Identification automatique de signaux grâce à la fonction d'autocorrélation

Source : https://connect.ed-diamond.com/MISC/mischs-022/identification-automatique-de-signaux-grace-a-la-fonction-d-autocorrelation

> *Vous connaissiez la chasse au trésor… Initiez-vous maintenant à la chasse au signal (signal hunting en anglais). La chasse au signal consiste à rechercher les signaux qui nous entourent dans l'espace invisible du spectre électromagnétique. Mais plus besoin de rester l'oreille collée sur un haut-parleur à tourner le bouton pour régler la fréquence. La SDR (Software Defined Radio) a révolutionné tout cela : une radio numérique et un PC est vous voilà armé pour découvrir ce monde que les professionnels dénomment le SIGINT (SIGnal INTelligence).*

Pour chasser un signal, il faut d'abord le détecter (détection) puis l'identifier (identification). Pourquoi l'identifier ? Pour connaître ses caractéristiques, mais surtout pour lui appliquer le « bon » démodulateur afin de décoder son contenu. L'identification peut se faire manuellement à l'oreille sur la mélodie basse fréquence du signal ou à l'œil sur la forme du spectre. Mais ces méthodes manuelles nécessitent des opérateurs expérimentés et ne permettent pas d'automatiser la démodulation du signal intercepté. Aujourd'hui, nous vous proposons d'aborder l'identification automatique de signaux. Nous commencerons par une présentation générale de la SDR et en

particulier des logiciels libres disponibles. Nous étudierons ensuite la théorie de l'identification sur la base de la fonction d'autocorrélation et présenterons une implémentation temps réel de l'identification automatique de signaux.

1. Le COMINT et le monde du logiciel libre

Le COMINT (*COMmunication INTelligence*), qui est une branche du SIGINT, recouvre les technologies d'écoute passive des transmissions radio principalement en gamme HF, VHF, UHF donc de quelques MHz jusqu'à 3 GHz typiquement. Le COMINT est utilisé pour remonter aux metadata (angle d'arrivée, coordonnées géographiques de l'émetteur, statistiques de trafic…) et, si les transmissions ne sont pas chiffrées, à leur contenu.

1.1 Radios large bande

L'apparition d'équipements radio large bande grand public (RTL-SDR, Hack RF, SDRPlay, LimeSDR Mini, Ettus pour ne citer que quelques-uns) a fait exploser la diffusion sur Internet de logiciels libres permettant l'écoute passive des communications radio. Pour 300 euros, vous disposez d'une radio 20 MHz-3 GHz présentant une bande d'analyse de 20 MHz avec des échantillons de 12 bits vous permettant de chasser large (LimeSDR Mini). En cassant un peu plus votre tirelire, vous pouvez même passer à une largeur de bande de 60 MHz avec les radios Ettus au format carte de crédit.

Les performances en réception de ces radios peuvent être améliorées par l'utilisation d'un préamplificateur faible bruit pour quelques dizaines d'euros (LNA en anglais) et d'une antenne performante. De plus, des radios comme Lime ou Ettus offrent une capacité d'émission qui ouvre la voie à du COMINT « offensif » qui sera limité par la puissance d'émission réduite de ces radios (< 100 mW max et beaucoup moins dans les fréquences élevées) et la législation de chaque pays.

1.2 Logiciels libres COMINT

L'écoute passive des communications a déjà donné lieu à plusieurs articles parus dans les colonnes de *MISC* que ce soit pour l'écoute des communications GSM **[1]** ou Iridium **[2]** sur la base de logiciels libres du monde Linux. D'autres logiciels libres sont disponibles sur Internet pour l'écoute passive de signaux comme les signaux DMR (logiciel DSD), TETRA (logiciel Telive ou Tetra-kit), STD –C (logiciel Scytale-C), AERO (logiciel Jaero), TETRAPOL (logiciel Tetrapol-kit) pour ne citer que les principaux. Ces logiciels permettent de remonter aux metadata et au contenu des transmissions si elles ne sont pas chiffrées (ou si le système de chiffrement présente des vulnérabilités). De même, des logiciels libres utilisant les capacités émission de ces radios comme OpenBTS, OpenBSC (BTS 2G) ou srsLTE et openLTE (enodeB 4G) peuvent être adaptés par tout amateur averti pour faire du COMINT « offensif » (voir **[3]** ou **[4]**).

La SDR prend ainsi tout son sens puisque sans changement de matériel (hormis l'antenne) vous pouvez passer en quelques secondes de l'interception de signaux satellites à l'interception de signaux VHF par exemple.

1.3 Matériels

Ces logiciels s'exécutent sans aucun problème sur un PC I3 sous Linux Ubuntu LTS 18.04. Si les Raspberry précédents étaient un peu justes en puissance de calcul dès que la bande d'analyse dépassait 2 MHz, l'apparition du RPI4 rend possible l'écoute passive avec des moyens informatiques qui coûtent moins de 100 euros sur des bandes d'analyse de plusieurs MHz. Ils sont bien évidemment de plus particulièrement compacts, légers et donc discrets (voir figure 1) moyennant l'utilisation d'une batterie externe pour smartphone.

(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_01.jpg)

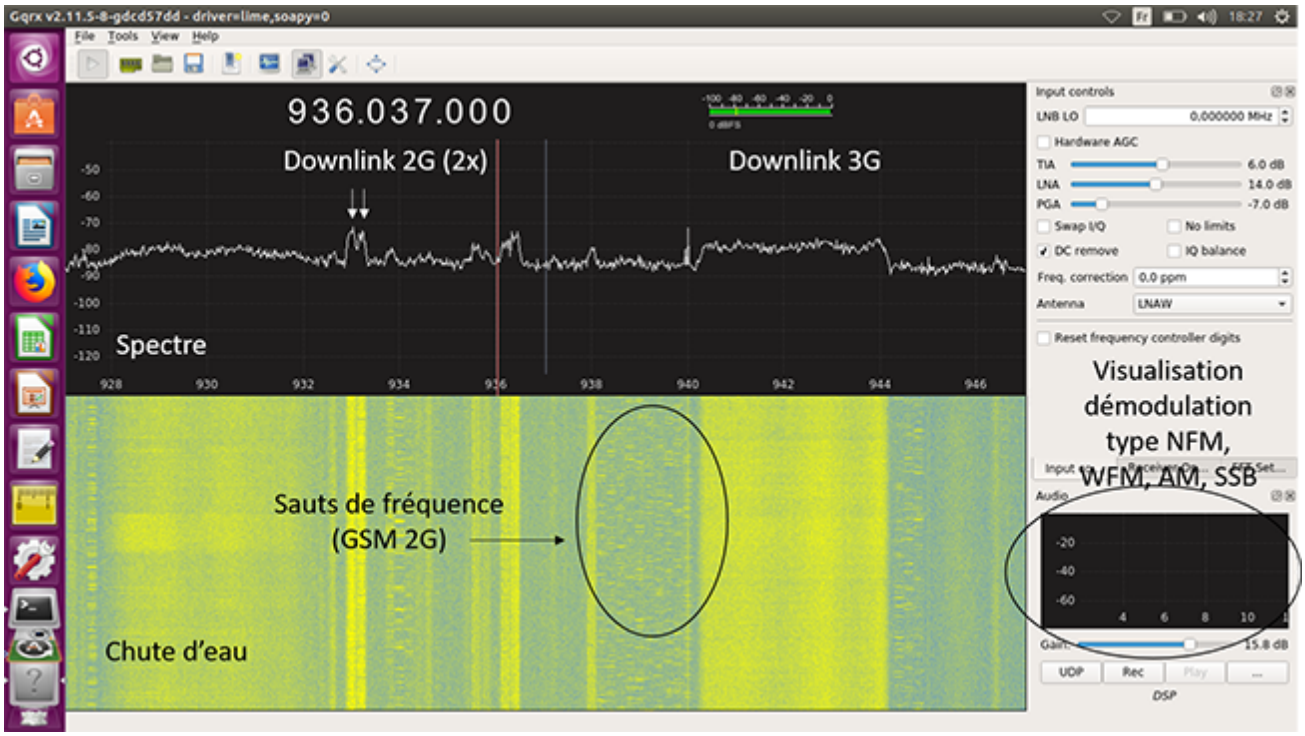
Fig. 1 : Radio LimeSDR mini et carte RPI4.

Des images (img) de carte SD (PISDR par exemple) intégrant l'OS et certains des logiciels précités sont aussi disponibles sur Internet permettant l'utilisation de ces logiciels sous RPi4 sans avoir à les installer, ce qui peut prendre un certain temps et être quelque peu rébarbatif **[5]**. De même, des machines virtuelles intégrant ces logiciels sont disponibles sous Oracle VM VirtualBox **[6]** ou VMware Workstation Player pour les PC Windows.

2. Le monitoring et l'analyse technique

Avant d'écouter un signal d'intérêt (donc appliquer au signal intercepté le « bon » démodulateur logiciel), il faut d'abord le détecter et l'identifier, c'est-à-dire découvrir sa fréquence d'émission puis ses paramètres de modulation/codage ou tout simplement de quel type de signal il s'agit.

Le monde du logiciel libre offre aussi une collection de logiciels permettant le monitoring du spectre radio électrique et donc la détection des signaux d'intérêt. Les plus célèbres logiciels du monde Linux sont GQRX **[7]** et plus récemment SDRAngel **[8]**. HDSDR **[9]** est un logiciel équivalent, mais sous Windows. Ces logiciels permettent de visualiser en temps réel une partie du spectre radioélectrique correspondant à la bande instantanée de la radio (20 MHz par exemple) et la « chute d'eau » qui est une visualisation dans le domaine temporel de l'activité électromagnétique sur la bande d'analyse de la radio. Ces logiciels permettent la détection de signaux continus, intermittents que ce soit à fréquence fixe ou à saut de fréquence. Ces logiciels intègrent des démodulateurs analogiques basiques (NFM, WFM, AM, USB). Plus récemment, SDRAngel a commencé à intégrer des démodulateurs numériques basiques tout en offrant la possibilité de faire plusieurs démodulations en parallèle (donc d'écouter passivement plusieurs signaux présents dans la bande d'analyse en simultané). L'interface homme<->machine GQRX est présentée à la figure 2.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_02.png)

Fig. 2 : Interface homme <->machine GQRX.

Ces logiciels peuvent être avantageusement complétés par un logiciel de scanning type QSpectrumAnalyzer qui n'est malheureusement plus soutenu. Il permet de représenter spectre et « chute d'eau » sur une bande supérieure à la bande instantanée d'analyse de la radio. Un spectre de 100 MHz sera ainsi analysé par tranche de 20 MHz, la radio revenant une fois sur cinq sur chaque tranche de 20 MHz. La vitesse de balayage équivalente atteint presque 500 MHz/s sur un PC I3.

L'ensemble de ces logiciels détectent la présence d'énergie, mais l'application du bon « démodulateur » repose uniquement sur l'expertise de l'opérateur. Certains signaux se retrouvent plus dans certaines parties du spectre, certains signaux ont un spectre ou une musique basse fréquence (nous y reviendrons) reconnaissables par les yeux ou l'oreille d'un opérateur expérimenté.

Au sein du COMINT, l'analyse technique permet justement de détecter les paramètres de la modulation et du codage ainsi que le type de protocole pour appliquer par la suite le « bon » démodulateur.

Le monde du logiciel libre offre aussi quelques logiciels permettant la détermination des paramètres techniques du signal. Il s'agit de logiciels comme URH déjà présenté dans les colonnes de *MISC* ou Inspector.

Enfin, le logiciel SigDigger **[10]**, l'excavatrice de signaux en anglais, offre en un seul logiciel les fonctions de monitoring (comme GQRX) et de scanning (comme QSprectrumAnalyzer) plus l'appel direct dans la même interface homme<->machine au logiciel d'analyse technique Inspector.

3. Identification automatique des signaux

L'identification automatique des signaux (le terme classification est parfois utilisé) est une autre branche de l'analyse technique. Elle ne cherche pas à déterminer les paramètres techniques d'un signal intercepté, mais à déterminer directement le type de signal intercepté, c'est-à-dire s'il s'agit d'un signal TETRA, TETRAPOL, GSM, UMTS, LTE, DMR, DAB, etc. Pour ce faire, l'identification automatique des signaux utilise une bibliothèque de signaux connus et compare le signal intercepté aux signaux de la bibliothèque pour déterminer le signal le plus ressemblant…ou pour le déclarer comme inconnu s'il n'est pas présent dans la bibliothèque de signaux.

Le monde du logiciel libre offre depuis peu quelques logiciels permettant l'identification automatique de signaux. Ces logiciels libres sont beaucoup moins connus et sont plus expérimentaux que les logiciels présentés précédemment. De plus, chaque logiciel utilise une technique différente pour faire la comparaison à la bibliothèque de signaux.

Le logiciel **[11]** utilise la technique des réseaux de neurones convolutifs (*Deep Learning*) pour déterminer le type de signal intercepté.

Le logiciel **[12]** utilise la « musique » basse fréquence du signal pour déterminer le type de signal intercepté. Il repose sur des algorithmes type « Shazam » disponibles dans le monde du logiciel libre. Il est donc supposé reconnaître la « musique » du signal intercepté comme Shazam le fait pour une chanson du Top 50 !

Une troisième technique repose sur la fonction d'autocorrélation des signaux (*AutoCorellation Function* pour ACF en anglais). La base de signaux SigidWiki **[13]** indique parfois la valeur de l'ACF des signaux de sa base. Si cette technique est décrite en termes de principe sur Internet et si le calcul de la fonction d'autocorrélation est implémenté dans des logiciels de monitoring comme HSDR **[14] [15]**, elle n'a pas donné lieu à une implémentation permettant l'identification automatique du signal en temps réel (*Automatic Signal Identification* pour ASI en anglais). La suite de cet article décrit une implémentation temps réel de la reconnaissance automatique des signaux sur la base de la fonction d'autocorrélation.

4. La fonction d’autocorrélation (FAC) pour identifier un signal

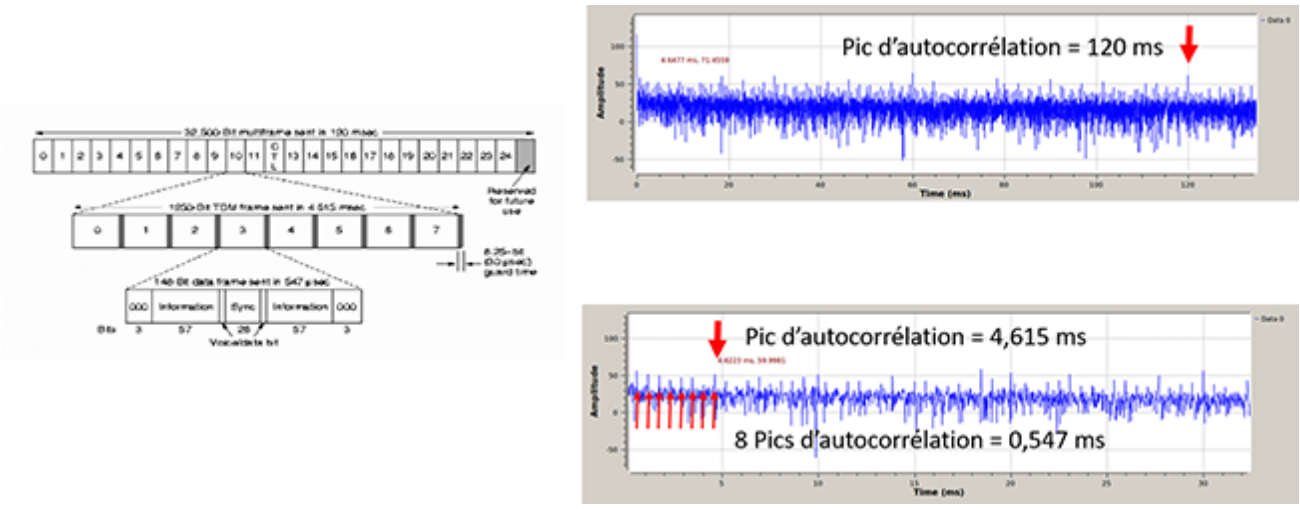
4.1 Fonction d’autocorrélation et signaux numériques

La fonction d’autocorrélation compare le signal à une réplique retardée de lui-même. Elle se représente sur un graphique 2D avec en abscisse le temps (en général en ms) et en ordonnée le coefficient de corrélation pour chaque valeur temporelle (plus la valeur est élevée et plus il y a corrélation entre le signal et sa réplique retardée de X ms : il s’agit alors d’un pic d’autocorrélation).

Les signaux numériques sont constitués de trames (et parfois de sous-trames) répétitives. La durée des trames (et des sous-trames) est un identifiant fort d’un signal numérique.

Ces trames répétitives donnent naissance à des pics d’autocorrélation à la récurrence de la trame quand on applique au signal complexe (I,Q) la fonction d’autocorrélation.

La figure 3 montre les trois premiers niveaux de trames GSM et la fonction d’autocorrélation d’un signal GSM avec les pics d’autocorrélation correspondant aux différents niveaux de trame.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_03.png)

Fig. 3 : Trames et fonction d’autocorrélation d’un signal GSM.

La durée de trame pour quelques signaux numériques classiques est indiquée dans le tableau 1. Les valeurs de durée de trame sont entre quelques ms et la centaine de ms.

	Trame	**Sous-trame**
TETRAPOL	20 ms	--
TETRA	56,67 ms	14,167 ms

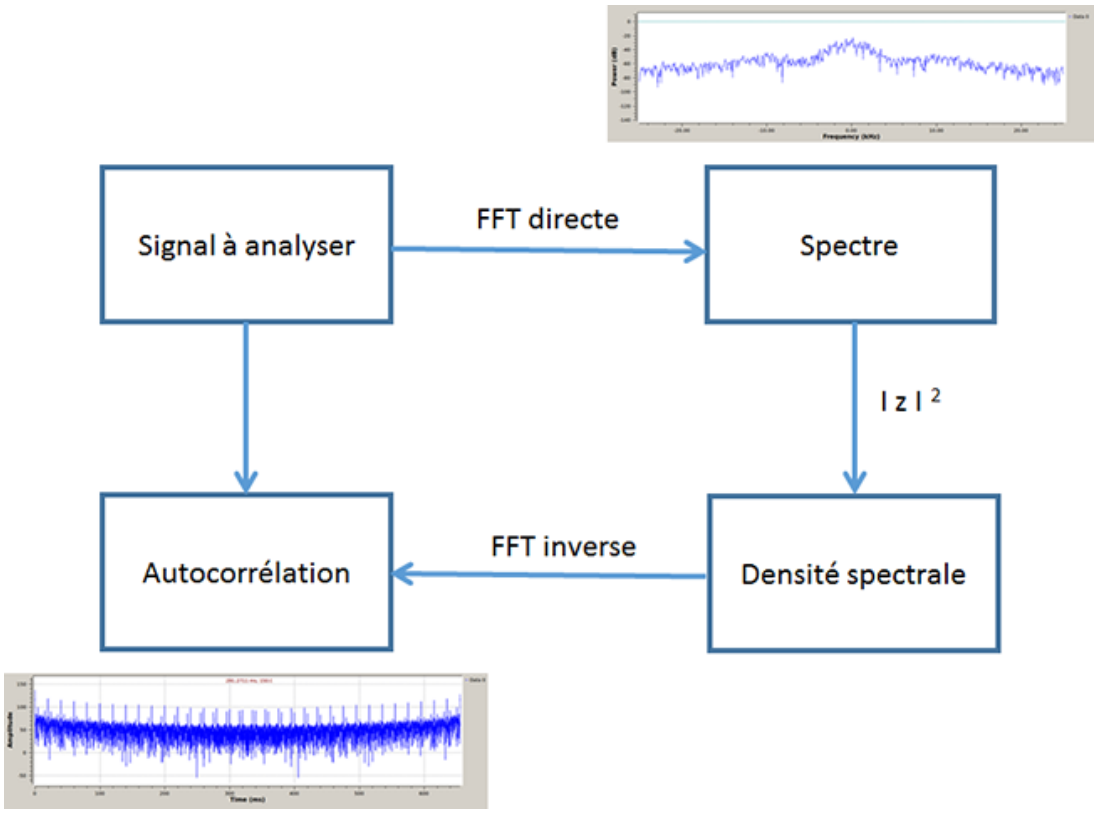
GSM	4,615 ms	0,547 ms
DMR	30 ms	--
STANAG 4285 / MPT1237	106,66 ms	--
POCSAG	26,67 ms	--
LTE	10 ms	--
STD-C	135 ms	--
DAB	96 ms	--

Tableau 1

4.2 Le théorème de Wiener-Kintchine

Le calcul temps réel de la fonction d'autocorrélation d'un signal complexe (I,Q) se fait par application du théorème de Wiener-Kintchine, qui date des années 30 bien que Einstein l'ait déjà évoqué en 1914 dans un mémo de deux pages, mais sans le démontrer. Ce théorème énonce que la transformée de Fourier de la fonction d'autocorrélation d'un signal est la densité spectrale de ce signal.

Pour calculer de manière économique en termes de puissance de calcul la fonction d'autocorrélation d'un signal, il suffit donc d'appliquer à ce signal une première FFT (*Fast Fourier Transform*) directe puis de calculer sa densité spectrale et d'appliquer enfin une deuxième FFT inverse comme indiqué à la figure 4.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_04.png)

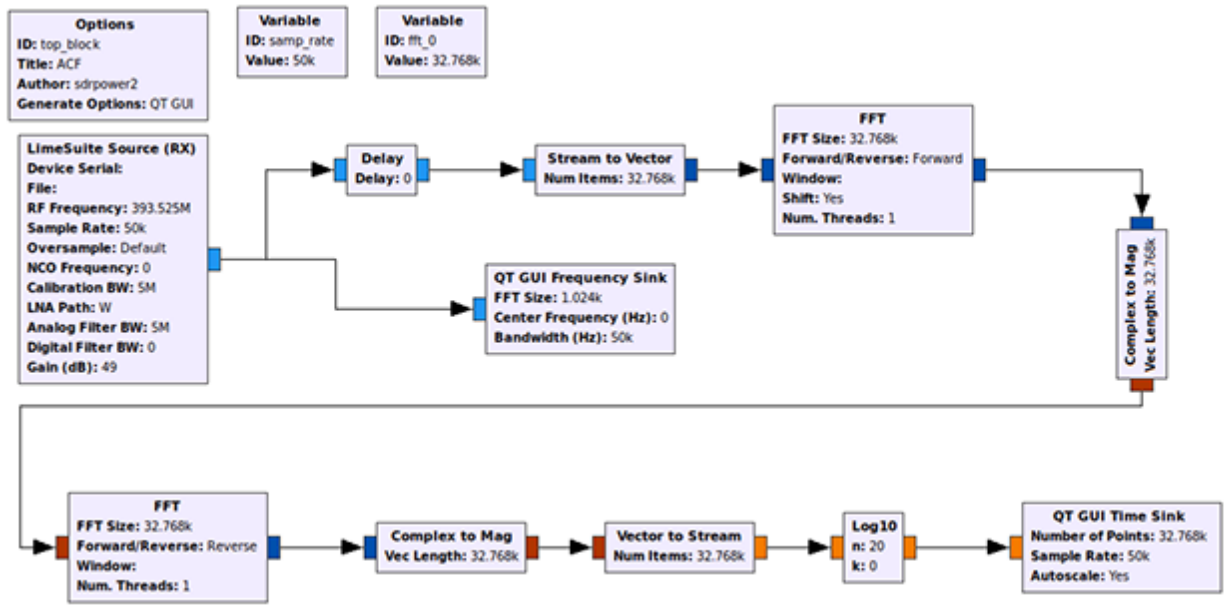
Fig. 4 : Calcul de la fonction d'autocorrélation.

4.3 Implémentation sur GNU Radio

Sur ces bases, il est extrêmement simple d'implanter la fonction d'autocorrélation dans le logiciel libre GNU Radio, le « couteau suisse » de la SDR, déjà présenté dans la revue. Les principaux blocs à mettre en œuvre sont les suivants :

- un bloc radio source (Rx) et un bloc de visualisation dans le domaine fréquentiel ;
- un bloc FFT direct ;
- un bloc « calcul du module » ;
- un bloc FFT inverse ;
- un bloc de visualisation dans le domaine temporel.

Le fichier Grc est disponible sous GitHub **[16]**. Son flowgraph est représenté à la figure 5.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_05.png)

Fig. 5 : Flowgraph de la fonction d'autocorrélation sous GNU Radio.

4.4 Paramètres d'analyse

À noter que le traitement est indépendant de la modulation radio utilisée : modulation de fréquence (large ou étroite), modulation d'amplitude, modulation bande latérale unique, etc.

Les paramètres importants d'analyse sont la fréquence d'échantillonnage (Fs) et la taille de la FFT (NFFT). Ils déterminent la profondeur temporelle de la fonction d'autocorrélation par la relation suivante :

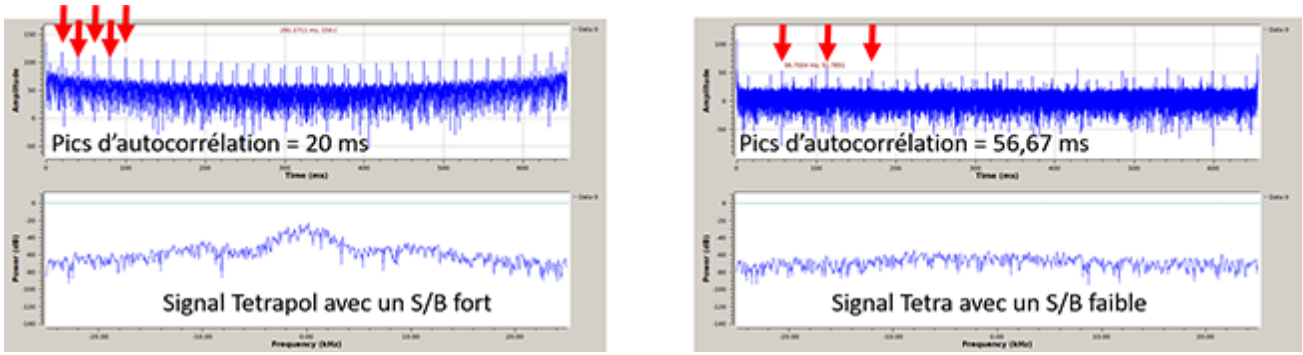
Profondeur temporelle = (NFFT/2) / Fs

Ainsi avec une fréquence d'échantillonnage de 50 KHz et une taille de FFT de 32768, la profondeur temporelle de la fonction d'autocorrélation est de 327ms, ce qui permet de détecter dans la pratique des pics d'autocorrélation jusqu'à environ 200ms de profondeur. Cette profondeur temporelle est jugée suffisante à la vue des durées de trame des principaux signaux numériques (voir tableau 1).

Augmenter la taille de la FFT augmenterait la profondeur temporelle de la fonction d'autocorrélation, mais exigerait plus de temps de calcul. Augmenter la fréquence d'échantillonnage diminuerait la profondeur temporelle de la fonction d'autocorrélation.

4.5 La fonction d'autocorrélation de signaux classiques

La figure 6 montre la fonction d'autocorrélation pour un signal Tetrapol (trame = 20 ms) et un signal Tetra (trame = 56,67 ms). On distingue pour les deux signaux les pics d'autocorrélation qui constituent la signature unique du signal. Le meilleur rapport signal à bruit du signal Tetrapol donne une fonction d'autocorrélation beaucoup plus propre que pour le signal Tetra qui souffre d'un faible rapport signal à bruit. Dans le cas du signal Tetra, on distingue en plus des pics d'autocorrélation correspondant aux sous-trames (sous-trames = 14,167 ms).



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_06.png)

Fig. 6 : Fonction d'Autocorrélation des signaux TETRAPOL et TETRA.

La vidéo **[17]** visualise le résultat de la fonction d'autocorrélation pour différents signaux classiques (LTE (805 MHz), GSM (958,4 MHz), Tetrapol (393,475 MHz), Tetra (426,611 MHz) et DMR (452,525 Mhz)). Cette visualisation permet à un opérateur expérimenté de déterminer le type de signal, mais d'autres traitements sont nécessaires pour arriver à l'identification automatique des signaux qui aidera un opérateur moins expérimenté.

5. Identification automatique des signaux à partir de la fonction d'autocorrélation

La mesure automatique de la récurrence des pics d'autocorrélation permet de déduire le type de signal intercepté du moment que cette récurrence a été enregistrée dans une bibliothèque pour chaque signal. Cela nécessite au préalable pour un signal inconnu de la déterminer soit par la connaissance à priori des trames utilisées soit par une mesure visuelle grâce grâce au flowgraph de la figure 5.

La mesure automatique de la récurrence des pics d'autocorrélation n'est pas possible dans GNU Radio avec les blocs existants. Il est donc nécessaire de développer la fonction d'autocorrélation en langage C avec l'API LMS. L'outil LimeScan de la suite LimeTools a été utilisé pour la partie acquisition radio à partir d'une radio LimeSDR mini. Il a été adapté pour implémenter une première FFT, le calcul de la densité spectrale et la seconde FFT comme expliqué précédemment. Le résultat est stocké dans un tableau qui va être exploité pour l'identification automatique du signal. Pour une fréquence d'échantillonnage de 50 kHz, le tableau contiendra la valeur d'autocorrélation au pas de 0,02 ms.

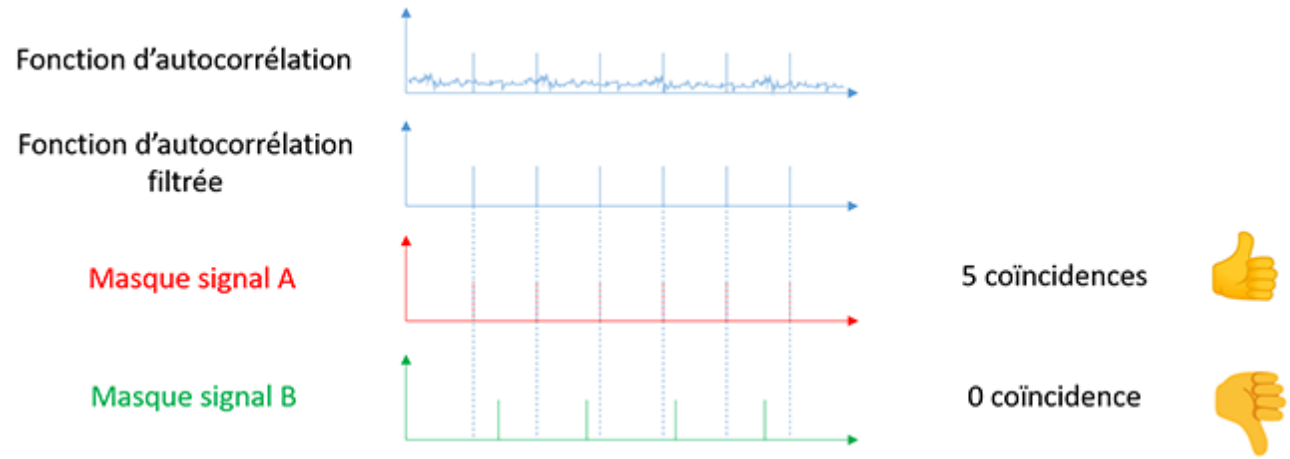
L'identification automatique du signal se fait en trois étapes.

La première étape consiste à filtrer le signal d'autocorrélation pour ne garder que les pics d'autocorrélation et éliminer le bruit. L'algorithme utilisé consiste à calculer le niveau moyen de la fonction d'autocorrélation. Un pic d'autocorrélation est déclaré dès lors que le niveau de la fonction d'autocorrélation dépasse l'écart-type de la moyenne.

La seconde étape repose sur la bibliothèque de signaux qui pour chaque signal de la bibliothèque indique la position attendue des pics d'autocorrélation (masque). Le masque de chacun des signaux de la bibliothèque est appliqué sur la fonction d'autocorrélation pour définir une note. L'algorithme

définira comme signal reconnu celui dont le masque génère la meilleure note. La comparaison des notes obtenues permet de définir un taux de confiance pour l'identification réalisée.

La figure 7 décrit le principe de ces deux algorithmes.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_07.png)

Fig. 7 : Algorithmes d'identification automatique.

Ces deux algorithmes permettraient d'identifier un signal sans autre algorithme si les pics d'autocorrélation de chaque signal de la base de données sont distincts. Cela serait par exemple le cas pour une base de données limitée à deux signaux : Tetra et Tetrapol. Il existe cependant des signaux qui ont des pics d'autocorrélation en commun (Tetrapol avec une récurrence de 20 ms et DMR avec une récurrence de 30 ms, soit un pic d'autocorrélation commun toutes les 60 ms) voire l'ensemble des pics d'autocorrélation en commun (Stanag 4285 (modulation série pour transmission HF) avec une récurrence de 106,66 ms et MPT1237 avec une récurrence de 106,66 ms).

Il est donc nécessaire de compléter les algorithmes décrits ci-dessus par une troisième étape. À titre d'exemple, les compléments possibles pour les deux cas décrits précédemment sont indiqués ci-dessous :

- Seuls les pics d'autocorrélation distincts sont renseignés dans le masque de chaque signal : 20, 40, 80, 100, 140, etc. pour Tetrapol et 30, 90, 150, etc pour DMR.
- La fréquence est utilisée pour discriminer en complément, les signaux STANAG4285 étant principalement en bande HF et les signaux MPT1237 principalement en bande V/UHF.

Le détail des compléments d'algorithme peut être consulté directement dans le code source publié sous GitHub **[16]**.

6. Logiciel LimeAuto

L'installation et le lancement du logiciel sont détaillés sous GitHub **[16]**.

Il nécessite comme prérequis l'installation de LimeSuite et SoapySDR et du paquet **`libfftw3-dev`**. Ces logiciels sont déjà installés dans la suite PISDR pour RPi4 permettant de s'affranchir de ces installations délicates.

Le logiciel LimeAuto s'installe de manière classique comme indiqué ci-dessous :

```
$ git clone https://github.com/sdrpower2/limeAuto
$ cd LimeAuto
$ mkdir build
$ cd build
$ cmake ..
$ make
$ sudo make install
```

Une base de données avec les masques d'autocorrélation a été constituée pour les signaux suivants: signaux bande étroite (TETRA, TETRAPOL, MPT1327, POCSAG, DMR, INMARSAT Standard C et STANAG4285) et signaux large bande (LTE, GSM et DAB).

Le logiciel LimeAuto se lance avec la commande indiquée ci-dessous (analyse sur la fréquence 466.050 MHz):

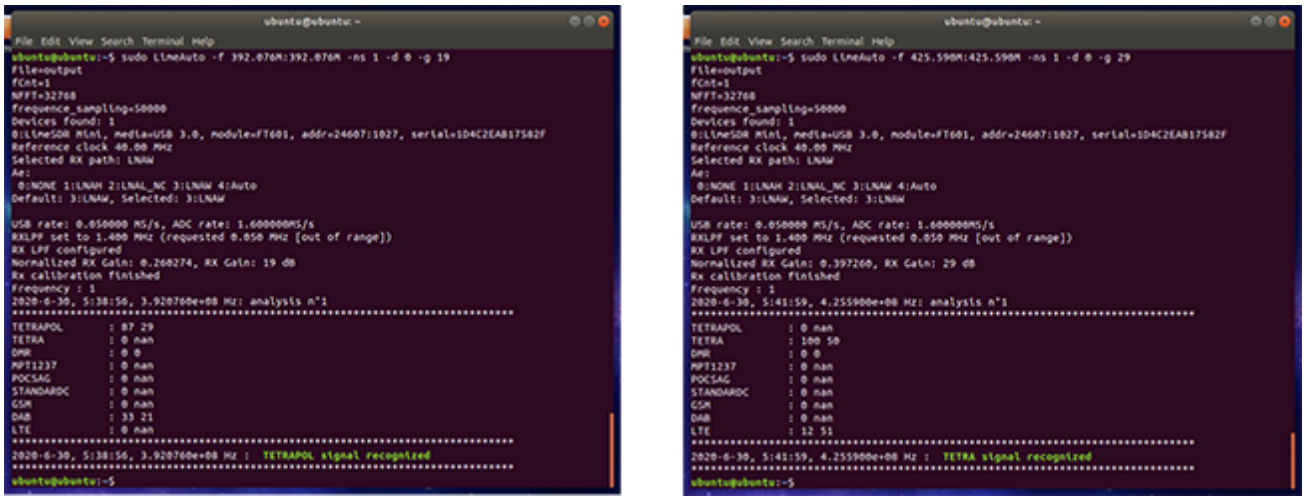
```
$ sudo LimeAuto –f 466.050M:466.050M –ns 1 –d 0 –g 29
```

Elle permet de choisir la fréquence à laquelle l'analyse sera effectuée **`-f`** et de fixer le gain radio en fonction du niveau du signal intercepté **`-g`**.

Le logiciel effectue en boucle l'analyse du signal. Il bouclera donc indéfiniment en attendant de reconnaître un signal en indiquant l'absence de reconnaissance d'un signal lors de chaque tentative d'identification. La recherche sera interrompue par [Crtl]-[C]. Cette boucle permet de réaliser l'analyse sur des signaux intermittents, le logiciel « attendant » l'apparition du signal à identifier.

Un autre mode du logiciel (mode SC pour SCan) permet de scanner une bande de fréquence à la recherche des signaux présents pour les identifier.

Le logiciel a été testé avec succès sur des signaux réels que ce soit sur un PC Linux I3 ou sur un RPi4. La figure 8 montre le résultat de LimeAuto sur deux signaux TETRAPOL et TETRA. Dans le mode standard, les notations obtenues pour tous les masques de la bibliothèque de signaux sont affichées. Cela permet de savoir si l'identification a été franche ou non.



(/sites/default/files/magazines/misc/mischs-022/ASI_ACF_figure_08.png)

Fig. 8 : Identification par LimeAuto.

La vidéo **[18]** montre le résultat de ces tests pour différents signaux : LTE (806 MHz), DAB (204,581 MHz), GSM (958,6 MHz), TETRAPOL (493,475 MHz), TETRA (426,611 MHz), DMR (452,525 MHz), POCSAG (460,050 MHz) et STANDARD-C (1539,675 MHz). Grâce à deux radios, la vidéo visualise le signal intercepté sous GQRX et l'analyse du signal. Le résultat de l'analyse est indiqué en vert quand un signal a été identifié ou en rouge quand aucun signal n'a été identifié (signal inconnu ou absence de signal).

La vidéo démontre aussi le mode Scan dans la bande Tetrapol.

7. Améliorations potentielles

Les améliorations potentielles sont les suivantes:

- optimisation du temps de traitement pour identifier plus rapidement notamment dans le mode SC ;
- capacité à lancer le processus d'identification du signal directement dans GQRX ou SDRAngel ou SigDigger sur un signal d'intérêt sans avoir à sortir du logiciel de monitoring ;
- lancement automatique de la démodulation du signal identifié avec des logiciels comme DSD (pour DMR) ou TetraLive (pour Tetra) pour éviter une étape manuelle ;
- et évidemment, ajouter de plus en plus de signaux numériques à la bibliothèque des signaux.

Conclusion

Les progrès technologiques et le monde du logiciel libre permettent sur le plan technique l'écoute de nombreux signaux radio. Un chiffrement robuste sans vulnérabilité système résiduelle est devenu indispensable et constitue un véritable challenge pour les nouveaux systèmes notamment en termes d'interopérabilité arrière (c'est-à-dire en termes de compatibilité entre anciennes et nouvelles générations d'un même système).

Peu à peu, le monde du logiciel libre a commencé à s'intéresser à l'analyse technique. Il s'intéresse aussi depuis peu à la mesure de l'angle d'arrivée du signal par la méthode du TDOA (*Time Difference Of Arrival*) avec les radios monovoies existantes ou avec des radios multivoies cohérentes permettant la mesure de la différence de phase reçue sur chaque antenne. Le récepteur KerberosSDR qui dispose de quatre voies cohérentes de 2 MHz de bande instantanée, est disponible pour moins de 250 euros avec les logiciels libres associés. Ce récepteur ouvre aussi la voie aux autres applications nécessitant la cohérence de phase comme le radar.

Nul doute que ce monde de la radio et du logiciel libre va continuer à se développer en révolutionnant le monde du COMINT.

Remerciements

À ma tendre et chère épouse ainsi qu'à mes enfants qui m'ont permis de passer le temps nécessaire à chasser le signal et à appréhender le monde passionnant de la SDR.

Références

[1] Interception passive et décodage de flux GSM avec gr-gsm - *MISC HS n°16* : https://connect.ed-diamond.com/MISC/MISCHS-016/Interception-passive-et-decodage-de-flux-GSM-avec-gr-gsm (https://connect.ed-diamond.com/MISC/MISCHS-016/Interception-passive-et-decodage-de-flux-GSM-avec-gr-gsm)

[2] Analyse et interception de flux des téléphones par satellite Iridium - *MISC n°104* : https://connect.ed-diamond.com/MISC/MISC-104/Analyse-et-interception-de-flux-des-telephones-par-satellite-Iridium (https://connect.ed-diamond.com/MISC/MISC-104/Analyse-et-interception-de-flux-des-telephones-par-satellite-Iridium)

[3] LTE Phone Number Catcher: A practical Attack against Mobile Privacy, https://www.hindawi.com/journals/scn/2019/7425235/ (https://www.hindawi.com/journals/scn/2019/7425235/)

[4] https://www.researchgate.net/publication/309264464_LTE_Redirection_Attack_-_Forcing_Targeted_LTE_Cellphone_into_Unsafe_Network (https://www.researchgate.net/publication/309264464_LTE_Redirection_Attack_-_Forcing_Targeted_LTE_Cellphone_into_Unsafe_Network)

[5] The SDR linux distro for Raspberry Pi, https://pisdr.luigifreitas.me/ (https://pisdr.luigifreitas.me/)

[6] DragonOS: Debian Linux with preinstalled open source SDR software, https://sourceforge.net/projects/dragonos-10/ (https://sourceforge.net/projects/dragonos-10/)

[7] https://github.com/csete/gqrx (https://github.com/csete/gqrx) et https://gqrx.dk/ (https://gqrx.dk/)

[8] https://github.com/f4exb/sdrangel (https://github.com/f4exb/sdrangel)

[9] http://www.hdsdr.de/ (http://www.hdsdr.de/)

[10] https://github.com/BatchDrake/SigDigger (https://github.com/BatchDrake/SigDigger)

[11] https://github.com/randaller/cnn-rtlsdr (https://github.com/randaller/cnn-rtlsdr)

[12] Radio signal identification «by ear»: Discrete Fast Fourier audio hashes comparison in Python, https://jcrueda.com/?p=916 (https://jcrueda.com/?p=916)

[13] https://www.sigidwiki.com/wiki/Signal_Identification_Guide (https://www.sigidwiki.com/wiki/Signal_Identification_Guide)

[14] https://sites.google.com/site/g4zfqradio/hdsdr-autocorrelation (https://sites.google.com/site/g4zfqradio/hdsdr-autocorrelation)

[15] https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxnNHpmcXJhZGlvfGd4OjUwMDRiZDRlYjY4MWU3M2E (https://docs.google.com/viewer?a=v&pid=sites&srcid=ZGVmYXVsdGRvbWFpbnxnNHpmcXJhZGlvfGd4OjUwMDRiZDRlYjY4MWU3M2E)

[16] https://github.com/sdrpower2/LimeAuto (https://github.com/sdrpower2/LimeAuto) : l'outil LimeAuto est une modification de LimeScan (https://github.com/myriadrf/lime-tools (https://github.com/myriadrf/lime-tools)) par l'auteur de l'article

[17] https://www.youtube.com/watch?v=5Oc8XUg7aFo&t=21s (https://www.youtube.com/watch?v=5Oc8XUg7aFo&t=21s)

[18] https://www.youtube.com/watch?v=O9Qmykk6N4U&t=25s (https://www.youtube.com/watch?v=O9Qmykk6N4U&t=25s)

Article rédigé par

KD

Kuchly Denis (/auteur/kuchly-denis)

1 articles

(/auteur/kuchly-denis)