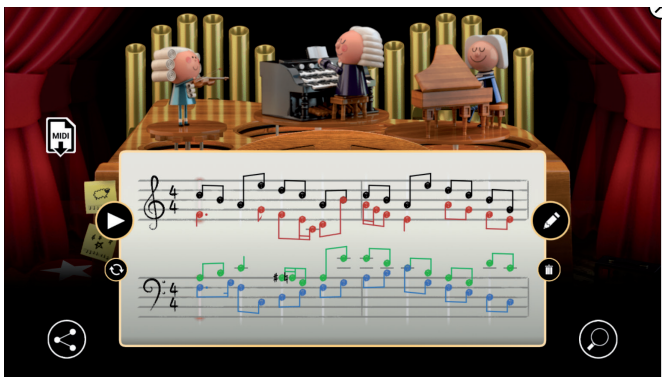
par Jean-Pierre Briot

Apprentissage profond et génération musicale

Si les algorithmes de création musicale ont été imaginés bien avant l'arrivée de l'informatique, les chaînes de Markov et, plus récemment, les réseaux profonds de neurones permettent de relever de nouveaux défis artistiques.

Jean-Pierre Briot est directeur de recherche CNRS au LIP6, laboratoire de recherche en informatique commun à Sorbonne Université et au CNRS.

Le « tsunami » actuel de l'apprentissage profond, retour hyper-vitaminé des réseaux de neurones artificiels, non seulement s'applique aux tâches traditionnelles de l'apprentissage machine statistique – la prédiction et la classification de chiffres manuscrits –, mais a d'ores et déjà conquis d'autres domaines, tels que la traduction (comme le service *Google Translate*), la reconnaissance (l'assistant *Siri* d'Apple) ou la synthèse vocale (l'assistant *Alexa* d'Amazon).



« Bach Doodle ». Exemple de génération contrapuntique de choral (mélodie soprano originelle en noir et celles générées en contrepoint en couleur).

Un nouveau domaine d'application est la génération de contenu créatif : texte, image, son, vidéo, musique…

Algorithmique et création musicale

Dans le domaine musical, en mars 2019, à l'occasion de l'anniversaire de la naissance de Johann Sebastian Bach, Google a présenté un Doodle (une variation sur son logo) créant un accompagnement par contrepoint associé à une mélodie définie interactivement par l'utilisateur, dans le style des chorals de Bach. L'architecture sous-jacente est un réseau de neurones profond.

Plusieurs jeunes entreprises, telles que AIVA (Artificial Intelligence Virtual Artist), Amper Music et JukeDeck (rachetée par la plateforme chinoise de partage de clips musicaux TikTok), se partagent déjà le marché tout juste naissant de la génération de musique pour des documentaires ou des publicités, voire de la pop (*cf.* la chanson *Break Free* de Taryn Southern en août 2017). Elles sont nombreuses à utiliser des architectures de réseaux profonds.

Pour mieux comprendre, il est utile de revenir au début de l'algorithmique musicale, que l'on peut faire remonter à la fin du XVIII[e] siècle. On l'attribue en effet (peut-être à tort) à Mozart, avec son jeu de dés musical (*Muzikalisches Wurfelspiel,* en 1787) dans lequel, en lançant à plusieurs reprises deux dés, on choisit chaque élément successif parmi un ensemble de segments mélodiques prédéfinis (et fixes).

L'un des tout premiers exemples de composition algorithmique à l'aide d'un ordinateur a été la création en 1957, par Lejaren Hiller et Leonard Isaacson, de *ILLIAC Suite* à l'université de l'Illinois aux États-Unis. Les deux compositeurs, ou plutôt « méta-compositeurs », sont à la fois scientifiques et musiciens. L'approche consiste en la combinaison de modèles de Markov*, pour la partie génération, et de règles (les contraintes) pour la sélection. D'autres modèles peuvent être utilisés, tels des grammaires génératives.

* Les *chaînes de Markov* sont des modèles stochastiques de transition pour lesquels la prédiction du prochain état ne dépend que de l'état présent et de la probabilité associée à chaque possible transition (voir *Tangente SUP* 69, 2013, ou *Markov : les chaînes de l'espoir* dans *Mathématiques et Médecine*, Bibliothèque Tangente 58, 2016).

Quand l'ordinateur apprend à composer

L'idée d'apprendre automatiquement à partir de nombreux exemples est à la base du changement de paradigme de l'approche actuelle. En plus d'une moindre difficulté pour définir un modèle, ce qui n'exempte pas néanmoins des ajustements en partie empiriques des caractéristiques de l'architecture et de l'apprentissage, les « hyper-paramètres », le processus est automatique et générique, puisque le style musical dépendra du corpus d'exemples musicaux choisis.

Un des pionniers en matière d'utilisation de réseaux de neurones artificiels pour générer de la musique a été Peter Todd en 1989, suivi par d'autres, du fait de progrès notables, entre autres en matière de réseaux récurrents (voir l'encadré), puis de l'avènement des réseaux profonds. Ce mouvement s'est à la fois accéléré et diversifié.

• *Étape 1 : le modèle*. Pour commencer, on se fixe un objectif. Ici, générer un accompagnement d'une mélodie. L'accompagnement peut, par exemple, se traduire sous la forme d'une séquence harmonique formée de suites d'accords, ou bien sous la forme de plusieurs mélodies formant un contrepoint, comme une polyphonie vocale. Ce cas est exactement celui du Bach Doodle. Le corpus choisi (et donc le style) est celui des chorals de Bach, un (sinon « le ») maître en matière de contrepoint. Dans le modèle des chorals, trois voix (alto, ténor et basse) sont créées et associées

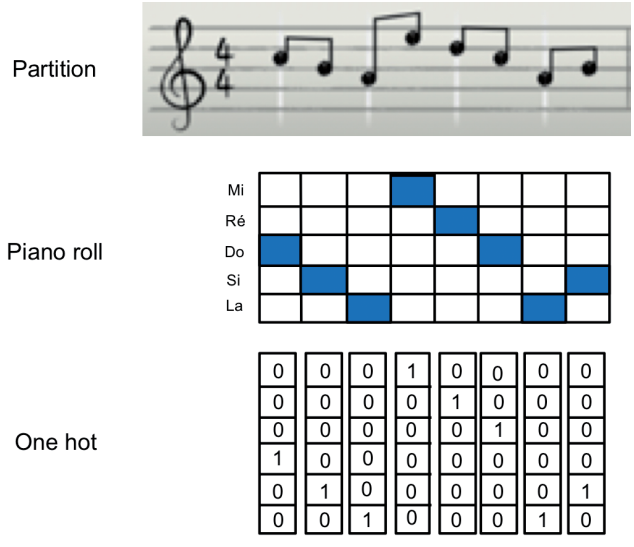
à la voix (soprano) originelle. Le problème d'apprentissage consiste donc à apprendre, en fonction d'une entrée (la mélodie soprano), la sortie (les trois mélodies associées).

• *Étape 2 : la représentation des données*. Il faut trouver un mode de représentation des notes d'une mélodie en vue de les transformer en variables d'entrée du réseau de neurones (et de manière duale pour les variables de sortie). L'approche la plus courante est le format de papier à musique (« *piano roll* »), inventé pour les pianos mécaniques et les orgues de Barbarie, et dont on considère ici la version numérique. Le remplissage d'une case, correspondant à une note donnée et à un temps donné, indique que la note associée sera jouée à ce moment. L'encodage naturel du *piano roll* est de faire correspondre à chaque *segment temporel* successif – la durée minimale des notes, pour la mélodie soprano, une croche – un vecteur ayant comme taille l'intervalle entre la note la plus basse et la note la plus haute (la *tessiture*, donc le nombre de lignes du *piano roll*) avec une valeur égale à 1 pour l'élément correspondant à la note actuelle et une valeur nulle (0) pour tous les autres. Cette forme d'encodage, inventée au départ pour l'électronique, se nomme ainsi *one-hot*.

Le cas du Bach Doodle est assez simplifié : pour la mélodie soprano, il n'y a pas d'altération (par exemple, pas de note *la#* entre *la* et *si*). Par ailleurs, il faut pouvoir représenter une note tenue, qui doit être distinguée d'une note répétée. Une solution simple est de représenter la note tenue comme un élément additionnel au vecteur de notes. Le silence peut être implicite et correspondre à une absence de note et en conséquence un vecteur *zero one-hot*, sans aucun 1.

• *Étape 3 : l'architecture*. La dernière étape est enfin de juxtaposer bout à bout les vecteurs de notes correspondant aux segments temporels élémentaires successifs et les faire correspondre aux différentes variables d'entrée, également appelés *nœuds de la couche d'entrée* du réseau. La couche de sortie représente, quant à elle, la concaténation des trois mélodies d'accompagnement. L'architecture présente un certain nombre de couches cachées, selon la profondeur du réseau (une seule dans le cas simplifié de la figure). Le décodage des valeurs produites procède à l'inverse de l'encodage de l'entrée. Pour chacune des trois voix, chaque vecteur successif représente la note produite. En pratique, dans l'interprétation déterministe la plus directe, il suffit de choisir la note dont la valeur de l'élément correspondant est la plus grande (et ainsi la plus probable).

Représentations successives de la première moitié de la mélodie soprano : partition, *piano roll*, vecteurs *one-hot*.



Lors de la phase d'apprentissage, le réseau est entraîné avec un grand nombre d'exemples (plus de trois cent cinquante pièces du répertoire de chorals de Bach), en présentant pour chaque exemple la mélodie soprano en entrée et les trois mélodies d'accompagnement en référence pour la sortie. L'algorithme d'apprentissage, à base de descente de gradient (ou plus sophistiqué) et de rétro-propagation, conduit à ajuster de manière incrémentale les poids des connexions entre les neurones, de manière à réduire l'*erreur de classification* (prédiction de chaque note pour chaque segment temporel et pour chaque voix).

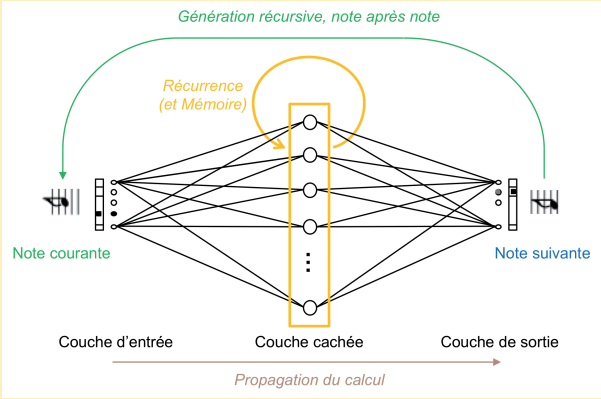
• *Étape 4 : la production*. Une fois le réseau entraîné, on peut débuter la phase de production (génération) : présenter des mélodies arbitraires, et générer le contrepoint associé, selon le principe d'une telle architecture, de type *feedforward* (à propagation avant). C'est exactement sur ce principe que fonctionnent le Doodle Bach de Google et des systèmes plus sophistiqués, qui produisent des chorals dans le style de Bach difficiles à distinguer d'un original.

Adapter le réseau à la stratégie de création

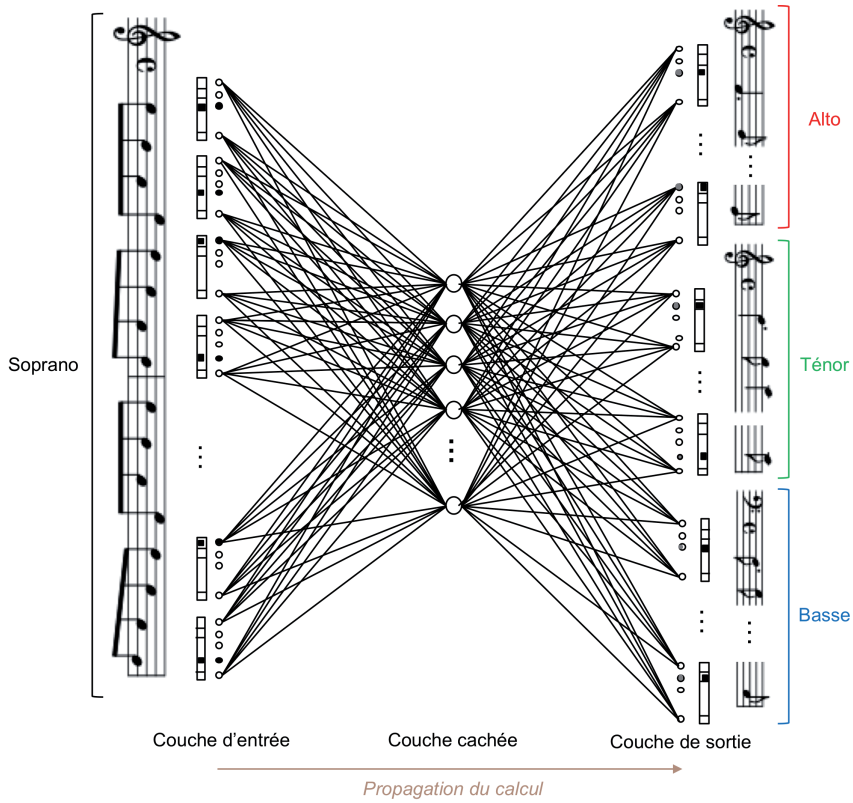
Une limitation importante de cette architecture est que la taille des mélodies générées est fixée par l'architecture. De manière à générer une mélodie de taille arbitraire, on peut utiliser une autre stratégie, en entraînant un réseau de neurones plus simplifié sur l'apprentissage de la correspondance entre une note et la suivante. On exerce alors le réseau sur un ensemble d'exemples comportant en entrée une note et en sortie la note suivante, selon une combinatoire de segments extraits d'un grand nombre de partitions (des mélodies utilisées par Bach, ou de type celtique…). La génération procède cette fois de manière itérative, et plus pré-

Les réseaux récurrents

Un *réseau de neurones récurrent* ajoute une mémoire au niveau de chaque neurone d'une couche cachée ainsi que des connexions récurrentes (réentrantes, également pondérées et ainsi ajustées comme toutes les connexions lors de la phase d'apprentissage). Cela permet de tenir compte du calcul des éléments précédents. La version moderne, appelée LSTM (pour *Long short-term memory*, « mémoire à court et long terme »), apprend à réguler l'accès à la mémoire et ainsi à minimiser les risques d'erreurs accumulées lors des calculs numériques des gradients par rétro-propagation temporelle pendant la phase d'apprentissage (le problème initial de la disparition ou de l'explosion du gradient est ainsi résolu).
Les réseaux récurrents sont utilisés de manière routinière pour des prédictions de séries (temporelles ou non) : prédiction du climat, traduction (voir en page 58), et donc ainsi en musique, où l'on peut considérer une mélodie comme une série temporelle de notes.



Architecture de réseau récurrent et génération récursive note par note.

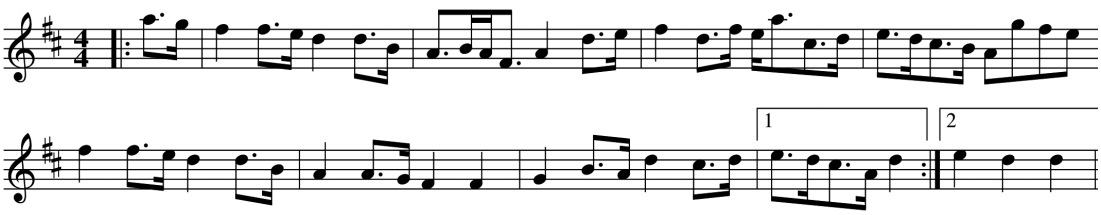


Architecture d'un réseau de neurones *feedforward* à une seule couche cachée générant un contrepoint.

cisément récursive, en présentant une note de départ, en générant la note suivante, qui servira de nouvelle entrée au réseau, et ainsi de suite, pour générer une suite de notes (mélodie) correspondant au style appris.

Cela fonctionne avec une architecture à propagation avant. Intuitivement, la transitivité va faire que les corrélations entre l'ensemble des notes successives découlent des corrélations deux à deux successives. En pratique, et même si le résultat est convenable, on s'aperçoit vite que le réseau n'a pas la capacité de capturer les relations à long terme et, par voie de conséquence, la mélodie générée manque de cohérence. L'utilisation d'une architecture récurrente (voir encadré) offre cette capacité de capture de relations à plus long terme.

Les exemples présentés portent sur des approches symboliques de la musique : portée, notes, accords… Il existe cependant une tendance croissante à les élargir à des représentations audio, spectre harmonique ou même signal brut de type forme d'onde. Les sys-

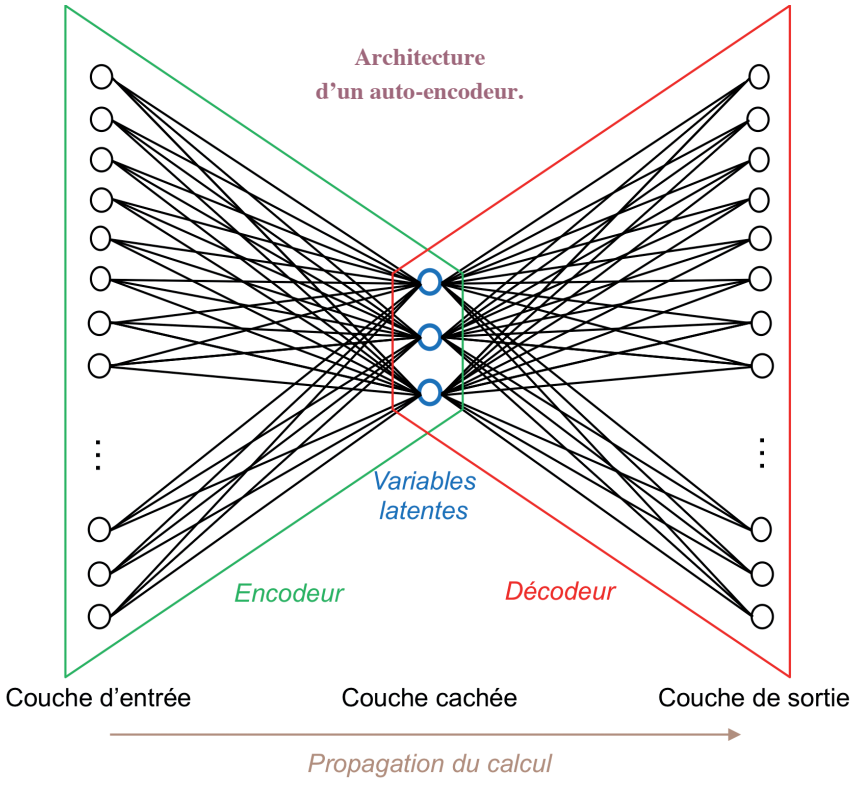


***The Mal's Copporim*, mélodie celtique générée automatiquement par l'architecture de réseau récurrent *folk-rnn*.**

L'auto-encodeur

Un *auto-encodeur* est un réseau de neurones à propagation avant pour lequel la couche de sortie est identique à la couche d'entrée. Lors de la phase d'apprentissage, on présente pour chaque exemple la même donnée en entrée et en sortie. L'objectif est d'entraîner l'auto-encodeur successivement à encoder l'information dans la couche cachée, puis la décoder en vue de reconstruire « au plus près » l'information initiale. La couche cachée présente en général un nombre de neurones très inférieur au nombre de neurones de la couche d'entrée (et de sortie) et peut intégrer des contraintes additionnelles : parcimonie (minimisation du nombre de neurones actifs en même temps, ce qui va entraîner leur spécialisation en fonction des caractéristiques discriminantes entre les exemples), suivi d'une distribution de loi normale (cas des auto-encodeurs variationnels)…

L'intérêt d'un auto-encodeur n'est pas dans la génération de la donnée de sortie mais dans l'effet induit d'apprendre une représentation condensée et représentative d'un ensemble de données d'un espace à un grand nombre de dimensions, appelé *variété* (ou *manifold*) en mathématiques, vers un espace à peu de dimensions appelé *espace des variables latentes*. L'auto-encodeur variationnel a l'avantage supplémentaire d'assurer une bonne correspondance entre une « petite » variation au niveau de l'espace latent et une « petite » variation au niveau de l'espace des données. Une analogie est l'approximation de surfaces terrestres à trois dimensions dans des cartes à deux dimensions.



tèmes de reconnaissance et de génération de voix des assistants de Google ou d'Amazon sont des exemples de telles applications.
Au bout du compte, quelle que soit la représentation initiale (notes, signal, pixels, lettres…), tout sera encodé sous la forme d'immenses vecteurs numériques pour alimenter les architectures de réseaux de neurones, ce qui assure cette généricité des architectures, indépendamment des représentations et des corpus (ensembles d'exemples d'entraînement).

De nouveaux défis

Les deux types d'architectures vus plus haut donnent de bons résultats quand le corpus est suffisamment important et cohérent et la structuration configurée par des hyper-paramètres bien choisis : bon nombre de couches, d'itérations… Même si les choix fins de représentation (notes tenues, silences…) peuvent influer grandement sur la qualité des résultats, les musiques produites correspondent étonnamment bien au style appris et peuvent être confondues avec des originaux par la plupart des auditeurs. Cependant, au-delà de l'aspect « test de Turing musical réussi », l'intérêt artistique reste limité. À quoi bon recréer des musiques dans un style déjà connu ? Un certain nombre de questions restent ainsi ouvertes.
• Structure : la musique générée, surtout si elle est longue, manque en général de direction et de structure.
• Contrôle : un musicien va en général vouloir spécifier certaines contraintes, sur la durée des notes, sur l'harmonie à suivre…
• Originalité : le musicien souhaite un équilibre entre une conformité à un style mais également que le contenu généré soit capable de le surprendre (et de lui plaire).
• Interactivité : la génération reste autonome et mécanique, avec pas ou peu d'interactivité avec le musicien, qui en général souhaite pouvoir retravailler certaines portions sans avoir à chaque fois à regénérer tout l'ensemble.

Ces défis sont des questions de recherche en cours et diverses stratégies et architectures plus complexes sont proposées et évaluées par les chercheurs du domaine. Quelques exemples plus sophistiqués ont déjà été développés.
• Les architectures d'*auto-encodeurs variationnels* permettent d'apprendre les caractéristiques des variations entre les différents exemples d'un corpus musical. La couche cachée peut être réduite à deux neurones. Il est ensuite possible d'explorer l'espace latent à deux dimensions correspondant (les deux dimensions de variabilité peuvent, par exemple, correspondre à la tessiture et à la variance de la durée des notes, et vont dépendre du corpus d'exemples) et de générer différentes instances du style appris. Il suffit de choisir des valeurs correspondant aux variables latentes, par interpolation, extrapolation, combinaison…, et de les propager en avant dans la partie « décodeur » de l'auto-encodeur pour générer une mélodie correspondant au style appris, tout en contrôlant les caractéristiques de variabilité. Il est possible d'emboîter un réseau récurrent dans la composante « encodeur » ainsi que dans la composante « décodeur » de l'auto-encodeur (voir encadré). Cela permet ainsi de combiner les avantages de l'exploration des caractéristiques discriminantes et du réseau récurrent de taille variable. Ce type d'architec-

ture, appelée RNN Encoder-Decoder, est d'ailleurs la base des systèmes de traduction actuels.

• Les architectures de *réseaux antagonistes génératifs* (en anglais, *generative adversarial networks*) sont un autre exemple, qui consiste en deux réseaux de neurones : le *générateur*, chargé de générer des éléments « les plus proches possibles » d'une base d'éléments (des photos, des tableaux, des musiques de référence…), et le *discriminateur*, chargé de déterminer si l'élément qu'on lui présente en entrée appartient à la base ou bien est un élément synthétique produit par le générateur. Les deux réseaux sont entraînés simultanément, jusqu'à ce que le générateur arrive à « tromper » suffisamment le discriminateur (lui-même devenu de plus en plus performant). En octobre 2018, la société de vente aux enchères Christie's a adjugé pour plus de 400 000 dollars un tableau intitulé *Portrait d'Edmond De Belamy* ainsi généré.

• Il existe également comme alternatives récentes aux architectures récurrentes des architectures de *réseaux convolutifs* (voir l'article qui suit), ainsi que des architectures basées sur un mécanisme d'attention multiple, portant sur différents éléments d'une séquence d'entrée, et contrôlé par l'apprentissage.

• Par ailleurs, des architectures de transfert de style (voir l'encadré « Les réseaux de neurones artistes » en page 24) peuvent être utilisées comme points de départ pour imposer des caractéristiques (structure, tonalité, rythme…) à des musiques existantes ou, encore mieux, en cours de génération.

• Enfin, il existe des reformulations et des combinaisons avec des modèles d'apprentissage *par renforcement*, qui permettent d'inclure des objectifs et le retour (*feedback*) de l'utilisateur.

Portrait d'Edmond De Belamy.

Ainsi, le domaine de la création de contenu artistique selon un style appris est un domaine en plein développement, aux niveaux scientifique, technique et économique. Les contenus peuvent être musicaux, mais également textuels (poésie…) ou visuels (images, tableaux, vidéos). Les réussites techniques sont d'ores et déjà là, tout en laissant encore nombre de questions ouvertes.
Au-delà des enjeux plus techniques se posent les questions des usages et de la collaboration entre humain (musicien) et machine, pour aller non pas vers des générateurs mécaniques de musique, mais vers des assistants proactifs permettant aux musiciens d'explorer et de produire dans une collaboration homme–machine des œuvres qu'il leur serait difficile d'imaginer sans cet apport.

J.-P. B.

RÉFÉRENCES :

• *Deep Learning Techniques for Music Generation*. Jean-Pierre Briot, Gaëtan Hadjeres et François-David Pachet, Springer Verlag, 2019.
• *Apprentissage artificiel – Deep learning, concepts et algorithmes*. Antoine Cornuéjols, Laurent Miclet et Vincent Barra, Eyrolles, 2018.
• *Mathématiques et musique*. Bibliothèque Tangente 11, 2010.
• Dossier « Mathématiques et musique contemporaine ». *Tangente* 188, 2019.
• *Hello World*. Divers artistes, avec l'aide de l'environnement FlowComposer, Flow Record, 2018.