À propos de l'auteur

Laurent Berger

Laurent Berger est maître de conférences à l'IUT du Mans. Il enseigne depuis 1992 le traitement du signal aux étudiants du DUT de Mesures Physiques en utilisant Scilab. Ces principaux travaux de recherche portent sur la simulation numérique de systèmes magnétiques, traitement d'images appliqué à la physique et en instrumentation météorologique infrarouge.

Préface

par Claude Gomez

Directeur général de Scilab Enterprises

Bienvenue dans Scilab : De la théorie à la pratique.

Le logiciel de calcul et de simulation numérique Scilab est aujourd'hui la référence des logiciels libres du domaine. Il comprend plus de 1 600 fonctions mathématiques et graphiques et inclut Xcos pour la simulation des systèmes dynamiques hybrides. C'est dire que, malgré l'aide en ligne, il est souvent difficile pour l'utilisateur d'appréhender toutes ses fonctionnalités sans véritable support. Et comme les domaines concernés couvrent ceux de toutes les mathématiques appliquées, calcul matriciel, simulation, traitement du signal, statistiques, commande, etc., il faudrait un livre pour chacun. C'est ce que vous propose Scilab : De la théorie à la pratique , aborder progressivement tous ces domaines pour vous fournir une vue d'ensemble complète du logiciel Scilab. Le monde logiciel est marqué par les mises à jour régulières et l'obsolescence rapide des supports traditionnels papiers, et l'originalité de D-BookeR est de vous permettre de consulter un livre toujours à jour à partir d'un site web.

Je vous souhaite une bonne lecture et une bonne pratique de Scilab !

Introduction

Ce manuel fait partie intégrante de l'ouvrage Scilab : De la théorie à la pratique, constitué de plusieurs modules autonomes. Son objectif est de présenter les principaux outils de traitement de signal disponibles dans Scilab. Il suppose que vous êtes déjà familier du logiciel et que vous en connaissez les manipulations de base. Si tel n'est pas le cas, nous vous invitons à commencer par le premier module portant sur les fondamentaux.

Scilab intègre un grand nombre de fonctions dédiées au traitement du signal dans sa version de base, que vous pouvez compléter pour répondre à des besoins par des modules ATOMS ou en développant vos propres outils. Pour notre part, nous recourrons ponctuellement au module Portaudio pour l'acquisition de son, à la Time frequency Toolbox pour disposer des méthodes Lagunas et Capon en analyse spectrale et de la transformée de Choi-Williams, ainsi qu'à la Scilab Wavelet Toolbox (ou SWT) pour la transformée en ondelettes.

Attention > Les modules complémentaires disponibles sur ATOMS ne sont pas tous maintenus par les développeurs de Scilab, et leur suivi n'est pas nécessairement garanti. Ce dépôt communautaire est en effet ouvert à quiconque souhaite partager les outils qu'il a développés. Dans notre cas, seul Portaudio est maintenu par Scilab Enterprises, les deux autres sont néanmoins régulièrement mis à jour.

1. Ce que ce livre est et n'est pas

Ce livre ne constitue pas un cours de traitement du signal. Néanmoins les programmes proposés sont présentés dans l'ordre d'un cours de traitement du signal. Le lecteur intéressé par l'aspect mathématique pourra se reporter à la bibliographie proposée en annexe.

Volontairement nous nous sommes restreints à l'analyse du signal sonore pour avoir des exemples démonstratifs. Ces exemples sont écrits dans le langage de script de Scilab.

Note > Le traitement du signal bidimensionnel comme l'image fera l'objet d'un autre manuel.

Les différents outils sont systématiquement présentés dans le contexte d'une application, si bien que ces pages illustrent également, d'une manière plus générale, comment tirer parti des fonctionnalités graphiques et de programmation de Scilab pour mettre en place des outils d'analyse et de simulation.

Seules les fonctionnalités spécifiques au traitement du signal sont détaillées dans leur emploi. Toutefois, nous nous sommes toujours efforcés de commenter les autres fonctions que nous utilisons afin que vous puissiez clairement comprendre comment les programmes sont construits et les décliner ensuite selon vos propres besoins.

2. Code source des exemples

Le code source des exemples est téléchargeable sur la page de présentation du livre sur le site des éditions D-BookeR, à l'onglet Compléments.

Pour reproduire les exemples, ouvrez les fichiers de scripts directement depuis Scilab ou copiez-en le code dans l'éditeur SciNotes, et exécutez-le. La commande `clear` en tête de listing permet d'effacer les variables créées auparavant par l'utilisateur. L'exécution du programme n'utilise ainsi que les variables créées par lui ou les variables globales de Scilab.

Le code source du module Anaspec présenté en étude de cas au dernier chapitre se trouve également sur Github.

3. Réglage de la largeur de l'écran

Vous trouverez de nombreux exemples de code, formatés dans une police à chasse fixe. Afin d'éviter des retours à la ligne inopportuns à l'intérieur d'une ligne de code, la longueur maximale des lignes de code a été fixée à 65 caractères, une valeur suffisamment basse pour être affichée sur la plupart des supports, tout en étant suffisante pour que le code puisse être correctement formaté.

Toutefois, il est possible que sur votre support la largeur maximale affichable soit inférieure à la limite fixée. Le paragraphe test ci-dessous permet de vérifier votre affichage. Il doit tenir sur deux lignes exactement :

```
00000000001111111112222222222333333333344444444455555555556666 6
012345678901234567890123456789012345678901234567890123456789012345678901234
```

Si ce n'est pas le cas, regardez si vous pouvez agrandir la taille de la fenêtre, diminuer la taille des marges ou diminuer la taille de la police d'affichage. Sur un téléphone portable, placez-le plutôt en mode paysage. Si vous n'y arrivez pas, ne vous inquiétez pas pour autant, la plupart des lignes de code sont inférieures à 65 caractères.

4. Affichage des équations MathML

Les équations de cet ouvrage ont été codées avec le standard MathML. Si vous lisez ce livre en ligne, assurez-vous de disposer d'un navigateur compatible. Vous pouvez vérifier son support du MathML avec cette page de test. Pour notre part, nous vous recommandons l'emploi de Firefox.

Dans un souci d'accessibilité, la version EPUB par défaut ne contient pas de MathML, les équations étant converties en images. Toutefois, si votre reader supporte le MathML, vous disposez aussi d'une version avec MathML afin de bénéficier d'un rendu optimal.

Note > Reportez-vous à la page de support du site des éditions D-BookeR pour plus d'informations.

Analyse du signal

Dans ce premier chapitre, nous allons effectuer avec Scilab des manipulations de base du traitement du signal : fenêtrage, échantillonnage, quantification, transformations de Fourier et application du théorème de Shannon.

Pour commencer, parce qu'il est difficile de parler de numérisation sans partir d'un signal analogique, nous vous montrerons comment acquérir un signal dans Scilab. Cette fonctionnalité n'est pas prévue nativement par le logiciel et nécessite l'installation du module externe portaudio. Pour mettre en œuvre cet exemple, vous devez également disposer d'un microphone. Si ce n'est pas le cas, un autre exemple permet de simuler la numérisation d'un signal à partir de son expression mathématique définie dans une fonction du langage Scilab et du vecteur temps construit à partir de la fréquence d'échantillonnage.

Nous vous présenterons également un programme de synthèse de son qui vise avant tout à aider les débutants à bien comprendre ce qu'est un signal numérique.

Nous nous intéresserons ensuite à la fonction Scilab `fft`, qui permet d'effectuer la transformée de Fourier discrète du signal. Les représentations graphiques du signal et de son spectre sont tracées en utilisant les fonctions graphiques et l'interpréteur Latex du logiciel. Les résultats obtenus sont comparés aux séries de Fourier et transformés de Fourier estimées en utilisant la fonction Scilab d'intégration et certaines fonctions graphiques 3D.

Enfin, nos deux derniers exemples se fonderont sur le bon usage de la numérisation ou en vocabulaire de traitement du signal *le théorème de Shannon*, où l'on verra notamment comment retrouver la valeur d'un signal analogique à partir du signal numérique.

Encadré : Sur la numérisation d'un signal

Un signal continu est une fonction mathématique dépendant d'une variable prenant une valeur. Ce signal est dit *continu* car la variable est un nombre réel.

Cette notion peut être évidemment élargie à la notion de fonction vectorielle dépendant de plusieurs variables, mais dans la plupart des exemples suivants la fonction dépendra du temps et prendra une valeur dans l'ensemble des réels.

L'objectif de la numérisation du signal est de construire une approximation du signal continu (analogique) en un signal numérique, qui puisse être traitée par un ordinateur. Cette numérisation se décompose en trois opérations, l'échantillonnage du signal évoluant en fonction du temps, le fenêtrage temporel du signal et la quantification du signal.

- L'*échantillonnage* consiste à ne retenir que quelques valeurs du signal continu sur l'échelle du temps, généralement à intervalles réguliers.
- Le *fenêtrage* est la définition de la zone temporelle d'observation du signal.
- La *quantification* est l'opération transformant une valeur continue en une valeur discrète.

Ces trois opérations sont faites par un convertisseur analogique-numérique (CAN ou en anglais DAC) intégré dans une carte d'acquisition.

Le téléphone portable est un exemple de dispositif faisant usage de la numérisation. La voix (captée par un microphone) doit être numérisée pour être transmise à une autre personne. Le fenêtrage du signal est défini à partir du moment où la communication est établie jusqu'à la fin de la communication. L'échantillonnage est effectué en ne prenant les valeurs du signal que toutes les $1/8000$ s, définissant ainsi la période d'échantillonnage. On utilisera le plus souvent la *fréquence d'échantillonnage*, qui est ici de 8000 Hz. La quantification, quant à elle, est faite par le convertisseur analogique-numérique, pouvant mesurer $8192=2^{13}$ valeurs possibles entre -1 et 1 volt.

Une fois le signal numérisé, les traitements numériques peuvent être effectués soit pour mesurer des caractéristiques du signal, soit pour restituer le signal traité de manière analogique. Dans le cas de l'exemple du téléphone portable, le signal est restitué sur le haut-parleur et donc devient de nouveau analogique.

L'étape de numérisation du signal, échantillonnage et conversion analogique-numérique peut être simulée : le signal continu est construit à l'aide de fonctions mathématiques, et nous créons une fonction dans Scilab qui jouera le rôle de convertisseur analogique-numérique. Le traitement numérique du signal est alors fait à partir de cette simulation.

La plupart des exemples que nous vous présenterons ici utiliseront ce type de simulation numérique. Dans les autres cas, le signal numérisé sera fourni sous forme de fichier son de type `.wav` ou bien de fichier texte.

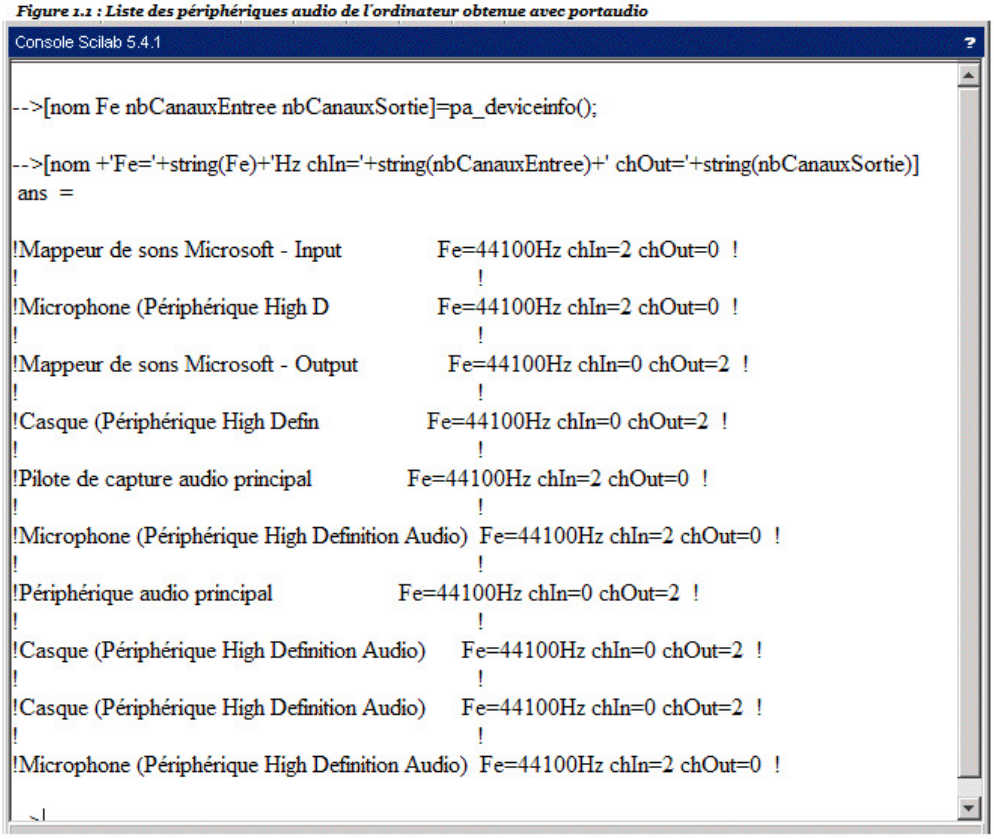
1. Acquérir, écouter et sauvegarder un son

Attention > Dans cette section, nous recourrons à quelques fonctions issues du module portaudio. Ce module, fondé sur la librairie éponyme, a été développé pour dialoguer avec la carte son. Il n'est disponible à ce jour que pour Windows[1]. Pour l'installer, lancez le gestionnaire de modules ATOMS depuis le menu Applications, et dans la liste des modules proposés, sélectionnez Instruments Control/Portaudio. N'oubliez pas de redémarrer votre application !

Avant toute chose, nous allons commencer par lister les périphériques audio de l'ordinateur. Pour cela, nous utilisons la fonction `pa_deviceinfo` du module Portaudio. Vous pouvez déterminer le nom des périphériques, la fréquence d'échantillonnage ou de restitution maximale, ainsi que le nombre de canaux en entrée ou en sortie en saisissant la commande `[nom Fe nbCanauxEntree nbCanauxSortie]=pa_deviceinfo()` dans la console (voir Figure 1.1).

```
[nom Fe nbCanauxEntree nbCanauxSortie]=pa_deviceinfo();
[nom +'Fe='+string(Fe)+'Hz chIn='+string(nbCanauxEntree)+' chOut='+string(nbCanauxSortie)]
```

Figure 1.1 : Liste des périphériques audio de l'ordinateur obtenue avec portaudio



Pour afficher le résultat de `pa_deviceinfo` sur une ligne, utilisez la fonction Scilab `string`, qui convertit un nombre en chaîne de caractères, et l'opérateur `+`, qui concatène deux chaînes de caractères.

Encadré :

Un son enregistré avec un seul microphone est un son monophonique. Un son enregistré par deux microphones est stéréophonique.

L'acquisition est possible si l'un des périphériques de la liste obtenue par `pa_deviceinfo` a un nombre de canaux en entrée non nul. Dans ce cas, l'acquisition du signal se fait par la commande `pa_recordwav`, où le premier argument est le nombre d'échantillons à prélever sur l'ensemble des canaux, le deuxième la fréquence d'échantillonnage et le troisième le nombre de canaux.

Par exemple, pour effectuer l'acquisition d'un son en stéréophonie pendant une durée D égale à 5 s (largeur de la fenêtre d'observation) avec une fréquence d'échantillonnage de 22050 Hz, vous saisirez la commande suivante dans la console :

```
D=5;Fe=22050;N=D*Fe;xSon=pa_recordwav(2*N,Fe,2);
```

Attention > Les paramètres de la fonction `record_wav` ne sont pas conformes à la documentation.