# 36-Séquençage de débit audio

## introduction

Ce tutoriel examine comment générer des événements temporisés dans Max en utilisant des signaux audio MSP comme source de synchronisation. Lorsque nous concevons des systèmes basés sur le temps dans Max, nous sommes habitués à utiliser les objets *metro*, *clocker*, *delay* et autres objets Max exécutés dans le programmateur Max pour générer des événements automatiquement. Cependant, quand on travaille avec MSP, il est souvent bénéfique de pouvoir synchroniser les événements directement à partir d'un signal MSP. Cette technique permet non seulement un contrôle plus précis de la synchronisation des événements (ce qu'on appelle le séquencement précis de l'échantillon), mais elle nous permet également d'utiliser un certain nombre d'objets MSP qui permettent la synchronisation de plusieurs flux d'événements basés sur un seul signal d'horloge maître. Notre premier tutoriel examine quelques-uns de ces objets et à leur fonctionnement.

## Fournir un signal d'horloge de synchronisation

• Jetez un coup d'œil au patcheur du tutoriel. Activez l'audio dans le patch en cliquant sur l'objet *ezdac* ~ en haut du patch. Les objets *scope* ~, ainsi que l'objet *multislider* au milieu du patch, devraient tous s'animer. Dans la zone du patcheur étiquetée **1**, cliquez plusieurs fois sur l'objet *button* à une vitesse constante. Remarquez les signaux changent et comment la boîte de *nombre* intitulée BPM s'ajuste.

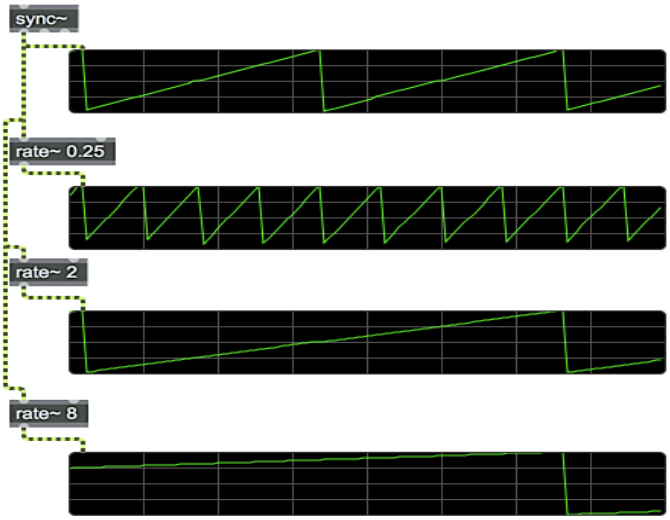
L'objet MSP qui commande le comportement dans notre patcheur est appelé *sync* ~ et émet le même type de signal qu'un objet *phasor* ~, c'est-à-dire une rampe de signal linéaire allant de **0.** à **1.** à une vitesse constante. Contrairement à l'objet *phasor* ~, cependant, l'objet comprend sa vitesse non pas en Hertz, mais en battements par minute (BPM), une définition standard du tempo musical utilisé dans la plupart des applications de séquençage audio. En cliquant sur l'objet *button* situé en haut du patcheur, on envoie des messages **bang** à l'objet *sync~,* qui simule un système de tempo à clic utilisé dans de nombreux séquenceurs. En cliquant sur le bouton à un rythme régulier, l'objet *sync~* ajuste sa vitesse pour envoyer une rampe unique pour chaque battement (intervalle) entre les messages **bang**. Le BPM de nos messages **bang** est calculé et envoyé à la deuxième sortie de l'objet *sync* ~, que nous utilisons ensuite pour définir une boîte de *nombre* que nous pouvons modifier pour définir le BPM manuellement.

• Dans la boîte de *nombre* en haut du patcheur, tapez **60**. et appuyez sur Retour. L'objet *sync* ~ fonctionne maintenant à 60 BPM, émettant une rampe de signal une fois par seconde. Dans la zone de patcheur intitulée **2** (colorée en vert), montez le curseur de *gain* ~ gauche et écoutez le résultat.

La sortie de l'objet *sync* ~ est une rampe de signal, tout comme la sortie de l'objet *phasor* ~. En tant que tel, nous pouvons l'utiliser directement comme un générateur d'enveloppe, pour contrôler le volume d'un signal audio allant au *dac* ~ (dans ce cas, un générateur de bruit *rand* ~). Bien que cela soit certainement utile dans certaines applications, le principal avantage de l'utilisation d'un signal pour fournir une synchronisation d'événements réside dans notre capacité à échelonner le signal dans le temps, et à générer ensuite des événements à partir de celui-ci.

## Plusieurs vitesses à la fois

• Remarquez que, dans le patcheur du didacticiel, chacun des objets *scope* ~ semble dessiner des rampes à différentes vitesses:




Différentes échelles de temps de rampe générées par l'utilisation des objets *sync* ~ et *rate* ~.

L'objet *rate* ~ nous permet de mettre à l'échelle temporelle les signaux de rampe générés par les objets *sync* ~ et *phasor* ~. L'argument de l'objet (ou une nouvelle valeur fournie dans l'entrée de droite) définit le facteur d'échelle, par ex. un objet *rate*~ avec un facteur de **2,0** générera une rampe qui prendra ***deux fois plus de temps*** à se répéter que le signal d'entrée. Un *rate*~ de **0,5**, par comparaison, ira ***deux fois plus vite*** que l'entrée. En utilisant ce système, nous pouvons avoir plusieurs zones de logique de signal dans notre patcheur MSP fonctionnant à différents multiples du signal de synchronisation original tout en restant parfaitement en temps.

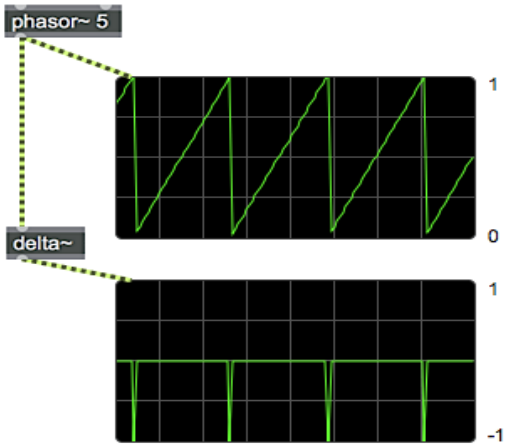
## Générer des événements Max à partir d'un signal de synchronisation

• Dans la zone **2** du patcheur, baissez le curseur gauche de *gain* ~ et montez celui de droite. Vous devriez entendre un «clic» métronomique régulier, généré par l'objet *noise* ~ avec une enveloppe déclenchée par l'objet *button* dans cette partie du patcheur.

Il y a deux choses à noter ici. La première est que le «clic» ne se produit pas à la vitesse que nous avons fixée dans l'objet *sync* ~, mais à quatre fois cette vitesse. C'est ce que fait l'objet *rate* ~ dans la zone **2** du patcheur, qui est réglée sur une valeur de **0,25**. Cela signifie que notre timing musical effectif pour un «beat» dans notre patcheur est divisé en quatre dans cette partie de la chaîne de signal MSP, ce qui nous permet de déclencher des événements plus rapides.

Le «clic» que nous entendons est généré par l'envoi d'un message **bang** (de l'objet *button*) pour générer une enveloppe de l'objet *line* ~, avec un fondu rapide de *noise* ~ pour produire un tic-tac aigu. Cette logique nous est familière grâce aux tutoriels précédents; la grande question est ici de savoir comment l'objet *button* est déclenché?

L'objet MSP *delta* ~ prend un signal en entrée et émet un signal en sortie, dont les nombres indiquent de combien le signal entrant a ***changé*** d'échantillon en échantillon:




Une rampe *phasor* ~ (en haut) et son signal *delta* ~ (en bas).

Dans le cas d'un signal en rampe, l'objet *delta*~ rapporte une valeur positive plus ou moins constante au fur et à mesure que le signal monte; cependant, lorsque la rampe revient à **0**., l'objet *delta* ~ rapporte une valeur négative soudaine, et élevée qui représente la chute soudaine de la rampe. Cette valeur négative momentanée du signal peut être détectée en utilisant un opérateur logique sur le signal pour tester s'il devient négatif. L'objet < ~ du patcheur fait cela, en sortant une valeur de signal de **1** lorsque le signal d'entrée correspond à la condition spécifiée par l'argument de l'objet, et un **0** sinon. Dans le cas présent, la sortie de < ~ sera un **1** chaque fois que le signal d'entrée est ***inférieur à* 0**, c'est-à-dire uniquement lorsque la rampe sortant de l'objet *rate* ~ se réinitialise, ce qui fait que l'objet *delta* ~ se réinitialise, ce qui fait que l'objet *delta* ~ sort une valeur négative.

L'objet *edge* ~ est un objet MSP qui prend des signaux d'entrée et les utilise pour déclencher des messages Max **bang**. Pour ce faire, il examine les états de transition de **0** à **1** (sortie gauche) ou de **1** à **0** (sortie droite). Lorsque l'objet *rate* ~ réinitialise sa rampe, la sortie gauche de l'objet *edge* ~ génère un **bang** en réponse à la transition logique **0** à **1** faux à vrai) sortant de l'objet <~. Ce **bang** peut déclencher n'importe quel événement Max, comme s'il était tiré d'un objet *metro* ou déclenché en cliquant manuellement sur un *button*.

• Baissez le curseur de *gain* ~ au bas de la zone de patch **2** et examinez la zone de patch **3**.

## Séquençage à débit audio en utilisant des intervalles de temps fixes

• Dans la zone du patcheur **3**, cliquez et dessinez dans les deux objets *multislider* intitulés "Pitch" et "Amplitude". Augmentez le curseur de *gain* ~ au bas de la logique de la zone de patcheur **3**. Vous devriez entendre une onde carrée répétant un motif mélodique qui correspond à la forme que vous avez dessinée dans les objets *multislider*. Essayez de dessiner différents motifs pour la séquence et d'écouter les résultats.

La mélodie répétitive spécifiée par nos objets *multislider* est déclenchée par un objet de séquençage précis à l'échantillon appelé *techno* ~. L'objet *techno* ~ conserve un certain nombre ***de pas*** de points de données en fonction de son paramètre de longueur (défini dans notre patcheur avec le message **length 16**). Chaque pas peut être associé à une hauteur et à une amplitude, qu'il émet ensuite sous forme de signaux à partir de ses sorties. Ces signaux contiennent toutes les informations nécessaires pour piloter directement les oscillateurs et les amplificateurs MSP (dans ce cas, les objets *rect*~ et * ~).

Les séquences sont chargées dans l'objet *techno* ~ en envoyant des messages **pitch** et **amplitude** avec comme arguments le numéro et la valeur de pas. Les objets *listfunnel* de notre patcheur créent les messages appropriés à cet fin, en prenant les listes de **16** membres des objets *multislider* et en les traduisant en seize messages individuels numérotés. De la même manière, d'autres attributs du processus de séquence peuvent être définis, soit pour chaque étape individuellement, soit pour la séquence entière.

• Au bas de la zone de patch **3**, réglez la boîte de *nombre* intitulée «Attack» sur **1.0**. Remarquez comment les notes passent de courtes à longues (ou de staccato à legato). Réglez la boîte de *nombre* intitulée 'Decay' sur **0 .**. Remarquez comment l'enveloppe des notes change pour produire un type différent de séparation de notes, où l'onde carrée s'estompe et se termine brusquement. Réglez les valeurs «Attack» et «Decay» sur **1.** et modifiez la boîte de *nombre* intitulée «Curve» en **0.1.** Écoutez comment les hauteurs glissent l'une sur l'autre.

Les messages d'attaque, de déclin et de courbe, tout comme les messages de hauteur et d'amplitude, nous permettent de régler avec précision le comportement du séquenceur interne de l'objet *techno* ~. Les valeurs **d'attaque** contrôlent la trajectoire de l'enveloppe qui sort de la sortie centrale de l'objet *techno* ~ lorsqu'elle monte au début d'une note. Avec une valeur de **0**, les notes s'enclenchent instantanément; avec une valeur de **1,0** les notes de la séquence prennent prennent tout le temps nécessaire pour s'estomper à partir de leur valeur précédente. Les valeurs de **decay** fonctionnent sur un principe similaire, mais en ce qui concerne la fin (ou le stade de «libération») de l'enveloppe de la note. Une décroissance de **1,0** permet aux notes de se fondre ensemble; une valeur de **0** garantit que chaque note tombe dans le silence entre les étapes du séquenceur. Ces deux valeurs interagissent pour créer des formes d'enveloppe dynamiques. Les messages de **courbe** contrôlent la quantité de ***portamento*** (ou glissement) appliquée au signal contrôlant la hauteur de la séquence. Les valeurs de courbe de **0** provoquent une «rupture» de note en note dans la séquence. En augmentant la valeur de la courbe, une partie de plus importante du battement sera occupée par une rampe de hauteur interpolée, ce qui fera glisser le son en fréquence.

Comme pour la hauteur et l'amplitude, les paramètres d'attaque, de décroissance et de courbe l'objet *techno* ~ sont fournis étape par étape. Remarquez comment nous utilisons l'objet *uzi* pour transmettre la même valeur seize fois afin de définir chaque étape de notre séquence sur les mêmes valeurs.

• Jouez avec les possibilités de séquençage avec *techno* ~. Une fois que vous avez compris le fonctionnement des commandes, baissez le curseur de *gain* ~ pour la zone de patch **3** et regardez à droite, dans la zone de patch **4**.

## Séquençage à débit audio de données arbitraires

• Montez le curseur de *gain* ~ pour la zone de patch **4**. À l'aide du clavier de votre ordinateur, tapez sur la rangée centrale des touches alphabétiques (A, S, D, F, G, H, J, K, L,;). Vous devriez entendre différentes notes jouées par un simple oscillateur en dents de scie. Si vous le souhaitez, double-cliquez sur l'objet *coll* nommé **keys** pour voir la correspondance entre les valeurs ASCII et les numéros de notes MIDI.

• Cliquez sur la touche "r" de votre clavier et jouez quelques notes sur la rangée du milieu. Remarquez que l'objet toggle attaché au message **record $ 1** devient actif. Cliquez une deuxième fois sur la touche "r", puis sur la touche "p". Une fois la séquence répétée, vous devriez entendre vos notes jouées.

L'objet *seq* ~ fonctionne comme un séquenceur audio d'événements numériques arbitraires. Le signal qu'il reçoit en entrée sert d'horloge de synchronisation pour l'objet, ce qui nous permet de l'alimenter avec une rampe de signaux provenant d'un objet *rate* ~ réglé sur huit mesures. Lorsque *seq* ~ reçoit un message **record 1**, il écoute les messages Max qui arrivent dans l'objet et les ***horodate*** en fonction du signal d'entrée. Lorsque *seq* ~ est en mode de lecture (via un message **play 1**), il utilise le signal d'entrée pour rechercher des événements en fonction de leur horodatage, et les restitue dans le bon ordre. Notez que, contrairement à l'objet *techno* ~, l'objet *seq* ~ produit des événements Max, et non des valeurs de signal. Il peut donc être utilisé pour stocker des données arbitraires.

> Bien qu'il n'entre pas dans le cadre de ce tutoriel, l'objet *seq* ~ peut stocker plusieurs séquences simultanément, peut superposer plusieurs passages de données, et peut lire et écrire des fichiers de données d'une manière similaire à l'objet *coll*. De plus, parce qu'il stocke des messages arbitraires et horodatés, il peut être utilisé pour séquencer des commandes Max autres que MIDI avec une précision d'échantillon, par exemple des listes de commandes de dessin vers un objet *lcd*, ou des messages vers une série d'objets Jitter, etc. Enfin, le signal fournissant l'horodatage pour l'objet *seq* ~ peut être à une échelle arbitraire; il n'a pas besoin d'être dans la gamme de **0** à **1**, et vous n' vez pas à lui envoyer une rampe linéaire. Cela élargit encore la flexibilité de l'objet.

• Augmentez le curseur de *gain* ~ inférieur dans la zone de patch **1** pour vous donner un clic. Appuyez sur la touche 'c' pour effacer la séquence stockée dans l'objet *seq* ~. Appuyez sur la touche "r" pour enregistrer une nouvelle séquence. Lorsque la boucle de séquence est bouclée, désactivez l'enregistrement et jouez votre séquence. Remarquez que l'objet *seq* ~ peut traiter des données avec n'importe quel intervalle de temps, contrairement à l'objet *techno*~, qui fonctionne par étapes discrètes.

## Résumé

MSP dispose d'une variété d'outils pour le séquençage du rythme audio. Les objets *sync* ~ et *rate* ~ peuvent être utilisés pour créer des rampes de synchronisation simples et des intervalles réguliers en utilisant des valeurs de battements par minute pour déterminer le tempo. Les objets d*elta* ~ et *edge* ~ sont utiles pour générer des événements Max à partir d'un signal de rampe MSP. L'objet *techno* ~ permet un séquençage précis à l'échantillon près d'un nombre fini d'étapes pour une seule paire oscillateur / amplificateur audio; pour un séquençage plus flexible, l'objet *seq* ~ permet l'horodatage de la fréquence audio d'événements Max arbitraires qui peuvent être enregistrés, édités et rejoués comme des séquences d'instructions.