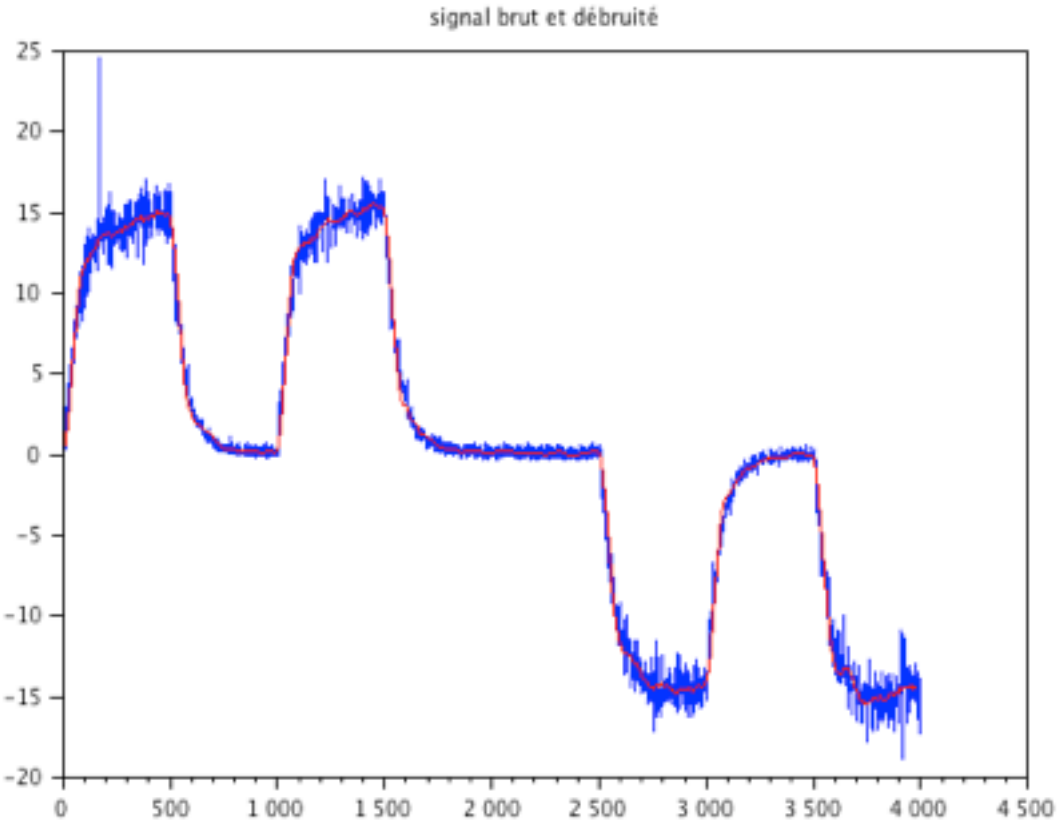
Débruitage temps réel avec une transformée en ondelettes

Manuel utilisateur du logiciel DebruitageGauche.sci

1. Résultat

Voici le résultat du test sur le signal représentant le contrôle d'attitude du satellite :



Pour le reconstituer, il convient de procéder ainsi :

- aller dans la directorie où se trouvent GR1.scikab, DebruitageGauche.sci en TestDebruitage.sce.
- exécuter DebruitageGauche.sci (exec('DebruitageGauche.sci')) pour mettre les fonctions à disposition de la session
- exécuter TestDebruitage.sce (exec('TestDebruitage.sce') pour appeleler la fonction DebruitageGauche
- attendre quelques minutes que le script s'exécute

Vous aurez ainsi le signal brut en bleu et le signal débruité en rouge.

2. Fonction DebruitageGauche(U,scale,delay,threshold)

C'est la seule fonction dans laquelle vous aurez à entrer.

En effet, elle est définie pour a totalité d'un signal U et non comme une boite xcos définissant l'évolution en temps réel de l'algorithme.

La fonction se présente ainsi :

```
function V=DebruitageGauche(U, scale, delay, threshold)
   // Débruitage par seuillage d'un signal u utilisant la transformée en ondelettes
   // par moyennisaton sur le demi-axe réel négatif (filtre presque causal avec un dÃ©lai)
    W=2^(scale-1)*6+delay; // largeur de la fenêtre
  // Initialisation
   x=Initialize(W);
   for n=2:delay
      // Mise Ã  jour de l'état sans calcul de la sortie
      u=U(n-1);
      x=Update(x,u);
   end
   for n=(delay+1):length(U)
      // Mise Ã  jour de l'état
      u=U(n-1);
      x=Update(x,u);
      // calcul de la sortie
      u=U(n);
      v=Outputs(x,u,scale, delay,threshold);
      V(n-delay)=v;
   end
endfunction
```

La structure de l'algorithme temps réel est une forme d'état, avec les éléments suivants :

- initialisation de l'état x, avec la fonction Initialize(W)
- mises à jour de l'état x, en fonction de l'état précédent et de l'entrée u à l'instant précédent (c'est normalement sous-entendu dans xcos) : fonction Update(x,u)
- élaboration de la sortie v en fonction de l'état x courant et de l'entrée u courante : fonction Outputs(x,u)

Dans notre cas, l'état x est la suite des W-1 entrées jusqu'à 2 échantillons avant la fin. L'évolution de l'état est donc un simple registre à décalage gauche.

C'est dans le calcul de la sortie que l'on fait la transformée en ondelettes de l'état, suivis du seuillage des détail (élimination des détails supérieurs au seuil 'threshold') et de la transformée en ondelettes inverse.

On termine par un retard 'delay', c'est-à-dire que la sortie est le résultat de la transformée en ondelettes inverse à un nombre d'échantillons 'delay' avant la fin.

C'est la nécessité de pouvoir calculer une transformée en ondelettes multirésolution à un nombre d'échelles 'scale' , puis de retarder de 'delay',qui détermine la largeur W de la fenêtre à $6 * 2^{scale-1} + delay$.

Il n'est pas nécessaire de comprendre la signification de cette formule pour utiliser le logiciel. Il suffit de l'appliquer correctement pour en déduire la longueur W-1 du vecteur d'état.

3. Utilisateur final

Pour utiliser la boîte xcos que vous allez déduire de mon logiciel, il conviendra de l'alimenter avec des données temps réel de type scalaire. Ce sera la variable u.

Le logiciel ne fonctionnera en régime permanent que lorsqu'un état complet aura été fourni, plus une valeur de u, soir W = $6 * 2^{scale-1} + delay$ valeurs. Cela représente 212 valeur pour une échelle de 6 et un délai de 20.

Justement, l'échelle 'xcale' doit être laissée à 6 pour que le logiciel donne tout son potentiel.

En ce qui concerne le retard 'delay' et le seuil 'threshold', ils doivent être réglés en fonction des caractéristiques du bruit de mesure (les erreurs variant rapidement).

Le retard 'delay' doit être environ de la période du bruit. Cette période se visualise en faisant un plot des premiers éléments du signal.

Par exemple, pour le satellite, en faisant 'plot(U(1:100))', puis 'xgrid', on voit que les oscillations ont une période de 20 échantillons environ :

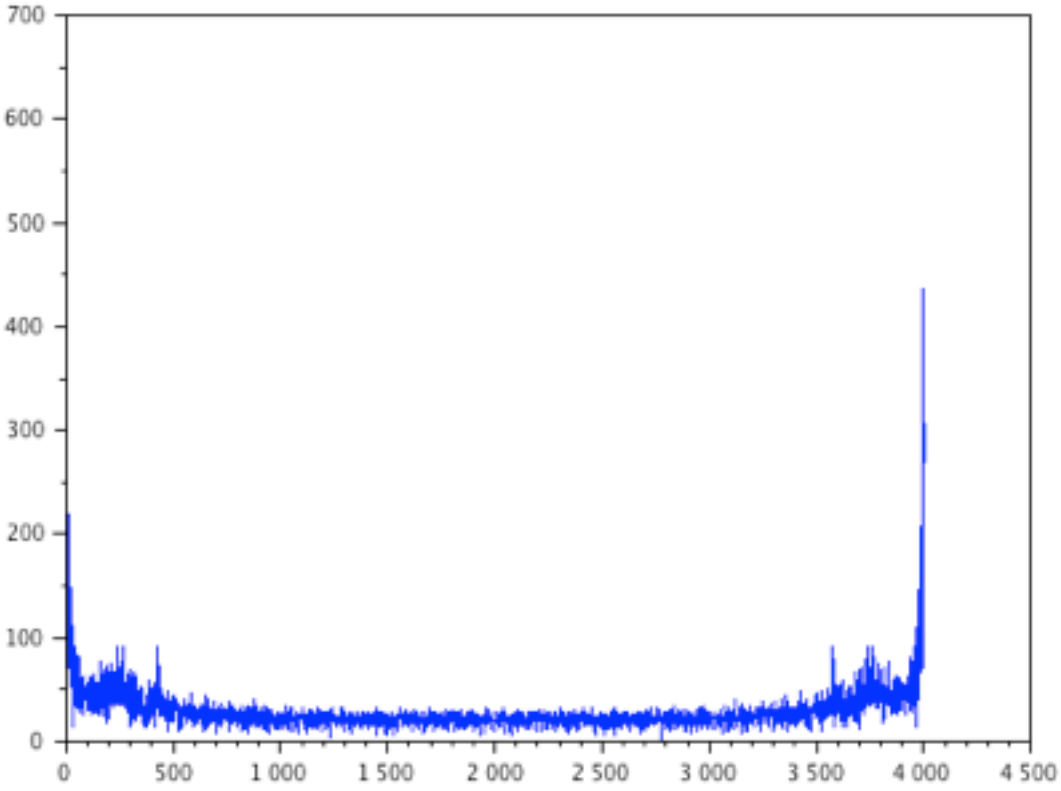
On fixe donc le retard à 'delay=20'.

Quant au seuil, il est fixé à 3 ou 4 fois le niveau de bruit. Celui-ci se détermine en tapant les commandes suivantes sur le signal d'entrée U composé des entrées scalaires u successives :

```
--> W=fft(U.^2);
--> W=sqrt(sbs(W));
--> figure;
--> plot(W(20:$-20))
```

On obtient le spectre, dont il convient d'extraire la parie centrale.

Par exemple, pour le satellite, on obtient:

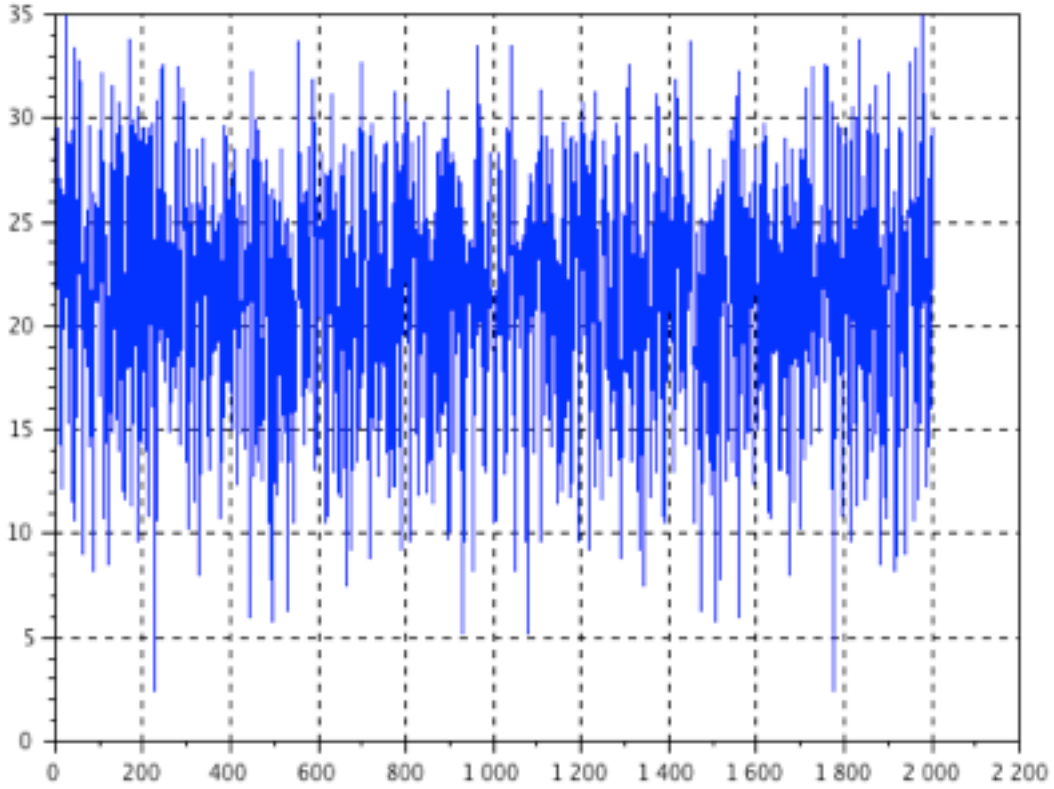


La partie où le spectre est plat est donc entre 1000 et 3000.

On ferme donc la figure et on tape :

```
--> plot(W(1000:3000)
--> xgrid
```

On obtient la figure suivante :



On voit que le niveau de bruit est d'environ 25 à 30.

Un seuil 'threshold=100' convient donc bien.

Il est à noter que la valeur du seuil n'est pas très sensible. Seul l'ordre de grandeur importe.

4. Synthèse

La fonction DebruitageGauche(U,scale,delay,threshold) doit être adaptée pour en faire la fonction d'activation d'une boite xcos, à appeler LeftWaveDenoise.sci sur le modèle de la S-fonction Matlab de François Chaplais.

Dans les entrées de cette fonction, il convient de remplacer la série d'entrées U par l'état x et une entrée scalaire u.

Les fonctions Initialize, Update et Ouputs peuvent a priori être utilisées telles quelles pour initialiser l'état, mettre à jour l'état et générer les sorties.

Le calcul de la largeur de fenêtre W, doit être fait comme indiqué.

Les autres fonctions doivent impérativement être laissées en l'état.

Par ailleurs, 'scale' doit être égal à 6, et 'delay' et 'threshold' sont à régler comm indiqué au § précédent.