Apprentissage de variété riemannienne pour l'analyse-synthèse de signaux non stationnaires

Han Han, Vincent Lostanlen, Mathieu Lagrange

Apprentissage de variété riemannienne pour l'analyse–synthèse de signaux non stationnaires

Han HAN[1] Vincent LOSTANLEN[1] Mathieu LAGRANGE[1]

[1]Nantes Université, École Centrale Nantes, CNRS, LS2N, UMR 6004, F-44000 Nantes, France

Résumé – La transformation de sons par ordinateur pose un problème inverse d'identification des paramètres de resynthèse adéquats. Empruntant au formalisme du traitement du signal différentiable (DDSP), nous proposons d'automatiser sa résolution par entrainement d'un réseau de neurones profond. Dans ce contexte, nous visons un compromis entre l'efficacité computationnelle de la perte paramétrique et la fidélité psychoacoustique de la perte spectrale. Notre approche, baptisée perceptuelle–neuronale–physique (PNP), consiste à estimer la métrique riemannienne associée à la composition entre synthèse paramétrique et diffusion temps–fréquence (JTFS). Ce faisant, nous linéarisons localement la perte spectrale et accélérons la convergence. De plus, le recours à une régularisation de Tikhonov améliore le conditionnement du problème inverse. Par rapport à l'état de l'art (*wav2shape* et DDSP), et pour une tâche difficile d'analyse–synthèse d'arpège musical, l'entrainement via PNP rapproche le signal reconstruit du signal de référence, d'après une mesure de similarité de timbre fondée sur la JTFS.

Abstract – Computer sound transformations poses an inverse problem, namely, the identification of matching resynthesis parameters. Borrowing from the differentiable digital signal processing (DDSP) framework, we propose to automate its resolution by training a deep neural network. In this context, we aim to reach a compromise between the computational efficiency of parametric loss (P-loss) versus the psychoacoustical fidelity of spectral loss. Our approach, named "perceptual–neural–physical" (PNP), estimates the Riemannian metric which is associated to the composition between parametric synthesis and time–frequency scattering. By doing so, we locally linearize spectral loss and accelerate convergence. Furthermore, resorting to Tikhonov regularization improves the conditioning of the inverse problem. On an analysis–synthesis task for musical arpeggios, PNP training outperforms state-of-the-art methods P-loss (wav2shape) and STFT-based DDSP, as measured in terms of JTFS-based similarity between reference signal and reconstructed signal.

1 Introduction

La synthèse de sons par ordinateur remonte aux années 1960 et constitue aujourd'hui un pilier de la création musicale, aussi bien « contemporaine » que « populaire » [8]. Dans ce contexte, un instrument de musique virtuel est défini comme un algorithme de génération $\boldsymbol{g}$, prenant en entrée un paramètre multidimensionnel $\boldsymbol{\theta}$ et retournant en sortie un signal temporel $\boldsymbol{x}$. Si $\boldsymbol{x}$ constitue la solution d'une équation aux dérivées partielles (EDP) dont les coefficients sont contenus dans $\boldsymbol{\theta}$, l'algorithme $\boldsymbol{g}$ résout un « problème direct » en acoustique musicale.

Réciproquement, le « problème inverse » consiste à identifier $\boldsymbol{\theta}$ à partir d'une prise de son naturelle de $\boldsymbol{x}$, au moyen d'un algorithme d'analyse $\boldsymbol{f}$. Autour de l'an 2000, l'émergence des « effets audio numériques » (DAFX pour *digital audio effects*) a motivé l'étude du problème inverse et du problème direct à travers un cadre méthodologique unifié d'analyse–synthèse [1]. Dans ce cadre, la musicienne opère en trois étapes : analyse par résolution du problème direct ($\boldsymbol{\theta} = \boldsymbol{f}(\boldsymbol{x})$) puis modification des paramètres (de $\boldsymbol{\theta}$ vers un certain $\boldsymbol{\theta'}$) et enfin synthèse par résolution du problème direct ($\boldsymbol{x'} = \boldsymbol{g}(\boldsymbol{\theta'})$).

Les premières méthodes d'analyse–synthèse, fondées sur l'analyse de Fourier à fenêtre et la poursuite adaptée *(matching pursuit)*, ont démontré leur intérêt pratique pour la transformation procédurale de sons : ainsi, le vocodeur de phase qui permet d'aligner des motifs de tempos différents ou le logiciel "Auto-Tune" qui corrige la hauteur d'une voix chantée. Mais ces méthodes ont un horizon temporel restreint, de l'ordre de $T = 25\,\mathrm{ms}$, ce qui complique l'intervention à des niveaux d'expressivité musicale plus abstraits, tels que le mode de jeu. De plus, elles font l'hypothèse que $\boldsymbol{g}$ produit un mélange de sinusoïdes dont les paramètres sont lentement modulés en temps–fréquence. Elles échouent donc sur les signaux impulsifs, bruités ou texturés, qui pourtant sont fréquents en situation réelle.

Dans cet article, on s'intéresse au cas où $\boldsymbol{g}$ construit une solution du problème direct mais où le problème inverse n'a pas de solution connue. De plus, les paramètres d'intérêt $\boldsymbol{\theta}$ peuvent affecter des propriétés à long terme du signal, tels que le taux de modulation d'amplitude *(vibrato)* ou la vitesse de décroissance relative des harmoniques aigues. On estime l'analyseur $\boldsymbol{f}$ par apprentissage supervisé sur une base de paires $(\boldsymbol{x_n}, \boldsymbol{\theta_n})$ avec $\boldsymbol{x_n} = \boldsymbol{g}(\boldsymbol{\theta_n})$. Plus précisément, on entraine un réseau de neurones « onde vers forme » *(wav2shape)* [4] dont les poids sont contenus dans un vecteur $\mathbf{W}$.

L'enjeu de notre article est de comparer différents choix de fonction de perte $\mathcal{L}$ pour le réseau de neurones $\boldsymbol{f}_{\mathbf{W}}$. La première, appelée « perte P » (*P-loss* pour *parameter loss*), est une simple distance euclidienne entre vecteurs $\boldsymbol{\theta_n}$ et $\tilde{\boldsymbol{\theta}}_{\boldsymbol{n}} = \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n})$. La deuxième, appelée « perte spectrale » (*spectral loss*), fait appel à un descripteur psychoacoustique $\boldsymbol{\Phi}$ pour comparer $\boldsymbol{x_n}$ et $\tilde{\boldsymbol{x}}_{\boldsymbol{n}} = \boldsymbol{g}(\boldsymbol{\theta_n})$. Bien que la perte spectrale soit plébiscitée par la communauté de recherche en traitement du signal différentiable (DDSP pour *differentiable digital signal processing*) [3], les seuls descripteurs $\boldsymbol{\Phi}$ connus qui soient fidèles à la perception humaine par rapport aux variations de $\boldsymbol{\theta}$ ont un cout calculatoire prohibitif pour l'entrainement d'un réseau de neurones. Face à ce dilemme, nous

proposons une « perte PNP » pour perceptuelle–neuronale–physique. La construction de la perte PNP repose sur une métrique riemannienne dans l'espace $\mathcal{M}$ associé à $\boldsymbol{\theta}$. En reformulant le problème d'appariement de son *(sound matching)* $\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) \approx \boldsymbol{\theta_n}$ comme un problème d'apprentissage de variété, nous atteignons un compromis favorable entre la fidélité perceptuelle de la perte spectrale et l'efficacité de la perte P.

2 Fonction de perte P, DDSP, et PNP

On se place dans un cadre d'apprentissage en ligne : $\mathcal{L}$ est indexée par l'exemple $\boldsymbol{\theta_n}$, ce qui implique une optimisation par descente de gradient stochastique (SGD). La perte P vaut

$$\mathcal{L}_n^{\mathrm{P}}(\mathbf{W}) = \left\| \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n} \right\|_2^2. \tag{1}$$

La perte spectrale, employée dans DDSP, vaut

$$\mathcal{L}_n^{\mathrm{sp}}(\mathbf{W}) = \left\| \left(\boldsymbol{\Phi} \circ \boldsymbol{g} \circ \boldsymbol{f}_{\mathbf{W}}\right)(\boldsymbol{x_n}) - \boldsymbol{\Phi}(\boldsymbol{x_n}) \right\|_2^2 \tag{2}$$

où $\boldsymbol{\Phi}$ désigne le descripteur perceptuel choisi. L'inconvénient de $\mathcal{L}_n^{\mathrm{sp}}$ est qu'elle doit réévaluer la fonction $\boldsymbol{\Phi}$ après chaque pas de SGD. Ceci pose un problème de passage à l'échelle quand le réseau de neurones $\boldsymbol{f}_{\mathbf{W}}$ requiert un grand nombre d'époques sur les données d'entrainement avant de converger.

Pour tout $\boldsymbol{\theta} \in \mathcal{M}$, on suppose qu'il existe un voisinage ouvert $\mathcal{U}_{\boldsymbol{\theta}}$ à la restriction duquel $(\boldsymbol{\Phi} \circ \boldsymbol{g})$ est un homéomorphisme. Avec ces cartes locales, on forme un atlas de classe $\mathcal{C}^1$ qui munit l'espace topologique $\mathcal{M}$ d'une structure de variété différentielle. De plus, on construit une métrique riemannienne $\boldsymbol{\eta}$ au moyen des formes quadratiques définies positives

$$\boldsymbol{\eta_\theta} : (\boldsymbol{\mu}, \boldsymbol{\nu}) \longmapsto \left\langle \boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})\boldsymbol{\mu} \middle| \boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta})\boldsymbol{\nu} \right\rangle \tag{3}$$

pour tout $\boldsymbol{\theta} \in \mathcal{M}$, avec $\boldsymbol{\mu}$ et $\boldsymbol{\nu}$ deux vecteurs tangents à $\mathcal{M}$ en $\boldsymbol{\theta}$. Pour tout $\boldsymbol{\theta}_n$ de la base d'entrainement, on réécrit la forme quadratique $\boldsymbol{\eta}_{\boldsymbol{\theta}_n}(\boldsymbol{\mu}, \boldsymbol{\nu})$ comme $\langle \boldsymbol{\mu} | \mathbf{M}_{\boldsymbol{n}} | \boldsymbol{\nu} \rangle$ avec $\mathbf{M}_{\boldsymbol{n}} = \boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta_n})^\top \boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}(\boldsymbol{\theta_n})$ une matrice symétrique définie positive. L'idée-clé de PNP est que les matrices $\mathbf{M}_{\boldsymbol{n}}$ sont indépendantes de l'architecture du réseau de neurones $\boldsymbol{f}_{\mathbf{W}}$; dès lors, elles peuvent être calculées avant l'entrainement, de façon parallèle sur une architecture multicœur, et sauvegardées en cache durant plusieurs époques de SGD, voire lors de l'entrainement de plusieurs modèles avec des hyperparamètres différents. Le nombre d'éléments dans $\mathbf{M}_{\boldsymbol{n}}$ est quadratique en la dimension de la variété $\mathcal{M}$ et constant en la dimension du descripteur $\boldsymbol{\Phi}$.

On définit la perte PNP comme la norme euclidienne carrée de $\left(\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n}\right)$ selon la métrique riemannienne $\boldsymbol{\eta}_{\boldsymbol{\theta_n}}$, soit

$$\begin{aligned} \mathcal{L}_n^{\mathrm{PNP}}(\mathbf{W}) &= \boldsymbol{\eta}_{\boldsymbol{\theta_n}}\left(\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n}, \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n}\right) \\ &= \left\langle \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n} \middle| \mathbf{M}_{\boldsymbol{n}} \middle| \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n} \right\rangle \\ &= \mathcal{L}_n^{\mathrm{sp}}(\mathbf{W}) + o\left(\mathcal{L}_n^{\mathrm{P}}(\mathbf{W})\right) \end{aligned} \tag{4}$$

après décomposition de Taylor. On voit ainsi que la perte PNP est une linéarisation locale de la perte spectrale et que l'erreur de linéarisation est asymptotiquement négligeable devant la perte P. Le premier avantage de cette linéarisation est que le cout de calcul du gradient $\boldsymbol{\nabla}\mathcal{L}_n^{\mathrm{PNP}}(\mathbf{W})$ est très inférieur à celui de $\boldsymbol{\nabla}\mathcal{L}_n^{\mathrm{sp}}(\mathbf{W})$, ce grâce au stockage en cache des matrices $\mathbf{M}_{\boldsymbol{n}}$ à travers plusieurs époques d'entrainement de $\boldsymbol{f}_{\mathbf{W}}$.

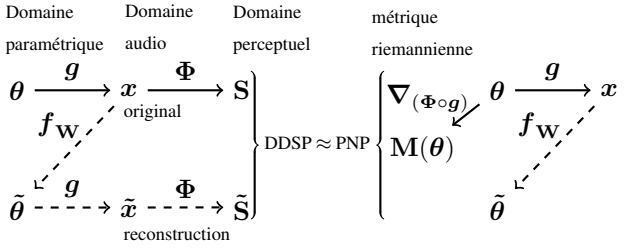

FIGURE 1 : Étant donné un synthétiseur $\boldsymbol{g}$ et un descripteur $\boldsymbol{\Phi}$, nous entrainons un réseau de neurones $\boldsymbol{f}_{\mathbf{W}}$ à minimiser la forme quadratique "perceptuelle–neuronale–physique" (PNP) associée à $(\boldsymbol{\Phi} \circ \boldsymbol{g})$. Ainsi, PNP approxime la perte spectrale employée dans DDSP sans besoin de rétropropager $\boldsymbol{\nabla}_{(\boldsymbol{\Phi}\circ\boldsymbol{g})}$ à chaque époque. Les opérateurs désignés par des flèches en traits pleins peuvent être précalculés, contrairement aux flèches en pointillés. Voir Section 2.

Un second avantage de la perte PNP est qu'il est facile d'y introduire un terme de régularisation statistique. En effet, si la matrice $\mathbf{M}_{\boldsymbol{n}}$ est de rang déficient, le problème inverse consistant à identifier $\boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x}_n)$ minimisant $\mathcal{L}_n^{\mathrm{PNP}}$ est mal posé. Face à cette instabilité numérique, on perturbe $\mathbf{M}_{\boldsymbol{n}}$ afin de la rendre de rend plein : dans le domaine des problèmes inverses, ceci revient à appliquer une régularisation de Tikhonov à une méthode de minimisation des moindres carrés. On choisit comme matrice de Tikhonov la moyenne uniforme $\overline{\mathbf{M}}$ de toutes les matrices $\mathbf{M}_{\boldsymbol{n}}$ déjà calculées, multipliée par un hyperparamètre λ indépendant de n. D'où une perte PNP régularisée

$$\mathcal{L}_n^{\mathrm{reg}}(\mathbf{W}) = \left\langle \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n} \middle| \mathbf{M}_{\boldsymbol{n}} + \lambda\overline{\mathbf{M}} \middle| \boldsymbol{f}_{\mathbf{W}}(\boldsymbol{x_n}) - \boldsymbol{\theta_n} \right\rangle. \tag{5}$$

On choisit $\lambda = 1$ dans la suite de cet article. Nous résumons le fonctionnement de notre méthode PNP en Figure 1.

3 Application à un arpège AM/FM

L'apprentissage par perte PNP est potentiellement applicable à n'importe quel problème "onde vers forme" tant que l'opérateur DDSP associé $(\boldsymbol{\Phi} \circ \boldsymbol{g})$ est une fonction différentiable de $\boldsymbol{\theta}$. Dans un article récent, nous avons démontré l'utilité de PNP pour un synthétiseur de percussions, fondé sur une EDP à cinq termes [5]. Dans cet article, nous élargissons notre étude au cas d'un synthétiseur qui ne produit pas des sons isolés mais articulés en arpège. Ainsi, on propose d'évaluer la capacité du réseau de neurones à restituer le rythme et les intervalles d'une mélodie ; ce, non en transcrivant chaque note mais de façon globale, telle que le propose la théorie de l'analyse de scène auditive *(auditory scene analysis)* en neurosciences [2].

Les trois paramètres de notre synthétiseur $\boldsymbol{g}$ sont la fréquence porteuse f_{c} en Hertz, le taux de modulation d'amplitude f_{m} en Hertz et le taux de modulation de fréquence γ en octaves par seconde. On définit l'arpège comme une séquence infinie de gazouillis *(chirps)* exponentiels $\boldsymbol{a_k}(t)\cos\boldsymbol{\varphi_k}(t)$ pour $n \in \mathbb{Z}$, fenêtrée par une gaussienne centrée réduite $\boldsymbol{\phi}$:

$$\boldsymbol{x} = \boldsymbol{g}(\boldsymbol{\theta}) : t \longmapsto \frac{1}{\gamma}\boldsymbol{\phi}(\gamma t) \sum_{n=-\infty}^{+\infty} \boldsymbol{a_k}(t)\cos\boldsymbol{\varphi_k}(t). \tag{6}$$

Chaque amplitude temporelle $\boldsymbol{a_k}$ est une demi-période de

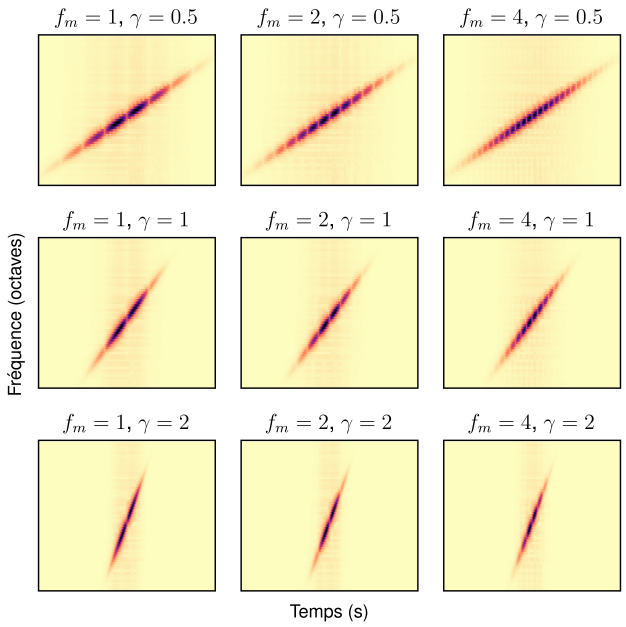

FIGURE 2 : Scalogramme en ondelettes de neuf arpèges AM/FM synthétisés par $\boldsymbol{g}$. La visualisation temps–fréquence représente une durée de 4 secondes et un intervalle de 2.5 octaves. De gauche à droite : variation du taux d'AM f_{m} en Hz. De haut en bas : variation du taux de FM γ en Hz. Sur ces exemples, le paramètre f_{c} est constant. Voir Section 3.

sinusoïde centrée autour de (k/f_{m}) et de durée $(1/f_{\mathrm{m}})$:

$$\boldsymbol{a_k}(t) = \sin(2\pi f_m t)\mathbb{1}\left(k \leq f_m t < k + \tfrac{1}{2}\right). \qquad (7)$$

Chaque phase $\boldsymbol{\varphi_k}$ est construite de sorte que la hauteur musicale instantanée est $\log \dot{\boldsymbol{\varphi}}_{\boldsymbol{k}}(k/f_m) = \log_2 2\pi f_{\mathrm{c}} + \gamma k/f_m$ et $\log_2(\ddot{\boldsymbol{\varphi}}_{\boldsymbol{k}}(t)/\dot{\boldsymbol{\varphi}}_{\boldsymbol{k}}(t)) = \gamma$ pour tout $t \in \mathbb{R}$. D'où :

$$\boldsymbol{\varphi_k}(t) = 2^{\gamma \frac{k}{f_{\mathrm{m}}}} \frac{2\pi f_{\mathrm{c}}}{\gamma \log 2} \exp\left[\gamma \log 2 \left(t - \tfrac{k}{f_{\mathrm{m}}}\right)\right]. \qquad (8)$$

La figure 2 donne neuf exemples d'arpèges AM/FM dans le domaine temps–échelle pour différentes valeurs des paramètres de synthèse. Dans ces exemples, une légère variation dans le taux de FM γ a tendance à désaligner les éléments de l'arpège, ce qui mène à une perte spectrale élevée et un gradient quasi nul. Ce phénomène d'instabilité de la perte spectrale basée sur le spectrogramme a déjà été constaté dans un article récent [9] dont le but était d'identifier $\boldsymbol{\theta}$ par descente de gradient sur $(\boldsymbol{\Phi} \circ \boldsymbol{g})$ à partir d'un paramètre aléatoire $\tilde{\boldsymbol{\theta}}$, sans recourir à un réseau de neurones. Ainsi, le scalogramme en ondelettes, et a fortiori la transformée de Fourier à fenêtre, est certes continu par rapport au paramètre $\boldsymbol{\theta}$, mais instable analytiquement.

La solution proposée dans l'article en question fut de remplacer le spectrogramme par un descripteur non linéaire de plus haut niveau d'abstraction, à savoir la diffusion temps–fréquence (JTFS pour *joint time–frequency scattering*). Cette représentation est particulièrement adaptée pour caractériser finement les modulations spectrotemporelles dans les signaux audio tout en étant plus stable que les représentations temps–fréquence pour ce type de signaux. De plus, la JTFS entretient des liens privilégiés avec la théorie des champs réceptifs spectrotemporels (STRF pour *spectrotemporal receptive fields*) en neurophysiologie de l'audition et permet de prédire la dissimilarité de timbre entre modes de jeux instrumentaux telle qu'elle est évaluée par un consensus de compositeurs et compositrices [6].

En accord avec des publications précédentes (notamment [6] et [7]), nous appliquons une transformation logarithmique aux coefficients JTFS d'ordres un et deux $\mathbf{S}\boldsymbol{x}$:

$$\boldsymbol{\Phi}(\boldsymbol{x}) = \log\left(1 + \frac{\mathbf{S}\boldsymbol{x}}{\varepsilon}\right), \qquad (9)$$

avec $\varepsilon = 10^{-3}$ constant. Quant à l'analyseur $\boldsymbol{f_{\mathbf{W}}}$, on choisit un réseau convolutif profond de type EfficientNet-B0. Cette architecture, relativement frugale en nombre de paramètres (4×10^6), a déjà donné des résultats probants dans divers problèmes de classification d'images et de sons. En entrée de ce réseau de neurones, nous calculons un scalogramme en ondelettes sur $J = 10$ octaves et $Q = 12$ filtres par octave, suivi d'une transformation logarithmique.

4 Résultats

Afin d'évaluer notre méthode, nous générons une base de données synthétique pour différentes valeurs de $\boldsymbol{\theta} = (f_0, f_{\mathrm{m}}, \gamma)$. Plus précisément, nous définissons une grille comprenant 30 pas selon chaque dimension, d'où $30^3 = 27 \cdot 10^3$ cellules tridimensionnelles en tout. On tire un point uniformément au hasard dans chacune des cellules afin de définir les points $\boldsymbol{\theta_n}$. Les $14^3 \approx 2.7 \cdot 10^3$ cellules situées au cœur de la grille constituent notre base validation et représentent environ 10% de l'ensemble des données. Quant aux cellules à la périphérie de la grille, on les répartit aléatoirement entre base d'entrainement et base de test, dans une proportion de huit pour un. Ainsi, nous avons environ 80% de données d'entrainement, 10% de données de validation, et 10% de données de test. Les bases d'entrainement et de test sont identiquement distribuées mais pas celle de validation : ceci afin d'évaluer la capacité de généralisation à des régions nouvelles de l'espace des paramètres et afin de détecter le surapprentissage pendant la SGD. Les valeurs extrêmes de la grille sont : 512–1024 Hz pour f_0, 4–16 Hz pour f_{m} et 0.5–4 Hz pour γ. Dans chaque dimension, les pas suivent une progression géométrique.

Notre analyseur $\boldsymbol{f_{\mathbf{W}}}$ est un réseau convolutif profond de type EfficientNet-B0, suivi par une unité linéaire entrebâillée *(LeakyReLU)*. Nous entrainons trois instances du même réseau pour l'estimation du paramètre θ, chacun avec une fonction de perte différente : P, PNP et PNP régularisée. Idéalement, nous souhaiterions entrainer une quatrième instance d'EfficientNet-B0 à l'aide d'une perte spectrale basée sur la distance JTFS ; mais cet entrainement n'est pas réalisable en pratique en raison de la complexité calculatoire relative à la rétropropagation du gradient pour cette fonction de perte. En effet, nous estimons que cette rétropropagation est environ 200 fois plus couteuse pour l'opérateur JTFS ($\boldsymbol{\Phi}$) que pour le réseau de neurones $\boldsymbol{f_{\mathbf{W}}}$ lui-même.

Nos résultats sont consignés en Table 1. Le temps d'entrainement durant 70 époques est d'une heure et demie pour les approches comparées dans cet article et est estimé à plus de 11 jours pour l'approche DDSP basée sur la perte spectrale JTFS. Nous évaluons les réseaux de neurones sur la base de test à l'aide de deux mesures objectives : la distance entre spectrogrammes multi-échelles (MSS pour *multi-scale spectrogram*)

TABLE 1 : Résultats d'analyse–synthèse d'arpège AM/FM sur la base de test. MSS : spectrogramme multi-échelle. JTFS : diffusion temps–fréquence jointe. La perte paramétrique ($\mathcal{L}_n^{\mathrm{P}}$) et la perte spectrale ($\mathcal{L}_n^{\mathrm{sp}}$) correspondent à l'état de l'art. La perte perceptuelle–neuronale–physique ($\mathcal{L}_n^{\mathrm{PNP}}$) et sa version régularisée ($\mathcal{L}_n^{\mathrm{reg}}$) sont nos contributions.

fonction de perte	distance MSS	distance JTFS	temps de calcul
$\mathcal{L}_n^{\mathrm{P}}$ [4]	1.22 ± 0.01	45.81 ± 0.3	1.5 h
$\mathcal{L}_n^{\mathrm{sp}}$ [3]	—	—	~ 280 h
$\mathcal{L}_n^{\mathrm{PNP}}$	0.74 ± 0.2	18.75 ± 6.9	1.5 h
$\mathcal{L}_n^{\mathrm{reg}}$	$\mathbf{0.60 \pm 0.1}$	$\mathbf{15.15 \pm 7.7}$	1.5 h

et la distance L^2 entre coefficients JTFS. Fréquemment utilisée comme mesure de distance audio, la MSS additionne les distances L^1 et $\log - L^1$ entre les spectres calculés à différentes échelles de temps [3]. JTFS s'en distingue en calculant, à partir d'un scalogramme de type CQT, des profils de modulation le long des axes de temps et de fréquence.

D'après ces métriques, nous observons que la perte PNP est toujours plus performante que la perte P. Cela est dû au fait que chaque dimension du paramètre θ se situe dans des plages radicalement différentes. Une perte P donne la priorité à l'optimisation de la précision de la dimension ayant la plus grande plage au détriment des autres : en l'occurrence, f_0 au détriment de f_{m} et γ. La perte PNP contourne ce déséquilibre en appliquant des poids qui reflètent l'importance de la fluctuation de chaque paramètre sur la perception auditive. La régularisation de la métrique riemannienne $\mathbf{M}$ (équation 5) améliore encore le résultat de la perte PNP. Ceci tend à confirmer notre hypothèse selon laquelle la matrice de Tikhonov $\bar{\mathbf{M}}$ atténue le caractère mal posé du problème inverse. Grâce au pré-calcul massivement parallèle des matrices $\mathbf{M}_{\boldsymbol{n}}$ décrivant le fibré tangent de la variété $\mathcal{M}$, notre méthode bénéficie de sa proximité avec la modélisation perceptuelle tout en restant suffisamment rapide à entrainer.

5 Conclusion

L'intérêt renouvelé pour les réseaux de neurones en analyse–synthèse de la musique suscite des problèmes de passage à l'échelle et de fidélité à la perception humaine. Dans cet article, nous avons montré que le formalisme du traitement du signal audionumérique (DDSP) appliqué à la diffusion temps–fréquence était apte à modéliser des structures AM/FM non stationnaires, tels que des arpèges. Nous avons présenté une nouvelle fonction de perte, appelée "perceptuelle–neuronale–physique" (PNP) et motivée par l'apprentissage de variété riemannienne. La perte PNP approxime localement la perte spectrale tout en accélérant le calcul de rétropropagation du gradient. À l'avenir, on cherchera à combiner ces connaissances "guidées par les données" avec un a priori "guidé par la physique" sur la distribution des paramètres de synthèse.

6 Remerciements

Nous remercions Brian McFee et Barbara Pascal pour leurs conseils.

Références

[1] Daniel ARFIB : DAFx98, DAFx99, DAFx00 ... (digital audio effects) un cycle de conférences européennes. *In Journées d'informatique musicale*, 2000.

[2] Alain de CHEVEIGNÉ : Analyse de scène auditive et parole. *In Journées d'Étude de la Parole*, 2000.

[3] Jesse ENGEL, Lamtharn (Hanoi) HANTRAKUL, Chenjie GU et Adam ROBERTS : DDSP : Differentiable Digital Signal Processing. *In Proceedings of the International Conference on Learning Representations (ICLR)*, 2020.

[4] Han HAN et Vincent LOSTANLEN : wav2shape : Hearing the shape of a drum machine. *In Forum Acusticum*, 2020.

[5] Han HAN, Vincent LOSTANLEN et Mathieu LAGRANGE : Perceptual-neural-physical sound matching. *arXiv preprint arXiv :2301.02886*, 2023.

[6] Vincent LOSTANLEN, Christian EL-HAJJ, Mathias ROSSIGNOL, Grégoire LAFAY, Joakim ANDÉN et Mathieu LAGRANGE : Time–frequency scattering accurately models auditory similarities between instrumental playing techniques. *EURASIP Journal on Audio, Speech, and Music Processing*, 2021(1):1–21, 2021.

[7] John MURADELI, Cyrus VAHIDI, Changhong WANG, Han HAN, Vincent LOSTANLEN, Mathieu LAGRANGE et George FAZEKAS : Differentiable Time-Frequency Scattering On GPU. *In Proceedings of the Digital Audio Effects Conference (DAFx)*, 2022.

[8] Jean-Claude RISSET : Le compositeur et ses machines. de la recherche musicale. *Esprit*, 3(99):59–76, 1985.

[9] Cyrus VAHIDI, Han HAN, Changhong WANG, Mathieu LAGRANGE, György FAZEKAS et Vincent LOSTANLEN : Mesostructures : Beyond spectrogram loss in differentiable time–frequency analysis. *arXiv preprint arXiv :2301.10183*, 2023.