IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

10/10/2024

BODIN Noé
COLIN Guillaume
DOUANT Antoine
LE COQ Justine

Rapport SIT213

Atelier Logiciel

Simulation d'un système de transmission

Étape 5

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Introduction

L’étape 5 du projet introduit l’intégration d’un codage de canal dans la chaîne de transmission afin de réduire le Taux d’Erreur Binaire dans des environnements bruités. Ce codage est appliqué après la source et avant la destination et permet de transformer chaque bit logique en une séquence de trois bits pour améliorer la qualité de la transmission. L'objectif est de faciliter la détection et la correction des erreurs qui peuvent survenir lors de la transmission dans des conditions bruitées.

Le codage de canal offre deux avantages majeurs: l'amélioration de la synchronisation entre l’émetteur et le récepteur, en empêchant l'envoi de séquences prolongées de bits identiques, et la correction d'erreurs simples sur les paquets de trois bits. Si une erreur affecte un seul bit dans un paquet, le décodeur est capable de la détecter et de la corriger, ce qui améliore considérablement la fiabilité de la transmission.

L’objectif de cette étape est de mesurer l'impact du codage de canal sur la qualité de la transmission, en comparant le TEB avec et sans codage dans différents scénarios, avec des variations de Eb/N0 et des formes d’onde. À travers l’implémentation du codage, nous cherchons à démontrer l'amélioration de la robustesse du système face aux perturbations dues au bruit.

1. Conception des composants

1.1 CodageEmission

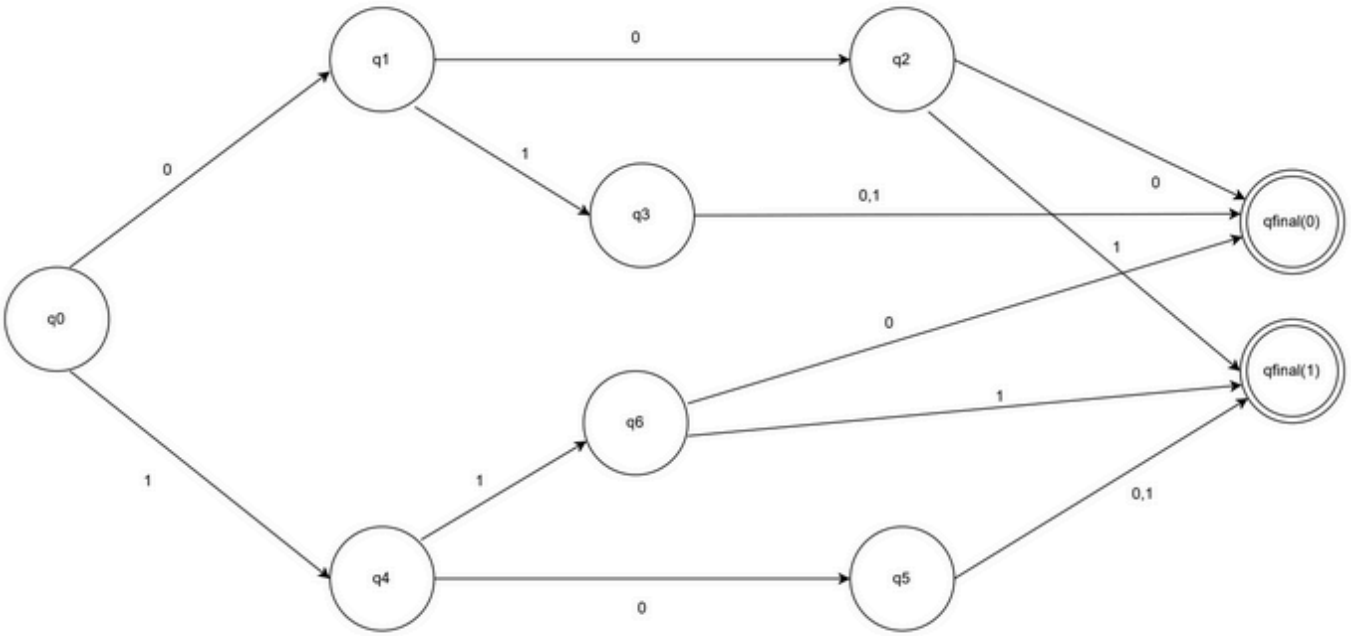
Le CodageEmission est un composant ajouté à la chaîne de transmission, juste après la source, pour transformer chaque bit logique en une séquence de trois bits. Défini dans le sujet.

Pour chaque bit **0** généré par la source, l'encodeur produit la séquence **010**. Pour chaque bit **1**, il produit la séquence **101**. Ce mécanisme ajoute une redondance qui permet au récepteur de détecter et corriger les erreurs survenant dans un paquet de trois bits.

1.2 DecodageReception

Le DecodageReception est le composant chargé de décoder les paquets de trois bits reçus en un bit unique. Ce composant est placé juste avant la destination pour corriger d'éventuelles erreurs dans les paquets reçus.

Le décodeur analyse les paquets de trois bits. Pour cela, nous utilisons un automate, créé pour être conforme aux consignes. C'est le suivant :



Il est implémenté en Java dans la méthode automate() de la classe.

2. Connexion des composants

La connexion des composants est essentielle pour permettre le passage fluide des informations depuis la source jusqu'à la destination finale, tout en tenant compte des transformations introduites par les trajets multiples et le bruit gaussien. Le simulateur suit une structure modulaire, où chaque composant est responsable d'une étape spécifique de la transmission.

2.1 Connexion dans la chaîne de transmission avec codage de canal

Dans cette étape, la chaîne de transmission inclut également un encodeur et un décodeur afin d'introduire un codage de canal pour améliorer la qualité de transmission, grâce à l'option -codeur. La connexion des composants se fait de la manière suivante, dans le cas d'une transmission bruité avec multi-trajet (la plus complète) :

- **Source → Codage d'émission** : la Source Fixe ou Source Aléatoire génère une séquence binaire. Cette séquence est transmise à l'encodeur (CodageEmission), qui transforme chaque bit logique en une séquence de trois bits.
- **Codage d'émission → Émetteur** : l'information encodée est ensuite transmise à l'émetteur analogique, qui convertit le signal binaire en signal analogique. Ce signal est modifié selon la modulation définie (NRZ, NRZT, RZ), le nombre d'échantillons par bit, et les amplitudes maximale et minimale
- **Émetteur → Transmetteur Analogique Multi-Trajet** : une fois que l'émetteur a converti le signal logique en signal analogique, ce signal est transmis au Transmetteur Analogique Multi-Trajet. Ce composant modélise les trajets multiples, définis par un retard (tau) et une atténuation (alpha). Chaque trajet supplémentaire est retardé et atténué avant d'être ajouté au signal global
- **Transmetteur Analogique Multi-Trajet → Transmetteur Analogique Bruité** : après avoir été modifié par les trajets multiples, le signal analogique est transmis au Transmetteur Analogique Bruité. Ce composant ajoute un bruit blanc gaussien à chaque échantillon du signal en fonction du SNR ou du Eb/N0 (-snr ou -snrpb, les deux sont implémentés :) spécifié dans les paramètres de simulation. Le bruit simule les perturbations présentes dans les canaux réels
- **Transmetteur Analogique Bruité → Récepteur** : le signal bruité est ensuite transmis au Récepteur Analogique, qui se charge de démoduler le signal analogique pour le convertir en information logique. Le récepteur utilise les amplitudes maximale

et minimale du signal pour déterminer les bits correspondants (1 ou 0), en fonction de la modulation utilisée.

- **Récepteur → Décodage de réception** : une fois l'information logique récupérée par le récepteur, elle est transmise au décodeur (DecodageReception). Ce composant prend les paquets de trois bits reçus et les convertit en un seul bit en détectant et corrigeant les erreurs éventuelles dans les paquets.
- **Décodage de réception → Destination finale** : après décodage, l'information binaire est transmise à la DestinationFinale, où la séquence reçue est comparée à l'information initialement émise par la source. Le TEB est ensuite calculé, permettant d'évaluer la qualité de la transmission en prenant en compte les erreurs introduites par les trajets multiples et le bruit.

Le codeur peut être utilisé sans les multi-trajets, ni le bruit, ni le passage analogique, avec le même type de fonctionnement pour chaque cas en changeant les transmetteurs entre le codeur et décodeur.

Pour avoir de meilleures performances, le récepteur pourrait être modifié, afin de détecter les blocs de trois bits et éventuellement différents décalages ou autre pour mieux détecter. Pour le moment le récepteur est le plus simpliste, il prend un seuil de moyenne en fonction de l'amplitude et décide par rapport à la moyenne du signal reçu. Avec le bit entier si la modulation est NRZ, sur le deuxième tiers du bit si RZ ou NRZT. Ce qui peut expliquer les moins bonnes performances de ces deux dernières modulations.

2.2 Rôles des sondes dans la connexion

Les sondes permettent de visualiser l'évolution du signal à différentes étapes de la chaîne de transmission. Elles fournissent des informations précieuses pour comprendre comment le signal est affecté par les différentes transformations, notamment la modulation, les trajets multiples, et le bruit gaussien. Grâce aux sondes, il est possible d'observer le comportement du système en temps réel et d'analyser les variations du signal.

Les sondes sont placées à chaque étape de la chaîne de transmission, lorsqu'elles sont activées via l'option -s. Après la source, après le codeur, après, l'émetteur, après le transmetteur multi-trajets, après le transmetteur bruité, après le récepteur, après le décodeur.

Les types de sondes sont :

- **SondeAnalogique** : Cette sonde capture et visualise le signal analogique à différents points du système, notamment après l'émetteur, après le transmetteur multi-trajet, et après le transmetteur bruité. Elle permet de suivre la déformation du signal à mesure qu'il traverse le canal de transmission.
- **SondeLogique** : La SondeLogique visualise la séquence binaire avant et après la modulation. Avant la modulation, elle montre le message logique tel qu'il est généré par la source, et après la démodulation, elle montre le message logique reçu par la destination.

3. Tests

3.1 Objectifs des tests

Les tests ont pour objectifs de :

- **Valider le comportement du codage et du décodage** : vérifier que le composant CodageEmission transforme chaque bit en une séquence de trois bits, et que le DecodageReception est capable de reconstruire le bit original tout en corrigeant les erreurs éventuelles dans les paquets de trois bits
- **Tester l'intégration des composants avec le codage** : s'assurer que le simulateur fonctionne correctement lorsque le codage est activé, que l'information est correctement encodée avant transmission, puis décodée à la réception. Le Taux d'Erreur Binaire doit diminuer grâce au mécanisme de correction d'erreurs du décodeur
- **Vérifier l'impact du bruit et des trajets multiples** : évaluer l'effet du bruit et des trajets multiples sur le signal encodé, et vérifier que le décodeur corrige efficacement les erreurs introduites lors de la transmission
- **Mesurer la performance du simulateur avec codage** : comparer le TEB avec et sans codage pour différents scénarios de transmission (SNR, trajets multiples) et mesurer l'amélioration

3.2 Structure des tests

Les tests sont organisés via JUnit, avec des tests unitaires pour valider les comportements individuels des composants de codage et décodage, ainsi que des tests d'intégration pour évaluer leur fonctionnement dans une chaîne de transmission complète.

Voici les principaux cas de test :

CodageEmissionTest :

- testRecevoirAndEmettreValidInformation : vérifie que l'encodage de bits se fait correctement. Par exemple, un 1 est transformé en 101 et un 0 en 010
- testRecevoirNullInformation et testEmettreNullInformation : s'assurent que des exceptions sont levées si des informations nulles ou vides sont reçues ou émises

- testRecevoirWithMultipleDestinations : teste que l'information encodée est correctement transmise à plusieurs destinations connectées, en s'assurant que toutes reçoivent la même séquence encodée

DecodageReceptionTest :

- testAutomate : teste la fonction automate, qui reconstruit le bit original à partir de trois bits encodés, en corrigeant si nécessaire
- testRecevoirInformationValide : vérifie que le décodeur reçoit correctement une information encodée, la traite et corrige les erreurs, si présentes
- testRecevoirNullInformation et testRecevoirEmptyInformation : s'assurent que des exceptions sont levées pour des informations nulles ou vides
- testRecevoirInformationIndivisibleParTrois : vérifie qu'une exception est levée si l'information reçue n'est pas divisible par trois (condition nécessaire pour le décodage)

3.3 Exécution des tests

Les tests sont exécutés via un script JUnit qui compile et exécute l'ensemble des tests. Cela garantit que chaque composant fonctionne correctement individuellement et dans le cadre d'une transmission avec codage et décodage activés. On utilise le script "./compile" pour compiler les tests, puis "./runTests" pour les exécuter, divers paramètres de transmission sont utilisés pour simuler des environnements avec du bruit et des trajets multiples, garantissant que le codage améliore la qualité de la transmission.

3.4 Résultats des tests

Les résultats des tests montrent que les composants CodageEmission et DecodageReception fonctionnent correctement selon leurs spécifications. Les tests ont confirmé que le codage transforme chaque bit logique en une séquence de trois bits, et que le décodage permet de reconstruire le bit d'origine en corrigeant les erreurs éventuelles dans les paquets reçus.

Les tests unitaires sur le CodageEmission ont démontré que les séquences générées pour chaque bit sont bien conformes aux attentes : 0 est transformé en 010 et 1 en 101. Les informations ont été correctement transmises et reçues par plusieurs destinations connectées.

De même, les tests du DecodageReception ont validé la capacité du décodeur à interpréter les paquets de trois bits et à corriger les erreurs d'un seul bit. Les exceptions ont été correctement levées lorsque des informations non conformes étaient reçues, telles que des messages vides ou des séquences non divisibles par trois.

3.5 Couverture de code avec Emma

L'outil Emma a été utilisé pour mesurer la couverture des tests sur l'ensemble du projet. Les résultats sont les suivants :

Element	Coverage	Covered Instructions	Missed Instructions	Total Instructions
sit213	84,9 %	8963	1593	10556
tests	91,3 %	5039	483	5522
src	77,9 %	3924	1110	5034
transmetteurs	94,9 %	1953	104	2057
Emetteur.java	95,4 %	433	21	454
DecodageReception.java	97,4 %	371	10	381
Recepteur.java	96,2 %	330	13	343
TransmetteurAnalogiqueBruite.java	86,0 %	277	45	322
TransmetteurAnalogiqueMultiTrajet.ja	98,9 %	278	3	281
CodageEmission.java	92,4 %	146	12	158
TransmetteurAnalogiqueParfait.java	100,0 %	47	0	47
TransmetteurParfait.java	100,0 %	36	0	36
Transmetteur.java	100,0 %	35	0	35
simulateur	60,6 %	1033	671	1704
visualisations	64,7 %	608	331	939
sources	100,0 %	174	0	174
information	96,6 %	113	4	117
destinations	100,0 %	43	0	43

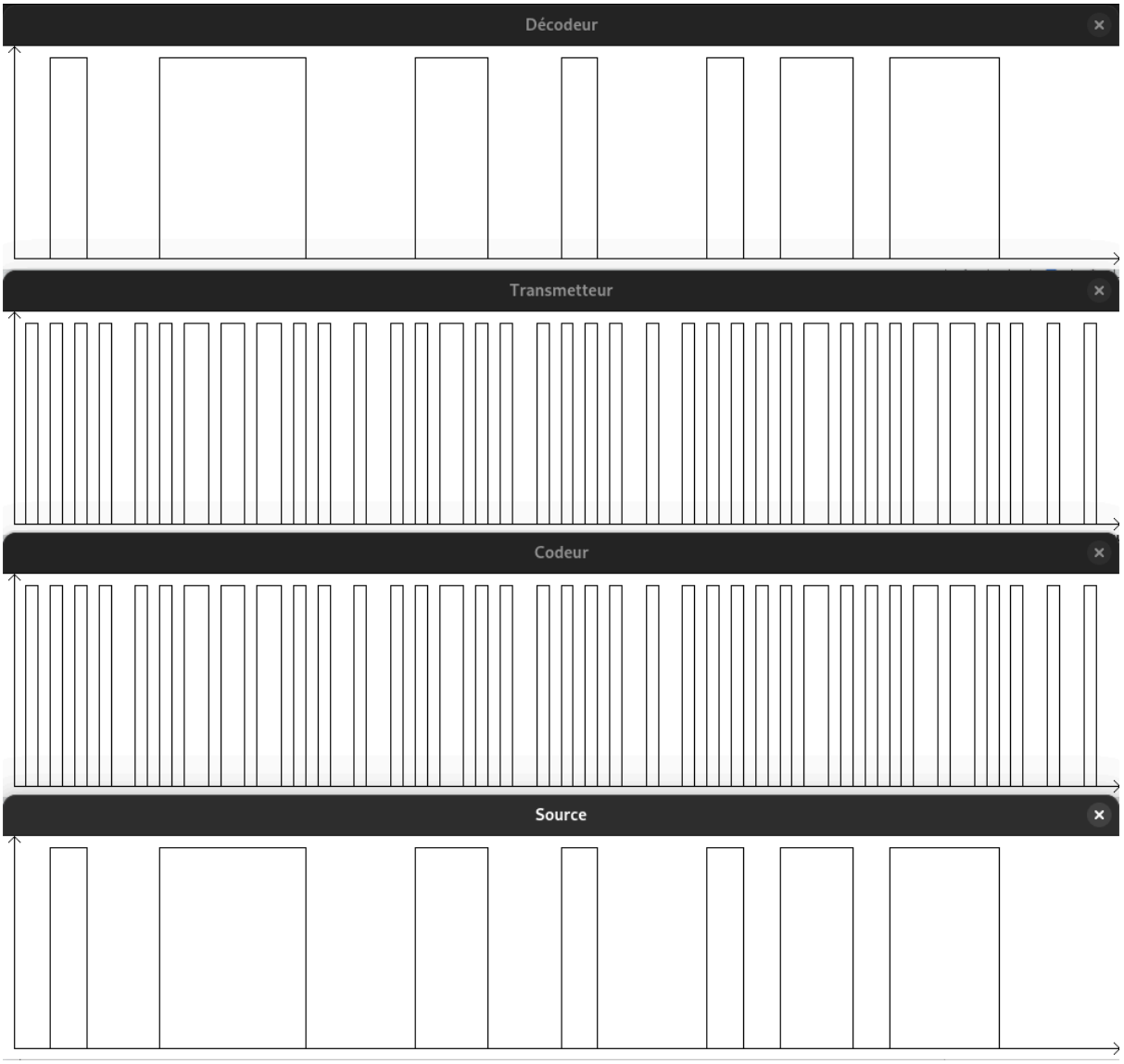
Les résultats montrent une couverture élevée, confirmant que l'essentiel du code a été testé et validé. La couverture légèrement plus faible de certains éléments, indique que des cas spécifiques peuvent être ajoutés lors des futures itérations pour renforcer la robustesse des tests. Globalement, la couverture est très satisfaisante, assurant que le projet a été largement vérifié pour détecter d'éventuelles anomalies ou régressions. Certaines classes ou méthodes n'ont pas besoin d'être testées (par exemple, la création des fichiers), ceci influe sur la couverture des tests.

4. Simulation d'une transmission avec codeur

Utilisation du codeur :

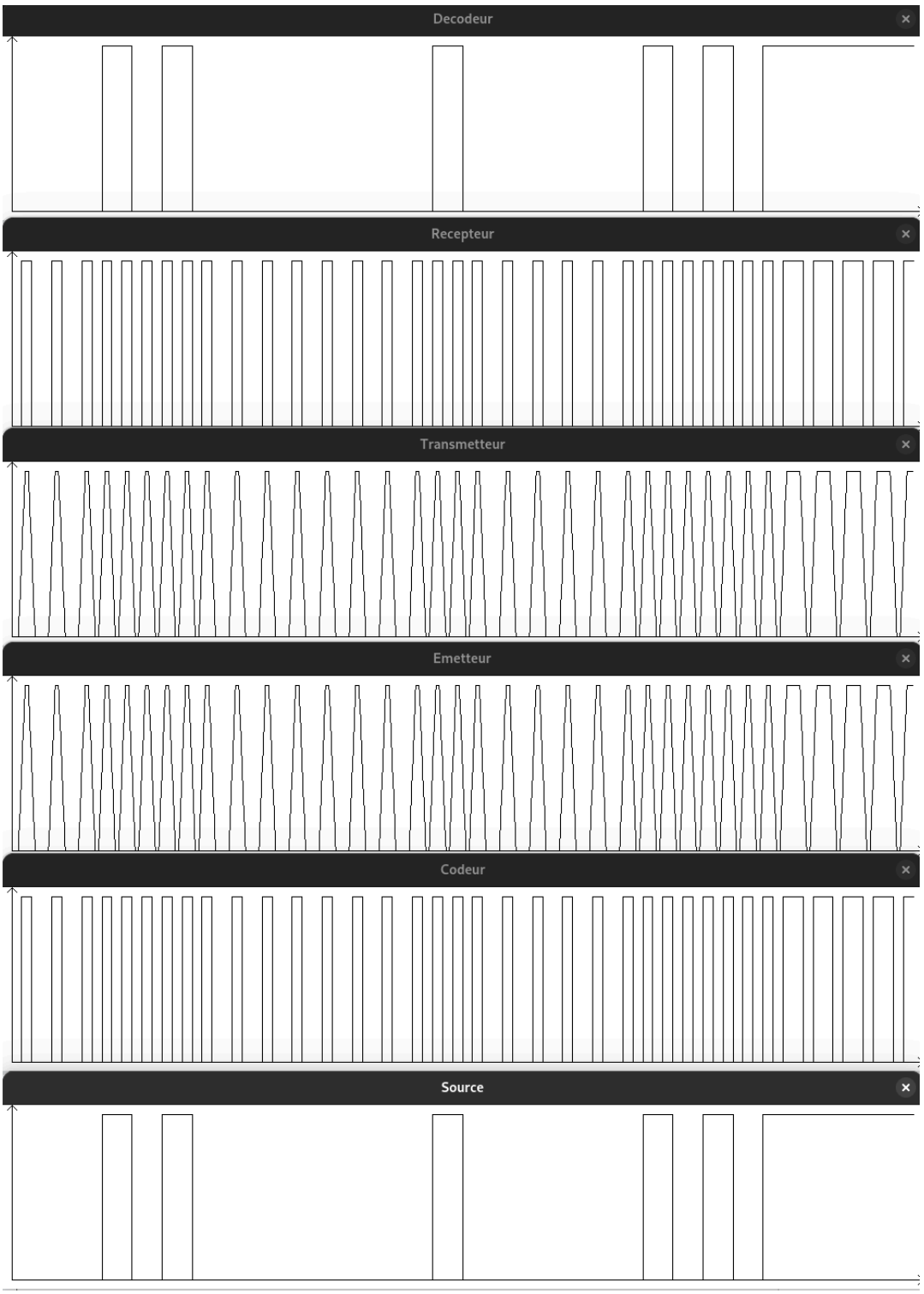
Transmission logique

```
~/sit213 main ❯ ./simulateur -s -mess 30 -codeur
java Simulateur -s  -mess  30  -codeur    =>   TEB : 0.0
```



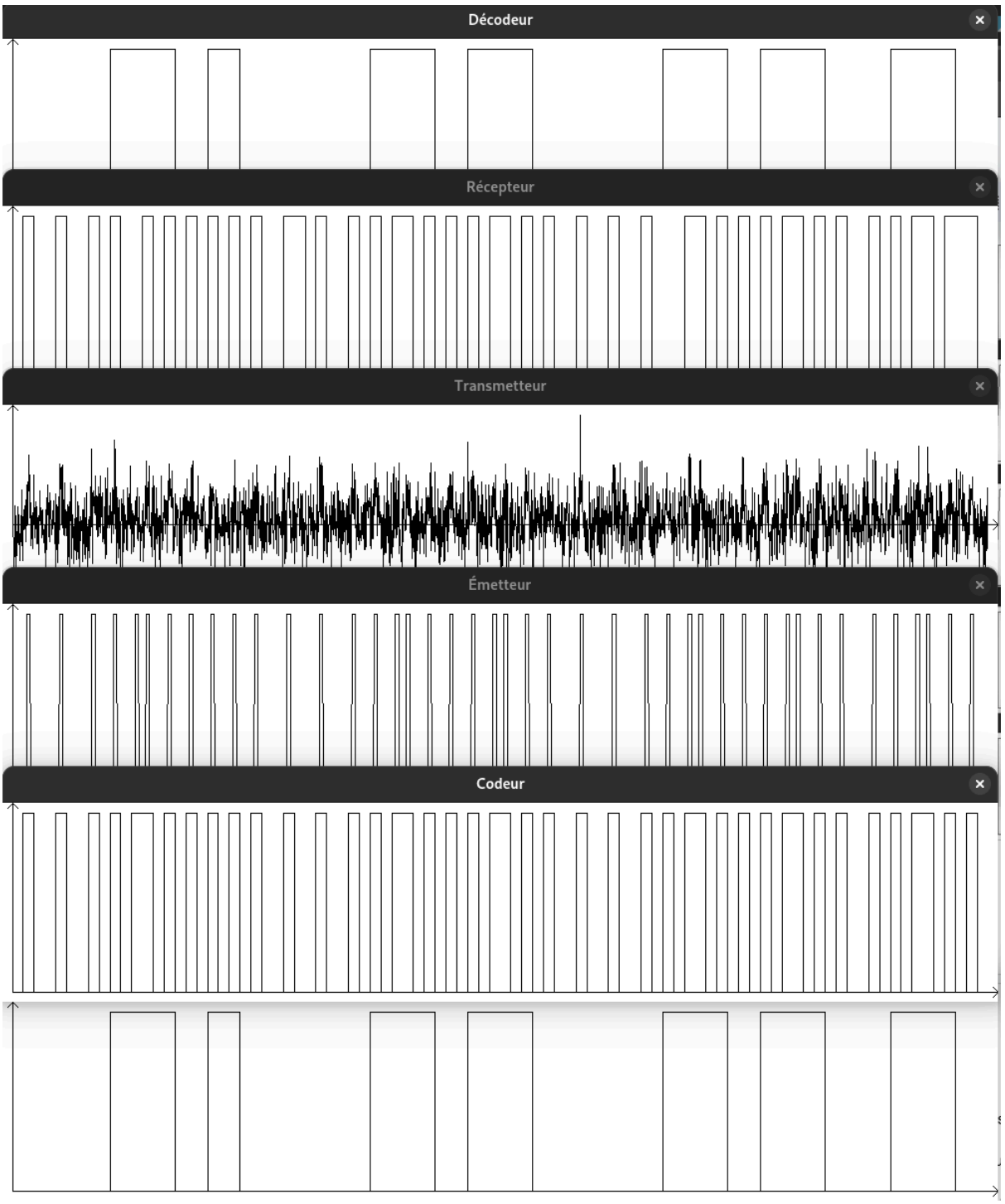
Transmission analogique parfaite

```
~/sit213 main ❯ ./simulateur -s -mess 30 -codeur -form NRZT
java Simulateur -s  -mess  30  -codeur  -form  NRZT    =>   TEB : 0.0
```



Transmission analogique bruité

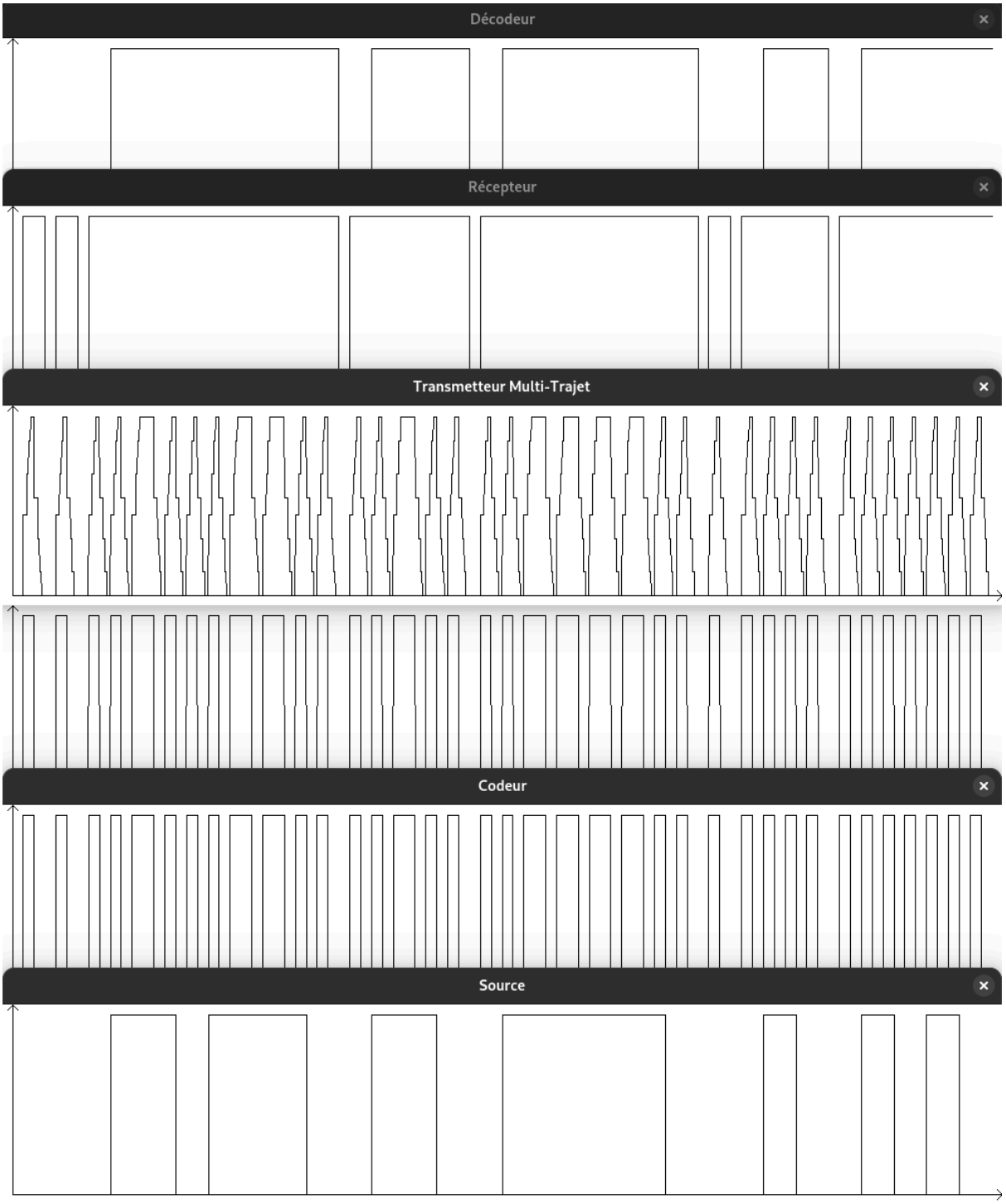
```
~/sit213 main ❯ ./simulateur -s -mess 30 -codeur -form RZ -snrpb 6
java Simulateur -s  -mess  30  -codeur  -form  RZ  -snrpb  6    =>   TEB : 0.0
```



On remarque bien la correction d'erreur, le codeur et récepteur sont légèrement différents. Cependant, il n'y a aucune erreur à la destination.

Analogique multi-trajets sans bruits

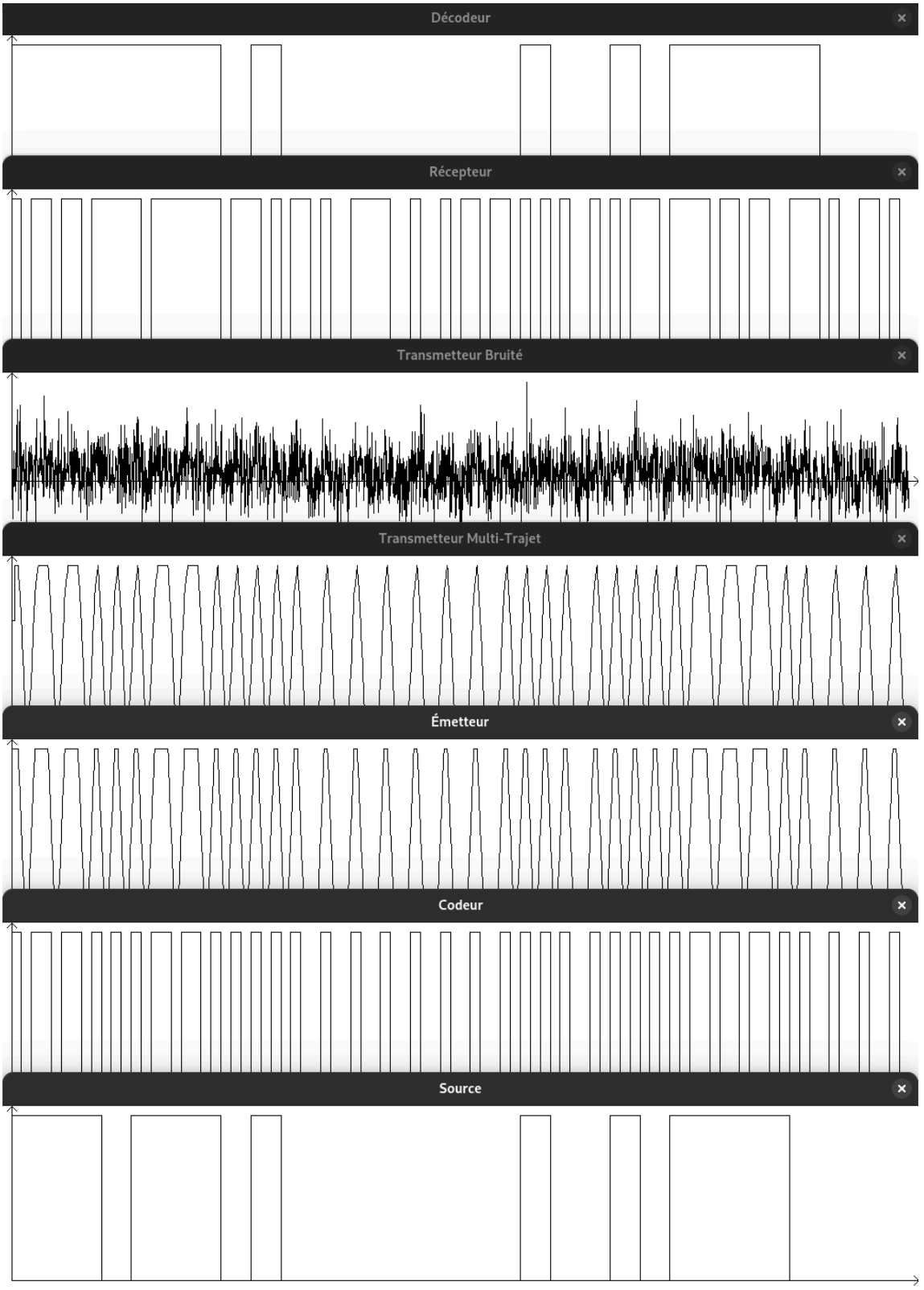
```
~/sit213 main ❯ ./simulateur -s -mess 30 -codeur -form NRZ -ti 10 0.5 15 0.4 20 0.3
java Simulateur -s  -mess  30  -codeur  -form  NRZ  -ti  10  0.5  15  0.4  20  0.3    =>   TEB : 0.2
```



Certaines erreurs sont corrigées, mais pas toutes.

Analogique multi-trajets avec bruits

```
~/sit213 main ❯ ./simulateur -s -mess 30 -codeur -form NRZT -ti 10 0.5 -snrpb 5
java Simulateur -s  -mess  30  -codeur  -form  NRZT  -ti  10  0.5  -snrpb  5    =>   TEB : 0.06666667
```

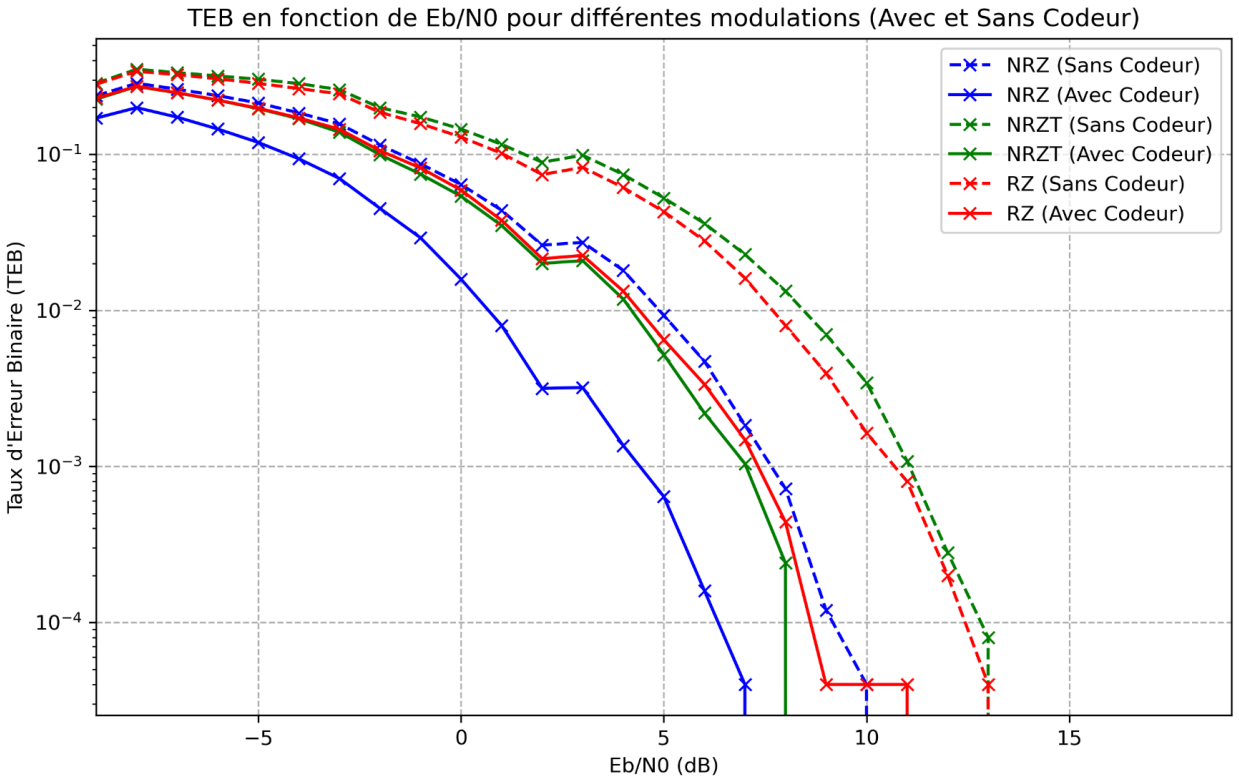


Deux erreurs, malgré le codeur et récepteur différents.

Pour vérifier le fonctionnement du codage et son efficacité, nous faisons des courbes de TEB en fonction de Eb/N0. Comme dans l'étape 3, nous faisons évoluer le Eb/N0 et regardons le SNR pour chaque type de modulations, cette fois avec et sans codage.

Nous affichons le Taux d'Erreur Binaire sous échelle logarithmique et Eb/N0 en échelle linéaire. Cela est différent de l'affichage à l'étape 3 qui n'était pas adapté et moins lisible. De même, un léger lissage est effectué (moyenne mobile) mais moins efficace, car moins de mesures.

Sur des messages de 5000 bits (la simulation a pris une heure).



On retrouve ainsi les trois types de modulations avec et sans codage, les transmissions avec codage sont toutes meilleures que celles sans. Le NRZ avec codeur est le plus performant, suivi du NRZT codeur, RZ codeur et NRZ sans codeur proches.

On gagne 3 dB en moyenne.

Nous remarquons que pour Eb/N0 = 2 dB et Eb/N0 = 3, TEB reste stable ou augmente légèrement pour toutes les modulations avec ou sans codage. Nous ignorons pourquoi. Avec plus de points, la courbe aurait pu être lisible.

Conclusion

En conclusion, le mécanisme de codage et de décodage mis en place permet d'améliorer la transmission des données en ajoutant la transformation des bits logiques en séquences de trois bits. Le CodageEmission et le DecodageReception fonctionnent de manière complémentaire pour encoder les bits à l'envoi et corriger les erreurs à la réception.

Les tests réalisés ont validé l'efficacité du système, notamment face au bruit et aux multi-trajets. L'intégration du codeur dans une chaîne de transmission complète avec modulation et bruit a montré des améliorations dans le Taux d'Erreur Binaire (TEB), notamment avec la modulation NRZ, où un gain de 3 dB a été observé.

Malgré les bonnes performances générales, certaines erreurs persistent dans des conditions de transmission bruitée et avec multi-trajets.