1 *Travail préparatoire seance 2: codage de filtres iir en virgule fixe*

AVERTISSEMENT: le codage des filtres *iir* est beaucoup plus complexe que le codage des filtres *fir*. Ce travail préparatoire obligatoire est un cours, qui a pour but de vous présenter les problèmes et méthodes liés au codage des filtres *iir*.

Pour être bien certain que vous le lisiez attentivement, et fassiez l'effort de comprendre le contenu, ***la partie 4 correspond aux exercices qui doivent être remis à l'enseignant au début de la séance (une copie manuscrite par personne, pas de traitement de texte)***. De plus l'enseignant se réserve le droit de vérifier que les étudiants ont effectivement rédigés eux-mêmes leur copie, sans se contenter de recopier bêtement sur quelqu'un d'autre. Dans le cas contraire, la note sera revue en conséquence.

Les pré-requis pour comprendre la programmation des filtres ***iir*** *sont*

- le codage des additions multiplications en nombres entiers
- le cours de systèmes discrets (transformée en z, bilinéaire)
- un minimum de probabilités (optionnel, mais aide à la compréhension)

2 Mise en évidence des problèmes sur un exemple simple (régulateur à avance de phase)

2.1 détermination de la fonction de transfert en z

2.1.1 *cahier des charges*

L'exemple traité est un correcteur numérique à avance de phase, de fonction de transfert $G(z)$ *, répondant au cahier des charges suivant :*

à la pulsation réelle : $\omega_u = 10\,rad/s$, la réponse fréquentielle $G\left(z = e^{i.\omega_u.T_e}\right)$ doit vérifier les 2 conditions suivantes :

gain du régulateur : $\left|G\left(z = e^{i.\omega_u.T_e}\right)\right| = G_u = 10$

argument du régulateur : $arg\left(G\left(z = e^{i.\omega_u.T_e}\right)\right) = \phi_u = 50°$

La fréquence d'échantillonnage est : $fe = 50\,hz$

la période d'échantillonnage est donc : $te = 0.02\text{s}$

2.1.2 traduction dans le plan W (transformation bilinéaire)

- la pseudo-pulsation v_u associée à la pulsation réelle ω_u vérifie :

$$v_u = \tan\left(\frac{\omega_u . te}{2}\right) \approx 0.1$$

- l'expression du correcteur à avance de phase dans le plan *W*, apportant le maximum d'avance de phase à la pseudo-pulstion $v = v_u$, est donnée par :

$$G_w(w) = c_0 . \frac{(1 + \frac{w}{v_u / a})}{(1 + \frac{w}{a . v_u})}$$

de plus pour un tel régulateur, on sait que

$$\begin{cases} \left|G_w(j.v_u)\right| = \left|C_0 . a\right| \{ \text{ doit être égal à } G_u \} \\ arg\left(G_w(j.v_u)\right) = 2.\text{atan}(a) - 90° \{ \text{doit être égal à } \phi_u \} \end{cases}$$

on en déduit la valeur des différents paramètres

$$\begin{cases} a = \tan\left(\frac{\phi_u + 90°}{2}\right) \approx 2.75 \\ C_0 = \frac{G_u}{a} \approx 3.64 \end{cases},$$

et donc finalement $G_w(w) = 3{,}64 . \frac{(1 + 27{,}5 . w)}{(1 + 3{,}64 . w)}$

2.1.3 retour dans le plan z, et vérification

On en déduit ensuite $G(z)$ par : $G(z) = G_w(w)_{\text{évalué en w}=\frac{z-1}{z+1}}$

soit encore $G(z) = 3{,}64 . \frac{(1 + 27{,}5 . \frac{z-1}{z+1})}{(1 + 3{,}64 . \frac{z-1}{z+1})} \approx \frac{22{,}35\, z - 20{,}79}{z - 0{,}569}$

noter que, par convention, on a normalisé à *1* le coefficient de plus haut degré en *z* du dénominateur

vérification : avec les arrondis effectués, on obtient

$\begin{cases} G(z=1) \approx 3{,}62 \\ G(z=-1) \approx 27{,}5 \end{cases}$ alors qu'en W : $\begin{cases} G_w(w=0) = 3{,}64 \\ G(w \to \infty) = 27.5 \end{cases}$

il y a donc une bonne correspondance entre l'expression de $G(z)$ et l'expression de $G_w(w)$, en dépit du fait que l'on ait arrondi les coefficients..

2.1.4 *expression en fonction de l'opérateur retard* $q = z^{-1}$

on préfère souvent pour l'implémentation écrire directement la fonction de transfert en fonction de l'opérateur retard $q = z^{-1}$, <u>en normalisant à 1 le coefficient de plus bas degré du dénominateur</u>, ce qui donne :

$$G(z) = \frac{22{,}35\, z - 20{,}79}{z - 0{,}569} . \frac{z^{-1}}{z^{-1}} = \frac{22{,}35 - 20{,}79\, z^{-1}}{1 - 0{,}569\, z^{-1}} = \frac{22{,}35 - 20{,}79\, q}{1 - 0{,}569\, q}$$

2.2 les quatre formes directes de réalisation de la fonction de transfert

notons $G(z) = \frac{b_0 + b_1 z^{-1}}{1 + a_1 z^{-1}} = \frac{b_0 + b_1 q}{1 + a_1 q}$

Si on se limite à représenter graphiquement les opérations simples suivantes
multiplication d'une variable par un coefficient b_i ou $-a_i$
additions
application de l'opérateur retard $q = z^{-1}$

il existe alors 4 schémas élémentaires de 'réalisation' de $G(z)$, connus sous le nom de formes directes.

2.2.1 forme directe 1, ou ***df1***

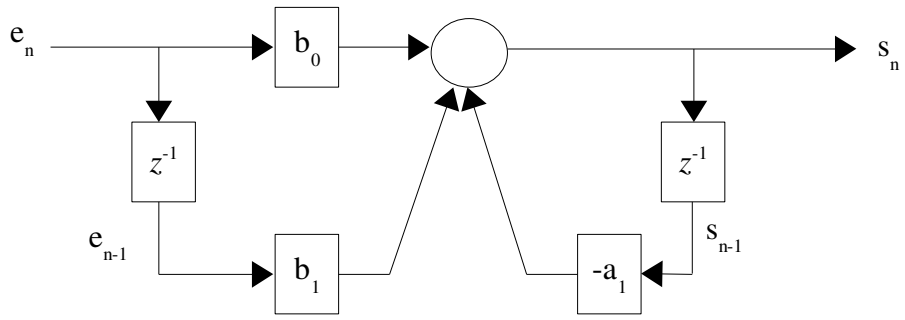

figure 1: réalisation sous forme directe *1 d'une cellule d'ordre 1*

2.2.2 *forme directe 2, ou* ***df2***

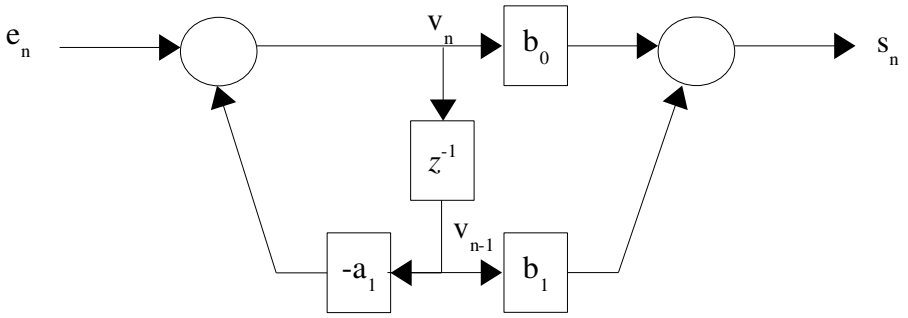

figure 2: réalisation sous forme directe 2 d'une cellule d'ordre 1

2.2.3 *forme directe 1 transposée,ou* ***df1t*** *(représentée à l'ordre 2)*

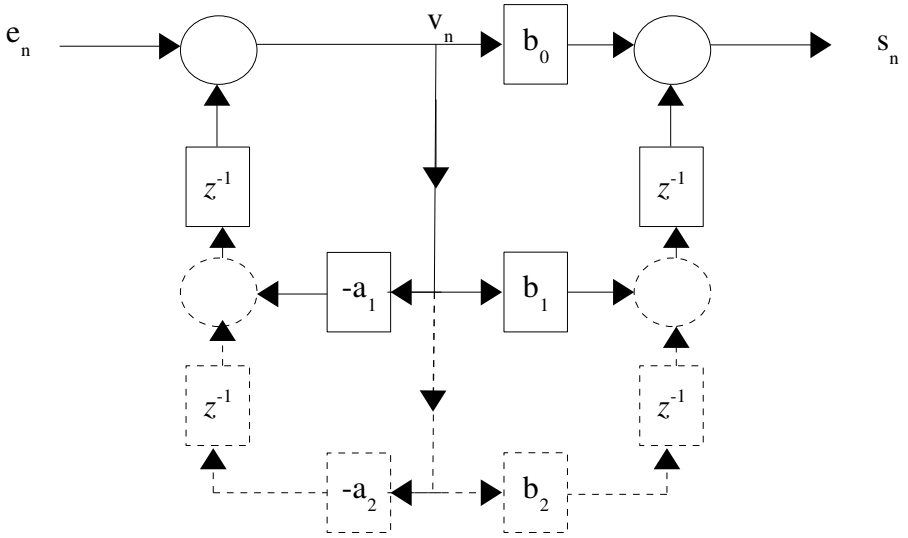

figure 3: réalisation sous forme directe 1 transposée d'une cellule d'ordre 1 ou 2

2.2.4 *forme directe 2 transposée, ou* ***df2t***

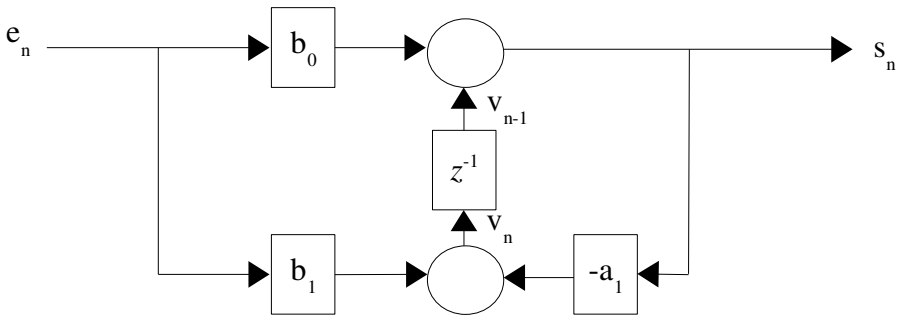

figure 4: réalisation sous forme directe 2 transposée d'une cellule d'ordre 1

2.3 analyse en vue de la programmation en nombres entiers

Quelle que soit la forme de programmation employée, on doit, avant la programmation en nombres entiers, procéder à l'analyse du schéma de principe que l'on veut réaliser. Cette analyse comprend

1-la détection des variables internes xi_n *du schéma {ce sont les variables à l'entrée des multiplications par des coefficients, qui doivent rester codables sur N bits }*

2-la prise en compte des bruits de quantification de signal { additionneurs de bruit en sortie des multiplications par les coefficients }

3-l'insertion intelligente d'un facteur d'échelle de signal $\lambda = 2^L$ *, dont le rôle est de minimiser l'effet des bruits de quantification sur la sortie, tout en veillant à ce que les variables internes* xi_n *restent codables sur l'échelle à N bits.*

on se limitera ici à présenter l'analyse détaillée pour la forme df2t

2.3.1 bruits de quantification de signal, et facteur d'échelle

1-Sans se soucier pour le moment de la façon dont il sera implémenté, on place un facteur d'échelle de signal $\lambda = 2^L$ en entrée de la cellule, et on divise la sortie par λ

2- on fait également apparaître, en sortie des multiplications par les coefficients $\{b_0, b_1, -a_1\}$, les bruits de quantification 'de signal' correspondants : $\{b_{b0}, b_{b1}, b_{a1}\}$

on obtient alors le schéma de pré-analyse figure 5

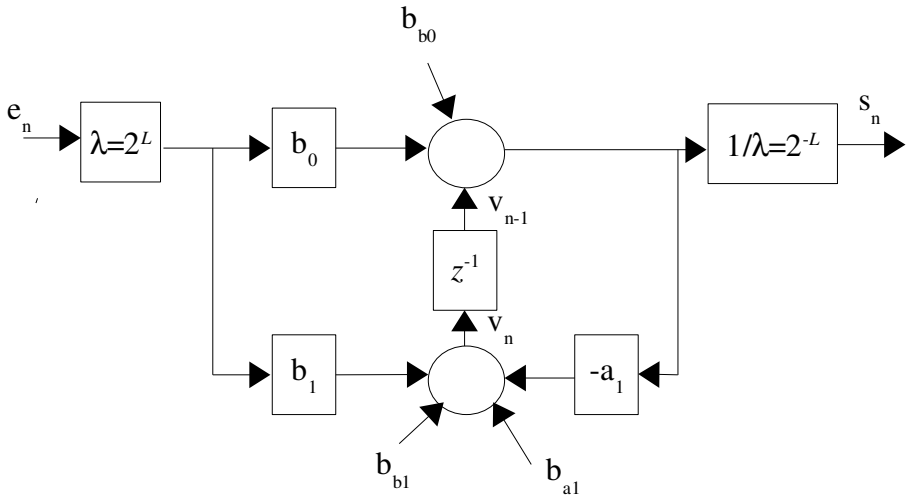

*figure 5: cellule **df2t** avec facteur d'échelle*

2.3.2 détection des variables internes

On repère ensuite les variables internes du schéma (celles qui doivent rester codables sur *N* bits, et qui sont susceptibles de dépasser cette échelle }. Il s'agit généralement des variables à l'entrée des multiplications par les coefficients.

Sur le schéma de pré-analyse de la forme ***df2t***, il y a 2 variables internes :

la variable $x1_n$ en entrée des multiplications par $\{b_0, b_1\}$

la variable $x2_n$ en entrée de la multiplication par $\{-a_1\}$

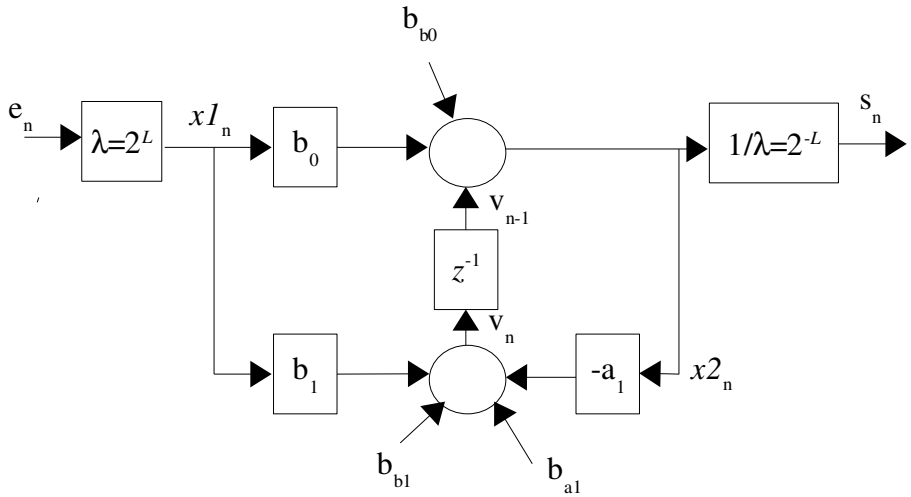

figure 6: schéma d'analyse df2t avec variables internes et bruits

2.3.3 problèmes posés par le calcul du facteur d'échelle de signal:

Le principe de calcul du facteur d'échelle $\lambda = 2^L$ *est le même que dans le cas de multiplications en parallèle: On cherche* $\lambda = 2^L$ *le plus grand possible, en veillant toutefois à ce que les variables internes* $\{x1_n, x2_n\}$ *ne dépassent pas l'échelle de codage sur N bits.*

1- Pour la variable $x1_n$ le calcul est facile, on a

$x1_n = \lambda . e_n$

Il faudra donc $|\lambda| \le \frac{|x1|_{max}}{|e|_{max}} = \frac{2^{[N-1]}}{2^{[N-1]}} = 1$ { N étant le nombre de bits de codage }

2- La détermination du maximum de $|x2_n|$ est beaucoup plus compliquée :
En oubliant les bruits de quantification de signal, on peut écrire

$$x2_n = b_0 . \lambda\, e_n + \underbrace{\left[b_1 \lambda\, e_{n-1} - a_1\, x2_{n-1} \right]}_{v_{n-1}}$$

Le lien entre l'entrée e_n *et la variable interne* $x2_n$ *est une équation récurrente*.
Dans ce cas il est impossible de déterminer $x2_n$ directement en fonction de e_n .
L'analyse des caractéristiques de $x2$ ne peut plus s'opérer qu'au moyen de la transformée en Z : il faut déterminer la fonction de transfert en z $H_{e\_x2}(z)$ entre l'entrée e_n et la sortie $x2_n$
Pour cela, On écrit la transformée en z (à conditions initiales nulles) de l'équation récurrente :

$$x2(z) = b_0 . \lambda\, e(z) + z^{-1} . \underbrace{\left[b_1 \lambda\, e(z) - a_1\, x2(z) \right]}_{Tr\ en\ Z\ [v_{n-1}]}$$

,et on en déduit l'expression de la fonction de transfert

$$x2(z) = \lambda \frac{b_0 + b1 . z^{-1}}{1 + a_1 . z^{-1}} . e(z) = H_{e\_x2}(z) . e(z)$$

2.3.4 maximum d'une variable, norme 1 d'une fonction de transfert

Le problème rencontré très souvent, lors de la détermination des valeurs maximales des variables internes xi_n *, est que le lien entre l'entrée* e_n *et les variables internes* xi_n *et une* ***<u>équation récurrente</u>****.*
<u>Dans ce cas il est impossible de calculer directement le maximum de</u> xi_n *en fonction du maximum de* e_n .

On va voir que pour effectuer ce calcul on doit employer la ***norme 1*** de la fonction de transfert $H_{e\_xi}(z)$ entre l'entrée e_n et la sortie xi_n , (notée $\left\| H_{e\_xi} \right\|_1$)

2.3.4.1 rappel, produit de convolution

soit $H_{e\_xi}(z) = \sum_{k=0}^{\infty} h_k . z^{-k}$, la fonction de transfert entre e_n et xi_n

et soit $e(z) = \sum_{l=0}^{\infty} e_l . z^{-l}$, la transformée en z de l'entrée e_n

à conditions initiales nulles, on a :

1- $xi(z) = H_{e\_xi}(z) . e(z) = \left[\sum_{k=0}^{\infty} h_k . z^{-k} \right] . \left[\sum_{l=0}^{\infty} e_l . z^{-l} \right] = \sum_{l=0}^{\infty} \sum_{k=0}^{\infty} h_k . e_l\, z^{-[k+l]}$,

ce qui donne, en posant $k+l=n$, et en faisant disparaître $k=n-l$

$$xi(z)=\sum_{l=0}^{\infty}\sum_{n=l}^{\infty} h_{n-l}.e_l\, z^{-n}$$

soit encore, en inversant les 2 sommes

$$xi(z)=H_{e\_xi}(z).e(z)=\sum_{n=0}^{\infty}\underbrace{\sum_{l=0}^{n} h_{n-l}.e_l}_{xi_n} z^{-n}=\sum_{n=0}^{\infty} xi_n.z^{-n}$$

on a donc finalement :

$$xi_n=\underbrace{\sum_{l=0}^{n} h_{n-l}.e_l}_{\text{produit de convolution de } h \text{ par } e \text{, à l'instant } n} = \underbrace{(h*e)_n}_{\text{produit de convolution de } h \text{ par } e \text{, à l'instant } n}$$

2.3.4.2 norme 1 d'une fonction de transfert (très important)

à partir de l'expression temporelle de xi_n , on peut à présent déterminer la valeur maximale de son module, à l'instant *n* :

$$xi_n=\sum_{l=0}^{n} h_{n-l}.e_l \Rightarrow |xi_n|_{max}=\left[\sum_{l=0}^{n}|h_{n-l}|\right].|e|_{max}$$

si maintenant on s'intéresse à la valeur maximale de $|xi_n|$, pour n'importe quel instant *n*, on peut écrire

$$|xi|_{max}=\underbrace{\left[\sum_{k=0}^{\infty}|h_k|\right]}_{\text{norme 1 de } H_{e\_xi}}.|e|_{max}=\left\|H_{e\_xi}\right\|_1.|e|_{max}$$

la norme 1 d'une fonction de transfert $H(z)$ ***est la somme des modules des échantillons*** h_n ***de sa transformée inverse en z (réponse impulsionnelle). Elle permet de calculer le taux d'amplification maximal entre l'entrée et la sortie.***

2.3.4.3 calcul de la valeur maximale des variables internes avec la norme 1

Concrètement, pour déterminer le maximum $|xi|_{max}$ d'une variable interne xi_n connaissant la valeur maximale $|e|_{max}$ de l'entrée e_n , on procède de la façon suivante :

étape 1- on détermine la fonction de transfert $H_{e\_xi}(z)$ entre l'entrée et la variable interne(telle que $xi(z)=H_{e\_xi}(z).e(z)$, à conditions initiales nulles)

étape 2- On en déduit la réponse impulsionnelle h_n de $H_{e\_xi}(z)=\sum_{n=0}^{\infty} h_n.z^{-n}$,

-soit par calcul analytique: décomposition en éléments simples de $\frac{H_{e\_xi}(z)}{z}$

- soit par simulation (cas le plus fréquent)

étape 3- on en déduit $\left\|H_{e\_xi}\right\|_1=\sum_{n=0}^{\infty}|h_n|$

soit par calcul analytique (limité à quelques cas particuliers)

soit par sommation sur un nombre d'échantillons *N* très grand :

$$\left\|H_{e\_xi}\right\|_1 \approx \sum_{n=0}^{N\ \text{très grand}} |h_n| \quad \text{(cas le plus fréquent, nécessite un calculateur)}$$

étape 4- on en déduit $|xi|_{max}$ en fonction de $|e|_{max}$ et de $\left\|H_{e\_xi}\right\|_1$ par :

$$|xi|_{max} = \left\|H_{e\_xi}\right\|_1 . |e|_{max}$$

2.3.4.4 calcul analytique de la norme 1 d'un système du premier ordre

la norme 1 d'une fonction de transfert $H(z) = \dfrac{b_0 + b_1 . z^{-1}}{1 + a_1 . z^{-1}}$ *du premier ordre s'écrit analytiquement :*

si $|a_1| < 1$: $\left\|H\right\|_1 = \sum_{n=0}^{\infty} |h_n| = |b_0| + |b_1 - a_1 . b_0| . \dfrac{1}{1 - |a_1|}$

si $|a_1| \geq 1$ $\left\|H\right\|_1 = \infty$ *{ système instable, la norme 1 est infinie}*

démonstration :

la décomposition en éléments simples de $H(z)$ conduit à :

$$H(z) = \left[b_0\right] + \left[b_1 - a_1 . b_0\right] . z^{-1} . \left[\frac{z}{z + a_1}\right]$$

on peut donc en déduire h_n par transformée inverse en *Z* (tables + théorème du retard)

$$h_n = \left[b_0\right] . \underbrace{\delta_n}_{\text{impulsion unité}} + \left[b_1 - a_1 . b_0\right] . [-a_1]^{n-1} . \underbrace{u_{n-1}}_{\text{échelon retardé de 1}}$$

et on en déduit finalement $\left\|H\right\|_1 = \sum_{n=0}^{\infty} |h_n|$:

$$\left\|H\right\|_1 = \sum_{n=0}^{\infty} |h_n| = |b_0| + |b_1 - a_1 . b_0| . \frac{1}{1 - |a_1|}$$, à condition que $|a_1| < 1$

remarques (cas particuliers):

pour $H(z) = \dfrac{b_0}{1 + a_1 . z^{-1}} = \dfrac{b_0 . z}{z + a_1}$, on a $\left\|H\right\|_1 = \dfrac{|b_0|}{1 - |a_1|}$

pour $H(z) = \dfrac{b_1 . z^{-1}}{1 + a_1 . z^{-1}} = \dfrac{b_1}{z + a_1}$, on a $\left\|H\right\|_1 = \dfrac{|b_1|}{1 - |a_1|}$

2.3.4.5 application à l'exemple considéré (calcul de $|x2|_{max}$ *et de* λ *)*

pour notre exemple, on a : $x2(z) = \lambda \dfrac{b_0 + b1 . z^{-1}}{1 + a_1 . z^{-1}} . e(z) = H_{e\_x2}(z) . e(z)$,

on aura donc : $|x2|_{max} = \left\|H_{e\_x2}\right\|_1 . |e|_{max}$,

avec $H_{e\_x2}(z) = \lambda \dfrac{b_0 . z + b1}{z + a_1} \approx \lambda \dfrac{22{,}35\, z - 20{,}79}{z - 0{,}569}$, on aura donc :

$$\left\|H_{e\_x2}\right\|_1 = |\lambda| . \left[|b_0| + |b_1 - a_1 . b_0| . \frac{1}{1 - |a_1|}\right] \approx 41 . |\lambda|$$

On peut donc à présent calculer le facteur d'échelle λ le plus grand possible

pour que $|x2|_{max} = \left\| H_{e\_x2} \right\|_1 . |e|_{max}$ reste codable sur l'échelle *N* bits :

1- on a : $|e|_{max} = 2^{N-1}$

2- on veut que : $|x2|_{max} < 2^{N-1} - 1$

3- il faut donc que : $\left\| H_{e\_x2} \right\|_1 < \frac{|x2|_{max}}{|e|_{max}} = \frac{2^{N-1}-1}{2^{N-1}} \approx 1$

4- application numérique : $41 . \lambda < 1 \Leftrightarrow \lambda < \frac{1}{41} \approx 2^{-5,4}$, on retient donc $\lambda = 2^{-6}$

2.3.5 *analyse des reports de bruit.*

Pour que cette partie soit plus claire, on va raisonner tout d'abord sur le schéma d'implémentation en forme ***df2t*** du régulateur avance de phase { on supposera que l'on travaille en arithmétique ***8/16*** bits signée }.

Le tableau des coefficients réels, entiers, quantifiés, et des décalages à droite associés, est donné ci-dessous

coeff réel c	*coeff entier* c_N	décalage L_c	*coeff quantifié* $c_q = 2^{-Lc} . c_N$
$b_0 \approx 22.35$	$b_{0N} = 89$	$L_{b0} = 2$	$b_{0q} \approx 22.25$
$b_1 \approx -20.75$	$b_{1N} = -83$	$L_{b1} = 2$	$b_{1q} \approx -20.75$
$-a_1 \approx 0.569$	$[-a_1]_N = 73$	$L_{a1} = 7$	$[-a_1]_q \approx 0.570$

2.3.5.1 cas d'une programmation naïve

Le schéma (peu judicieux) d'implémentation , correspondant au schéma de principe en 2.3.1 est donné ci-dessous{ on a représenté les décalages à droite ou a gauche par les caractères **>>** ou **<<** }

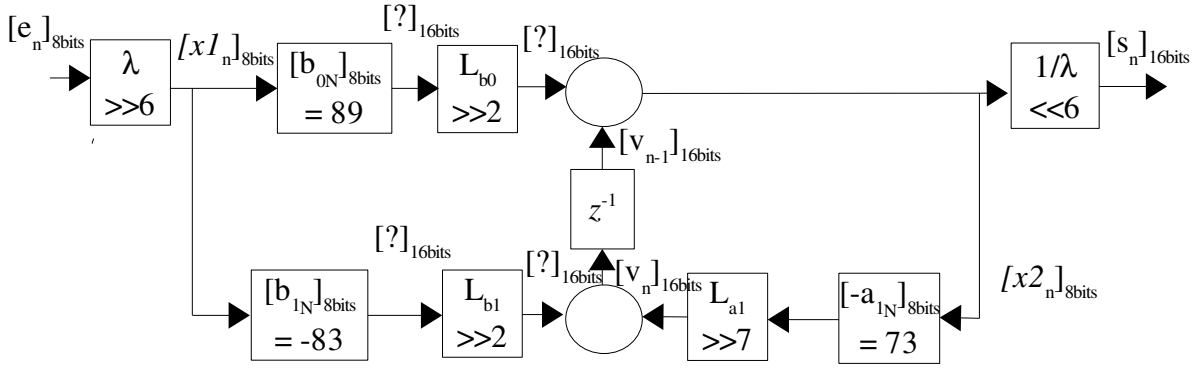

figure 7: schéma d'implémentation naïf du correcteur sous forme df2t

on en déduit le schéma d'analyse associé { un bruit de quantification s'ajoute en sortie de chaque décalage à droite } figure 8

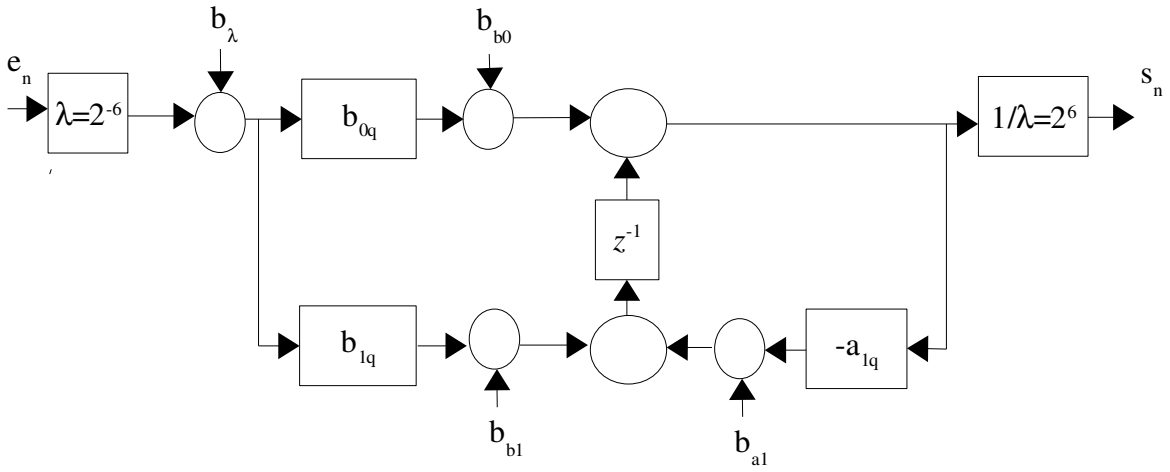

figure 8: schéma d'analyse associé au schéma d'implémentation naïf

L'analyse de l'effet des bruits de quantification sur la sortie s_n s'effectue en écrivant sa transformée en Z $S(z)$, uniquement en fonction des bruits {ici $b_\lambda, b_{b0}, b_{b1}, b_{a1}$ }.
tous calculs faits, on obtient :

$$S(z)=\frac{1}{\lambda}.\frac{1}{1+a1_q.z^{-1}}.\left[b_{b0}(z)+z^{-1}.b_{a1}(z)+z^{-1}b_{b1}(z)+b_{0q}.b_{\lambda(z)}\right]$$
$$=H_{b0\_s}(z).b_{b0}(z)+H_{b1\_s}(z).b_{b1}(z)+H_{a1\_s}(z).b_{a1}(z)+H_{\lambda\_s}(z).b_\lambda(z)$$

avec $$H_{b0\_s}=\frac{1}{\lambda}.\frac{1}{1+a_{1q}.z^{-1}}$$,

$$H_{b1\_s}=H_{a1\_s}=\frac{1}{\lambda}.\frac{z^{-1}}{1+a_{1q}.z^{-1}}$$

$$H_{\lambda\_s}=\frac{1}{\lambda}.\frac{b_{0q}}{1+a_{1q}.z^{-1}}$$

On peut alors en déduire la valeur maximum de $|s_n|$ due aux bruits de quantification de signal, au moyen des normes 1 (calculées analytiquement avec la formule en 2.3.4.4):

$$|s|_{max}=\underbrace{\left\|H_{b0\_s}\right\|_1}_{\approx 149}.\underbrace{|b_{b0}|_{max}}_{=1}+\underbrace{\left\|H_{b1\_s}\right\|_1}_{\approx 149}.\underbrace{|b_{b1}|_{max}}_{=1}+\underbrace{\left\|H_{a1\_s}\right\|_1}_{\approx 149}.\underbrace{|b_{a1}|_{max}}_{=1}+\underbrace{\left\|H_{\lambda\_s}\right\|_1}_{\approx 3312}.\underbrace{|b_\lambda|_{max}}_{=1}\approx 3759$$

dans le pire des cas, le bruit de sortie peut atteindre 3759

Pour donner une signification à cette valeur, il faut la comparer à la portion utile de la sortie {= partie de la sortie s_n due uniquement à l'entrée e_n :
en négligeant les bruits, on a :

$$S(z)=\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)=H_{e\_s}(z).e(z)$$

Le maximum de la portion utile de la sortie s'écrit donc :

$$|s|_{max} = \underbrace{\left\| H_{e\_s} \right\|_1}_{\approx 41} . \underbrace{|e|_{max}}_{= 2^{8-1} = 128} \approx 5248$$

Dans le pire des cas, le bruit de sortie occupe plus de la moitié du maximum de la valeur utile de sortie : c'est parfaitement inacceptable!...

il est donc parfaitement inutile de chercher à programmer ce schéma, sauf pour le plaisir de perdre son temps...

2.3.5.2 règles à appliquer pour une programmation plus professionnelle

On peut améliorer nettement les choses en appliquant les règles suivantes

règle 1- ***les décalages à droite doivent être regroupés autant que possible derrière les multiplications par des coefficients entiers***

=> le schéma d'implémentation figure 9

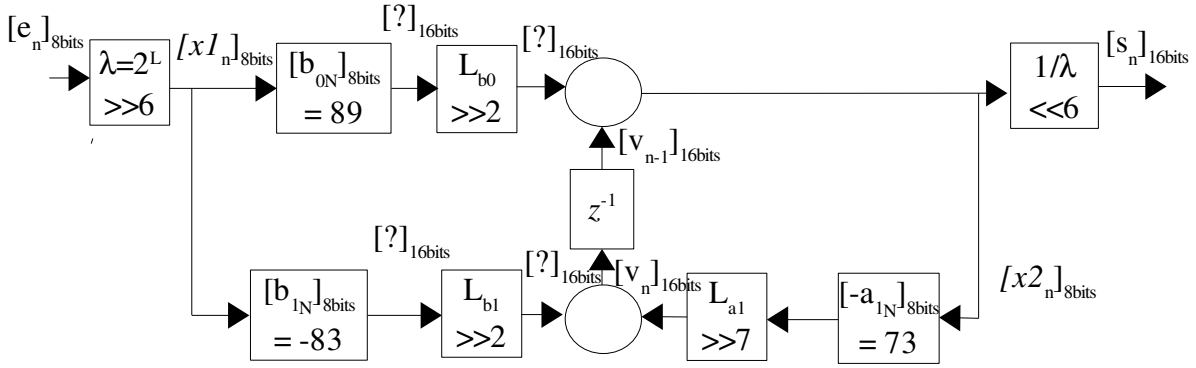

figure 9: schéma d'implémentation naïf

doit être remplacé par celui ci { décalage de 6 bits à droite lié à $\lambda = 2^{-6}$ reporté en sortie des multiplications par b_{0N}, b_{1N} }

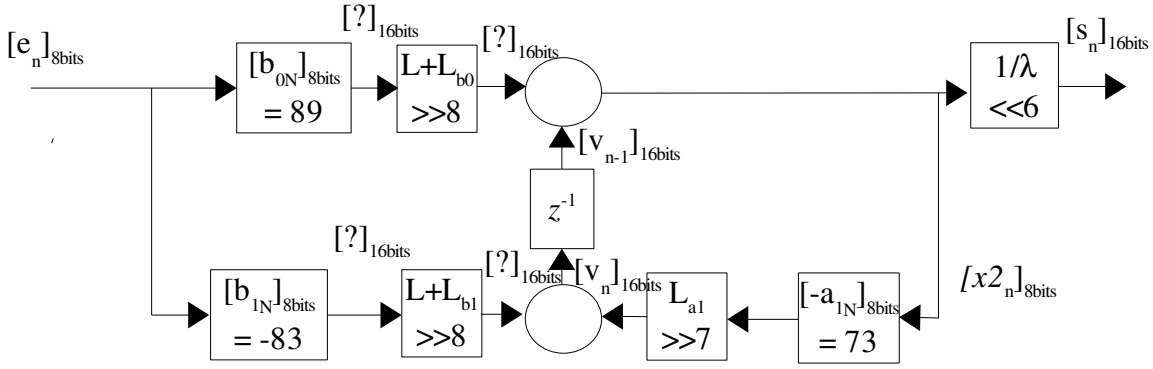

figure 10: schéma d'implémentation après regroupement des décalages à droite en sortie des multiplications

règle 2 : ***lorsqu'il y a plusieurs multiplications en parallèle, les décalages à droite doivent être regroupés autant que possible en sortie des additionneurs (en veillant à ne pas générer de dépassement d'échelle)***

=> les décalages à droite en sortie des multiplications par $b_{0N}, b_{1N}, [-a_{1N}]$ peuvent être regroupés derrière l'additionneur commun, ce qui donne le schéma d'implémentation optimisé:

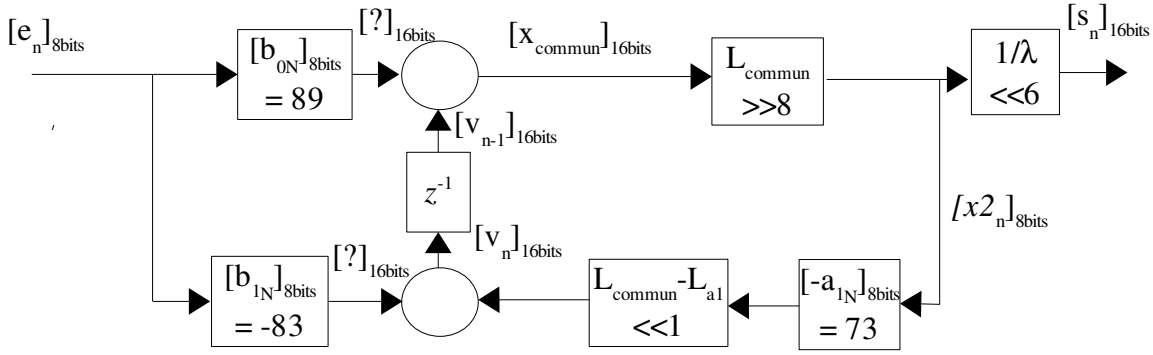

figure 11: schéma après regroupement des décalages à droite en sortie des additionneurs

***remarque importante** : le schéma précédent fait généralement très peur aux informaticiens, pour la raison suivante*

La variable $\left[x2_n\right]_{8bits}$ est multipliée par le coefficient 73, puis le résultat est décalé à gauche de 1 bit => on peut avoir un dépassement d'échelle sur le résultat, et donc aussi sur les variables $\left[v_n\right]_{8bits}$ et $\left[v_{n-1}\right]_{8bits}$

-ceci est parfaitement exact , et n'a aucune importance...

- ce qui est important, c'est que la variable $\left[x2_n\right]_{8bits}$ ne dépasse pas la pleine échelle de codage sur 8 bits, et que la variable $\left[x_{commun}\right]_{16bits}$ ne dépasse pas la pleine échelle de codage sur 16 bits.

Or, dans la partie 2.3.4.5 , on a calculé le facteur d'échelle λ pour que la variable $\left[x2_n\right]_{8bits}$ ne dépasse pas la pleine échelle sur 8 bits... Plus précisément, on sait que $\left|x2\right|_{max} \approx \underbrace{\left\|H_{e\_x2}\right\|_1}_{\approx 41} . \underbrace{\lambda}_{=2^{-6}} . \underbrace{\left|e\right|_{max}}_{=128} \approx 82$

Il n'y a donc aucun risque de dépassement sur la variable $\left[x_{commun}\right]_{16bits}$, tant que le décalage L_{commun} reste inférieur à 8 { $2^8.82 = 20992 < 2^{15} = 32768$ }!...

- On peut également (calcul inutile, mais plus systématique) calculer directement le maximum de $\left[x_{commun}\right]_{16bits}$, de la même façon qu'en 2.3.4.3

1- on a $x_{commun}(z) = \dfrac{89 - 83.z^{-1}}{1 - \underbrace{73.2^{-L_{a1}}}_{-a_{1q}} . z^{-1}} . e(z) = H_{e\_x_{commun}}(z) . e(z)$

2- on a donc $\left|x_{commun}\right|_{max} = \underbrace{\left\|H_{e\_x_{commun}}\right\|_1}_{\approx 164} . \underbrace{\left|e\right|_{max}}_{=128} \approx 20992 < 2^{15}$

même conclusion :Il n'y a aucun risque que la variable $\left[x_{commun}\right]_{16bits}$ dépasse la pleine échelle de codage sur 16 bits!...

Règle 3: on doit essayer d'employer autant que possible les variables aux échelles les plus précises {bannir les décalages à gauche qui suivent des décalages à droite}.

En effet un décalage à droite ajoute en sortie un bruit de quantification, qui sera ensuite amplifié par le décalage à gauche, ce qu'il faut absolument proscrire...

Par exemple, sur le schéma, le décalage à gauche de 6 bits, correspondant à la

multiplication par $\frac{1}{\lambda}$ *, est placé juste après un décalage à droite de 8bits {correspondant à la mutiplication par* $2^{-L_{commun}}$ *}. C'est une ânerie monumentale!..*

En effet, de cette façon, on calcule :

$$s_n = 2^6 . \left[2^{-8} . x_{commun} + bruit_{xcommun} \right] = 2^{-2} . x_{commun} + 2^6 . bruit_{xcommun}$$

Il est nettement préférable d'opérer directement un décalage à droite de 2 bits de la variable x_{commun}

en effet, de cette façon, on calcule : $s_n = 2^{-2} . x_{commun} + 1 . b_s$ *, ce qui correspond à un niveau de bruit beaucoup plus faible...*

Le schéma de programmation à retenir finalement, en appliquant les 3 règles ci-dessus { que vous devrez appliquer chaque fois que vous aurez à concevoir un code en nombres entiers } est représenté figure 12

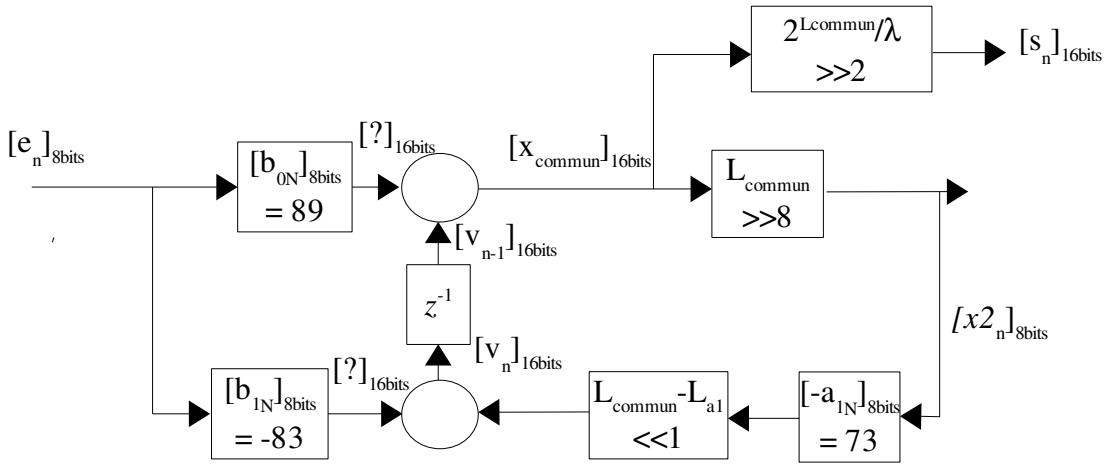

figure 12: schéma d'implémentation optimisé

2.3.5.3 programme correspondant en langage C

Il est très facile de passer d'un schéma d'implémentation au programme correspondant.

- La seule difficulté conceptuelle réside dans la façon dont on programme les retards z^{-1} *.{ comment passe-t-on de* v_n *à* $v_{[n-1]}$ *dans le programme}.*

Pour cela on utilise une *mémoire*, qui stocke la valeur de la variable, d'un pas d'échantillonnage au suivant.

A CHAQUE PAS D'ECHANTILLONNAGE :

- au début, le contenu de la *mémoire* représente la variable $v_{[n-1]}$ (la sortie du retard z^{-1}).

- à la fin, on met à jour le contenu de la *mémoire* pour qu'il corresponde à l'expression de v_n (l'entrée du retard z^{-1})

AU TOUT DEBUT(INITIALISATION) :

la mémoire initiale représente v_{-1} . Donc, avant le premier pas, il faut l'initialiser (typiquement à 0) .

voici par exemple le programme en langage *c*, correspondant au schéma optimisé du régulateur à avance de phase (lire les commentaires)

```
   // types utilises entiers 8 bits et entiers 16 bits signes
#define int_8 char
#define int_16 short int
typedef struct {
  int_16 memoire_v; // memoire associee a la variable [vn]16
} s_filter;
typedef s_filter *p_filter; // p_filter=pointeur sur une structure de type
s_filter
// fonction initialisant les coeffs et les memoires du filtres
  p_filter new_filter(){
    p_filter p;
    p=(p_filter) malloc(sizeof(s_filter)); // reserve memoire pour p
  // initialisation de la memoire associee a [vn]
    p->memoire_v=0;
    return(p);
  }
 //-----------------------------------------------------
 // fonction appelee a chaque pas d'echantillonnage,
 //   calcule la sortie sn_16 du filtre
 //   en fonction de l'entree en_8, et de la structure p
 //   (met egalement a jour les memoires du filtre)
 //-----------------------------------------------------
   int_16 one_step_filter(int_8 en, p_filter p){
     int_16 sn_16;
     int_16 acc_16; // accumulateur 16 bits a tout faire
     int_8 x2_8; // variable [x2n]8bits
   // phase 1, calcul de sn en fonction de en et vn-1
      acc_16=89*en_8;       // acc=89.en
   // a ce stade p->memoire_v contient la valeur precedente de v[n], donc v[n-1]
      acc_16+=p->memoire_v; // acc=acc+v[n-1]
      sn_16=acc_16>>2;  // calcul de sn_16
   // mise a jour de la memoire=> calcul de vn
      acc_16>>=7; // acc=2^-8 . acc, avec arrondi
      acc_16+=1;
      acc_16>>=1; // a ce stade, acc represente x2, mais est code sur 16 bits
   // on fait apparaitre x2 uniquement par souci de lisibilite...
      x2_8=(int_8)acc_16; // x2<- acc, converti en entier 8 bits, aucun risque
      acc_16<<73*x2_8;
      acc_16<<=1; // acc=acc << 1
      acc_16+=(-83)*en_8; // acc=acc-83*en
   // mise a jour de la memoire associee a vn pour la prochaine fois
      p->memoire_v=acc_16;
   // on renvoie sn
      return(sn_16);
   }
```

2.3.5.4 analyse des reports d'erreur

le schéma d'analyse associé au schéma d'implémentation final est représenté figure 13 { *sur ce schéma il vaut mieux faire apparaître les coefficients quantifiés, plutôt que les coefficients entiers. Cela permet de mieux comprendre le rôle des facteurs d'échelle* }

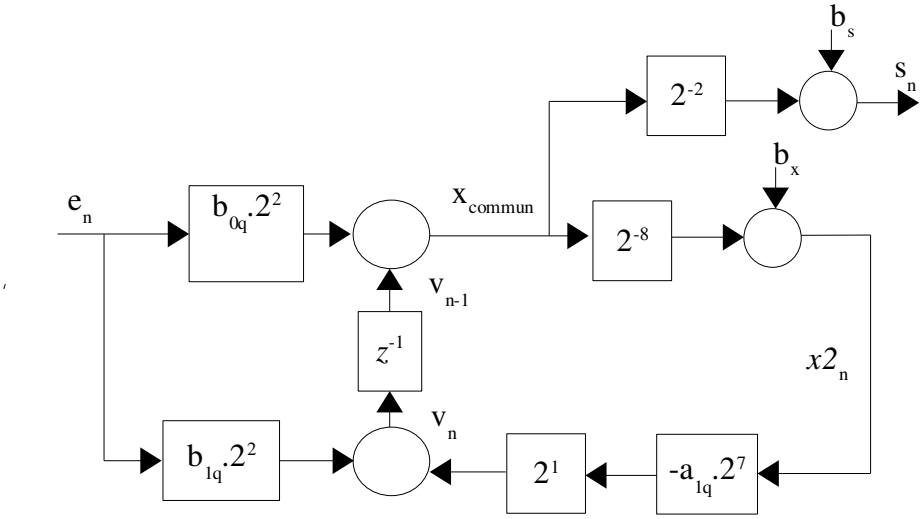

figure 13: schéma d'analyse associé au schéma d'implémentation optimisé

les équations en z de ce schéma sont alors :

$$x_{commun}(z)=2^2.\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)+2^8.\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)$$

$$s(z)=2^{-2}.x_{commun}(z)+b_s(z)=\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)+2^6.\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)+b_s(z)$$

la partie de la sortie uniquement due aux bruits, s'écrit :

$$s(z)=\underbrace{2^6.\frac{z^{-1}}{1+a_{1q}.z^{-1}}}_{H_{bx\_s}}.b_x(z)+\underbrace{1}_{H_{bs\_s}}.b_s(z)$$

on peut alors en déduire analytiquement la valeur maximum du bruit de sortie{portion de s_n due uniquement aux bruits} :

$$|s_n|_{max}=\underbrace{\left\|H_{bx\_s}\right\|_1}_{\approx 148}.\underbrace{|bx|_{max}}_{=0.5}+\underbrace{\left\|H_{bs\_s}\right\|_1}_{=1}.\underbrace{|bs|_{max}}_{=1}\approx 75$$

cette valeur est à comparer à la valeur *3759* précédemment obtenue en 2.3.5.1 , et à la valeur maximale *5248* de la portion utile de la sortie.

Remarques (qui n'étaient pas évidentes a priori):
1- on divise pratiquement par 2 le bruit de sortie en quantifiant par arrondi la multiplication par 2^{-8} correspondant au bruit b_x (ce qui explique l'arrondi dans le programme)
2-il n'est pas utile d'arrondir la sortie de la multiplication par 2^{-2} , qui ne produit que peu d'erreur...

3 conception et analyse 'standard' d'un filtre numérique iir

3.1 schéma d'analyse standard

les équations du schéma optimisé de la forme **df2t** en exemple s'écrivent

1- équation de la sortie s_n

$$s(z)=\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)+\frac{1}{\lambda}.\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)+b_s(z)$$

2- équation de la variable interne $x2_n$

$$x2(z)=\lambda.\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)+\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)$$

ces équations correspondent au schéma d'analyse figure 14...

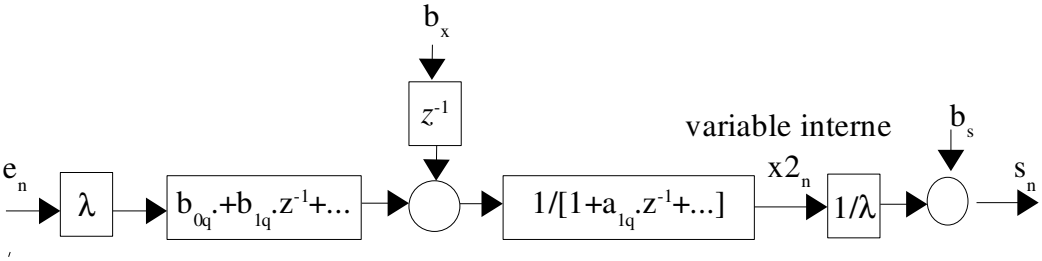

*figure 14: schéma d'analyse standard de la forme **df2t***

Ce schéma permet, sans se soucier de la façon dont on mènera les différentes optimisations, de déterminer a priori la valeur maximale du facteur d'échelle λ et d'analyser les performances obtenues. À l'avenir, tant que l'on ne passera pas au codage à proprement parler, on se contentera d'analyser les performances de la forme ***df2t*** en raisonnant uniquement sur ce schéma!...

3.2 Rôle (pas évident) du nombre N de bits de codage, et gain relatif de bruit

le schéma d'analyse standard figure 14 et les équations correspondantes sont, si l'on y pense, assez surprenants :

en supposant que les coefficients quantifiés b_{iq}, a_{jq} sont proches des coefficients réels b_i, a_j

1- ***le plus grand facteur d'échelle λ ne dépend pratiquement pas du nombre N de bits de codage !...***:

en notant : $H_{e\_s}(z)=\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}$

on obtient $\lambda_{max}=\frac{1}{\left\|H_{e\_s}\right\|_1}.\frac{|x2_{max}|}{|e_{max}|}\approx\frac{1}{\left\|H_{e\_s}\right\|_1}$, qui ne dépend pas de N

2-le bruit de sortie ne dépend pas non plus du nombre de bits de codage!..

la transformée en z de la sortie, en présence uniquement de bruit, s'écrit

$$s(z)=\frac{1}{\lambda}.\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)+b_s(z)$$

comme les niveaux de bruits ne dépendent pas du nombre N de bits de codage, mais juste de la façon dont on quantifie la sortie des décalages à droite, il est donc évident que le niveau de bruit de sortie ne dépend pas non plus de N .

avec un raisonnement pareil, on est bien parti pour conclure qu'avec un processeur sur 4 bits on obtient les mêmes performances qu'avec un processeur sur 32 bits!...

ON A TOUTEFOIS OUBLIE UN (TRES GROS) DETAIL

les niveaux absolus de bruit sont à comparer aux niveaux absolus de la partie utile des variables...

par exemple, pour l'exemple traité, on a (***indépendamment de*** N)

1- $\lambda_{max} \approx \frac{1}{\left\| H_{e\_s} \right\|_1} = 2^{-6}$

2- bruit de sortie max : $|s|_{max} = \underbrace{\frac{1}{\lambda}}_{2^6} . \underbrace{\left\| \frac{z^{-1}}{1 + a_{1q} . z^{-1}} \right\|_1}_{2.32} . \underbrace{|b_x|_{max}}_{0.5} + \underbrace{|b_s|_{max}}_{1} \approx 75$

mais d'un autre côté, on a (***et là les valeurs dépendent de*** N ***!...***)

1- $|e|_{max} = 2^{(N-1)}$

2-partie utile de la sortie $|s|_{max} = \underbrace{\left\| H_{e\_s} \right\|_1}_{41} . |e|_{max} \approx 41.2^{N-1}$

en termes de valeurs relatives, le niveau de bruit de sortie, relativement au niveau de sortie utile, est inversement proportionnel au nombre de bits de codage :

$$\frac{|s|_{max} \text{ due au bruits}}{|s|_{max} \text{due à l'entrée}} \approx \frac{75}{41} . \frac{1}{2^{N-1}} \approx 1{,}82 . \frac{1}{2^{N-1}}$$

le nombre de bits de codage N permet de fixer le niveau relatif de bruit, pour qu'il colle au niveau relatif de bruit désiré .

*Le gain relatif de **1,82** est une caractéristique essentielle de la cellule : il détermine son comportement intrinsèque en termes d'amplification de bruit (ce terme est appelé **gain relatif de bruit**)*

- le ***gain relatif de bruit*** permet

1- d'évaluer le rapport bruit/ signal, connaissant le nombre de bits de codage :

pour $N = 8$, on a un rapport bruit /signal $\frac{1{,}82}{2^7} \approx 0.014$

2- de déterminer le nombre de bits de codage pour un rapport signal/ bruit donné

si à présent on veut un rapport bruit/signal inférieur à 10^{-3} {en norme 1}, il faudra choisir N tel que :

$$2^{N-1} > \frac{1{,}82}{10^{-3}} \Leftrightarrow N - 1 > \log_2\left(\frac{1{,}82}{10^{-3}}\right) \approx 10{,}8 \Leftarrow N = 12 \text{ bits de codage}$$

3- et surtout de <u>comparer les qualités intrinsèques</u> de plusieurs implémentations possibles.

Si par exemple la ***forme directe 1*** a un gain relatif de bruit égal à **0,05**,
il faudra la préférer à la ***df2t***, qui a un gain relatif de bruit égal à **1,82**

3.3 schéma d'analyse standard et équations <u>en virgule fixe</u>

On a volontairement passé sous silence au début de cette présentation les notions de virgule fixe, pour détailler l'implémentation et l'analyse en nombres entiers d'un filtre simple (régulateur à avance de phase sous forme **df2t**)...

- Après un long chemin de croix (quantification des coefficients, analyse des reports de bruits, optimisation du code), on a fait apparaître un schéma d'analyse standard, permettant de déterminer les caractéristiques (facteur d'échelle, gain relatif de bruit), du filtre programmé pour une forme **df2t**

Les spécialistes du domaine (qui n'ont pas de temps à perdre) raisonnent exactement de la façon inverse, et ***c'est comme ça qu'il faudra procéder à l'avenir*** :

1- ils se servent du(es) schéma(s) d'analyse standard (tel que celui figure 14) pour analyser les caractéristiques du filtre (calcul des facteurs d'échelle, gain relatif de bruit, choix du nombre de bits, comparaison des différentes implémentations)
2- lorsqu'ils sont satisfaits des performances obtenues, et seulement à ce moment, ils passent au codage.
3- qui plus est , ils raisonnent généralement sur un schéma ,'à virgule fixe', ce qui est l'objet de ce paragraphe:

Considérons que nous disposions d'un schéma d'analyse 'standard', tel que celui figure 14
1***- On peut alors raisonner sur ce schéma 'en nombres entiers'***
On considère un module maximum de l'entrée égal à 2^{N-1} , et un module maximum des bruits de quantification égal à 1 ou 0.5 { selon que l'on quantifie par troncature ou par arrondi les sorties des décalages à droite }=> cela revient à normaliser par rapport au niveau de bruit, c'est le niveau d'entrée qui varie en fonction du nombre N de bits de codage. C'est le raisonnement ***en nombres entiers*** que nous avons employé...
2- ***On peut également raisonner 'en virgule fixe'***
- les spécialistes normalisent par rapport au niveau d'entrée: ils considèrent un module maximum d'entrée égal à 1 , et un module maximum des bruit égal à $2^{-[N-1]}$ ou 2^{-N} { selon que l'on quantifie par troncature ou par arrondi les sorties des décalages à droite }. ils normalisent par rapport au niveau d'entrée, c'est le niveau de bruit qui varie en fonction du nombre N de bits de codage. C'est le raisonnement '***en virgule fixe'*** *qui est généralement utilisé...*

Noter que les résultats finalement obtenus (facteurs d'échelle, gains relatif de bruit, rapport bruit/signal, valeurs des coefficients quantifiés) et l'implémentation finale du filtre sont exactement les mêmes, que l'on raisonne en ***virgule fixe*** *ou en* ***nombres entiers****...*

Comme toute la littérature de ce domaine est basée sur le raisonnement ***en virgule fixe****, il vaut mieux s'y habituer et raisonner de cette façon sur les schémas standard...*

3.4 norme 2 d'un système, et utilisation

3.4.1 *définition et signification*

Pour le moment,l'analyse des performances s'est limitée à la détermination de la valeur maximale du module bruit de sortie.
D'autres caractéristiques très importantes peuvent être examinées, en particulier la variance, ou l'écart-type, du bruit de sortie. Pour cela on emploie la **norme 2**.
1- la norme 2 d'une fonction de transfert $H(z)$ est définie par :

$$\left\|H(z)\right\|_2 = \sqrt{\sum_{n=0}^{\infty} |h_n|^2}$$

la ***norme 2*** peut se calculer analytiquement, (résolution d'équation de *Lyapunov*, mais cela sort du cadre de ce cours), on peut aussi l'évaluer en sommant sur un nombre 'suffisamment grand' d'échantillons : $\left\|H(z)\right\|_2 \approx \sqrt{\sum_{n=0}^{N\ très\ grand} |h_n|^2}$

signification :
Le carré de la ***norme 2*** du filtre traduit le taux d'amplification 'en variance' du filtre, dans le cas d'une entrée blanche , plus précisément :

-Lorsque l'entrée e_n *de* $H(z)$ *est un signal aléatoire*
de moyenne nulle $E\{e_n\} = 0$
de puissance (moyenne du carré) $P_e = E\{e_n^2\}$
dont tous les échantillons sont indépendants entre eux (on parle alors d'une entrée blanche): $E\{e_n . e_{n+k}\} = 0, k \neq 0$

-la puissance $P_s = E\left\{s_n^2\right\}$ *de la sortie* s_n *s'écrit alors :*
$P_s = \left\|H(z)\right\|_2^2 . P_e$.

lorsque l'entrée e_n n'est pas à moyenne nulle, mais que ses échantillons restent indépendants entre eux, le carré de la norme 2 donne le taux d'amplification en 'variance' .(la variance est la moyenne du carré des variations...)

$$V_s = \left\|H(z)\right\|_2^2 . V_e$$

lorsque qu'on a plusieurs entrées indépendantes et blanches, ce sont les puissances (et aussi les variances) qui s'ajoutent :

$$S(z) = \sum_i H_i(z) . e_i(z) \quad => \quad V_s = \sum_i \left\|H_i(z)\right\|_2^2 . V_{ei}$$

La *norme 2* permet donc de déterminer la variance du bruit de sortie, en supposant que les bruits de quantification sont des variables aléatoires blanches indépendantes entre elles...

remarque:
La variance et la puissance d'un signal sont homogènes au carré du signal, et pour cette raison leur valeur est assez peu parlante. On préfère généralement calculer leurs racines carrées, appelées respectivement ***écart-type***, et ***valeur efficace,*** car elles sont homogènes au (ont la même unité que le) signal :

valeur efficace de s_n = racine carrée de la puissance:

$$s_{eff} = \sqrt{P_s = E\left\{s_n^2\right\}}$$

{on rappelle que la puissance est égale à la *variance + moyenne au carré* }

écart-type = racine carrée de la variance (puissance des variations) :

$$\sigma_s = \sqrt{v_s = E\left\{\underbrace{\left[s_n - E\{s_n\}\right]}_{s - moyenne\ de\ s}^2\right\}}$$

3.4.2 *calcul analytique de la norme 2 d'un système du premier ordre*

soit un système de fonction de transfert $H(z)=\frac{b_0+b_1.z^{-1}}{1+a_1.z^{-1}}$,

sa réponse impulsionnelle est alors (voir 2.3.4.4)

$$h_n=\left[b_0\right].\underbrace{\delta_n}_{\text{impulsion unité}}+\left[b_1-a_1.b_0\right].\left[-a_1\right]^{n-1}.\underbrace{u_{n-1}}_{\text{échelon retardé de 1}}$$

La norme 2 du système s'écrit donc

$$\left\|H(z)\right\|_2=\sqrt{\sum_{n=0}^{\infty}|h_n|^2}=\sqrt{b_0^2+\frac{\left[b_1-a_1.b_0\right]^2}{1-a_1^2}}$$

3.4.3 *application au calcul de l'écart-type du bruit de sortie, sur l'exemple considéré*

pour l'exemple considéré, le schéma d'analyse standard de la forme df2t en 3.1 conduisait à l'expression suivante de *S(z) :*

$$s(z)=\frac{b_{0q}+b_{1q}.z^{-1}}{1+a_{1q}.z^{-1}}.e(z)+\frac{1}{\lambda}.\frac{z^{-1}}{1+a_{1q}.z^{-1}}.b_x(z)+b_s(z)$$

en supposant que b_x et b_s sont des bruits blancs indépendants, de variances (puissance des variations) respectives v_{bx} et v_{bs} , on peut en déduire la variance v_s du bruit de sortie (de la portion de la sortie, due uniquement aux bruits), par :

$$v_s=\underbrace{\left\|\frac{1}{\lambda}.\frac{z^{-1}}{1+a_{1q}.z^{-1}}\right\|_2^2}_{\approx 78^2}.v_{bx}+\underbrace{\left\|1\right\|_2^2}_{=1^2}.v_{bs}$$

raisonnement en nombres entiers :

On suppose que les bruits b_x et b_s sont à densité de probabilité uniforme

entre 0 et 1 { si on quantifie par troncature}

entre -0.5 et 0.5 { si on quantifie par arrondi}

dans les 2 cas, la variance des bruits est la même :

$$v_{bx}=v_{bs}=\frac{1}{12}$$

la puissance est égale à somme de la variance et du carré de la moyenne

donc dans le cas d'une troncature : $P_{bx}=E\left(b_x^2\right)=\frac{1}{12}+\left[\frac{1}{2}\right]^2=\frac{1}{3}$

et dans le cas d'un arrondi : $P_{bx}=E\left(b_x^2\right)=\frac{1}{12}+0^2=\frac{1}{12}$

variance du bruit de sortie

connaissant les variances des bruits, on en déduit la variance, puis l'écart-type, du bruit de sortie

variance du bruit de sortie : $v_s=78^2.\frac{1}{12}+1^2.\frac{1}{12}\approx 507$

écart-type du bruit de sortie : $\sigma_s = \sqrt{(v_s)} \approx 23$

la valeur maximale de l'entrée est (en fonction du nombre N de bits de codage)

$|e|_{max} = 2^{N-1}$ et on peut en déduire le niveau maximal de sortie due à l'entrée :

$$s_{max} = \underbrace{\left\| \frac{b_{0q} + b_{1q}.z^{-1}}{1 + a_{1q}.z^{-1}} \right\|_1}_{\approx 41} . \underbrace{2^{N-1}}_{e_{max}}$$

le rapport entre l'écart-type du bruit de sortie et la valeur maximale de l'entrée est un rapport bruit/signal, qui vaut :

$$\frac{\sigma_s \text{dû aux bruits}}{s_{max} \text{due a l'entree}} \approx \frac{23}{41} . \frac{1}{2^{N-1}} \approx 0{,}56 . \frac{1}{2^{N-1}}$$

raisonnement en virgule fixe :

On suppose que les bruits b_x et b_s sont à densité de probabilité uniforme

entre 0 et $2^{-[N-1]}$ { si on quantifie par troncature}

entre -2^{-N} et 2^{-N} { si on quantifie par arrondi}

dans les 2 cas, la variance des bruits est la même :

$$v_{bx} = v_{bs} = \frac{2^{-2.[N-1]}}{12} \quad ,$$

la puissance est égale à somme de la variance et du carré de la moyenne

donc dans le cas d'une troncature : $P_{bx} = \frac{2^{-2.[N-1]}}{12} + \left[\frac{2^{-[N-1]}}{2} \right]^2 = \frac{2^{-2.[N-1]}}{3}$

et dans le cas d'un arrondi : $P_{bx} = \frac{2^{-2.[N-1]}}{12} + 0^2 = \frac{2^{-2.[N-1]}}{12}$

variance du bruit de sortie

connaissant les variances des bruits, on en déduit la variance, puis l'écart-type, du bruit de sortie

variance du bruit de sortie : $v_s = 78^2 . \frac{2^{-2.[N-1]}}{12} + 2^{-2.[N-1]} . \frac{1}{12} \approx 507 . 2^{-2.[N-1]}$

écart-type du bruit de sortie : $\sigma_s = \sqrt{(v_s)} \approx 23 . 2^{-[N-1]}$

la valeur maximale de l'entrée est

$|e|_{max} = 1$ et on peut en déduire le niveau maximal de sortie due à l'entrée :

$$s_{max} = \underbrace{\left\| \frac{b_{0q} + b_{1q}.z^{-1}}{1 + a_{1q}.z^{-1}} \right\|_1}_{\approx 41} . \underbrace{1}_{e_{max}}$$

le rapport entre l'écart-type du bruit de sortie et la valeur maximale de l'entrée est un rapport bruit/signal, qui vaut :

$$\frac{\sigma_s \text{dû aux bruits}}{s_{max} \text{due a l'entree}} \approx \frac{23}{41} . \frac{2^{-[N-1]}}{1} \approx 0{,}56 . \frac{1}{2^{N-1}}$$

*ce qui donne (c'était évident) exactement le même résultat qu'en **nombres entiers**...*

3.5 norme H_∞ d'un système, et utilisation

3.5.1 définition et signification

La norme H_∞ d'un système de fonction de transfert $G(z)$ est le maximum de sa réponse fréquentielle(dans le cas scalaire): $\|G(z)\|_{H\infty} = max|G(e^{j.\omega.Te})|$

signification

Le carré de la norme H_∞ est le taux maximum d'amplification, en puissance, du système :

$$\|G(z)\|_{H\infty}^2 = max \frac{\text{puissance } P_s \text{ de la sortie}}{\text{puissance } P_e \text{ de l'entrée}} = \frac{E\{s_n^2\}}{E\{e_n^2\}}$$

La norme H_∞ est donc le taux maximum d'amplification, en valeur efficace, du système:

$$\|G(z)\|_{H\infty} = max \frac{\text{valeur efficace } s_{eff} \text{ de la sortie}}{\text{valeur efficace } e_{eff} \text{ de l'entrée}} = \frac{\sqrt{E\{s_n^2\}}}{\sqrt{E\{e_n^2\}}}$$

calcul :

on dispose de méthodes efficaces pour calculer la norme H_∞ d'un système (hors du cadre de ce cours).

Le plus simple consiste à évaluer la réponse fréquentielle $G(e^{j.\omega.Te})$ sur un nombre suffisamment grand de pulsations ω , puis à relever le module maximum...

3.5.1 *calcul analytique de la norme H_∞ d'un système du premier ordre*

dans le cas d'un système du premier ordre, de fonction de transfert $G(z) = \frac{b_0 + b_1.z^{-1}}{1 + a_1.z^{-1}}$, la norme H_∞ est très facile à calculer, en raisonnant avec la transformée bilinéaire.

En effet, la fonction de transfert en *W* sera également du 1er ordre :

$$G(w) = \frac{n_0 + n_1.w}{d_0 + d_1.w}$$

Or le module de la réponse fréquentielle d'une fonction de transfert du premier ordre en w est maximum

soit en $w \to \infty$

soit en $w \to 0$

et on sait de plus que $z^{-1} = \frac{1-w}{1+w}$

donc le module de $G(z) = \frac{b_0 + b_1.z^{-1}}{1 + a_1.z^{-1}}$ est maximum ,

soit en $[w \to \infty] \Leftrightarrow [z^{-1} \to -1]$, et vaut alors $|G(z=-1)| = \left|\frac{b_0 - b_1}{1 - a_1}\right|$

soit en $[w \to 0] \Leftrightarrow [z^{-1} \to 1]$, et vaut alors $|G(z=1)| = \left|\frac{b_0 + b_1}{1 + a_1}\right|$

d'où on conclut que pour un système du premier ordre :

$$\|G\|_{H\infty} = max\left(\left|\frac{b_0 - b_1}{1 - a_1}\right|, \left|\frac{b_0 + b_1}{1 + a_1}\right|\right)$$

3.5.2 ***utilisation pour le calcul du facteur d'échelle de signal***

La norme H_∞ est très souvent employée, à la place de la norme 1, pour déterminer la plus grande valeur possible du facteur d'échelle λ

la norme 1 permet de calculer la plus grande valeur possible des variables internes, rarement atteinte en pratique. Elle permet de déterminer λ pour que cette plus grande valeur possible des variables internes ne dépasse pas l'échelle de codage. Cela conduit souvent à une valeur de λ inutilement petite, et donc à dégrader le rapport bruit / signal.

La norme H_∞ permet d'obtenir une valeur plus raisonnable de λ . On le détermine pour que la plus grande valeur efficace possible des variables internes ne dépasse pas l'échelle de codage. Théoriquement, il n'est plus du tout certain que les variables internes ne dépassent pas la pleine échelle de codage.
En pratique, les résultats sont le plus souvent tout à fait satisfaisants.

r***emarque 1***: l'emploi de la norme 1 ou de la norme H_∞ pour le calcul du facteur d'échelle λ dépend essentiellement du contexte dans lequel on se situe.
Pour un pilote automatique d'avion, il vaut mieux employer la norme 1 (c'est la seule norme absolument sûre)
Pour un filtre dans un lecteur MP3, on peut employer la norme H_∞ (les conséquences d'un éventuel dépassement d'échelle ne seront pas dramatiques)
***remarque 2*:** Dans le domaine de l'audio, beaucoup de personnes emploient carrément la norme 2 pour calculer le facteur d'échelle λ .
la norme 2 permet de calculer l'écart-type des variables internes, lorsque l'entrée est un bruit blanc de moyenne nulle, et d'écart-type égal à la pleine échelle de codage. Dans ce cas il faut s'attendre à des dépassements d'échelle au niveau des variables internes, et prévoir des systèmes pour que le comportement reste correct malgré tout.

3.5.3 Application à l'exemple considéré (calcul de λ ***avec la norme*** H_∞ ***)***

Le schéma d'analyse standard de la forme ***df2t*** figure 14 conduit à l'expression suivante de la transformée en z de la variable interne $x2_n$

$$x2(z) = \lambda . \frac{b_{0q} + b_{1q} . z^{-1}}{1 + a_{1q} . z^{-1}} . e(z) + \frac{z^{-1}}{1 + a_{1q} . z^{-1}} . b_x(z)$$

*E*n ignorant l'effet du bruit b_x , et en raisonnant sur les *valeurs efficaces*, on aura donc : $max\left(x2_{eff}\right) = \left\| \lambda . \frac{b_{0q} + b_{1q} . z^{-1}}{1 + a_{1q} . z^{-1}} \right\|_{H\infty} . e_{eff}$

raisonnement en virgule fixe :

On suppose $e_{eff} = 1$, et on veut $max\left(x2_{eff}\right) < 1$

on a $\left\| \lambda . \frac{b_{0q} + b_{1q}.z^{-1}}{1 + a_{1q}.z^{-1}} \right\|_{H\infty} = |\lambda| . max(|\frac{b_{0q} + b_{1q}}{1 + a_{1q}}|, |\frac{b_{0q} - b_{1q}}{1 - a_{1q}}|) \approx 27{,}4 . |\lambda|$

il faudra donc choisir $\lambda = 2^L$ tel que : $[\lambda = 2^L] < \underbrace{\left[max \frac{(x2_{eff})}{e_{eff}} \right]}_{=1} . \frac{1}{27{,}4}$

soit encore : $L < \log_2 \left(\frac{1}{27{,}4} \right) \approx -4{,}8$

on retiendra donc $L = -5$ => $\lambda = 2^{-5}$

Par rapport au cas précédent ($\lambda = 2^{-6}$ calculé avec la norme 1), on va diviser par 2 le bruit de sortie.
Par contre il faudra être prudent, et vérifier en simulation que l'on ne génère pas de dépassement sur la variable interne $x2_n$

3.6 *scaling* et *analyse*, extension de la notion de *gain relatif de bruit*

les 'normes' sont employées lors du calcul des filtres pour 2 usages très différents

3.6.1 *phase de scaling, ou de mise à l'échelle*

Les 'normes' permettent de déterminer la valeur maximale du facteur d'échelle λ *de façon à ce que les variables internes occupent correctement l'échelle de codage: C'est la phase de* ***scaling****, ou de* ***mise à l'échelle****, du filtre.*

le scaling peut être effectué, suivant le contexte, avec n'importe quelle norme

- ***la norme 1***, si on ne veut courir absolument aucun risque, mais qui conduit à un rapport bruit / signal assez élevé
- ***la norme*** H_∞ , qui permet d'améliorer le rapport bruit/signal, avec un risque très modéré de dépassement d'échelle
- ***la norme 2****,* qui améliore franchement le rapport bruit/signal, au prix de dépassements d'échelle assez fréquents, dont il faudra tenir compte lors du codage {en saturant les variables internes => dans les phases de saturation, le filtre ne fonctionnera plus correctement}

3.6.2 phase d'analyse

Les 'normes' permettent également de déterminer, pour une valeur donnée du facteur d'échelle λ *,*

1- le ***niveau de bruit de sortie*** *(partie de la sortie qui ne dépend que des bruits)*

2- le ***niveau de sortie utile****(partie de la sortie qui ne dépend que de l'entrée)*

C'est la phase ***d'analyse*** *des performances du filtre.*

Cette analyse peut être effectuée avec n'importe quelle norme

- ***la norme 1***, si le niveau est la ***valeur absolue maximale*** *du signal*
- ***la norme*** H_∞ , si le niveau est la ***valeur efficace maximale*** *du signal*
- ***la norme 2****,* si le niveau est l'***écart-type*** du signal, *lorsque les* ***entrées sont supposées blanches et indépendantes entre elles***

3.6.3 *rapport bruit / signal* NSR_j^i

le rapport bruit sur signal { *NSR* pour *Noise Signal Ratio*) correspondant est défini par

$$NSR = \frac{\text{niveau de bruit (en norme 1,2, ou } H_\infty)}{\text{niveau de signal utile (en norme 1,2, ou } H_\infty)} \quad ,$$

Les normes pour évaluer le *niveau de bruit e*t le *niveau utile* peuvent être différentes. Il faut donc préciser dans le rapport bruit sur signal *NSR*, quelles sont les normes employées, en le notant NSR_j^i .

$i = 1,2,\mathrm{H}_\infty$ précise la norme employée pour évaluer le *niveau de bruit*

$j = 1,2,\mathrm{H}_\infty$ précise la norme employée pour évaluer le *niveau utile*

exemple

NSR_1^2 est le *rapport bruit sur signal* lorsque

le niveau de bruit de sortie est évalué avec la *norme 2*,

le niveau de signal utile est évalué avec la *norme 1*

ce rapport représente donc (voir 3.6.2) le rapport entre

l'écart-type du bruit de sortie, lorsque les bruits sont blancs et indépendants

et la valeur absolue maximale du signal utile de sortie

3.6.4 *gain relatif de bruit* RNG_j^i *(**R**elative **N**oise **G**ain)*

Le rapport bruit sur signal NSR_j^i peut toujours s'écrire, en fonction du nombre N de bits de codage, sous la forme : $NSR_j^i \approx \frac{1}{2^{N-1}} . RNG_j^i$

La quantité RNG_j^i représente un taux d'amplification intrinsèque du niveau de bruit en *norme i*, relativement au niveau de signal en *norme j*.

Pour cette raison elle est appelée *gain relatif de bruit ,* ou *Relative Noise Gain* en anglais

Cette quantité permet de comparer directement, sans se préoccuper du nombre N de bits de codage, les qualités intrinsèques de telle ou telle implémentation d'un filtre (forme directe 1,2,...)

4 EXERCICES

4.1 avertissement

dans cette partie, vous allez devoir concevoir complètement un filtre numérique, sous forme directe 2 (***df2***), qui servira de base au travail de la séance de travaux pratiques correspondante. Cela signifie que vous devez traiter tous les exercices, et pour cela il est probable que vous deviez lire et comprendre tout ce qui précède (parties 1,2, et 3). Le travail global est évalué à environ 10 heures de travail personnel, ne pas le faire au dernier moment!... Vous pouvez travailler en équipe, mais on attend une copie manuscrite (pas de traitement de texte!..) par personne. Ne vous contentez pas de recopier le travail du voisin, on vérifiera en TP que vous avez effectivement fourni l'effort demandé, sinon la note sera revue en conséquence.

4.2 Cahier des charges

On veut écrire un programme en langage C correspondant à l'implémentation sous forme directe 2 (***df2***) d'un filtre numérique passe-bas du 1er ordre.

La fréquence d'échantillonnage est $fe = 8000\mathrm{Hz}$.

Le filtre passe-bas doit avoir un gain statique égal à1, et une fréquence de coupure à -3db $f_0 = 100\mathrm{Hz}$

En raisonnant dans le plan W, la fonction de transfert du filtre est donc :

$F(w)=\dfrac{1}{1+\dfrac{w}{v_0}}$, avec $v_0=tg\left(\pi.\dfrac{f_0}{fe}\right)$ (machines en radians svp)

4.3 ***calcul et vérification de la fonction de transfert en z***

-déterminer à partir de l'expression de $F(w)$ la fonction de transfert en z correspondante, sous la forme $F(z)=\dfrac{b_0+b_1.z^{-1}}{1+a_1.z^{-1}}$

donner la valeur numérique des coefficients, avec une précision suffisante pour que les 2 équations suivantes soient vérifiées avec une précision meilleure que 10^{-5}

équation 1- $F(z=1)=F(w=0)=1$,

équation 2- $F(z=-1)=F(w\to\infty)=0$

4.4 exploitation du schéma d'analyse standard

le schéma d'analyse standard de la forme directe 2 (avec facteur d'échelle).est donné figure 15. (vous le retrouverez à la fin des exercices)

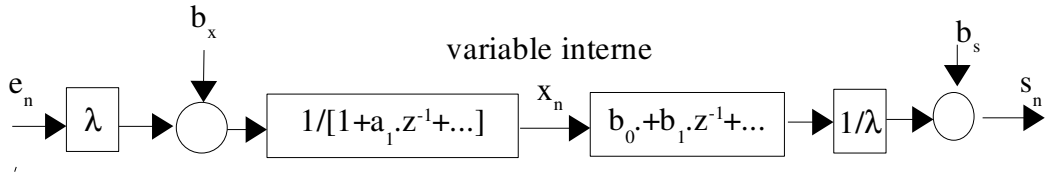

*figure 15: schéma d'analyse standard de la forme directe 2 (**df2**)*

en raisonnant sur ce schéma d'analyse

- déterminer l'expression de la transformée en z $x(z)$ de la variable interne x_n , sous la forme

$x(z)=F_{e\_x}(z).e(z)+F_{bx\_x}(z).b_x(z)$

- déterminer l'expression de la transformée en z $s(z)$ de la sortie s_n , sous la forme

$s(z)=F_{e\_s}(z).e(z)+F_{bx\_s}(z).b_x(z)+F_{bs\_s}(z).b_s(z)$

4.5 ***calcul des normes***

compléter le tableau suivant, correspondant au calcul numérique des différentes normes des fonctions de transfert (certaines normes dépendent de $|\lambda|$, à mettre en facteur des valeurs numériques dans le tableau)

$F(z)$	$\|F(z)\|_1$	$\|F(z)\|_2$	$\|F(z)\|_{H\infty}$
$F_{e\_s}(z)$			
$F_{e\_x}(z)$			
$F_{bx\_s}(z)$			
$F_{bs\_s}(z)$			

4.6 ***Scaling, calcul de*** $\lambda=2^L$

on suppose que le module maximum de l'entrée est égal à 1 (raisonnement en virgule fixe : $max\{|e_n|\}=1$), et on ne prend pas en compte l'effet du bruit b_x sur la variable interne x_n

- déterminer alors (en précisant la fonction de transfert et la norme utilisée), le module maximum de la variable interne x_n , en fonction de λ

$max(|x_n|)=??$

on pose $\lambda=2^L$, avec L entier

- déterminer dans ce cas la plus grande valeur de λ , et la valeur correspondante de L , pour que $max(|x_n|)<1$ (raisonnement en virgule fixe)
- pour cette valeur de $\lambda=2^L$, en déduire la valeur absolue maximum de la variable interne x_n : $max(|x_n|)=??$

4.7 analyse, calcul du bruit de sortie

dans cette partie, vous allez déterminer les caractéristiques du bruit de sortie de votre filtre, en fonction du nombre N de bits de codage. On supposera que les coefficients quantifiés sont suffisamment proches des coefficients réels pour pouvoir négliger cet effet.

4.7.1 Caractéristiques des bruits

on s'arrangera à programmer le filtre pour que le bruit de quantification b_x soit un bruit de quantification par arrondi, et que le bruit de quantification b_s soit un bruit de quantification par troncature.

-En raisonnant en virgule fixe,

donner les caractéristiques suivantes des bruits, en fonction du nombre N de bits de codage

- valeurs absolues maximales de b_x et b_s
- valeurs efficaces de b_x et b_s (supposés à densité de proba. uniforme)
- écarts-types de de b_x et b_s (supposés à densité de proba. uniforme)

4.7.2 Caractéristiques du bruit de sortie

-donner l'expression de la sortie $s(z)$ due uniquement aux bruits (lorsque $e(z)=0$). $s(z)$ représente alors la transformée en z du *bruit de sortie*

- en déduire, en fonction de N et en précisant à chaque fois la norme employée

1- la valeur absolue maximale du *bruit de sortie*

2- la valeur efficace maximale du *bruit de sortie*

3- l'écart type du bruit de sortie (lorsque b_x et b_s sont des bruits blancs indépendants)

Application numérique (le filtre sera codé en arithmétique 16/32 bits => $N=16$

4.7.3 Caractéristiques de la sortie utile

-donner l'expression de la sortie $s(z)$ due uniquement à l'entrée (lorsque $b_x(z)=b_s(z)=0$). $s(z)$ représente alors la transformée en z de la *sortie utile*

- en déduire, en précisant à chaque fois la norme employée

1- la valeur absolue maximale de la *sortie utile*

2- la valeur efficace maximale de la *sortie utile* (en supposant que la valeur efficace de l'entrée est égale à 1)

3- l'écart type de la *sortie utile* (lorsque l'entrée est un bruit blanc d'écart-type égal à 1)

4.7.4 rapport bruit sur signal NSR_j^i

donner la valeur numérique des rapports bruit sur signal NSR_1^2 , $NSR_1^{H\infty}$ et dire si ces valeurs vous semblent satisfaisantes...

4.8 codage du filtre

En ce qui concerne l'enseignant, les performances lui paraissent suffisamment correctes pour passer à la phase de codage, en arithmétique 16/32 bits, donc c'est parti...

4.8.1 quantification des coefficients

compléter le tableau suivant, le nombre de bits de codage étant $N=16$ (veiller à indiquer au moins 5 décimales pour chacun des coefficients réels)

coeff réel c	*coeff entier* c_N	décalage L_c	*coeff quantifié* $c_q=2^{-Lc}.c_N$
$b_0\approx$	$b_{0N}=$	$L_{b0}=$	$b_{0q}\approx$
$b_1\approx$	$b_{1N}=$	$L_{b1}=$	$b_{1q}\approx$
$-a_1\approx$	$[-a_1]_N=$	$L_{a1}=$	$[-a_1]_q\approx$

4.8.2 Schéma naïf

représenter le schéma naïf d'implémentation de votre filtre, sous forme directe 2 ***df2***, en n'oubliant pas le facteur d'échelle $\lambda=2^L$, et sans optimiser quoi que ce soit

4.8.3 Schéma optimisé

en appliquant les règles d'optimisation présentées pour l'exemple traité par le prof., en déduire le schéma d'implémentation optimisé.

4.8.4 codage

en vous inspirant de la façon dont le prof. a écrit son programme en *c* pour l'exemple traité, écrire le programme en c correspondant à ce schéma optimisé {ne pas oublier d'arrondir le bruit en sortie du premier décalage commun}

4.8.5 schéma d'analyse (et la boucle est bouclée)

-représenter le schéma d'analyse correspondant à votre schéma d'implémentation optimisé.

-montrer que ce schéma d'analyse correspond bien au schéma standard d'analyse de la forme ***df2*** , représenté figure 15 (on peut effectivement négliger les bruits associés aux multiplications par $b_{0q}, b_{1q}, [-a_{1q}]$, une fois que l'on a optimisé le schéma).