

Fonctions MATLAB Utiles dans le domaine du codage

M. Belkasmi

ENSIAS 2011-2012

Fonctions pour codage convolutionnel



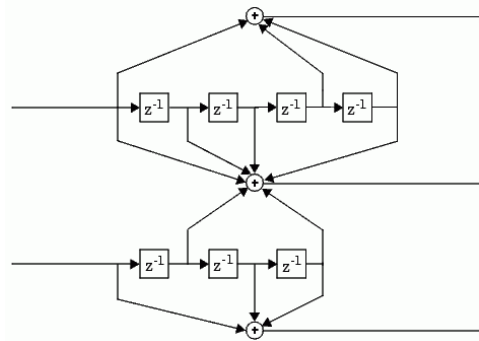
- Fonction MATLAB pour réaliser codage CV:
 - code = `convenc`(msg,trellis) 
'msg' représente le vecteur binaire en entrée
 - treillis=`poly2trellis`(LongCon,Generateur)
- treillis=`poly2trellis`(LongCon,Generateur) 
 - Converti une représentation polynomiale d'un codeur cv (feedforward) en une structure de treillis
- NB: paramètre qu'on peut ajouter 'INIT_STATE'
 - État initial= INIT_STATE.
 - code = `convenc`(msg,trellis,Init_state)

Diagramme d'un registre à décalage



Exemple d'un code de taux 2/3, l'entrée du codeur est un vecteur de longueur k , et la sortie est vecteur de longueur n . Ici $k=2$, $n=3$.

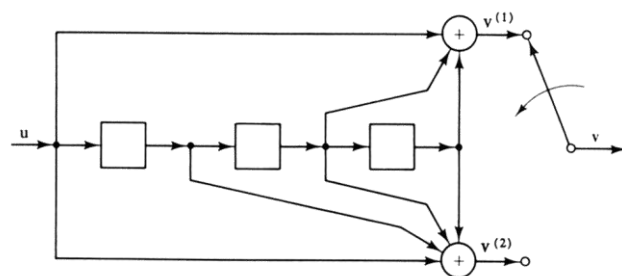
La génération de la structure Treillis

- Avant le codage on aura besoin du treillis
 - `treillis=poly2treillis(LongCon,Generateur)`
 - **LongCon** est un vecteur $1 \times k$ précisant le délai imposé à chaque flux parmi les k flux en entrée du codeur.
 - **Generateur** est une matrice $k \times n$ de nombres en **octal** spécifiant les connexions entrées/ sorties.

La structure treillis en MATLAB

- Un treillis Matlab est représenté par une structure avec les champs suivants:
 - numInputSymbols, (nombre de symboles en entrée)
 - numOutputSymbols, (nombre de symboles en sortie)
 - numStates, (nombre d'états)
 - nextStates, (matrice de l'état suivant)
 - outputs, (matrice de sortie du codeur)

Exemple 2 :



Ici "entree" $k=1$, "sortie" $n=2$;

LongCon est un vecteur $1 \times k = 1 \times 1$: [4];

Generateur est une matrice $k \times n = 1 \times 2$: [13,17]

$$4 = 3 + 1, m + 1$$


13(octal)=1011 (binaire)=11(decimal)

17 (octal)=1111(binaire)=15(decimal)

Exemple 2 (suite) :


- `treillis=poly2trellis(LongCon, Generateur)`
 - `Treillis=poly2trellis([4],[13,17]);`
treillis =
 - numInputSymbols: 2
 - numOutputSymbols: 4
 - numStates: 8
 - nextStates: [8x2 double]
 - outputs: [8x2 double]
- Avec la séquence binaire en entrée ,
 - `Msg=[1 0 1 1 1];`
- La séquence codée en sortie du codeur:
 - `seqcodee=convenc(msg, treillis);`
`seqcodee = 1 1 0 1 0 0 0 1 0 1`
- **NB: Le résultat ici est un vecteur de longueur 10 et non pas 14!**

Décodage cv: algorithme Viterbi

- `decodee = vitdec(seqrec,treillis,tblen,opmode,dectype);`
 - **seqrec** : séquence reçue en entrée du décodeur
 - **treillis** : le même treillis que celui utilisé dans le codage
 - **tblen** : longueur de traceback
 - **opmode** : 
 - “trunc”- démarre a l’etat plein-0. Traceback à partir de l’état de meilleur métrique, i.e , pas de délai dans la sortie
 - “term” – Début et fin dans l’état plein-0. Traceback à partir l’état plein-0
 - “cont” – comme pour “trunc”, mais avec un délai de “tblen” symboles en sortie.

Décodage cv: algorithme Viterbi

– dectype :

- “unquant” – Symboles en entrée avec 1 représente le 0 logique, -1 représente le 1 logique.
- “Hard” – bits en entrée avec des valeurs 0 et 1,  Décodage hard-decision.
- “soft” – Décodage soft decision, “nsdec” à ajouter pour un décodage soft-decision.
Symboles en entrée sont dans l'intervalle $[0: 2^{\text{nsdec}}-1]$.

- D'autres paramètres peuvent aussi être ajouté dans cette fonction comme “finalstate”, “finalinput”, etc.

Décodage Soft-decision

- `decodee = vitdec(seqrec,treillis,tblen,opmode,'soft',nsdec);`
 - Elle décode le vecteur seqrec en utilisant un décodeur soft-decision.
 - Symboles en entrée consiste en des entiers compris entre 0 et $2^{\text{nsdec}}-1$, où 0 représente le 0 plus fiable et $2^{\text{nsdec}}-1$ représente le 1 le plus fiable.
 - Des symboles en entrée peuvent être transformé pour la soft decision en utilisant la fonction suivante :
 - `[index,code] = quantiz(info, partition, codebook);`
 - Par exemple:
`[x,qcode] = quantiz(1-2*ncode,[-.75 -.5 -.25
0 .25 .5 .75], [7 6 5 4 3 2 1 0])`
- qcode = valeurs comprises entre 0 et 2^3-1 (métriques)

Exemple: Décodage hard-decision

■ Rappel codage cv:

- msg=[1 0 1 1 1];
- seqcod=convenc(msg, treillis);
- seqcod = 1 1 0 1 0 0 0 1 0
1

■ En utilisant un décodage hard decision :

- Decodee=vitdec(seqcod, treillis, 1, 'cont', 'hard')
- Decodee= [0 1 0 1 1]
- Decodee=vitdec(seqcod, treillis, 1, 'trunc', 'hard')
- Decodee=[1 0 1 1 1]

Tlben=1

Aucun Delai

Exemple: Décodage soft-decision

- msg=[1 0 1 1 1];
- seqcod=convenc(msg, treillis);
- seqcod = 1 1 0 1 0 0 0 1 0 1

■ En utilisant un décodage soft decision :

- seqrec=quantiz(seqcod, [0.125, 0.25, 0.375, 0.5, 0.625, 0.75, 0.875], [0:7]);
- seqrec= [7 7 0 7 0 0 0 7 0 7]
- Decodee=vitdec(seqrec, treillis, 1, 'cont', 'soft', 3)
- Decodee= [0 1 0 1 1]
- Decodee=vitdec(seqrec, treillis, 1, 'trunc', 'soft', 3)
- Decodee=[1 0 1 1 1]

Tlben=1

Aucun delai