Vérification formelle et filtres numériques

Diane Gallois-Wong

Encadrants : Sylvie Boldo et Thibault Hilaire

Université Paris-Sud, LRI (Orsay)

EJCIM - mars 2018

Vérification formelle

Erreurs humaines dans les programmes informatiques

$\rightsquigarrow$ explosion de la fusée Ariane 5 et beaucoup d'autres exemples

Vérification formelle / Méthodes formelles

Raisonner rigoureusement, à l'aide d'une logique, sur des programmes informatiques vus comme des objets mathématiques.

$\rightsquigarrow$ lemmes, théorèmes sur le comportement des programmes

J'utilise l'assistant de preuve **Coq** 

- Énoncer les définitions, (beaucoup de) lemmes, les théorèmes
- Guider les preuves : détailler toutes les étapes, même celles qui semblent évidentes

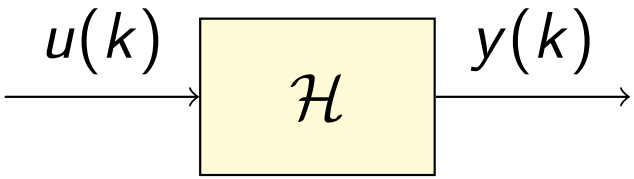
Filtres numériques

Domaines : traitement du signal, contrôle-commande

Applications : audio, télécommunications, automobile, robotique, aéronautique, aérospatial...

Définition : filtres numériques

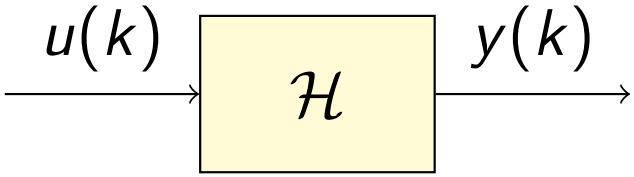
Algorithmes qui transforment des signaux numériques.



Signaux à temps discret :

$u, y : \mathbb{Z} \to \mathbb{R}$

Filtres numériques



Signaux à temps discret :
$u, y : \mathbb{Z} \to \mathbb{R}$

$y(k)$ dépend de $u(k)$, $u(k-1)$, ... , $u(0)$

Filtre linéaire $\rightsquigarrow$ additions, multiplications par des constantes

Exemple : $$y(k) = \sum_{i=0}^{n} b_i u(k-i) - \sum_{i=1}^{n} a_i y(k-i)$$

n : *ordre* du filtre, $(a_i)_{1 \leq i \leq n}$ et $(b_i)_{0 \leq i \leq n}$: *coefficients* du filtre

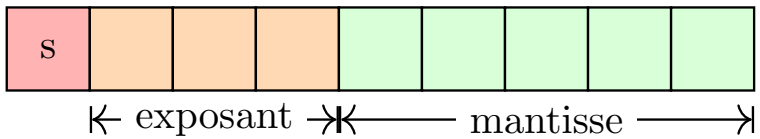
Implémentation : précision finie

Modèle : nombres réels

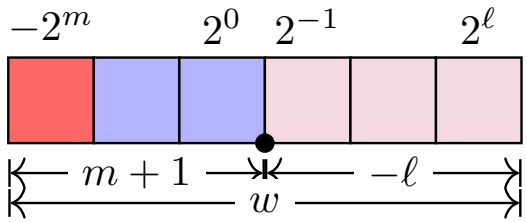
↓

Implémentation : arithmétique en virgule flottante ou virgule fixe

Virgule flottante



Virgule fixe



- erreurs d'arrondi à chaque opération
- overflow quand on dépasse la plus grande valeur représentable

Implémentation : précision finie

Modèle : nombres réels

↓

Implémentation : arithmétique en virgule flottante ou virgule fixe

Exemple : $$y(k) = \sum_{i=0}^{n} b_i u(k-i) - \sum_{i=1}^{n} a_i y(k-i)$$

$$y^*(k) \approx \sum_{i=0}^{n} b_i^* u(k-i) - \sum_{i=1}^{n} a_i^* y^*(k-i)$$

- erreurs d'arrondi
  - coefficients
  - chaque addition, multiplication
  - propagation des erreurs et risque d'**accumulation**
- overflow quand on dépasse la plus grande valeur représentable

Objectif de ma thèse et contributions

Objectif : **preuve formelle** de l'analyse des erreurs d'arrondi dans les filtres numériques

afin de pouvoir garantir qu'une implémentation en précision finie est suffisamment proche d'un modèle utilisant les nombres réels.

Première étape : formalisation des filtres avec des nombres réels

Preuve en Coq  de trois résultats classiques sur les filtres :

- Représentation unificatrice des filtres
- Théorème sur la propagation des erreurs d'arrondi
- Théorème du *Worst-Case Peak-Gain*

Une représentation pour les décrire toutes

Problème : les filtres sont définis sous diverses formes.

$$y(k) = \sum_{i=0}^{n} b_i u(k-i) - \sum_{i=1}^{n} a_i y(k-i)$$ est juste une des possibilités !

Une représentation pour les décrire toutes

Problème : les filtres sont définis sous diverses formes.

$y(k) = \sum_{i=0}^{n} b_i u(k-i) - \sum_{i=1}^{n} a_i y(k-i)$ est juste une des possibilités !

Les ramener toutes à la même représentation, appelée *State-Space* :

$$\begin{cases} \mathbf{x}(k+1) &= A\mathbf{x}(k) + Bu(k) \\ y(k) &= C\mathbf{x}(k) + Du(k) \end{cases}$$

$\mathbf{x} : \mathbb{Z} \to \mathbb{R}^m$: *vecteur d'état* qui contient des valeurs auxiliaires

Une représentation pour les décrire toutes

Problème : les filtres sont définis sous diverses formes.

$y(k) = \sum_{i=0}^{n} b_i u(k-i) - \sum_{i=1}^{n} a_i y(k-i)$ est juste une des possibilités !

Les ramener toutes à la même représentation, appelée *State-Space* :

$$\begin{cases} \mathbf{x}(k+1) &= A\mathbf{x}(k) + Bu(k) \\ y(k) &= C\mathbf{x}(k) + Du(k) \end{cases}$$

$\mathbf{x} : \mathbb{Z} \to \mathbb{R}^m$: *vecteur d'état* qui contient des valeurs auxiliaires

Pour plusieurs représentations usuelles, formalisation en Coq de cette transformation et preuve que le résultat décrit le même filtre

Propagation des erreurs d'arrondi

Modèle (nombres réels) :

$$\mathcal{H} \quad \begin{cases} \mathbf{x}(k+1) & = A\mathbf{x}(k) + Bu(k) \\ y(k) & = C\mathbf{x}(k) + Du(k) \end{cases}$$

Implémentation (précision finie) :

$$\mathcal{H}^* \quad \begin{cases} \mathbf{x}^*(k+1) & = (A \otimes \mathbf{x}^*(k)) \oplus (B \otimes u(k)) \\ y^*(k) & = (C \otimes \mathbf{x}^*(k)) \oplus (D \otimes u(k)) \end{cases}$$

Problème : propagation des erreurs

$$\mathbf{x}^*(k+1) \hookleftarrow \mathbf{x}^*(k) \hookleftarrow \mathbf{x}^*(k-1) \hookleftarrow \ldots$$

Propagation des erreurs d'arrondi

Modèle (nombres réels) :

$$\mathcal{H} \quad \begin{cases} \mathbf{x}(k+1) & = A\mathbf{x}(k) + Bu(k) \\ y(k) & = C\mathbf{x}(k) + Du(k) \end{cases}$$

Implémentation (précision finie) :

$$\mathcal{H}^* \quad \begin{cases} \mathbf{x}^*(k+1) & = A\mathbf{x}^*(k) + Bu(k) + \varepsilon_x(k) \\ y^*(k) & = C\mathbf{x}^*(k) + Du(k) + \varepsilon_y(k) \end{cases}$$

$\varepsilon_x(k)$, $\varepsilon_y(k)$: erreur pour chaque ligne, inconnue mais bornée selon le format utilisé

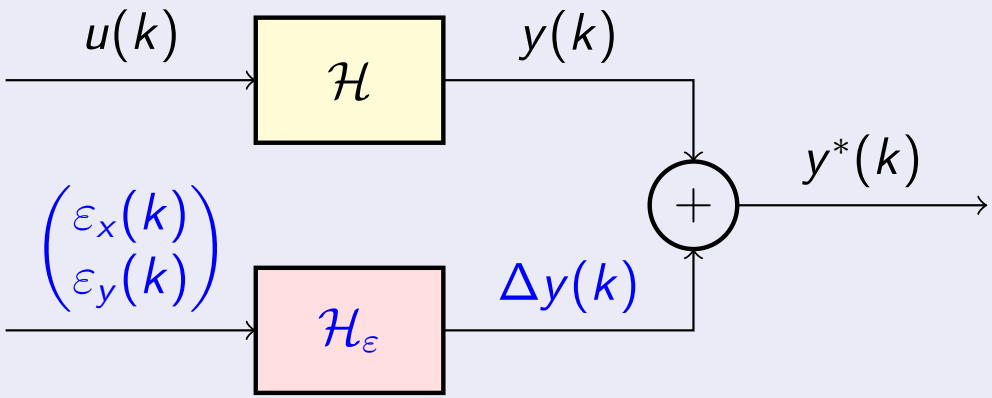
Problème : propagation des erreurs

$$\mathbf{x}^*(k+1) \hookleftarrow \mathbf{x}^*(k) \hookleftarrow \mathbf{x}^*(k-1) \hookleftarrow \ldots$$

Propagation des erreurs d'arrondi

Théorème

On sait calculer un filtre d'erreur $\mathcal{H}_\varepsilon$ tel que



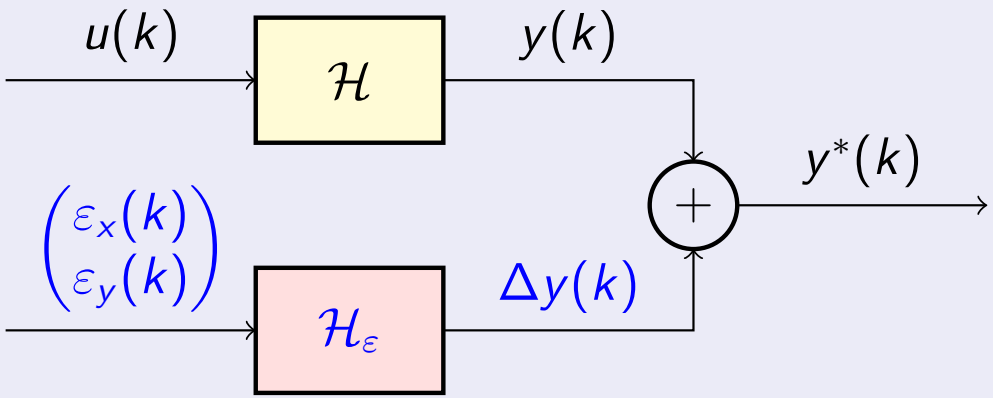
$y(k)$: sortie du modèle $y^*(k)$: sortie de l'implémentation

$\Delta y(k)$: erreur finale qui tient compte de la propagation

Propagation des erreurs d'arrondi

Théorème

On sait calculer un filtre d'erreur $\mathcal{H}_\varepsilon$ tel que



✓ Prouvé

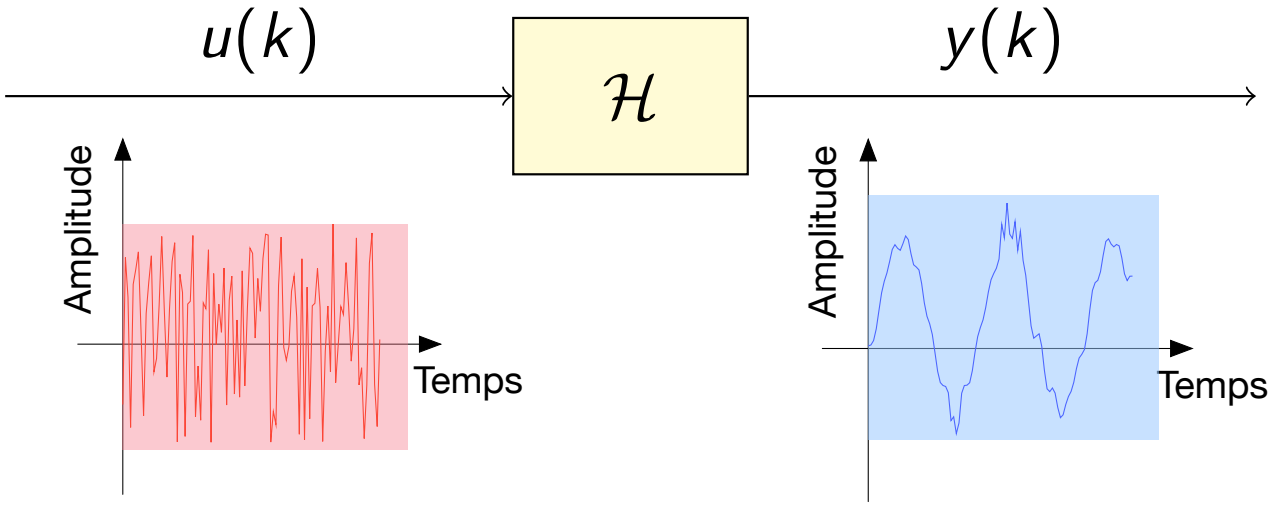
$y(k)$: sortie du modèle $y^*(k)$: sortie de l'implémentation
$\Delta y(k)$: erreur finale qui tient compte de la propagation

Worst-Case Peak-Gain

Théorème

Pour un filtre donné $\mathcal{H}$, on peut déterminer $wcpg(\mathcal{H})$ tel que

$$\forall u, \forall M, \quad \forall k, |u(k)| \leq M \implies \forall k, |y(k)| \leq wcpg(\mathcal{H}) \times M$$



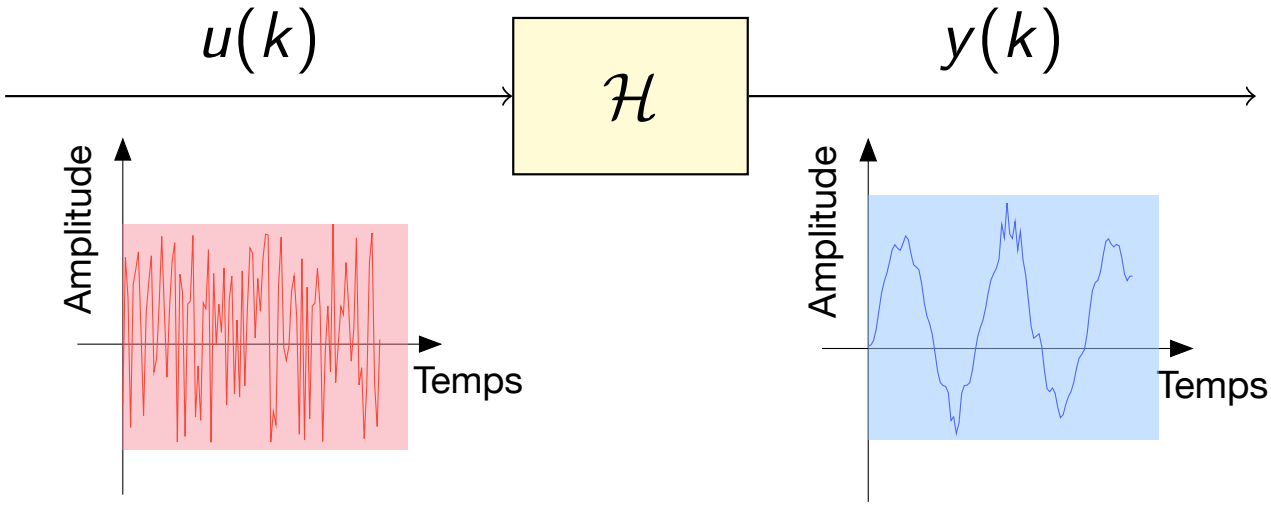
Worst-Case Peak-Gain

Théorème

Pour un filtre donné $\mathcal{H}$, on peut déterminer $wcpg(\mathcal{H})$ tel que

$$\forall u, \forall M, \quad \forall k, |u(k)| \leq M \implies \forall k, |y(k)| \leq wcpg(\mathcal{H}) \times M$$

✓ Prouvé



Preuve formelle de l'analyse des erreurs d'arrondi dans l'implémentation d'un filtre numérique

Formalisation utilisant seulement des nombres réels :

- ✓ Représentation unificatrice des filtres : le *State-Space*
- ✓ Propagation des erreurs d'arrondi : *filtre d'erreur*
- ✓ *Worst-Case Peak-Gain*
- Transformer plus de représentations usuelles en State-Space
- Prouver le calcul approché de $wcpg(\mathcal{H})$ (somme infinie)

Formalisation utilisant des arithmétiques en **précision finie** :

- **Formaliser** l'arithmétique en virgule fixe en **complément à 2**
- Borner les $\varepsilon_x(k)$, $\varepsilon_y(k)$ en fonction de l'arithmétique utilisée
- Prendre en compte la précision finie dans le calcul de $wcpg(\mathcal{H})$