Flat latent manifolds for music improvisation between human and machine

Nutan Chen¹, Djalel Benbouzid¹, Francesco Ferroni², Mathis Nitschke³, Luciano Pinna⁴ and Patrick van der Smagt¹
¹Machine Learning Research Lab, Volkswagen Group
²ArgoAI
³St. Pölten University of Applied Sciences
⁴St Joost School of Art and Design

Abstract
The use of machine learning in artistic music generation leads to controversial discussions of the quality of art, for which objective quantification is nonsensical. We therefore consider a music-generating algorithm as a counterpart to a human musician, in a setting where reciprocal improvisation is to lead to new experiences, both for the musician and the audience. To obtain this behaviour, we resort to the framework of recurrent Variational Auto-Encoders (VAE) and learn to generate music, seeded by a human musician. In the learned model, we generate novel musical sequences by interpolation in latent space. Standard VAEs however do not guarantee any form of smoothness in their latent representation. This translates into abrupt changes in the generated music sequences. To overcome these limitations, we regularise the decoder and endow the latent space with a flat Riemannian manifold, i.e., a manifold that is isometric to the Euclidean space. As a result, linearly interpolating in the latent space yields realistic and smooth musical changes that fit the type of machine–musician interactions we aim for. We provide empirical evidence for our method via a set of experiments on music datasets and we deploy our model for an interactive jam session with a professional drummer. The live performance provides qualitative evidence that the latent representation can be intuitively interpreted and exploited by the drummer to drive the interaction. Beyond the musical application, our approach showcases an instance of machine-centred design of machine-learning models, driven by interpretability and the interaction with the end user.

1 Introduction
Music composition is a traditionally done by humans, for humans. While tools exist which support the composer in instrumentation, counterpoint, etc., full-fledged automatic composition is somewhat beside the point, leading to existential questions on the value of art. Still, music generation can benefit from machine augmentation, and to demonstrate this we decided to put human and machine on equal footing: can we create a jam session between two musicians, where the one is human, the other machine?

Such a machine is therefore required to generate a sequence of music, based on what has been played before. We address this within the framework of recurrent variational auto-encoders (recurrent VAEs) and define an optimisation scheme that results in a meaningful and interpretable latent representation. Our method – we call it Flat-Manifold Music-VAE (or FM-Music-VAE) – infers a latent space that is isometric to the Euclidean space, hence enabling linear interpolations to be smooth and to reflect natural-sounding changes in generated musical sequences.

This work therefore takes generative sequential models (e.g., [Bayer and Osendorfer, 2014; Fraccaro et al., 2016; Krishnan et al., 2015; Karl et al., 2017]) a step further by not just focussing on reducing the loss of a predicted sequence, but rather quantifying the quality of prediction by the interaction between human and generative model; as it were, directly in the control loop.

Recurrent VAEs have been successful in generating long musical sequences [Roberts et al., 2018], however, as for VAEs in general, they notoriously suffer from a strong regularisation: the prior distribution of the latent variables, being a standard Normal distribution, hinders the learning and prevents the posterior distribution from correctly representing the structure of the data [Alemi et al., 2018]. As a result, interpolating in the latent space exhibits sharp and abrupt changes, which is unsuitable for musical applications.

FM-Music-VAE solves the aforementioned limitation by learning a flat manifold [Chen et al., 2020]. By treating the latent space as a Riemannian manifold, we constrain the topology of the decoder to be isometric to the Euclidean space. In order to relax the over-regularisation of the standard Gaussian prior, we define the recurrent VAE as a hierarchical model with two stochastic layers where the prior distribution is learnt as well. The full model is jointly learnt with a constrained optimisation scheme.

We show that our method infers a latent representation that is used as a theoretically-motivated distance measure via a set of experiments on music data. Furthermore, we describe how our method was deployed in an artistic representation, with a back-and-forth ‘drum dialogue’ between the model and a professional drummer. The drummer was not only answering to the model’s musical generations but also visualise the latent
space and tangibly perceive the trajectory of the interplay.

2 Method

2.1 Background on VAEs

Given the observable data \( x \in \mathbb{R}^{N_x} \), and the latent variables \( z \in \mathbb{R}^{N_z} \), we define the Latent-variable models

\[
p(x) = \int p(x|z) p(z) \, dz.
\]

Since computing \( p(x) \) is usually infeasible, the variational auto-encoders (VAEs) \([\text{Kingma and Welling, 2014; Rezende et al., 2014}]\) approximate it by a parametric inference \( q_\phi(z|x) \) to maximise the evidence lower bound (ELBO):

\[
\mathbb{E}_{p_D(x)} \left[ \log p_\theta(x) \right] \geq \mathbb{E}_{p_D(x)} \left[ \mathbb{E}_{q_\phi(z|x)} \left[ \log p_\theta(x|z) \right] \right] - \text{KL} \left( q_\phi(z|x) \parallel p(z) \right),
\]

where the data is \( D = \{x_i\}_{i=1}^{N} \) and \( p_D(x) = \frac{1}{Z} \sum_{i=1}^{N} \delta(x - x_i) \) represents the empirical distribution. The approximate posterior \( q_\phi(z|x) \) and the likelihood \( p_\theta(x|z) \) are decoder and encoder, respectively. The distribution parameters, \( \theta \) and \( \phi \), are approximated by neural networks. Usually, a Gaussian distribution is selected as the prior \( p(z) \).

2.2 Recurrent VAEs

Recurrent VAEs are the simplest sequential flavour of VAEs, wherein the feed-forward encoder and decoder are replaced with recurrent networks while the latent variables remain time-independent \([\text{Bowman et al., 2016}]\). Several architectures for the encoder and decoder are possible. In the present work, the decoder follows the two-stage architecture in MusicVAE \([\text{Roberts et al., 2018}]\) as it is suitable for handling long sequences. Sharing a similar architecture also helps comparing the two algorithms in the experiments. While MusicVAE makes use of bidirectional LSTMs \([\text{Hochreiter and Schmidhuber, 1997; Schuster and Paliwal, 1997}]\), we use Gated Recurrent Units (GRUs) \([\text{Cho et al., 2014}]\) as they yield similar performance while being simpler \([\text{Chung et al., 2014}]\) and empirically faster to train.

The encoder receives a sequence of music data as input \( x = \{x^1, x^2, \ldots, x^{N_x}\} \) consisting in \( N_x \) frames and \( N_d \) dimensions per frame, and outputs the parameters of the posterior distribution \( q_\phi(z|x) \), the mean and standard deviation of a Normal distribution in our case. The input data is passed to stacked bidirectional GRUs, the output of which, the two state vectors, is concatenated and fed into a fully-connected layer. The last layer parametrises the mean and standard deviation of the latent variables.

In the decoder, a recurrent network dubbed the conductor, learns the high-level structure of the musical sequence. Every step of the conductor triggers a lower-level recurrent network which acts locally and is limited to one bar. Each low-level network is initialised with the hidden state of the conductor at the corresponding time and the conductor itself is initialised with a sample from the latent variables, as depicted in fig. 1.

The hierarchical decoder also prevents the so-called posterior collapse \([\text{Bowman et al., 2016; Kingma et al., 2016}]\), i.e., when the high-capacity recurrent network of the decoder tends to ignore the input from the latent variables.

| 1 | 2 | 3 |
|---|---|---|
| \( x^1 \) | \( x^2 \) | \( x^N \) |
| \( z \) |

Figure 1: The architecture of FM-Music-VAE, inspired by the Music-VAE architecture. \( \otimes \) denotes vector concatenation.

2.3 Learning Flat Latent Manifolds with Recurrent VAEs

VAEs, including their recurrent version, do not make any assumption on the inferred distances in the latent space. In particular, they do not guarantee that the Euclidean distance in the latent space reflects any similarity between the sequences of the observation space, hence not satisfying our desideratum. Therefore, we resort to flattening the latent manifold, i.e., constrain it to act locally as Euclidean space. We define the latent space inferred by the VAE as a Riemannian manifold, as motivated by \([\text{Chen et al., 2018a}]\). Given a trajectory \( \gamma : [0, 1] \rightarrow \mathbb{R}^{N_z} \) in the latent space, the length in the observation space is:

\[
L(\gamma) = \int_0^1 \sqrt{\dot{\gamma}(t)^T G(\gamma(t)) \dot{\gamma}(t)} \, dt,
\]

where \( G \in \mathbb{R}^{N_z \times N_z} \) is the Riemannian metric tensor, and \( \dot{\gamma}(t) \) the time derivative. The observation-space distance is defined as the shortest possible path between two data points \( D = \text{min}_{\gamma} L(\gamma) \). The trajectory \( \hat{\gamma} = \text{arg min}_{\gamma} L(\gamma) \) is defined as the (minimising) geodesic. \( \gamma \) is projected to the observation space by the decoder of VAEs. With the the Jacobian of the decoder \( J \), the metric tensor is \( G(z) = J(z)^T J(z) \).

To measure the observation-space distance using simple Euclidean distance in latent space, we have

\[
D \propto \|z(1) - z(0)\|_2,
\]

which indicates that the Riemannian metric tensor is \( G \propto 1 \). A manifold with this property is defined as flat manifold \([\text{Lee, 2006}]\). To learn a model with a flat latent manifold, \([\text{Chen et al., 2020}]\) presents (i) a flexible prior which allows the model to learn complex latent representations of the data by empirical Bayes; (ii) a regularisation \( G \propto 1 \); and (iii) data augmentation which enables the model to have the same property in the low data density area.

Hierarchical prior and constrained optimisation

It has been shown that the approximated posterior can be over-regularised when the prior distribution is a standard Normal, which in turn makes the latent representation less informative \([\text{Tomczak and Welling, 2018}]\). For instance, a large distance is required for the posteriors between two data points with a large
gap, if there is a lack of training data between them, which is
difficult to be trained by the model with a standard Normal
prior. Hence, the motivation to introduce flexible priors arises,
such as VampPrior [Tomczak and Welling, 2018] and VHP
[Klushyn et al., 2019].

Due to its empirical results and its stable learning, we resort
here to the hierarchical prior defined in [Klushyn et al., 2019],
\( p_0(z) = \int p_0(z|c) \; p(c) \; dc \), where \( p(c) \) is the standard
normal distribution. Sampling from \( q_0(z|x) \), we approximate
the integral by an importance-weighted (IW) bound [Burda et al.,
2015]. The upper bound on the KL term corresponding to our
two stochastic layers is then,

\[
\begin{align*}
\mathbb{E}_{P_D(x)} \mathbb{KL} (q_\phi(z|x)\| p(z)) & \leq \mathcal{F}(\phi, \Theta, \Phi) \\
& \equiv \mathbb{E}_{P_D(x)} \mathbb{E}_{q_\phi(z|x)} \left[ \log q_\phi(z|x) \\
& - \mathbb{E}_{q_\phi(z|x)} \left[ \log \frac{1}{K} \sum_{i=1}^{K} p_\theta(z|c_i) \; p(c_i) \right] \right],
\end{align*}
\]

where \( K \) is the number of importance samples. Based on
[Rezende and Viola, 2018], the ELBO is rewritten as the
Lagrangian of a constrained optimisation problem:

\[
\mathcal{L}_{VHP-VAE}(\phi, \Theta, \Phi; \lambda) \equiv \mathcal{F}(\phi, \Theta, \Phi) + \lambda \left( \mathbb{E}_{P_D(x)} \mathbb{E}_{q_\phi(z|x)} \left[ C_\theta(x, z) \right] - \kappa^2 \right),
\]

with the optimisation objective \( \mathcal{F}(\phi, \Theta, \Phi) \), the inequality con-
straint \( \mathbb{E}_{P_D(x)} \mathbb{E}_{q_\phi(z|x)} \left[ C_\theta(x, z) \right] \leq \kappa^2 \), and the Lagrange
multiplier \( \lambda \). \( C_\theta(x, z) \) is defined as the reconstruction-error-
related term in \( -\log p_\theta(x|z) \). Thus, we obtain the following
optimisation problem:

\[
\min_{\phi, \Theta, \Phi} \max_{\lambda, \mu} \min_{\theta, \lambda} \mathcal{L}_{VHP-VAE}(\theta, \phi, \Theta, \Phi; \lambda) \text{ s.t. } \lambda \geq 0.
\]

**Efficient Jacobian approximation**

To improve the computational efficiency, we approximate
the Jacobian with a first-order Taylor expansion [Rifai et al.,
2011; Chen et al., 2020]. In the previous work, each dimension of \( z \)
was computed independently using eq. (2),

\[
J_t(z) = \lim_{\sigma \to 0} \frac{1}{\sigma} \mathbb{E}_{z \sim \mathcal{N}(0, \sigma^2)} [f(z + \sigma e_t) - f(z)],
\]

where \( J_t \) is the Jacobian of the \( t \)-th latent dimension and \( e_t \)
represents a standard basis vector.

We alternatively use a different sampling-based approxima-
tion that scales more gracefully as \( N_z \) becomes larger
[Nesterov and Spokoiny, 2017],

\[
J(z) = \frac{1}{\mu} \mathbb{E}_u (f(z + \mu u) - f(z) u),
\]

where \( u \in \mathbb{R}^{N_z} \) is a normally distributed Gaussian vector
and \( \mu \) is a bound. The computation of eq. (3) depends
on the sample size used to approximate the expectation of \( u \).
Therefore, for small \( N_z \), eq. (2) is employed, while we use
eq. (3) for large \( N_z \).

**Latent space mixup**

The Jacobian regularisation only affects the vicinity of the
training examples, therefore, we use mixup to augment the
training data [Zhang et al., 2018]. While the original data-
augmentation method is applied to the observed data, some
previous work [Verma et al., 2019; Chen et al., 2020] extended
it to the latent representations.

We augment the data by randomly interpolating between
two points \( z_i \) and \( z_j \) in the latent space:

\[
g(z) = (1 - \alpha) z_i + \alpha z_j,
\]

with \( \alpha \sim U(-\alpha_0, 1+\alpha_0) \). To compute more efficiently, we
obtain \( z_i \) and \( z_j \) by shuffling a batch of \( z \). In order to take into
account the outer edge of the data manifold, we have \( \alpha_0 > 0 \).

With the metric tensor regularisation in the entire la-
tent space enclosed by the data manifold, the decoder \( f(z) \)
is distance-preserving. Given the distance in the obser-
vation space \( D_z \), and in the latent space \( D_z \), we obtain
\( D_z(f(z_i), f(z_j)) \approx c \cdot D_z(z_i, z_j) \).

3 **Experiments and live deployment**

To assess the quality of our latent representation, we mea-
sure the smoothness and quality of the interpolation and ex-
plor how musical attributes are projected into the latent space.
Throughout the experiments, we obtain the hyper parameters
of the models (see appendix A.3) via random search. Gener-
ated audio samples are also available online\(^1\). We compare
our approach with MusicVAE [Roberts et al., 2018], as the
state-of-the-art model for interpolating music data. In our
paper, MusicVAE uses linear interpolation instead of Slerp
interpolation, and the smoothness resulting from the Slerp
interpolation is studied in appendix A.9.

Quantitative evaluation is done on three music datasets.

\(^1\)https://sites.google.com/view/fm-music-vae
3.1 Datasets

**Drum dataset** We evaluate our model on a drum/groove dataset [Gillick et al., 2019] with two variations consisting in two- and four-bar sequences respectively. Each bar consists of 16 time steps. It is recorded on a Roland TD-11 electronic drum kit and we map the kit to nine categories (see table 3 of appendix A.2). In each time step, the data (MIDI notes) consists of binary-valued hits (the drum score for nine instruments), continuous-valued velocity (how strong the drums are struck), and continuous-valued offsets (indicating the micro-timing of how soon/late the note is played). Since the splits of the training, validation and testing in [Gillick et al., 2019] easily cause overfitting and early stopping, we randomly re-split the training and validation.

**Piano and cello datasets** We obtain the piano dataset from Ryan’s Mammoth of the music21 toolkit [Cuthbert and Ariza, 2010] and the cello dataset from the Bach Cello Suites (www.kunstderfuge.com). Each dataset is augmented by pitch shifting. Therefore, the note distribution covers the full range of the 88 piano keys. One sequence contains 4 bars, i.e., 64 time steps in total and each time step has 90 dimensions consisting of 88 ‘note-on’, one ‘note-off’, and one ‘rest’ [Roberts et al., 2018].

For each dataset, we randomly split the data into 80%, 10% and 10%, respectively for training, validation and testing in sections 3.2, 3.4 and 3.5. The dataset splitting is different for interpolation quality evaluation in section 3.3.

3.2 Evaluating the interpolations’ smoothness

We take the hits and on-notes for the drum and piano datasets respectively to compute smoothness. Since they are binary data, we define the smoothness of a sequence as \( \max_{t \in \{0, 1, \ldots, N - 1\}} D_h(x_t, x_{t+1}) \), where \( D_h \) represents Hamming distance, and \( x_t \) is a sequence at step \( t \) of the interpolation. Figure 2 shows the smoothness of 1 000 pairs and 20 interpolated points. Interpolated samples are shown in fig. 3. With MusicVAE, the smoothness improves with a sufficiently large number of dimensions in the latent space, however, our approach is able to get overall better smoothness with a significantly lower-dimensional latent space.

3.3 Evaluating the interpolations’ quality

Evaluating the quality of the interpolations via human ratings is subject to a substantial bias. Hence, we quantify the quality of the interpolation by measuring some generalisation properties in between the training data points, namely, the velocity and offset for the drum, the pitch for the piano, and the notes density for both. All of these properties are part of the input data except for the notes density which we manually count for each two bars. Using the middle decile or the middle 50 percentile as a ground truth of testing datasets, we obtain the samples between the other two ends (training and validation datasets) of the distribution for each of the aforementioned properties. The results for each property are summarised in table 1. Since the piano and cello datasets are one-hot encoded, the reconstruction error rate \( \delta \) is reported for the whole frame, i.e., \( \delta = N_f / N_s \), with \( N_f \) being the number of the incorrectly reconstructed frames in a sequence. On the other hand, the error rate for the drum data is computed on a per-beat level for the density attribute, i.e., \( \delta = N_{fp} / (N_s N_f) \), where \( N_{fp} \) is the number of wrongly predicted beats in a sequence. As for the velocity and offset, we report the mean square errors (MSEs) per beat. All the reported numbers, the mean and standard deviation, are evaluated over the test datasets.

3.4 Exploiting the structure of the latent space with attribute vectors

The structure of the latent spaces can be exploited using attribute vectors [Roberts et al., 2018; White, 2016]. In particular, we exploit the density, velocity and offset attributes in the drum dataset, as well as the density and pitch attributes in the piano dataset. The density attribute is computed by...
Table 1: Evaluating the quality of the interpolation. The bold font indicates the best result. The models of the drum datasets are 32D latent space, while of the piano and cello datasets are 64D latent space. The first column shows the dataset and the percentile used for test datasets.

| DATASET (PERCENTILE) | MODEL               | DENSITY   | VELOCITY     | OFFSET      | PITCH      |
|----------------------|---------------------|-----------|--------------|-------------|------------|
| DRUM, FOUR BARS      | FM-MUSIC-VAE        | 0.039 ± 0.023 | 0.0116 ± 0.0080 | 0.0039 ± 0.0022 | -          |
|                      | MUSIC-VAE           | 0.051 ± 0.031 | 0.0151 ± 0.0123 | 0.0042 ± 0.0024 | -          |
| DRUM, FOUR BARS      | FM-MUSIC-VAE        | 0.064 ± 0.030 | 0.0195 ± 0.0116 | 0.0050 ± 0.0027 | -          |
|                      | MUSIC-VAE           | 0.094 ± 0.037 | 0.0286 ± 0.0163 | 0.0058 ± 0.0031 | -          |
| PIANO, TWO BARS      | FM-MUSIC-VAE        | 0.195 ± 0.093 | -            | -           | 0.165 ± 0.099 |
|                      | MUSIC-VAE           | 0.234 ± 0.222 | -            | -           | 0.431 ± 0.088 |
| CELLO, FOUR BARS     | FM-MUSIC-VAE        | 0.145 ± 0.183 | -            | -           | 0.117 ± 0.165 |
|                      | MUSIC-VAE           | 0.616 ± 0.194 | -            | -           | 0.560 ± 0.211 |

3.5 Ablation study
Motivating our design choices, we conduct an ablation study in order to assess the effect of the hierarchical prior described in section 2.3. Table 2 shows that while FM-Music-VAE with a standard Normal prior does improve over the state-of-the-art, adding the hierarchical prior further improves the performance. Hence, the rest of the experiments are conducted with the full model.

3.6 Model deployment
The final test of FM-Music-VAE was in a live artistic event. During the session, which took place in November 2020, a professional drummer interacted with our model by playing two bars of drum music, to which our model ‘responded’ with another generated two-bars drum sequence. Once received, the drummer’s playing was mapped into a point in the latent space. Using this point as the centre of a Gaussian distribution, a second point was then sampled and decoded into a drum sequence that the drummer could listen to – and hopefully be inspired by. The back-and-forth interplay lasted for about ten minutes. While playing, the drummer was also able to visualise the three-dimensional embedding of his own playing as well as that of the model’s generations. The set of embedded points were also visually linked to reflect their occurrence in time, hence forming the two ‘trajectories’ of the two players in a three-dimensional space. The goal was to let the drummer try to ‘influence’ the direction the model was taking, a task that is only feasible if the latent representations can be properly interpreted.

Although the performance itself excludes any scientific pretension, we argue that the deployment of our model in this context brings value on three different fronts. First, it shows that our latent-variable model can be used for interpretability purposes, even by a non-machine-learning practitioner, as it exhibits meaningful distances. We also make the point that deploying models is valuable in itself, even in the context of scientific research, as it is anchored in real needs [Wagstaff,
Table 2: The correlation of the attribute and the latent space (Pearson’s correlation coefficient). The bold font represents the best result.

| DATASET       | MODEL                     | DENSITY | VELOCITY | OFFSET | PITCH |
|---------------|---------------------------|---------|----------|--------|-------|
| Drum, two bars| FM-Music-VAE (32D)        | 0.89    | 0.84     | 0.69   | -     |
|               | MusicVAE (32D)            | 0.50    | 0.27     | 0.21   | -     |
|               | MusicVAE (512D)           | 0.74    | 0.56     | 0.52   | -     |
| Drum, four bars| FM-Music-VAE (32D)        | 0.89    | 0.87     | 0.72   | -     |
|               | FM-Music-VAE w/o VHP (32D)| 0.87    | 0.88     | 0.60   | -     |
|               | MusicVAE (32D)            | 0.73    | 0.31     | 0.39   | -     |
| Piano, two bars| FM-Music-VAE (64D)        | 0.89    |          |        | 0.77  |
|               | MusicVAE (64D)            | 0.37    |          |        | 0.42  |
|               | MusicVAE (512D)           | 0.48    |          |        | 0.50  |
| Cello, four bars| FM-Music-VAE (64D)        | 0.91    |          |        | 0.81  |
|               | MusicVAE (64D)            | 0.79    |          |        | 0.60  |

2012]. Finally, contributing to a novel musical experience in which an artist has the possibility to explore music both audibly and visually constitutes an artistic value. An excerpt of the performance video is available online and the technical details of the architecture are in appendix A.1.

4 Related work

**Music with VAEs**  MusicVAE [Roberts et al., 2018], which improved over sketch-RNN [Ha and Eck, 2018] using a hierarchical decoder, generates musical sequence from a single point in the latent space. A few works followed the line of MusicVAE since then. GrooVAE [Gillick et al., 2019] was proposed to humanise (or groove) drum datasets by adapting the velocity and the offset. MusicVAE was further improved for disentanglement [Yang et al., 2019] such that it constrains the features linearly to disentangle during training. However, it requires manually setting the disentanglement factors.

**Interpolation in latent space**  Spherical interpolation (Slerp) was proposed for GANs and VAEs in order to improve upon the linear interpolation when generating realistic images [White, 2016]. Following the geodesics of VAEs was also shown to generate smooth interpolations [Chen et al., 2019; Chen et al., 2018a; Chen et al., 2018b; Arvanitidis et al., 2018]. These methods search for non-Euclidean trajectories as a postprocessing, i.e., once the model is trained.

The work of [Chen et al., 2020] has Jacobian/isometric regularisation. This flattens the latent space, leading to smooth outputs even with linear interpolation. Similar works have been proposed in [Kumar and Poole, 2020]. Realistic interpolation has also been improved by an adversarial regulariser [Berthelot et al., 2019]. These approaches improved the models during training and were able to interpolate using Euclidean trajectories after training. None of these methods have been evaluated on sequential data.

**Distance of sequences**  Dynamic time warping (DTW) [Müller, 2007] measures similarity between two temporal sequences by computing the optimal match between two sequences. Various extensions of DTW have been studied, such as triplet constraints [Mei et al., 2015]. Sequence distance measurement through a ground Mahalanobis metric was presented by [Su and Wu, 2019]. The sequence distance metrics are learnt from labelled ground truth [Garreau et al., 2014]. Some prior work measures distances of sequences using RNNs, e.g., [Bayer et al., 2012]. In our model, we measure the sequence distances by the metric tensor of the generative model and without labels. In the music area, the importance of the similarity was emphasised in [Volk et al., 2016]. [Knees and Schell, 2013] conducts a survey of music similarity and its application to recommendation systems.

5 Conclusion and future work

We presented FM-Music-VAE, a theoretically-motivated approach to live interaction with a human musician while learning a high-level music representation. Our method in effect learns a distance metric for sequence data that allows for smooth and meaningful linear interpolations. Using drums and piano datasets, we experimentally demonstrated that our principled constrained optimisation scheme properly regularises the decoder and thus yields realistic music sequences. Finally, the interactive music improvisation session between the model and a drummer illustrates that the proposed approach is able to be intuitively used by a human. In the future, using high-level control in latent space, e.g., through, reinforcement learning, for generating longer musical sequences would be a natural follow-up.

Acknowledgements

The authors would like to thank Matthias Lachenhmayer who is the professional drummer for testing our model. We thank Alexej Klushyn for some code of the related work. We thank Botond Cseke for useful discussions.

References

[Alemi et al., 2018] Alexander A Alemi, Ben Poole, Ian Fischer, Joshua V Dillon, Rif A Saurous, and Kevin Murphy. Fixing a broken ELBO. *ICML*, 2018.

[Arvanitidis et al., 2018] Georgios Arvanitidis, Lars Kai Hansen, and Søren Hauberg. Latent space oddity: on the curvature of deep generative models. In *ICLR*, 2018.
[Bayer and Osendorfer, 2014] Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. arXiv preprint arXiv:1411.7610, 2014.

[Bayer et al., 2012] Justin Bayer, Christian Osendorfer, and Patrick van der Smagt. Learning sequence neighbourhood metrics. In ICANN, pages 531–538. Springer, 2012.

[Berthelot et al., 2019] David Berthelot, Colin Raffel, Aurko Roy, and Ian Goodfellow. Understanding and improving interpolation in autoencoders via an adversarial regularizer. In ICLR, 2019.

[Bowman et al., 2016] Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. In CoNLL, pages 10–21, 2016.

[Burda et al., 2015] Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. Importance weighted autoencoders. CoRR, abs/1509.00519, 2015.

[Chen et al., 2015] Nutan Chen, Justin Bayer, Sebastian Urban, and Patrick van der Smagt. Efficient movement representation by embedding dynamic movement primitives in deep autoencoders. In Humanoids, pages 434–440, 2015.

[Chen et al., 2018a] Nutan Chen, Alexej Klushyn, Richard Kurle, Xueyan Jiang, Justin Bayer, and Patrick van der Smagt. Metrics for deep generative models. In AISTATS, pages 1540–1550, 2018.

[Chen et al., 2018b] Nutan Chen, Alexej Klushyn, Alexandros Parpaschos, Djalel Benbouzid, and Patrick van der Smagt. Active learning based on data uncertainty and model sensitivity. IEEE/RSJ IROS, 2018.

[Chen et al., 2019] Nutan Chen, Francesco Ferroni, Alexej Klushyn, Alexandros Parpaschos, Justin Bayer, and Patrick van der Smagt. Fast approximate geodesics for deep generative models. In ICANN, pages 554–566. Springer, 2019.

[Chen et al., 2020] Nutan Chen, Alexej Klushyn, Francesco Ferroni, Justin Bayer, and Patrick van der Smagt. Learning flat latent manifolds with vaes. In ICML, 2020.

[Cho et al., 2014] Kyunghyun Cho, Bart Van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259, 2014.

[Chung et al., 2014] Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. NIPS Workshop on Deep Learning, 2014.

[Cuthbert and Ariza, 2010] Michael Scott Cuthbert and Christopher Ariza. music21: A toolkit for computer-aided musicology and symbolic music data. 2010.

[Fraccaro et al., 2016] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. NIPS, 2016.

[Garreau et al., 2014] Damien Garreau, Rémi Lajugie, Sylvain Arlot, and Francis Bach. Metric learning for temporal sequence alignment. NIPS, 2014.

[Gillick et al., 2019] Jon Gillick, Adam Roberts, Jesse H. Engel, Douglas Eck, and David Bamman. Learning to groove with inverse sequence transformations. In ICML, 2019.

[Ha and Eck, 2018] David Ha and Douglas Eck. A neural representation of sketch drawings. In ICLR, 2018.

[Hochreiter and Schmidhuber, 1997] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. Neural computation, (9):1735–1780, 1997.

[Karl et al., 2017] Maximilian Karl, Maximilian Soelch, Justin Bayer, and Patrick Van der Smagt. Deep variational bayes filters: Unsupervised learning of state space models from raw data. ICLR, 2017.

[Kingma and Welling, 2014] Diederik P. Kingma and Max Welling. Auto-encoding variational Bayes. ICLR, 2014.

[Kingma et al., 2016] Diederik Kingma, Tim Salimans, Rafal Jozefowicz, Xi Chen, Ilya Sutskever, and Max Welling. Improved variational inference with inverse autoregressive flow. In NIPS, volume 29, 2015.

[Kloushyn et al., 2019] Alexej Klushyn, Nutan Chen, Richard Kurle, Botond Cseke, and Patrick van der Smagt. Learning hierarchical priors in VAEs. NeurIPS, 2019.

[Knees and Schedl, 2013] Peter Knees and Markus Schedl. A survey of music similarity and recommendation from music context data. TOMM, 2013.

[Krishnan et al., 2015] Rahul G Krishnan, Uri Shalit, and David Sontag. Deep kalman filters. arXiv, 2015.

[Kumar and Poole, 2020] Abhishek Kumar and Ben Poole. On implicit regularization in β-vaes. In ICML, 2020.

[Lee, 2006] John M Lee. Riemannian manifolds: an introduction to curvature, volume 176. Springer Science & Business Media, 2006.

[Mei et al., 2015] Jiangyuan Mei, Meizhu Liu, Yuan-Fang Wang, and Huijun Gao. Learning a mahalanobis distance-based dynamic time warping measure for multivariate time series classification. IEEE Trans Cybern, 46, 2015.

[Müller, 2007] Meinard Müller. Dynamic time warping. Information retrieval for music and motion, pages 69–84, 2007.

[Nesterov and Spokoiny, 2017] Yurii Nesterov and Vladimir Spokoiny. Random gradient-free minimization of convex functions. FoCM, 17(2):527–566, 2017.

[Rezende and Viola, 2018] Danilo Jimenez Rezende and Fabio Viola. Taming VAEs. arXiv, 2018.

[Rezende et al., 2014] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In ICML, volume 32, pages 1278–1286, 2014.

[Rifai et al., 2011] Salah Rifai, Grégoire Mesnil, Pascal Vincent, Xavier Muller, Yoshua Bengio, Yann Dauphin, and Xavier Glorot. Higher order contractive auto-encoder. In ECML-PKDD, pages 645–660. Springer, 2011.

[Roberts et al., 2018] Adam Roberts, Jesse Engel, Colin Raffel, Curtis Hawthorne, and Douglas Eck. A hierarchical...
latent vector model for learning long-term structure in music. In \textit{ICML}, volume 80, pages 4364–4373, 2018.

\cite{schuster1997bidirectional} Mike Schuster and Kuldip K Paliwal. Bidirectional recurrent neural networks. \textit{IEEE Trans. on Signal Processing}, 45(11):2673–2681, 1997.

\cite{su2019learning} Bing Su and Ying Wu. Learning distance for sequences by learning a ground metric. In \textit{ICML}, 2019.

\cite{tomczak2018vae} Jakub M. Tomczak and Max Welling. VAE with a vampprior. In \textit{AISTATS}, 2018.

\cite{verma2019manifold} Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In \textit{ICML}, volume 97, pages 6438–6447, 2019.

\cite{volk2016music} Anja Volk, Elaine Chew, Elizabeth Hellmuth Margulis, and Christina Anagnostopoulou. Music similarity: Concepts, cognition and computation. \textit{JNMR}, 45(3):207–209, 2016.

\cite{wagstaff2012machine} Kiri Wagstaff. Machine learning that matters. \textit{ICML}, 2012.

\cite{white2016sampling} Tom White. Sampling generative networks. \textit{arXiv preprint arXiv:1609.04468}, 2016.

\cite{yang2019inspecting} Ruihan Yang, Tianyao Chen, Yiyi Zhang, and Gus Xia. Inspecting and interacting with meaningful music representations using vae. \textit{arXiv}, 2019.

\cite{zhang2018mixup} Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. \textit{ICLR}, 2018.
A Appendix

A.1 Drummer-model interactive system

To facilitate the musical dialog between a human drummer and the FM-Music-VAE, the infrastructure was built up around realtime socket connections. The latency, introduced by processing of the data, was overcome by working in bars of notes instead of sending over individual hits. The drummer played on a midi drum set and the individual notes were captured by a Max application and sequenced into two bars. The bars were sent over to a central server which acted as a broker between the AI model and the human drummer. The forwarded bars were processed by the model into two response-bars and sent back to the Max application in reverse route. For both human drummer and AI a separate sampler was used to convert the midi into hearable audio. Figure 5 depicts the architecture deployed during the live session.

Figure 5: The architecture of the jam session.
### A.2 Drum categories

Table 3: Drum categories. The mapping from the pitch to the nine drum categories. [Gillick et al., 2019] shows more details of the mapping.

| DRUM CATEGORY | PITCH |
|---------------|-------|
| Bass          | 36    |
| Snare         | 38    |
| High Tom      | 50    |
| Low-Mid Tom   | 47    |
| High Floor Tom| 43    |
| Open Hi-Hat   | 46    |
| Closed Hi-Hat | 42    |
| Crash Cymbal  | 49    |
| Ride Cymbal   | 51    |

### A.3 Model Architectures

Table 4: Model architectures. FC is fully-connected layers. $\nu$ is defined in [Klushyn et al., 2019]. The models were trained on a single 16-G GPU.

| DATASET         | OPTIMISER | ARCHITECTURE |
|-----------------|-----------|--------------|
| Human Motion    | ADAM      | INPUT 32×50  |
|                 |           | LATENTS 2    |
|                 |           | $q_0(\mathbf{z}|\mathbf{x})$ BiGRU 1024, 1024. |
|                 |           | $p_0(\mathbf{x}|\mathbf{z})$ CONDUCTOR 512, GRU 512, 512 |
|                 |           | $q_0(\mathbf{z}|\mathbf{c})$ FC 256, 256, RELU activation. |
|                 |           | $p_0(\mathbf{c}|\mathbf{z})$ FC 256, 256, RELU activation. |
|                 |           | OTHERS $\kappa = 0.08, \nu = 1, K = 8, \eta = 8000$. |
| Drum (two bars) | ADAM      | INPUT 32×27  |
|                 |           | LATENTS 32   |
|                 |           | $q_0(\mathbf{z}|\mathbf{x})$ BiGRU 512, 512. |
|                 |           | $p_0(\mathbf{x}|\mathbf{z})$ CONDUCTOR 512, GRU 512, 512. |
|                 |           | $q_0(\mathbf{z}|\mathbf{c})$ FC 256, 256, RELU activation. |
|                 |           | $p_0(\mathbf{c}|\mathbf{z})$ FC 256, 256, RELU activation. |
|                 |           | OTHERS $\kappa = 0.14, \nu = 1, K = 8, \eta = 6000$. |
| Drum (four bars)| ADAM     | INPUT 64×27  |
|                 |           | LATENTS 32   |
|                 |           | $q_0(\mathbf{z}|\mathbf{x})$ BiGRU 1024, 1024. |
|                 |           | $p_0(\mathbf{x}|\mathbf{z})$ CONDUCTOR 512, GRU 512, 512. |
|                 |           | $q_0(\mathbf{z}|\mathbf{c})$ FC 256, 256, RELU activation. |
|                 |           | $p_0(\mathbf{c}|\mathbf{z})$ FC 256, 256, RELU activation. |
|                 |           | OTHERS $\kappa = 0.15, \nu = 1, K = 16, \eta = 2500$. |
| Piano           | ADAM      | INPUT 32×90  |
|                 |           | LATENTS 64   |
|                 |           | $q_0(\mathbf{z}|\mathbf{x})$ BiGRU 1024, 1024. |
|                 |           | $p_0(\mathbf{x}|\mathbf{z})$ CONDUCTOR 512, GRU 512, 512. |
|                 |           | $q_0(\mathbf{z}|\mathbf{c})$ FC 256, 256, RELU activation. |
|                 |           | $p_0(\mathbf{c}|\mathbf{z})$ FC 256, 256, RELU activation. |
|                 |           | OTHERS $\kappa = 0.03, \nu = 1, K = 8, \eta = 2000$. |
| Cello           | ADAM      | INPUT 64×90  |
|                 |           | LATENTS 64   |
|                 |           | $q_0(\mathbf{z}|\mathbf{x})$ BiGRU 1024, 1024. |
|                 |           | $p_0(\mathbf{x}|\mathbf{z})$ CONDUCTOR 512, GRU 512, 512. |
|                 |           | $q_0(\mathbf{z}|\mathbf{c})$ FC 256, 256, RELU activation. |
|                 |           | $p_0(\mathbf{c}|\mathbf{z})$ FC 256, 256, RELU activation. |
|                 |           | OTHERS $\kappa = 0.06, \nu = 1, K = 8, \eta = 2000$. |
A.4 Interpolation of the drum dataset

Figure 6: Drum interpolation. See fig. 3 for more details.

Figure 7: Drum interpolation. See fig. 3 for more details.
A.5 Drum density attribute

Figure 8: Drum density attribute. See fig. 4 for more details.

Figure 9: Drum density attribute. See fig. 4 for more details.
A.6 Drum velocity attribute

Figure 10: Drum velocity distribute. The colour represents the velocity. See fig. 4 for more details.

Figure 11: Drum velocity attribute. The colour represents the velocity. See fig. 4 for more details.
A.7 Interpolation of the piano dataset

Figure 12: Piano interpolation. See fig. 3 for more details.

Figure 13: Piano interpolation. See fig. 3 for more details.
A.8 Piano density attribute

Figure 14: Piano density attribute. See fig. 4 for more details.

Figure 15: Piano density attribute. See fig. 4 for more details.
A.9 Understanding the latent space via a toy human motion dataset

Since it is difficult to compress music data into a two-dimensional latent space for plotting, we use a toy dataset, CMU human motion, to understand the behaviour of the latent representation. In this experiment, we observe the latent space and evaluate its quality on the CMU human motion dataset (http://mocap.cs.cmu.edu/). We obtain five movements: walking, jogging, balancing, punching and kicking. To match the architecture of the model for music data, we split the sequences into two ‘bars’ and 16 frames per ‘bar’. Using the data pre-processing in [Chen et al., 2015], each frame has 50-dimensional joint angles. Since it is a small dataset, we add Gaussian noise with a standard deviation of 0.03 to the joint vectors to generate training and validation datasets.

Figure 16 shows the two-dimensional latent spaces of FM-Music-VAE and MusicVAE respectively. The contour lines represent equal Riemannian distances around one random sample from each class. The distances are computed using ?? with linear trajectories in the latent space. The more the model preserves the distances, the rounder the contour lines should look. Corroborating the results of [Chen et al., 2020], this experiment shows that flat manifolds meaningfully preserve the distances in the latent spaces. However, unlike the previous work, the data in our case is treated sequentially.

Spherical interpolation (Slerp) was previously proposed to interpolate between two points to get realistic and sharp output from the decoder [Roberts et al., 2018; White, 2016]. It prevents sampling from the area of the latent space which is likely to generate unrealistic outputs (near the midpoint of the two selected locations [White, 2016]), which leads the output to be close to the training dataset. However, for a usage beyond interpolation, one cannot avoid sampling these areas. In addition, Slerp decreases the smoothness of interpolation (see fig. 17). In contrast to image datasets, we expect the models to generate interesting combinations of points and have strong generalisation. Thus, to exploit the entirety of the latent space, we use linear interpolation in experiments, unless noted differently.

In order to quantitatively evaluate the latent space, we use smoothness as a metric for the human motion dataset. We randomly select 1000 pairs of points from the dataset and linearly interpolate 25 time steps in-between, including the starting and end points. We compute two kinds of smoothness: the smoothness of a single motion and the smoothness when interpolating between motions. The latter is analogous to the smoothness of interpolating between two bars in musical data. The FM-Music-VAE outperforms MusicVAE both with linear interpolation and Slerp in terms of the smoothness between movements, and keeps high quality of smoothness within movements (see fig. 17). Some examples of the interpolation are shown in fig. 18.
Figure 18: Interpolation of the human motion dataset in two-dimensional latent spaces. In each subfigure, given the top and bottom sequences, we show three linear interpolated sequences. For better visualisation, each sequence is down-sampled to 16 time steps and the sequences are down-sampled to five. Taking (b) as an example, there is an abrupt change between the second and the third sequence, while the FM-Music-VAE in (a) is more smooth.