha_N(\boldsymbol{A}(\boldsymbol{x})) = \frac{1}{2\lambda_2 \sigma^2 s_N(\boldsymbol{z})^2}, \tag{41}$$

due to the inclusion principle (Horn & Johnson, 2013). Eq. (37) follows from Eq. (41).

As mentioned before, when the metric function is square of L2 norm, $\boldsymbol{A}(\boldsymbol{x})$ is the identity matrix $\boldsymbol{I}_M$. In this case, Eq. (36) holds and we have $J_{sv}(\boldsymbol{z}) = (1/2\lambda_1 \sigma^2)^{N/2}$†.

*Consider covering the original region in $\mathbb{R}^N$ by infinitesimal hypercubes.

†This can also be directly confirmed by taking determinants of Eq. (35) after substituting $\boldsymbol{A}(\boldsymbol{x}) = \boldsymbol{I}_M$.

C. Effect of $h(x)$

In this section, the effects of $h(d)$ is discussed. By training the encoder and the decoder to be exact inverse functions of each other regarding the input data, the mapping becomes much rigidly isometric. Actually, for this purpose, it is important to choose $h(d)$ appropriately depending on metric function.

In this appendix we evaluate the behaviors of encoder and decoder in a one dimensional case using simple parametric linear encoder and decoder. Lex x be a one dimensional data with the normal distribution:

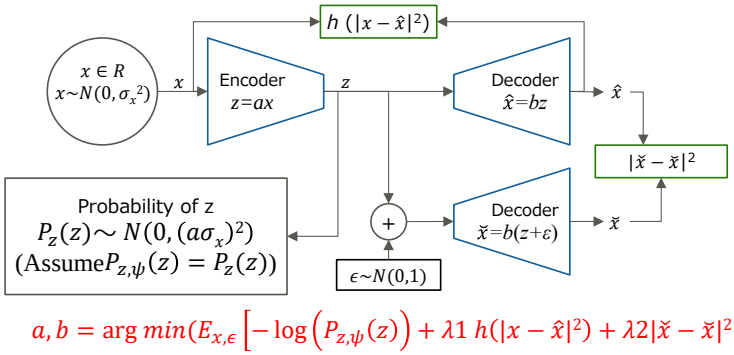


Figure 9. Simple encoder/decoder model to evaluate $h(d)$

$$\begin{aligned} x &\in \mathbb{R}, \\ x &\sim \mathcal{N}(0, {\sigma_x}^2). \end{aligned}$$

Lex z be a one dimensional latent variable. Following two linear encoder and decoder are provided with parameter a and b:

$$\begin{aligned} z &= ax, \\ \hat{x} &= bz. \end{aligned}$$

Due to the above relationship, we have

$$P_z(z) = \mathcal{N}(0, (a\sigma_x)^2). \tag{42}$$

Here, square error is used as metrics function $D(x, y)$. The distribution of noise ϵ added to latent variable z is set to $N(0, 1)$. Then $\breve{x}$ is derived by decoding $z + \epsilon$ as:

$$\begin{aligned} D(x, y) &= |x - y|^2, \\ \epsilon &\sim \mathcal{N}(0, 1), \\ \breve{x} &= b(z + \epsilon). \end{aligned}$$

For simplicity, we assume parametric PDF $P_{z,\psi}(z)$ is equal to the real PDF $P(z)$. Because the distribution of latent variable z follows $N(0, (a\sigma_x)^2)$, the entropy of z can be expressed as follows:

$$\begin{aligned} H(z) &= \int -P_z(z) \log(P_z(z)) \mathrm{d}z \\ &= \log(a) + \log(\sigma_x \sqrt{2\pi e}). \end{aligned} \tag{43}$$

Using these notations, Eqs. (7) and (9) can be expressed as follows:

$$\begin{aligned} L &= E_{x\sim\mathcal{N}(0,{\sigma_x}^2),\ \epsilon\sim\mathcal{N}(0,1)}[-\log P_z(z) + \lambda_1 h(|x - \hat{x}|^2) + \lambda_2 |\hat{x}, \breve{x}|^2] \\ &= \log(a) + \log(\sigma_x \sqrt{2\pi e}) + \lambda_1 E_{x\sim N(0,{\sigma_x}^2)} \left[h(|x - \hat{x}|^2)\right] + \lambda_2 b^2. \end{aligned} \tag{44}$$

At first, the case of $h(d) = d$ is examined. By applying $h(d) = d$, Eq. (44) can be expanded as follows:

$$L = \log(a) + \log(\sigma_x\sqrt{2\pi e}) + \lambda_1(ab-1)^2{\sigma_x}^2 + \lambda_2 b^2. \tag{45}$$

By solving $\frac{\partial L}{\partial a} = 0$ and $\frac{\partial L}{\partial b} = 0$, a and b are derived as follows:

$$\begin{aligned} ab &= \frac{\lambda_1{\sigma_x}^2 + \sqrt{{\lambda_1}^2{\sigma_x}^4 - 2\lambda_1{\sigma_x}^2}}{2\lambda_1\sigma_x^2}, \\ a &= \sqrt{2\lambda_2}\left(\frac{\lambda_1{\sigma_x}^2 + \sqrt{{\lambda_1}^2{\sigma_x}^4 - 2\lambda_1{\sigma_x}^2}}{2\lambda_1\sigma_x^2}\right), \\ b &= 1/\sqrt{2\lambda_2}. \end{aligned}$$

If $\lambda_1{\sigma_x}^2 \gg 1$, these equations are approximated as next:

$$\begin{aligned} ab &\simeq \left(1 - \frac{1}{2\lambda_1{\sigma_x}^2}\right), \\ a &= \sqrt{2\lambda_2}\left(1 - \frac{1}{2\lambda_1{\sigma_x}^2}\right), \\ b &= 1/\sqrt{2\lambda_2}. \end{aligned}$$

Here, ab is not equal to 1. That is, decoder is not an inverse function of encoder. In this case, the scale of latent space becomes slightly bent in order to minimize entropy function. As a result, good fitting of parametric PDF $P_z(z) \simeq P_{z,\psi}(z)$ could be realized while proportional relationship $P_z(z) \propto P_x(x)$ is relaxed.

Next, the case of $h(d) = \log(d)$ is examined. By applying $h(d) = \log(d)$ and introducing a minute variable Δ, Eq. (44) can be expanded as follows:

$$L = \log(a) + \log(\sigma_x\sqrt{2\pi e}) + \lambda_1 \log\left((ab-1)^2 + \Delta\right) + \lambda_2 b^2. \tag{46}$$

By solving $\frac{\partial Loss}{\partial a} = 0$ and $\frac{\partial Loss}{\partial b} = 0$ and setting $\Delta \to 0$, a and b are derived as follows:

$$\begin{aligned} ab &= 1, \\ a &= \sqrt{2\lambda_2}, \\ b &= 1/\sqrt{2\lambda_2} \end{aligned} \tag{47}$$

Here, ab is equal to 1 and decoder becomes an inverse function of encoder regardless of the variance ${\sigma_x}^2$. In this case, good proportional relation $P_z(z) \propto P_x(x)$ could be realized regardless of the fitting $P_{z,\psi}(z)$ to $P_z(z)$.

Considering from these results, there could be a guideline to choose $h(d)$. If the parametric PDF $P_{\boldsymbol{z},\psi}(\boldsymbol{z})$ has enough ability to fit the real distribution $P_{\boldsymbol{z}}(\boldsymbol{z})$, $h(d) = \log(d)$ could be better. If not, $h(d) = d$ could be an option.

D. Expansion of SSIM and BCE to Quadratic Forms

In this appendix, it is shown that SSIM and BCE can be approximated in quadratic forms with positive definite matrices except some constants.

Structural similarity (SSIM) (Wang et al., 2004) is widely used for picture quality metric since it is close to human subjective evaluation. In this appendix, we show $(1 - SSIM)$ can be approximated to a quadratic form such as Eq.(8).

Eq. (48) is a SSIM value between cropped pictures $\boldsymbol{x}$ and $\boldsymbol{y}$ with a $W \times W$ window:

$$SSIM_{W\times W}(\boldsymbol{x}, \boldsymbol{y}) = \frac{2\mu_x\mu_y}{{\mu_x}^2 + {\mu_y}^2}\frac{2\sigma_{xy}}{{\sigma_x}^2 + {\sigma_y}^2}. \tag{48}$$

In order to calculate SSIM index for entire pictures, this window is shifted in a whole picture and all of SSIM values are averaged. If $(1 - SSIM_{W\times W}(\boldsymbol{x}, \boldsymbol{y}))$ is expressed in quadratic form, the average for a picture $(1 - SSIM_{picture})$ can be also expressed in quadratic form.

Let $\Delta \boldsymbol{x}$ be a minute displacement of $\boldsymbol{x}$. Then SSIM between $\boldsymbol{x}$ and $\boldsymbol{x}+\Delta \boldsymbol{x}$ can be expressed as follows:

$$SSIM_{W\times W}(\boldsymbol{x}, \boldsymbol{x}+\Delta \boldsymbol{x}) = 1 - \frac{{\mu_{\Delta \boldsymbol{x}}}^2}{2{\mu_x}^2} - \frac{{\sigma_{\Delta \boldsymbol{x}}}^2}{2{\sigma_x}^2} + O\left(\left(|\Delta \boldsymbol{x}|/|\boldsymbol{x}|\right)^3\right) \tag{49}$$

Then ${\mu_{\Delta \boldsymbol{x}}}^2$ and ${\sigma_{\Delta \boldsymbol{x}}}^2$ can be expressed as follows:

$${\mu_{\Delta \boldsymbol{x}}}^2 = \Delta \boldsymbol{x}^\top \boldsymbol{W}_m \Delta \boldsymbol{x}, \tag{50}$$

$${\sigma_{\Delta \boldsymbol{x}}}^2 = \Delta \boldsymbol{x}^\top \boldsymbol{W}_v \Delta \boldsymbol{x}, \tag{51}$$

where

$$\boldsymbol{W}_m = \frac{1}{W^2}\begin{pmatrix} 1 & 1 & \dots & 1 \\ 1 & 1 & \dots & 1 \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & \dots & 1 \end{pmatrix}, \boldsymbol{W}_v = \frac{1}{W^2}\begin{pmatrix} W-1 & -1 & \dots & -1 \\ -1 & W-1 & \dots & -1 \\ \vdots & \vdots & \ddots & \vdots \\ -1 & -1 & \dots & W-1 \end{pmatrix}. \tag{52}$$

It should noted that matrix $\boldsymbol{W}_m$ is positive definite and matrix $\boldsymbol{W}_v$ is positive semidefinite. As a result, $(1 - SSIM_{W\times W}(\boldsymbol{x}, \boldsymbol{y}))$ can be expressed in the following quadratic form with positive definite matrix:

$$1 - SSIM_{W\times W}(\boldsymbol{x}, \boldsymbol{x}+\Delta \boldsymbol{x}) \simeq \Delta \boldsymbol{x}^\top \left(\frac{1}{2{\mu_x}^2}\boldsymbol{W}_m + \frac{1}{2{\sigma_x}^2}\boldsymbol{W}_v\right)\Delta \boldsymbol{x}. \tag{53}$$

Binary cross entropy (BCE) is also a reconstruction loss function widely used in VAE (Kingma & Welling, 2014). BCE is defined as follows:

$$BCE(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{M}(-x_i \log(y_i) - (1-x_i)\log(1-y_i)). \tag{54}$$

BCE can be also approximated by a quadratic form with positive definite matrix. Let $\Delta \boldsymbol{x}$ be a small displacement of $\boldsymbol{x}$ and Δx_i be its i-th component. Then BCE between $\boldsymbol{x}$ and $\boldsymbol{x}+\Delta \boldsymbol{x}$ can be expanded as follows:

$$\begin{aligned} BCE(\boldsymbol{x}, \boldsymbol{x}+\Delta \boldsymbol{x}) &= \sum_i(-x_i \log(x_i+\Delta x_i) - (1-x_i)\log(1-x_i-\Delta x_i)) \\ &= \sum_i\left(-x_i \log\left(x_i\left(1+\frac{\Delta x_i}{x_i}\right)\right) - (1-x_i)\log\left((1-x_i)\left(1-\frac{\Delta x_i}{1-x_i}\right)\right)\right) \\ &= \sum_i\left(-x_i \log\left(1+\frac{\Delta x_i}{x_i}\right) - (1-x_i)\log\left(1-\frac{\Delta x_i}{1-x_i}\right)\right) \\ &\quad + \sum_i(-x_i \log(x_i) - (1-x_i)\log(1-x_i)). \end{aligned} \tag{55}$$

Here, the second term of the last equation is constant depending on $\boldsymbol{x}$. The first term of the last equation is further expanded as follows by using Maclaurin expansion of logarithm:

$$\begin{aligned} &\sum_i\left(-x_i\left(\frac{\Delta x_i}{x_i} - \frac{{\Delta x_i}^2}{2{x_i}^2}\right) - (1-x_i)\left(-\frac{\Delta x_i}{1-x_i} - \frac{{\Delta x_i}^2}{2(1-x_i)^2}\right) + O\left({\Delta x_i}^3\right)\right) \\ &= \sum_i\left(\frac{1}{2}\left(\frac{1}{x_i}+\frac{1}{1-x_i}\right){\Delta x_i}^2 + O\left({\Delta x_i}^3\right)\right). \end{aligned} \tag{56}$$

Then, let a matrix $\boldsymbol{A}(\boldsymbol{x})$ be defined as follows:

$$\boldsymbol{A}(\boldsymbol{x}) = \begin{pmatrix} \frac{1}{2}\left(\frac{1}{x_1}+\frac{1}{1-x_1}\right) & 0 & \dots \\ 0 & \frac{1}{2}\left(\frac{1}{x_2}+\frac{1}{1-x_2}\right) & \dots \\ \vdots & \vdots & \ddots \end{pmatrix}. \tag{57}$$

Obviously $\boldsymbol{A}(\boldsymbol{x})$ is a positive definite matrix. As a result, BCE between $\boldsymbol{x}$ and $\boldsymbol{x}+\Delta\boldsymbol{x}$ can be approximated by a quadratic form with $\boldsymbol{x}$ depending constant offset as follows:

$$BCE(\boldsymbol{x},\boldsymbol{x}+\Delta\boldsymbol{x}) \simeq \Delta\boldsymbol{x}^\top \boldsymbol{A}(\boldsymbol{x})\Delta\boldsymbol{x} + \sum_i(-x_i\log(x_i)-(1-x_i)\log(1-x_i)). \tag{58}$$

Note that BCE is typically used for binary data. In this case, the second term in Eq. (58) is always 0.

E. "Continuous PCA" Feature of Isometric Embedding for Riemannian Manifold

In this section, we explain that the isometric embedding realized by RaDOGAGA has a continuous PCA feature when the following factorized probability density model is used:

$$P_{\boldsymbol{z},\psi}(\boldsymbol{z}) = \prod_{i=1}^{N} P_{zi,\psi}(z_i). \tag{59}$$

Here, our definition of "continuous PCA" is the following. 1) Mutual information between latent variables are minimum and likely to be uncorrelated to each other: 2) Energy of latent space is concentrated to several principal components, and the importance of each component can be determined: 3) These features are held for all subspace of a manifold and subspace is continuously connected.

Next we explain the reason why these feature is acquired. As explained in Appendix A, all column vectors of Jacobian matrix of decoder from latent space to data space have the same norm and all combinations of pairwise vectors are orthogonal. In other words, when constant value is multiplied, the resulting vectors are orthonormal. Because encoder is a inverse function of decoder ideally, each row vector of encoder's Jacobian matrix should be the same as column vector of decoder under the ideal condition. Here, $f_{ortho,\theta}(\boldsymbol{x})$ and $g_{ortho,\phi}(\boldsymbol{z}_\theta)$ are defined as encoder and decoder with these feature. Because the latent variables depend on encoder parameter θ, latent variable is described as $\boldsymbol{z}_\theta = f_{ortho,\theta}(\boldsymbol{x})$, and its PDF is defined as $P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)$. PDFs of latent space and data space have the following relation where $J_{sv}(\boldsymbol{z}_\theta)$ is the product of the singular values of $\boldsymbol{J}(\boldsymbol{z}_\theta)$ which is a Jacobian matrix between two spaces as explained in Section 4.2 and Appendix B.

$$P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta) = J_{sv}(\boldsymbol{z}_\theta)P_{\boldsymbol{x}}(\boldsymbol{x}) \propto \left(\prod_{j=1}^{N}\alpha_j(\boldsymbol{A}(\boldsymbol{x}))\right)^{-\frac{1}{2}} P_{\boldsymbol{x}}(\boldsymbol{x}). \tag{60}$$

As described before, $P_{\boldsymbol{z},\psi}(\boldsymbol{z})$ is a parametric PDF of the latent space to be optimized with parameter ψ.

By applying the result of Eqs. (24) and (31), Eq. (7) can be transformed as Eq. (61) where $\hat{\boldsymbol{x}} = g_{ortho,\phi}(f_{ortho,\theta}(\boldsymbol{x}))$.

$$\begin{aligned} L_{ortho} &= -\log\left(P_{\boldsymbol{z},\psi}(\boldsymbol{z}_\theta)\right) + \lambda_1 h\left(D(\boldsymbol{x},\hat{\boldsymbol{x}})\right) + N/2. \\ s.t.\ &\left(\frac{\partial g_{ortho,\phi}(\boldsymbol{z}_\theta)}{\partial z_{\theta_i}}\right)^\top \boldsymbol{A}(\boldsymbol{x}) \left(\frac{\partial g_{ortho,\phi}(\boldsymbol{z}_\theta)}{\partial z_{\theta_j}}\right) = \frac{1}{2\lambda\sigma^2}\delta_{ij}. \end{aligned} \tag{61}$$

Here, the third term of the right side is constant, this term can be removed from the cost function as follows:

$$L'_{ortho} = -\log\left(P_{\boldsymbol{z},\psi}(\boldsymbol{z}_\theta)\right) + \lambda_1 h\left(D(\boldsymbol{x},\hat{\boldsymbol{x}})\right). \tag{62}$$

Then the parameters of network and PDF are obtained according to the following equation:

$$\theta,\phi,\psi = \arg\min_{\theta,\phi,\psi}(E_{\boldsymbol{x}\sim P_{\boldsymbol{x}}(\boldsymbol{x})}[L'_{ortho}]). \tag{63}$$

$E_{\boldsymbol{x}\sim P_{\boldsymbol{x}}(\boldsymbol{x})}[L'_{ortho}]$ in Eq. (63) can be transformed as the next:

$$\begin{aligned} E_{\boldsymbol{x}\sim P_{\boldsymbol{x}}(\boldsymbol{x})}[L'_{ortho}] &= \int P_{\boldsymbol{x}}(\boldsymbol{x})\left(-\log\left(P_{\boldsymbol{z},\psi}(\boldsymbol{z}_\theta)\right) + \lambda_1\, h\left(D(\boldsymbol{x},\hat{\boldsymbol{x}})\right)\right)\mathrm{d}\boldsymbol{x} \\ &= \int \left(P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)J_{sv}(\boldsymbol{z}_\theta)^{-1}\right)\left(-\log\left(P_{\boldsymbol{z},\psi}(\boldsymbol{z}_\theta)\right)\right)\, J_{sv}(\boldsymbol{z}_\theta)\mathrm{d}z_\theta + \lambda_1 \int P_{\boldsymbol{x}}(\boldsymbol{x})\, h\left(D(\boldsymbol{x},\hat{\boldsymbol{x}})\right)\mathrm{d}\boldsymbol{x}. \end{aligned} \tag{64}$$

At first, the first term of the third formula in Eq.(64) is examined. Let $\mathrm{d}\boldsymbol{z}_{\theta/i}$ be a differential of $(N-1)$ dimensional latent variables where i-th axis $z_{\theta i}$ is removed from the latent variable $\boldsymbol{z}_\theta$. Then a marginal distribution of $z_{\theta i}$ can be derived from the next equation:

$$P_{z,\theta i}(z_{\theta i}) = \int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\mathrm{d}\boldsymbol{z}_{\theta/i}. \tag{65}$$

By using Eqs.(59) and (65), the first term of the third formula in Eq. (64) can be expanded as:

$$\begin{aligned}
\int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\left(-\log\left(P_{\boldsymbol{z},\psi}(\boldsymbol{z}_\theta)\right)\right)\mathrm{d}\boldsymbol{z}_\theta &= \int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\left(-\log\left(\frac{\prod_{i=1}^N P_{z_i,\psi}(z_{\theta i})}{\prod_{i=1}^N P_{z,\theta i}(z_{\theta i})}\right)\right)\mathrm{d}\boldsymbol{z}_\theta \\
&\quad + \int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\left(-\log\left(\prod_{i=1}^N P_{z,\theta i}(z_{\theta i})\right)\right)\mathrm{d}\boldsymbol{z}_\theta \\
&= \sum_{i=1}^N \int\left(\int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\mathrm{d}\boldsymbol{z}_{\theta/i}\right)\left(-\log\left(\frac{P_{z_i,\psi}(z_{\theta i})}{P_{z,\theta i}(z_{\theta i})}\right)\right)\mathrm{d}z_{\theta i} \\
&\quad + \sum_{i=1}^N \int\left(\int P_{\boldsymbol{z},\theta}(\boldsymbol{z}_\theta)\mathrm{d}\boldsymbol{z}_{\theta/i}\right)\left(-\log\left(P_{z,\theta i}(z_{\theta i})\right)\right)\mathrm{d}z_{\theta i} \\
&= \sum_{i=1}^N D_{KL}(P_{z,\theta i}(z_{\theta i})\|P_{z_i,\psi}(z_{\theta i})) + \sum_{i=1}^N H(z_{\theta i}).
\end{aligned} \tag{66}$$

Here $H(X)$ denotes the entropy of a variable X. The first term of the third formula is KL-divergence between marginal probability $P_{z,\theta i}(z_{\theta i})$ and factorized parametric probability $P_{z_i,\psi}(z_{\theta i})$. The second term of the third formula can be further transformed using mutual information between latent variables $I(\boldsymbol{z}_\theta)$ and equation (60).

$$\begin{aligned}
\sum_{i=1}^N H(z_{\theta i}) &= H(\boldsymbol{z}_\theta) + I(\boldsymbol{z}_\theta) \simeq -\int J_{sv}(\boldsymbol{z}_\theta)P_{\boldsymbol{x}}(\boldsymbol{x})\log(J_{sv}(\boldsymbol{z}_\theta)P_{\boldsymbol{x}}(\boldsymbol{x}))J_{sv}(\boldsymbol{z}_\theta)^{-1}\mathrm{d}\boldsymbol{x} + I(\boldsymbol{z}_\theta) \\
&= H(\boldsymbol{x}) - \int P_{\boldsymbol{x}}(\boldsymbol{x})\log\left(\left(\frac{1}{2\lambda_2\sigma^2}\right)^{\frac{N}{2}}\left(\prod_{j=1}^N \alpha_j(\boldsymbol{A}(\boldsymbol{x}))\right)^{-\frac{1}{2}}\right)\mathrm{d}\boldsymbol{x} + I(\boldsymbol{z}_\theta)
\end{aligned} \tag{67}$$

At second, the second term of the third formula in Eq. (64) is examined. When $\boldsymbol{x}$ and $\hat{\boldsymbol{x}}$ are close, the following equation holds.

$$D(\boldsymbol{x},\hat{\boldsymbol{x}}) \simeq (\boldsymbol{x}-\hat{\boldsymbol{x}})^\top \boldsymbol{A}(\boldsymbol{x})(\boldsymbol{x}-\hat{\boldsymbol{x}}). \tag{68}$$

Note that with given distribution $\boldsymbol{x} \sim P_{\boldsymbol{x}}(\boldsymbol{x})$, the first and the second term in the right side of Eq. (67) are fixed value. Therefore, by using these expansions, Eq.(64) can be expressed as:

$$\begin{aligned}
E_{\boldsymbol{x}\sim P_{\boldsymbol{x}}(\boldsymbol{x})}[L'_{ortho}] &\simeq \sum_{i=1}^N D_{KL}(P_{z,\theta i}(z_{\theta i})\|P_{z_i,\psi}(z_{\theta i})) \\
&\quad + I(\boldsymbol{z}_\theta) + E_{\boldsymbol{x}}\left[(\boldsymbol{x}-\hat{\boldsymbol{x}})^\top \boldsymbol{A}(\boldsymbol{x})(\boldsymbol{x}-\hat{\boldsymbol{x}})\right] + \mathrm{Const.}
\end{aligned} \tag{69}$$

Here, the real space $\mathbb{R}^M$ is divided into a plurality of small subspace partitioning $\Omega\boldsymbol{x}_1$, $\Omega\boldsymbol{x}_2$, $\cdots$. Note that $\mathbb{R}^M$ is an inner product space endowed with metric tensor $\boldsymbol{A}(\boldsymbol{x})$. Let $\Omega\boldsymbol{z}_1$, $\Omega\boldsymbol{z}_2$, $\cdots$ be the division space of the latent space $\boldsymbol{z} \in \boldsymbol{R}^N$ corresponding to $\Omega\boldsymbol{x}$.

Then Eq. (69) can be rewritten as:

$$\begin{aligned}
E_{\boldsymbol{x}\sim P_{\boldsymbol{x}}(\boldsymbol{x})}[L'_{ortho}] &\simeq \sum_{i=1}^N D_{KL}(P_{z,\theta i}(z_{\theta i})\|P_{z_i,\psi}(z_{\theta i})) \\
&\quad + \sum_k \left(I(\boldsymbol{z}_\theta \in \Omega\boldsymbol{z}_{\theta k}) + E_{\boldsymbol{x}\in\Omega\boldsymbol{x}_k}\left[(\boldsymbol{x}-\hat{\boldsymbol{x}})^\top \boldsymbol{A}(\boldsymbol{x})(\boldsymbol{x}-\hat{\boldsymbol{x}})\right]\right) + \mathrm{Const.}
\end{aligned} \tag{70}$$

For each subspace partitioning, Jacobian matrix for the transformation from $\Omega \boldsymbol{x}_k$ to $\Omega \boldsymbol{z}_{\theta k}$ forms constantly scaled orthonormal system with respect to $\boldsymbol{A}(\boldsymbol{x})$. According to Karhunen-Loève Theory (Rao & Yip, 2000), the orthonormal basis which minimize both mutual information and reconstruction error leads to be Karhunen-Loève transform (KLT). It is noted that the basis of KLT is equivalent to PCA orthonormal basis.

As a result, when Eq. (70) is minimized, Jacobi matrix from $\Omega \boldsymbol{x}_k$ to $\Omega \boldsymbol{z}_{\theta k}$ for each subspace partitioning should be KLT/PCA. Consequently, the same feature as PCA will be realized such as the determination of principal components etc.

From these considerations, we conclude that RaDOGAGA has a "continuous PCA" feature. This is experimentally shown in Section 5.1 and Appendix F.6.

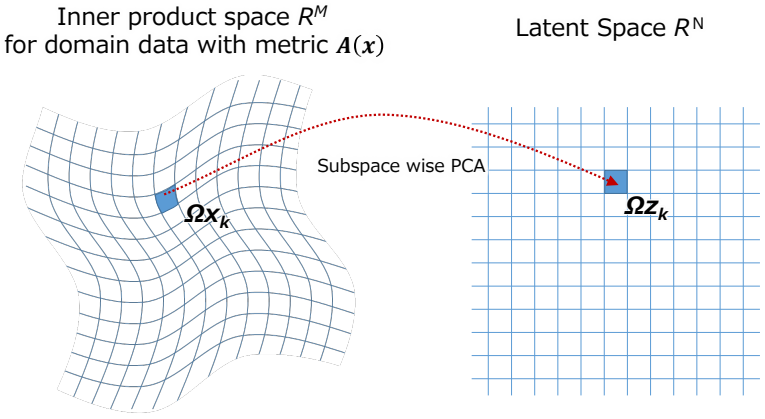


Figure 10. Continuous KLT (PCA) Mapping from input domain to latent space. For all small subspace partitioning $\Omega \boldsymbol{x}_k$ domain space (which is inner product space with metric tensor $\boldsymbol{A}(\boldsymbol{x})$), mapping from $\Omega \boldsymbol{x}_k$ to $\Omega \boldsymbol{z}_{\theta k}$ can be regarded as PCA

F. Detail and Expansion Result of Experiment in Section 5.1

In this section, we will provide further detail and a result of the complemental experiment regarding section 5.1.

F.1. GDN Activation

GDN activation function (Ballé et al., 2016) is known to suitable for image compression. For implementation, we use a TensorFlow library[‖].

F.2. Other Training Information

The batch size is 64. The iteration number is 500,000. We use NVIDIA Tesla V100 (SXM2).
For RaDOGAGA, since our implementation was done based on the source code for image compression, entropy rate is calculated as $-\log((P_{\boldsymbol{z},\psi}(\boldsymbol{z}))/(M \log 2)$, meaning bit per pixel. In addition, for RaDOGAGA, the second term of Eq. (7) is always MSE in this experiment. This is because we found that the training with $1 - SSIM$ as the reconstruction loss is likely to diverge at the beginning step of the training. Therefore, we tried to start training with MSE and then fine-tuned with $1 - SSIM$. Eventually, the result is almost the same as the case without finetuning. Therefore, to simplify the training process, we do not usually finetune.

F.3. Generation of $\boldsymbol{v}_z$ and $\boldsymbol{w}_z$

To evaluate the isometricity of the mapping, it is necessary to prepare random tangent vector $\boldsymbol{v}_z$ and $\boldsymbol{w}_z$ with a scattered interior angle. We generate two different tangent vectors $\boldsymbol{v}_z = \{v_{z1}, v_{z2}, \ldots, v_{zn}\}$ and $\boldsymbol{w}_z = \{w_{z1}, w_{z2}, \ldots, w_{zn}\}$ in the following manner. First, we prepare $\boldsymbol{v}'_z \in \mathbb{R}^N$ as $\{1.0, 0.0, \ldots 0.0\}$. Then, we sample $\boldsymbol{\alpha} = \{\alpha_1, \alpha_2, \ldots, \alpha_{n-1}\}$ $(\alpha_{1 \cdots n-2} \sim U(0, \pi), \alpha_{n-1} \sim U(0, 2\pi))$ to set $\boldsymbol{w}'$ as the conversion of polar coordinate $\{r, \boldsymbol{\alpha}\} \in \mathbb{R}^N$ to rectangular

[‖] https://github.com/tensorflow/compression/tree/master/docs/api_docs/python/tfc

coordinates, where $r = 1$. Thus, the distribution of interior angle of $\boldsymbol{v}'_z$ and $\boldsymbol{w}'_z$ also obey $\alpha_1 \sim U(0, \pi)$. Next, we randomly rotate the plane $\mathbb{R}^N$ in which interior angle of $\boldsymbol{v}'_z$ and $\boldsymbol{w}'_z$ is α_1 in the following way (Teoh, 2005) and obtain $\boldsymbol{v}_z$ and $\boldsymbol{w}_z$.

$$\boldsymbol{\rho} = -\frac{\cos\alpha_1}{\sin\alpha_1}\boldsymbol{v}'_z + \frac{1}{\sin\alpha_1}\boldsymbol{w}'_z, \quad \boldsymbol{\tau} = \boldsymbol{v}'_z,$$

then,

$$\begin{pmatrix} \boldsymbol{v}_z \\ \boldsymbol{w}_z \end{pmatrix} = \begin{bmatrix} -\sin\omega & \cos\omega \\ \cos\omega\sin\alpha_1 - \sin\omega\cos\alpha_1 & \sin\omega\sin\alpha_1 + \cos\omega\cos\alpha_1 \end{bmatrix} \begin{pmatrix} \boldsymbol{\rho} \\ \boldsymbol{\tau} \end{pmatrix}, \tag{71}$$

where $\omega \sim U(0, 2\pi)$ is the rotation angle of the plane. Note that since this is the rotation of the plane, the interior angle between $\boldsymbol{v}$ and $\boldsymbol{w}$ is kept to α_1. Finally, we normalize the norm of $\boldsymbol{v}_z$ and $\boldsymbol{v}_z$ to be 0.01.

F.4. Experiment with MNIST Dataset and BCE

Besides of the experiment in main paper, we conducted an experiment with MNIST dataset (LeCun et al., 1998)[††] which contains handwritten digits binary images with the image size of 28×28. We use 60,000 samples in the training split. The metric function is BCE, where $\boldsymbol{A}(\boldsymbol{x})$ is approximated as Eq. (57). Autoencoder consists of FC layers with sizes of 1000, 1000, 128, 1000, and 1000. We attach $softplus$ as activation function except for the last of the encoder and the decoder. In this experiment, we modify the form of the cost function of beta-VAE as

$$L = -L_{kl} + \lambda_1 h\left(D\left(\boldsymbol{x}, \hat{\boldsymbol{x}}\right)\right) + \lambda_2 D\left(\hat{\boldsymbol{x}}, \breve{\boldsymbol{x}}\right), \tag{72}$$

where $\hat{\boldsymbol{x}}$ is the output of the decoder without noise, and $\breve{\boldsymbol{x}}$ is the output of the decoder with the noise of reparameterization trick. We set (λ_1, λ_2) as (10, 1) for beta-VAE and (0.01, 0.01) for RaDOGAGA. Optimization is done with Adam optimizer with learning late 1×10^{-4} for beta-VAE 1×10^{-5} for RaDOGAGA. The batch size is 256 and the training iteration is 30,000. These parameters are determined to make the $PSNR = 20\log_{10}\left(\frac{MAX_{\boldsymbol{x}}^2}{MSE}\right)$, where $MAX_{\boldsymbol{x}} = 255$, between input and reconstruction image approximately 25 dB.

Figure 11 depicts the result. We can observe that map of RaDOGAGA is isometric as well even for the case the metric function is BCE. Consequently, even if the metric function is complicated one, the impact of the latent variable on the metric function is tractable. We expect this feature promotes further improving of metric learning, data interpolation, and so on.

(a) beta-VAE (b) RaDOGAGA

Figure 11. Plot of $\boldsymbol{v}_z \cdot \boldsymbol{w}_z$ (horizontal axis) and ${\boldsymbol{v}_x}^\top \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_x$ (vertical axis). In beta-VAE (left), the correlation is week while in our method (right) we can observe proportionality.

F.5. Isometricity of Encoder Side

In Section 5.1, we showed the isometricty of decoder side because it is common to analyse the behavior of latent variables by observing the decoder output such as latent traverse. We also clarify that the embedding by encoder f keep isometric. Given two tangent vector $\boldsymbol{v}_x$ and $\boldsymbol{w}_x$, ${\boldsymbol{v}_x}^\top \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_x$ is compared to $\mathrm{d}f(\boldsymbol{v}_x) \cdot \mathrm{d}f(\boldsymbol{v}_x)$. $\mathrm{d}f(\boldsymbol{w}_x)$ is also approximated by $f(\boldsymbol{x} + \boldsymbol{v}_x) - f(\boldsymbol{x})$. As Fig. 12 shows, the embedding to the latent space is isometric. Consequently, it is experimentally supported that our method enables to embed data in Euclidean space isometrically. The result of the same experiment for the case of the metric is $1 - SSIM$ is provided in Appendix F.

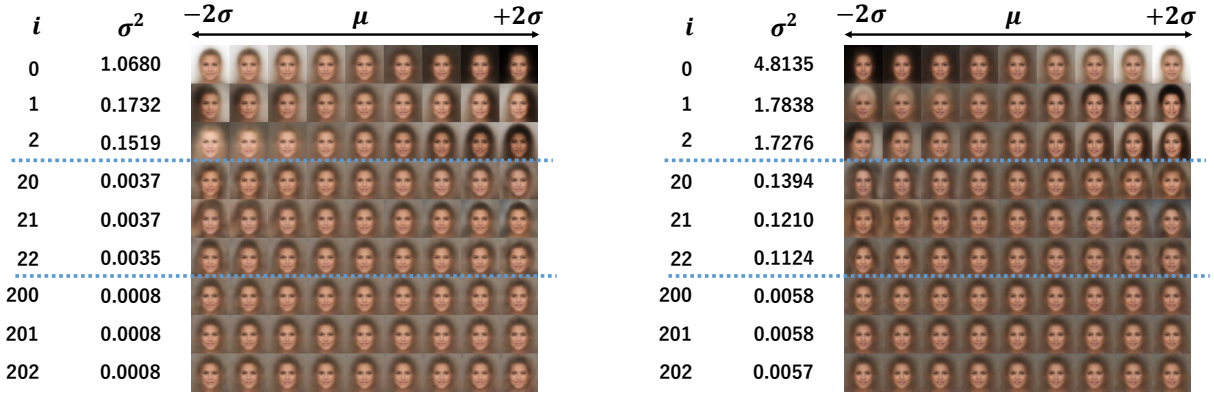
[††] http://yann.lecun.com/exdb/mnist/

(a) Variance of $\boldsymbol{z}$ (b)

Figure 12. ${\boldsymbol{v}_x}^\top \boldsymbol{A}(\boldsymbol{x})\boldsymbol{w}_x$ vs $\mathrm{d}f(\boldsymbol{v}_x) \cdot \mathrm{d}f(\boldsymbol{w}_x)$. The mapping by encoder is also isometric.

F.6. Additional Latent Traverse

In Section 5.1, the latent traverse for variables with the top 9 variances was provided. To further clarify whether the variance is corresponding to visual impact, the latent traverse of RaDOGAGA for z_0, z_1, z_2, z_{20}, z_{21}, z_{22}, z_{200}, z_{201}, and z_{202} are shown in Fig. 13. Apparently, a latent traverse with a larger σ makes a bigger difference in the image. When the σ^2 gets close to 0, there is almost no visual difference. Accordingly, the behavior as continuous PCA is clarified throughout the entire variables.



(a) MSE (b) $1 - SSIM$

Figure 13. Latent space traversal of z. For the top-3 variables, pictures look significantly different. In the middle range(z_{20}, z_{21}, z_{22}), the difference is smaller than the upper three but still observable. For the bottom three, there is almost no difference.

G. Detail of the Experiment in Section 5.3

In this section, we provide further detail of experiment in Section 5.3.

G.1. Datasets

We describe the detail of following four public datasets:

KDDCUP99 (Dua & Graff, 2019) The KDDCUP99 10 percent dataset from the UCI repository is a dataset for cyber-attack detection. This dataset consists of 494,021 instances and contains 34 continuous features and 7 categorical ones. We use one hot representation to encode the categorical features, and eventually obtain a dataset with features of 121 dimensions. Since the dataset contains only 20% of instances labeled -normal- and the rest labeled as -attacks-, -normal- instances are used as anomalies, since they are in a minority group.

Thyroid (Dua & Graff, 2019) This dataset contains 3,772 data sample with 6-dimensional feature from patients and can be divided in three classes: normal (not hypothyroid), hyperfunction, and subnormal functioning. We treat the hyperfunction class (2.5%) as an anomaly and rest two classes as normal.

Arrhythmia (Dua & Graff, 2019) This is dataset to detect cardiac arrhythmia containing 452 data sample with 274-dimensional feature. We treat minor classes (3, 4, 5, 7, 8, 9, 14, and 15, accounting for 15% of the total) as anomalies, and the others are treated as normal.

KDDCUP-Rev (Dua & Graff, 2019) To treat "normal" instances as majority in the KDDCUP dataset, we keep all "normal" instances and randomly pick up "attack" instances so that they compose 20% of the dataset. In the end, the number of instance is 121,597.

Data is max-min normalized toward dimension through the entire dataset.

G.2. Hyperparameter and Training Detail

Hyperparameter for RaDOGAGA is described in Table 2. First and second column is number of neurons. (λ_1, λ_2) is determined experimentally. For DAGMM, the number of neuron is the same as Table 2. We set (λ_1, λ_2) as (0.1, 0.005) referring Zong et al. (2018) except for Thyroid. Only for Thyroid, (λ_1, λ_2) is (0.1, 0.0001) since (0.1, 0.005) does not work well with our implementation. Optimization is done by Adam optimizer with learning rate 1×10^{-4} for all dataset. The batch size is 1024 for all dataset. The epoch number is 100, 20000, 10000, and 100 respectively. We save and the test models by every 1/10 epochs and early stop is applied. For this experiment, we use GeForce GTX 1080.

Table 2. Hyper parameter for RaDOGAGA

Dataset	Autoencoder	EN	$\lambda_1(d)$	$\lambda_2(d)$	$\lambda_1((log(d))$	$\lambda_2(log(d))$
KDDCup99	60, 30, 8, 30, 60	10, 4	100	1000	10	100
Thyroid	30, 24, 6, 24, 30	10, 2	10000	1000	100	1000
Arrhythmia	10, 4, 10	10, 2	1000	1000	1000	100
KDDCup-rev	60, 30, 8, 30, 60	10, 2	100	100	100	100

G.3. Experiment with different network size

In addition to experiment in main page, we also conducted experiment with same network size as in Zong et al. (2018) with parameters in Table 3

Table 3. Hyper parameter for RaDOGAGA(same network size as in Zong et al. (2018)

Dataset	Autoencoder	EN	$\lambda_1(d)$	$\lambda_2(d)$	$\lambda_1((log(d))$	$\lambda_2(log(d))$
KDDCup99	60, 30, 1, 30, 60	10, 4	100	100	100	1000
Thyroid	12, 4, 1, 4, 12	10, 2	1000	10000	100	10000
Arrhythmia	10, 2, 10	10, 2	1000	100	1000	100
KDDCup-rev	60, 30, 1, 30, 60	10, 2	100	100	100	1000

Now, we provide results of setting in Table 3. In Table 4, RaDOGAGA- and DAGMM- are results of them and DAGMM is result cited from Zong et al. (2018). Even with this network size, our method has boost from baseline in all dataset.

Table 4. Average and standard deviations (in brackets) of Precision, Recall and F1

Dataset	Methods	Precision	Recall	F1
KDDCup	DAGMM	0.9297	0.9442	0.9369
	DAGMM-	0.9338 (0.0051)	0.9484 (0.0052)	0.9410 (0.0051)
	RaDOGAGA-(L2)	0.9455 (0.0016)	0.9608 (0.0018)	0.9531 (0.0017)
	RaDOGAGA-(log)	0.9370 (0.0024)	0.9517 (0.0025)	0.9443 (0.0024)
Thyroid	DAGMM	0.4766	0.4834	0.4782
	DAGMM-	0.4635 (0.1054)	0.4837 (0.1100)	0.4734 (0.1076)
	RaDOGAGA-(L2)	0.5729 (0.0449)	0.5978 (0.0469)	0.5851 (0.0459)
	RaDOGAGA-(log)	0.5729 (0.0398)	0.5978 (0.0415)	0.5851 (0.0406)
Arrythmia	DAGMM	0.4909	0.5078	0.4983
	DAGMM-	0.4721 (0.0451)	0.4864 (0.0464)	0.4791 (0.0457)
	RaDOGAGA-(L2)	0.4897 (0.0477)	0.5045 (0.0491)	0.4970 (0.0484)
	RaDOGAGA-(log)	0.5044 (0.0364)	0.5197 (0.0375)	0.5119 (0.0369)
KDDCup-rev	DAGMM	0.937	0.939	0.938
	DAGMM-	0.9491 (0.0163)	0.9498 (0.0158)	0.9494 (0.0160)
	RaDOGAGA-(L2)	0.9761 (0.0057)	0.9761 (0.0056)	0.9761 (0.0057)
	RaDOGAGA-(log)	0.9791 (0.0036)	0.9799 (0.0035)	0.9795 (0.0036)