EISTI - DÉPARTEMENT MATHÉMATIQUES

ANALYSE NUMÉRIQUE - LABO N° 1

15 mars 2010

ANALYSE DES ERREURS

INTRODUCTION

La quantification est le procédé qui permet de représenter numériquement des quantités physiques. Par exemple la temperature affichée par un thermomètre numérique ou l'heure affichée par une montre à quartz. Nous pouvons assimiler ce procédé à l'approximation numérique, à la valeur la plus proche, d'un nombre réel effectuée par un ordinateur.

La figure 1 présente le fonctionnement d'un type particulier de quantificateur, appelé *quantificateur uniforme*. q est l'*unité de quantification*, c'est-à-dire c'est la différence entre deux valeurs quantifiées successives. La sortie

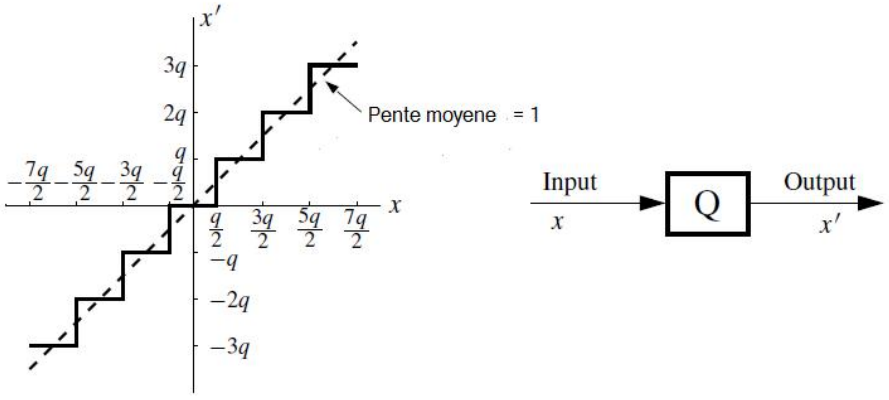

FIGURE 1 – Principe de la quantification

du quantificateur x' diffère de celle de l'entrée x. Ainsi, on a

$$v = x' - x$$

Dans le cadre du traitement du signal v est considéré comme un *bruit de quantification*.

Comme opérateur mathématique, le quantificateur est une fonction non linéaire qui, associée à un opérateur d'échantillonnage et appliquée à une fonction continue, fournit une fonction continue en escalier (voir figure 2)

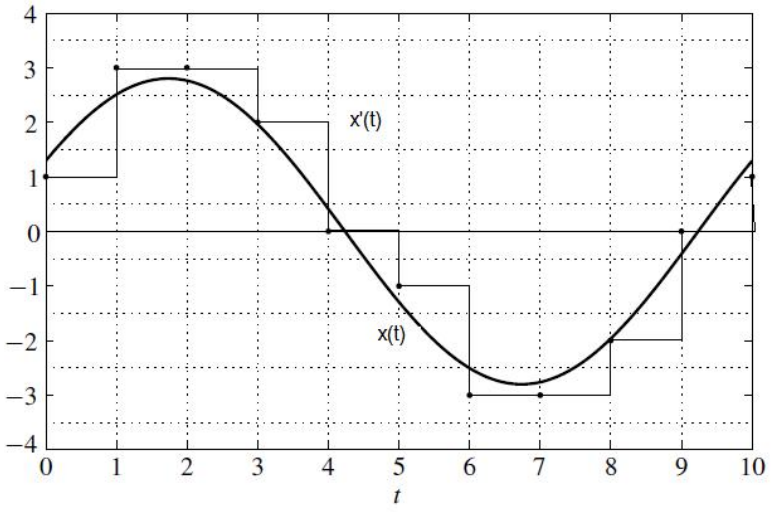

FIGURE 2 – Échantillonnage et quantification d'une fonction continue $x(t)$

Le but de ce labo est d'examiner la qualité de la quantification en fonction du nombre de bits utilisés pour le stockage de la mantisse et de l'exposant.

Travail à faire

PREMIÈRE PARTIE

1. Récupérer du site du cours `http ://sifoci.eisti.fr` les deux programmes Scilab `ieee-754.sci` et `bin2decsci`. Le deuxième réalise la conversion d'un nombre binaire stocké selon le standard IEEE-754, en décimal. Le premier est le programmes principal. Ce programme a comme données la mantisse (en tenant compte du bit caché) et l'exposant biaisé du nombre décimal `465.463` et il appelle la fonction `bin2dec` pour réaliser la conversion de ce nombe en décimal à partir de la représentation selon le standard IEEE-754.

2. Écrire la fonction `fonction[s, m, e] = dec2bin(x, p, q)` qui effectue la conversion du nombre réel `xx` en nombre binaire selon le standard IEEE-754. On a
   - `s` : signe (= 0 : positif ou nul, =1 : négatif) ;
   - `m` : vecteur-ligne ($1 \times p$) contenant la mantisse ;
   - `e` : vecteur-ligne ($1 \times q$) contenant l'exposant ;
   - `p` : nombre de bits de la mantisse (`0 < p =< 52`);
   - `q` : nombre de bits de l'exposant (`0 < q =< 11`).
3. Vérifier, en utilisant les six valeurs $x = \pm 465.463; x = \pm 1.463$ et $x = \pm 0.463$, du bon fonctionnement de votre programme.

DEUXIÈME PARTIE

1. Créer et visualiser la fonction sin à l'aide des commandes

```
t = 0.2;                    //  t = p\'eriode d'\'echantillonnage
x = [0:0.2:4*\%pi]';        //  x = points d'\'echantillonnage
y = sin(x);                 //  Fonction sin exhantillonn\'{e}e
plot(y);                    //  Visualisation de la fonction
a=gca();                    //  Instructions pour tracer les axes.
a.x_location = "origin";
a.y_location = "origin";
```

   Le vecteur y contient
2. Faire la quantification du signal stocké dans `y`, c'est-à-dire pour chaque valeur dans `y`, donner sa représentation selon le standard IEEE-754 et ensuite convertir cette représentation en valeur décimale. Pour le standard IEEE-754, on prendra $p = 23, q = 8$.
3. Tracer la fonction quantifiée et échantillonnée. (Vous devez obtenir une figure analogue à la figure 2).
4. Répéter la même opération avec pour le standard IEEE-754 les valeurs $p = 37, q = 10$.
5. Répéter les étapes 1 à 4 en prenant comme période d'échantillonnage $t = 0.5$.

TROISIÈME PARTIE

1. Comparer les résultats obtenus. (On pourrait s'aider pour la comparaison en utilisant un ou plusieurs tableaux.)

2. Écrire un rapport, en LaTeX, dans lequel figurera
   (a) le cahier d'analyse et de programmation du programme `dec2bin`.
   (b) les résultats des tests effectués avec les valeurs fournies.
   (c) les graphiques des fonctions sinus échantillonées et non échantillonnées.
   (d) Analyse détaillée des résultats obtenus du point de vue de leur qualité et recommandations.
3. Avec les programmes et le rapport, vous créerez un fichier compacté que vous posterez en utilisant le bouton approprié qui est sur le site du cours.

Délai pour l'envoi des rapports : Mardi 23 mars avant 23h59.

ANNEXE

Petit guide pour l'écriture du programme `dec2bin`

1. Il faut séparer la partie entière de la partie décimale du nombre réel : $x = V.D$, où V partie entière et D partie décimale.
   Notons que dans la suite on utilise la valeur absolue de$x$.
2. Écrire une fonction qui fait la conversion de l'entier D en binaire. L'algorithme se trouve sur la page 9 du poly et on le répète ici.
   Notons par $(v_{i-1}; v_i, r_{i-1})$ le résultat de la division de v_{i-1} par 10, c'est-à-dire $v_{i-1} = v_i \times 2 + r_{i-1}$.
   Posons $v_0 = v$ et considérons la suite de divisions

   $$(v_0; v_1, r_0),\ (v_1; v_2, r_1 a,\ \ldots,\ (v_{p-1}; v_p, r_{p-1})$$

   avec $r_{p-1} < 2$.
   L'entier D s'écrit en binaire : $v_p r_p r_{p-1} \ldots r_1 r_0$.
3. Écrire une fonction qui fait la conversion du fractionnaire D en binaire. L'algorithme se trouve sur la page 9 du poly et on le répète ici.
   Notons par $[f_i; d_{i+1}, f_{i+1}$ le résultat de la multiplication de $0.f_i$ par 2, c'st-à-dire $d_{i+1}.f_{i+1} = 0.f_i \times 2$.
   Posons $D = f_0$ et considérons aussi la suite de multiplications

   $$[f_0; d_1, f_1],\ [f_1; d_2, f_2],\ \ldots,\ [f_{q-1}; d_q, f_q]\ \ldots$$

   Le nombre $0.D$ s'écrit en binaire $0.d_1 d_2 \ldots d_q \ldots$.

4. En tenant compte du bit caché, calculer l'exposant biaisé en représentation IEEE-754. Le biais est donné par la formule $B = 2^{q-1} - 1$ (p. 20 du poly). Pour ce faire, il faut d'abord évaluer en décimal la valeur de l'exposant biaisé et ensuite utiliser la fonction de conversion d'entier en binaire pour obtenir cette même valeur en binaire. Il faut ici tenir compte du fait que l'exposant, sous forme binaire, doit avoir obligatoirement q bits.
5. Construire la mantisse en tenant compte du bit caché.
6. Calculer le bit du signe.
7. Retourner au programme appelant le triplet `[s, m, e]` où `s` est le bit du signe, `m` est le tableau où est stocké la mantisse et `e` est le tableau où est stocké l'exposant.