Méthodes de stéganographie fondées sur la prise en compte du bruit de capteur

Théo Taburet

N°d'ordre: 402

CENTRALE LILLE

THÈSE

Présentée en vue
d'obtenir le grade de

DOCTEUR

En

Spécialité: Automatique, Génie informatique, Traitement du Signal et image

Par

Théo TABURET

DOCTORAT DÉLIVRÉ PAR CENTRALE LILLE

Titre de la thèse:

Méthodes de stéganographie fondées sur la prise en compte du bruit de capteur

Soutenue le 12 Octobre 2020 devant le jury d'examen:

Président,	Jean-Michel MOREL,	Pr ENS Cachan, Laboratoire CMLA
Rapporteur,	Caroline FONTAINE,	DR CNRS, Laboratoire LSV
Rapporteur,	Florent RETRAINT,	Pr UTT, Laboratoire ICD
Examinateur,	Marc CHAUMONT,	MdC HdR Univ de Nimes, Laboratoire LIRMM
Directeur de thèse,	Patrick BAS,	DR CNRS, Laboratoire CRIStAL
Co-directeur de thèse,	Wadih SAWAYA,	MdC IMT, Laboratoire CRIStAL

Thèse préparée dans le Laboratoire CRISTAL
Ecole Doctorale SPI 072

centralelille Université de Lille CRIStAL Centre de Recherche en Informatique, Signal et Automatique de Lille cnrs

Remerciements

Je souhaiterais en premier adresser mes remcerciements les plus sincères aux membres du jury. Merci à Caroline Fontaine et Florent Retraint d'avoir accepté d'être les rapporteurs du présent manuscrit, ainsi qu'à Marc Chaumont et Jean-Michel Morel d'avoir examiner cette thèse. C'est un réel honneur. Merci pour vos remarques, questions, critiques et encouragements. Dans ce contexte particulier de pandémie du Covid-19, vous vous êtes rendu disponible et vous m'avez permis de soutenir ma thèse en présentiel et j'en très reconnaissant.

Ces trois ans de travaux n'auraient pas pu aboutir sur ce présent manuscript sans les conseils éclairés, la bienveillance et l'appui de mes directeurs de thèse, Patrick Bas et Wadih Sawaya. Ces années ont été extrêmements enrichissantes sur tout les plans, je ne rends à peine compte de la chance qui m'a été offerte, ainsi, merci à Patrick de m'avoir donné cette opportunité il y a un peu plus de trois ans et merci à Wadih pour ses conseils avisé et son temps précieux durant toute la durée de cette thèse. Merci pour m'avoir accordé de l'autonomie et votre confiance.

Merci à l'équipe SigMA: Amine, Arnaud, Ayoub, Barbara, Clément, Christelle, François, Guillaume, Jean-Michel, Kevin, Jérémie, John, Julien, Mahmoud, Nouha, Ouafae, Patrick, Pierre, Pierre-Antoine, Philippe, Quentin, Solène, Rémi, Rémy, Rui, Victor, Vincent, Wadih. J'ai beaucoup apprécié être votre collègue, merci de m'avoir intégré à cette sphère d'individus partageant cafés, Sully burger's, et discussions sur les DPPs.

Merci à Ayoub et Nouha d'avoir partagé ce bureau à Centrale puis au bâtiment ESPRIT, merci pour ces nombreuses heures à refaire le monde.

Merci à Andrew Ker pour ces échanges sur la "Square Root Law" et de m'avoir donné l'opportunité de me rendre Oxford et d'assister à sa chorale.
Merci à Jessica Fridrich de m'avoir m'avoir acceuili au sein de son laboratoire à Binghamton dans lequel j'ai pu y rencontrer ses doctorants et stagiaires.
Merci à Jan Butora, Yassin Yousfi, Quentin Giboulot pour ces moments passés à Binghamton et à Paris, pour ces moments forts amusants et votre aide précieuse.

Merci à Remi Cogranne pour ces échanges, ta sympathies et tes conseils.

Merci à Patrick et Ingrid pour leur hospitalité, je n'oublierai jamais ces soirées d'équipe à dévorer les pizzas de Julien et Laura, le Gravlax de Rémi et ces sessions de saunas.

Merci à tous ceux qui de près ou de loin ont permit de faire en sorte que cette aventure aboutisse à son terme, que ce soit des amis, des voisins ou des collègues.
Merci à mes amis, qui me soutiennent et m'inspirent, que ce soit à distance où pas.
Merci à mes premiers amis Lillois Ydriss, Axel et Tarik, pour votre temps et votre amitié. Merci à Adrien, Camille, Clélia, Florient, Khesen, Lilian, Ludovic, Quentin, Jérémy, Jordan, Jordanne, Nathan, Nicolas, Thomas, Romain, Vivien qui ont si souvent bravé les kilomètres pour me rendre visite. Mention spéciale à Brieuc, Brice, Léa, Léon, Lou, Salma, Fanny, Mélody qui sont là depuis le début. Merci à tous ceux qui à défaut de m'avoir rendu visite étaient connectés sur Facebook à toute heure.
Merci à mes voisins Emilien, Louise, Aline et Eléa pour ces week-ends passés sur les balcons.
Merci à ces nombreux employés de CRIStAL pour leur travail sérieux et leur sympathie.
Merci à mes parents, qui m'ont soutenu et encouragé et qui ont su me transmettre ce goût de la curiosité, merci à mes frères Félix et Pol pour leur présence et leur sympathie.

Merci à mon chat Mawie-Théwèse pour sa douceur hors du commun et dont le minois peuple le manuscrit.

Enfin, par-dessus tout, merci à Gisèle, sans qui ces quelques années Lilloises n'auraient pas eu la même saveur.

Plan

Résumé en français

L'étymologie grecque du terme « stéganographie » est la concaténation des mots « stego »: garder secret et « graphia »: l'écriture.

La stéganographie est donc un terme pour désigner l'art de réaliser une communication secrète (ou discrète). La caractéristique fondamentale de la stéganographie est par essence qu'il doit être impossible pour un système de détection de distinguer les objets anodins de ceux qui contiennent un message secret.

De manière analogue à la cryptographie, dont la discipline duale est la cryptanalyse visant à décrypter le message chiffré, la stéganographie a également sa discipline duale : la stéganalyse. L'objectif de la stéganalyse étant par essence, à minima, de détecter la présence d'un message caché.

Dans le cadre des images digitales on peut définir la perturbation induite par l'insertion d'un message secret sur son image de couverture comme l'ajout d'un signal spécifique, l'indétectabilité de ce message va ainsi reposer sur le fait que le signal ajouté ne va pas perturber les propriétés statistiques de l'image initiale (l'image cover). C'est en partant de ce principe que le paradigme de la Stéganographie Naturelle est née, celle-ci s'attache à utiliser un bruit inhérent aux capteurs photographiques (le bruit photonique) qui peut être modélisé par une loi normale distribuée indépendamment sur chaque photo-site. Le message est ainsi inséré par l'imitation de ce bruit naturel lors de la capture d'une image digitale. De ce fait l'image ainsi générée (l'image stego) dispose des propriétés statistiques d'une image anodine, ce qui garantit une importante sécurité pratique. Les travaux dans ce domaine n'avaient montré jusqu'à présent que des résultats dans le domaine spatial.

Ce manuscrit présente en partie des contributions qui prolongent cette méthode dans le domaine JPEG en exploitant un processus de développement des images très précis. À partir de considérations sur l'indépendance de certains coefficients DCT de l'image, nous avons également pû contribuer à sécuriser des schémas d'insertions classiques.

Pour présenter nos travaux dans ce manuscrit, la présentation des concepts de base (la stéganographie, la stéganalyse, le processus de formation d'une image) est effectuée à travers les deux premiers chapitres. Le premier fournit un état de

l'art en stégnographie et en stéganalyse, le suivant introduit des notions sur le de développement des images.

Les trois chapitres qui les suivent font état de nos contributions. Le chapitre 3 étudie l'origines des dépendances entre coefficients DCT, proposant une méthode pour modéliser celles-ci à l'aide d'une matrice de covariance dont nous avons obtenus une expression analytique.

Le chapitre 4 emploit les résultats du chapitre 3 pour dériver un schéma d'insertion utilisant cette matrice de covariance, permettant une insertion préservant la distribution statistique du bruit de capteur dans le domaine DCT, et préservant de ce fait les dépendances entre les coefficients d'une image.

Enfin à la lumière des chapitres 3 et 4, l'exploitation de la chaîne de développement nous a permis d'élaborer une approche de sécurisation de schémas d'insertion classiques en utilisant des modifications synchronisées.

Nos deux approches ont montré des résultats supérieurs à l'état de l'art, permettant dans le premier cas d'expliciter une méthodologie pour préserver le bruit de capteur lors d'une insertion dans le domaine JPEG, et ouvrant dans le deuxième des possibilités d'amélioration de schémas déjà existants afin qu'ils produisent des images moins détectables qu'auparavant.

Résumé en anglais

The Greek etymology of the term « steganography » is the concatenation of the words « stego »: to keep secret and « graphia »: writing.

Steganography is therefore a term for the science of secret (or discreet) communication. The fundamental characteristic of steganography is by essence that it must be impossible for a detection system to distinguish innocuous objects from those containing a secret message.

Similar to cryptography, whose dual discipline is cryptanalysis to decrypt the encrypted message, steganography also has its dual discipline: steganalysis. The objective of steganalysis is, basically, to detect the presence of a hidden message.

In the context of digital images, the disturbance induced by the embedding of a secret message on its cover image can be defined as the addition of a specific signal. The undetectability of this message will thus be based on the fact that the added signal will not disturb the statistical properties of the initial image (the cover image). It is from this principle that the paradigm of Natural Steganography was born, this one is attached to use a noise inherent to the photographic sensors (the photonic noise) which can be modelled by a normal law distributed independently on each photo-site. The message is thus embedded by imitating this natural noise when capturing a digital image.

Thus the image thus generated (the stego image) has the statistical properties of an anodyne image, which guarantees an important practical security. Work in this field has so far only shown results in the spatial domain.

This manuscript partly presents contributions that extend this method into the JPEG domain by exploiting a very precise image development process. Based on considerations of the independence of some of the DCT coefficients of the image, we have also been able to contribute to the security of classical insertion schemes.

In order to present our work in this manuscript, the basic concepts (steganography, steganalysis, the image development process) are presented through the first two chapters. The first one provides a state of the art in steganography and steganalysis, the next one introduces some concepts on image development.

The three following chapters report on our contributions. Chapter 3 studies the origins of the dependencies between DCT coefficients, introducing a method to

model these dependencies using a covariance matrix from which we have obtained an analytical expression.

Chapter 4 exploits the results of chapter 3 to derive an embedding scheme which use this covariance matrix, allowing an embedding which preserves the statistical distribution of sensor noise in the DCT domain, and thus preserving the dependencies between the coefficients of an image.

Finally, based on chapters 3 and 4, the exploitation of the development chain allowed us to develop an approach to secure classical insertion schemes using synchronized modifications.

Our two approaches have shown superior results to the state of the art, allowing in the first case to elaborate a methodology to preserve sensor noise during embedding in the JPEG domain, and opening in the second case possibilities to improve existing schemes so that they generate images less detectable than before.

Introduction

Dans le présent manuscrit, nous nous intéressons à la stéganographie dans les images numériques naturelles, et plus particulièrement aux images JPEG qui est le format d'images le plus populaire sur internet. On appellera cover le medium de couverture (qui ici sera une image JPEG) et stego (du grec $\sigma\tau\varepsilon\gamma o$ qui peut être traduit par "à garder secret") le résultat de l'opération de stéganographie sur l'image cover après insertion d'un message.

Pour que l'insertion stéganographique ne soit pas détectable, une stratégie possible est de réaliser une insertion de façon à ce que celle-ci perturbe le moins possible le modèle de l'image cover. Cette opération peut-être réalisée de nombreuses façons, comme cela sera abordé dans le chapitre dressant un état de l'art de la discipline. Il est par exemple possible d'obtenir un modèle local utilisé par la suite par des algorithmes dits "adaptatifs" qui analysent les signaux faibles relativement au contenu des images pour réaliser des modifications minimisant un coût heuristique.

À travers ce manuscrit nous verrons qu'au lieu de tenter d'acquérir et d'utiliser un modèle des images ou de leur contenu, nous nous sommes attachés à étudier et utiliser le modèle statistique du bruit issu du capteur employé pour acquérir l'image. Ce bruit étant naturellement purement aléatoire, cette approche avait déjà apporté des résultats très convaincants pour des images spatiales (images brutes de capteur dites "RAW", ou dans le domaine "pixel"). Cependant les travaux pour l'étendre aux images JPEG n'avaient jusqu'à lors pas fourni les mêmes performances.

Les contributions présentées dans ce manuscrit portent sur l'analyse et l'utilisation du modèle statistique du bruit photonique en stéganographie. Les différentes modélisations du bruit photonique dans le domaine DCT permettent d'une part, une extension des travaux dans le domaine spatial au domaine JPEG. Ensuite, nous proposons une dérivation de ce principe d'insertion afin d'améliorer les insertions qui sont réalisées à partir d'algorithmes adaptatifs (dans le domaine JPEG). Le but ici étant d'incorporer des informations additionnelles afin que les images ainsi produites soient cohérentes au sens du bruit de capteur.

La section suivante apporte plus de précision sur le contenu du manuscrit.

1 Organisation du manuscrit

Le présent manuscrit se décompose en 6 chapitres, les deux premiers chapitres 1 et 2 permettent ici de poser les briques nécessaires à la compréhension des chapitres suivants. Les chapitres 3, 4 et 5 présentent l'ensemble des contributions dont l'essentiel a été publié dans des conférences et dans un journal international.

Enfin le dernier chapitre 6 fait office de conclusion, les perspectives d'améliorations y sont également mentionnées.

Le chapitre 1 présente un état de l'art en stéganographie et stéganalyse consignant les différentes notions de base de ces deux disciplines. Nous y abordons ainsi la notion de détectabilité, puis nous introduisons les solutions permettant de minimiser la détectabilité, en préservant le modèle des images ou en minimisant des coûts d'insertion. Nous évoquons également la synchronisation des modifications produites lors de l'insertion afin de réduire encore la détectabilité. Cette stratégie sera améliorée dans le chapitre 5. Une section traitant de la stéganalyse permet de présenter les outils de l'état de l'art mais l'accent est ici essentiellement porté sur l'évocation de la méthodologie d'évaluation dont nous avons eu l'usage en stéganographie.

Le chapitre 2 est dédié à l'acquisition d'images par des capteurs photographiques et à leur processus de développement. Il fournit les ingrédients nécessaires à l'appréhension des contributions qui suivent en insistant notamment sur le processus de dématriçage, l'usage de la transformée en cosinus discret (DCT), et plus généralement le processus de compression JPEG. Ce chapitre fournit également une première présentation des propriétés associées au bruit de capteur.

Le chapitre 3 présente l'ensemble de nos travaux réalisés pour modéliser le processus de développement des images RAW en JPEG en insistant sur leurs propriétés statistiques. En effet, du point de vue stéganographique, il est essentiel d'identifier les composantes qui sont dépendantes les unes des autres car leurs modifications entraînent nécessairement une dissemblance entre les propriétés statistiques de l'image stego par rapport à celle de sa version cover.

Dans ce chapitre nous étudions en premier lieu l'indépendance de bruit de capteur sur des photo-sites voisins, nous adopterons cette hypothèse de travail comme une affirmation. Puis, nous questionnons la véracité de l'adoption d'un modèle gaussien pour modéliser les réalisations de ce bruit. Nous définissons ensuite un cadre d'étude, composé d'un processus de développement linéaire des images, nous permettant une modélisation du bruit après développement par un modèle gaussien multivarié.

Ce modèle nous permet d'expliquer et de modéliser les dépendances entre les différents coefficients d'une image JPEG.

À partir de la chaîne de développement étudiée nous proposons en premier lieu un calcul explicite des dépendances entre coefficients (DCT) à partir d'une matrice de covariance dont nous fournissons l'expression analytique. Puis, nous exploitons un processus de développement plus général à partir duquel nous montrons qu'il décorrèle certains coefficients, ouvrant ainsi la possibilité de réaliser des insertions indépendantes.

Les contributions relatives à l'écriture analytique de la matrice de covariance et l'explication de l'origine des dépendances ont donné lieu notamment à une publication dans un journal international [93] et ont été exploitées dans un article [81] de conférence.

Le chapitre 4 présente un schéma d'insertion que nous avons dérivé des contributions du chapitre 3 en considérant un processus de développement linéaire des images.

Il s'agit d'un schéma d'insertion par stéganographie naturelle (imitant un changement de sensibilité ISO du capteur) préservant les dépendances entre les coefficients DCT d'une image JPEG lors de l'une insertion. Dans un premier temps nous utiliserons ce schéma pour produire des images stegos JPEG à partir de matrices de covariance calculées analytiquement. Enfin nous exploiterons ce schéma à partir d'une matrice de covariance estimée au préalable et ajustée en conséquence. Les performances en termes de sécurité empirique nous permettent d'appuyer la pertinence de notre approche. Ces résultats ont été publiés dans plusieurs conférences [85] [86] [81], et ont inspiré la méthodologie détaillée au chapitre 5. Le chapitre contient également une dérivation non publiée de l'approche d'insertion utilisant une matrice de covariance calculée analytiquement dans le domaine couleur.

Le chapitre 5 propose d'utiliser l'analyse statistique des corrélations (étudiée dans le chapitre 3) entre coefficients DCT pour concevoir une nouvelle stratégie de synchronisation des modifications pouvant être utilisée pour des schémas stéganographiques exploitant des coûts dans le domaine JPEG. Dans ce chapitre nous proposerons une approche pour dériver à partir de ces corrélations, des coûts d'insertion permettant de préserver les dépendances liées au processus de développement. Il est important de remarquer ici que cet algorithme exploite les réalisations précédentes pour effectuer les suivantes. Cette approche nous permet ainsi d'exploiter de l'information adjacente, renforçant de ce fait la sécurité de nos insertions. Ces travaux ont donné lieu à une publication à la conférence IH&MMSec'20 [82].

Le chapitre 6 dresse les conclusions de ces trois années de recherche et évoque des pistes d'améliorations.

2 Contributions

Journaux

2020: T Taburet, P Bas, W Sawaya, J Fridrich. "Natural Steganography in JPEG Domain with a Linear Development Pipeline" [93]. IEEE Transactions on Information Forensics & Security.

Conférences

2020: T Taburet, P Bas, W Sawaya, R Cogranne, "JPEG Steganography and Synchronization of DCT Coefficients for a Given Development Pipeline" [83]. IH&MMSec '20: ACM Workshop on Information Hiding and Multimedia Security.

2019: T Taburet, P Bas, W Sawaya, J Fridrich, "Computing dependencies between DCT coefficients for natural steganography in JPEG domain" [81]. IH&MMSec '19: ACM Workshop on Information Hiding and Multimedia Security.

2019: T Taburet, P Bas, W Sawaya, J Fridrich, "Stéganographie naturelle pour images JPEG" [86]. Colloque GRETSI (Groupement de Recherche en Traitement du Signal et des Images).

2018: T Taburet, L Filstroff, P Bas, W Sawaya, "An Empirical Study of Steganography and Steganalysis of Color Images in the JPEG Domain" [87]. Electronic Imaging'19 .

2018: T Taburet, P Bas, W Sawaya, J Fridrich, "A natural steganography embedding scheme dedicated to color sensors in the JPEG domain" [84]. IWDW, International Workshop on Digital Forensics and Watermarking.

Workshops

2019: Participation aux REDOCS'19 (Rencontre Entreprises DOCtorants en Sécurité) sur le sujet proposé par Thalès.

Recherche reproductible

2020: T Taburet, P Bas, W Sawaya, J Fridrich. "Natural Steganography in JPEG Domain with a Linear Development Pipeline" [93]. IEEE Transactions on Information Forensics & Security. L'implémentation est disponible à: https://gitlab.cristal.univ-lille.fr/ttaburet/tifs-paper

2020: Notebooks associés aux implémentations des contributions disponible à: https://gitlab.cristal.univ-lille.fr/ttaburet/thesis-theo-notebooks

Notations

Afin de faciliter la compréhension, et pour éviter toute confusion, nous fixons dans cette section les différentes notations que nous utiliserons tout au long de ce manuscrit.

Nous utilisons des majuscules pour les variables aléatoires X et leurs symboles minuscules correspondants pour leurs réalisations x. Les matrices sont écrites en caractères gras majuscules $\mathbf{A}$ et des vecteurs (de variables scalaires ou aléatoires) en caractères gras minuscules $\mathbf{a}$. La transposition de la matrice est indiquée par l'exposant $\mathbf{A}^t$. Les indices $\square_p$ et $\square_d$ seront respectivement associés au domaine du photo-site et au domaine DCT.

Opérations Dans cet article, la vectorisation des matrices selon les lignes ou les colonnes est utilisée. Pour une $m \times n$ matrices $\mathbf{A}$, la vectorisation par lignes et colonnes respectivement est définie comme suit:

Pour:

$$\mathbf{A} \in \mathbb{R}^{m \times n} / \mathbf{A} = \begin{bmatrix} a_{1,1} & \dots & a_{1,n} \\ \vdots & & \vdots \\ a_{m,1} & \dots & a_{m,n} \end{bmatrix} \tag{1}$$

la vectorisation par les colonnes (C) et respectivement par les lignes (R) est définie comme ce qui suit:

$$\mathrm{vec_C}(\mathbf{A}) = [a_{1,1}, \dots, a_{m,1}, \dots, a_{1,n}, \dots a_{m,n}] \in \mathbb{R}^{mn} \tag{2}$$

$$\mathrm{vec_R}(\mathbf{A}) = [a_{1,1}, \dots, a_{1,n}, \dots, a_{m,1}, \dots a_{m,n}] \in \mathbb{R}^{mn} \tag{3}$$

Symboles

- $\delta_{i,j}$ est le symbole de Kronecker défini tel que:

$$\delta_{i,j} : (i,j) \to \begin{cases} 0 & \text{si } i \neq j \\ 1 & \text{sinon} \end{cases} \tag{4}$$

- $\stackrel{d}{=}$ représente l'égalité de distribution de deux variables aléatoires.
- $\lfloor x/y \rfloor$ est le résultat de la division euclidienne de x par y.

- $x \bmod y$ est le reste de la division euclidienne de x par y.
- $\otimes$ représente le produit de Kronecker, pour $\mathbf{A} \in \mathbb{R}^{m\times n}$ et $\mathbf{X} \in \mathbb{R}^{p\times q}$

$$\mathbf{A} \otimes \mathbf{X} = \begin{pmatrix} a_{1,1} \cdot \mathbf{X} & \dots & a_{1,n} \cdot \mathbf{X} \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot \mathbf{X} & \dots & a_{m,n} \cdot \mathbf{X} \end{pmatrix} \in \mathbb{R}^{mp\times nq}, \tag{5}$$

- $\oplus$ est la somme directe, définie (avec les notations précédentes) comme:

$$\mathbf{A} \oplus \mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{X} \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & x_{11} & \cdots & x_{1q} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & x_{p1} & \cdots & x_{pq} \end{bmatrix} \tag{6}$$

- $\odot$ est le produit coefficient par coefficient entre deux matrices de mêmes dimensions.
- $\oslash$ est la division coefficient par coefficient entre deux matrices de mêmes dimensions.
- La multiplication et division des blocs $(k \times k)$ (pour $k \in \mathbb{N}^*$) d'une matrice $\mathbf{X} \in \mathbb{R}^{k\cdot n\times k\cdot n}$ par une matrice $\mathbf{Q} \in \mathbb{R}^{n\times n}$ sont définies respectivement comme:

$$\mathbf{X} \odot_{n,n} \mathbf{Q} = \mathbf{X} \odot_{n,n} (\mathbf{1}_{k,k} \otimes \mathbf{Q}), \tag{7}$$

$$\mathbf{X} \oslash_{n,n} \mathbf{Q} = \mathbf{X} \oslash_{n,n} (\mathbf{1}_{k,k} \otimes \mathbf{Q}). \tag{8}$$

Le reste des notations sera abordé dans les chapitres et sections concernés.

Chapitre 1

Principes de bases en stéganographie et stéganalyse

3 Introduction

3.1 Préambule

Un matin, Bob, reçoit dans un SMS d'un numéro inconnu signé d'une certaine Alice. Le SMS contient un message, deux phrases d'une longueur proche :

> « Elle se prélasse chez bibi, tranquille non ?
> Wesh maman craque depuis huit mois... »

À première vue, il peut sembler que cet échange soit une simple communication en argot entre deux membres d'une même famille ayant pour objet une personne actuellement chez Alice dont l'attitude passive tourmenterait la mère de celle-ci.

Cependant Bob est un informateur et Alice une employée d'une agence gouvernementale : ces deux individus avaient convenu en amont d'une syntaxe pour leurs communications. La syntaxe utilisée par Alice correspond ici à une position pour un rendez-vous. En effet, en extrayant uniquement la première lettre de chaque mot, nous obtenons pour chaque phrase :

> E S P C B T N
> W M C D H M

En associant pour chaque caractère (de A à Z), un indice allant de 0 à 25, et, en faisant correspondre l'indice de chacune des lettres à son reste par la division euclidienne par 10, les deux chaînes de caractères deviennent :

> 4 8 5 2 1 9 3
> 2 2 2 3 7 2

Bob n'a plus qu'à rajouter les coordonnées manquantes pour rendre intelligible cette suite numérique, obtenant ainsi les coordonnées suivantes :

48°52'19.3 N
2°22'37.2 E

La position envoyée par Alice correspond à la station de métro « Belleville » à Paris, où se croisent la ligne n°2 et la ligne n°11 du métro.

Cet exemple montre une application de la stéganographie, à savoir le fait de cacher une communication, potentiellement sensible, entre deux parties. De plus, cette communication est secrète puisqu'il est à priori impossible pour un tiers qui ne disposerait pas de la syntaxe de décodage de saisir le réel sens de la communication entre Alice et Bob.

3.2 Principes Généraux

Nous proposons une définition de la stéganographie à partir de l'exemple précédent :

> *La stéganographie est l'art de communiquer un message secret à travers un message anodin sans que la communication soit détectable.*

Pour la suite du manuscrit notre objet de couverture sera une image digitale appelée 'cover' et notée $\mathbf{X}$, Alice sera la personne communiquant un message secret à travers une image 'stego' créée à partir de l'image cover et notée $\mathbf{Y}$.

Nous supposons que Alice va communiquer cette image à travers un canal de communication public et non-bruité. Un troisième acteur, Eve qui peut être vue comme un adversaire, sera en mesure d'avoir accès aux images échangées et cherchera à détecter la communication secrète. Dans le scénario que nous considérons, Eve est passive, car elle ne va pas pouvoir modifier les images mais uniquement les lires.

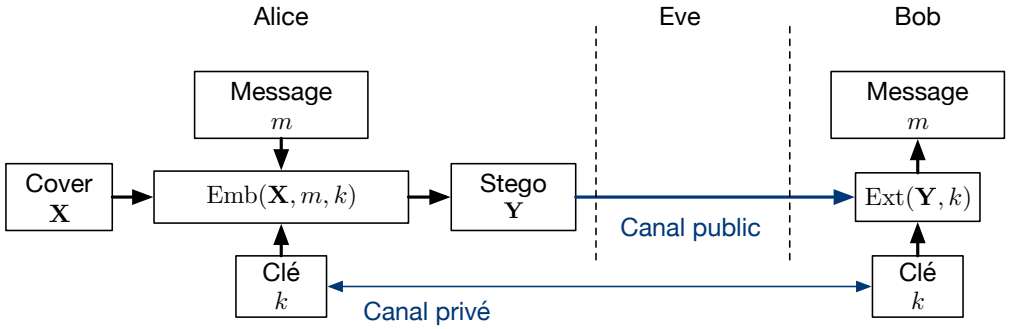
Enfin Bob, comme expliqué à la figure 1.1, va à partir de l'image 'stego' reçue via le canal de communication établi entre lui et Alice, pouvoir extraire le message initialement inséré.

Afin de garantir qu'une tierce personne ne puisse pas avoir accès au message échangé, Alice et Bob partagent une clef commune k leur permettant d'insérer via l'opération

$$\mathbf{Y} = \mathrm{Emb}(\mathbf{X}, m, k), \tag{1.1}$$

le message m dans l'image cover $\mathbf{X}$, et dans un second temps extraire le message m de l'image stego via l'opération

$$m = \mathrm{Ext}(\mathbf{Y}, k). \tag{1.2}$$



(a)

Figure 1.1: Composition d'un canal stéganographique.

3.3 Définition de la sécurité en stéganographie

Puisque le but de la stéganographie est de faire en sorte que la communication entre Alice et Bob reste secrète, la notion de sécurité questionne l'essence même de cette discipline.

Lors du développement de nouveaux algorithmes de stéganographie et de stéganalyse, il est ainsi essentiel de pouvoir mesurer leurs performances en termes de sécurité, afin de pouvoir les comparer équitablement entre eux et avec l'état de l'art.

Il existe deux approches principales à ce problème : une approche théorique pour laquelle l'image cover et sa fonction d'insertion peuvent être décrites de manière analytique (voir sous-section 3.3.2), et dans le cas contraire une approche empirique (voir sous-section 3.3.1). Ces deux approches sont présentées dans la suite de cette section.

3.3.1 Sécurité théorique

En 1998, C. Cachin a proposé une définition théorique de la sécurité d'un schéma d'insertion stéganographique décrite ci-dessous.

Si nous notons $P_{\mathbf{X}}$ et $P_{\mathbf{Y}}$ les distributions discrètes respectives de l'image cover $\mathbf{X}$ et stego $\mathbf{Y}$, nous assimilons ainsi covers et stegos à des réalisations de variables aléatoires X et Y respectivement, avec $X \sim P_{\mathbf{X}}$ et $Y \sim P_{\mathbf{Y}}$. En posant $\mathcal{A}$ le support de ces deux distributions de probabilités, l'auteur définit la sécurité en stéganographie comme la divergence de Kullback-Leibler (D_{KL}) entre $P_{\mathbf{X}}$ et $P_{\mathbf{Y}}$ sur $\mathcal{A}$:

$$D_{\mathrm{KL}}(P_{\mathbf{X}} \| P_{\mathbf{Y}}) = \sum_{a \in \mathcal{A}} P_{\mathbf{X}}(a) \log \left(\frac{P_{\mathbf{X}}(a)}{P_{\mathbf{Y}}(a)} \right). \tag{1.3}$$

De ce fait, disposant de la distribution des images covers et des images stegos nous pouvons définir la sécurité du schéma d'insertion comme sa capacité à générer des images stegos dont la distribution reste proche, au sens de la divergence, de celle des images covers.

Ainsi, en utilisant cette métrique, un schéma stéganographique est dit ϵ-sûr si $D_{\mathrm{KL}}\left(P_{\mathrm{c}} \| P_{\mathrm{s}}\right) < \epsilon$ et parfaitement sûr si $D_{\mathrm{KL}}\left(P_{\mathrm{c}} \| P_{\mathrm{s}}\right) = 0$.

Comme nous le verrons plus tard dans ce mémoire, cette mesure théorique est une source d'inspiration pratique puisque si Alice a accès à la distribution de la cover ou à un modèle précis de celle-ci, elle est alors en mesure d'optimiser l'opération d'insertion ($\mathrm{Emb}(\mathbf{X}, m, k)$) sous contrainte de minimisation de D_{KL} de l'image stego par rapport à l'image cover.

En utilisant ce formalisme, la sécurité en stéganographie peut également être formulée comme un test d'hypothèse, tel que :

$$\begin{aligned} \mathrm{H}_0 : &\quad \mathbf{X} \sim P_{\mathrm{c}}, \\ \mathrm{H}_1 : &\quad \mathbf{X} \sim P_{\mathrm{s}}. \end{aligned} \tag{1.4}$$

Afin de mettre en œuvre la détection d'images stego, Eve peut alors chercher à construire un détecteur dont le but est de réaliser une prédiction sur la nature d'une image afin de tenter de détecter si l'image est de nature « cover » ou « stego ». Le détecteur est ainsi totalement définit par sa frontière de décision et par un seuil qui lui permettra de réaliser une prédiction sur la nature de l'image : cover ou stego. Eve va ensuite par exemple tenter de minimiser l'erreur de détection du détecteur P_E (métrique qui sera utilisée tout au long du document).

Il est important de préciser que la définition de la sécurité en stéganographie peut ainsi servir de fil conducteur lors de la construction d'un schéma d'insertion. Le problème tel qu'il a été formulé par Cachin (voir [15]) cherche ainsi à préserver le modèle de l'image cover après insertion.

Comme nous le verrons plus amplement dans la prochaine section 4, cette définition de la sécurité a cependant inspirée de nombreux schémas de stéganographie, nous pouvons par exemple citer MBS (Model-Based Steganography), HUGO, MiPOD et les algorithmes de stéganographie naturelles (NS). En effet :

- Dans le cas de MBS (voir aussi la section 4.2.1) l'auteur s'est attaché à développer un algorithme d'insertion préservant l'histogramme des coefficients DCT AC de l'image cover en modélisant les coefficients DCT AC comme les réalisations d'une distribution de Cauchy généralisée.
- De manière analogue les algorithmes de stéganographie naturelle (voir aussi la section 4.2.1) trouvent leur origine dans une insertion imitant les propriétés statistiques du bruit photonique.

Il convient cependant de préciser que la complexité des images naturelles rend la description statistique de celles-ci par des modèles simples peu aisée. En effet leur hétérogénéité et leur non-stationnarité (notamment dans le cas de textures)

représente un obstacle de taille dans la quête d'un modèle local précis.

L'absence de modèle laisse ainsi une porte ouverte à des modèles heuristiques et plus récemment à des modèles utilisant des réseaux de neurones profonds (voir la section 4.2.1). Ces schémas n'étant pas basés sur une définition théorique de la sécurité : il faut ainsi trouver une autre méthode d'évaluation de la sécurité, cette-ci fois plus générale.

3.3.2 Sécurité pratique

En pratique, le succès d'Eve lorsqu'elle attaque le canal de communication stéganographique réside dans la possibilité de distinguer les images covers des images stegos avec une probabilité qui soit supérieure à celle d'un choix aléatoire.

Cette décision est réalisée en pratique par l'utilisation d'un classifieur automatique qui va chercher à partir d'un modèle d'images non-suspectes de détecter celles qui s'en écartent : les images stegos. Cette tache de détection d'anomalies implique de pouvoir connaître, apprendre ou estimer le modèle des images cover et stegos, par exemple par apprentissage supervisé. Ainsi la performance d'un système de stéganalyse est étroitement liée à la précision des modèles des images (cover et stego) et par conséquent à leur source.

La sécurité pratique est souvent évaluée par un score de mise en échec d'un classifieur de l'état de l'art. La métrique associée est une probabilité d'erreur, notée par exemple P_E dont la définition est plus largement détaillée à la Section 5.2.

Par conséquent, le stéganographe (Alice) a tout intérêt à maximiser cette probabilité d'erreur (afin que les prédictions d'Eve reviennent à un choix aléatoire, e.g. à $P_E = 50\%$), tandis que le stéganalyste : Eve, va à l'opposé tenter de minimiser cette valeur.

Dans ce scénario nous pouvons comprendre que la sécurité du schéma d'insertion va dépendre des outils dont Eve dispose pour mettre en échec la discrétion de celui-ci. En utilisant le principe de Kerckhoffs, Eve est supposée connaître le protocole d'insertion (algorithme et taille de messages) et elle est ainsi en mesure de cibler son activité sur ce protocole afin de mieux y faire face (en exploitant les faiblesses de celui-ci), il s'agit ainsi ici de « Stéganalyse clairvoyante » et c'est le scénario qui sera utilisé tout au long de ce manuscrit sauf mention contraire.

Les taux d'erreurs et le principe de Kerckhoffs vont donner lieu à une méthodologie commune de l'évaluation d'un algorithme d'insertion face à un stéganalyste qui sera défini de la façon suivante.

- Dans un premier temps le stéganalyste pourra sélectionner la source des images covers (par exemple BOSSBase, MONOBase, E1Base...),
- il pourra ensuite reproduire le protocole d'insertion du stéganographe : choix de l'algorithme d'insertion et insertion avec des messages d'une taille fixe.

- ensuite le stéganalyste pourra construire le détecteur selon le type de stéganalyse qu'il souhaite employer : la stéganalyse statistique, la stéganalyse par caractéristiques dont la difficulté majeure réside dans la modélisation des images par un ensemble de caractéristiques artificielles, ou obtenus par un réseau de neurones (voir section 5).
- Enfin le stéganalyste évalue ses capacités de détection par la métrique de son choix.

4 Principes de base en stéganographie

Cette partie du chapitre présente les briques de base de la stéganographie d'images numériques, à savoir :

- les schémas types d'insertion stéganographique,
- les deux stratégies cherchant à favoriser l'indétectabilité statistique (préservation du modèle et calcul de coûts),
- des exemples de schémas dans le domaine spatial,
- des exemples de schémas dans le domaine JPEG,
- les méthodes d'insertion adverses,
- les méthodes synchronisant les modifications,
- l'utilisation de l'information adjacente,
- les briques de base de la stéganographie couleur,
- enfin, les notions élémentaires de codage pour la stéganographie.

De nombreux éléments de ces briques seront ensuite repris dans les contributions proposées dans la deuxième partie de ce manuscrit (chapitres 3 à 5).

4.1 Schéma général

À travers cette section nous allons présenter les principes de bases en stéganographie d'images numériques. Nous nous placerons dans le cas où Eve, le gardien, est passif et a connaissance du protocole d'insertion (voir sous-section 3.3.2). Eve n'aura qu'un accès en lecture au canal de communication entre Alice et Bob (voir Schéma 1.1). Dans la section précédente nous avons évoqué la problématique principale de la stéganographie, à savoir la communication secrète d'un message. La discrétion de la communication représente ici un enjeu majeur lors de la réalisation d'une insertion.

Alice va de ce fait chercher à minimiser la détectabilité du message qu'elle va insérer tout en maximisant sa taille, face à un gardien Eve qui va tout mettre en œuvre pour être capable de déceler les artefacts générés par l'activité d'Alice.

Depuis la fin des années 90, les algorithmes d'insertions par modification de l'image cover proposés suivent généralement le processus suivant :

1. Identification du modèle/paramètre(s)/caractéristique(s) à préserver.
2. Insertion du message.

Ces étapes sont brièvement abordées dans les sections qui suivent et illustrées par des exemples de l'état de l'art afin de mieux situer les apports de la littérature.

D'un point de vue pratique, afin de maximiser la sécurité, deux stratégies sont communément utilisées :

- la préservation d'un modèle de l'image cover,
- la minimisation de l'impact de notre insertion en pondérant le coût des modifications effectuées.

La figure 1.2 illustre les différentes directions possibles pour la création d'un schéma d'insertion pratique.

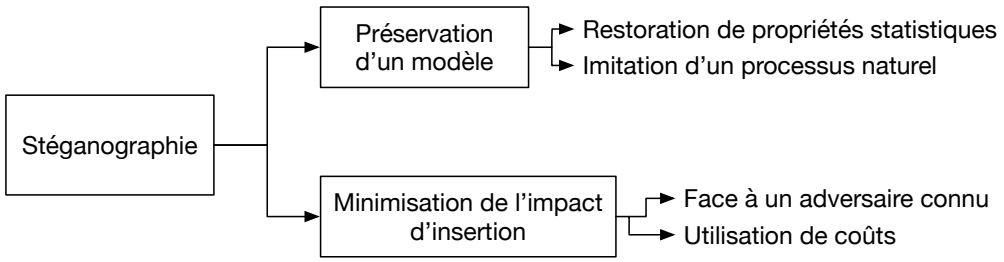


Figure 1.2: Schéma général des méthodes pratiques d'insertion en stéganographie moderne.

Dans les sections qui suivent nous présenterons également les méthodes d'insertions qui nous semblent les plus pertinentes en proposant un aperçu sur les algorithmes qui leur sont associées. Nous insisterons sur les techniques d'insertion par préservation du modèle, par minimisation d'un coût d'insertion et par synchronisation.

4.2 Stratégies permettant de minimiser la détectabilité statistique

Depuis une décennie, la plupart des algorithmes d'insertion stéganographique cherchent à préserver un modèle mathématique pour certains ou de préserver un modèle local pour d'autres. **Dans le premier cas** le stéganographe insère son message en garantissant que l'image stego suive le même modèle supposé connu de l'image cover, rendant l'insertion indétectable par rapport à ce modèle. **Dans le second cas** le stéganographe cherche à préserver le modèle local sans avoir de modélisation exacte de celui-ci, et ce, via l'attribution à chaque échantillon (pixel ou coefficient DCT) d'un coût de modification selon certaines considérations heuristiques ou mathématiques.

4.2.1 Stéganographie par préservation d'un modèle mathématique

Dans cette sous-section nous donnerons deux exemples d'algorithmes s'attachant à préserver le modèle mathématique de l'image cover.

L'algorithme MBS développé par Phill Salee s'attache à préserver des caractéristiques du 1er ordre (histogramme) des coefficients DCT alors que la stéganographie naturelle (« NS ») a pour but de préserver la distribution du bruit photonique.

MBS : « Model Based Steganography » Cet algorithme part du constat qu'en 2003, les caractéristiques de stéganalyse capables de mettre en échec des algorithmes d'insertion tels que F5 [95] s'appuient uniquement sur des histogrammes des coefficients DCT.

Comme illustré à la figure 1.3, la distribution des coefficients DCT AC sur des images naturelles est caractérisée par une symétrie relative (autour de 0), un maximum en 0 et une décroissante rapide à gauche et droite de son maximum. Dans son papier [78], P. Salee propose alors de modéliser la distribution des coefficients DCT par une distribution de Cauchy généralisée.

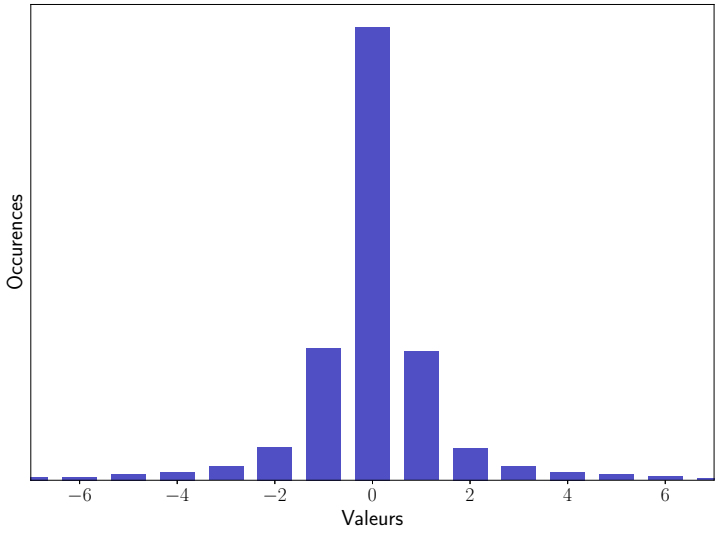


Figure 1.3: Histogramme des coefficients AC DCT coefficient de l'image d'exemple

L'auteur, décrit un schéma d'insertion ne modifiant que les bits de poids faibles (LSB) des coefficients afin d'y insérer un message. Pour cela l'image cover est ainsi divisée en deux composantes, $\mathbf{x} = (\mathbf{x}_{inv}, \mathbf{x}_{emb})$ avec $\mathbf{x}_{inv}$ les éléments invariants par insertion et $\mathbf{x}_{emb}$ les éléments susceptibles d'être modifiés par l'insertion. L'auteur construit ainsi le modèle des images covers par les probabilités conditionnelles $P(\mathbf{x}_{inv}|\mathbf{x}_{emb})$. Ces probabilités sont essentielles pour l'insertion et l'extraction du message.

Ainsi dans un premier temps, le message est inséré en utilisant les symboles de $\mathbf{x}_{emb}$, ces symboles sont ensuite modifiés de sorte à valider $P\left(\mathbf{x}_{inv}|\mathbf{x}_{emb}\right)$. Le modèle utilisé est basé sur une description paramétrique des marginales associées aux coefficients DCT. Les coefficients DC (caractérisants la luminance) ne sont pas modifiés, car ils sont peu aisément descriptibles par un modèle paramétrique et leur modification engendrerait des artefacts entre les blocs. De la même manière, les coefficients initialement nuls ne seront pas modifiés car leur modification risquerait d'engendrer des artefacts visibles. Enfin, les coefficients AC restants sont modélisés par la fonction de densité suivante qui est modélisée ici comme par forme spécifique d'une distribution de Cauchy généralisée :

$$P(u) = \frac{p-1}{2s}(\left|\frac{u}{s}\right| + 1)^{-p}, \tag{1.5}$$

avec u la valeur du coefficient et $p > 1$, $s > 0$. Ainsi l'auteur peut en utilisant uniquement $\mathbf{x}_{inv}$ ajuster la distribution de ses modifications pour correspondre au modèle choisi et l'utiliser pour coder un message m_{bits} tout en conservant les statistiques du modèle $P\left(\mathbf{x}_{inv}|\mathbf{x}_{emb}\right)$.

Le codage est effectué en utilisant le codage arithmétique afin que chaque symbole suive la loi de probabilités prescrite par le modèle.

Stéganographie naturelle (NS): La stéganographie naturelle est basée sur le principe que la stéganographie par conservation de modèle puisqu'elle insère un message dont le signal stego associé tente d'imiter les propriétés statistiques du bruit photonique de l'appareil photo numérique, c'est-à-dire le bruit d'acquisition, qui est lui lié à la sensibilité ISO du capteur.

Plus formellement, pour un paramètre de sensibilité ISO_1 donné, le bruit photonique $N_{i,j}^{(1)}$ appliqué sur chaque photo-site de position (i, j) peut être approximé par un bruit indépendant distribué selon une loi normale et dont la variance est directement en relation affine avec la valeur moyenne du photo-site $\mu_{i,j}$ comme illustré à la figure 1.4.

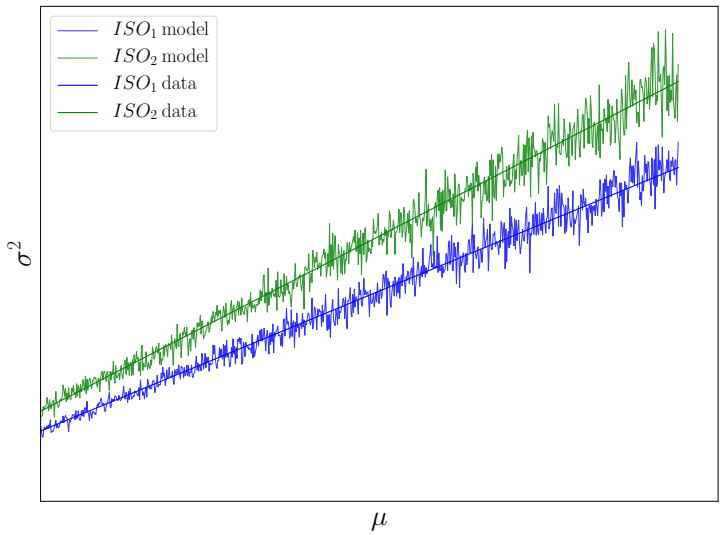


Figure 1.4: Relation linéaire entre la variance et la moyenne non-bruitée des photo-sites

En partant d'une image de couverture acquise à ISO_1, l'insertion est conçue de telle sorte que l'image que l'image stego ressemble à une image acquise à une sensibilité $ISO_2 > ISO_1$ plus importante. Cette stratégie est appelée stéganographie par changement de source [9] car elle repose sur le changement du modèle de la source (cover) pendant le processus d'insertion. Elle permet ainsi de générer des images stegos distribuées comme des images covers capturées à ISO_2, le processus de stéganalyse peut alors être résumé schématiquement par la figure 1.5.

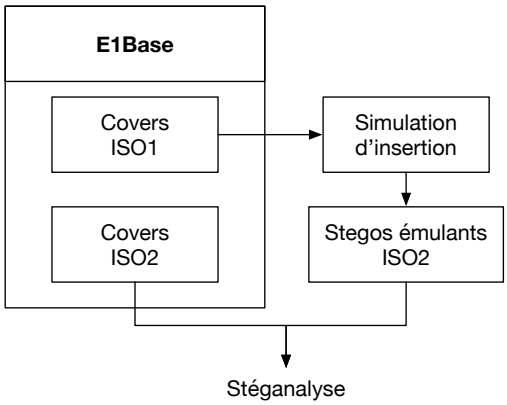


Figure 1.5: Configuration du processus de stéganalyse lors de l'évaluation des performances d'un algorithme d'insertion par changement de source.

Cette idée a été exploitée dans le domaine spatial depuis 2016 par P. Bas tout

d'abord pour un capteur monochrome (Leica M Monochrome Type 2030), et en 2018, T. Denemark et al. [31] se sont attachés à poser les premiers fondements de la stéganographie naturelle dans le domaine JPEG. Pour une image au format RAW, les auteurs estiment par une méthode de Monte-Carlo les fonctions de masse de probabilité pour chaque coefficient DCT afin de réaliser une pseudo-insertion où chaque coefficient est distribué, selon sa loi marginale, comme une cover capturée à ISO_2 après développement (dématriçage et transformée DCT).

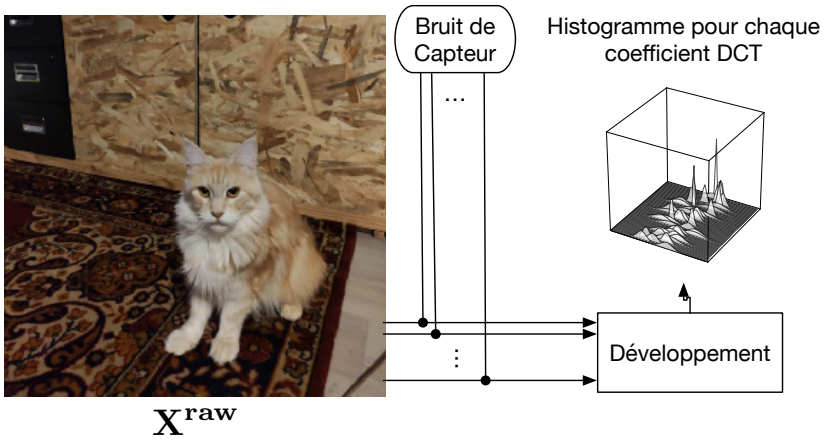


Figure 1.6: Estimation de la moyenne et de la variance des coefficients DCT d'une image par méthode de Monte-Carlo.

Si la mise en œuvre de cette méthode est aisée dans le cas d'un capteur monochrome, elle s'avère beaucoup plus compliquée lorsque les images sont acquises par un capteur couleur. Cette difficulté provient notamment du processus de développement de l'image créant des dépendances entre pixels, annulant de ce fait l'hypothèse d'indépendance du bruit photonique après traitement. Dans le cas d'un capteur couleur cette approche ne s'est pas non plus avérée pertinente, car elle ne prend pas en compte les dépendances induites par le reste du développement. La complexité de cette méthode réside ici dans la modélisation de ce signal après développement dans le domaine DCT.

Nous verrons dans la suite du manuscrit, notamment dans le chapitre 4, comment appliquer cette méthode dans le domaine JPEG pour des images en niveaux de gris et couleurs, en assurant une capacité élevée (la taille du message divisé par le nombre de coefficient modifiables) et une sécurité empirique proche de $P_E = 50\%$.

Cette approche permet également de définir des méthodes efficaces de synchronisation permettant d'exploiter des informations supplémentaires sur la source des images pour préserver des corrélations qui peuvent coexister entre les coefficients DCT comme cela est réalisé au chapitre 5. Cependant, l'état de l'art contient quelques schémas de synchronisation ayant également permis une augmentation de

la sécurité empirique mais basés sur des heuristiques (voir section 4.6).

4.2.2 Stéganographie par minimisation du coût d'insertion

Nous avons dans la section précédente présenté les éléments de base d'un schéma de stéganographie basée sur la conservation d'un modèle, mais qui en modifiant l'image cover va nécessairement introduire des artefacts. Or, pour un pixel donné et selon son voisinage, la valeur de la modification et la dynamique de celle-ci n'aura pas le même impact sur la détectabilité de l'image stego. Par exemple la modification d'un pixel dans une zone uniforme serait extrêmement détectable par un stéganalyste. Intuitivement, il serait judicieux d'attribuer **un coût de modification** du coefficient considéré qui soit différent suivant le voisinage local. Ce coût pourrait être de ce fait élevé pour un pixel se trouvant dans une zone homogène ou sur un contour comme cela est illustré à la figure 1.7.

(a) (b)

Figure 1.7: Illustration de zones de 10×10 pixels : (a) dans la zone illustrée une modification sera extrêmement détectable au vu de l'homogénéité de cette zone, (b) cette zone est plus propice à la modification que la zone (a).

C'est le principe de stéganographie par minimisation du coût d'insertion : le problème initial du stéganographe est d'**insérer en maximisant la sécurité** qui peut être transposé en un problème d'optimisation : **comment minimiser la distorsion induite par l'insertion**.

Après avoir fixé les coûts $\rho_i^{\pm k}$ associés à la modification du coefficient i d'une valeur $\pm k$, la première étape consiste ainsi à définir un carte de détectabilité de

l'image. Cette carte permet d'évaluer le coût de modification de chaque zone afin de déterminer lesquelles sont les moins modifiables au sens de la stéganalyse. Pour expliquer cette étape, nous considérerons ici que les images étudiées seront ici des images en niveaux de gris codées sur 8 bits.

Déterminer la carte des coûts $((\rho_{i,j})_{\substack{0\leq i<N\\0\leq j<M}})$ d'une image de taille $N\times M$, consiste à élaborer une carte des coefficients (pixels ou coefficients DCT) pour lesquels la modification implique le moins de détectabilité.

$$\rho = \{\rho_i \in [0,\infty]\}_{i=0}^{n-1}\,, \tag{1.6}$$

où n est le nombre de coefficients de l'image.

Il est possible de faire une équivalence mathématique entre coûts d'insertion et probabilités de modifications (pour une taille de message donnée). En effet, un coefficient se voyant attribuer un coût d'insertion élevé sur la carte des coûts se verra attribué une probabilité de modification faible et réciproquement.

Dans la majorité des cas, la distorsion D pour une image cover $\mathbf{X}$ et une image stego $\mathbf{Y}$ est définie comme une fonction modélisant l'impact d'insertion, lequel devant être le plus faible possible.

Généralement cette distorsion est une fonction qui mesure la norme entre cover et stego comme :

$$D : (\mathbf{X},\mathbf{Y}) \to \|f(\mathbf{X}) - f(\mathbf{Y})\|, \tag{1.7}$$

où f est une fonction qui retourne un vecteur à valeurs réelles caractérisant l'image par un ensemble de caractéristiques dans un espace de dimension finie.

Afin de simplifier le problème de la minimisation de l'impact de l'insertion, nous faisons l'hypothèse que la modification d'un échantillon n'affecte pas la détectabilité de son voisinage, et à chaque pixel, est associé un coût de détectabilité ρ_i. Il s'agit alors d'une **distorsion additive** car la distorsion totale D peut ainsi s'écrire comme somme de l'ensemble des contributions des éléments de la différence entre cover et stego, et **dans le cas d'une insertion binaire**, en supposant que le coût associé à aucune modification est nul, cette quantité s'écrit comme :

$$D(\mathbf{X},\mathbf{Y}) = \sum_{i=1}^{i=n} \rho_i^{+1} \left|X_i - Y_i\right|. \tag{1.8}$$

Enfin, une fois les coûts calculés il est possible de calculer les probabilités de modifications et réaliser l'insertion.

Dans le cas d'une insertion ternaire où $\pi_i^{-1} = \pi_i^{+1} (\Longleftrightarrow \rho_i^{-1} = \rho_i^{+1})$, en posant $\pi_i^{-1}, \pi_i^{+0}, \pi_i^{+1}$ les probabilités de modifications pour réaliser respectivement $-1, 0, +1$, du coefficient i. La distorsion totale s'écrit alors :

$$D(\mathbf{X},\mathbf{Y}) = \sum_{i=1}^{i=n} \rho_i^{sgn(X_i-Y_i)} \left|X_i - Y_i\right|. \tag{1.9}$$

Pour définir la contrainte dans le problème de minimisation, on définit l'entropie

ternaire H_3 telle que :

$$H_3(\pi_i) = \sum_{k\in\{-1,0,+1\}} -\pi_i^k \log(\pi_i^k). \tag{1.10}$$

La taille du message à insérer peut-être contrainte comme ceci :

$$\mathbb{H}\left[\pi_0, \cdots, \pi_n\right] = \sum_{i=0}^{n-1} H_3(\pi_i) = m_{bits}, \tag{1.11}$$

ce qui permet d'exprimer ainsi la distorsion totale comme :

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{i=1}^{i=n} \rho_i^{sgn(X_i - Y_i)} \left|X_i - Y_i\right| = \sum_{i=0}^{n-1} \pi_i^{\pm 1} \cdot \rho_i^{\pm 1}. \tag{1.12}$$

Or $\pi_i^{-1} = \pi_i^{+1}$, ainsi :

$$D(\mathbf{X}, \mathbf{Y}) = \sum_{i=0}^{n-1} \pi_i^{+1} \cdot \rho_i^{+1}, \tag{1.13}$$

avec $\pi_i^{-1} + \pi_i^{+0} + \pi_i^{+1} = 1$. Ce qui revient à résoudre un problème d'optimisation, à savoir : insérer un message d'une taille donnée tout en minimisant la distorsion induite. Il est possible de résoudre le problème précédent par la méthode des multiplicateurs de Lagrange, fournissant ainsi une expression explicite des probabilités d'insertion des coefficients :

$$\rho_i^{+1} = \lambda_1 \log\left(\frac{1 - 2\pi_i^{+1}}{\pi_i^{+1}}\right) \Longleftrightarrow \pi_i^{+1} = \pi_i^{-1} = \frac{\exp\left(-\rho_i/\lambda_1\right)}{1 + 2\exp\left(-\rho_i/\lambda_1\right)} = \frac{1 - \pi_i^0}{2}, \tag{1.14}$$

avec $\rho_i^{+1} = \rho_i^{-1}$ et λ_1 dont la valeur peut être obtenue par dichotomie. De ce fait, lorsque d'une insertion adaptative pour un message de taille m, les modifications de l'image ne seront pas effectuées uniformément, afin de rendre la stéganalyse plus difficile.

Pour le cas ternaire où $\pi_i^{-1} \neq \pi_i^{+1} (\Longleftrightarrow \rho_i^{-1} \neq \rho_i^{+1})$, en reprenant la même méthode que précédemment, les calculs fournissent ainsi pour le cas d'une insertion ternaire pour laquelle $\pi_i^{-1} \neq \pi_i^{+1}$ respectivement :

$$\pi_i^{+1} = \frac{\exp(-\rho_i^{+1}/\lambda_1)}{1 + \exp(-\rho_i^{+1}/\lambda_1) + \exp(-\rho_i^{-1}/\lambda_1)}, \tag{1.15}$$

et

$$\pi_i^{-1} = \frac{\exp(-\rho_i^{-1}/\lambda_1)}{1 + \exp(-\rho_i^{+1}/\lambda_1) + \exp(-\rho_i^{-1}/\lambda_1)}. \tag{1.16}$$

4.3 Exemples de fonctions coûts dans le domaine spatial

Le premier algorithme dont le mécanisme d'insertion prend en considération les interactions des changements réalisés est **HUGO** (Highly Undetectable steGO [94]), développé lors la compétition de stéganalyse BOSS.

Il s'agit aussi du premier algorithme dit **adaptatif**. Cet algorithme part du constat qu'en stéganalyse les caractéristiques extraites ont pour objectif de réduire la dimension de l'espace décrivant les images (telle que la dimension soit inférieure au nombre de pixels multiplié par le nombre de canaux), permettant ainsi à rendre numériquement possible l'apprentissage à un classifieur de la différence entre les covers et les stegos.

De ce fait l'intuition ici est d'utiliser un ensemble de caractéristiques afin de déterminer les coûts d'insertion et comme caractéristiques de faible dimension du modèle de l'espace des stegos. Les SPAM [74] sont des caractéristiques qui modélisent les probabilités de transitions entre pixels voisins et ceci suivant 8 directions $\{\leftarrow, \rightarrow, \downarrow, \uparrow, \nwarrow, \searrow, \swarrow, \nearrow\}$ (le calcul des co-occurences sur des résidus peut être ici vu comme du filtrage des basse-fréquence, exposant ainsi les résidus et atténuant ainsi le contenu). Les caractéristiques SPAM ont ainsi été utilisées comme point de départ pour la conception de leur schéma d'insertion.

On définit ainsi une distance entre l'espace de description entre covers et stegos et pixels modifié ce qui permet d'obtenir un coût. Ainsi de façon itérative, l'insertion est réalisée de sorte à minimiser cette distance et donc ce coût. Enfin l'insertion est réalisée de sorte à ce que le modèle estimé des images stegos corresponde à celui de leur version cover respective comme expliqué sur la figure 1.8.

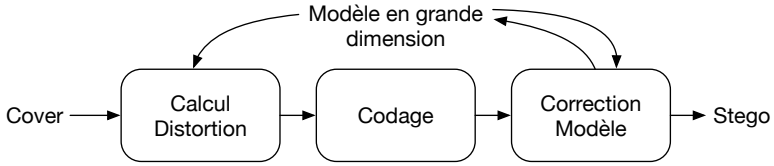


Figure 1.8: Processus d'insertion effectué par HUGO.

La finalité de cette approche est que la carte de coût ainsi générée permet d'insérer les échantillons sur lesquels l'adversaire (le stéganalyste) a de faibles chances de déceler les modifications.

D'autres algorithmes ont succédé à HUGO, notamment ceux basées sur la fonction de distorsion universelle **UNIWARD** [53] (Universal Distortion Function for Steganography in an Arbitrary Domain) qui est un algorithme d'insertion ternaire.

Cette fonction de coûts exploite une banque de filtres passe-hauts directionnels (ondelettes de Daubechies). Ces filtres permettent d'extraire les bords horizontaux, verticaux et diagonaux des résidus des images. Le but étant de mesurer l'impact des modifications dans chaque direction, attribuant de ce fait un faible coût d'insertion à un pixel si le contenu local est imprédictible dans toutes les directions. L'algorithme

évite ainsi les zones homogènes, les bords, et concentre ainsi les coûts faibles dans les zones texturées.

$$D(\mathbf{X},\mathbf{Y}) \triangleq \sum_{k=1}^{3}\sum_{u=1}^{n_1}\sum_{v=1}^{n_2}\frac{\left|W_{uv}^{(k)}(\mathbf{X}) - W_{uv}^{(k)}(\mathbf{Y})\right|}{\sigma + \left|W_{uv}^{(k)}(\mathbf{X})\right|}, \tag{1.17}$$

où $\mathbf{X}$ et $\mathbf{Y}$ représentent respectivement l'image cover et l'image stego dans le domaine spatial. $W_{uv}^{(k)}(\mathbf{X})$ et $W_{uv}^{(k)}(\mathbf{Y}), k = 1, 2, 3, u \in \{1, \ldots, n_1\}, v \in \{1, \ldots, n_2\}$, leurs coefficients u, v par transformée en ondelettes de Daubechies dans la k-ième bande et $\sigma > 0$ une constante de stabilisation pour les calculs.

Un autre algorithme du domaine spatial est **MiPOD** (Minimizing the Power of Optimal Detector) qui tire son efficacité de l'estimation de la variance des pixels, qui, sont supposés être statistiquement indépendants et modélisés comme les réalisations d'une distribution Gaussienne : $X_{m,n} \sim \mathcal{N}\left(\mu_{m,n}, \sigma_{m,n}^2\right)$. La figure 1.9 propose un schéma explicatif succinct pour illustrer son fonctionnement.

Sur la base de ce modèle statistique de pixels, la méthode d'insertion consiste essentiellement à trouver la probabilité d'insertion sur chaque pixel $\pi^{i,j}$ tout en minimisant la puissance du test de rapport de vraisemblance le plus puissant (LRT). Ce qui revient ici à minimiser le coefficient déflexion (ou détectabilité) du détecteur d'un stéganalyste omniscient qui peut être ici exprimée comme : $\mathbf{U}^{(c)}, c \in \{\mathrm{Y}, \mathrm{Cb}, \mathrm{Cr}\}$.

$$\varrho^2 = \sum_{i,j}\frac{(\pi^{i,j})^2}{\sigma_{i,j}^4}. \tag{1.18}$$

La minimisation de la déflexion sous-contrainte de capacité mène à la résolution d'un problème d'optimisation dont une solution peut-être obtenue par la méthode des multiplicateurs de Lagrange.

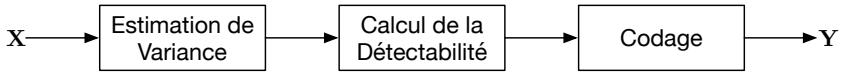


Figure 1.9: Schéma d'insertion simplifié pour l'algorithme MiPOD.

4.4 Exemples de fonctions coûts dans le domaine JPEG

Dans le domaine spatial UNIWARD est directement appliqué sur l'image où la modification d'un pixel affect son voisinage 16×16. Appliqué dans le domaine JPEG, UNIWARD traite des images JPEG après décompression dans le domaine spatial, puis les coefficients d'ondelettes sont calculés, sa mise en œuvre répond à la dénomination **J-UNIWARD**. Dans le domaine JPEG le changement d'un coefficient affecte ici son voisinage 23×23

La distorsion entre deux images ($\mathbf{X}$ et $\mathbf{Y}$) JPEG est calculé après décompression des coefficients JPEG dans le domaine spatial, par conséquent la formule de calcul de

la distorsion est identique à 1.17 la différence de l'étape préalable de décompression.

On peut également citer UERD (Uniform Embedding RevisiteD) [47] dont l'idée sous-jacente repose sur une insertion uniforme, répartissant uniformément les modifications sur les coefficients quantifiés de la transformée en cosinus discrets sur toutes les amplitudes possibles. Les changements moyens des statistiques du premier et du second ordre peuvent être ainsi minimisés, ce qui conduit à une détectabilité statistique moindre.

4.5 Stéganographie par insertions adverses

Afin d'adresser au mieux le problème de la détectabilité, une alternative aux approches classiques est d'utiliser la stéganalyse pour renforcer un schéma d'insertion. Cette ligne de recherche a donné lieu à des approches dites adverses qui tirent parti des connaissances provenant de la stéganalyse, et insérer un message sous contrainte de minimisation de détectabilité face à un adversaire.

4.5.1 ASO : Adaptative Steganography by Oracle

Parmi ces approches, ASO [64] prend en compte la distribution statistique d'une base d'images complète afin de mettre à jour les coûts d'insertion.

Cette stratégie calcule ses coûts d'insertions à partir d'un ensemble classifieur. De ce fait ce schéma d'insertion, pionnier en son genre, est construit comme une offensive sur un détecteur (qu'un adversaire pourrait utiliser), réalisant ainsi les modifications sur une image à partir des sorties d'un ensemble classifieur comme illustré sur la figure 1.10. Les coûts d'insertion sont mis à jour en tenant compte des connaissances des classifieurs dont l'adversaire pourrait disposer.

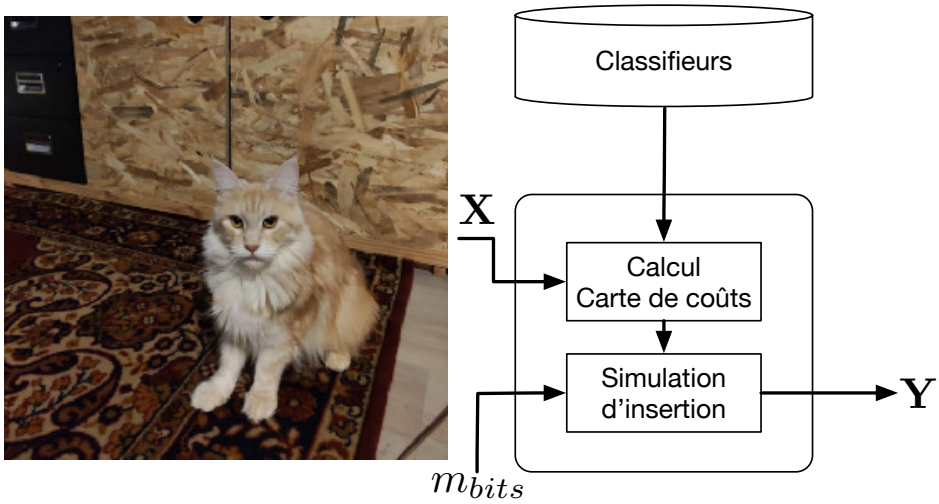


Figure 1.10: Schéma d'insertion ASO.

En utilisant les notations du papier ASO [64], l'ensemble des classifieurs de type Discriminants Linéaires de Fisher (FLD pour « Fisher Linear Discriminents », voir Section 5.4) sont notés $\mathcal{F}$ tandis que l'ensemble des FLDs extraits d'un ensemble des classifieurs pré-entraînés est noté $\hat{\mathcal{F}}$.

Les coûts d'insertion sont ainsi définis comme :

$$\begin{aligned} \rho_i^{(+)} &= \sum_{l=1}^{\mathrm{Card}(\hat{\mathcal{F}})} \rho_i^{(l)(+)}, \\ \rho_i^{(-)} &= \sum_{l=1}^{\mathrm{Card}(\hat{\mathcal{F}})} \rho_i^{(l)(-)}, \end{aligned} \tag{1.19}$$

où $\rho_i^{(l)(+)}$ et $\rho_i^{(l)(-)}$ représentent respectivement les coûts d'incrémenter et décrémenter de $+1$ et -1 le pixels i pour le classifieur $F_l \in \hat{\mathcal{F}}$.

L'utilisation d'un oracle permet ici de modéliser le fait qu'on suppose connue la stratégie de stéganalyse employée par le stéganalyste. Cette logique permet de connaître l'impact pratique en termes de détectabilité de la modification d'un échantillon de l'image. La connaissance de cet impact permet ainsi de minimiser de façon pratique la détectabilité des images en particulier face à une stéganalyste dont les capacités sont connues. De plus, cette approche offre l'opportunité de choisir potentiellement les images les plus sécurisées lors de sa transmission.

En s'appuyant sur des considérations similaires, les auteurs du papier qui suit (voir 4.5.2) ont dégagé un protocole d'insertion dont l'objet principal repose sur la sélection de la « meilleure » image stego pour faire face à des classifieurs entraînés itérativement.

4.5.2 Insertions Adverses

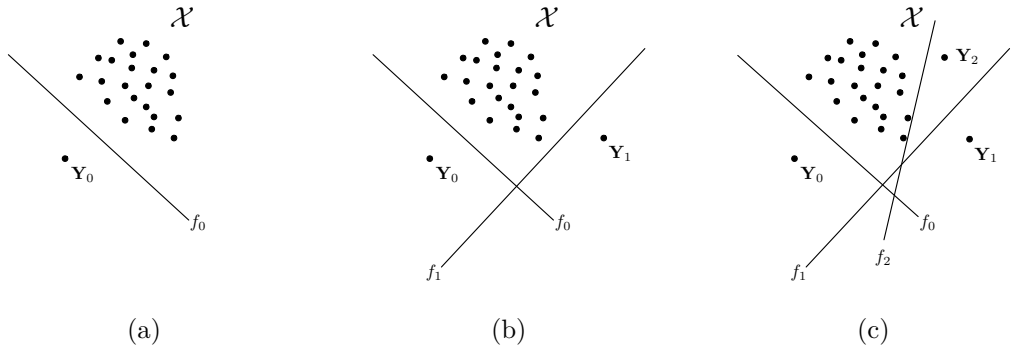
S. Bernard et al. ont ainsi proposé de construire itérativement une fonction de distorsion pour accroître la sécurité empirique d'un schéma d'insertion à chaque itération du protocole [11].

Cette méthode repose sur l'attaque d'un classifieur ou plusieurs classifieurs avec l'emploi d'une stratégie *min-max* permettant de sélectionner à chaque itération l'image stego la plus difficile pour le classifieur le plus efficace.

On peut décomposer le processus de la sorte :

1. Le processus est initialisé par la création préalable d'un ensemble d'images stego $\mathcal{Z}_0 = \mathcal{Y}_0$ en utilisant un algorithme d'insertion arbitraire (les auteurs ont utilisé J-UNIWARD). Puis un classifieur f_0 est entraîné afin de discriminer $\mathcal{X}$ et $\mathcal{Y}_0$. Ce premier classifieur est ainsi ajouté à l'ensemble des classifieurs disponibles $\mathcal{F} = \{f_0\}$.
2. À l'itération suivante, le stéganographe génère un nouveau set d'images stegos $\mathcal{Z}_1$ pour attaquer f_0 et génère $\mathcal{Y}_1$ en utilisant $\{\mathcal{Z}_1, \mathcal{Z}_0\}$ de sorte à ce que les images générées parviennent à tromper au mieux le classifieur f_0.
3. L'étape précédente est répétée k fois.

Les étapes de ces itérations sont illustrées à la figure 1.11.



(a) (b) (c)

Figure 1.11: Illustration de la séquence d'obtention d'une image stego $\mathbf{Y}_2$ à l'itération $k = 2$.

4.5.3 Stéganographie par réseaux génératifs adverses

Les réseaux génératifs adverses, aussi appelés GANs (« Generative Adversarial Networks ») constituent une classe particulière de réseaux neuronaux aux applications remarquables. Ils consistent en un système dual, composés de deux réseaux de neurones, le Générateur et le Discriminateur, qui se font face. Notons que les GANs

ont une pléthore d'applications, car ils peuvent apprendre à imiter des distributions de données de presque n'importe quel type.

Dans le cadre de la stéganographie, à partir d'un ensemble d'images covers de référence, le générateur tente de produire des images stegos qui peuvent tromper le Discriminateur en lui faisant croire qu'ils sont des images covers. Le discriminateur est donc entraîné pour essayer de distinguer les images stegos des images covers. En utilisant cette approche itérative, le générateur entraîné permet de générer des échantillons similaires aux échantillons cibles.

Dans [91, 90, 97] sont proposés des stratégies basées sur l'exploitation d'un GAN pour créer à partir d'une image cover une carte des probabilités de modifications.

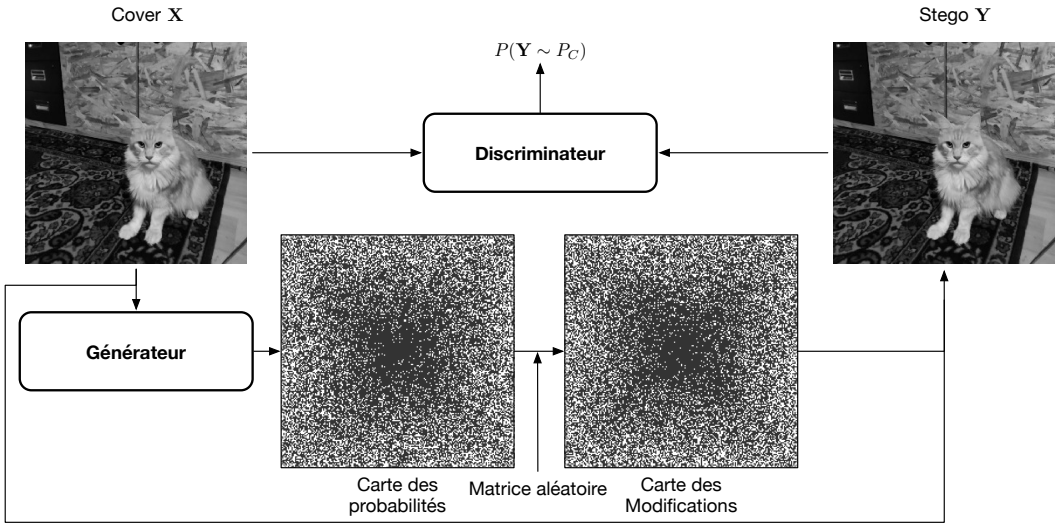


Figure 1.12: Schéma d'un réseau de type GAN apprenant à générer des cartes de probabilités face à un discriminant.

Dans la configuration de leur article, les auteurs de [97] obtiennent après convergence de leur réseau, une amélioration de la sécurité empirique face à un stéganalyste utilisant les SRM [43](Spatial Rich Models) comme ensemble de caractéristiques pour la stéganalyse (voir Section 5) allant jusqu'à 1.29%, 2.74% et 7.14% par rapport à HILL [67], MiPOD [79] et S-UNIWARD [53] respectivement. Cette augmentation est d'autant plus élevée face à un stéganalyste équipée de la même architecture de réseau de neurones que celle exploitée par le discriminateur : affichant ainsi un gain de 7.8%, 6.23% et 9.22% pour HILL, MiPOD et S-UNIWARD respectivement.

4.6 Stéganographie par synchronisation des modifications

Afin de prendre en compte l'impact mutuel des modifications sur des coefficients voisins, certains auteurs ont proposé des fonctions de distorsion non-additives permettant de capturer l'impact des modifications déjà effectuées sur l'image sur les futures modifications à effectuer. C'est le cas des approches par regroupement des directions de modification (CMD) [68] et par continuité des frontières des blocs (BBC) [69] qui exploitent toutes deux des treillis disjoints afin d'utiliser les modifications jointes entre coefficients (pixels pour CMD et DCT pour BBC).

4.6.1 Synchronisation par regroupement des directions de modification (CMD)

CMD (Clustering Modification Directions) est une approche de synchronisation dans le domaine spatial datant de 2015 pour laquelle l'insertion est effectuée de façon séquentielle sur des sous-treillis. Les changements réalisés pour un voisinage donné et sur les sous-treillis précédents sont utilisés pour réduire les coûts relatifs à la majorité des changements déjà effectués.

Plus précisément, en première étape de CMD, une image cover est décomposée en 4 sous-treillis entrelacés. Cette étape est représentée sur la figure 1.13 pour un groupe de 6×6 pixels.

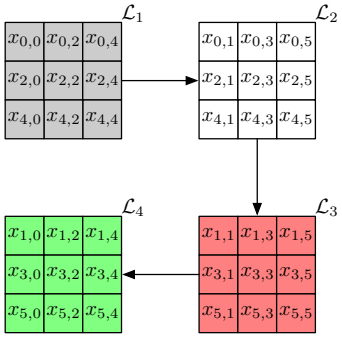
$x_{0,0}$	$x_{0,1}$	$x_{0,2}$	$x_{0,3}$	$x_{0,4}$	$x_{0,5}$
$x_{1,0}$	$x_{1,1}$	$x_{1,2}$	$x_{1,3}$	$x_{1,4}$	$x_{1,5}$
$x_{2,0}$	$x_{2,1}$	$x_{2,2}$	$x_{2,3}$	$x_{2,4}$	$x_{2,5}$
$x_{3,0}$	$x_{3,1}$	$x_{3,2}$	$x_{3,3}$	$x_{3,4}$	$x_{3,5}$
$x_{4,0}$	$x_{4,1}$	$x_{4,2}$	$x_{4,3}$	$x_{4,4}$	$x_{4,5}$
$x_{5,0}$	$x_{5,1}$	$x_{5,2}$	$x_{5,3}$	$x_{5,4}$	$x_{5,5}$

Figure 1.13: Décomposition en sous-treillis d'un tableau de 6×6.

La distorsion au sein de chaque sous-treillis est définie sous une forme additive avec les coûts $\rho_{i,j}$ afin que les STCs (définis à la sous-section 4.9) puissent y être utilisés. Comme expliqué à la figure 1.14, les treillis $\mathcal{L}_2$, $\mathcal{L}_3$ et $\mathcal{L}_4$ sont définis pour un tableau 6×6 comme les ensembles :

$$\begin{aligned}
\mathcal{L}_1 &= \{(i,j) |\ \text{mod}\ (i,2) = 1 \text{ and } \ \text{mod}\ (j,2) = 1\}, \\
\mathcal{L}_2 &= \{(i,j) |\ \text{mod}\ (i,2) = 1 \text{ and } \ \text{mod}\ (j,2) = 0\}, \\
\mathcal{L}_3 &= \{(i,j) |\ \text{mod}\ (i,2) = 0 \text{ and } \ \text{mod}\ (j,2) = 0\}, \\
\mathcal{L}_4 &= \{(i,j) |\ \text{mod}\ (i,2) = 0 \text{ and } \ \text{mod}\ (j,2) = 1\}.
\end{aligned} \tag{1.20}$$

Ensuite 1/4 du message est inséré dans chaque sous-treillis en commençant par $\mathcal{L}_1$ comme schématisé à la figure 1.14.



(a)

Figure 1.14: Illustration de l'ordre d'insertion pour chaque sous-treillis.

Pour les treillis suivants $\mathcal{L}_2$, $\mathcal{L}_3$ et $\mathcal{L}_4$ l'algorithme va encourager la modification du pixel (i, j) à correspondre à la majorité des changements déjà effectués dans son voisinage du premier ordre.

En d'autres termes : pour un pixel donné, si les changements déjà effectués sur son voisinage sont majoritairement positifs alors le coût pour réaliser $+1$ est divisé par 9 (facteur d'échelle calculé de façon empirique). Réciproquement, pour une majorité de changements négatifs, le coût pour réaliser -1 est divisé par 9. Enfin, une moyenne nulle ne donne lieu à aucune modifications du coût initialement attribué.

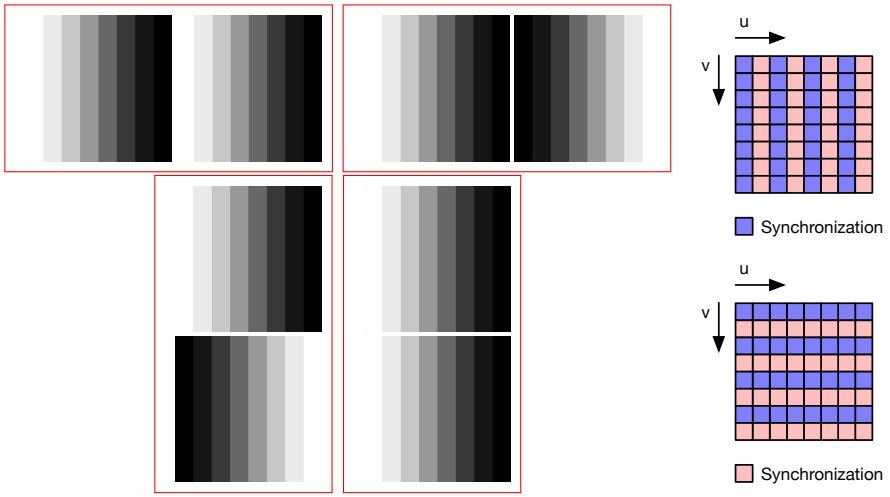
On peut souligner ici que les auteurs ont constaté avec leur protocole expérimental que le taux de modifications (le ratio entre le nombre de pixels modifiés et le nombre total de pixels) était plus important en plus d'une sécurité empirique plus élevée.

4.6.2 Synchronisation par préservation de continuité des frontières des blocs

Dans le domaine JPEG des travaux similaires ont été publiés permettant d'augmenter sensiblement la sécurité empirique de schémas d'insertion de l'état de l'art [68, 32, 100]. Les auteurs de [69] ont développé une stratégie d'insertion appelée « Block Boundary Continuity » (BBC) afin de définir la distorsion conjointe dans le domaine JPEG, avec pour objectif de réduire les artefacts inter-blocs DCT causés par les modifications effectuées sur les blocs adjacents. Avec ce schéma, la synchronisation ou la désynchronisation des modifications adjacentes entre blocs est liée au mode DCT et à la direction adjacente des coefficients inter-blocs (horizontale ou verticale) comme illustré à la figure 1.15. Les auteurs ont ainsi articulé leur papier autour

de la création d'une méthodologie permettant de préserver au mieux la continuité spatiale entre les blocs DCT afin de limiter l'apparition d'artefacts trop visibles.

Par exemple pour le mode DCT $(0, 1)$ la figure 1.15 permet de mettre en lumière les cas de continuités/discontinuités inter-bloc que ce soit dans le cas où le couple de blocs est connecté selon la verticale ou l'horizontale.



(a) (b)

Figure 1.15: (a) Illustration des continuités/discontinuités pour des blocs voisins avec un même modes pour des polarités identiques et différentes, (b) Schéma de synchronisation des modes.

Ainsi pour un mode et une direction donnée, afin d'éviter l'apparition de discontinuités entre les blocs (inter-blocs), les auteurs du papier ont ainsi développé un algorithme permettant d'encourager les modifications préservant la continuité entre les blocs adjacents dans le cas horizontal et vertical.

Ces résultats sont obtenus en posant $\Phi_{u,v}^{hor}\left(x, \Delta F_1, \Delta F_2\right)$ (pour deux blocs connectés horizontalement) et $\Phi_{u,v}^{ver}\left(x, \Delta F_1, \Delta F_2\right)$ (pour deux blocs connectés verticalement) les fonctions qui, pour un mode donné de coordonnées (u, v) dans un bloc DCT et un couple de modification $(\Delta F_1, \Delta F_2) \in [\![-1, +1]\!]^2$ associe ± 1. La valeur 1 correspondant à la nécessité d'encourager la synchronisation et -1 une synchronisation en opposition de phase entre les blocs adjacents pour le mode en question.

Enfin, le papier suggère une définition d'une fonction de distorsion conjointe permettant l'ajustement des coûts afin de réduire les artefacts que les auteurs s'attachent à éviter. Pour (u, v) un couple de coordonées (désignant un bloc DCT) et pour i un mode DCT, on définit un couple de coefficients DCT noté $B_{u,v}^{(i)} = \left(d_{u,v}^{i,1}, d_{u,v}^{i,2}\right)$ où 1 et 2 désignent respectivement deux blocs connectés soit

horizontalement, soit verticalement. On note aussi $c^{i,1}_{u,v}(\Delta F_1)$ les couts initiaux attribués aux coefficients DCT $\left(d^{i,1}_{u,v}, d^{i,2}_{u,v}\right)$.

La distorsion conjointe est définie comme :

$$\rho^{(i)}_{u,v}(\Delta F_1, \Delta F_2) = \omega_{u,v}(\Delta F_1, \Delta F_2) \times \left(c^{i,1}_{u,v}(\Delta F_1) + c^{i,2}_{u,v}(\Delta F_2)\right), \tag{1.21}$$

avec $\omega_{u,v}$ un facteur de mise à l'échelle :

$$\omega_{u,v}(\Delta F_1, \Delta F_2) = \begin{cases} 1/\alpha & \text{si } \Phi^{\text{hor}}_{u,v}(\Delta F_1, \Delta F_2) = 1, \\ \alpha & \text{si } \Phi^{hor}_{u,v}(\Delta F_1, \Delta F_2) = -1, \\ 1 & \text{sinon.} \end{cases} \tag{1.22}$$

Et $\alpha > 1$ un réel dont la valeur optimale a été estimée de façon empirique. Finalement cette approche permet aux auteurs d'augmenter la sécurité empirique de schémas d'insertion tel que UERD et J-UNIWARD.

4.7 Utilisation de l'information adjacente

L'information adjacente fait référence à toute connaissance que le stéganographe peut utiliser sur son image cover afin de parfaire la sécurité empirique d'un schéma d'insertion (e.g : l'image RAW, plusieurs images de la même scène, les erreurs d'arrondis...).

Cette information permet de compenser l'éventuelle manque de connaissance de notre modèle (comme détaillé dans [72]), en utilisant ce que nous appellerons par la suite une image pré-cover. Il s'agit dans la majorité des cas d'une version de l'image cover en haute qualité : typiquement une image RAW avant qu'elle ne subisse une quelconque opération de type post-traitement, développement ou une compression liée à son format de sauvegarde (quantifiée sur 16 ou 8 bits ou compressée en JPEG par exemple).

Un des schémas découlant de la fonction de coût universelle UNIWARD est **SI-UNIWARD** [33], il s'agit d'un schéma d'insertion (ternaire) dans le domaine JPEG basé sur J-UNIWARD [53] (voir sous-section 4.3). Ce schéma améliore la sécurité empirique de celui-ci en modulant les coûts obtenus par $|0.5 - e_i|$ (avec $e_i \in]-0.5, 0.5]$ l'erreur d'arrondis du i^{eme} coefficient DCT lors de la compression de la pré-cover), ce schéma utilise une méthode appelée « Perturbed Quantization » [42].

À partir d'une image cover RAW convertie dans le domaine DCT et non-quantifiée il est possible d'obtenir les coefficients $d_{i,j}$, le stéganographe a alors le choix d'incrémenter ou de décrémenter sa valeur afin de moduler sa parité. Cette approche a tendance à favoriser les changements sur les éléments dont les erreurs d'arrondis sont proches de $\pm 1/2$ car ce sont ces coefficients qui seront les plus aptes à naturellement changer de valeur s'ils sont soumis à une petite perturbation.

Par exemple (voir figure 1.16), un coefficient dont la valeur serait égale à 3.45 qui serait normalement arrondi à 3 serait autorisé à être modifié de sorte à ce qu'il

soit égal à 4 avec un faible coût d'insertion, alors qu'un coefficient dont la valeur flottante serait proche de 4 se verrait attribué un coût de modification très élevé pour un passage à 5.

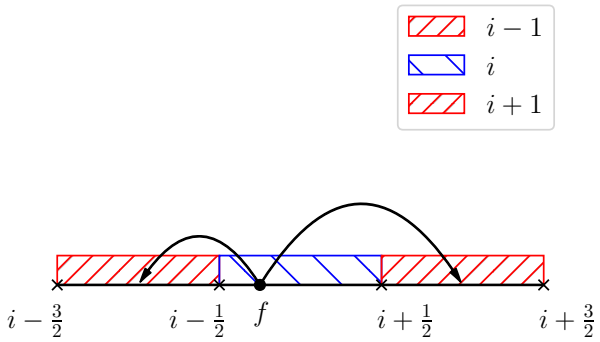


Figure 1.16: Exemple pour lequel une modification réalisant un arrondi vers la gauche doit être plus naturelle (donc associé à un coût de modification plus faible) qu'un arrondi vers la droite.

4.8 Stéganographie couleur

Puisque la stéganographie peut être utilisée pour cacher des messages potentiellement sensibles à l'intérieur de formats d'images conventionnels, il est surprenant de constater que la majorité des contributions académiques en stéganographie et en stéganalyse portent sur des formats d'images exotiques tels que le format RAW (PGM, PPM) ou les images JPEG en niveaux de gris.

Cet état de fait a été notamment constaté dans [55] et par les auteurs du challenge ALASKA [24], ont estimé qu'entre 2016 et 2019, seulement 16% des contributions académiques en stéganographie et stéganalyse étudiaient les images couleurs.

Deux papiers proposent cependant des approches exclusivement dédiées aux images couleurs. Le premier, proposé par W. Tang et al. [89] est une extension de l'approche CMD évoquée plus haut; le second, soumis par R. Cogranne [23] propose une extension de l'approche MiPOD afin de définir une fonction de coûts sur des fondamentaux statistiques.

On pourra également noter que les premiers pas de la communauté vers des images couleurs furent effectués en 2014 dans [44] et [58].

En préambule de cette sous-section nous introduirons deux stratégies de répartitions du message. Une stratégie simple pour effectuer l'insertion consiste à fixer le même taux de modification pour les trois canaux couleur. Cette stratégie omet que dans le cas d'une image JPEG, les composantes de chrominance contiennent en moyenne moins d'information que la composante de luminance (voir [87]) et que celle-ci est quantifiée de façon différente.

Une autre stratégie dans le domaine JPEG est de répartir différemment la taille du message entre les canaux couleurs [87]. Cette stratégie s'appuie notamment sur le fait que les canaux de chrominance sont susceptibles de contenir moins d'informations et d'ainsi ne pas pouvoir intégrer la même taille de message que le canal de luminance avec la même distorsion.

Enfin, la stratégie par répartition fixée sur les canaux couleurs (appelée CCFR dans [23]) consiste à fixer une fraction γ du message R:

$$R_Y = \delta(1-\gamma)R, \tag{1.23}$$

$$R_{Cb} = R_{Cr} = \delta\gamma R. \tag{1.24}$$

Avec δ un facteur d'échelle permettant d'atteindre la contrainte sur la taille du message à insérer.

Alternativement à la minimisation de la détectabilité basée sur des coûts d'insertion, des approches alternatives basées sur la minimisation de la détectabilité statistique ont émergé en se basant sur le modèle statistique des pixels.

La contribution des auteurs [23] vise à exploiter cette approche basée sur un modèle statistique des images covers afin de concevoir des fonctions de distorsion pertinentes pour les images JPEG en couleur.

Contrairement à des travaux antérieurs qui nécessitent une connaissance précise du processus de traitement des images, cette contribution vise à fournir des solutions pratiques dans le cas le plus général où le stéganographe ne reçoit qu'une image numérique déjà compressée.

L'apport de cette contribution est double, elle apporte une extension à MiPOD afin de définir une fonction de distorsion pour les images JPEG couleur basées sur des fondements statistiques et propose une méthode pertinente pour la stéganographie dans le domaine JPEG.

En effet, la question de la répartition du message à travers les différents canaux est épineuse, car celle-ci peut avoir un impact significatif sur la sécurité empirique comme cela a été étudié dans [88]. Plusieurs stratégies ont été envisagées par les auteurs : (1) répartir différemment le message entre le canal de luminance et les canaux chromatiques, (2) concaténer les canaux pour les traiter comme une image en niveaux de gris, (3) égaliser la déflexion ϱ induite sur chaque canal et (4) égaliser la distorsion induite à partir de coûts.

Il apparaît que l'approche (3) fournit les résultats les plus intéressants, cependant la causalité n'est pas encore claire.

4.9 Codage pour la stéganographie

Le codage effectif du message lors d'une insertion peut-être effectuée de plusieurs manières. L'état de l'art actuel de cette opération est réalisé par les STCs (Syndrom-Trellis Codes) [37], qui permettent tout en conservant l'optimalité de l'insertion de s'adapter à n'importe quelle mesure de distorsion additive et ainsi à n'importe quel schéma construit suivant ce raisonnement. L'autre avantage des STCs par rapports aux méthodes antérieures est qu'ils permettent de minimiser le coût avec une complexité qui reste linéaire.

Le but reste toujours d'insérer un message d'une taille donnée tout en minimisant l'impact (distorsion) de cette opération. Comme précisé précédemment, la tache d'insertion sous contrainte de minimisation de la distorsion peut se présenter sous deux formes : (1) insérer un message m fixe de m_{bits} bits tout en minimisant la distorsion moyenne, ou (2) maximiser la taille moyenne du message inséré tout en introduisant une distorsion moyenne fixe. En pratique ces deux formes (1) & (2) peuvent être satisfaites en utilisant le codage par syndromes.

En faisant ici l'hypothèse d'une insertion binaire, et en posant $\mathcal{P}(x_{i,j} \in \mathbf{X}) = x_{i,j} \bmod 2$ une fonction de parité disponible pour Alice et Bob, l'opération d'insertion et d'extraction doit ainsi vérifier :

$$\mathrm{Ext}(\mathrm{Emb}(\mathbf{X}, m_{bits})) = m, \quad \forall \mathbf{X} \in \mathcal{X}, \forall m \in \{0,1\}^m, \tag{1.25}$$

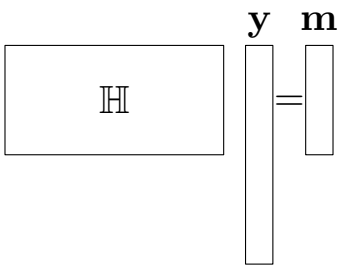
Avec $\mathcal{X}$ l'ensemble des images cover.

Lors du codage par syndromes, l'insertion et l'extraction sont réalisées par un code linéaire $\mathcal{C}$ de longueur n et de dimension $n - m$:

$$\begin{aligned} &\mathrm{Emb}(\mathbf{x}, m_{bits}) = \arg \min_{\mathcal{P}(\mathbf{y}) \in \mathcal{C}(m)} D(\mathbf{x}, \mathbf{y}), \\ &\mathrm{Ext}(\mathbf{y}) = \mathbb{H}\mathcal{P}^t(\mathbf{y}), \end{aligned} \tag{1.26}$$

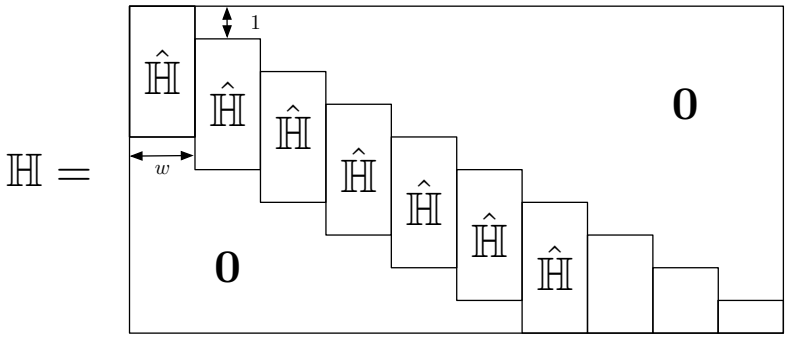
où $\mathcal{P}(\mathbf{y}) = \left(\mathcal{P}\left(y_1\right), \ldots, \mathcal{P}\left(y_n\right)\right), \mathbb{H} \in \{0,1\}^{m \times n}$.

Si de manière générale, l'insertion est une étape NP difficile à cause de la construction de la matrice de vérification de parité $\mathbb{H}$, la contribution principale de ce papier a été ici d'introduire de la structure dans la matrice $\mathbb{H}$ afin de réduire la complexité de son calcul. Pour ce faire, les auteurs définissent $\hat{\mathbb{H}} \in \{0,1\}^{h \times w}$ de façon pseudo-aléatoire, avec $h \in 1, \ldots, 15$, $w = 1/\alpha$ et α la taille de message relative en bits/coefficients. Puis, ils construisent $\mathbb{H}$ comme un pavage par $\hat{\mathbb{H}}$ comme illustré à la figure 1.18.



(a)

Figure 1.17: Illustration de la construction de $\mathbb{H}$ dans le cas d'une insertion pratique.



(a)

Figure 1.18: Construction de $\mathbb{H} \in \{0,1\}^{m \times n}$ à partir des $\hat{\mathbb{H}}$.

La distorsion est en pratique minimisée lors du codage par syndromes en utilisant l'algorithme de Viterbi, qui permet de résoudre l'équation 1.26 de façon optimale tout en trouvant le « chemin » idéal de modifications.

5 Principes de base en stéganalyse

Dans la section précédente, nous avons étayé les grandes lignes de la stéganographie moderne, en présentant ainsi brièvement différentes méthodes de dissimulation d'informations dans des images numériques. Les méthodes de stéganographie que nous avons présentées font l'hypothèse que l'adversaire « Eve » est une gardienne passive. Son rôle dans ce scénario est de distinguer dans une liste d'images les images covers des images stegos.

La stéganalyse s'est naturellement développée en parallèle de la stéganographie afin de fournir des garanties en termes de sécurité.

5.1 Différents types de stéganalyse

Eve, est comme précisé plus haut, une gardienne passive mais par adjonction du principe de Kerckhoffs, elle est en mesure de disposer d'informations supplémentaires lors de son accès au canal de communication utilisé par Alice et Bob.

Ainsi Eve va pouvoir avoir accès au système d'insertion, c'est-à-dire l'algorithme, la façon dont Alice produit des images cover et stego, ainsi qu'éventuellement à la taille des messages insérés.

Parmi les nombreux travaux proposés, trois grandes familles qui correspondent pour chacune à des scénarios bien particuliers sont principalement évoquées, nous allons brièvement les décrire dans cette section.

1. Le scénario le plus communément exploité par l'état de l'art et à travers cette thèse dérive de la **stéganalyse clairvoyante** qui suppose qu'Eve connaît le type de média employé, l'algorithme d'insertion, la taille du message, les propriétés statistiques des covers, etc.
2. D'autres approches étudient la taille éventuelle du message, c'est le cas en **stéganalyse quantitative** où le but est d'estimer la taille d'un possible message dissimulé.
3. Enfin, il est également possible de modéliser la stéganographie comme un problème d'allocation d'un message dans plusieurs images, et ainsi supposer que le stéganographe a dispersé ce message sur un lot d'images. Cette tâche beaucoup plus délicate est étudiée par la **stéganalyse par lots** où toutes les images du lot peuvent être des images covers, ou bien seulement certaines, ou encore peuvent contenir une proportion variable du message.

Les sous-sections suivantes aborderont ainsi différentes voies pour Eve permettant de réaliser sa tâche de stéganalyse dans un cadre classique, la stéganalyse clairvoyante. Ces pré-requis peuvent sembler éloignés de la réalité et pour tenter de dégager de nouvelles méthodes, métriques et conclusions, récemment R. Cogranne et al. ont organisé un concours international appelé ALASKA [24]. Construit pour refléter les difficultés de mise en pratique des méthodes de stéganalyses. L'accent a été mis sur la constitution d'une base d'images hétérogènes de part leurs scènes, sensibilités d'acquisition (ISO), développements, facteurs de qualité, sous-échantillonnages chromatique et taille de messages.

5.2 Métriques utilisées en stéganalyse

La stéganalyse d'images digitales ne peut pas exploiter une représentation totale de l'image à cause de sa complexité et de sa dimension. De ce fait, des modèles simplifiés sont utilisés afin de rendre le problème plus simple à mettre en œuvre.

Dans la majorité des cas, les modèles utilisés décrivent les images par un ensemble de caractéristiques ciblés par le stéganalyste ou bien automatiquement

obtenus par des réseaux de neurones entraînés. Chaque image $\mathbf{X} \in \mathcal{C}$ est ainsi traduite en un vecteur de caractéristiques de dimension d: $\mathbf{f} \in \mathbb{R}^d$. Par conséquent, les variables aléatoires représentant respectivement les sources des covers $\mathbf{X} \sim P_C$ et des stegos $\mathbf{Y} \sim P_S$ sont ainsi traduites pas les variables aléatoires correspondantes : $\mathbf{f_X} \sim P_{f_C}$ et $\mathbf{f_Y} \sim P_{f_S}$.

Par ailleurs, un test est une fonction $\tau : \mathbf{v}_{obs} \mapsto \{0, 1\}$ qui à partir d'un vecteur d'observation $\mathbf{v}_{obs}$ retourne une valeur booléenne $\{0, 1\}$ tel que $\mathcal{H}_0$ une hypothèse sur $\mathbf{v}_{obs}$ est validée si $\tau(\mathbf{v}_{obs}) = 0$. Un test peut produire des erreurs qui peuvent être capturés comme étant des FP (Faux-Positifs ou False Alarm) ou FN (Faux-Négatifs ou Missed Detection) comme illustré au Tableau 1.1. Par un exemple, la classification d'une image cover comme image stego est une fausse alarme.

Réalité / Résultat	$\mathcal{H}_0$ cover	$\mathcal{H}_1$ stego
Accepter $\mathcal{H}_0$	TP	MD
Accepter $\mathcal{H}_1$	FA	TN

Table 1.1: Illustration des possibles en termes de détection lors de la mise en place d'un classifieur.

Ainsi un détecteur peut faire deux types d'erreurs : des Fausses Alarmes (FA) ou Détections Manquées (MD). On définit alors la probabilité de fausses alarme P_{FA} comme la probabilité qu'une variable aléatoire distribuée selon P_{f_C} soit détectée comme une image stego, tandis que P_{MD} est la probabilité qu'une variable aléatoire distribuée selon P_{f_S} soit détectée comme une image cover. Quel que soit le détecteur que nous construisons, nous calculons au final un scalaire que nous seuillons ensuite afin d'obtenir la décision finale. Cette étape induit dans la plupart des cas des erreurs de classification comme illustré à la figure 1.19.

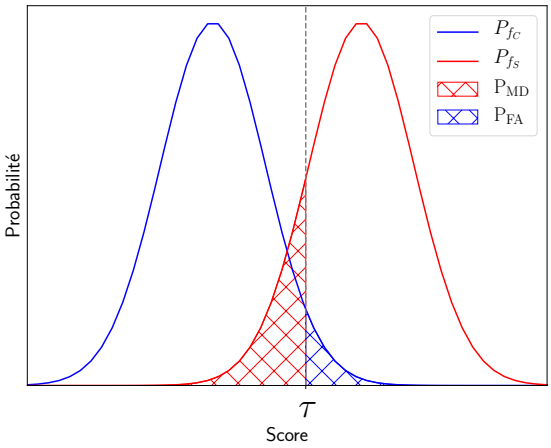


Figure 1.19: Construction d'un détecteur D attribuant chaque image à la réalisation d'une variable aléatoire générée par P_{f_S} ou P_{f_S}. τ représente le seuil de décision.

Pour un détecteur donné, le graphe de la fonction associant un taux de fausse alarme (P_{FA}) à un taux de détection de détection P_D est appelé courbe ROC (« Receiver-Operating-Characteristic ») et permet de décrire les performances du détecteur. Plus un détecteur est mauvais plus sa courbe ROC sera proche de la diagonale, et à l'inverse meilleur sera un détecteur, plus l'aire sous sa courbe (AUC : Area Under the Curve en anglais) ROC sera élevée.

À partir de cette courbe et afin d'évaluer la performance d'un détecteur le stéganalyste dispose ainsi d'un panel de plusieurs métriques, scalaires extraits à partir de la courbe ROC, qualifiant la performance du détecteur :

- AUC: l'aire sous la courbe ROC, $AUC = 0$ correspond à un détecteur qui coïncide avec la diagonale et qui génère des prédictions aléatoires, tandis que $AUC = 1$, correspond à un détecteur parfait.
- P_E: la Probabilité d'Erreur, il s'agit de la moyenne minimale des faux positifs (FA) et de faux négatifs (MD). Il s'agit donc du taux d'échec du classifieur.
- FP_{50} [56]: Taux de faux positifs (FA) pour 50% de faux négatif (MD).
- MD_5 [24]: Taux de faux négatifs (MD) à un taux de fausses alarme fixé à 5%.

Il est important de préciser que toutes ces métriques peuvent être obtenues à partir de la courbe ROC du classifieur D comme indiqué à la figure 1.20.

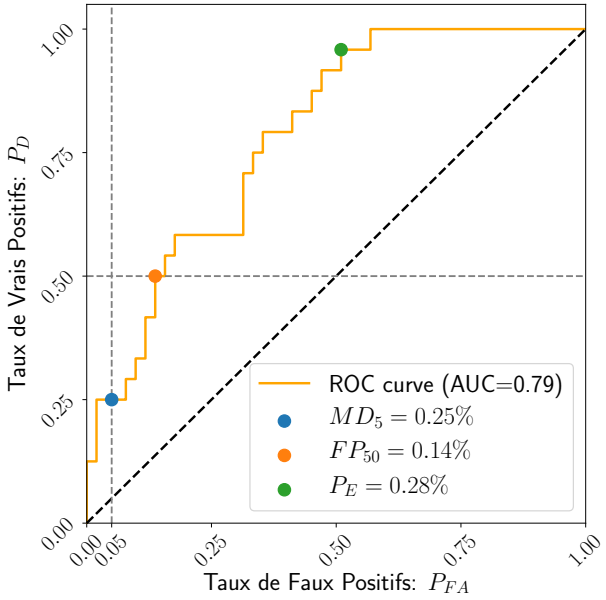


Figure 1.20: Courbe ROC d'un détecteur D arbitraire.

5.3 Stéganalyse statistique

Le scénario le plus favorable pour le stéganalyste se produit lorsque l'algorithme d'insertion est connu et qu'il existe un modèle statistique pour les images cover. Dans ce cas, il est ainsi possible de mettre en place une méthode de détection optimale en utilisant la théorie de la décision statistique, c'est ce qu'ont proposé R. Cogranne *et al.* dans [20].

Ainsi, en considérant la stéganalyse comme un problème de détection d'un signal ou d'une signature stéganographique, il est peut-être reformulé comme le test d'hypothèse(s) suivant:

Soit P_β^θ représente la distribution de l'image stego $\mathbf{Y}_\beta$ dont le taux d'insertion est β, le vecteur θ contient des paramètres relatifs à la source des covers comme la taille de l'image, le modèle du capteur photographique et ses paramètres.

Ainsi, lorsque les paramètres (β, θ) sont connus, le problème du stéganalyste devient alors celui de faire le choix entre les deux hypothèses suivantes :

$$\mathcal{H}_0 = \left\{\mathbf{X} \sim P_0^\theta\right\}, \tag{1.27}$$

et

$$\mathcal{H}_1 = \left\{\mathbf{X} \sim P_\beta^\theta\right\}, \tag{1.28}$$

ce qui revient à avoir à faire un choix sur l'origine de la distribution dont l'image x pourrait être une des réalisations. Le Lemme de Neyman-Pearson est un test de

rapport de vraisemblance (LRT : Likelihood Ration Test):

$$\delta^{\text{LRT}} = \begin{cases} \mathcal{H}_0 \text{ if } \Lambda(\mathbf{X}) = \frac{P_\beta^\theta[\mathbf{Y}]}{P_0^\theta[\mathbf{X}]} < \tau, \\ \mathcal{H}_1 \text{ if } \Lambda(\mathbf{X}) = \frac{P_\beta^\theta[\mathbf{Y}]}{P_0^\theta[\mathbf{X}]} \geq \tau. \end{cases} \tag{1.29}$$

Où τ le seuil de décision, est choisi pour rendre le LRT optimal, dans le sens suivant : parmi tous les tests qui garantissent une probabilité minimale de fausse alarme P_{FA}, le LRT est le test qui maximise la probabilité de détection correcte.

Ce type de méthodes de détection sont peu aisées à mettre en œuvre en pratique car nécessitent la connaissance des modèles statistiques très précis liés aux images covers et stegos.Cependant en 2019, J. Butora a présenté dans [14] une illustration très convaincante de la stéganalyse statistique. Il s'agit d'une méthode limitée aux facteurs de qualité $QF \in \{99, 100\}$ permettant la détection des images stego JPEG avec un taux d'erreur très faible. Cette méthode a été améliorée par R. Cogranne dans [21] en permettant de contrôler le taux de fausse alarme tout en améliorant encore les performances de détection.

La section suivante montre que lorsque les connaissances des distributions des images covers et stégos font défauts, le stéganalyste peut se tourner vers l'exploitation d'ensembles de caractéristiques permettant de réduire la dimension des images et de transposer le problème de stéganalyse comme un problème de classification de deux classes par un apprentissage machine supervisé.

5.4 Stéganalyse par extraction de caractéristiques et classification

Les images sont des objets dont la complexité ne rend pas leur modélisation faisable ou aisée. Ainsi, l'utilisation d'un ensemble de caractéristiques de dimension finie d jugées pertinentes au sens de la classification covers/stegos est souvent envisagée pour décrire les images afin de réduire leur dimension. Dans cette section les processus usuels de stéganalyse par des caractéristiques pré-définies par le stéganalyste seront brièvement expliqués.

En d'autres termes et pour reprendre les notions utilisées plus haut, il faut choisir un ensemble de caractéristiques telle que les clusters $\mathbf{f_X}$ et $\mathbf{f_Y}$ aient une zone de recouvrement qui soit la plus faible possible.

Ces caractéristiques sont par conséquent extraites en vue de l'entraînement d'un classifieur afin de discriminer les images covers et les images stegos. Il s'agit ainsi d'un processus se décomposant en deux étapes :

1. Extraction de caractéristiques pour décrire les images de la base.
2. Apprentissage supervisé de la frontière d'un classifieur (linéaire ou non) pour discriminer les images stegos des images covers.

Ce processus est succinctement illustré par la figure 1.21. La première étape

d'extraction de caractéristiques peut être vue comme une étape de réduction de dimension. A chaque image $\mathbf{X}$ de dimensions égale à son nombre de coefficient est associé un vecteur de caractéristiques : $\mathbf{f_X} \in \mathbb{R}^p$.

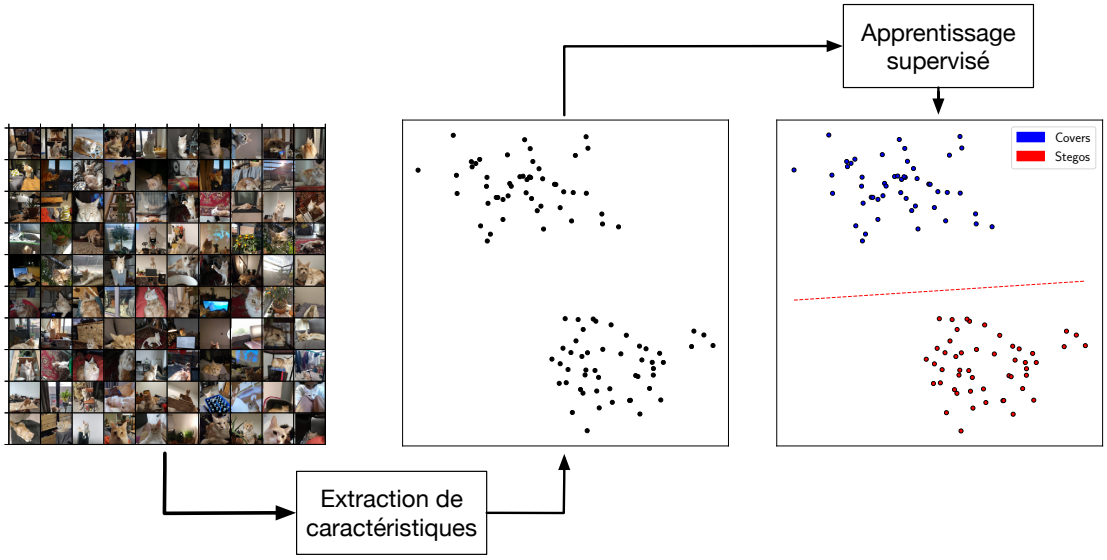


Figure 1.21: Illustration de l'apprentissage supervisé d'un classifieur arbitraire binaire D à partir d'une base d'images labellisées dont la dimension est réduite par une étape d'extraction de caractéristiques.

La dernière étape est la classification, permettant de conclure sur le caractère suspect ou non d'une image en décidant si une image appartient ou non à la classe cover ou stego.

L'opération d'apprentissage du classifieur est supervisée et celui-ci utilise deux bases disjointes, une base d'entraînement et une base de test afin de valider les performances du classifieur. Par analogie avec la géométrie, il est possible d'interpréter et d'illustrer sur la figure 1.21 l'étape de classification comme l'apprentissage de la frontière séparant au mieux les points bleus (représentant les contenus covers) des points rouges (représentant les contenus stegos) dans l'espace des caractéristiques. En effet, un classifieur linéaire repose sur une projection des caractéristiques $\mathbf{c}$ sur un vecteur de discrimination $\mathbf{w}$, l'opération est une somme pondérée des caractéristiques dont le but est de trouver $\mathbf{w}$ qui permet après projections sur les $\mathbf{c}$ de séparer les covers des stegos. Cette méthode est usuellement ramenée à un problème de minimisation au sens des moindres carrés dont une solution analytique peut être fournie par l'expression suivante :

$$\mathbf{w} = \left(\mathbf{F}^\top \mathbf{F}\right)^{-1} \mathbf{F}^\top \mathbf{l}, \tag{1.30}$$

où $\mathbf{F}$ regroupe les caractéristiques de toutes les images de la base covers et stegos

confondues et où $\mathbf{l}$ contient les labels de chaque image : « cover » ou « stego ». Afin d'augmenter les performances de classification et limiter un éventuel sur-apprentissage il est possible d'ajouter à ce vecteur $\mathbf{w}$ un coefficient de régularisation.

Les méthodes de classification linéaires ont donnés naissance à d'autres méthodes linéaires ou non, comme les méthodes de classification par ensemble qui s'appuient sur la classification par « Fisher Linear Discriminant » (FLD). Le classifieur par ensembles, largement utilisé en stéganalyse, remplace la règle de classification linéaire par un vote à la majorité. Le principe du classifieur par ensemble est ainsi de faire apprendre à un nombre L de classifieurs distincts une règle de décision. Pour ce faire, chacun de ces classifieurs est entraîné sur un sous-ensemble de caractéristiques tiré aléatoirement dont la dimension est plus faible que l'ensemble des caractéristiques initial.

Cependant les auteurs de [22] ont proposés en 2015 un classifieur linéaire de complexité calculatoire plus faible offrant des performances similaires au classifieur par ensemble.

5.4.1 Extraction de caractéristiques dans le domaine spatial

La stéganalyse d'images spatiales peut s'effectuer par une famille de caractéristiques appelée les SRM pour « Spatial Rich Models ».

Cette famille s'attache à extraire des images des caractéristiques qui ne dépendent pas de l'algorithme d'insertion, elles sont ensuite utilisées pour l'entraînement d'un classifieur à posteriori. L'extraction de ces caractéristiques, est dans ce cas, menée suivant la méthodologie illustrée à la figure 1.22 et dont les étapes sont les suivantes :

1. Chaque image est filtrée afin de réaliser l'extraction des résidus, définis comme la différence entre l'image originale et sa version dé-bruitée.
2. À partir de ces résidus, des histogrammes d'ordre n (ou des matrices de co-occurrences de dimension n) sont calculées comportant $(2T+1)^n$ bins (T le nombre de valeurs non signées possibles par dimension).
3. Chacun de ces histogrammes (ou matrices de co-occurrences) fournit ainsi des caractéristiques (après élimination de doublons qui correspondent aux symétries de l'image).

L'extraction de résidus est réalisée par des masques de convolutions linéaires. L'image de résidu i est ainsi calculée comme $\mathbf{R_I} = \mathbf{X}_i * \mathbf{K}_i$ où $\mathbf{K}_i$ est le noyau de convolution du filtre i. Afin d'augmenter la diversité des résidus, les résidus obtenus à partir de filtres non-linéaires sont également ajoutés à cette liste. Ce sont principalement le minimum et le maximum calculés sur des voisinages différents.

Les modèles riches ont ainsi donné lieu à deux ensembles de caractéristiques dont la dimension est 34671, une version amputée de la moitié de ses caractéristiques, plus rapide à calculer SRMQ1 de dimension plus faible (12753) qui offre des

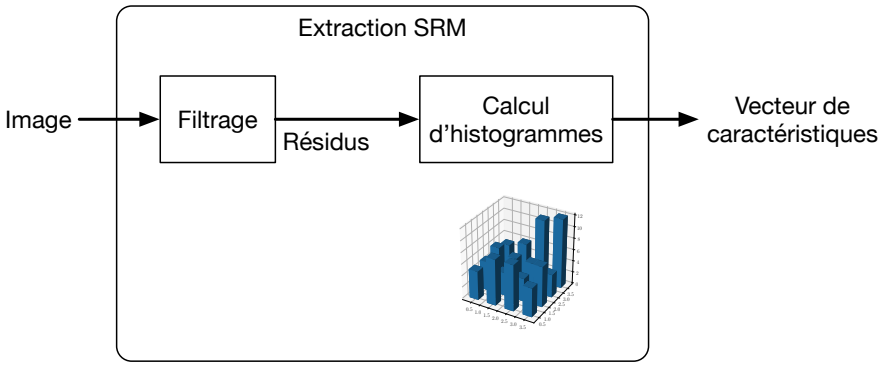


Figure 1.22: Schéma simplifié d'extraction de caractéristiques SRM.

Nom	Dimensions
SPAM [75]	686
SRM [43]	34671
SRMQ1 [43]	12753
PSRM [52]	12870
maxSRM [35]	34671
SCRMQ1, CRMQ1 [45]	12753 + 5404
CFA-aware CRM [44]	-
sigma-features [34]	1980

Table 1.2: Ensembles de caractéristiques et leur dimension pour le domaine spatial.

performances proches.

Il existe d'autres ensembles de caractéristiques dont une liste peu exhaustive se trouve dans le Tableau 1.2.

Les caractéristiques SRM peuvent être utilisées sur des images JPEG, mais il est possible d'obtenir des performances plus élevées en construisant des caractéristiques calculées à partir des statistiques sur les coefficients DCT. C'est le cas des caractéristiques DCTR.

5.4.2 Domaine JPEG

La stéganalyse dans le domaine JPEG est un des sujets les plus actifs de la communauté en stéganalyse, cela peut être justifié par la popularité de ce format de fichier qui est de très loin le plus utilisé pour le stockage et la communication d'images sur internet.

Il existe un nombre important d'outils permettant d'extraire des caractéristiques à partir d'images numériques. Par analogie avec les SRM, une méthodologie similaire peut être envisagée afin de capturer les dépendances qui peuvent coexister au sein

Nom	Dimensions
CHEN [17]/CC-CHEN	486/976
LIU [71]	216
PEV [76]/CC-PEV [59]	548
CDF [63]	1234
CC-C300 [60]	48600
CF [62]	7850
CC-JRM [61]	22510
DCTR [49]	8000
PHARM [51]	12600
GFR [80]	17000
SCA-DCTR/GFR/PHARM [36]	-

Table 1.3: Ensembles de caractéristiques "hand-crafted" et leur dimension pour le domaine JPEG.

d'une image JPEG. Le Tableau 1.3 en fournit une liste de ces caractéristiques dans le domaine JPEG.

Dans ce manuscrit nous utiliserons principalement les DCTR pour leur faible complexité et leurs performances. De ce fait, la sous-section suivante décrit succinctement leur fonctionnement.

Caractéristiques DCTR Les DCTR [49] sont un ensemble de caractéristiques conçues comme des statistiques de premier ordre (histogrammes) des résidus de bruit quantifiés obtenus à partir de l'image JPEG décompressée en utilisant 64 noyaux issus de la DCT non-quantifiée. Cette approche peut être interprétée comme un équivalent aux SRM mais dans le domaine DCT. Les aspects les plus attrayants de ces caractéristiques pour la stéganalyse sont sa faible complexité de calcul, sa faible dimensionnalité par rapport à d'autres modèles plus riches et ses bonnes performances.

Le calcul des DCTR peut être décomposé par les étapes suivantes :

1. La décompression de l'image JPEG : La version DCT décompressée de l'image JPEG peut s'écrire comme un ensemble de 64 convolutions 2D de l'image JPEG initiale avec les 64 motifs DCT $\mathbf{B}^{(k,l)}$ tel que :
$$\mathcal{U}(\mathbf{X}) = \left\{\mathbf{U}^{(k,l)} | 0 \leq k, l \leq 7\right\},$$
avec :
$$\mathbf{U}^{(k,l)} = \mathbf{X} \star \mathbf{B}^{(k,l)}.$$
Les motifs 8×8 de la base DCT $\mathbf{B}^{(k,l)} = \left(B_{mn}^{(k,l)}\right), 0 \leq m, n \leq 7$ sont générés

comme :

$$B_{m,n}^{(k,l)} = \frac{w_k w_l}{4} \cos \frac{\pi k(2m+1)}{16} \cos \frac{\pi l(2n+1)}{16},$$

où $w_i = \begin{cases} \frac{1}{\sqrt{2}} & i = 0, \\ 1 & i \neq 0. \end{cases}$

2. Le calcul des résidus par convolution avec des filtres passe-haut (bases DCT) : Pour chaque mode DCT de coordonnées $(k,l) \in [0,7]^2$ et chaque coordonnée relative $(a,b) \in [0,7]^2$, avec les $\mathbf{U}_{a,b}^{(k,l)}$, des sous-matrices de $\mathbf{U}^{(k,l)}$ telles que :

$$\mathbf{U}_{a,b}^{(k,l)} \in \mathbb{R}^{\frac{M-8}{8} \times \frac{N-8}{8}}.$$

Pour obtenir 64 sous-images $\mathbf{U}^{(k,l)}$ pour chaque couple (k,l) l'image DCT filtrée non-quantifiée $\mathbf{U}^{(k,l)} = \cup_{a,b=0}^{7} \mathbf{U}_{a,b}^{(k,l)}$ est ainsi le résultat d'un sous-échantillonnage.

3. La formation de statistiques de premier ordre : Des histogrammes sur 64 sous-treillis (phases JPEG) après quantification des valeurs absolues de tous les éléments du DCT non-quantifiés.

$$\mathbf{h}_{a,b}^{(k,l)}(r) = \frac{1}{\left|\mathbf{U}_{a,b}^{(k,l)}\right|} \sum_{u \in \mathbf{U}_{a,b}^{(k,l)}} \left[Q_T(|u|/q) = r\right].$$

avec Q_T désignant un opérateur de quantification, q est le pas de quantification correspondant pour le mode courant et le facteur de qualité sélectionné. Enfin le terme entre les crochets d'Iverson $[\square]$ est égale à 1 si l'assertion entre crochet est vraie, 0 sinon.

4. Les histogrammes extraits à partir des 64 sous-images peuvent ainsi être combinés afin de réduire la dimension de l'espace des caractéristiques en utilisant les propriétés de symétrie.

Les caractéristiques DCTR augmentent par exemple de 3% les performances de détection en comparaison aux caractéristiques SRMQ1 sur SI-UNIWARD pour un facteur de qualité $QF95$ et un taux d'insertion fixé à 0.5 bpnzAC. Ces résultats ont été obtenu sur BOSSbase 1.01 [5] contenant 10 000 images de taille 512×512 en niveaux de gris et la stéganalyse et été réalisée en employant un classifieur linéaire de type FLD [22].

5.5 Stéganalyse par réseaux de neurones profonds

Depuis quelques années, l'apprentissage machine et en particulier les réseaux de neurones profonds démontrent leur supériorité dans bien des domaines du traitement des signaux et la stéganalyse ne fait pas exception.

5.5.1 Vue d'ensemble d'un réseau de neurones

Réseaux de neurones artificiels Un neurone ou perceptron est définit comme une unité de calcul basique disposant de plusieurs entrées mais d'une seule sortie. Dans notre cas, le premier étage est composé d'un classifieur linéaire qui est activé à partir d'un certain seuil. Chaque entrée se trouve pondérée avec un poids dont le but est de donner plus d'importance à certaines entrées qu'à d'autres. De ce fait, l'entraînement ou l'apprentissage d'un neurone se fait lors de cette phase d'ajustement de ces poids. Le vecteur des entrées est multiplié par celui des poids et une fonction d'activation est appliquée en sortie.

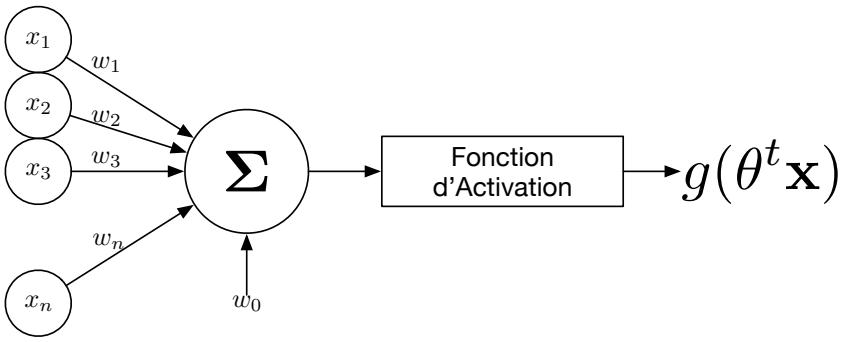


Figure 1.23: Schéma d'un perceptron avec $\theta^t = (w_0, w_1, \ldots, w_n)^t$ pour vecteur des poids, $\mathbf{x}$ le vecteur des entrées et g la fonction d'activation : sigmoid, softmax, etc.

Pour simplifier, l'apprentissage profond peut être considéré comme l'arrangement de perceptrons en de nombreuses couches cachées en plus de la couche d'entrée et celle de sortie. Le but étant de permettre à ces neurones d'identifier des schémas complexes entre les entrés par l'ajustement des poids de chacun des neurones de chaque couche.

L'étape d'apprentissage commence par la transmission des caractéristiques d'entrée aux neurones qui ont des poids initialisés à une valeur aléatoire. Les sorties des neurones de la couche d'entrée sont transmises à ceux des couches cachées. Il s'agit de la propagation par l'avant (forward propagation), pour laquelle on propage le vecteur des entrées à travers le réseau afin d'obtenir le résultat et de comparer celui-ci à la valeur réelle du label correspondant, on obtient ainsi l'erreur.

Ensuite, pour minimiser l'erreur, on exploite la rétro-propagation qui consiste de ce fait à propager le gradient de l'erreur pour réaliser la mise à jour des coefficients/poids de chaque neurone de la dernière couche vers la première couche afin que la fonction coût puisse être minimisée.

Cette étape est répétée jusqu'à ce que l'erreur moyenne du réseau sur sa base d'entraînement converge vers un minimum local.

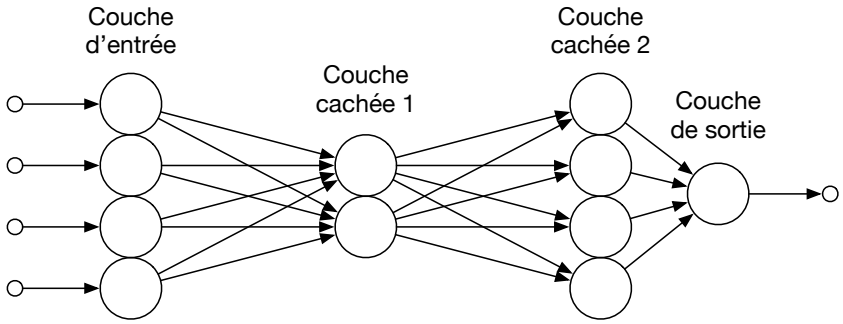


Figure 1.24: Schéma d'un réseau de neurones.

La descente de gradient stochastique permet ici de trouver un minimum local dans la quête de minimisation de l'erreur du réseau lors de l'étape de propagation par l'arrière.

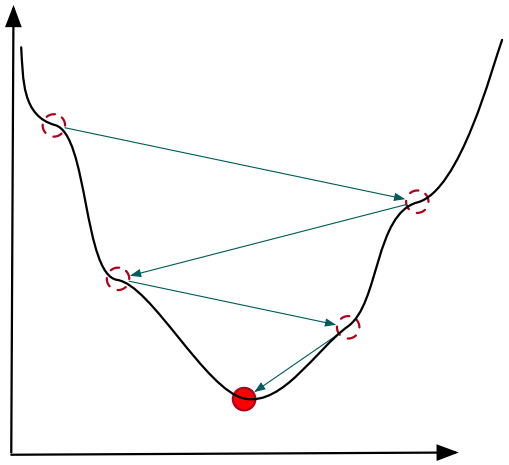


Figure 1.25: Illustration de la recherche d'un minimum local par un descente de gradient.

Réseaux de neurones convolutifs Les réseaux de neurones convolutifs (Convolutionnal Neural Network), très populaires en vision assistée par ordinateur sont basés sur la même architecture que celle illustrée à la figure 1.24. Des couches permettant des opérations de convolutions sont cependant ajoutées dans le but de réaliser des opérations de filtrages sur l'image pour en extraire des caractéristiques voir figure 1.26. Les couches cachées d'un CNN sont constituées de couches convolutives, de couches de fusion (afin de sélectionner les plus grandes valeurs sur les cartes des caractéristiques i.e les couches convolutives et de les utiliser comme données d'entrée pour les couches suivantes), de couches entièrement connectées (où toutes les entrées d'une couche sont connectées à chaque unité d'activation de

la couche suivante) et de couches de normalisation. Les opérations de convolutions sont réalisées par un noyau de convolution dont les coefficients sont obtenus par apprentissage à l'aide de la rétro-propagation. La fusion des cartes de caractéristiques est un processus de discrétisation dont le but est de sous-échantillonner une matrice de sortie en réduisant sa dimensionnalité. Elle permet de faire une sélection des caractéristiques contenues dans des sous régions de l'image. Enfin, les sorties des couches convolutives et les couches de fusion représentent des caractéristiques de haut niveau de l'image d'entrée. L'objectif de la couche dense est d'utiliser ces caractéristiques pour classer l'image d'entrée dans différentes classes en fonction de l'ensemble de données d'entraînement. Par exemple, la tache de classification de l'image en stéganalyse permet en sortie des couches denses, d'obtenir une probabilité que l'image d'entrée soit cover ou stego.

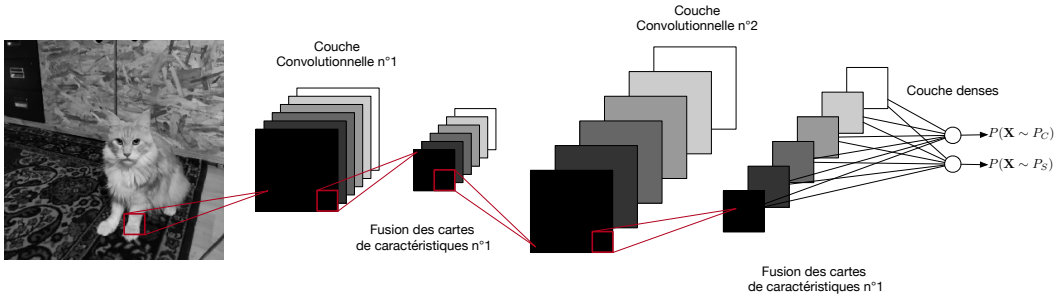


Figure 1.26: Schéma d'un CNN arbitraire extrayant des caractéristiques à partir d'une image.

L'état de l'art en stéganalyse est porté à l'heure où ces lignes sont écrites par des réseaux de neurones dont les trois architectures sont les suivantes :

- SR-Net [13][JPEG + Spatial]
- Xu-Net [96][JPEG]
- Yedroudj-Net [99][Spatial]

Dans les trois cas il s'agit de réseaux de neurones (profonds) convolutifs, qui prennent en entrée de leur architecture une image brute et sont capables d'apprendre de manière jointe l'espace latent des images covers et stego. Ils peuvent ainsi pour une image $\mathbf{X}$ fournir une décision sur la classe à laquelle elle appartiendrait ($\mathbb{P}\{\mathbf{X} \sim P_S\}$) comme expliqué à la figure 1.27.

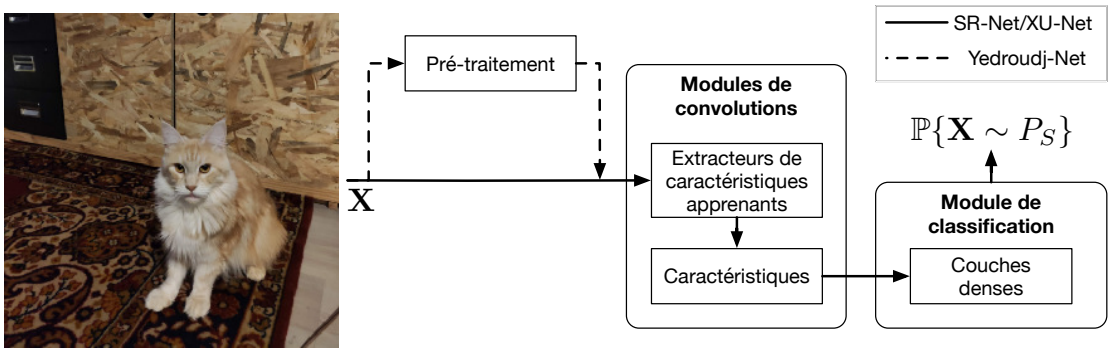


Figure 1.27: Illustration de la classification d'une image $\mathbf{X}$ par classifieur D arbitraire pré-entraîné.

Le processus d'apprentissage d'un tel détecteur est similaire à celui utilisé pour la stéganalyse par caractéristiques, la différence principale est qu'ici les caractéristiques sont directement apprises sur la base d'entraînement. L'avantage de cette méthode est que les caractéristiques sont ainsi sensibles à la base d'apprentissage. Une base d'apprentissage suffisamment générale permettrait en théorie d'obtenir des résultats sur un nombre de sources plus large.

Cependant, le problème de cette approche est qu'elle nécessite un très grand nombre d'images pour y parvenir. Dans l'architecture Yedroudj-Net, le pré-traitement est assuré par une banque de 30 filtres passe-haut [98] afin d'accélérer l'éventuelle convergence du CNN.

5.6 Conclusions du chapitre

À la lumière des sections précédentes, il est possible de réaliser une analogie entre stéganographie/stéganalyse et le jeu du chat et de la souris. En effet, lorsque l'un va développer des méthodes pour mieux se cacher dans son environnement, l'autre va développer des méthodes plus efficaces pour chasser. Le fait que ces deux disciplines soient développées en compétition nous invite à des directions de réflexion tenant compte de l'état de l'art en stéganalyse pour la création de nos algorithmes d'insertion. Il est de ce fait essentiel de développer des méthodes d'évaluations des algorithmes de stéganographie comme de stéganalyse afin de satisfaire les deux mondes dans une quête de performances de sécurité ou de détection.

Si l'objectif initial de ce chapitre était de donner au lecteur un aperçu général des notions essentielles en stéganographie et en stéganalyse, ce chapitre a également présenté des points clés qui sont repris dans la suite de ce document, à savoir :

- les méthodes d'insertion préservant un modèle, notamment la Stéganographie Naturelle,

- l'utilisation de l'information adjacente,
- l'importance de la synchronisation des modifications,
- les méthodes de stéganographie dans le domaine JPEG,
- les méthodes de stéganalyse dans le domaine JPEG.

Chapitre 2

Développement d'une image RAW en JPEG : notions de bases

Ce chapitre est dédié à la modélisation de l'acquisition d'images par des capteurs photographiques. Ces capteurs sont ainsi la première étape de la manifestation d'imperfections liées au processus d'acquisition et/ou de sauvegarde de l'image. Ces imperfections sont essentielles en stéganographie, il s'agit de bruit lié à la capture de l'image, de variations liées au capteur photographique, etc. Plus généralement Alice va être intéressée par toutes sources d'incertitudes modélisables et non-stationnaires pouvant ainsi être utilisées en stéganographie.

1 Acquisition d'une image par un capteur photographique

Cette section est consacrée aux capteurs photographiques, qui sont à la source même de la formation des images digitales. Nous étudierons brièvement l'acquisition des images par des capteurs photographiques grand public afin de fournir les bases nécessaires à l'utilisation de ce système d'acquisition et de traitement pour la stéganographie.

1.1 Images brutes

Il existe deux technologies populaires utilisées pour la réalisation de capteurs photographiques : CCD (Charge-Coupled Device) et CMOS (Complementary Metal-Oxide-Semiconductor) mettant en application l'effet photoélectrique. Cet effet permet de quantifier la quantité de photons reçue par chaque cellule photographique (aussi appelée photo-site) et de créer une représentation numérique de l'état du capteur après capture de la scène. Les deux technologies utilisent le même principe :

en premier lieu les photons sont absorbés et une charge fonction de leur nombre est générée par effet photoélectrique. La différence entre CCD et CMOST réside dans le transfert de cette charge et sa conversion en une tension.

Les cellules photographiques (photo-sites) enregistrent tous les photons incidents dans le domaine visible produisant une image en niveaux de gris. La lumière visible pouvant être décrite par une superposition d'ondes électromagnétiques de longueur d'ondes $\lambda \in [380, 750]$ nanomètres, afin de produire une image en couleur il est nécessaire d'utiliser un filtre de couleur permettant le passage d'une gamme plus faible de longueurs d'ondes.

Pour ce faire on place devant chaque photo-sites un photo-filtre laissant passer le rouge, le vert ou le bleu (dans le cas d'un filtre de Bayer) le principe de ce filtrage est expliqué à la figure 2.2. Cet ensemble de filtres est appelé un CFA (« Color Filter Array » en anglais), il en existe plusieurs types, mais ils remplissent la même fonction. Comme indiqué à la figure 2.1, dans le cas du motif de Bayer il s'agit de paver l'ensemble de la surface du capteur avec le motif représenté sur la figure 2.3. · · Un photo-site donné produira une tension proportionnelle à l'éclairement

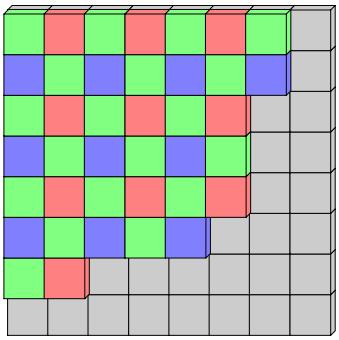
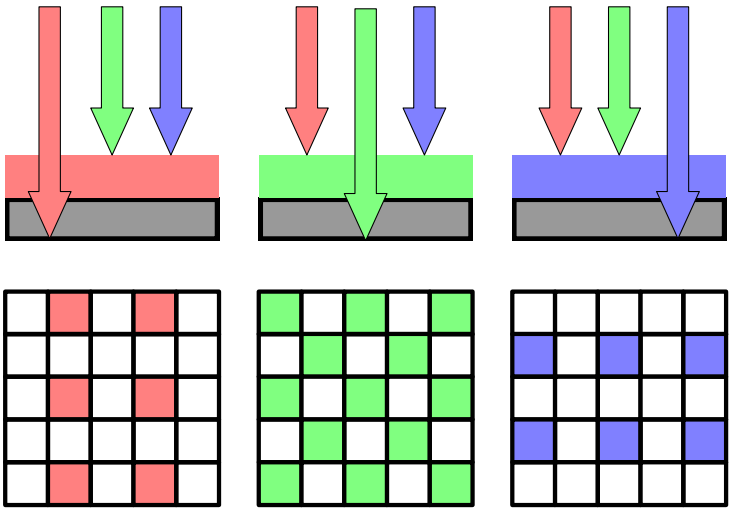


Figure 2.1: Motif de Bayer.

par la composante monochromatique du filtre derrière lequel il se trouve.

Le capteur produit ainsi une matrice de valeurs proportionnelles à l'éclairement des photo-sites pour chaque composante couleur, et ces trois composantes sont entrelacées (voir figure 2.2). Afin de refléter la sensibilité de l'œil humain aux couleurs, le filtre de Bayer contient deux fois plus de cellules représentant la couleur verte. Ainsi la moitié des pixels portent l'information de la composante verte, un quart des pixels porte l'information de la composante rouge et un quart portent l'information de la composante bleue. Notons également que d'autres filtres existent, mais ils ne seront pas abordés ici car moins populaires que filtre de Bayer.



(a) R (b) G (c) B

Figure 2.2: Décomposition en composantes monochromatiques (R, G, B) d'une onde lumineuse par les filtres de la matrice de Bayer.

1.2 Dématriçage

La formation d'une image digitale couleur est réalisée par interpolation des canaux couleurs (aussi appelé dématriçage). Le voisinage de chaque photo-site est mis à profit afin de prédire pour ce photo-site, sa valeur pour les deux canaux couleurs manquants.

Dans le cas du motif de Bayer, illustré à la figure 2.3 il existe de nombreuses méthodes permettant de réaliser l'interpolation couleur. Parmi celles-ci, nous pouvons mentionner le dématriçage bi-linéaire qui est la forme la plus simple permettant de réaliser cette tâche. Cependant afin d'obtenir des images de meilleure qualité visuelle, d'autre algorithmes ont été développés utilisant par exemple les gradients, permettant de réaliser une interpolation plus précise le long des contours. Pour ne pas surcharger le document, seul l'algorithme de dématriçage bi-linéaire sera abordé ici.

Le dématriçage par **interpolation bilinéaire** consiste à partir de l'image RAW, brute de capteur, notée $\mathbf{I}^{RAW}$, à estimer les niveaux manquants des sous-images $(\mathbf{I}_R^{RAW}, \mathbf{I}_G^{RAW}, \mathbf{I}_B^{RAW})$ (respectivement les canaux couleurs R, G, B) en calculant la moyenne des plus proches voisins. Pour cela il suffit d'appliquer un filtre de convolution à chaque sous-image afin d'obtenir les canaux (R, G, B). La composante verte $\mathbf{G}$ est calculée à partir de :

$$\mathbf{G} = \mathbf{I}_G^{RAW} * \mathbf{K}_G, \tag{2.1}$$

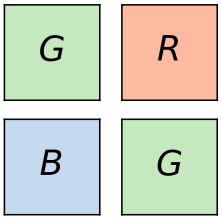


Figure 2.3: Motif de Bayer.

et les composantes rouges $\mathbf{R}$ et bleue $\mathbf{B}$ s'écrivent respectivement :

$$\begin{aligned} \mathbf{R} &= \mathbf{I}_R^{RAW} * \mathbf{K}_R, \\ \mathbf{B} &= \mathbf{I}_B^{RAW} * \mathbf{K}_B, \end{aligned} \tag{2.2}$$

avec : $\mathbf{K}_R = \mathbf{K}_B = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$ et $\mathbf{K}_G = \frac{1}{4}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}$

1.3 Opérations additionnelles

Le terme développement fait généralement référence aux différentes opérations que le fichier RAW subit (si traitement il y a), et, pour des capteurs couleurs le dématricage est la toute première de ces opérations. Cependant la chaîne de développement peut également comporter d'autres opérations, incluant ainsi en plus du dématriçage, la balance des blancs, une correction couleur, une correction gamma, du dé-bruitage, la correction de la netteté, etc.

Nous présentons rapidement les plus importantes de ces opérations.

La balance des blancs est un ajustement multiplicatif utilisé pour corriger les différences de spectre de la lumière ambiantes, elle est notamment utilisée pour faire en sorte que les zones blanches d'une scène après capture demeurent blanches sur la photo finale. Elle est définie comme :

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} := \begin{bmatrix} g_R & 0 & 0 \\ 0 & g_G & 0 \\ 0 & 0 & g_B \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}, \tag{2.3}$$

où les constantes g_R, g_G, g_B représentent les gains respectifs des canaux couleurs R, G et B.

Après dématriçage, le signal peut également être sujet à une correction couleur. Le but est d'ajuster les quantités respectives de rouges, verts et bleus afin que l'image soit correctement visible sur un écran. Il s'agit, également d'une opération

linéaire pouvant être définie comme :

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} \coloneqq \begin{bmatrix} c_{1,1} & c_{1,2} & c_{1,3} \\ c_{2,1} & c_{2,2} & c_{2,3} \\ c_{3,1} & c_{3,3} & c_{3,3} \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{2.4}$$

La réponse du capteur photographique étant linéairement proportionnelle à l'intensité lumineuse, elle diffère de la réponse de l'œil humain à la lumière (qui offre une réponse logarithmique à la lumière). De ce fait, le signal peut être corrigé par une transformation gamma :

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} \coloneqq \begin{bmatrix} R^{\gamma} \\ G^{\gamma} \\ B^{\gamma} \end{bmatrix}. \tag{2.5}$$

Ces opérations sont généralement accompagnées de dé-bruitage, de correction des pixels défectueux, de correction de la netteté, etc. Le format de sauvegarde est également une des opérations généralement inclue dans le développement, réalisée à la fin. La section 2 fournit plus de détails sur le processus de sauvegarde d'un fichier JPEG.

1.4 Bruits d'acquisition

Inévitables lors de l'acquisition d'une image, il existe de nombreuses sources de bruits ayant la capacité d'influer sur l'image.

Parmi ces bruits, certains sont issus de processus aléatoires, tandis que d'autres sont indissociables du capteur et donc sont répétés quasiment à l'identique sur deux captures de la même scène.

Ces imperfections quelle que soient leurs source peuvent trouver leur genèse durant l'acquisition de l'image (i.e. le comptage des photons par les photo-sites) ou durant la lecture de la valeur des photo-sites.

Nous présentons ici des réalisations du **bruit photonique** qui, comme évoqué précédemment, sera utilisé tout au long du manuscrit. Il est lui lié à la nature quantique de la lumière. Cela permet entre autres de considérer le nombre de photons incident sur chaque photo-site comme une variable aléatoire suivant une loi de Poisson et pouvant être approximée par une loi Normale [38]. Ce bruit est fonction de nombreux paramètres dont la qualité du capteur, la taille de la cellule photo-sensible, le temps d'exposition et la sensibilité du capteur (ISO).

En capturant une image dans le noir complet il est possible de voir apparaître des pixels dont la valeur sera différente de la valeur nulle (voir figure 2.5). Ce phénomène est relatif aux imperfections des cristaux de silicium composant le capteur lui-même, ce bruit appelé **courant de fuite** (« dark current » en anglais) dépend essentiellement de la sensibilité et de la température du capteur.

Une autre conséquence des imperfections de la matrice de silicium et de la

(a)

(b) ISO_{2000} (c) ISO_{8000} (d) ISO_{16000} (e) ISO_{32000}

Figure 2.4: Illustrations du bruit photonique lors de la capture de la même scène pour différentes sensibilités ISO avec un Sony A7III.

Figure 2.5: Illustration du courant de fuite pour le Sony A7III équipé de son cache avec un temps d'exposition de 30 secondes et une sensibilité ISO à 800.

matrice de filtres CFA est la **non-uniformité de la photo-réaction**, i.e. le **PRNU**(Photo-Response Non-Uniformity en anglais). Il s'agit d'un artefact systématique qui n'a jusqu'à présent pas pu être utilisé en stéganographie mais qui trouve son usage en criminalistique de l'image [19, 18].

Notons que d'autres bruits liés à la capture des images existent, tels que le bruit lié à l'efficacité des transferts de charge (pour les capteurs CCD), le bruit d'amplification (lié à la collecte des charges électrique de la matrice de photo-sites) ou le bruit de quantification (lié à la perte d'information), mais leur étude ne sera pas faite ici.

2 Compression d'une image au format JPEG

Dans la chaîne de production d'une image, la compression JPEG est la dernière étape menant à l'enregistrement d'une image. L'acronyme JPEG correspond à « Joint Photographic Experts Group ». Il s'agit d'un comité d'experts qui édite des normes de compression d'images, cependant cet acronyme désigne en général une norme « ISO/CEI 10918-1 UIT-T Recommendation T.81 » qui résulte de travaux sur la compression d'images numériques datant de 1978. JPEG normalise ainsi un algorithme de compression et son décodage. Il existe deux classes de processus de compression JPEG, l'une avec pertes dans sa version populaire, et l'autre sans pertes qui est peu utilisée.

Les processus de compression et celui de décompression comportent chacun six étapes représentées à la figure 2.6.

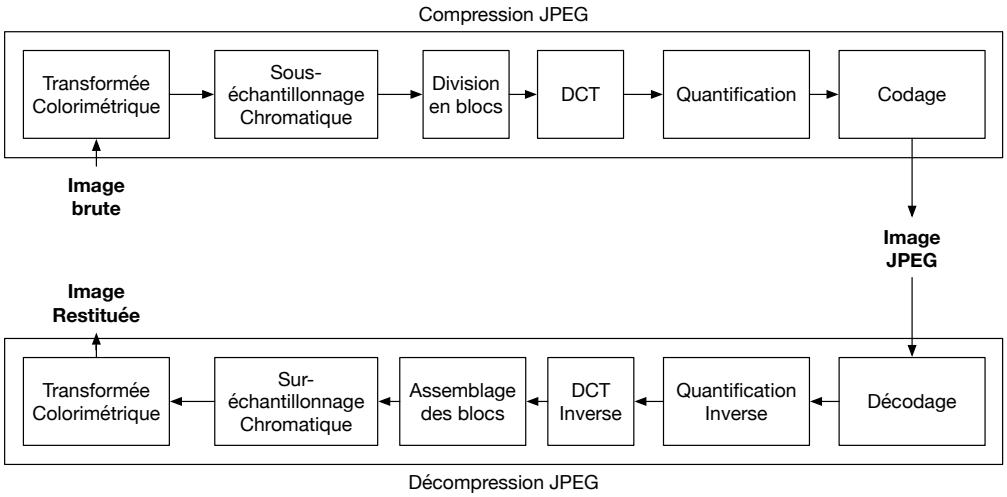


Figure 2.6: Vue d'ensemble du processus de compression JPEG.

2.1 Espaces de couleurs

Le modèle (Y, C_b, C_r) est une manière de représenter l'espace colorimétrique, dont l'origine remonte au passage en couleurs des transmissions vidéo hertziennes. Le but de ce modèle était initiallement d'assurer la compatibilité entre les récepteurs télévisions en noir et blanc et les récepteurs couleurs.

L'algorithme de compression JPEG est en mesure de traiter les couleurs sous n'importe quel espace colorimétrique, cependant nous verrons dans la section suivante qu'en tirant parti des défauts de sensibilité de l'œil humain à la chrominance (teinte) par rapport à la luminance, il est possible d'obtenir des taux de compression plus intéressants.

En général l'image d'entrée est une image (R, G, B), dont les composantes sont les 3 canaux : Rouge, Vert et Bleu. Elle sera ainsi transposée dans le domaine (Y, C_b, C_r) dont les composantes représentent respectivement la luminance, la chrominance bleue, et la chrominance rouge.

Pour accompagner cette section, la figure 2.7 illustre la décomposition d'une image couleur sur les canaux (R, G, B) tandis que la figure 2.8 représente la décomposition (Y, C_b, C_r) pour la même image couleur.

(a) Original (b) R (c) G (d) B

Figure 2.7: Illustration de la décomposition en canaux (R, G, B) d'une image couleur.

2.2 Sous-échantillonnage chromatique

L'œil humain étant moins sensible aux changements de chrominance que de luminosité, les canaux chromatiques C_b et C_r peuvent être sous-échantillonnés avant d'appliquer la transformée en cosinus discrets (DCT) afin d'atteindre un meilleur rapport de compression.

(a) Original (b) Y (c) C_B (d) C_R

Figure 2.8: Illustration de la décomposition en canaux (Y, C_b, C_r) d'une image couleur. Nous pouvons remarquer que les composantes C_b et C_r comportent visuellement moins d'informations que la luminance.

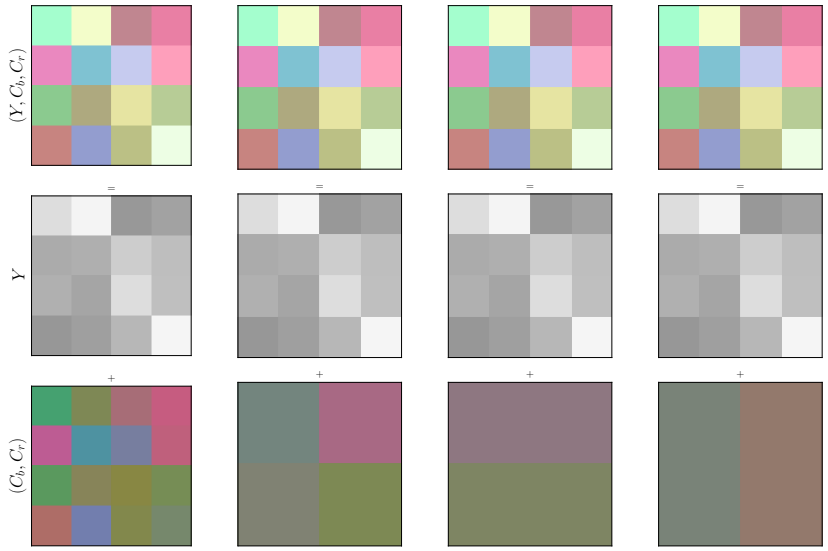
Cette opération est exécutée en divisant l'image en macro-blocs de 16×16 pixels. Chaque macro-bloc produit 4 blocs de taille 8×8, si un bloc donné est sous-échantillonné par un facteur 2 dans chaque direction, alors chaque macro-bloc aura ainsi un seul bloc de chrominance C_b et C_r associés de taille 8×8 (et 4 blocs de luminance). Le sous-échantillonnage de l'exemple précédemment utilisé est noté : $4:1:1$. Si les canaux chromatiques ne sont pas sous-échantillonnés alors la notation associée est : $4:4:4$, $4:2:2$ signifie de le sous-échantillonnage des deux canaux chromatiques est effectué dans une seule direction. Ainsi, chaque type de sous-échantillonnage utilisé dans une opération de compression est spécifié par la notation : $J:a:b$, dont l'interprétation de la notation est la suivante : "J:" représente la largeur de la plus petite matrice de pixels considérée (généralement 4), "a": le nombre de composantes de chrominance dans la première ligne, enfin "b:", le nombre de composantes de chrominance supplémentaires dans la deuxième ligne.

L'exemple de la figure 2.9 illustre pour une image couleur aléatoire $4 \times 4 \times 3$ différents sous-échantillonnages chromatiques usuels.

L'opération de sous-échantillonnage conduit à une perte d'information qui ne peut être récupérée par la suite, il s'agit de ce fait d'une opération de compression irréversible.

2.3 Transformée DCT

Dans cette section nous présentons la transformée en cosinus discrets utilisée lors de la compression et nous détaillons sa mise en place et ses propriétés. Celle-ci s'applique sur une image spatiale après subdivisions de celle-ci en blocs de 8×8



(a) 4 : 4 : 4 (b) 4 : 2 : 2 (c) 4 : 1 : 1 (d) 4 : 2 : 0

Figure 2.9: Sous-échantillonnage de la chrominance : (a): Pas de sous-échantillonnage, (b) 1/2 résolution horizontale et résolution verticale intégrale, (c) 1/2 résolution horizontale et verticale, 1/4 résolution horizontale et résolution verticale intégrale.

pixels. La transformée DCT (Discrete Cosine Transform, en français transformé en cosinus discrets) est une transformation numérique qui est appliquée à chaque bloc. Cette transformée est une variante de la transformée de Fourier, qui se opère sur chaque bloc 8×8 séparément, et qui est considérée comme une fonction numérique à deux variables, la somme de fonctions cosinus oscillant à des fréquences horizontales et verticales différentes. Chaque bloc de chaque canal est ainsi décrit comme une combinaison linéaire de motifs appelés modes DCT. Pour un bloc de 8×8 pixels de luminance Y: $(B_{i,j})_{\substack{0 \leq i < 8 \\ 0 \leq j < 8}}$, on crée un bloc de 8×8 coefficients DCT $(d_{k,l})_{\substack{0 \leq k < 8 \\ 0 \leq l < 8}}$:

$$d[k,l] = \sum_{i,j=0}^{7} \frac{w[k]w[l]}{4} \cos\frac{\pi}{16}k(2i+1)\cos\frac{\pi}{16}l(2j+1)B_{i,j}, \tag{2.6}$$

avec :

$$w : x \mapsto \begin{cases} 1/\sqrt{2} & x = 0 \\ 1 & x \neq 0 \end{cases} \tag{2.7}$$

Les structures basse-fréquences sont concentrées dans le coin haut gauche de chaque bloc DCT.

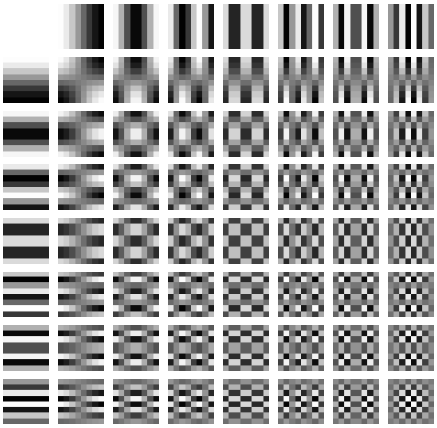
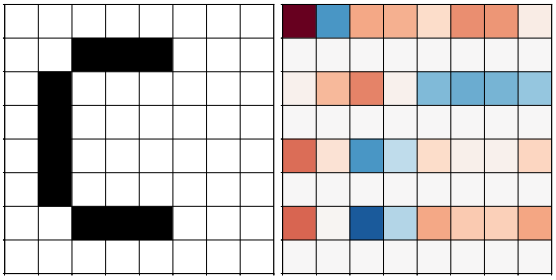


Figure 2.10: La transformée DCT permet d'exprimer chaque bloc de 8×8 pixels comme combinaison linéaire de ces 64 motifs (aussi appelés modes). Chaque motif représente les $(B_{i,j})_{(i,j) \in [\![0,7]\!]^2}$ pour chaque mode DCT (k,l).

La transformée DCT inverse est exprimée par :

$$B[i,j] = \sum_{k,l=0}^{7} \frac{w[k]w[l]}{4} \cos\frac{\pi}{16}k(2i+1)\cos\frac{\pi}{16}l(2j+1)d_{k,l}. \tag{2.8}$$

À titre d'exemple nous utiliserons un bloc représentant la lettre « C » sur 8×8 pixels, dont la représentation spatiale et en cosinus discrets sont respectivement visibles à la figure 2.11.



(a) Spatial (b) DCT

Figure 2.11: Chaque motif représente les $(B_{i,j})_{(i,j)\in[\![0,7]\!]^2}$. La figure (b) représente la figure (a) dans le domaine DCT, les coefficients tendant vers la couleur rouge témoignent d'une valeur positive et inversement pour les ceux signalés en bleu.

Comme évoqué dans cette même section, la transformée DCT permet d'exprimer chaque bloc spatial de 8×8 pixels en une combinaison linéaire de 64 motifs, pour l'exemple choisi, les termes de cette combinaison linéaires sont représentés à la figure 2.11, avec en rouge ceux dont les coefficients sont positifs et en bleu négatifs.

2.4 Quantification d'une image DCT

La quantification est l'étape de compression JPEG durant laquelle la majeure partie de la perte d'information a lieu. Cette étape induit alors une perte de la qualité visuelle.

L'objectif est ici d'atténuer les variations hautes fréquences de l'image, c'est-à-dire celles auxquelles l'œil humain est très peu sensible. Les informations structurelles essentielles (concentrées dans le coin en haut à gauche de chaque bloc) sont ainsi préservées au mieux pour représenter le bloc, il s'agit des variations basse-fréquences.

Les coefficients haute-fréquences ayant généralement dans les images naturelles des amplitudes faibles, ils sont d'avantage atténués par la quantification, certains coefficients DCT sont ainsi souvent ramenés à 0. Cela a pour effet de créer des artefacts structurels laissant apparaître la subdivision de l'image en blocs (8×8). Une autre source d'artefacts est liée à la différence entre les matrices de quantification du canal luminance et des canaux chromatiques, ce qui provoque des aberrations chromatiques sur certains blocs. En effet, l'œil humain étant plus sensible à la luminance qu'à la chrominance les matrices de quantification

chromatiques ont tendances à avoir des pas de quantification plus grand que ceux du canal luminance pour les mêmes modes.

Ces différents artefacts sont illustrés notamment à la figure 2.12.

Figure 2.12: Illustration de la quantification d'une image couleur avec un taux de compression en baisse et donc une qualité en hausse, de gauche à droite avec $QF \in \{5, 10, 25, 50, 100\}$.

Le calcul de la quantification peut être effectué de la manière suivante :

$$\mathbf{D}^q(u,v) = \text{round}\left(\mathbf{D}_{u,v} \oslash_{8,8} \mathbf{Q}_{QF}\right). \tag{2.9}$$

Et pour la quantification inverse :

$$\mathbf{D}^r(u,v) = \text{round}\left(\mathbf{D}_{u,v} \odot_{8,8} \mathbf{Q}_{QF}\right). \tag{2.10}$$

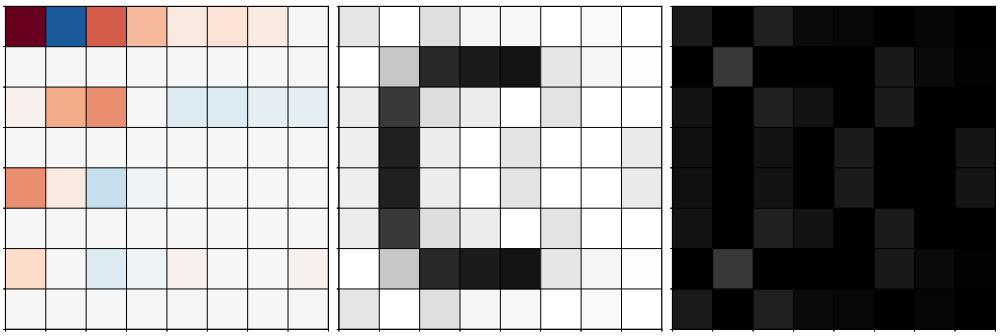
Après quantification en utilisant la matrice de quantification de luminance correspondant au facteur de qualité 50 est appliqué à l'image de l'exemple utilisé aux sections précédentes. L'image obtenue après DCT inverse est proche visuellement de l'image initiale. Mais, comme illustrées à la figure 2.13 les erreurs de reconstruction témoignent de la perte d'information liée à la quantification.

L'opération de quantification augmente ainsi fortement la redondance des données contenues dans le bloc. Cette redondance est exploitée dans l'étape de codage qui sera expliquée dans la suite.

2.5 Codage

Le codage du standard JPEG est une forme de compression sans pertes qui est réalisé en parcourant chaque bloc DCT par dans le sens d'un zigzag (voir figure 2.14). Ce parcours permet de concentrer les coefficients non-nuls (car la quantification élimine la plupart des coefficients hautes-fréquences), et coder plus efficacement les longues suites de zéros générés après quantification.

Ainsi, chaque bloc DCT quantifié, est parcouru selon un zigzag et la séquence résultante est terminée par un caractère signifiant la fin de la séquence de coefficients



(a) (b) (c)

Figure 2.13: (a): bloc DCT après quantification, (b): bloc spatial après quantification et (c): erreurs de reconstruction liées à une quantification à $QF50$ par le module “Pillow” de Python.

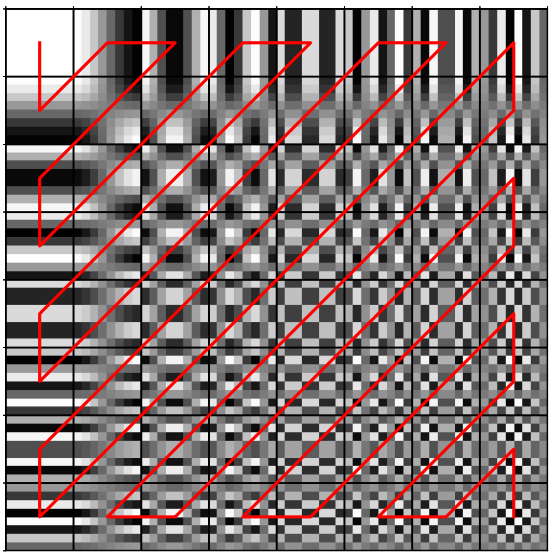


Figure 2.14: Utilisation d’un zigzag pour parcourir les coefficients DCT d’un bloc .

non-nuls.

Cette séquence est ensuite compressée en utilisant un algorithme de codage RLE (Run-length encoding) dont la sortie est une séquence plus courte, sans les longues suites de zéros. Un codage entropique (de type Huffman) est ensuite utilisé, favorisant le codage efficace des suites de valeurs les plus fréquentes.

3 Conclusions du chapitre

Ce chapitre a permis de présenter les bases techniques liées au contenu cover, à savoir une image numérique initialement au format RAW (contenant un bruit photonique) qui est ensuite développée en une image au format JPEG après avoir subit différents traitements (dématriçage, transformation couleur, transformation Gamma).

Après un premier chapitre sur la stéganographie et ce chapitre sur le processus de formation d'une image JPEG, nous présentons dans les chapitres suivant les contributions liées à cette thèse. Dans un premier temps la modélisation mathématique du bruit photonique dans le domaine DCT, puis dans un second et troisième temps, deux méthodes de stéganographie qui s'appuient sur cette modélisation.

Chapter 3

Statistical properties of the sensor and the development pipeline

The principle of Natural Steganography (NS) [10, 6, 30, 84], plainly detailed in the next chapter, is based on the same principle as model based steganography (see section 4.2.1) since it embeds message whose associated stego signal tries to mimic the statistical properties of the camera photonic noise, a.k.a. camera shot noise (see sub-section 1.4).

The purpose of this chapter is to obtain explicit informations about the statistical properties of both the photonic noise and the development pipeline. We have conducted this analysis for both different sensors and different pipelines. This chapter addresses different statistical properties, and proposes different analyses which are all listed below:

- Section 1.1 analyses the independency property of the photonic noise between photo-site neighbors. The Gaussian assumption is also checked for different sensors.
- Section 2 analyses the multivariate distribution of the sensor noise in the DCT domain, considered as Gaussian. A linear mathematical model is proposed which enables to obtain an explicit formulation of the covariance matrix.
- Section 4 leverages the empirical estimation of the covariance matrix for stationary noise to analyze a development pipeline which is closed to the one used to generate BOSSBase.
- Finally, section 4 analyses the impacts of different software and hardware developments.

1 Verification of working assumptions

1.1 Motivations

From the steganographic point of view it is essential to identify the components that are dependent on each other because their modifications would necessarily lead to a difference in the statistical properties of the stego image compared to its cover version. Indeed it can be shown that some of the coefficients of an image are dependent on each other and thus a naive modification, unaware of these dependencies could alter these them and thus become detectable. Therefore, to create a stego image preserving the model of its cover, Alice has to investigate and model the mutual impact of any change performed from the initial cover image.

For two elements of an independent cover image x, y whose values are issued for the random values X, Y, if these two random variables are independent it is possible to write their joint distribution as the product of their marginal distribution:

$$P(X \leq x, Y \leq y) = P(X \leq x)P(Y \leq y). \tag{3.1}$$

The fact that the components (photo-sites, pixels or DCT coefficients) are independent allows a simplified writing of the statistical model, and within the framework of steganography it allows to use additive costs since a change made to one of the coefficients of the image is not influenced by the changes made to the other coefficients. However the independence of the covered images is a rather particular case, the M9 for example is a monochrome sensor and in the absence of demosaicking the pixels are thus independent between them.

In this section we thus use the mutual information to empirically characterize the possible independence degree of the photo-sites of different sensors.

1.2 Independence between photo-sites

Mutual information is an information theoretic measure that evaluates the mutual dependence between two random variables. Let (X, Y) be a pair of random variables with values over the space $\mathcal{X} \times \mathcal{Y}$, the mutual information $I(X;Y)$ quantifies the "amount of information" obtained about the random variable X by observing the other random variable Y. So, by knowing the value of one of the random variable in a system, there is a corresponding reduction in uncertainty for predicting the other one, and mutual information measures that reduction in uncertainty.

More formally, the mutual information of the two jointly discrete random variables X and Y is computed as:

$$I(X;Y) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x,y) \log \frac{p(x,y)}{p(x)\,p(y)} \tag{3.2}$$

where $p_{(X,Y)}$ is the joint probability mass function of X and Y, and p_X and p_Y are the marginal probability mass functions of X and Y respectively.

From an information theory point of view, $I(X_1; X_2) = 0$ is equivalent to assumes that the observations of X_1 will not reveals anything about X_2 and consequently that they are independent.

For a photographic sensor, it can be assumed that its photo-diodes (photo-sites) are independent, and in order to check the independency hypothesis we used two ways:

1. We compute the mutual information between neighboring photo-site
2. We also display the scatter plot of neighboring photo-sites, if the scatter plot is isotropic, it is also a good clue that there are no dependency between the pairs.

The plots have been achieved by extracting the R, G, B photo-sites from a homogeneous RAW area of size 100×100 (resulting of de-focused shot of a white wall under diffuse illumination) and by plotting them in different scatter plots. The process is illustrated at Figures 3.1.

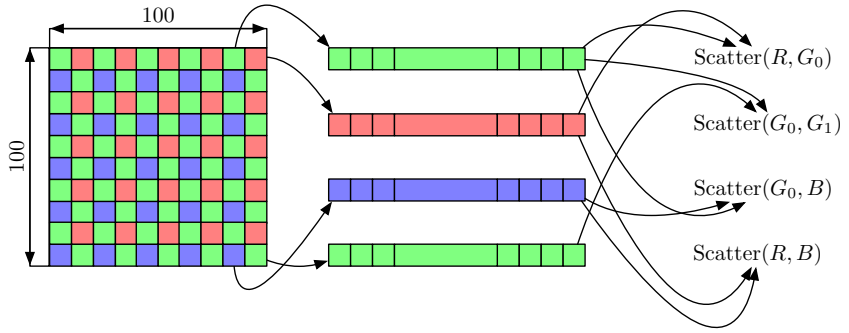


Figure 3.1: Illustration of the building process of the scatter plots.

The Figuress 3.2 3.3 3.5 thus represent realizations of the red channel on the x-axis and the green channel on the y-axis in the domain for a given sensor (Sony A7III, Z-CAM E1 and Sigma DP3). Table 3.1 presents the mutual information between neighboring photo-sites for each sensors.

Before a detailed analysis of the results, note that the Foveon X3 sensor used in the Sigma DP3 camera is using three stacked photo-diodes array whose each has a different spectral sensitivity, allowing it to respond differently to different wavelengths (see Figures 3.4). However, because the collection depth of the deepest sensor layer is comparable to collection depths in other silicon CMOS and CCD sensors, thus some diffusion of electrons and loss of sharpness in the longer wavelengths occurs [66]. On one hand, because demosaicking is not required for the Foveon X3 sensor can produce full-color images, therefore, the color artifacts associated with this process does not occurs. On the other hand, the method of color

MI \ Sensor	Sony A7III	Z-Cam E1	Sigma DP3
$I(R, G_0)$	0.069	0.031	0.361
$I(G_0, G_1)$	0.071	0.027	0.233
$I(G_0, B)$	0.076	0.03	0.237
$I(R, B)$	0.068	0.033	0.134

Table 3.1: Mutual informations for the different sensors between the different photo-sites.

separation by silicon penetration depth seems to gives more cross-contamination between color layers (as depicted at Figures 3.5), this might produces more issues with color accuracy.

One can draw relatively non-trivial conclusions from these observations:

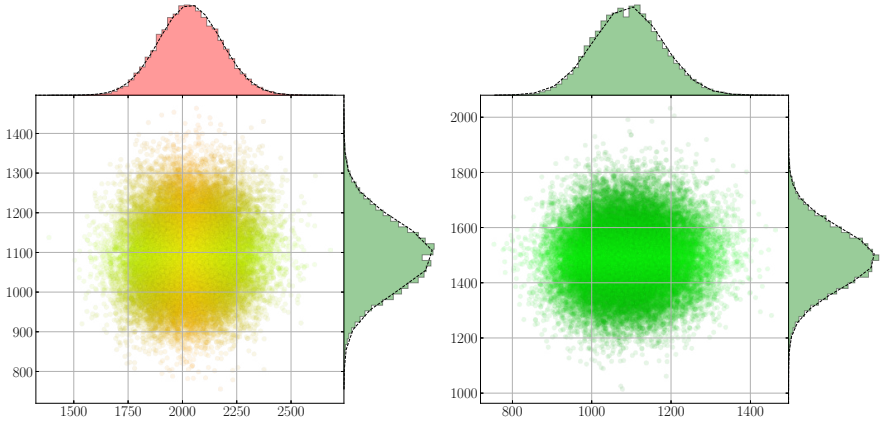
- CMOS sensors Sony A7III, Z-CAM E1 show independecies properties since the global shape of the scatter plot is anisotropic and the mutual information is very small.
- the Foveon X3 sensor, whose main design argument is based on the non-necessity of color interpolation, has a strong anisotropy of the joint distribution between the different color channels. Their independence would translate into a point cloud whose global shape would be anisotropic, besides, one can clearly observe a direction for each cloud displayed and note that in each case the mutual information is not null (even if they are weak). This statement is confirmed by the computation of the mutual information in 3.1.

To briefly conclude this sub-section, sensor noise is independent between neighboring photo-sites for a CMOS type sensor but this does not apply for the Foveon. In the light of these results we will use the independence between the photo-sites as a working hypothesis, but only for classical CDD or CMOS sensors.

1.3 Gaussianity of sensor noise distribution

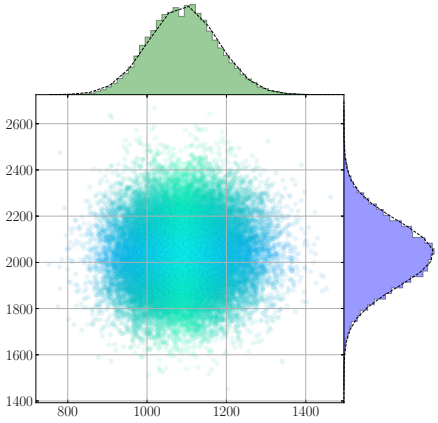
In order to generate the steganographic signal it is essential to find an adequate model. Ihe literature reporting the use of Gaussian distributions can be found in [79] steganography and in [28] [101] [25] [26] steganalysis (for steganalysis of LSB replacement and LSB matching). The advantages are threefold: Gaussian distributions are well known and many theorems are associated with them, the use of linear operations can thus be handled using multivariate Gaussians and in this case Gaussian modeling is particularly adequate to capture the signal properties.

It has been shown in [8] that the photo-sites values in a digital image acquired with an imaging sensor are typically corrupted by an independent Gaussian noise

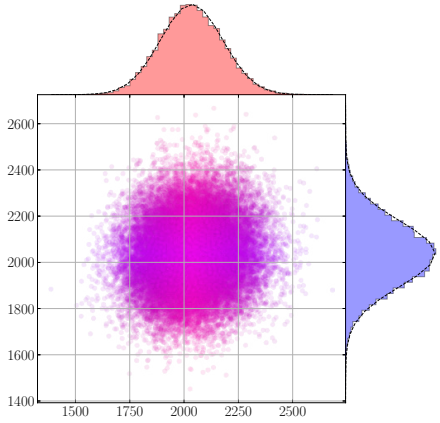


(a) R vs G

(b) G_0 vs G_1

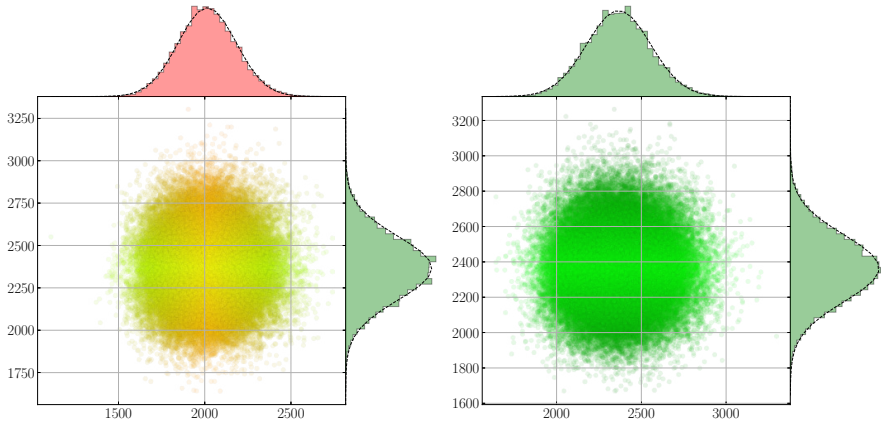


(c) G_0 vs B

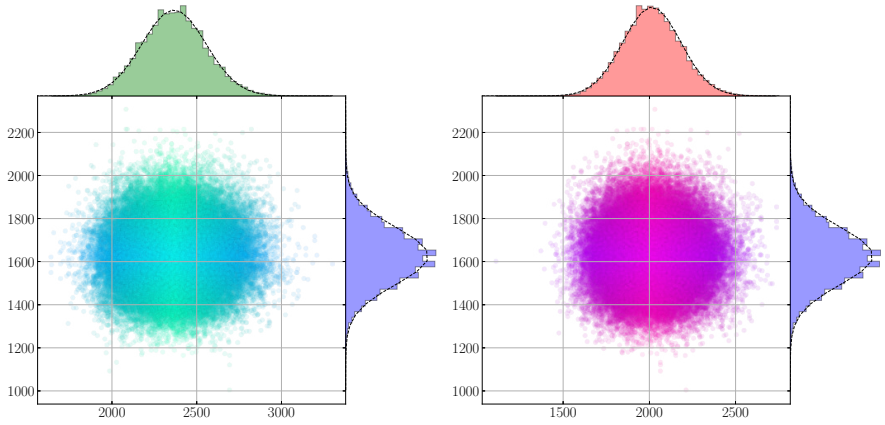


(d) R vs B

Figure 3.2: Scatter plots for SONY A7III to illustrate cross-channel correlation.



(a) R vs G (b) G_0 vs G_1



(c) G_0 vs B (d) R vs B

Figure 3.3: Scatter plots for Z-CAM E1 to illustrate cross-channel correlation:.

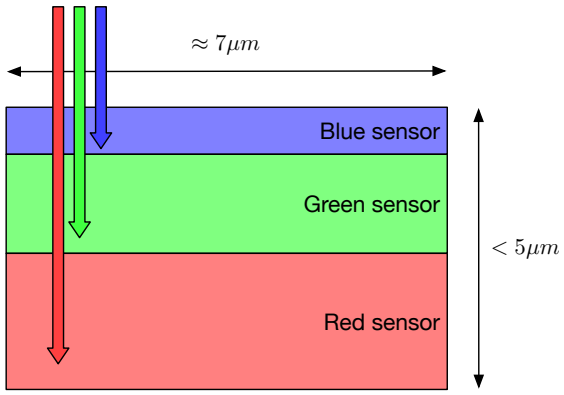
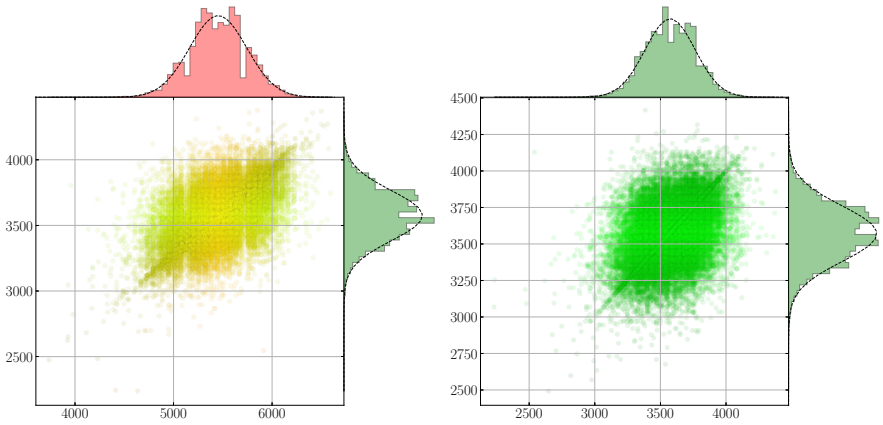
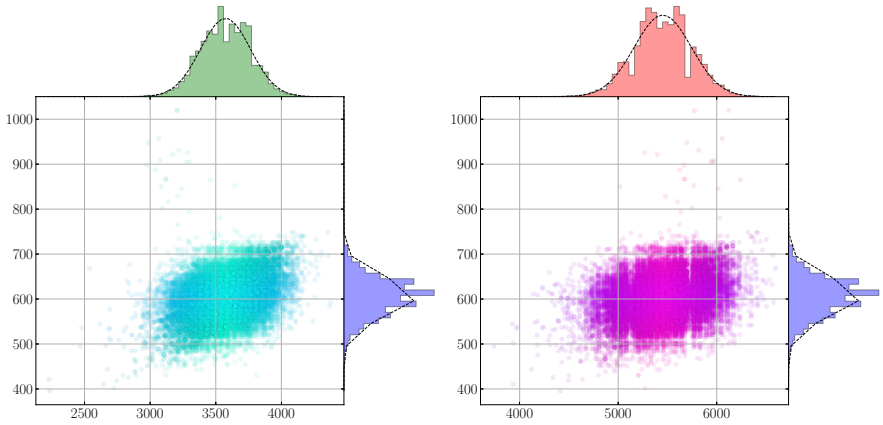


Figure 3.4: Scheme of the Foveon X3 sensor stack for one photo-sites.



(a) R vs G (b) G_0 vs G_1



(c) G_0 vs B (d) R vs B

Figure 3.5: Scatter plots for Sigma DP3 to illustrate cross-channel correlation.

with variance dependent on the pixel light intensity, the shot noise (Eq 3.3) (and by temperature and exposure (dark-current), and readout and electronic noise as well but this noises sources will be considered as negligible here).

Shot noise is the result of the discrete quantum nature of light, it has been studied in numerous academic publications such as [40], [39], [4] and have already been used in image forensics for camera device identification [77], [92] and denoising [38]. This noise is due to the fact that the number of electrons released by each photo-sites is related to the number of incidental photons (see also section 1.4 of this memoir).

The number of photons ξ captured by each photo-site during a Δt exposure is a random variable that follow a Poisson distribution, this is unveiled using the law of rare events:

$$\mathbb{P}\left[\xi = k\right] = \frac{e^{\lambda \Delta t}(\lambda \Delta t)}{k!} = p[k] \tag{3.3}$$

With mean and variance:

$$\mathbb{E}[\xi] = \sum_{k \geq 0} p[k]k - (\lambda \Delta t)^2 = \lambda \Delta t, \tag{3.4}$$

$$\mathrm{Var}[\xi] = \sum_{k \geq 0} p[k]k^2 = \lambda \Delta t. \tag{3.5}$$

Where $\lambda > 0$ is the number of photons expected to be captured during an unit time interval. Thus, by increasing the number of photons the relative variations of ξ is decreasing. Therefore, reducing the shot noise is possible by increasing the size of the photo-sites and the exposure time. This Poisson distribution can be approximated with a Gaussian distribution such that for large $\lambda \Delta t$ its mean and variance are respectively $\lambda \Delta t$ (3.4) and $\lambda \Delta t$ (3.5).

This model can only be applied to linear sensors such as CDD or CMOS sensors, but the majority of modern digital cameras are built according to one of this two process.

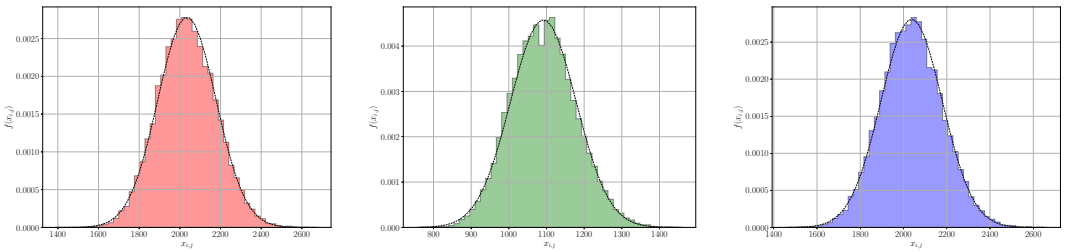
The presence of these random components during image acquisition has some critical consequences for steganography since it can be used to derived an embedding process leading to "perfectly secure steganography" (see Chapter 6.2 in [41] for more details) by setting up a steganographic system that preserves perfectly the distribution of covers and thus makes the stego indistinguishable from their cover version. Indeed, the class of Natural Steganography (NS) [9], [7], [31], [85], developed in the next chapter, generates the stego signal by mimicking the sensor noise associated with a larger ISO sensitivity.

1.3.1 Assessing Gaussianity using histogram inspection

We thus conducted experiments to analyze the distribution of the photonic noise associated to different photographic sensors.

To estimate the distributions of the sensor noise, we used here a simple technique: we shot a white wall under a diffuse lighting condition at a distance of 1 meter from the sensor and out of focus in order to obtain an image with average constant illumination. We then computed the histogram from the RAW image, using the photo-site values of a 200×200 patch centered on the image to avoid vignetting for one given color channel (we checked that our observations were consistent for all channels and for different patches sizes).

The Figuress 3.6 3.7 3.8 3.9 show some comparisons between the distributions of shot noise coming from four different sensors (Sony A7III, Z-CAM E1, SIGMA DP3 and Leica M9) for each colored photo-sites and for one monochromatic sensor. The histograms are computed from the photo-site values of one given channel (for color sensors) on a uniform patch. Dashed lines represent Gaussian distributions with the same mean and variance as the histogram. We have empirically confirmed that for most of these sensors the noise was distributed according to a Gaussian except for the M9. As a matter of fact, we noticed that the shot noise on the M9 sensor does not have a Gaussian distribution at least as at high ISO sensitivities. We will consequently not use this device to conduct experiments on Natural Steganography.

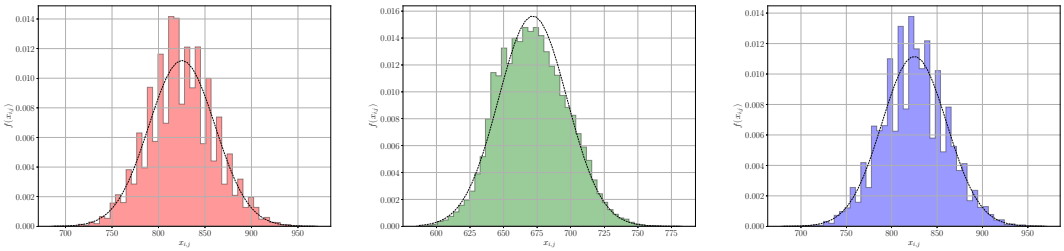


(a) R (b) G (c) B

Figure 3.6: Histogram of photo-sites values for each CFA filters for Sony A7III capturing a homogeneous patch.

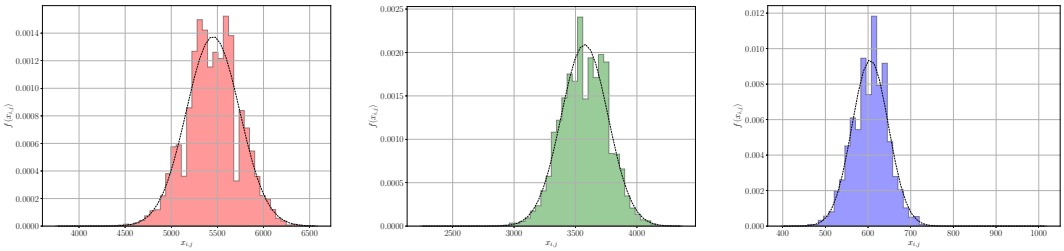
1.4 Gaussianity of sensor noise distribution in the DCT domain

As briefly discussed in chapter two of this manuscript, there are several steps required to convert a RAW image to JPEG format. In a simplified case, it can be considered that in order to convert a raw sensor image into the JPEG, it is



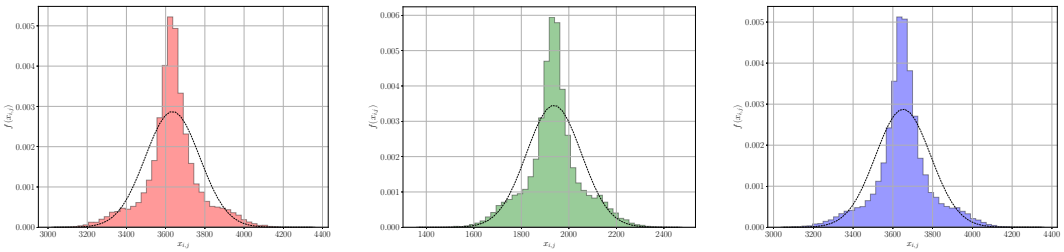
(a) R (b) G (c) B

Figure 3.7: Histogram of photo-sites values for each CFA filters for Z-CAM E1 capturing a homogeneous patch.



(a) R (b) G (c) B

Figure 3.8: Histogram of photo-sites values for each CFA filters for Sigma DP3 capturing a homogeneous patch.



(a) R (b) G (c) B

Figure 3.9: Histogram of photo-sites values for Leica M9 capturing a homogeneous patch.

only the target of a demosaicking, a luminance transform, a transform in the DCT domain and finally a quantization (as depicted in Figures 3.10).

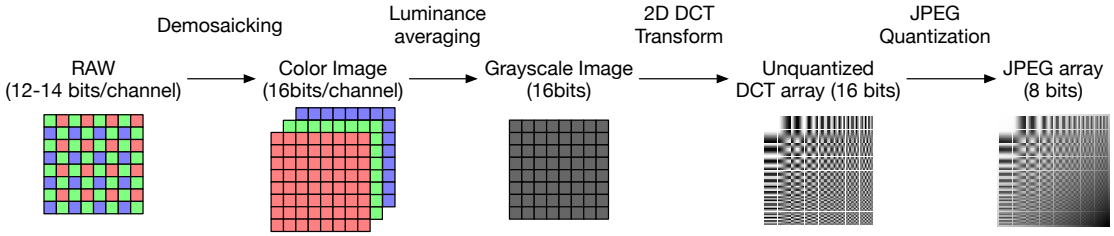


Figure 3.10: The image developing pipeline.

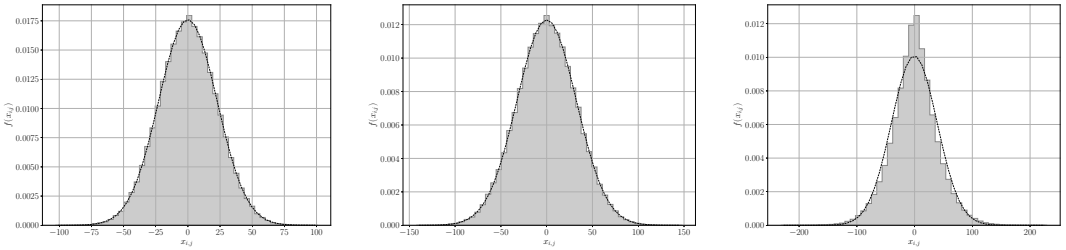
The first step of this pipeline consist in interpolating for each photo-site the two missing color components via the demosaicking process (see also section 1.2). Popular demosaicking schemes include: bi-linear filtering which use linear interpolation from the neighbors of the same channel, Variable Number of Gradients (VNG) [16], Patterned Pixel Grouping (PPG) [70] and Adaptive Homogeneity-Directed (AHD) [48] are more advanced schemes using the correlation between the channels and also edge-directionality and color homogeneity to predict the photo-site values. Note that the implementations of these algorithms are available in dcraw or the library libraw [1] or are available using some commercial softwares such as Lightroom or DXO. One notable point of the demosaicking procedure is that regardless of the algorithm the recorded photo-sites values stay unchanged. This process can be linear or non-linear while the operations that follow demosaicking up to quantification according to the quality factor are all linear operations. Since the luminance and DCT transform operations are linear, the demosaicking is the gateway to non-linearities for the distribution of unquantified DCT coefficients. Thus, if we assume that the demosaicking operation is linear, and since the stego signal at the photo-site level is Gaussian, the stego-signal at the DCT level is distributed as a Multivariate Gaussian Distribution (MGD) (see [85]).

From a constant RAW image corrupted with a stationary noise and processed (using demosaiking, luminance averaging and 2D-DCT transform) in the undecimated DCT domain, the histograms in Figures 3.11 shows the DCT coefficient 56 after demosaicking using the BIL, VNG, AAHD algorithms.

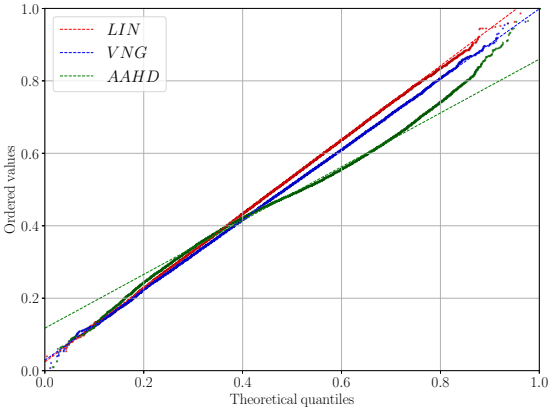
The Figures 3.12 is also assessing Gaussianity using a **QQ(quantile-quantile)-plot**. This has been achieved by plotting a points cloud from sorted samples from DCT coefficient 56 vs. sorted samples from a standard normal distribution. If the distributions are linearly related, the points cloud in the Normality plot will approximately lie on a line, but not necessarily on the line $y = x$, and deviations from a straight line suggest departures from normality.

With both Figures 3.11 and Figures 3.12, one can understand that in the case

of the BIL and VNG algorithm the point cloud distribution visually validates that the distributions are "visually" Gaussian, this is not the case in the case of AAHD.



(a) Bi-linear (b) VNG (c) AAHD



(d) QQ-plots

Figure 3.11: Histograms of the values of the DCT mode $(7, 0)$ after demosaicking and DCT transform: (a) Bi-Linear, (b) VNG, (c) AAHD.

It can be concluded here that non-linear operations can suppress the Gaussian character of the sensor noise depending on the demosaicking used. Indeed the bi-linear demosaicking is the only one that mathematically preserves the gaussianity of the sensor noise, whereas VNG visually provide the same result only visually. Finally it is clear that AAHD clearly does not preserve the gaussianity of the sensor noise. Therefore we will use in the rest of the manuscript the bi-linear development.

2 Analytical model of the photonic noise using multivariate Gaussian

We now present a way to model the photonic noise in the DCT domain after a linear development pipeline. This model relies on the working assumptions highlighted in

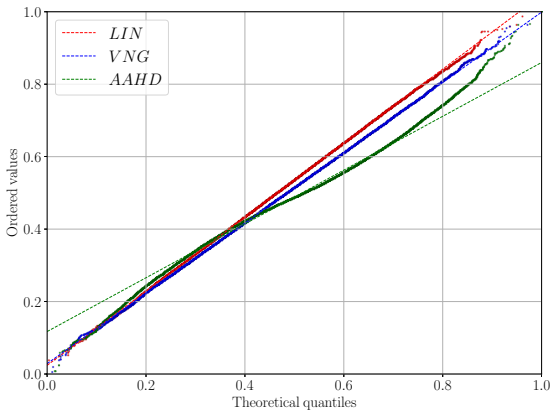


Figure 3.12: QQ-plots of samples of DCT coefficient 56 using after demosaicking using Bi-Linear, VNG, AAHD vs. samples drawn from a Normal distribution

the previous section and it will be used in the next chapter to perform embedding.

Note first that the images we study are acquired with a color sensor, developed with linear demosaicking, converted to gray-scale, and JPEG compressed. The difference between monochrome and color sensors was studied in [31] with the conclusion that independent modification performed on each DCT coefficient offers high empirical security for monochrome sensors, but not for color sensors. This major difference is due to the fact that demosaicking introduces dependencies among neighboring DCT coefficients. When these dependencies are not taken into account, the modification performed on the DCT coefficient becomes highly detectable at high JPEG Quality Factors (QF). To overcome this problem, it is possible to model theses dependencies (for a linear development pipeline) using a multivariate Gaussian model by computing the covariance matrix of the stego signal in the DCT domain as depicted on Figures 3.13.

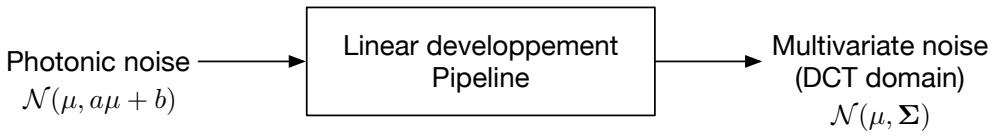


Figure 3.13: Impact of the development process on the distribution of heteroscedactic photonic noise towards a multivariate Gaussian distribution.

In the case of a bi-linear demosaicking it is possible to analytically compute the covariance matrix in order to obtain the dependencies between the DCT modes, within the same block (intra block correlations) and between the DCT modes belonging to the neighboring blocs (inter block correlations).

2.1 Analytical formulation of the covariance matrix

For a CCD or CMOS sensor, the photonic noise N at photo-site i, j due to the error of photonic counts during acquisition is assumed to be independent across photo-sites with a widely adopted heteroscedastic model [4]:

For a CCD or CMOS sensor, the photonic noise N at photo-site i, j due to the error of photonics count during acquisition is assumed to be independent across photo-sites with a widely adopted heteroscedastic model [4]:

$$N_{i,j}^{(1)} \sim \mathcal{N}\left(0, a_1 \mu_{i,j} + b_1\right), \tag{3.6}$$

where $\mu_{i,j}$ is the noiseless photo-site value at photo-site i, j, and (a_1, b_1) a pair of parameters depending only on the ISO_1 sensitivity and the specific sensor. The acquired photo-site sample $x_{i,j}^{(1)}$ is thus a realization $x_{i,j}^{(1)} = \mu_{i,j} + n_{i,j}^{(1)}$ of a Gaussian variable distributed as

$$X_{i,j}^{(1)} \sim \mathcal{N}\left(\mu_{i,j}, a_1 \mu_{i,j} + b_1\right). \tag{3.7}$$

In this section, we use a linear development pipeline for the processing of RAW images: each RAW image will be demosaicked, and converted to a gray-scale spatial array after luminance averaging, finally after a 2D-DCT transform the image is quantized to the selected quality factor to be saved as a JPEG file. The overall purpose of the approach described in this section is to derive the dependencies that may exist between the DCT coefficients from the image development process. These dependencies will then be exploited in Chapter 4 in order to perform embeddings preserving their own existence and thus allowing the generated stego images to correspond to a given model, mimicking the natural photonic noise distribution of a cover image captured at ISO_2 (in the DCT domain).

The work presented in [10, 6] shows that for monochrome sensors, this model in the spatial domain can be used to derive the distribution of the stego signal in the spatial domain after quantization, gamma correction, and image down-sampling using bilinear kernels.

2.1.1 The development pipeline

Since the distribution of the shot noise at the photo-site level is Gaussian and independant (with diagonal covariance matrix), and because the pipeline up to the DCT transform is a succession of linear operations, one main result of statistical signal processing [12] it that its distribution in the DCT domain is also a multivariate Gaussian distribution, but with arbitrary covariance matrix. The linear development allows us also to derive the covariance matrix of this distribution. We can write that $\mathbf{y}_p = \mathbf{x}_p + \mathbf{s}_p$, where $\mathbf{x}_p$ is the vectorized version of a block of photo-site values of the cover image, and $\mathbf{s}_p$ the vectorized values of the added stego signal in the

photo-site domain, which has also a Gaussian, independent distribution (see next chapter).

The goal of this section is to model the development pipeline as a linear equation in the form of:

$$\mathbf{y}_d = \mathbf{m}\mathbf{y}_p \Leftrightarrow \mathbf{s}_d = \mathbf{m}\mathbf{s}_p, \tag{3.8}$$

where $\mathbf{y}_d$ and $\mathbf{s}_d$ represent the vectors of respectively the stego content and the stego signal in the developed domain.

Since the only random component is the stego signal $\mathbf{s}_p$, the covariance matrix $\mathbf{\Sigma}_d = \mathbf{cov}(\mathbf{s}_d)$ of the multivariate distribution in the DCT domain will then be given by:

$$\mathbf{\Sigma}_d \ = \mathbf{M}\,\mathbf{\Sigma}_p\,\mathbf{M}^t, \tag{3.9}$$

where $\mathbf{\Sigma}_p = \mathrm{Cov}(\mathbf{s}_p)$ is the covariance matrix of the considered block of the stego signal in the photo-site domain given the cover $\mathbf{x}$.

Denoting now i the index of one photo-site in $\mathbf{x}_p$, the covariance matrix $\mathbf{\Sigma}_p$ of the stego-signal is a diagonal matrix with diagonal terms equal to $(a_2 - a_1)\,x_i + (b_2 - b_1)$ (where (a_1, b_1) and (a_2, b_2) are constants coming from the embedding operation presented in the next chapter).

In order to compute $\mathbf{M}$, we consider the different steps of the pipeline and decompose the computation of $\mathbf{M}$ into the following steps (see Figures 3.14):

1. Demosaicking: this step predicts for each photo-site the two missing colors that are not recorded by the sensor. We use bilinear filtering as a linear interpolation process.
2. Luminance averaging: (we only consider embedding in gray-scale JPEG image) the demosaicked vector undergoes luminance averaging following the ITU-R BT 601 standard.
3. 2D-DCT transform is computed independently on each block of 8×8 pixels.
4. Quantization: the DCT coefficients are quantized using the quantization table matching a selected JPEG quality factor (QF) to generate a set of JPEG coefficients. Note that since this operation is non-linear, it is not captured by equation (3.8).

We now detail the different linear operations which are detailed on content vectors $\mathbf{y}_f$ (the subscript f denoting the operation), but can also be written w.r.t. $\mathbf{s}_f$ thanks to the linear formulation by switching $\mathbf{y}_f$ by $\mathbf{s}_f$.

2.1.2 Considered photo-sites

Since the color interpolation step uses the neighboring photo-sites to interpolate colors, this creates correlations between adjacent 8-connected blocks of 8×8 photo-sites. These correlations between blocks can be very weak, especially between diagonal blocks. On the contrary, it is important to note that two blocks which

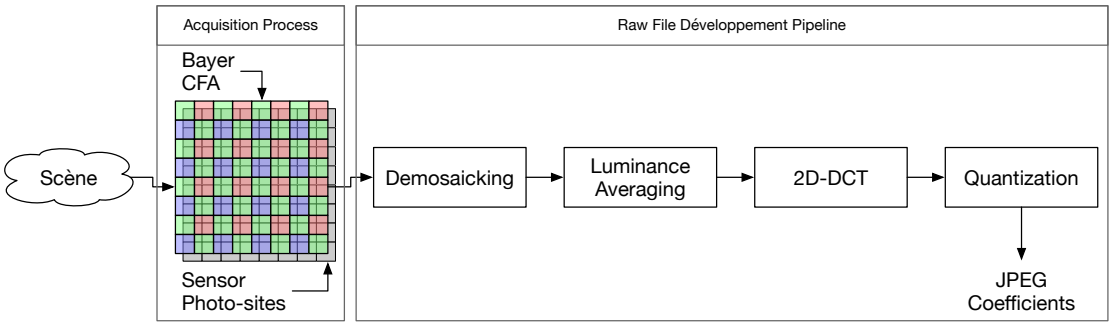


Figure 3.14: Development pipeline: From a scene captured by a color sensor to luminance JPEG coefficients.

are not 8-connected represent independent realizations of the sensor-noise after demosaicking. This property will be used in Section (2) to design the embedding scheme. Both correlation between adjacent blocks and uncorrelated blocks are illustrated in Figures 3.15. On this Figures we can see that two diagonal blocks can share only two correlated photo-sites, and the correlations can either come from three photo-site values coming from vertical, horizontal, and diagonal blocks (this is the case between NE and SW neighbors), or two photo-site values coming from horizontal and vertical blocks only (this is the case for NW or SE neighbors). On the contrary two blocks that are disconnected are associated to uncorrelated stego signals. In order to capture all the correlations between DCT coefficients, we consequently need to consider a matrix $\mathbf{Y}_p$ of $(3 \times 8 + 2) \times (3 \times 8 + 2)$ photo-sites, which gives after vectorization $\mathrm{vec}_{\mathrm{R}}(\mathbf{Y}_p)$ a vector $\mathbf{y}_p$ of 676 photo-sites as an input of our linear system as illustrated in Figures 3.16.

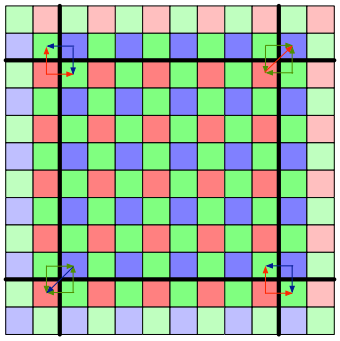


Figure 3.15: Locations of photo-sites (dark colors) used to interpolate pixel values within one block using bilinear demosaicking. Diagonal blocks are involved in the computation on two pixels for the blue channel (up right) and the red channel (bottom, left).

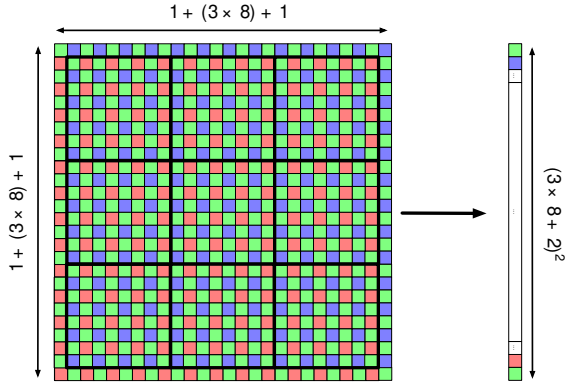


Figure 3.16: RAW photo-sites and its outer border.

2.1.3 Demosaicking

It is possible to write the demosaicking operations as matrix multiplications. For each component R, G and B, we define the matrices $\mathbf{D}_r$, $\mathbf{D}_g$, $\mathbf{D}_b$ of size $(24+2)^2 \times (24+2)^2$, such that the result of the matrix multiplication of $\mathbf{y}_p$ with one of these matrices is the vectorized version of the corresponding color channel after demosaicking:

$$\mathbf{y}_r = \mathbf{D}_r \mathbf{y}_p,\ \mathbf{y}_g = \mathbf{D}_g \mathbf{y}_p,\ \mathbf{y}_b = \mathbf{D}_b \mathbf{y}_p. \tag{3.10}$$

Denoting i the index of one photo-site in $\mathbf{y}_i$, one row i of $\mathbf{D}_k$, $k \in \{r, g, b\}$ is obtained by vectorization of a $(24+2) \times (24+2)$ matrix with zeros entries except a specific kernel $\mathbf{K}_i$ centered on (i, i). This kernel models any kind of interpolation between neighboring photo-sites and y_i:

$$\mathrm{row}_i(\mathbf{D}_k) = \mathrm{vec}_R \begin{bmatrix} \mathbf{0} & \cdots & & & & \\ \vdots & \ddots & & & & \\ & & \ddots & & & \\ & & & \mathbf{K}_i & \mathbf{0} & \cdots \\ & & & \mathbf{0} & \mathbf{0} & \\ & & & \vdots & & \ddots \end{bmatrix}. \tag{3.11}$$

Without loss of generality we now focus on the computation of $\mathbf{D}_g$, we consequently have two possibilities in this case:

- If index i corresponds to a green photo-site on the Bayer CFA, this photo-site does not need color interpolation, i.e.:

$$\mathbf{K}_i = [\mathbf{1}]. \tag{3.12}$$

- If index i corresponds to a pixel which needs to be interpolated, then:

$$\mathbf{K}_i = \begin{bmatrix} 0.25 & 0 & 0.25 \\ 0 & \mathbf{0} & 0 \\ 0.25 & 0 & 0.25 \end{bmatrix}. \tag{3.13}$$

The kernel coefficient in bold representing the location (i, i). For the red and blue channels, we use four different convolution kernels $\mathbf{K}_i$ to build $\mathbf{D}_r$ and $\mathbf{D}_b$, which are:

$$[\mathbf{1}], \begin{bmatrix} 0.5 \\ \mathbf{0} \\ 0.5 \end{bmatrix}, \begin{bmatrix} 0.5 & \mathbf{0} & 0.5 \end{bmatrix} \text{and} \begin{bmatrix} 0.25 & 0 & 0.25 \\ 0 & \mathbf{0} & 0 \\ 0.25 & 0 & 0.25 \end{bmatrix}, \tag{3.14}$$

and are respectively used for duplication, interpolation between vertical or horizontal photo-sites and interpolation between four diagonal photo-sites.

2.1.4 Luminance averaging

To perform luminance averaging, we can define the matrix $\mathbf{L}$ following the ITU-R BT 601 standard as:

$$\mathbf{y}_l = \underbrace{(0.2126 \cdot \mathbf{D}_r + 0.7152 \cdot \mathbf{D}_g + 0.0722 \cdot \mathbf{D}_b)}_{\mathbf{L}} \mathbf{y}_p, \tag{3.15}$$

with $\mathbf{y}_l \in \mathbf{R}^{(24+2)^2 \times 1}$.

2.1.5 Pixel selection

As stated above, the surrounding edges of 3×3 blocks of samples are included in order to take into account the convolution window during demosaicking. Once the demosaicking operations have been carried out, the photo-sites not present in the DCT blocks can then be discarded. Let us denote $\mathbf{Y}_l$ the $(24+2) \times (24+2)$ photo-sites matrix with its outer border, and $\mathbf{Y}_s$ without it as depicted in Figures 3.17. The selection matrix $\mathbf{S} \in \mathbb{R}^{(24)^2 \times (24+2)^2}$ can then be defined such that:

$$\begin{aligned} \mathbf{y}_s = \text{vec}_{\text{R}}(\mathbf{Y}_s) = \mathbf{S} \\ \text{vec}_{\text{R}}(\mathbf{Y}_l) = \mathbf{S}\mathbf{y}_l. \end{aligned} \tag{3.16}$$

and we also can write: $\mathbf{y}_s = \mathbf{S}\,\mathbf{L}\mathbf{y}_p$.

2.1.6 Blocks permutation and block selection

Blocks permutation and block selection are not mandatory, but they are useful to compute conditional probabilities while limiting the computational load (see chapter 2). Depending on the lattice considered during the embedding (see again chapter 2), the correlation matrix can be computed for DCT coefficients belonging to one, five or nine adjacent blocks.

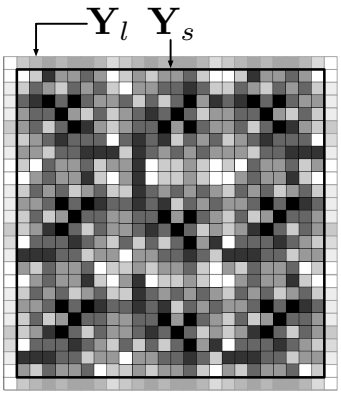


Figure 3.17: Block representation of the pixel selection operation.

We recall that DCT is performed independently on each of these blocks, it is thus simpler to perform a DCT transform if the coefficients to be processed are clustered.

In order to mathematically express a block permutation and selection as the matrix multiplication such that:

$$\mathbf{y}_{pe} = \mathbf{P}\,\mathbf{y}_s, \tag{3.17}$$

we define $\mathbf{Y}_s \in \mathbb{R}^{24\times 24}$ as an array composed of the 3×3 blocks of 8×8 pixels each, such that the vector $\mathbf{y}_s = \mathrm{vec_R}(\mathbf{Y}_s)$, with:

$$\mathbf{Y}_s = \begin{bmatrix} \mathbf{B}_{0,0} & \mathbf{B}_{0,1} & \mathbf{B}_{0,2} \\ \mathbf{B}_{1,0} & \mathbf{B}_{1,1} & \mathbf{B}_{1,2} \\ \mathbf{B}_{2,0} & \mathbf{B}_{2,1} & \mathbf{B}_{2,2} \end{bmatrix}. \tag{3.18}$$

When vectorizing according to the lines, the coefficients of the 3×3 blocks are stored as[1] :

$$\begin{aligned} \mathbf{y}_s = [\quad & \mathbf{B}_{0,0}[0,:], \quad \ldots, \quad \mathbf{B}_{0,2}[0,:], \\ & \mathbf{B}_{0,0}[1,:], \quad \ldots, \quad \mathbf{B}_{0,2}[1,:], \\ & \quad\vdots \qquad\qquad\quad \vdots \\ & \mathbf{B}_{0,0}[7,:], \quad \cdots, \quad \mathbf{B}_{0,2}[7,:], \\ & \quad\vdots \qquad\qquad\quad \vdots \\ & \mathbf{B}_{2,0}[7,:], \quad \cdots, \quad \mathbf{B}_{2,2}[7,:] \quad]. \end{aligned} \tag{3.19}$$

It would be simpler for to group these elements within a new vector according to their original block, such as:

$$\mathbf{y}_{pe} = \left[\ \mathrm{vec_R}(\mathbf{B}_{0,0}), \quad \ldots, \quad \mathrm{vec_R}(\mathbf{B}_{0,2}), \quad \ldots \quad \mathrm{vec_R}(\mathbf{B}_{2,2})\ \right], \tag{3.20}$$

where $\mathbf{B}_{i,j} \in \mathbb{R}^{8\times 8}$ are blocks of 8×8 pixels, $0 \leq i,\, j \leq 2$.

[1]The pythonic notation $\mathbf{A}[0,:]$ means that for a 2D array $\mathbf{A}$ of arbitrary size $\mathbf{A}[0,:]$ is the array of all elements of line 0.

Therefore, the first step here is to perform a permutation of the coefficients of the vector $\mathbf{y}_s$ such that the coefficients of each block would be grouped by belonging to their respective block. The result of this operation will be written $\mathbf{y}_{pe}$ and can be obtained by a matrix multiplication :

$$\mathbf{y}_\pi = \mathbf{P}_\pi \mathbf{y}_s. \tag{3.21}$$

It is possible to obtain an expression of $\mathbf{P}_\pi$ by the following method:

1. Starting from a matrix $\mathbf{A} \in \mathbb{R}^{24\times 24}$ whose coefficients reflect the order in which they are stored during the vectorization operation $\mathbf{a} = \mathrm{vec_R}(\mathbf{A})$.
   As a result, we obtain :
   $$\mathbf{A} = \begin{bmatrix} 0 & 1 & \dots & 23 \\ 24 & 25 & \dots & 2\cdot 24 - 1 \\ \vdots & & & \vdots \\ (3\cdot 8)(2\cdot 8) & & \dots & (3\cdot 8)^2 - 1 \end{bmatrix} = (a_{i,j})_{0\le i,j<8} = (8\cdot i + j)_{0\le i,j<8}\,, \tag{3.22}$$
   and thus :
   $$\mathbf{a} = \mathrm{vec_R}(\mathbf{A}) = (i)_{0\le i<(3\cdot 8)^2}\,. \tag{3.23}$$

2. Then from a block matrix $\tilde{\mathbf{A}} \in \mathbb{R}^{24\times 24}$ whose elements indicate the order in which its coefficients must be stored in a vector so that they can be grouped together according to the block they come from. $\tilde{\mathbf{A}} \in \mathbb{R}^{24\times 24}$ is thus defined as :
   $$\tilde{\mathbf{A}} = \left[\begin{array}{c|c|c} (a_{i,j})_{0\le i,j<8} & \cdots & (2\cdot 64 + a_{i,j})_{0\le i,j<8} \\ \hline \cdots & \cdots & \cdots \\ \hline \cdots & \cdots & (7\cdot 64 + a_{i,j})_{0\le i,j<8} \end{array}\right], \tag{3.24}$$
   and thus :
   $$\tilde{\mathbf{a}} = \mathrm{vec_R}(\tilde{\mathbf{A}}). \tag{3.25}$$

3. Now the $\mathbf{P}_\pi$ matrix can be defined as the permutation matrix to switch from vectorization by lines to vectorization by block-wise lines:
   $$\mathbf{P}_\pi = (\delta_{a(i),\,\tilde{a}(j)})_{\substack{0\le i<(3\cdot 8)^2 \\ 0\le j<(3\cdot 8)^2}}. \tag{3.26}$$

Hence, it becomes easy to define a matrix $\mathbf{P}_s$ to reorder and select the elements from the desired blocks.

We illustrate this with two examples.

Example 1: Suppose we need to extract from $\mathbf{Y}_s$ the vectorized form of the central block $\mathbf{B}_{1,1}$, i.e., $i = 1$ and $j = 1$. We then have :

$$\mathbf{y}_{pe} = \mathbf{P}_1 \mathbf{y}_\pi = \left(\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix} \otimes \mathbf{I}_{64} \right) \begin{bmatrix} \mathrm{vec_R}\left(\mathbf{B}_{0,0}\right) \\ \vdots \\ \mathrm{vec_R}\left(\mathbf{B}_{1,1}\right) \\ \vdots \\ \mathrm{vec_R}\left(\mathbf{B}_{2,2}\right) \end{bmatrix} \tag{3.27}$$

The corresponding vector permutation matrix is then $\mathbf{P} = \mathbf{P}_1 \mathbf{P}_\pi$ and $\mathbf{y}_{pe} = \left(\mathbf{P}_1 \mathbf{P}_\pi\right) \mathbf{y}_s$.

Example 2: This additional example is useful for the understanding of the manuscript (see Section 2.1). Let us extract from $\mathbf{y}_s$ the vector resulting from the concatenation of the vectorized version of five 8×8 blocks of pixels in a given order,

$$\mathbf{y}_{pe} = \left[\mathrm{vec_R}(\mathbf{B}_{1,1}), \mathrm{vec_R}(\mathbf{B}_{0,0}), \mathrm{vec_R}(\mathbf{B}_{0,2}), \mathrm{vec_R}(\mathbf{B}_{2,0}), \mathrm{vec_R}(\mathbf{B}_{2,2})\right]^t$$

The corresponding matrix operation will be:

$$\mathbf{y}_{pe} = \mathbf{P}_5 \mathbf{y}_\pi = \left(\begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \otimes \mathbf{I}_{64} \right) \begin{bmatrix} \mathrm{vec_R}\left(\mathbf{B}_{0,0}\right) \\ \vdots \\ \mathrm{vec_R}\left(\mathbf{B}_{1,1}\right) \\ \vdots \\ \mathrm{vec_R}\left(\mathbf{B}_{2,2}\right) \end{bmatrix} \tag{3.28}$$

The corresponding vector permutation matrix is then $\mathbf{P} = \mathbf{P}_5 \mathbf{P}_\pi$ and $\mathbf{y}_{pe} = \left(\mathbf{P}_5 \mathbf{P}_\pi\right) \mathbf{y}_s$.

2.1.7 2D-DCT Transform

For a 8×8 block in the spatial domain, $\mathbf{B}$, its 2D-DCT block version written here as $\mathbf{B}_d$ can be expressed by the following matrix multiplication:

$$\mathrm{DCT}(\mathbf{B}) = \mathbf{A} \cdot \mathbf{B} \cdot \mathbf{A}^t = \mathbf{A} \cdot (\mathbf{A} \cdot \mathbf{B}^t)^t, \tag{3.29}$$

with:

$$\mathbf{A} = \begin{bmatrix} a & a & a & a & a & a & a & a \\ b & d & e & g & -g & -e & -d & -b \\ c & f & -f & -c & -c & -f & f & c \\ d & -g & -b & -e & e & b & g & -d \\ a & -a & -a & a & a & -a & -a & a \\ e & -b & g & d & -d & -g & b & -e \\ f & -c & c & -f & -f & c & -c & f \\ g & -e & d & -b & b & -d & e & -g \end{bmatrix}, \tag{3.30}$$

and:

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \cos(\frac{\pi}{4}) \\ \cos(\frac{\pi}{16}) \\ \cos(\frac{\pi}{8}) \\ \cos(\frac{3\pi}{16}) \\ \cos(\frac{5\pi}{16}) \\ \cos(\frac{3\pi}{8}) \\ \cos(\frac{7\pi}{16}) \end{bmatrix}. \tag{3.31}$$

It should be observed that the multiplication by $\mathbf{A}$ and $\mathbf{A}^t$ is due to the fact that the DCT transform is separable and processes the columns and rows independently. In order to compute the covariance matrix of the spatial signal $\mathbf{B}$, we use vector notation by transforming the matrix $\mathbf{B}\in\mathbb{R}^{8\times 8}$ into a vector $\mathbf{b}\in\mathbb{R}^{64}$ by concatenating the columns. As a result, the 8×8 matrix $\mathbf{A}$ is transformed into a 64×64 matrix $\mathbf{A}_v$ given by:

$$\mathbf{A}_v = \begin{bmatrix} \mathbf{A} & \mathbf{0} & \dots & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \mathbf{0} & \vdots \\ \vdots & \mathbf{0} & \ddots & \mathbf{0} \\ \mathbf{0} & \cdots & \mathbf{0} & \mathbf{A} \end{bmatrix}. \tag{3.32}$$

We also define a transpose operator $\mathbf{T}_r \in \mathbb{R}^{64\times 64}$ such as :

$$\mathrm{vec_c}(\mathbf{X_S^T}) = \mathbf{T}_r \cdot \mathrm{vec_c}(\mathbf{X_S}) = \mathbf{T}_r \cdot \mathbf{X_S}$$

, with:

$$\mathbf{T}_r = (\delta_{r(i),\, c(j)})_{\substack{0\le i<64 \\ 0\le j<64}}, \tag{3.33}$$

and,

$$\begin{aligned} r(i) &= 8\,\lfloor i/8 \rfloor + (i \bmod (8)), \\ c(j) &= 8(j \bmod (8) + \lfloor j/8 \rfloor, \end{aligned} \tag{3.34}$$

$\delta_{r(i),\, c(j)}$ being the Kronecker function applied to row $r(i)$ and column $c(j)$.

The transpose operation $\mathbf{B}^t$ is then equivalent to the multiplication $\mathbf{T}_r \cdot \mathbf{B}$, and the vector form of the DCT 8×8 block $DCT(\mathbf{B})$ finally becomes:

$$\mathrm{DCT}(\mathbf{b}) = \underbrace{\mathbf{A}_v\, \mathbf{T}_r\, \mathbf{A}_v\, \mathbf{T}_r}_{\mathbf{T}_b}\, \mathbf{b} \tag{3.35}$$

In order to compute the DCT of n blocks of size 8×8 ($n \in \{1, 5, 9\}$), we now define:

$$\mathbf{T} = \begin{pmatrix} \mathbf{T}_b & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \mathbf{T}_b \end{pmatrix}. \tag{3.36}$$

With $\mathbf{T}$ a block diagonal matrix with n matrices $\mathbf{T}_b$ on its diagonal.

2.1.8 Whole covariance matrix

The development pipeline can be then explicitly formulated as

$$\mathbf{s}_d = \mathbf{M}\mathbf{s}_p = \underbrace{\mathbf{T}\,\mathbf{P}\,\mathbf{S}\,\mathbf{L}}_{\mathbf{M}}\,\mathbf{s}_p, \tag{3.37}$$

and the covariance matrix is computed as:

$$\mathbf{\Sigma}_d \;\; = \mathbf{M}\,\mathbb{E}\left[\mathbf{s}_p\,\mathbf{s}_p^t\right]\,\mathbf{M}^t. \tag{3.38}$$

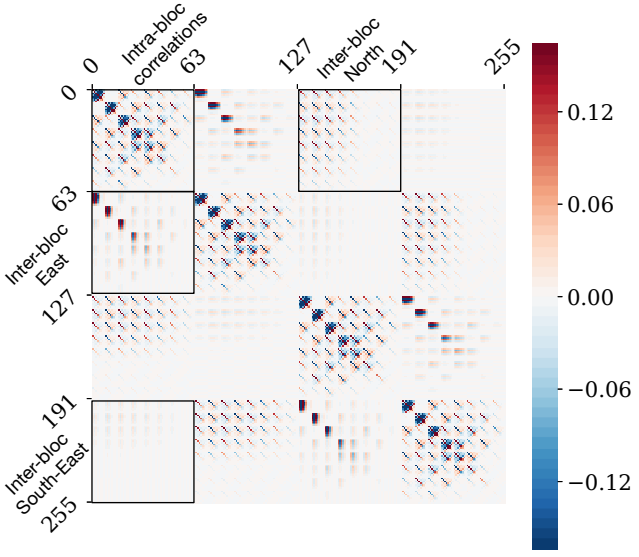
Note that for a uniform constant RAW image defined by $\mu = \text{const.}$ (i.e., $\mathbb{E}\left[\mathbf{s}_d \cdot \mathbf{s}_d^t\right] \propto \mathbf{I}$), we obtain $\mathbf{\Sigma}_d \propto \mathbf{M}\mathbf{M}^{\mathbf{T}}$. Depending of the number of blocks n considered in the neighborhood ($n \in \{1, 5, 9\}$, see 2.1), the size of $\mathbf{\Sigma}_d$ is $(n \times 64, n \times 64)$.

Note that this analysis is beneficial in order to understand the causes of the observed covariances. This understanding enables to decompose the embedding scheme into independent lattices (see section 2) but also to pave the road for other synchronization strategies applied to other development pipelines. For example, in [83], the covariance matrix is limited to the effect of averaging and can be used to synchronize DCT coefficients of classical schemes such as UERD or J-UNIWARD. In [69], relationships between DCT coefficients to preserve continuities are in line with the presented analysis of the inter-block correlations (see 3.5).

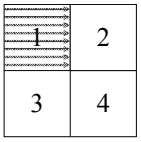
As non connected blocks are uncorrelated, we focus here on only four adjacent 8×8 blocks of unquantized DCT coefficients, as depicted in Figures (3.18b). This selection enables us to analyze correlations within a block, but also correlations between horizontal, vertical and diagonal neighboring blocks. By observing Figures (3.18a) together with the scan order depicted in Figures (3.18b), we can decompose the entire covariance matrix into four types of matrices of size 64×64 as illustrated in Figures (3.18c): It is worth noting that the stationary behavior that appears here in $\mathbf{\Sigma}_d$ is not true for real images where the input signal is not identically distributed.

3 Analysis of the covariance matrix

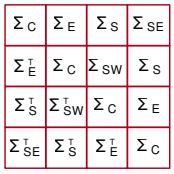
We now analyze in this section the different properties of the covariance matrix, giving explanations on the occurrences of different patterns in intra-block and inter-block covariance matrices. This analysis enables to highlight the impact of demosaicking, low pass filtering and block continuity. For the BOSSBase development pipeline, it also allows to decompose the set of DCT coefficients into 8 groups of mutually uncorrelated coefficients.



(a)



(b)



(c)

Figure 3.18: (a) 256×256 covariance matrix of DCT coefficients of a color sensor with bilinear demosaicking for an i.i.d signal (the correlation values are thresholded for visualization purposes). (b): scan order by blocks and coefficients. (c): types of sub-matrices representing the 9 covariance matrices.

3.1 Considered development pipeline

In order to leverage the correlations induced by the development pipeline, we explain in this section the development pipeline used to develop the raw images of BOSSBase. Since this database is composed of images coming from different cameras, the sensors have different sizes (from CR2 of size 2602×3906, to DNG of size 3472×5216, NEF of size 2014×3039, and PEF files of size 3124×4688), thus to be able to have the same down-sampling factor for each image it is important to find the minimum length or width dimension for all the images.

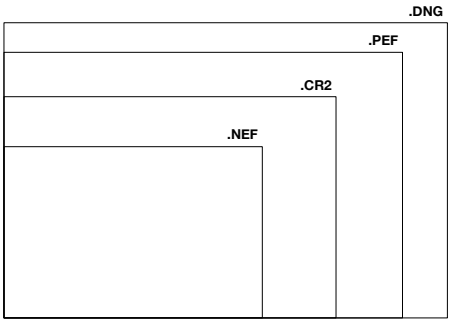


Figure 3.19: Illustration of the cropping process.

As a result, to estimate the covariance matrix for each image we developed the image using bi-linear demosaicking, luminance averaging, bilinear downscaling, and then performed a centered crop of width and height equal to $l_{min} = 2014$ (see Figures 3.19). Note that except for the crop operation and the demosaicking and down-sampling kernels, this pipeline is very similar to the one use to build the BOSSBase database.

We analyze the covariance matrix between DCT coefficients of neighboring 8×8 DCT blocks after a development pipeline similar to the one used to generate BOSSBase (see Section 3.2 for more details on the development pipeline). Since the correlations related to the host content are difficult to model, we focused our analysis on the statistical model of the photonic sensor noise. We computed the covariance matrix of 3×3 neighboring blocks of size 8×8 in the DCT domain (i.e. before quantization). The covariance matrix is estimated from 1000 RAW images with constant photo-site values $\mu = 2^{12}$ coded on 14 bits and corrupted with an additive i.i.d. signal $S \sim \mathcal{N}(0,\, a\mu + b)$, demosaicked with the bi-linear algorithm, down-sampled to a 512×512 images, and transformed into a 2D-DCT array.

3.2 Empirical estimation of the covariance matrix

When the development process is not accessible and or is not linear, it is not possible to explicitly compute the covariance matrix as was done in the previous

section. Therefore, to obtain information on the interactions between the different DCT modes, we will estimate this covariance matrix.

In order to estimate the empirical mean $\mathbf{m}$ and the covariance matrix $\mathbf{\Sigma}$ without explicitly knowing the development pipeline, the following experiment has been conducted in [85]: First, from a constant RAW image with all photo-site values 2^{12} is first generated [2]. Then a stego signal $\mathbf{s}$ at the photo-site level associated for a given pair of parameters (a, b) is added to the RAW image to simulate embedding, the resulting RAW image is stationary and can be used to estimate statistics in the developed domain. The image is processed using the development pipeline shown in Figures 3.20 to compute a vector of DCT coefficients $\mathbf{y}_t$.

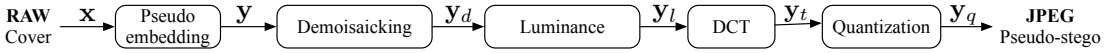


Figure 3.20: Development of a pseudo-stego 2.1 in the JPEG domain.

A set of N_o observations are generated by extracting 24×24 non-overlapping patches from the same developed image in order to gather 3×3 JPEG blocks. Note that since the image is stationary, we do not need generate several pseudo-stego images but instead we gather observations from the same developed image.

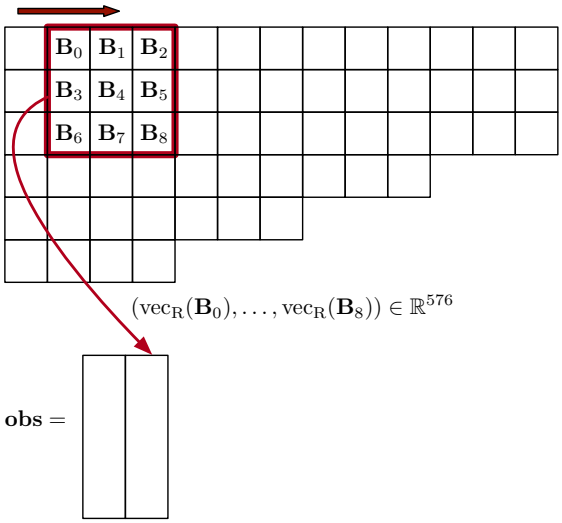


Figure 3.21: Illustration of the sliding window process.

.

Finally, the covariance matrix Σ of dimension 576×576 is finally computed from these observations, in order to get an accurate estimation of the covariance

[2]Note that the maximum value of each photo-site for the sensors considered in the present document is 2^{14}. The values are multiplied by 4 to be encoded with two bytes.

matrix, we used $N_o = 6 \times 10^4$ observations. One advantage of the covariance matrix estimation over explicit calculus is that we can blindly extract the covariance matrices for any demosaicking scheme, including development processes that are not publicly known.

3.3 Intra-block and Inter-block covariance matrices

In order to take into account symmetries of the whole 576×576 covariance matrix, for example the fact that the covariance between two horizontal neighbors is identical, the analysis of only a portion of the covariance matrix can be conducted by considering only 2×2 adjacent blocks, hence only a 256×256 covariance matrix. The scan order for the four 8×8 DCT blocks consists of a scan by rows within each block and a block-wise scan across the four blocks as shown in Figures 3.22.

By observing Figures 3.22 together with the scan order and the decomposition of the matrix into different types, we can decompose the entire covariance matrix into four different types of 64×64 matrices: one intra-block covariance matrix and three inter-block covariance matrices:

- The intra-block 64×64 covariance matrix $\mathbf{\Lambda}_{intra}$ captures the correlations between DCT coefficients of the same block.
- The horizontal and vertical covariance matrices $\mathbf{\Lambda_{0,1}}_{inter}$ and $\mathbf{\Lambda_{0,2}}_{inter}$ captures correlations between horizontal blocks and vertical blocks respectively.
- The diagonal inter-block covariance matrix $\mathbf{\Lambda_{0,3}}_{inter}$ captures correlation between diagonal blocks.

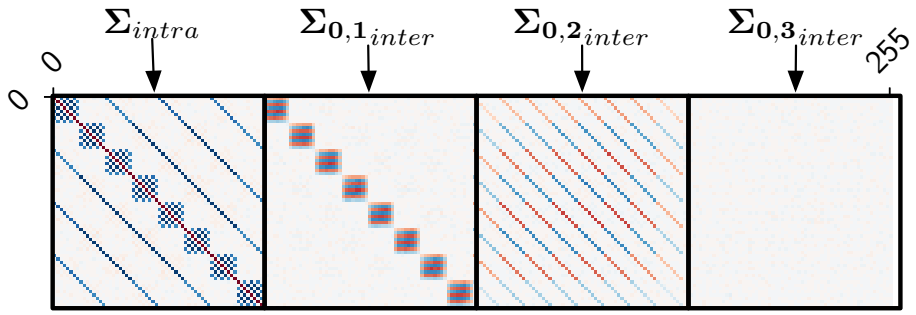


Figure 3.22: (a): scan order by block and coefficients, (b) Intra and inter correlations exhibited by the correlation matrix $\hat{\mathbf{R}}$. Blue colors denote negative correlation coefficients. See also the Appendices for the list of correlated DCT modes.

.

We can draw important remarks can be highlighted from the analysis of these covariance matrices:

- They are sparse, i.e. lot of DCT coefficients are uncorrelated.

- Within one 8×8 DCT block, one coefficient is correlated with 6 other ones, and for vertically or horizontally adjacent blocks one coefficient is correlated with 8 other ones in each connected block.
- Two diagonal blocks are nearly uncorrelated, i.e. the correlation values are very low, and in the following we consider diagonal blocks as uncorrelated.
- The patterns of the covariance matrix are immune to the type of demosaicking or down-sampling kernel. We tested the different demosaicking algorithms offered by the "rawpy" library together with different down-sampling kernels, in each case the patterns (but not the correlation values) were similar.

3.4 Intra-block correlations

The coefficients of the covariance matrix for intra-block correlations are of two types: they are either due to demosaicking artifacts (see Paragraph 3.4.1), or the consequence of low-pass filtering (see Section 3.4.2).

3.4.1 Effect of demosaicking

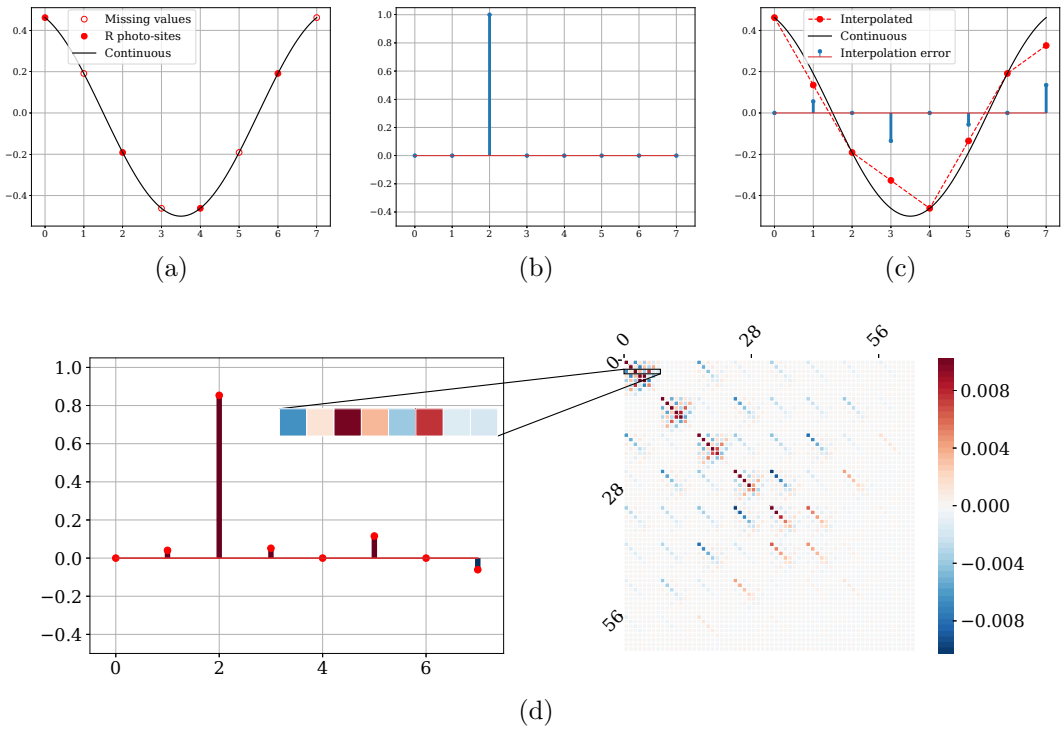
In order to emphasize the effect of demosaicking, we select only one color channel, the red one, and we investigate the intra-block correlations when the luminance computation operation is not taken into account. The demosaicking operation introduces dependencies within the same block and this is both due to the structure of the CFA itself and the color interpolation algorithm. For a given waveform of the DCT mode , i.e. its representation in the spatial domain[3], the demosaicking operation, which can be seen as a succession of sub-sampling and linear interpolation, introduces artifacts coming from interpolation errors, such that the final result is a linear combination of the other 63 DCT modes. The initial mode is encoded with a larger magnitude than the others as summed up in the following expression:

$$\mathrm{DCT}\left(\mathrm{Dem}\left(\mathrm{mode_i}\right)\right) = A_i \cdot \mathrm{mode_i} + \underbrace{\sum_{i \neq j} A_j \cdot \mathrm{mode_j}}_{\text{DCT artifacts}}, \tag{3.39}$$

here mode_i represents the spatial representation of DCT mode i after demosaicking (the Dem() function). The appearance of the A_j terms is due to small interpolation errors of mode i. These artifacts are illustrated in Figures 3.23. This Figures can be explained as follows: in order to encode continuous waveforms that are interpolated during the demosaicking process, the interpolation process has to deal with missing values (see Figures 3.23a), which encode other frequencies in the DCT domain (see Figures 3.23c). So, instead of encoding one component (see Figures 3.23b), it also encodes other DCT components (see Figures 3.23d).

[3] a.k.a. the pixel domain.

In Figures 3.23d, we also compare the covariance matrix computed by interpolating only the red channel on continuous DCT waveforms and the DCT of the interpolated waveform. Note that the fourth line of the covariance matrix is very similar with the components depicted in Figures 3.23d.



(a) (b) (c)

(d)

Figure 3.23: Impact of demosaicking on correlation between intra-block DCT coefficients: (a) visualization of one line $\mathbf{b}_c$ of the $(0, 2)$ mode in the spatial domain. (b) $DCT(\mathbf{b}_c)$. (c) Continuous signal, interpolated signal $\mathbf{b}_i$ and interpolation error. (d) comparison between the DCT transform of the interpolated waveform (left) and the covariance matrix obtained from interpolated pure DCT modes (right).

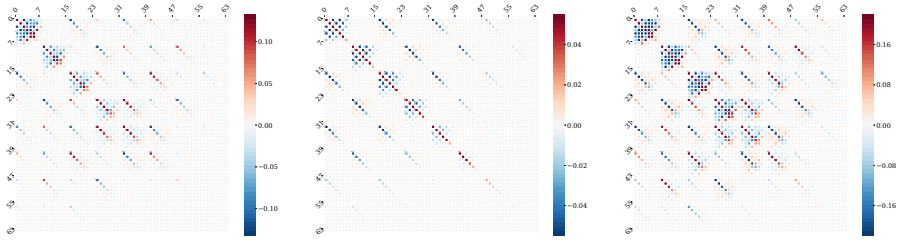
In the 2D spatial domain, for a single mode applied to a 8×8 photo-sites array, the demosaicking algorithm creates artifacts such that the resulting image in the DCT domain is a linear mixture of the different DCT modes.

3.4.2 Effect of low pass filtering

The second category of artifacts is due to a low-pass filter, which can be related to the conversion from RGB to luminance or to any down-sampling operation. In order to simulate the effect of low pass filtering, we use a random independent noise as a RAW image and convolve this input with a standard low pass filter, such as:

$$L = \frac{1}{12} \cdot \begin{bmatrix} 1 & 1 & 1 \\ 1 & 4 & 1 \\ 1 & 1 & 1 \end{bmatrix}. \tag{3.40}$$

The covariance matrix obtained by incorporating the low-pass filter in the development process is complementary to the covariance matrix obtained considering only the demosaicking artifacts. Figures 3.24 shows these relationships: the total intra-covariance matrix (Figure 3.24c) can be approximated as the superposition of the covariance matrix of signals representing the demosaicking artifacts (Figure 3.24a) and the covariance matrix of the independent signal at the photo-site level undergoing low-pass filtering (Figure 3.24b).



(a) (b) (c)

Figure 3.24: (a): Covariance matrix computed after randomly generating DCT continuous modes that are interpolated using bilinear filtering. (b): Intra correlations within a block after low-pass filtering using filter L. (c): Intra-block covariance matrix for $\mu = \text{const}$. The correlation values are thresholded for visualization purposes.

3.5 Inter-block correlations

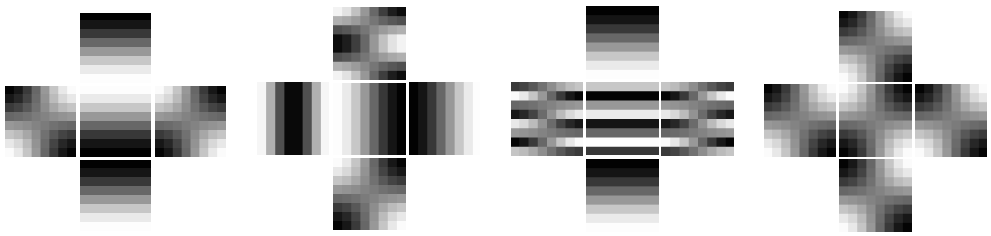
Inter-block correlations between DCT coefficients are also caused by demosaicking, which averages adjacent photo-site values to interpolate the missing color values. It creates correlations between neighboring pixels, including pixels belonging to two different DCT blocks. This interpolation process highlights the low-pass component of the sensor noise, and this is consistent across different demosaicking methods (see [84]). This phenomenon is illustrated in Figures 3.25, which shows for different DCT modes in the spatial domain, the arrangements of blocks that are the most correlated for the horizontal and vertical neighbors. For each arrangement, we can notice that the continuity from one block to its neighbor is preserved.

The most significant correlations correspond to the surrounding vertical and horizontal blocks. This is due to the large number of neighboring photo-sites

involved in the interpolation process. Note that the largest correlations are for the same vertical or horizontal frequency due to frequencies consistency between adjacent blocks.

The sign of the correlations represents the preservation of continuity between blocks in order to guarantee spatial continuity. For example, alternating signs are due to the topology of the waveforms. For example for mode $(1, 0)$, all modes $(i, 0)$ have a white top line but the bottom line alternates between white and black w.r.t. i.

It is interesting to connect this analysis with the recent steganographic scheme proposed by Li *et al.* [69] which synchronizes embedding changes between several DCT modes by empirically adjusting costs in order to favor continuities between blocks. This practical rationale is now theoretically justified by our analysis.



(a) $(1, 0)$ (b) $(0, 1)$ (c) $(0, 5)$ (d) $(1, 1)$

Figure 3.25: Different DCT modes (central blocks) and their most correlated modes (represented by horizontal or vertical blocks). The presented locations of the blocks correspond to the spatial locations of the blocks. We can notice that the most correlated blocks preserve continuities between neighboring blocks.

3.6 Decomposition into groups of uncorrelated coefficients

Because the covariance matrix after the BOSSBase development pipeline is sparse, we can decompose the set of DCT coefficients into groups where each group is only composed of uncorrelated coefficients. We end up with 8 groups, because of the following observations:

- To deal with intra-block correlations, we notice that we can find 4 sets of coefficients uncorrelated to one another. The 4 subsets (groups) $\mathbf{\Lambda_i} \in \mathbb{N}^{16}$ with $i \in \{0, \ldots, 3\}$ of these mutually decorated modes indexes are arranged thanks to a

permutation matrix $\mathbf{P}$ such that:

$$\mathbf{R}_{intra} = \mathbf{P} \underbrace{\begin{bmatrix} \mathbf{I_{16}} & \mathbf{\Lambda_{\Lambda_0,\Lambda_1}} & \cdots & \mathbf{\Lambda_{\Lambda_0,\Lambda_3}} \\ \mathbf{\Lambda_{\Lambda_1,\Lambda_0}} & \mathbf{I_{16}} & \ddots & \vdots \\ \vdots & \ddots & \mathbf{I_{16}} & \Sigma_{\Lambda_2,\Lambda_3} \\ \mathbf{\Lambda_{\Lambda_3,\Lambda_0}} & \cdots & \mathbf{\Lambda_{\Lambda_3,\Lambda_2}} & \mathbf{I_{16}} \end{bmatrix}}_{\mathbf{R_p}^{intra}} \mathbf{P}^{-1} \tag{3.41}$$

The displayed correlation matrix 3.26 after permutation of the indexes highlights the fact that within $\mathbf{\Lambda_i}$, each coefficient at least is only with itself. However, we also notice that a coefficient belonging to $\mathbf{\Lambda_i}$ with $0 < i \neq 4$, is correlated with two others coefficients from each other groups belonging to the same block.

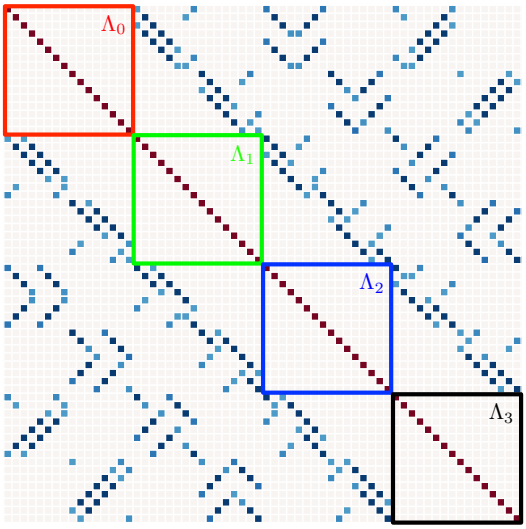
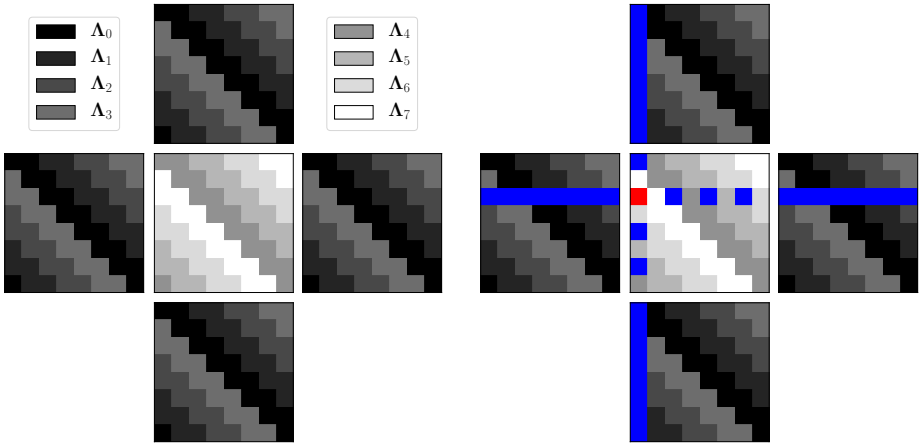


Figure 3.26: Intra-block correlations matrix after permutation $\mathbf{R_p}^{intra}$ for the 4 groups $\{\Lambda_0, \ldots, \Lambda_3\}$, colored blocks denotes the associated groups.

- To deal with inter-block correlations, we proceed in the same way. This time, we can see from the analysis of the covariance matrix that each mode is correlated with 8 modes from each connected block (see. Figures 3.22, where on sub-matrices $\mathbf{\Lambda_{0,1}}_{inter}$ and $\mathbf{\Lambda_{0,2}}_{intra}$ each DCT mode is positively or negatively correlated with 8 other coefficients). We also notice that since two diagonally-connected block are uncorrelated, we can thus sub-divide the image into two lattices, on the one hand those containing the independent blocks, and on the other those correlated with their neighbors.



(a) (b)

Figure 3.27: (a) Decomposition of the DCT modes into 8 groups, (b) The 34 modes used to compute the conditional probability (blue) of mode $(2, 0) \in \Lambda_7$ (red), the Table 6.7 explicitly lists the set of correlated modes required to sample the modes from $\mathbf{\Lambda_7}$.

	$\mathbf{\Lambda}_0$	$\mathbf{\Lambda}_1$	$\mathbf{\Lambda}_2$	$\mathbf{\Lambda}_3$	$\mathbf{\Lambda}_4$	$\mathbf{\Lambda}_5$	$\mathbf{\Lambda}_6$	$\mathbf{\Lambda}_7$
K	0	2	4	6	32	34	36	38

Table 3.2: Number of correlated coefficients k for each group considering only previous groups.

Based on the above considerations, each image can be split into 8 disjoint groups in order to sample a stego signal in the DCT domain preserving both intra-block and inter-block correlations. It is worth pointing out here that one of the drawbacks of this approach is that it ignores the content of the images.

Figure 3.27 (a) shows the locations of the uncorrelated coefficients for the different groups, and Figures 3.27 (b) highlights the locations of correlated coefficients belonging to previous groups for one given mode.

Table 3.2 indicates for group $\mathbf{\Lambda_i}$ the number of correlated coefficients, denoted K, for the groups $\{\mathbf{\Lambda_{i-1}}, \ldots, \mathbf{\Lambda_0}\}$. Tables 6.1, 6.2, 6.3, 6.4, 6.5, 6.6 and 6.7 (see the Appendices) exhibit for each mode of each groups the different correlated modes belonging to previous groups for the same block or adjacent ones as depicted on Figures 3.28.

This decomposition of the coefficients into subgroups will then be used to perform insertions from cost maps of heuristic-based algorithms while integrating additional knowledge on co-existing dependencies between DCT coefficients (see 5).

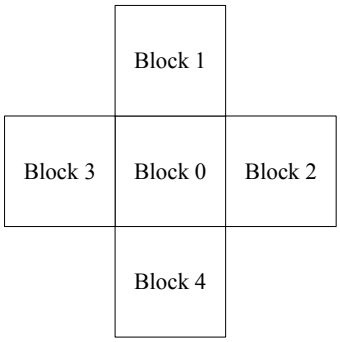


Figure 3.28: Block naming convention.

4 Impact of software and hardware developments

4.1 Impact of software development

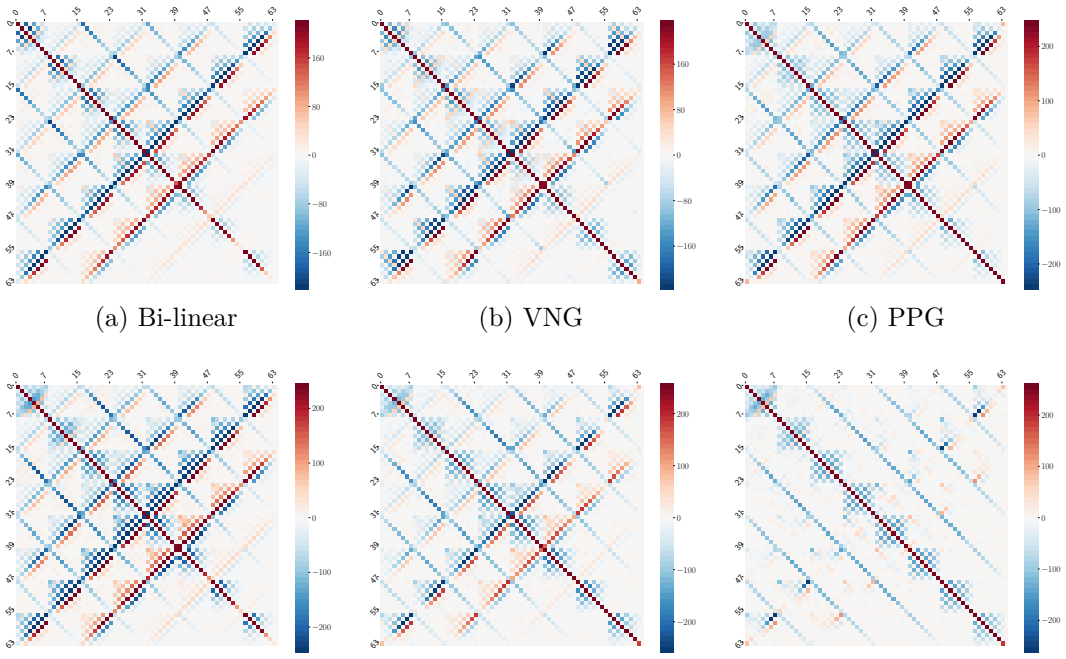
In this sub-section, we will briefly discuss the induction of correlations between the modes of the same block and with its neighboring blocks for different open source and proprietary image development algorithms.

On the Figures 3.29 we can observe the different covariance matrices for the algorithms: BIL, VNG, PPG, DCB, AHD and AAHD, all these algorithms are available via libraw. Using numerous RAW synthetics noisy images, each of these images are defined as constant RAW images with all photo-site values 2^{12} to which we add a heteroscedastic noise $\mathcal{N}(0, a\mu_{i,j} + b)$.

It is worth noticing that the intra-block correlations generated by the BIL and VNG algorithms are very similar (as it has been detailed at section 1.4). PPG and VNG algorithm seem to exhibit as well the same kind of intra-correlations, while AHD has a more sparse correlation matrix than PPG and DCB despite similarities in the sign of the correlations, AAHD produces a covariance matrix whose correlations are overwhelmingly negative in sign.

Using the same strategy the Figuress 3.30 illustrate the intra-block correlations generated by the demosaicking algorithms from Mac OSX, Adobe Lightroom and DXO.

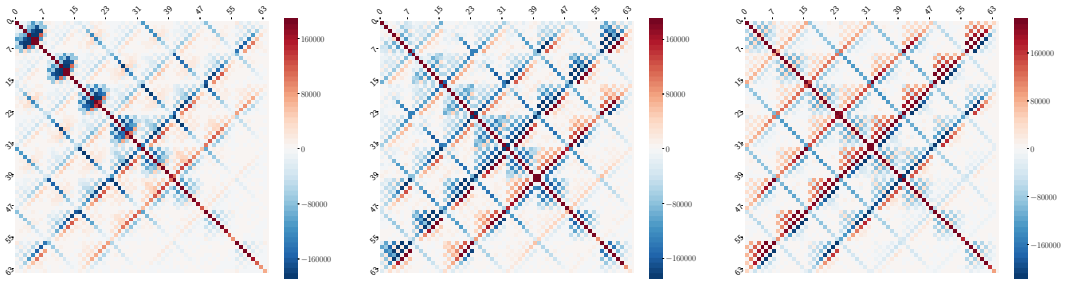
It can be noted that the demosaicking algorithm proposed by the "Preview" software of Mac OSX performs an aggressive low-pass filtering inducing correlations similar to those observed in paragraph 3.4.2. By contrast, the correlation matrices induced by the LightRoom and DXO demosaicking are much less sparse but very different nevertheless. Despite the fact that there are similarities in the demosaicking algorithms, in particular due to the low-pass filter character of the demosaicking algorithms, we can visually notice differences in these covariance matrices, which illustrates the fact that we can theoretically define a "fingerprint"



(a) Bi-linear (b) VNG (c) PPG

(d) DCB (e) AHD (f) AAHD

Figure 3.29: Intra-block covariance matrix for $\mu = const$ for different demosaicking algorithms

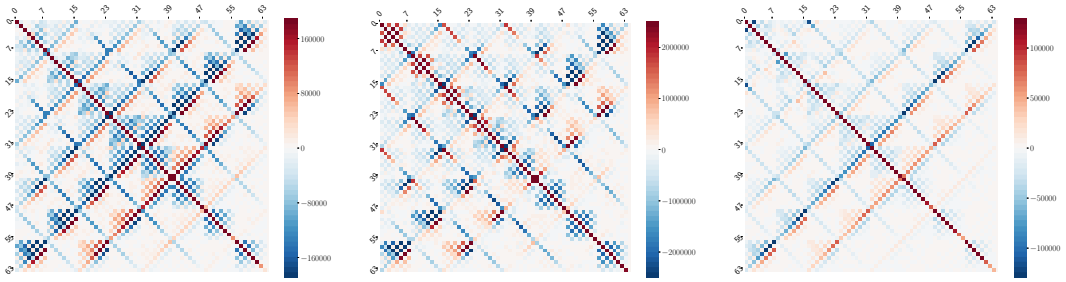


(a) Mac (b) Lightroom (c) DXO

Figure 3.30: Software

of the demosaicking algorithms.

These fingerprints are also strongly affected by other operations related to the development of an image, such as clarity (local contrast tuning), sharpness and luminance noise reduction as depicted in Figures 3.31.



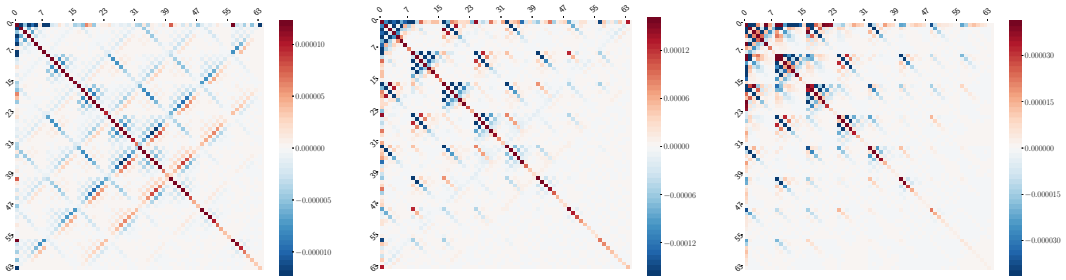
(a) Clarity (b) Sharpness (c) Luminance noise reduction

Figure 3.31: Intra-block covariance matrix for $\mu = const$ for different demosaicking algorithms

In the rest of the manuscript (in chapters 4 and 5), we will show that this statistical fingerprint needs to be modeled correctly in order to preserve undetectability during an embedding.

4.2 Impact of hardware development

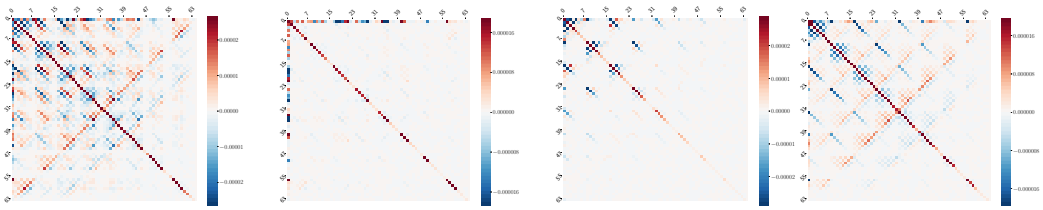
To estimate the distributions of the sensor noise after hardware development and exhibit its covariance matrix for intra-correlation in the DCT domain, we used here a the same technique that we use in 1.3: we shot a white wall under a diffuse lightning condition at a distance of 1 meter from the sensor and out of focus in order to obtain a spatial image with average constant illumination for numerous images. This operation has been performed for the following sensors: Sony A7III, Sigma DP3 and an iPad and the result is visible on Figures 3.32. We could therefore estimate the intra-correlation in the DCT domain for these different sensors. We observe that for mode $(0, 0)$ the correlations are extremely noisy, this could have been avoided by introducing images with different brightness in order to introduce diversity in the samples for mode $(0, 0)$. On the iPad and Sigma DP3, we can notice that the correlations between high frequency coefficients are small in comparison with low frequency coefficient. This can be due to the denoising operations performed in the device.



(a) Sony A7III (b) iPad (c) Sigma DP3

Figure 3.32: Covariance of DCT coefficients from images acquired by sensors and demosaicked by their respective hardwares pipelines.

For two sensors it was impossible to obtain an image without any JPEG compression, thus here are the covariances matrices resulting from the development process of the Xiaomi MI9T and the Samsung NX1100. On Figures 3.33 the covariance matrix (a) Xiaomi MI9T has been plotted using the native driver, (b) has been produced using the "Open Camera" Android application while (c) is from the native processing pipeline of a DSLR: the Samsung NX1100.



(a) MI9T OpenCamera Driver (b) MI9T Native Driver (c) Z-CAM E1 (d) Samsung NX110

Figure 3.33: Covariance of DCT coefficient from images acquired by 2 different sensors and processed by different hardwares pipelines, JPEG compression has been applied to those images thus reducing the magnitude of the correlations among high frequencies modes.

5 Conclusions of the chapter

We can conclude here on the fact that each development process introduces dependencies that are specific to this process and that are likely to be disturbed if the embedding is not carried out in such a way as to preserve them correctly. In the rest of this manuscript (in Chapters 4 and 5) we will exploit this concept in a

specific case in order to demonstrate the feasibility of a steganographic embedding that preserves a specific image model after development.

Chapter 4

Natural Steganography in the JPEG domain

In this chapter we present the implementation of a process related to the embedding in the JPEG domain within the framework of natural steganography. As a reminder, this work was initiated by the work of Patrick Bas in the spatial domain and, he published a first paper [31] dealing with the problem of an embedding in the JPEG domain published with Tomáš Denemark.

Two relatively close methods are presented in this chapter, both methods employing covariances matrices and thus assuming that the demosaicking used is linear as well as the rest of the rest of the development chain is linear. This chapter is decomposed into four main sections:

Thus, both approaches will be deployed in the context of bi-linear demosaicking coupled only to a luminance transform before proceeding to a DCT transformation.

- The first section presents the basic of Natural Steganography and outlines the different embeddings associated to this paradigm.
- In one case (section 2), the covariance matrix is analytically computed from the multivariate model detailed in the previous chapter,
- In the other case (section 3), the covariance matrix is estimated from a large number of undecimated DCT images resulting from this development process,
- The last section investigates the possibility to perform Natural Steganography on color JPEG image.

1 Embedding in the framework of Natural Steganography

The main idea behind NS relies on the principle of cover-source switching [10], which consists of generating stego content that is statistically similar to the cover

source acquired at a different camera setting. Here, the source is defined by the shot noise at the photo-site level due to the photon counting process occurring on CCD or CMOS sensors. This noise is independent across photo-sites and only depends on the sensor model and the ISO setting.

After embedding, the stego image generated from a cover acquired at sensitivity ISO_1 should have the same statistical properties as a cover image acquired at sensitivity ISO_2 in order to guarantee high empirical security. As stated in subsection 1.3 the shot noise $N^{(i)}$ at ISO_i, $i \in \{1, 2\}$, follows a zero-mean Gaussian distribution with variance determined by the luminance of the noiseless photo-site value μ:

$$N^{(i)} \sim \mathcal{N}(0, a_i\mu + b_i), \tag{4.1}$$

where the pair of parameters (a_i, b_i) depends only on the sensor and ISO_i.

Consequently at the photo-site level, because the sum of two independent Gaussian variables is still Gaussian with a variance being the sum of the variances of the two variables, **the stego signal to be added mimicking an image acquired at sensitivity ISO_2 has to be distributed as**:

$$S \sim \mathcal{N}(0, (a_2 - a_1)\mu + b_2 - b_1). \tag{4.2}$$

Since the noiseless photo-site value is unknown, we make the approximation that $\mu \approx x$, X being the random variable representing the photo-site. Then we have:

$$S \sim \mathcal{N}(0, (a_2 - a_1)x + b_2 - b_1). \tag{4.3}$$

This idea was the cornerstone of NS embedding schemes proposed for monochrome sensors in the spatial domain assuming a simplified development pipeline that includes quantization, gamma transform, and downscaling [10, 6] (the method was shown to provide high steganographic capacity with high empirical security).

Three types of steganographic embedding can be distinguished in this manuscript, as illustrated in the Figures 4.1:

- *Pseudo-embedding*,
- *Simulated embedding*,
- *True embedding*.

Pseudo-embedding means that practical embedding is not possible with the proposed implementation. Pseudo-embedding is performed at the photo-site level with the stego signal generated using 4.3. These results can be considered as a baseline but do not correspond to any practical embedding scheme in the JPEG domain. It acts as a generic mathematical operation (a reference) which outputs the so-called *pseudo-stego* image should be statistically distributed like the stego image.

In *simulated embedding*, the embedding is executed directly in the JPEG domain by manipulating the quantized DCT coefficients. The embedding changes are performed according to a given selection channel using the probability $\pi_i(k)$ of

modifying the i^{th} cover sample by magnitude $k \in \{-K, ..., K\}$. These modifications do not really correspond to the addition of information, but only to modifications of the coefficients taking into account their interdependencies. This allows us to evaluate whether the algorithm correctly models the dependencies that we wish to preserve.

(True) embedding can be achieved using multilayered STCs [37] based on costs $\rho_i(k)$ directly computed from the set of embedding probabilities $\pi_i(k)$, with $\rho_i(k) = \log(\pi_i(0)/\pi_i(k))$. The STC algorithm minimizes the sum of embedding costs while embedding the payload using a Viterbi algorithm.

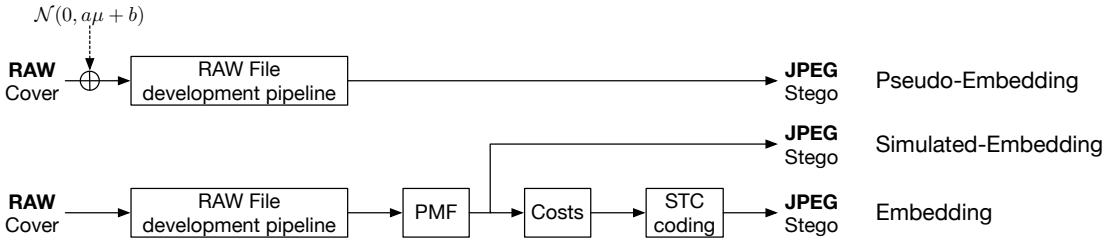


Figure 4.1: Differences between embedding, ***simulated embedding***, and pseudo embedding.

Modifying the photo-sites directly leads to pseudo-embedding. However, as mentioned in [10], it can also be directly used for ***simulated embedding*** or true embedding in the spatial domain for monochrome sensors. Based on these results the **Pseudo-Embedding is used as a baseline in terms of steganalysis performance**.

The main purpose of this chapter is to detail how to perform modifications on quantized DCT coefficients in order to perform ***simulated embedding***. The modeling of the stego signal and its dependencies in the DCT domain are crucial for the embedding to be secure. We thus focused on modeling the image development process in order to firstly derive the statistical characteristics of the stego signal in the DCT domain, then compute the modification probabilities for each DCT coefficient, and finally perform ***simulated embedding***.

In sub-section 2.1 of chapter 3, we explain how we reach the first goal and in sub-section 2.5 we detail the algorithm used to perform ***simulated embedding***.

As seen in the previous chapter, inter-block correlations between DCT coefficients are caused by demosaicking, which averages adjacent photo-site values to interpolate the missing color values and thus creates correlations between neighboring pixels and neighboring JPEG blocks.

The use of correlations between DCT coefficients is not new, as discussed in the chapter 2, and the next sub-section 1.1 briefly recalls some of the algorithms whose purpose is also to use correlations to increase empirical security or to increase the

performance of a detector.

1.1 Relationships with previous art

Current synchronization schemes in steganography often relies on a general heuristic to take into account interactions between embedding changes. However these approaches, while often competitive, lack a solid model to define the relationship between coefficients (pixels or DCT coefficients).

It is worth pointing out that using joint or conditional probability distributions among frequency domain pairs of coefficients during embedding has recently been proposed to improve the empirical security of JPEG steganography. Li *et al.* [69] define a joint distortion function for JPEG steganography to improve the empirical security of UERD and J-UNIWARD against GFR and DCTR by using block boundary constraints during embedding. Their idea relies on the principle of block boundary continuity (BBC) in frequency domain (DCT).

In order to preserve pair block realizations, the authors used simultaneous modifications on inter-block neighbors. A $+1$ on the DCT mode would imply a $+1$ on the horizontally connected neighboring blocks and a -1 on the vertically connected neighboring blocks. This allowed them to be more consistent with block boundary dependencies. By using the dependencies exhibited in the covariances matrices, we can show that BBC assumptions are a specific case of how to use our estimated covariance matrix for embedding. Despite its generality, the success of this approach has been quite mild as it did not shed any light on the origin of the correlations it tries to preserve.

Regarding steganalysis, in JRM [61] the authors proposed features built using "unions of sub-models formed as joint distributions of DCT coefficients from their frequency and spatial neighborhoods" and thus make a classifier able to take into account a wide range of statistical dependencies. The authors also showed that the strongest dependencies among DCT coefficients are those within the same block and those between the closest neighboring blocks, confirming our analysis of the covariance matrix.

2 Embedding using the computed covariance matrix

We present here an embedding scheme which relies on the use of the true covariance matrix presented in section 2 of the previous chapter. This starts with the **decomposition of the set of DCT coefficient into lattices of independent coefficients**.

2.1 Decomposition into macro-lattices

The use of four macro-lattices follows from the fact that the embedding process must take into account these three elements:

1. Intra-block dependencies within each 8×8 DCT block.
2. Inter-block dependencies between one central block and its horizontal, vertical and diagonal neighbors. For the neighboring of 3×3 DCT blocks (shown at Eq 4.4) the block in black is dependent with the blocks of its first and second order neighborhood.

$$\begin{array}{ccc} \square & \square & \square \\ \square & \blacksquare & \square \\ \square & \square & \square \end{array} \tag{4.4}$$

3. Independence of blocks that are not neighbors. For the neighborhood of 3×3 DCT blocks (shown at Eq 4.5) the two black blocks are independent.

$$\begin{array}{ccc} \blacksquare & \square & \square \\ \square & \square & \square \\ \square & \square & \blacksquare \end{array} \tag{4.5}$$

(1) means that we practically have to use 64 lattices (one lattices per DCT mode) to perform simulated embedding in one DCT block, (2) and (3) mean that we need a maximum of four macro-lattices $\{\Lambda_1, \Lambda_2, \Lambda_3, \Lambda_4\}$ to perform embedding in each DCT block while maintaining the correlations exhibited by the computed covariance matrix.

The different macro-lattices are illustrated in Figure 4.2 together with the neighboring blocks that are involved.

Considering a vector $\mathbf{s}_d^{3\times3}$ of 3×3 blocks of the stego signal in the DCT domain, let $\mathbf{s}_d^C$ be the central block and $\mathbf{s}_d^{NW}, \mathbf{s}_d^N, \mathbf{s}_d^{NE}, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^{SW}, \mathbf{s}_d^S, \mathbf{s}_d^{SE}$ be respectively the north-west, north, north-east, west, east, south-west, south, and south-east blocks w.r.t. the central one.

We can build the vector of interest $\mathbf{s}^\star$, used to compute conditional probabilities (see next sub-section 2.2), as follows:

- For $\mathbf{\Lambda}_1$, only the intra-block covariance matrix is necessary, computed w.r.t. $\mathbf{s}^\star = \mathbf{s}_d^C$,
- For $\mathbf{\Lambda}_2$, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^{NW}, \mathbf{s}_d^{NE}, \mathbf{s}_d^{SW}, \mathbf{s}_d^{SE}]$. The matrix below represents the exploited central DCT block and the exploited blocks, the square means that the block has been ignored.

$$\begin{bmatrix} \mathbf{S}_d^{NW} & \square & \mathbf{S}_d^{NE} \\ \square & \mathbf{S}_d^C & \square \\ \mathbf{S}_d^{SW} & \square & \mathbf{S}_d^{SE} \end{bmatrix} \tag{4.6}$$

- For $\mathbf{\Lambda}_3$, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^N, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^S]$,

$$\begin{bmatrix} \square & \mathbf{S}_d^N & \square \\ \mathbf{S}_d^W & \mathbf{S}_d^C & \mathbf{S}_d^E \\ \square & \mathbf{S}_d^N & \square \end{bmatrix} \tag{4.7}$$

- For $\mathbf{\Lambda}_4$, $\mathbf{s}^\star = [\mathbf{s}_d^C, \mathbf{s}_d^{NW}, \mathbf{s}_d^N, \mathbf{s}_d^{NE}, \mathbf{s}_d^W, \mathbf{s}_d^E, \mathbf{s}_d^{SW}, \mathbf{s}_d^S, \mathbf{s}_d^{SE}]$.

$$\begin{bmatrix} \mathbf{S}_d^{NW} & \mathbf{S}_d^N & \mathbf{S}_d^{NE} \\ \mathbf{S}_d^W & \mathbf{S}_d^C & \mathbf{S}_d^E \\ \mathbf{S}_d^{SW} & \mathbf{S}_d^N & \mathbf{S}_d^{SE} \end{bmatrix} \tag{4.8}$$

We end up with a decomposition of the image into $4 \times 64 = 256$ lattices (four macro lattices with one lattice per DCT mode). In each lattice, the covariance matrix can be expressed as:

$$\mathbf{\Sigma}_d = \begin{bmatrix} \mathbf{\Sigma}_{[0:64][0:64]} & \mathbf{\Sigma}_{[0:64][64:n\times 64]} \\ \mathbf{\Sigma}_{[64:n\times 64][0:64]} & \mathbf{\Sigma}_{[64:n\times 64][64:n\times 64]} \end{bmatrix} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^t & \mathbf{C} \end{bmatrix}, \tag{4.9}$$

with n denoting the number of blocks in $\mathbf{s}^\star$ (see footnote [1]) and $n = 1$ for $\mathbf{\Lambda_1}$, $n = 5$ for $\mathbf{\Lambda_2}$ and $\mathbf{\Lambda_3}$ and $n = 9$ for $\mathbf{\Lambda_4}$, see Figures (4.2).

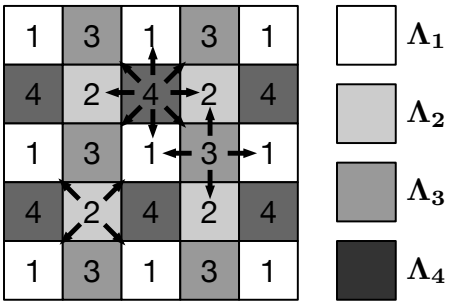
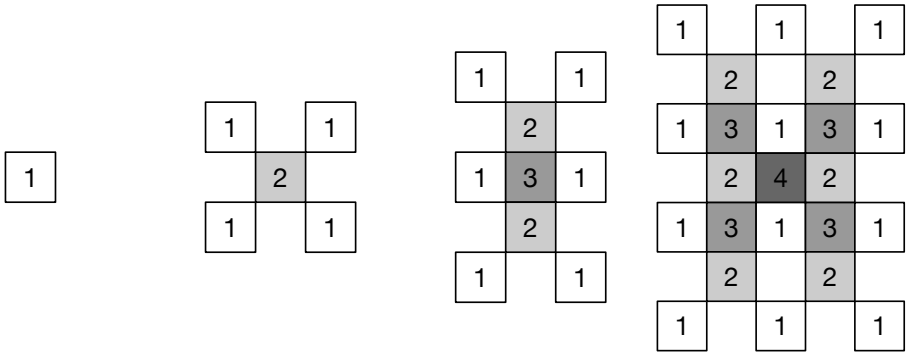


Figure 4.2: The four macro lattices used for embedding. Arrows indicate the neighborhood used to compute conditional probabilities.

It is important to notice that in order to compute $\mathbf{\Lambda_1}$, $\mathbf{\Lambda_2}$, $\mathbf{\Lambda_3}$ or $\mathbf{\Lambda_4}$ we need to have already sampled the coefficients from other lattices or from the same block. For example, to compute the distribution of the last DCT coefficient of a block belonging to the 4^{th} lattice, the dependencies between blocks of different lattices imply to have already sampled 12 blocks of $\mathbf{\Lambda_1}$, 6 blocks of $\mathbf{\Lambda_2}$ and 4 blocks of $\mathbf{\Lambda_4}$ together with 63 coefficients of the current block. Figure 4.3 shows the locations of the different blocks involved in the computation for each block depending of which lattice its belongs. We can note that the dependecies for lattice Λ_4 are important since we have here to consider a window of size 54×40 coefficients.

[1] The pythonic notation $[i : j]$ means that all indexes from the interval$[i, j - 1]$ are considered.



(a) $\mathbf{\Lambda_1}$ (b) $\mathbf{\Lambda_2}$ (c) $\mathbf{\Lambda_3}$ (d) $\mathbf{\Lambda_4}$

Figure 4.3: JPEG blocks, associated with their lattice, involved in the computation the samples of a block belonging to: (a) $\mathbf{\Lambda_1}$, (b) $\mathbf{\Lambda_2}$, (c) $\mathbf{\Lambda_3}$, (d) $\mathbf{\Lambda_4}$.

2.2 Conditional sampling

Using the macro lattice decomposition, changes can be drawn independently according to the pmf π_i for ***simulated embedding*** in each lattice, or using a STC based on costs ρ_i. In order to derive the pmf $\pi_i(k)$ for each sample i and the modification magnitude k, we need to use conditional sampling, a variation of Gibbs sampling, which enables to sample from a multivariate distribution using only conditional distributions.

Without loss of generality, if we focus on the set of 4 macro lattices defined in (section 2.1) (but this can be applied on any number of lattices that are conditionally independent), the chain rule of conditional probabilities gives:

$$\begin{aligned} P(\mathbf{s}_d) &= P(\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3}, \mathbf{s}_{\Lambda_4}), \\ &= P(\mathbf{s}_{\Lambda_1}) P(\mathbf{s}_{\Lambda_2}|\mathbf{s}_{\Lambda_1}) P(\mathbf{s}_{\Lambda_3}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}) P(\mathbf{s}_{\Lambda_4}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3}). \end{aligned} \tag{4.10}$$

where $\mathbf{s}_d$ is a random vector which groups together the whole set of vectorized DCT coefficients related to the stego signal in the DCT domain, and $\mathbf{s}_{\Lambda_i}$ represents the DCT coefficients belonging to lattice $\mathbf{\Lambda}_i$.

This means that we can perform (simulated) embedding first in macro lattice $\mathbf{\Lambda}_1$ by sampling according to $P(\mathbf{s}_{\Lambda_1})$, then embed in the second lattice by sampling according to $P(\mathbf{s}_{\Lambda_2}|\mathbf{s}_{\Lambda_1})$ and so on until embedding in macro lattice $\mathbf{\Lambda}_4$ by sampling according to $P(\mathbf{s}_{\Lambda_4}|\mathbf{s}_{\Lambda_1}, \mathbf{s}_{\Lambda_2}, \mathbf{s}_{\Lambda_3})$.

2.2.1 Conditional distribution in the continuous domain

Let denote for the next example $\mathbf{x}$ and $\mathbf{y}$ two vectors living respectfully in $\mathbb{R}^n$ and $\mathbb{R}^m$, now let assume that we have a vector $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{n+m}$ drawn from a mutlivariate normal distribution whose covariance is the following symmetric positive-definite matrix :

$$\mathbf{\Sigma} = \begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{B}^t & \mathbf{C} \end{bmatrix} \tag{4.11}$$

where $\mathbf{A} \in \mathbb{R}^{n \times n}$ is the covariance matrix of $\mathbf{x}$, $\mathbf{C} \in \mathbb{R}^{m \times m}$ is the covariance matrix of $\mathbf{y}$ and $\mathbf{B} \in \mathbb{R}^{n \times m}$ is the covariance matrix between $\mathbf{x}$ and $\mathbf{y}$.

Then, we can compute **the conditional covariance** 4.12 of $\mathbf{x}$ given $\mathbf{y}$ as the **Schur complement** [73] of $\mathbf{C}$ in $\mathbf{\Sigma}$ (defined in 4.11) and the conditionnal mean 4.13 of $\mathbf{x}$ given $\mathbf{y}$ by:

$$\mathrm{Cov}(\mathbf{x} \mid \mathbf{y}) = \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^t \tag{4.12}$$

$$\mathbb{E}(\mathbf{x} \mid \mathbf{y}) = \mathrm{E}(\mathbf{x}) + \mathbf{B}\mathbf{C}^{-1}(\mathbf{y} - \mathrm{E}(\mathbf{y})) \tag{4.13}$$

We explain now how we can compute the conditional probability related to a particular DCT coefficient.
For each macro lattice $\mathbf{\Lambda}_k$, $k \in 1, .., 4$ and block ℓ, the random vector of stego signal components conditioned by the previous embeddings follows a Multivariate Gaussian Distribution: $\mathcal{N}(\mathbf{m}_{k,\ell}, \mathbf{\Sigma}_{k,\ell})$, where $\mathbf{m}_{k,\ell}$ and $\mathbf{\Sigma}_{k,\ell}$ can be computed using the Schur complement [73]) of the full covariance matrix (eq 4.9).

For example, if we perform the embedding in block ℓ from lattice $\mathbf{\Lambda}_4$, the mean vector $\mathbf{m}_{4,\ell}$ and the covariance matrix $\mathbf{\Sigma}_{4,\ell}$ are computed conditionally to the embedding performed in $\{\Lambda_1, \Lambda_2, \Lambda_3\}$ (recall that the mean of $\mathbf{s}_d$ is 0):

$$\begin{aligned} \mathbf{m}_{4,\ell} &= \mathbf{\Sigma}_{[0:64][64:n\times 64]}\mathbf{\Sigma}^{-1}_{[64:n\times 64][64:n\times 64]}\mathbf{s}_{\Lambda_1,\Lambda_2,\Lambda_3} \\ &= \mathbf{B}\mathbf{C}^{-1}\mathbf{s}_{\Lambda_1,\Lambda_2,\Lambda_3}, \end{aligned} \tag{4.14}$$

and the Schur complement is given by:

$$\begin{aligned} \mathbf{\Sigma}_{4,\ell} &= \mathbf{\Sigma}_{[0:64][0:64]} - \mathbf{\Sigma}_{[0:64][64:n\times 64]}\mathbf{\Sigma}^{-1}_{[64:n\times 64][64:n\times 64]}\mathbf{\Sigma}_{[64:n\times 64][0:64]} \\ &= \mathbf{A} - \mathbf{B}\mathbf{C}^{-1}\mathbf{B}^t, \end{aligned} \tag{4.15}$$

for the stego-signal $\mathbf{s}_{\Lambda_1,\Lambda_2,\Lambda_3}$ defined by the surrounding blocks belonging to the three first lattices (see Figures 4.2).

At this stage of the study, it is possible to generate the 64 stego signal values $\mathbf{s}_{k,\ell} = (c_0, \ldots, c_{63})^t_{k,\ell}$ in the DCT domain.

For each of the 64 lattices in each macro lattice, we sample by using the **Cholesky decomposition** of the corresponding covariance matrix $\mathbf{\Sigma}_{k,\ell}$, denoted $\mathbf{L}_{k,\ell}$, which is a lower triangular matrix such that $\mathbf{\Sigma}_{k,\ell} = \mathbf{L}_{k,\ell} \cdot \mathbf{L}^t_{k,\ell}$.

Let $(N_1, N_2, \cdots, N_{63}) \sim \mathcal{N}(\mathbf{0}, \mathbf{I}_{64})$ a standard multivariate Gaussian distribution, and $\mathbf{n} = (n_0, \ldots, n_{63})$ a random sample of it. Then $\mathbf{s}_{k,\ell} \sim \mathcal{N}(\mathbf{m}_{k,\ell}, \mathbf{\Sigma}_{k,\ell})$ can be sampled by computing $\mathbf{s}_{k,\ell} = \mathbf{m}_{k,\ell} + \mathbf{L}_{k,\ell}\mathbf{n}$. More precisely, because we need

to generate $\mathbf{s}_{k,\ell}$ iteratively, as $\mathbf{\Sigma}_{k,\ell}$ is factorized by Cholesky decomposition and we can successively calculate $s_0, s_{1|0}, \ldots$ omitting here indexes (k,ℓ) for writing convenience, we have:

$$\begin{cases} s_0 = m_0 + L\,(0,0)\cdot n_0 \\ s_{1|0} = \underbrace{m_1 + L\,(1,0)\cdot n_0}_{m_{1|0}} + \underbrace{L\,(1,1)}_{\sigma^2_{1|0}}\cdot n_1 \\ \quad \vdots \end{cases}, \tag{4.16}$$

and each coefficient can thus be sampled according to the following law :

$$S_{i|i-1\ldots,0} \sim \mathcal{N}\left(m'_i, \sigma'^2_i\right) \quad 1 \leq i \leq 63, \tag{4.17}$$

with $m'_i = m_i + \sum_{l=0}^{i-1} L(i,l)n_l$, and $\sigma'^2_i = L^2(i,i)$, $i \geq 1$, $m'_0 = m_0$, $\sigma'^2{}_i = L^2(0,0)$.

Equation (4.17) gives consequently the conditional distribution of each sample of the stego signal in the continuous domain.

2.3 Computation of the probability mass functions and sampling

Using the JPEG quantization matrix, the stego signal undergoes a quantization and the conditioned probability density function has to be to converted into a probability mass function which takes into account the associated quantization table for the chosen quality factor QF. To compute $\pi_i(k) = \Pr[\bar{S}_i = k]$, the probability that the stego signal produces a change of magnitude $k \in \mathbb{Z}$ at a coefficient $i \in \mathbb{N}$ for a given block, we compute the quantized version of the real valued random variable S_i. This probability mass function is given by:

$$\begin{aligned} \pi_i(k) &= \Pr\Big[u_k < \frac{S_i}{Q_i} \leq u_{k+1}\Big], \\ &= \int_{u_k}^{u_{k+1}} \frac{1}{\sqrt{2\pi\hat{\sigma}_i^2}} \exp\Big(-\frac{(x-\hat{m}_i)^2}{2\hat{\sigma}_i^2}\Big)\mathrm{d}x, \\ &= \frac{1}{2}\Big[\mathrm{erf}\Big(\frac{u_{k+1}-\hat{m}_i}{\sqrt{2}\hat{\sigma}_i}\Big) - \mathrm{erf}\Big(\frac{u_k-\hat{m}_i}{\sqrt{2}\hat{\sigma}_i}\Big)\Big], \end{aligned} \tag{4.18}$$

where $u_k = [\hat{m}_i] - 0.5 + k$, $\hat{m}_i = m'_i/Q_i$, $\hat{\sigma}_i = \sigma'_i/Q_i$ for parameters m'_i and σ'_i before quantization associated with a quantization step Q_i. At each step i, the parameters m'_i and σ'_i have to be generated in the continuous domain with the knowledge of values drawn at steps $0 \leq l \leq i-1$. The previous continuous samples are then needed to compute m'_i and σ'_i. Once a sample has been drawn in the discrete domain, we must then find a suitable candidate in the continuous domain

that could have led to the sampled discrete value. This could be done for example by using rejection sampling, where we can obtain for each discrete sample $\bar{s}_i$ its continuous candidate $S_{i|\bar{s}_i}$.

In this case, rejection sampling can be use in the following way: for each discrete sampled value, we sample according to the continuous distribution $S_{i|i-1\ldots,0} \sim \mathcal{N}(m'_i, \sigma'^2_i)$ until we find the appropriate candidate $S_i|\bar{s}_i$ such that:

$$u_k < S_i|\bar{s}_i < u_{k+1}. \tag{4.19}$$

where $\bar{s}_i = k$, $u_k = [\hat{m}_i] - 0.5 + k$, and $k \in \mathbb{Z}$ the symbol sampled as a modification in the discrete domain.

It should be pointed out that during this step, we need to both to embed/sample on JPEG coefficients, and to sample in the continuous domain in order to be able to compute the conditional distribution using (4.17), which is illustrated on Figures 4.4 and explained in the next sub-section 2.4.

2.4 Entropy estimation

In order to evaluate the embedding capacity of this methodology on the image database, we can evaluate the binary entropy of the added signal after embedding. From the probability mass function obtained in the previous section, the binary entropy associated with the steganographic signal for the i-th coefficient can be calculated. Given the alphabet $\mathcal{A} = (-K, \ldots, 0, \ldots, K)$, $k \in \mathbb{N}^+*$, the binary entropy can be computed by:

$$H(\mathcal{A}, i) = -\sum_{k \in A} \pi'_i(k) \log_2 \pi'_i(k), \tag{4.20}$$

where $\pi'_i(k) = \pi_i(k)$ for $i \in \{-K-1, \ldots, K-1\}$, $\pi'_i(-K) = -\sum_{i=-\infty}^{i=-K} \pi_i(k)$ and $\pi'_i(K) = -\sum_{i=K}^{i=+\infty} \pi_i(k)$.

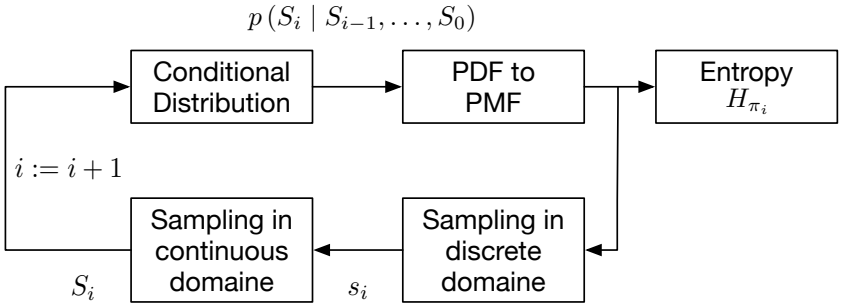


Figure 4.4: Sequential computation of the PMF needed to perform ***simulated embedding***.

2.5 Final embedding algorithm

The resulting embedding algorithm (named **J-Cov-NS**) can be decomposed into the following steps, summed up in the pseudo code presented in Algorithm 1. The use of the key in not explicit, but it can be used to shuffle the coefficients withing each lattice. The embedded payload is such that its size matches Eq. (4.20).

Algorithm 1 J-Cov-NS embedding scheme.

- **Inputs**: the cover RAW image $\mathbf{X}_p$, the payload, a secret key
- **Develop** $\mathbf{X}_p$ in the DCT domain, before quantization to obtain $\mathbf{X}_d$ and in the JPEG domain to obtain $\mathbf{X}_j$;
- **Divide** $\mathbf{X}_p$ into 4 macro-lattices $\mathbf{\Lambda}_1$, $\mathbf{\Lambda}_2$, $\mathbf{\Lambda}_3$, $\mathbf{\Lambda}_4$; - **For** each macro-lattice $\mathbf{\Lambda}_i$ **do**:
  - **For** each DCT block of $\mathbf{\Lambda}_i$ **do**:
    - Compute the covariance matrix for each set of DCT blocks (Eq. (4.21);

$$\mathbf{\Sigma}_d \quad = \mathbf{M}\,\mathbb{E}\left[\mathbf{S}_p\,\mathbf{s}_p^t\right]\,\mathbf{M}^t. \tag{4.21}$$

    - Compute the conditional mean vector (Eq. (4.14)) and the conditional covariance matrix (Eq. (4.15)) w.r.t. the embeddings done on the previous lattices;
      - **For** each DCT coefficient of $\mathbf{X}_d$ **do**:
        - Compute the conditional distribution Eq. (4.17) given the previous embedding changes;
        - Compute the PMF $\pi_i(k)$, Eq. (4.18);
        - Perform the modification on $\mathbf{X}_j$ by sampling according to $\pi_i(k)$;
        - Sample the continuous variable related to the modification, Eq (4.19);
- **Return** the JPEG stego image $\mathbf{Y}_j$.

2.6 Steganalysis Setup

We evaluate the proposed embedding scheme to test on images taken by the Micro 4/3 16 MP CMOS sensor from the Z CAM E1 action camera.

Note that this steganalysis setup is relatively unconventional compared to the state of the art (see Figures 4.5). This is due to the fact that the goal of the classifier here is to distinguish between cover images captured at ISO_2 from stego images coming from cover images captured at ISO_1 but emulating sensor noise captured at ISO_2.

We thus used the E1Base 2.6.1 image base composed of pairs of images acquired at ISO1 and ISO2.

2.6.1 Generation of E1Base

Raw images coming from the E1 sensor are acquired with two ISO settings (ISO 100 and ISO 200) and constitute *E1Base*. This database can be downloaded at https://gitlab.cristal.univ-lille.fr/ttaburet/e1base and is built according to the following requirements:

- It contains an equal number of images of equivalent scenes captured at both $ISO_1 = 100$ and $ISO_2 = 200$. The training and testing sets have been generated from 200 Raw images (DNG format, with a 12 bits dynamic range) that have been developed and cropped without overlapping in order to provide $10,800$ images of size 512×512. This dataset has already been used under similar circumstances in [30, 84, 81].

- A particular care has been taken in order to ensure that the only important difference between the database acquired at ISO_1 and the database acquired at ISO_2 is the sensor noise. In the same way as the MonoBase was acquired by a monochrome sensor [10], the average focus and average luminance are both similar between the two databases. This step is mandatory in order to guarantee that the steganalyzer is not using semantic information to distinguish between the cover and stego datasets. This requirement is specific to the benchmarking process of Natural Steganography since the cover and stego images do not come from the same source in this case.

For this given database, the value used to compute the variance of the sensor noise at the photo-site level are $(a_2 - a_1) = 1.15$ and $(b_2 - b_1) = -1150$ (the variance is set to zero when it is negative).

Classically, *E1 Base* is split into two halves, 5400 pairs of images are used for training and 5400 pairs for testing.

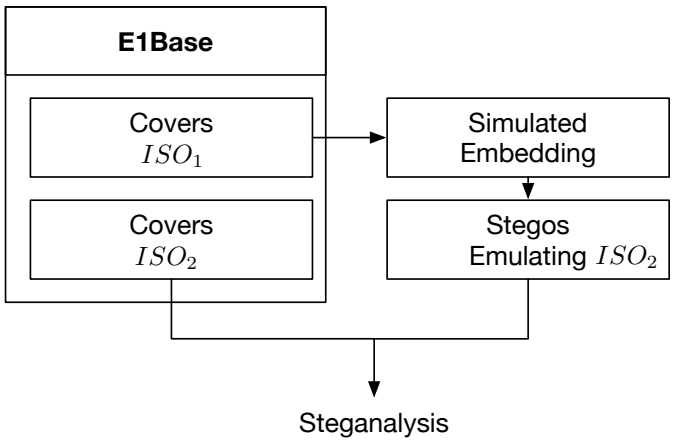


Figure 4.5: Steganalysis setup when benchmarking NS.

2.7 Results

We propose a large variety of results at different JPEG quality factors and for different alphabet sizes in the following sub-section 2.7.

2.7.1 Practical security

We adopt the DCTR features set [50] combined with a low complexity linear classifier [27] to perform the steganalysis with the threshold set in order to minimize the total classification error probability under equal priors, $P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}})$, with P_{FA} and P_{MD} standing for the false-alarm and missed-detection rates, respectively.

For comparison with the current state of the art (of side informed schemes in the JPEG domain), we embedded all images also with SI-UNIWARD with an embedding rate of 1 bit per nzAC coefficient. In this case, the steganalysis task is the classic one: try to distinguish stegos images (produced by SI-UNIWARD) from covers acquired at ISO_2.

P_E (%) / QF	H (bpnzAC)	**J-Cov-NS**	Pseudo emb.	**J-Cov-NS-scaling** [84]	Independent emb. [30]	Intra-block only	SI-Uniward [53] 1 bpnzAC
100	2.0	**42.9**	40.2	13.9	0.0	0.0	0.0
95	2.2	**41.2**	40.9	30.3	0.5	0.2	0.4
85	2.4	**41.2**	41.9	39.8	10.8	15.8	12.3
75	7.0	**41.6**	41.3	40.4	27.0	25.2	24.8

Table 4.1: Empirical security (P_{E} in %) and average embedding capacity (H) for different quality factors and embedding strategies on E1Base. DCTR features combined with regularized linear classifier are used for steganalysis.

Table 4.1 compares the proposed embedding scheme (called **J-Cov-NS**) for different JPEG QF with other embedding strategies which are:

- *Pseudo embedding* in the photo-site domain and applying the process depicted in the top raw of Figures 3.20,
- *J-Cov-NS-scaling* : estimating the empirical covariance matrix from a stationary signal and scaling it according to the average RGB values of the raw image, which is one solution to circumvent the explicit calculus of the covariance matrix [84],
- *Independent embedding* : embedding without taking into account correlations between DCT coefficients, this is performed by computing an empirical histogram of each DCT mode estimated after multiple embeddings and Monte-Carlo simulations [30],
- *Intra-block only* : embedding taking into account only intra-block correlations, this is performed by using only the computation of the intra-block covariance matrix, no inter-block correlations are consequently considered here,

- *SI-UNIWARD* [53], one state of the art embedding scheme in the JPEG domain which use side-informed embedding from the RAW image.

We can notice that computing the covariance matrix for each DCT block enables us to achieve about the same practical security than pseudo-embedding. Contrary to the previous scheme proposed in [84], which relies on an approximation of the covariance matrix using a scaling factor dependent on the RGB values of each block, **J-Cov-NS** does not exhibit any security loss for high QFs. The comparison with independent embedding, which offers good practical security for monochrome sensors, highlights the fact that the latter scheme is not adapted to color sensors, and that it is extremely important to take into account correlations between DCT coefficients, especially for high QFs. Note also that if only the intra-block correlations are taken into account, the embedding scheme still remains highly detectable. Finally, the comparison with SI-UNIWARD shows that this state-of the art scheme is not secure for very high embedding rates (1 bit pnzAC coefficient here). This is not surprising since SI-UNIWARD does not rely on cover-source switching and does not use all the information provided by the development pipeline.

2.7.2 Evaluation for other steganalysis strategies

Furthermore, we also assessed **J-Cov-NS** w.r.t to other steganalysis strategies devoted to JPEG images. For that purpose, we carried out steganalysis using other JPEG feature sets based on the residuals extracted using Gabor filters (GFR, see [80]) and also using the nonlinear set classifier [**kodovsky2012set**] for different JPEG FQs. The results are reported in the table 4.2 and show that both strategies are equivalent with the first one, with a slight advantage on GFR characteristics compared to DCTRs (-1% to -3%). Note however that the GRF features have a higher dimensionality (17.10^3 vs 8.10^3) and are longer to perform. However, the use of the ensemble classifier also allows to decrease the detectability, but with a narrow margin of 1% maximum, as well as a computational cost of about an order of magnitude.

We hypothesize that (i) GFR are not adequate to model the photonic noise and (ii) that ensemble classifier, which are composed of a set of weak classifiers, are in this case not appropriate because each base learner is too weak.

Since deep neural network-based steganalysis offers the advantage of automatically finding the relevant features regardless of the embedding scheme, we also benchmark the J-Cov-Net w.r.t. SRNet, a state-of-the-art network in spatial steganalysis or JPEG [13]. The network was trained using mini batches of 32 512×512 images (16 covers and 16 stego) using the Nvidia Quadro P6000 GPU (24 GB memory), the learning rate was initially set at 10^{-3} and decreases by 10% with each 5000 iterations. The size of the training set is 4000 pairs (increased by rotations and flip transforms), 1000 pairs are used for validation to select the best trained

network, and the rest for testing. The results reported in the table 4.3 are obtained after convergence has been reached, i.e. after 100,000 iterations. It can be noted that DNN-based steganalysis increases the detectability performance by about 10% w.r.t the DCTR combined with the low complexity linear classifier. With an average error rate of more than 30%, this does not undermine the detectability of the presented embedding. In addition, it should be pointed out that this improved detectability may be due to the fact that the automatic feature extraction provided by SRNet's convolutional layers is able to detect possible slight overall content discrepancies between E1Base images acquired at ISO 100 and 200.

QF /	Linear Classifier		Ensemble Classifier	
P_E (%)	DCTR	GFR	DCTR	GFR
100	42.9	40.3	40.8	39.6
95	41.2	39.2	41.3	38.4
85	41.2	39.1	41.0	38.1
75	41.6	40.3	41.4	39.1

Table 4.2: Practical security of **J-Cov-NS** for other steganalysis strategies: DCTR and GFR features sets using the Linear Classifier and the Ensemble Classifier.

QF	100	95	75
P_E (%)	37.4	31.2	35.0

Table 4.3: Practical security of **J-Cov-NS** against SRNet.

2.7.3 Embedding capacity

We investigate in this section the distribution of the embedding capability across the E1Base database, and compute its average value for JPEG $QF \in \{75, 85, 95, 100\}$ and for different alphabet sizes. Therefore, for each 512×512 image, we estimate the binary entropy, compute the proportion of nzAC and get $H_{bits/pixels}$ and $H_{bits/nzAC}$ based on the size of the alphabet chosen for each QF. Figure 4.6a and Figures 4.6b illustrate, respectively, the evolution of $H_{bits/pixels}$ and $H_{bits/nzAC}$ when the size of the alphabet for insertion increases from $\left[\begin{array}{ccc} -1 & 0 & +1 \end{array} \right]$ to $\left[\begin{array}{ccc} -5 & \dots & +5 \end{array} \right]$.

The average embedding capacity in bits per nzAC is relatively high, around 2 bits pnzAC for JPEG QF$\in \{85, 95, 100\}$ and over 7 bits pnzAC for $QF \in \{75\}$. The alphabet size has a minor impact on the capacity. However, QF $\in \{75, 85\}$ highlights an exotic case, since on the one hand the embedding is concentrated on the DC coefficients, and on the other hand there are only few nzAC coefficients at

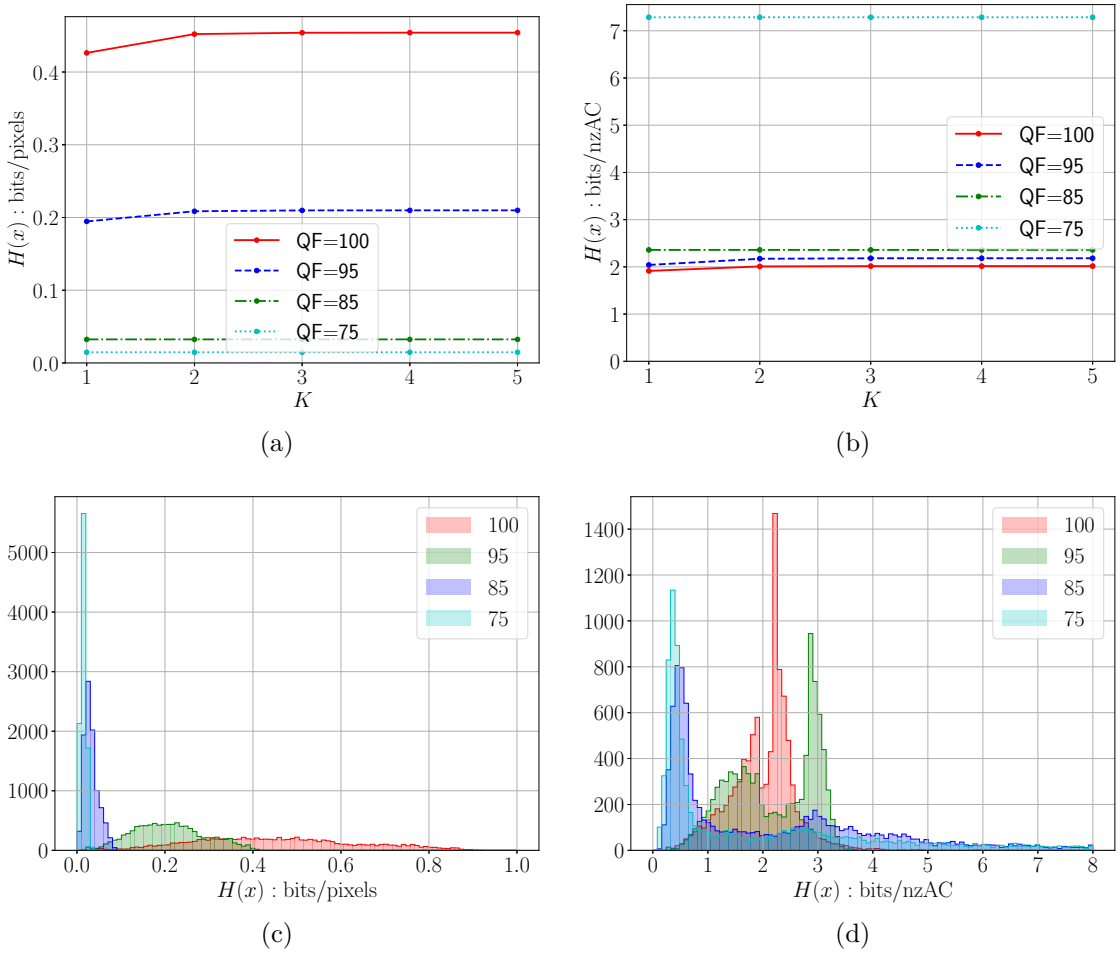
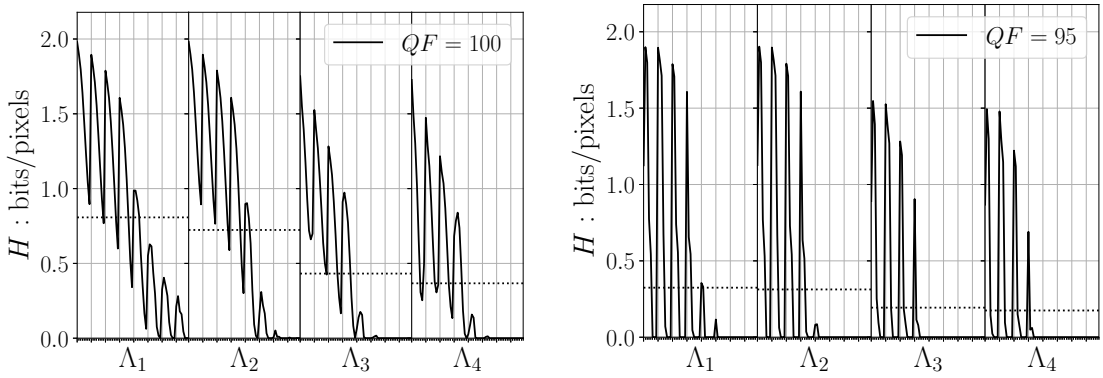


Figure 4.6: Average entropy H (bits) of **J-Cov-NS** over the database (a) per pixel, (b) per nzAC as a function of K for different JPEG QFs. Histograms of H (bits) across images for different QFs in (c) per pixel, (d) per nzAC.

$QF \in \{75, 85\}$. For example, given a 512×512 image with an average embedding rate of 1 bit per DC coefficient and having only 100 non-zero AC coefficients, this image has a total embedding rate of 40.96 bits per nzAC (at $QF = 75$)!

Figure (4.7) shows the embedding capacity computed on a synthetic constant cover RAW image for each DCT coefficient on the four lattices $\mathbf{\Lambda}_1$, $\mathbf{\Lambda}_2$, $\mathbf{\Lambda}_3$, and $\mathbf{\Lambda}_4$ at $QF100$ and $QF95$. Within each block, row scan is used. Two remarks can be drawn:

1. the capacity decreases w.r.t. the coefficient frequency, this is due to demosaicking and the fact that the stego signal is mainly encoded by low frequency components. For $QF95$, this is also due to the fact that the quantization steps are larger for high frequencies.
2. the capacity decreases w.r.t. the lattice index, with an average value at $QF100$ of 0.8 bpp for $\mathbf{\Lambda}_1$ to 0.4 bpp for $\mathbf{\Lambda}_4$. This is because conditioning reduces the entropy of a random variable [29]. At $QF100$, where the quantization is the same for each DCT mode, this is particularly noticeable by examining the entropy of the last 8 coefficients of each block, which are up to 0.3 bpc for $\mathbf{\Lambda}_1$ but, due to conditioning, are reduced to zero for $\mathbf{\Lambda}_4$.



(a) $QF100$ (b) $QF95$

Figure 4.7: Evolution of the embedding rates computed from an i.i.d. Gaussian RAW image for each DCT mode and each sub-lattice for different JPEG QFs. Row scan is used within each sub-lattice. Dotted lines denote the average embedding rate within each sub-lattice.

2.7.4 Impact of the demosaicking algorithm used

The Figures 4.8 highlight the consequences on the empirical security of sampling the covers using a demosaicking algorithm that does not preserve the Gaussianity of the sensor noise distribution. The stego images have been sampled using the four

lattices strategy with the analytically computed covariance matrices and covers have been generated using bi-linear, DCB, VNG and AAHD demosaicking methods.

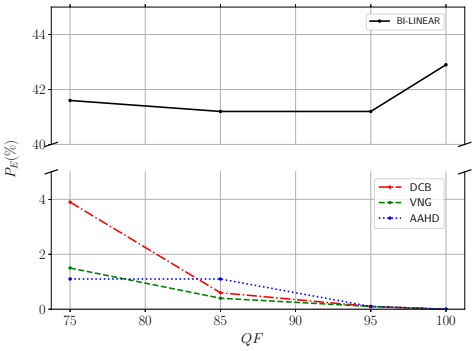


Figure 4.8: Investigation on the choice of the demosaicking algorithm for the generation of the covers while the stegos were sampled using analytically computed covariance matrices.

2.7.5 Impact of the alphabet size

The impact of the alphabet size $(2K+1)$ on the implementation of **J-Cov-NS** is presented in Table 4.4 for different JPEG QF. We can notice that ternary embedding $(K=1)$ is associated with a very detectable implementation for $QF95$ and $QF100$. This is due to the fact that the truncation of the modification changes alters considerably the distribution of the stego signal which cannot mimic anymore the ISO switch for small quantization steps. On the other hand, heptary embedding offers detectability comparable to that of an infinite alphabet for $QF95$ and should be used for true embedding combined with multi-layer STC in this case. We can also notice that for $QF \leq 85$ ternary embedding offers already the same practical security than pentary embedding.

QF / P_E in %	$K=1$	$K=2$	$K=3$	$K=5$
100	1.0	12.9	28.7	40.4
95	3.5	23.6	39.3	40.9
85	39.8	39.8	39.8	41.8
75	40.4	40.4	40.4	41.2

Table 4.4: Practical security of **J-Cov-NS** w.r.t. alphabet size and different QF.

3 Embedding using the estimated covariance matrix

The difference between monochrome and color sensors was studied in [30] with the conclusion that independent embedding on each DCT coefficient offers high empirical security for monochrome sensors, but not for color sensors. This is due to the fact that demosaicking introduces dependencies among neighboring DCT coefficients. When these dependencies are not taken into account, the embedding scheme becomes highly detectable at high JPEG Quality Factors (QF). To overcome this problem, the authors of paper [84] modeled these dependencies using the multivariate Gaussian model with the covariance matrix of the stego signal in the DCT domain $\hat{\mathbf{\Sigma}}$ estimated from a constant-luminosity RAW image altered by shot-noise.

3.1 Covariance matrix estimation

In order to estimate the empirical mean $\mathbf{m}$ and the covariance matrix $\mathbf{\Sigma}$ without explicitly knowing the development pipeline, the following experiment has been conducted:

- A constant RAW image with all photo-site values 2^{12} is first generated. Note that the maximum value of each photo-site for the sensors considered in this document is 2^{14}. The values are multiplied by 4 to be encoded with two bytes.
- A stego signal $\mathbf{s}$ at the photo-site level associated for a given pair of parameters (a, b) is added to the RAW image to simulate embedding. The resulting RAW image is stationary and can be used to estimate statistics in the developed domain.
- We develop the image using the development pipeline shown in Figures 3.20 to compute a vector of DCT coefficients.
- A set of N_o observations are generated by extracting 24×24 non-overlapping patches from the same developed image in order to gather 3×3 JPEG blocks. Note that since the image is stationary, we do not need to generate several pseudo-stego images but we can instead gather observations from the same developed image.
- The covariance matrix Σ of dimension 576×576 is finally computed from these observations. In order to get an accurate estimation of the covariance matrix, we used $N_o = 6 \times 10^4$ observations.

The embedding was then designed to respect the required covariance among stego DCT coefficients. The empirical security of a simulated-embedding scheme[2]

[2]As commonly done for cost-based embedding schemes, here the message is not embedded but the embedding changes are simulated by computing the embedding change probabilities and

was indeed larger than when the embedding was assuming independent DCT coefficients [30].

The embedding scheme is slightly the same that the one described by the algorithm 1, the only difference is that here the covariance matrix is estimated and is subsequently adjusted by a scaling step. The embedding scheme is summarized

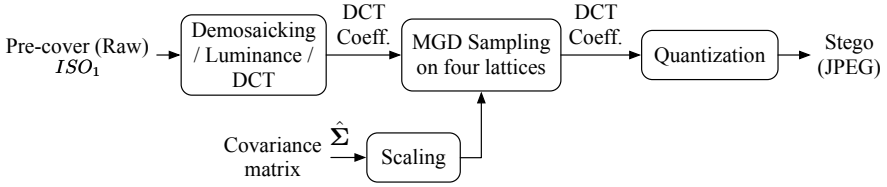


Figure 4.9: Overview of the embedding scheme presented in [84].

in Figures 4.9 and can be thus be decomposed into different steps described below:

1. **Estimation of the covariance matrix $\hat{\mathbf{\Sigma}}$ between DCT coefficients of 3×3 neighboring 8×8 blocks.** Since it is computationally infeasible to estimate the covariance matrix for each block of the cover image, the estimation is performed on a constant-luminosity RAW image with photo-site values $\mu = 2^{12}$ coded on 14 bits and corrupted with the stego signal S.This estimation uses $N_o = 6 \times 10^4$ observations of 24×24 DCT coefficients obtained from a developed (3480×4640) RAW image.
2. **Beginning of the development (demosaicking, luminance transform, and DCT).** The cover RAW image follows a classical development pipeline to generate gray-scale JPEG images. After demosaicking, the standard RGB to luminance transform[3] given by: $\mathbf{y}_l = 0.299\,\mathbf{y}_r + 0.587\,\mathbf{y}_g + 0.114\,\mathbf{y}_b$, is applied, followed by a 2D-DCT transform on 8×8 blocks.
3. **Scaling of $\hat{\mathbf{\Sigma}}$.** Since we assume that the development is linear, and in order to take into account the conversion from RGB to luminance, the covariance matrix associated with each block is scaled as $\hat{\mathbf{\Sigma}}' = \gamma\hat{\mathbf{\Sigma}}$, where γ represents the scaling factor given by:
$$\gamma = \frac{0.299^2(a\bar{x}_r + b) + 0.587^2(a\bar{x}_g + b) + 0.114^2(a\bar{x}_b + b)}{(0.299^2 + 0.587^2 + 0.114^2)(a2^{12} + b)}, \tag{4.22}$$
where $\bar{x}_r$, $\bar{x}_g$, and $\bar{x}_b$ represent, respectively, the average photo-site value of the red, green, and blue component of the block that is sampled.
4. **Sampling on four lattices.** In [84], the authors have shown that for this development, the stego signal generated on two non 8-connected blocks is independent and that the dependencies between 8-connected blocks are solely

sampling according to them.

[3]Without loss of generality we assume that no other color transform is applied, however if one is applied, it only changes the different weights.

due to demosaicking. Consequently, we can use four lattices $\{\Lambda_1, \Lambda_2, \Lambda_3, \Lambda_4\}$ depicted in Figures 4.2 to sample the stego signal in the DCT domain. DCT blocks belonging to lattice Λ_1 are sampled independently, blocks belonging to lattice Λ_2 are sampled conditionally on the four diagonal blocks, blocks belonging to lattice Λ_3 are sampled conditionally on the four vertical and horizontal blocks, and blocks of Λ_4 are sampled conditionally on the 8 surrounding blocks. We can show that for a linear development, the distribution of the stego signal for each lattice Λ_j ($j \in \{1, .., 4\}$) and block i follows a Multivariate Gaussian Distribution (MGD):

$$\mathcal{N}(\mathbf{m}_{i,j}, \mathbf{\Sigma}_{i,j}), \tag{4.23}$$

where for $j \in \{2, 3, 4\}$, the expectation vector $\mathbf{m}_{i,j}$ and covariance matrix $\mathbf{\Sigma}_{i,j}$ of the conditional distribution are computed using the Schur complement of the estimated covariance matrix $\hat{\mathbf{\Sigma}}$ (see section 2).

5. **JPEG quantization.** The simulated stego signal is quantized using the JPEG quantization matrix for a given Quality Factor (QF). One can also compute the pmf of the stego signal on JPEG coefficients at the expense of increased complexity. Note that in order to perform practical embedding, one must use 64 sub-lattices in each block (one for each DCT coefficient), this can be done by computing the conditional pmf and the associated costs (see [84]). The pmf $\pi_{q,i}$ for each coefficient i considering a Q-arry alphabet and symbol q is also used to estimate the average payload embedded in each coefficient by computing the entropy

$$H(\pi_{q,i}) = -\sum_{q=1}^{Q} \pi_{q,i} \log_2 \pi_{q,i}. \tag{4.24}$$

As detailed in [84], the $\pi_{q,i}$ are computed by dividing the normal marginal distribution for coefficient i into Q bins and then computing the pmf.

The proposed ***simulated embedding*** scheme has pros and cons. On the one hand, it offers good empirical security for medium JPEG QFs (85 and 75) and for linear or close to linear demosaicking algorithms (bilinear or VNG, see Table 4.8 in sub-section 2.7.4). On the other hand, the estimation of the covariance matrix combined with its scaling (4.22) are only approximations that decrease the empirical security of the whole scheme, especially for high QFs.

We consequently derive in this chapter a closed-form of the covariance matrix $\mathbf{\Sigma}$ for bilinear demosaicking. This matrix is directly computed from the photo-site values of the RAW cover image and does not need to be scaled. This approach is validated in sub-section 3.2 by showing the results are equivalent to the ones obtained by ***simulated embedding*** at the photo-site level.

3.2 Results

3.2.1 Practical security

In this section, we evaluate the empirical security of NS in the JPEG domain for images acquired with a color sensor. A total of 4800 512×512 images were obtain by partitioning into non-overlapping patches 100 RAW images acquired at ISO_2 using the Z-CAM-E1 camera [30] to build the covers subset. In a likewise way the pseudo-stego subset of 4800 512×512 images is built from 100 RAW images acquired at ISO_1 and undergoes a switch [10] from $ISO_1 = 100$ to $ISO_2 = 200$. The parameters to perform the switch from ISO_1 to ISO_2 were set to $(a, b) = (1.15, -1150)$. The empirical security is evaluated as the minimal total classification error probability under equal priors, $P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}})$, with P_{FA} and P_{MD} standing for the false-alarm and missed detection rates. The JPEG images are steganalyzed with the DCTR features set [50] and the low-complexity linear classifier [27].

The following embedding schemes are compared:

- *Pseudo-embedding*, ***simulated embedding*** is performed at the photo-site level with the stego signal. These results can be considered as a baseline but do not correspond to any practical embedding scheme in the JPEG domain.
- *4 Lattices-scaling (**J-COV-NS-scaling**)*: The embedding pipeline uses the estimated covariance matrix to perform the sampling. The covariance matrix used is $\hat{\mathbf{\Sigma}}$ as explained in Section 3.
- *4 Lattices-analytic (**J-COV-NS**)*: The sampling mechanism is the same but we use the closed-form expression for the covariance matrix $\mathbf{\Sigma}$, detailed in Sub-Section 2.1, to simulate the stego signal in the DCT domain.
- *SI-UNIWARD*: For comparison with the current state of the art, all images have also been embedded using SI-UNIWARD [53] with the embedding rate set to 1 bit per nzAC DCT coefficient which corresponds to the maximal payload of this embedding scheme.

The capacity of both 4-Lattices implementations ranges from approximately 1 bpnzAC at $QF75$ to 2 bpnzAC at $QF100$ (see Figures 4.6). The results of these experiments are shown in Table 4.1. First, observe that there is no difference between generating the stego signal at the photo-sites or in the JPEG domain, which validates the fact that our statistical model in the JPEG domain is equivalent to the one at the photo-site level. Second, the closed-form of the covariance matrix provides a security gain w.r.t. the scheme proposed in [84], especially for high JPEG QFs. This is due to the fact that the covariance estimation proposed in [84] deals with blocks of constant photo-site values. This approximation is detrimental whenever ones wants to generate a high resolution stego signal. Note, however, that the computation of the covariance matrix is associated at a high computational burden since the Shur complement matrix needed to compute the conditional

probability distribution has to be evaluated for each block and not only once as in [84]. The comparison with SI-UNIWARD shows that cost-based SI-embedding is more detectable than NS embedding.[4]

In Table 4.6 we evaluate the sensitivity of our methods w.r.t. other popular demosaicking schemes and compare the security of NS implemented with the closed-form of the covariance matrix and with scaled estimated covariance as proposed in [84]. Cover images are developed using a speficic demosaicking method and stego images are generated either assuming bilinear demosaicking (the value on the left) or with the scaled estimated covariance matrix [84] as also explained in Section 2. While the closed-form of the covariance matrix offers the best performance for bilinear demosaicking, it cannot be used to model other demosaicking schemes, in which case it is better to estimate the covariance matrix and scale it, especially for high QFs.

JPEG QF	Pseudo embedding	**J-Cov-NS-scaling** [84]	SI-Uniward 1 bpnzac
100	40.2	13.9	0.0
95	40.9	30.3	0.4
85	41.9	39.8	12.3
75	41.3	40.4	24.8

Table 4.5: Empirical security (P_{E} in %) for different quality factors and embedding strategies on E1Base with bilinear demosaicking. DCTR features combined with regularized linear classifier are used for steganalysis.

3.2.2 Impact of the demosaicking algorithm used

One advantage of the covariance matrix estimation over explicit calculus is that we can blindly extract the covariance matrices for any demosaicking scheme, including development processes that are not publicly known. In this experiment, we evaluate the practical security for three demosaicking processes that are not linear (VNG, DCB, and AAHD). Note that the stego signal is generally not following a MGD for non-linear demosaicking. Table 4.6 presents the results of the demosaicking used on detectability w.r.t the JPEG QF.

The detectability varies quite significantly, e.g., compare the classifier errors for AAHD and VNG.

[4]Note that SI-UNIWARD only needs an uncompressed image while NS needs substantially more information – the RAW image.

Interestingly, when we compare the detectability with the statistical distribution of DCT modes (see 3.11 in Section 1.4 of Chapter 3), we can see that for demosaicking schemes with high detectability, such as AAHD, the empirical histogram and the Gaussian distribution with same mean and variance have very different distributions. On the contrary, VNG, which increases the detectability only slightly over Bi-linear demosaicking, is associated with an empirical histogram that is close to the Gaussian distribution. Consequently, one drawback of the proposed approach is that it cannot be used with demosaicking algorithms that have strong non-linearities with the exception of low QFs.

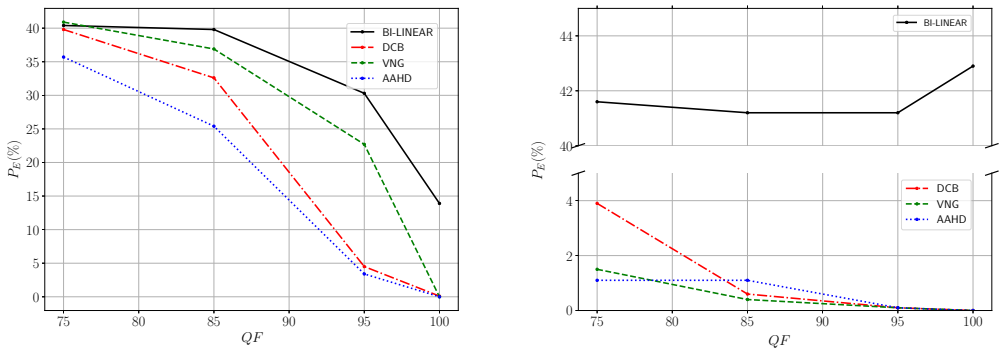
JPEG QF	Bi-linear	VNG	DCB	AAHD
100	13.9	0.0	0.1	0.0
95	30.3	22.7	4.5	3.4
85	39.8	36.9	32.6	25.4
75	40.4	40.9	39.8	35.7

Table 4.6: Empirical security (P_E (%) and sensitivity w.r.t the demosaicking algorithm used to develop cover images using scaled estimated covariance matrix (right) [84].

On the Figures 4.10 for (a) the embedding has been done using the scaling of the covariance matrix : the stego images have been sampled using the four lattices strategy and covers have been generated using Bi-linear, DCB, VNG and AAHD demosaicking methods. Meanwhile, the results obtained to produce Figures (b) have been obtained using the same setup except that the stego were sampled using the analytically computed covariance matrices. Compared to SI-UNIWARD (which is, as a reminder, an embedding scheme also using adjacent information), for the two approaches we observe a much higher empirical safety in both cases. It is also observed that this safety decreases with the quality factor for the approach using an estimated covariance matrix. To this one can add that both approach provides much more higher embedding capacities for the same empirical security.

3.2.3 Impact of the alphabet size

The impact of the alphabet size K is now measured in terms of practical security. Table (4.7) presents the results w.r.t the JPEG QF for the proposed implementation. Notice that using only ternary embedding makes the scheme very detectable for $QF95$. In fact, this setup required heptary embedding to offer detectability comparable to that of an infinite alphabet. On the contrary, ternary embedding is already sufficient for $QF75$.



(a) Scaled Covariance

(b) Computed Covariance

Figure 4.10: Investigation on the choice of the demosaicking algorithm for the generation of the covers. For (a) the stego were sampled using covariance scaling while for (b) the stegos were sampled using analytically computed covariance matrices.

QF / P_E in %	$K = 1$	$K = 2$	$K = 3$
95	2.8 / 3.5	19.3 / 23.6	29.5 / 39.3
85	39.8	39.8	39.8
75	40.4	40.4	40.4

Table 4.7: Practical security w.r.t. alphabet size K. Right and left values are respectively for 4-lattice embedding and pseudo-embedding when values differ. For $QF85$ and $QF75$, the 2 implementations give identical results.

4 Embedding in color JPEG domain in the framework of natural steganography

Since image steganography can be used to hide potentially sensitive messages inside mainstream image formats, it is curious to notice that most of academic contributions in steganography and steganalysis deals with image formats such as lossless raw coding (PGM, PPM) or gray-scale JPEG images.

In addition, if a steganographic implementation deals with the most popular image format on the Web, i.e. color JPEG images, it is usually performed by ignoring chromatic channels and only focusing on the luminance channel.

More precisely, when embedding is performed over digital color images in the pixel domain, it is most often performed independently on each component see [2], [44], [61], but more advanced implementations use a synchronization strategy to achieve more consistent integration changes throughout the system.

The general lack of solutions for color JPEG steganography and steganalysis is also due to the diversity of color JPEG images, not all of which are uniquely encoded. For example, since the color sub-sampling option also varies from one image to another, the size of the color components of a JPEG image depends on the acquisition device or development software. In addition, for chromatic channels, the quantization matrices used are generally different from the one used for luminance channel quantization.

Because this sampling format is relatively popular as shown in 4.8, taken from paper [87], where the authors used 10,000 images from the Flickr site in order to observe the popularity of the different chroma sub-sampling methods on this photo-sharing platform.

Chroma sub-sampling	4:4:4	4:2:2	4:2:0
Proportion	65%	8%	27%

Table 4.8: Statistics of chroma sub-sampling strategies for 10,000 "Explored" images downloaded at full resolution from Flickr.com.

4.1 Dependencies between the Y, C_b, C_r channels

As demonstrated at subsection 2.1 in chapter 1 for the luminance channel, by their definition the chrominance channels (C_b and C_r) are subject to the same phenomenons as the luminance channel. The calculation process to obtain the luminance and the chrominance channels from the R, G and B channels is as

follows:

$$\begin{bmatrix} Y \\ C_b \\ C_r \end{bmatrix} = \begin{bmatrix} 0.299 & 0.587 & 0.114 \\ -0.168736 & -0.331264 & 0.5 \\ 0.5 & -0.418688 & -0.081312 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix} + \underbrace{\begin{bmatrix} 0 \\ 2^{15} \\ 2^{15} \end{bmatrix}}_{\substack{\text{offset to} \\ \text{work on} \\ \text{uint16}}} . \quad (4.25)$$

By re-using essentially the same development process that the one used in the calculation of the covariance matrix in the case where we are only focusing on the luminance channel, it is possible to derive in the same way an analytical writing of the covariance matrix between the DCT blocks of the luminance channel and with the chrominance as well.

By picking up equation 3.37 from chapter 3 we can derive the equivalent including the Y,C_b and C_r channels:

$$\begin{aligned} \mathbf{s}_{d_Y} &= \mathbf{M}_Y \mathbf{s}_p = \mathbf{TPSD}_Y \cdot \mathbf{s}_p, \\ \mathbf{s}_{d_{C_b}} &= \mathbf{M}_{C_b} \mathbf{s}_p = \mathbf{TPSD}_{C_b} \cdot \mathbf{s}_p, \\ \mathbf{s}_{d_{C_r}} &= \mathbf{M}_{C_r} \mathbf{s}_p = \mathbf{TPSD}_{C_r} \cdot \mathbf{s}_p. \end{aligned} \quad (4.26)$$

So let's write: $\mathbf{TPS} = \hat{\mathbf{M}}$, and $\mathbf{I}_n \in \mathbb{R}^{n \times n}$ the identity matrix of size $n \times n$, thus:

$$\begin{bmatrix} \mathbf{s}_{d_Y} \\ \mathbf{s}_{d_{C_b}} \\ \mathbf{s}_{d_{C_r}} \end{bmatrix} = \underbrace{\begin{bmatrix} \hat{\mathbf{M}} \cdot \mathbf{D}_Y & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \hat{\mathbf{M}} \cdot \mathbf{D}_{C_b} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{M_{C_r}} \cdot \mathbf{D}_{C_r} \end{bmatrix}}_{\mathbf{M}} \cdot \underbrace{[\mathbf{s}_p, \mathbf{s}_p, \mathbf{s}_p]^t}_{\begin{bmatrix} \mathbf{I}_{(24+2)^2} \\ \mathbf{I}_{(24+2)^2} \\ \mathbf{I}_{(24+2)^2} \end{bmatrix} \cdot \mathbf{s}_p} . \quad (4.27)$$

But the matrix $\mathbf{M}$ could be written as:

$$\mathbf{M} = (\mathbf{I_3} \otimes \hat{\mathbf{M}}) \cdot (\mathbf{D}_Y \oplus \mathbf{D}_{C_b} \oplus \mathbf{D}_{C_r}) \cdot \begin{bmatrix} \mathbf{I}_{(24+2)^2} \\ \mathbf{I}_{(24+2)^2} \\ \mathbf{I}_{(24+2)^2} \end{bmatrix}, \quad (4.28)$$

Where $\otimes$ represent the Kronecker product, for $\mathbf{A} \in \mathbb{R}^{m \times n}$ and $\mathbf{X} \in \mathbb{R}^{p \times q}$

$$\mathbf{A} \otimes \mathbf{X} = \begin{pmatrix} a_{1,1} \cdot \mathbf{X} & \dots & a_{1,n} \cdot \mathbf{X} \\ \vdots & \ddots & \vdots \\ a_{m,1} \cdot \mathbf{X} & \dots & a_{m,n} \cdot \mathbf{X} \end{pmatrix} \in \mathbb{R}^{mp \times nq}, \quad (4.29)$$

And $\oplus$ is the direct sum, defined such as:

$$\mathbf{A} \oplus \mathbf{X} = \begin{bmatrix} \mathbf{A} & \mathbf{0} \\ \mathbf{0} & \mathbf{X} \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ a_{m1} & \cdots & a_{mn} & 0 & \cdots & 0 \\ 0 & \cdots & 0 & x_{11} & \cdots & x_{1q} \\ \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & 0 & x_{p1} & \cdots & x_{pq} \end{bmatrix} \tag{4.30}$$

The development pipeline can be then explicitly formulated as

$$\begin{bmatrix} \mathbf{s}_{d_Y} \\ \mathbf{s}_{d_{C_b}} \\ \mathbf{s}_{d_{C_r}} \end{bmatrix} = \mathbf{M} \cdot \mathbf{y}. \tag{4.31}$$

and the covariance matrix is computed as:

$$\mathbf{\Sigma}_d \ = \mathbf{M}\, \mathbb{E}\left[\mathbf{S}_p\, \mathbf{s}_p^t\right]\, \mathbf{M}^t. \tag{4.32}$$

Note that for a uniform constant RAW image defined by $\mu = \text{const.}$ (i.e., $\mathbb{E}\left[\mathbf{s}_d \cdot \mathbf{s}_d^t\right] \propto \mathbf{I}$), we obtain $\mathbf{\Sigma}_d \propto \mathbf{M}\mathbf{M}^{\mathbf{T}}$.

Using this formula it is possible to exhibit the inter-channels correlations for a single block using a covariance matrix of DCT coefficients of a color sensor with bilinear demosaicking for an i.i.d signal. This cross-correlation dependencies are depicted on Figures 4.11 where the correlation values are thresholded for visualization purposes.

It is important to note here that the intra-correlations we observed in chapter 3 are comparable in magnitudes to the cross-channel-correlations. Therefore, it can be quickly argued here that the importance of preserving these dependencies is crucial to ensure the consistency of the statistical model of sensor noise in the DCT domain. For this purpose we have adapted our four lattice approach to be able to sample realizations of sensor noise in the DCT domain preserving correlations: intra-block, inter-block and inter-channel. The easiest solution is to keep the algorithm and reorder the coefficients so that we can continue to use 4-lattices but with three times as many coefficients in each of them. Thus we choose to group by blocks the coefficients of the same frequency on the different channels. This produces an ordered covariance matrix as shown in Figures 4.13.

The result of this operation on the previously displayed covariance matrix 4.12 is shown in Figures 4.13. This display 4.13 does not bring much understanding on the dependencies, however it shows the very strong correlations that can exist between modes of the same block, of the same frequency but of different channels.

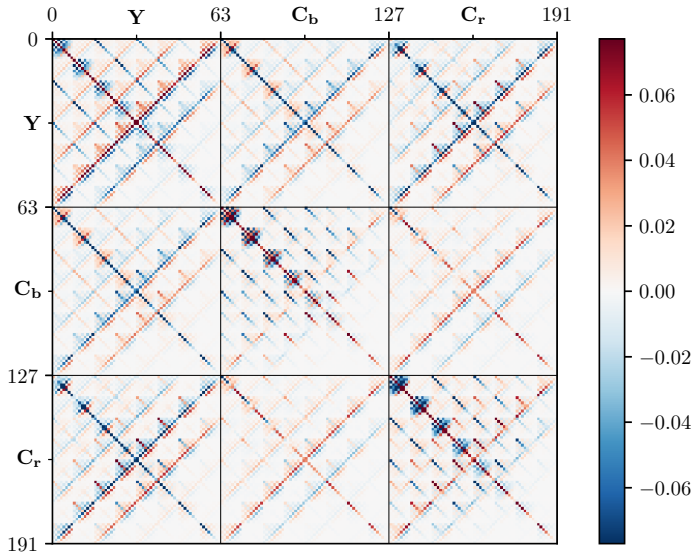


Figure 4.11: 192×192 covariance matrix of DCT coefficients of channels Y, C_b, C_r of a color sensor with bilinear demosaicking for an i.i.d signal showing the impact of demosaicking on for cross-channel correlations between intra-block DCT coefficients. Colors dynamics have been clipped for visualization purposes.

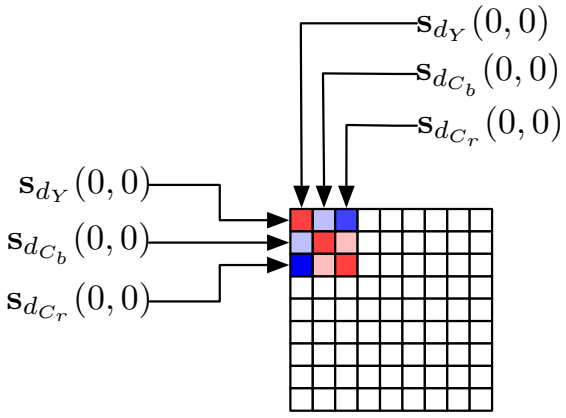


Figure 4.12: Reordering of covariance matrix to cluster by triplets the modes of the same frequency between the different channels for each block.

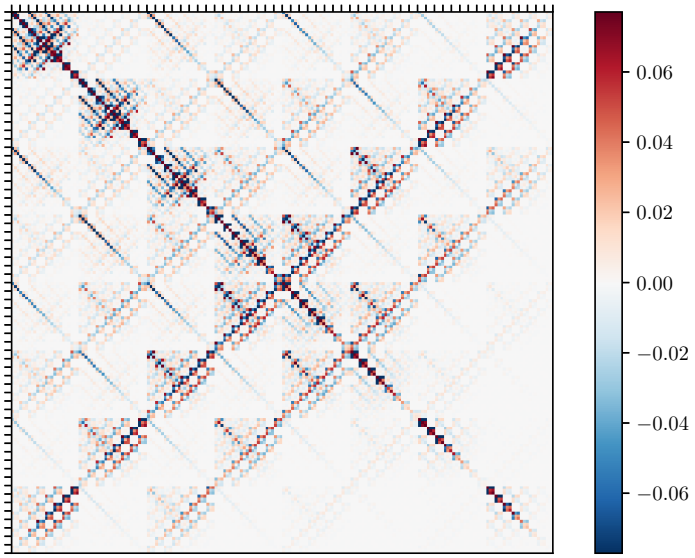


Figure 4.13: Reordered 192×192 covariance matrix of DCT coefficients of channels Y, C_b, C_r of a color sensor with bilinear demosaicking for an i.i.d signal showing the impact of demosaicking on for cross-channel correlations between intra-block DCT coefficients. Colors dynamics have been clipped for visualization purposes.

4.2 Results

Through this section, to study the impact of these cross-correlations we performed an embedding approach preserving the intra-block and inter-block correlations but not preserving the inter-channel dependencies. This method lead to a block diagonal covariance matrix (see Figures Figures 4.14) for a single block using a covariance matrix of DCT coefficients of a color sensor with bilinear demosaicking for an i.i.d signal. This is due to the fact that there are no longer any cross-channel dependencies.

To these strategies we have added a slightly modified version (here the C_b and C_r channels have been added) of the "pseudo-embedding" algorithm mentioned in Section 2 of Chapter 4 for the needs of the experiment, this strategy will allow us to have a baseline with regard to the empirical security values reached during steganalysis.

The evaluation strategies are the same as for the 4-Lattices approach in the monochromatic domain with the difference that we will use both the concatenated DCTR and the concatenated GFR as the features set, so the dimensionality of the images in the features domain is here 3 times the dimensionality of the gray-scales images in the features domain. But, to the best of our knowledges this features

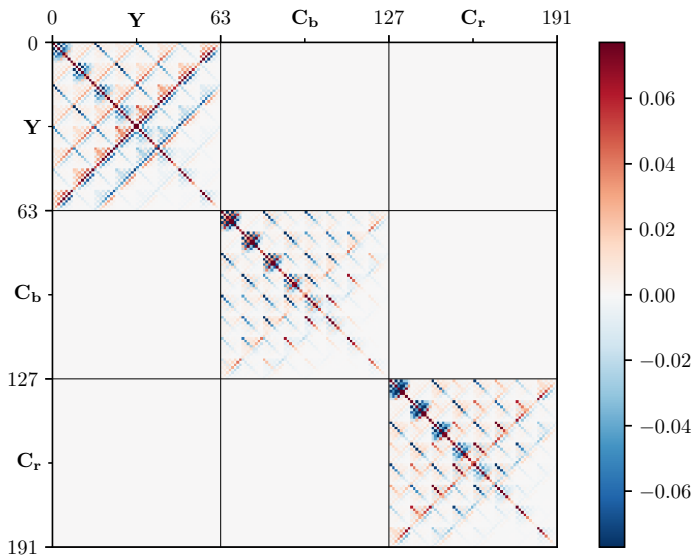


Figure 4.14: Covariance matrix intra-block DCT coefficients without cross-channels dependencies. Colors dynamics have been clipped for visualization purposes.

sets does not propose by their nature (concatenation) the possible exploitation of correlations between the different channels as the SCRMQ1 features set [46]. However, it seems that the discrepancies between the different channels can be captured discrepancies by C-DCTR and C-GFR. In order to take into account cross-channel statistics, we believe that differential characteristics between the characteristics of these channels should be introduced.

The results of these experiments are recorded in the following Table 4.9 for C-DCTR and in Table 4.10 for C-GFR:

Strategies / QF	Pseudo	4-Lattices Without cross-correlation
100	38.278	1.315
95	37.679	13.686
85	39.059	35.105
75	39.781	38.68

Table 4.9: empirical security (P_{E} in %) for different quality factors and embedding strategies on E1Base with bilinear demosaicking in Y, C_b, C_r domain. Concatenated DCTR features over the different channels combined with regularized linear classifier are used for steganalysis.

Strategies / QF	Pseudo	4-Lattices Without cross-correlation
100	38.911	4.36
95	37.22	12.10
85	37.39	33.06
75	38.06	36.57

Table 4.10: empirical security (P_{E} in %) for different quality factors and embedding strategies on E1Base with bilinear demosaicking in Y, C_b, C_r domain. Concatenated GFR features over the different channels combined with regularized linear classifier are used for steganalysis.

The contribution of the present section is mainly two-fold, firstly, the concatenation of features on the different channels allows the classifier to take into account the correlations between the different channels. This observation echoes to [2] where it was observed that the integration of these correlations between channels indeed allows a slight increase in the performance of steganalysis of stego color images. Secondly, within the framework of "pseudo-embedding", despite the use of the same benchmark for color and monochrome steganalysis, it is observed that the empirical safety is lower for color images. This could be due to the fact that the steganalyst has in the color case three times more coefficients to train his classifier and that therefore he would be more efficient in the covers/stegos classification as it has been stated by [3] back to 1996:

> "Thanks to the Central Limit Theorem, the more covertext we give the warden, the better he will be able to estimate its statistics, and so the smaller the rate at which the steganographer will be able to tweak bits safely. The rate might even tend to zero..."

Apart from that, this was later slightly more formalized by Andrew Ker in [57] and [54].

Despite their absence in Tables 4.9 and 4.10, we conducted experiments to take into account inter-channel correlations in addition to intra/inter-correlation, but the results were not consistent with our expectations. We suspect that this is due to an implementation flaw.

4.3 Complexity consideration

This embedding algorithm is computationally expensive since the complexity of computing the conditional distribution increases as the complexity of the Cholesky decomposition of the covariance matrix, i.e., as $\mathcal{O}(n^3)$ where $n \leq i \times 64$, where

$i = 1$ for $\mathbf{\Lambda}_1$, $i = 5$ for $\mathbf{\Lambda}_2$ and $\mathbf{\Lambda}_3$, and $i = 9$ for $\mathbf{\Lambda}_4$ (see Figures 4.2). On a 3.5 GHz Intel Core i7, our python implementation of ***simulated embedding*** is executed at 4000 block/s for blocks belonging to $\mathbf{\Lambda}_1$, 30 blocks/s for $\mathbf{\Lambda}_2$, 30 blocks/s for $\mathbf{\Lambda}_3$ and 10 blocks/s for $\mathbf{\Lambda}_4$. A 512×512 stego image is generated in approximately 171s without using hyper-threading. This value is 3 times higher when generating a color image with decorated channels, and $3{\times}9$ times higher when correlating channels leading to an embedding duration of more than 1 hour for a 512×512 image. It will have required 25000 hours of computation on the IDRIS cluster (Le supercalculateur Jean Zay) to generate the 10000 stego images.

5 Conclusions of the chapter

This chapter including the previous one (chapter 3) draws important conclusions both in image processing and image steganography. By deriving the covariance matrix of the random vector of stego signal components in the DCT domain, we have shown that for this basic development pipeline there are medium range correlations between DCT coefficients, and that for a given coefficient, it is correlated with the coefficients belonging to the same blocks, but also with the coefficients belonging to 8-connected blocks.

Previous works on the estimation of the covariance matrix were conducted for denoising applications using non-local Bayesian estimation [65], but to the best of our knowledge, it is the first time that an analytical expression of the covariance matrix is derived in the DCT domain (i.e. Eq. (3.37) and (3.38)), exhibiting intra-block, inter-block correlations and later on cross-channels correlations.

In the monochromatic domain, the derivation of the covariance matrix enables to generate a stego signal that mimics the photonic noise in the DCT domain and consequently to achieve high practical security ($P_E \geq 40\%$ for DCTR features set) while reaching high capacity (> 2 bpnzAC).

In order to preserve the joint Gaussian distribution after embedding in the quantized DCT domain, both the **J-Cov-NS** and **J-Cov-NS-scaling** embedding schemes need to use a large number of lattices (4×64) where conditional probability mass functions are derived for each lattice.

Our experimental analysis shows that for high JPEG QF, being able to perform conditioning is essential to achieve high practical security. A similar synchronization strategy was also adopted for adaptive schemes using empirical costs in [69] and [83], this approach is explained in the next chapter 5.

Chapter 5

Synchronization of embedding changes for cost-based JPEG steganography

This chapter proposes to use the statistical analysis of the correlation between DCT coefficients to design a new synchronization strategy that can be used for cost-based steganographic schemes in the JPEG domain.

It relies on the analysis of the covariance matrix after a pipeline similar to the BOSSBase development performed in Chapter 3 section 3.6, and belongs to the family of synchronisation schemes presented in Chapter 1, section 4.6.

Firstly, we convert the empirical costs associated to one each coefficient into a Gaussian distribution whose variance is directly computed from the embedding costs. Secondly we derive conditional Gaussian distributions from a multivariate distribution considering only the correlated coefficients which have been already modified by the embedding scheme. This covariance matrix takes into account both the correlations exhibited by the analysis of the covariance matrix and the variance derived from the costs.

1 Main ideas

The present chapter proposes a novel method that combines the advantages of both prior works [69, 93]. On one hand, the method can be easily applied in practice in the sense that, as proposed in [69], we use a cost map derived from a classical JPEG embedding scheme such as UERD [47] or J-UNIWARD [53]. On the other hand, the main contribution of the proposed method relies on its statistically-based foundation since, as in [93], it exploits the correlations induced by the development pipeline to synchronize the embedding changes. However,

contrary to [93], the proposed synchronization method can be applied with any cost based steganographic scheme. The main idea proposed in this chapter is to leverage the natural correlations induced by the development pipeline on the photonic noise to perform synchronization in the JPEG domain.

Based on the observations made in Section 3.6 of this manuscript, we derived an embedding scheme designed to incorporate the correlations induced by image processing into the modifications made to the images during cover generation. We then switch from an additive insertion cost (obtained from the cost maps of heuristic algorithms such as J-UNIWARD or UERD) to a non-additive cost because we are now considering the values of the previous modifications in order to make the forthcoming ones.

2 Embedding scheme

We detail now how we can leverage both the analysis of the covariance matrix presented in section 3.2 and the group decomposition presented in 3.6 to enable to synchronization of embedding changes for cost-based embedding schemes.

Figures 5.1 summarizes the different mandatories steps necessary to perform embedding which can be decomposed into five steps:

1. The computation of the correlation coefficients and the correlation matrix, as presented in section 3.2.
2. The decomposition of the image into 8 groups as presented in 3.6.
3. The computation of a covariance matrix using both the costs derived from the additive steganographic scheme and the correlation matrix (computed at the first step). In order to do so, we convert empirical costs into Gaussian distributions. This can be justified by the fact that in order to leverage the covariance matrix of the sensor noise, we need to model the stego signal by a multivariate Gaussian distribution since it is the only distribution that can be defined only by its expectation and its covariance. The derivation of variances from costs is detailed in section 2.1.
4. The computation of the conditional embedding probabilities which take into account both the correlations between DCT coefficients and the modifications done on the previous groups. This is detailed in section 2.3.
5. The modification of the coefficients to obtain the stego image. This is detailed in section 2.4.

2.1 From costs to Gaussian distributions

Without loss of generality, we assume that the developed steganographic scheme uses ternary embedding. For a coefficient of coordinates (i, j) into a 8×8 DCT

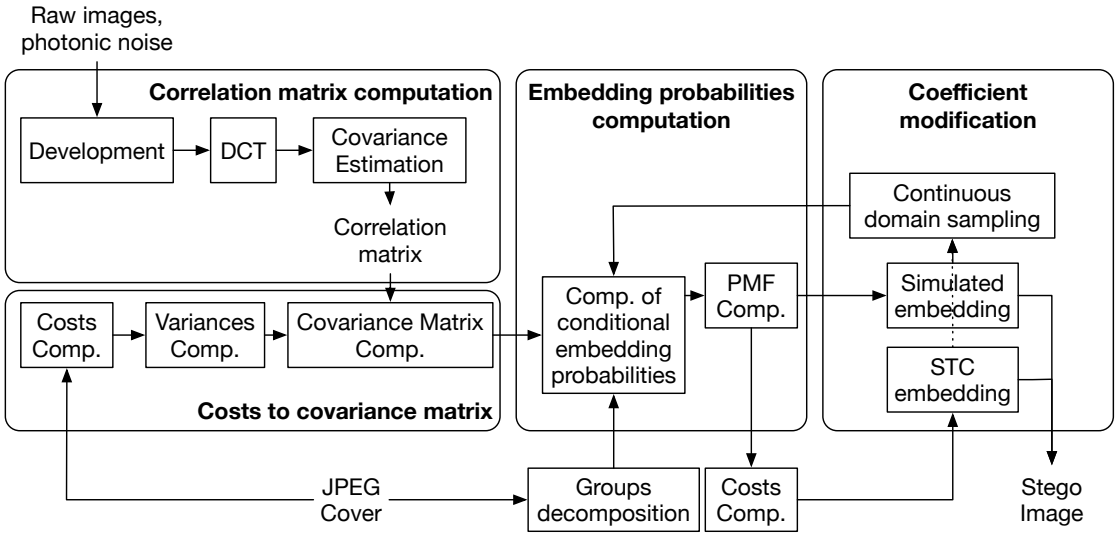


Figure 5.1: Overview of the embedding scheme.

block, we assume that the underlying unquantized stego signal is associated with an Normal distribution with zero mean and a variance $\sigma_{i,j}^2$, i.e. $S_{i,j} \sim \mathcal{N}\left(0, \sigma_{i,j}^2\right)$. As explained below, the variance is determined w.r.t both the costs computed by an heuristic algorithm (UERD or J-UNIWARD here), and to the payload size $\mathbf{m}$.

For each coefficient (i, j) we can compute the triplet of costs $(\rho_{i,j}^{-1}, \rho_{i,j}^{0}, \rho_{i,j}^{+1})$ respectively associated to the embedding changes $-1, 0, +1$. Since we use non side-informed schemes, we also assume that $\rho_{i,j}^{-1} = \rho_{i,j}^{+1}$.

We can convert the costs into embedding probabilities using Lagrangian optimization [37] by using the formula:

$$P_{i,j}(k) = \frac{\exp\left(-\lambda \rho_{i,j}^{k}\right)}{\exp\left(-\lambda \rho_{i,j}^{0}\right) + \exp\left(-\lambda \rho_{i,j}^{+1}\right) + \exp\left(-\lambda \rho_{i,j}^{-1}\right)}, \tag{5.1}$$

with $k \in -1, 0, +1$, and λ following the payload constraint.

Denoting $q_{i,j}$ the JPEG quantization step associated to coefficient (i, j), we now assume that the embedding probabilities correspond to the probabilities of a quantized Gaussian distribution using three quantization bins, respectively $]-\infty, -q_{i,j}/2]$, $]-q_{i,j}/2, q_{i,j}/2]$, $]q_{i,j}/2, +\infty]$ for $-1, 0, +1$. Since

$$P_{i,j}(-1) = \frac{1}{2}\mathrm{erf}\left(-\frac{q_{i,j}}{2\sqrt{2}\sigma_{i,j}}\right), \tag{5.2}$$

and $2P_{i,j}(-1) + P_{i,j}(0) = 1$, the relation between $\sigma_{i,j}^2$ and the embedding probabilities is then given by:

$$\sigma_{i,j}^2 = \frac{q_{i,j}^2}{8\left(\mathrm{erf}^{-1}\left(P_{i,j}\left(0\right)\right)\right)^2}. \tag{5.3}$$

2.2 Construction of the covariance matrix

The estimation of the covariance matrix presented in Chapter 3, section 3.2 is performed in order to highlight correlations between DCT component of the sensor noise and to try to mimic them during the embedding.

However, in order to be invariant to the noise power which depends of various parameters such as the sensor model or the ISO settings, we can convert the *covariance* matrix into a *correlation* matrix, where each diagonal terms equals 1 and each off-diagonal term is divided by $\sigma_i \sigma_j$.

Practically, since each term of the empirical covariance matrix is defined as:

$$\Sigma_{i,j} = \frac{1}{N} \sum_{k=1}^{N} (C_i(k) - \bar{C_i}])(C_j(k) - \bar{C_j}]), \tag{5.4}$$

(where $\bar{C_i}$ is the empirical mean of coefficient C_i), each term of the correlation matrix is consequently defined as:

$$\xi_{i,j} = \frac{\Sigma_{i,j}}{\sqrt{\Sigma_{i,i}\Sigma_{j,j}}}. \tag{5.5}$$

Note that after this normalization, correlation coefficients which are not close to zero are rather small. For our experiments correlation coefficients for two distinct DCT modes belong to the range $[0.03; 0.07]$.

The covariance matrix $\tilde{\mathbf{\Sigma}}$ used during the embedding is sequentially built for each DCT coefficient of each group in order to take into account the embedding changes of correlated coefficient that have already been made on the previous groups. Its size is consequently $(K+1) \times (K+1)$, with K given in Table 3.2.

The diagonal terms of $\tilde{\mathbf{\Sigma}}$ are given by (5.3) and its off-diagonal terms take into account the correlation coefficients $\xi_{i,j}$ estimated using (5.5).

More specifically for a given mode, the covariance matrix is built using the variances $\{\sigma_1^2, \ldots, \sigma_K^2\}$ of the K correlated coefficients that have been already modified during the embedding. Theses variances are then weighted by the inter-correlations coefficients ξ associated to these coefficients using (5.5). The resulting covariance matrix $\tilde{\mathbf{\Sigma}}$ is given by:

$$\tilde{\mathbf{\Sigma}} = \begin{bmatrix} \sigma_1^2 & \xi_{1,2}\sigma_1\sigma_2 & \cdots & & \xi_{1,m}\sigma_1\sigma_{K+1} \\ \xi_{1,2}\sigma_1\sigma_2 & \sigma_2^2 & & & \\ \vdots & & \ddots & & \vdots \\ & & & \sigma_K^2 & \\ \xi_{1,K+1}\sigma_1\sigma_m & & \cdots & & \sigma_{K+1}^2 \end{bmatrix}, \tag{5.6}$$

$$\tilde{\mathbf{\Sigma}} \doteq \begin{bmatrix} \tilde{\mathbf{\Sigma}}_d & \tilde{\mathbf{\Sigma}}_c \\ \tilde{\mathbf{\Sigma}}_r & \sigma_{K+1}^2 \end{bmatrix}, \tag{5.7}$$

where $\tilde{\mathbf{\Sigma}}_d$ is the $(K \times K)$ matrix with the $(K \times K)$ first entries of $\tilde{\mathbf{\Sigma}}$, $\tilde{\mathbf{\Sigma}}_c$ is the $(K \times 1)$ matrix with the K first entries of the last column of $\tilde{\mathbf{\Sigma}}$ and $\tilde{\mathbf{\Sigma}}_r$ is the $(1 \times K)$ matrix with the K first entries of the last row of $\tilde{\mathbf{\Sigma}}$.

2.3 Computation of embedding probabilities

By computing the Schur complement (in the same way as described in the subsection 2.2) we can derive the conditional pdf of $C_{K+1}|c_1, \ldots, c_K$ distributed as $\mathcal{N}(\tilde{\mu}, \tilde{\sigma}^2)$, with:

$$\tilde{\mu} = \tilde{\mathbf{\Sigma}}_r \tilde{\mathbf{\Sigma}}_d^{-1} [c_K, \ldots, c_1]^T, \tag{5.8}$$

$$\tilde{\sigma}^2 = \sigma_{K+1}^2 - \tilde{\mathbf{\Sigma}}_r \tilde{\mathbf{\Sigma}}_d^{-1} \tilde{\mathbf{\Sigma}}_c. \tag{5.9}$$

Note that because of conditioning (and of synchronization), the mean of the Gaussian distribution is not anymore equal to zero. We can afterward compute the pmf by again integration over the 3 intervals $]-\infty, -q_{i,j}/2]$, $]-q_{i,j}/2, q_{i,j}/2]$, $]q_{i,j}/2, +\infty]$ for $-1, 0, +1$.

2.4 Coefficient modification

Once the pmf is computed, either we sample from it, or we convert the probabilities to costs using the relation $\rho_{i,j}^k = \log(p_{i,j}^0 / p_{i,j}^k)$, and use a STC. Moreover, in order to compute (5.9), we need to draw samples $c_{m-1}, \ldots, c_1$, which correspond to the embedding changes performed on the K DCT coefficients belonging to the previous groups, which already are carrying a portion of the payload. This can be done for example by rejection sampling, i.e. by sampling over the Gaussian distributions until each sample belongs to the interval corresponding to the right embedding change.

3 Results

3.1 Database development

In order to leverage the correlations induced by the development pipeline, we explain in this section the development pipeline used to develop the raw images of BOSSBase. Since this database is composed of images coming from different cameras, the sensors have different sizes (from CR2 of size 2602×3906, to DNG of size 3472×5216, NEF of size 2014×3039, and PEF files of size 3124×4688), thus to be able to have the same down-sampling factor for each image it is important to find the minimum length or width dimension for all the images. As a result, for each image we we developed the image using bi-linear demosaicking, luminance

averaging, bilinear downscaling, and then performed a centered crop of width and height equal to $l_{min} = 2014$ before performing JPEG compression to build our BOSSBase-SD (Same Dimensions). Note that except for the crop operation and the demosaicking and down-sampling kernels, this database is very similar to the BOSSBase database.

3.2 Benchmark setup

The empirical security is evaluated as the minimal total classification error probability under equal priors, $P_{\mathrm{E}} = \min_{P_{\mathrm{FA}}} \frac{1}{2}(P_{\mathrm{FA}} + P_{\mathrm{MD}})$, with P_{FA} and P_{MD} standing for the false-alarm and missed detection rates. The JPEG images are steganalyzed with the DCTR features set [50] and the low-complexity linear classifier [27], training and testing sets both equal to 2×5000, the P_E is averaged after 10 random splits on training and testing sets.

The presented adaptations, named *Cov-J-UNIWARD* and *Cov-UERD* (which use respectively the costs computed by J-UNIWARD and UERD) are compared with *J-UNIWARD* and *UERD*. However, since the synchronized version of theses algorithms use conditioning, the achievable entropy is slightly attenuated of about 1% as can be seen on Table 5.1. Consequently, in order to make an fair comparison we have compared J-UNIWARD and UERD to their synchronized versions by using the payload size computed from *Cov-J-UNIWARD* and *Cov-UERD* respectively to J-UNIWARD and UERD (see also Figures 5.2). This operation is performed over the whole image base for $\mathrm{H_{in}}$ (bits/nzAC) $\in \{0.1, 0.2, \ldots, 1.0\}$.

Note also that the embedding is performed by considering quantization steps of size 1, which correspond to a targeted JPEG quality factor of 100. Extensive test with appropriate values of q are left for future researches.

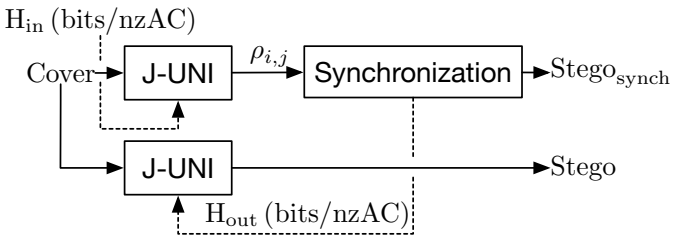


Figure 5.2: Embedding setup to ensure that the stego image carry the payload in the case of a J-UNIWARD embedding.

Targeted	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
True	0.099	0.198	0.298	0.397	0.496	0.595	0.694	0.794	0.893	0.992

Table 5.1: Targeted payload vs True embedding rate in pbnzac due to synchronization for one sample image.

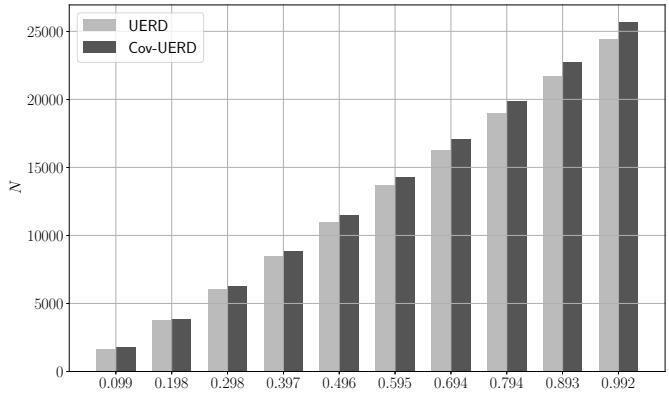


Figure 5.3: Comparison between the number of modifications for UERD and Cov-UERD for same embedding rates.

$\mathbf{\Lambda}_0$	$\mathbf{\Lambda}_1$	$\mathbf{\Lambda}_2$	$\mathbf{\Lambda}_3$	$\mathbf{\Lambda}_4$	$\mathbf{\Lambda}_5$	$\mathbf{\Lambda}_6$	$\mathbf{\Lambda}_7$
1.94	0.95	0.92	1.52	1.82	0.88	0.86	1.43

Table 5.2: Average entropy by coefficients ($\times 10^{-2}$) over the 8 groups for $QF95$, for a targeted payload of 0.3 bpnzAC on one sample image.

P_E (%) / JPEG QF	Cov-UERD	UERD	Cov-JUNI	JUNI
75	23.341 ± 0.116	20.368 ± 0.08	21.089 ± 0.104	21.606 ± 0.059
85	29.167 ± 0.145	24.896 ± 0.11	27.62 ± 0.136	27.269 ± 0.109
95	42.442 ± 0.242	35.64 ± 0.11	45.282 ± 0.113	37.205 ± 0.045
100	27.797 ± 0.094	27.351 ± 0.069	33.129 ± 0.08	31.733 ± 0.095

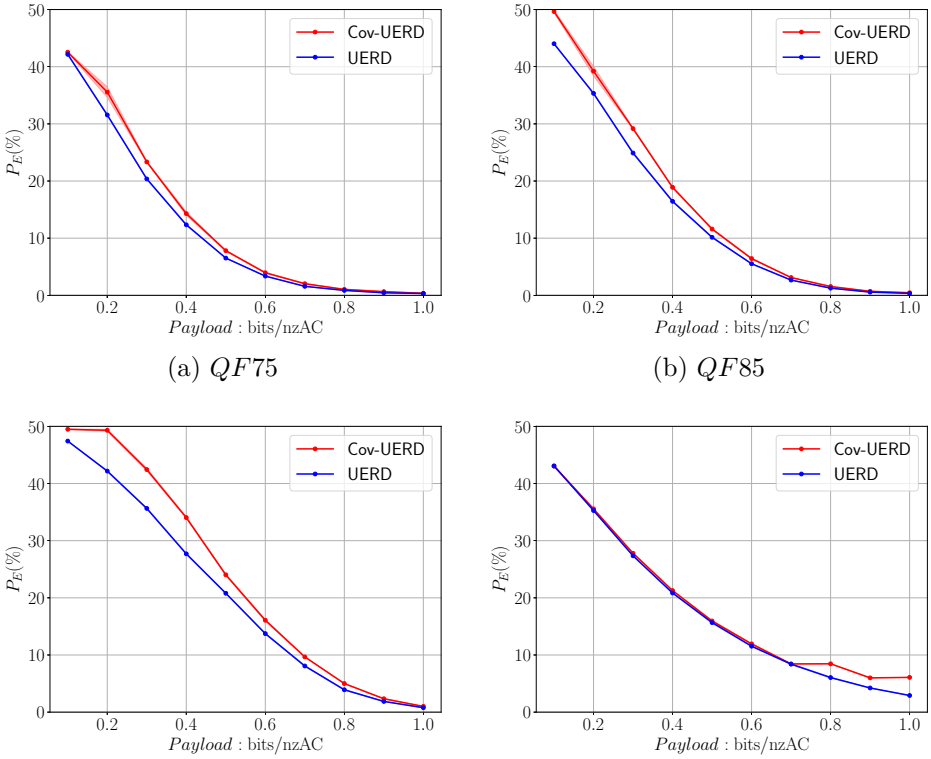
Table 5.3: Average empirical security (P_{E} in %) and associated standard deviation over 10 runs for different quality factors and embedding strategies on BOSSBase SD with bilinear demosaicking, and downscaling but the same payload of 0.28 bpnzac. DCTR features combined with regularized linear classifier are used for steganalysis.

3.3 Comparison with UERD and J-UNIWARD

Results for both schemes are presented in Table 5.3 for an embedding rate of 0.28 bpnzac and for a range of embedding rates withing $[0, 1.0]$ on Figuress 5.5 and 5.4. Several observations can be made: Firstly, the JPEG quality factor offering the best performance improvement over the classical schemes is $QF95$ for both schemes.

Secondly, this improvement can be substantial with a maximum gain of around 7% for both schemes at 0.28 pbnzac. On the other hand for high embedding rates (i.e. ≥ 0.4 pbnzac), the impact of synchronization can be either negative for J-UNIWARD, or nonexistent for UERD. This can be due to the fact that the model of the stego signal for these additive schemes might be too different with the model of the stego signal of the sensor noise, which makes the synchronization either useless or detrimental.

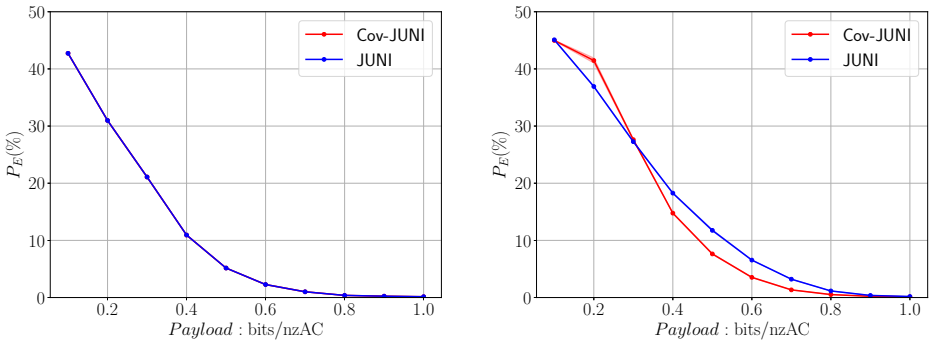
Finally, the costs provided by UERD seem on average to be more suited to the synchronization procedure than J-UNIWARD, which at $QF75$ for example, does not show any gain.



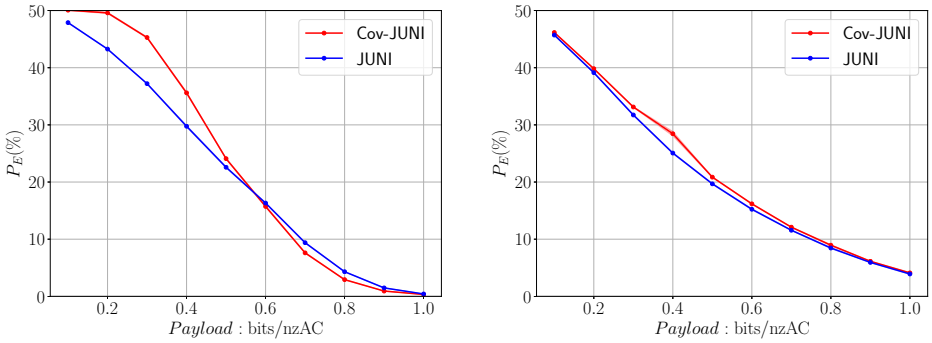
(a) $QF75$ (b) $QF85$

(c) $QF95$ (d) $QF100$

Figure 5.4: UERD and its synchronized version $QF \in \{75, 95, 85, 100\}$ for respectively (a), (b), (c) and (d).



(a) $QF75$ (b) $QF85$



(c) $QF95$ (d) $QF100$

Figure 5.5: J-UNIWARD and its synchronized version for $QF \in \{75, 95, 85, 100\}$ for respectively (a), (b), (c) and (d).

3.4 Effects of synchronization

The synchronization w.r.t. previous embedding changes on previous groups naturally induces fluctuations in the final embedding probabilities. One can observe on Figures 5.6 that if the same embedding changes are performed on DCT coefficients belonging to group $\mathbf{\Lambda_0}$ between the synchronized and the non-synchronized version of UERD, the embedding probabilities on other coefficients belonging to $\{\mathbf{\Lambda_1}, \ldots, \mathbf{\Lambda_7}\}$ can undergo important bias, going up to ± 0.15 for several coefficients.

Table 5.2 presents the average entropy for one sample image for each group at $QF95$. One can notice a small decrease of the entropy between $\mathbf{\Lambda_k}$ and $\mathbf{\Lambda_{k+4}}$ ($k \in \{0, \ldots, 3\}$) which corresponds to same DCT modes on two adjacent diagonal blocks, and which is due to synchronization. This behavior can be explained by the fact that for two random coefficients (C_1, C_2) coding the same DCT mode belonging to two vertically or horizontally connected blocks, $H(C_2|C_1) \leq H(C_2)$.

Figures 5.3 compare the number of embedding changes between UERD and Cov-UERD for same embedding rates and one sample image. Logically the proposed synchronization procedure induces more embedding changes, but at the same time decreases the detectability for small embedding rates.

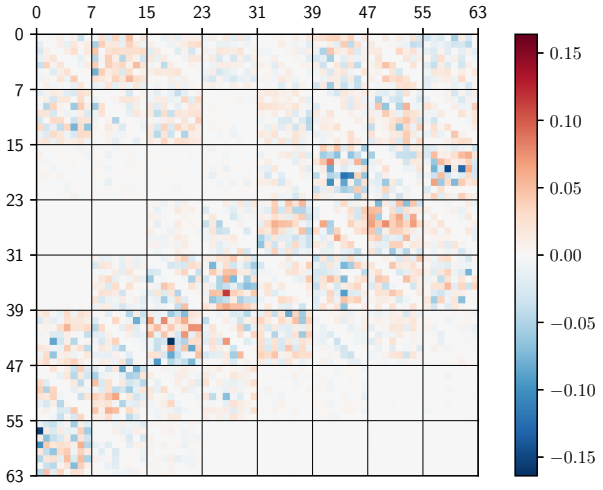


Figure 5.6: Difference of probabilities map to sample a +1 between UERD and Cov-UERD for a sample image (cropped to a 64×64 array), $QF100$, 0.48 bpnzAC. Identical embedding changes for the two schemes have been performed on coefficients belonging to group Λ_0.

3.5 Complexity

This embedding algorithm is computationally expensive because the complexity of computing the conditional distribution increases with the complexity of the Cholesky decomposition of the covariance matrix, i.e., as $\mathcal{O}(n^3)$ where $n = K + 1$, which depend of which group the considered mode belongs: $n = 1$ for $m \in \mathbf{\Lambda}_0$, $n = 3$ for $m \in \mathbf{\Lambda}_3$ and $n = 39$ for $m \in \mathbf{\Lambda}_7$. On a 1.6 GHz Intel Core i5, our python implementation of simulated embedding on a 512×512 image is performed in 1min 46s while an UERD simulated embedding takes 2 seconds.

4 Conclusions of the chapter

We have proposed a synchronization mechanism for JPEG steganography that can be used for classical additive cost-based embedding schemes. The synchronization is done by leveraging the correlations between DCT coefficients after the development

from RAW to DCT of an image composed of photonic noise. The embedding scheme requires the use of 8 groups of disjoint DCT coefficients in order to synchronize one coefficient with potentially 6 coefficients of the same block and 24 coefficients belonging to adjacent horizontal or diagonal blocks. The correlations are taken into account by converting classical heuristic costs into marginal Gaussian distributions, and then building a multivariate Gaussian distribution associated with a covariance matrix that takes into the correlation between DCT coefficients.

Our encouraging results show that this methods enables to increase the practical security by around 7% for an embedding rate of 0.28 bpnzac at $QF95$.

Chapter 6

Conclusions and perspectives

It appears from the state of the art presented in Chapter 1 that the use of steganography techniques by modifying the medium of coverage is often the most convenient choice for the digital concealment of secret information. Among the various insertion methods presented, we focused on steganographic methods preserving the model, and more particularly on Natural Steganography. The key idea of this is to propose a steganographic scheme where the message embedding will be equivalent to switching from one source $\mathcal{S}_1$ to another $\mathcal{S}_2$. This is practically achieved by designing an embedding such that, when applied on a sample of $\mathcal{S}_1$, this sample mimics the statistical properties of one sample that would have been produced $\mathcal{S}_2$.

We were also investigating a mechanism for preserving correlations between DCT coefficients. Among the various embedding methods presented in this thesis (see Chapter 1), we have also addressed adaptive steganography methods. The purpose of these methods is to modify the host medium in order to embed the secret message, while minimizing the impact of the embedding. For this purpose, current adaptive methods use a costs map, which associates to each element of the "cover" a modification cost, reflecting the impact on detectability related to this modification. As this cost map is crucial for the insertion of the message, it must reflect as well as possible the statistical detectability of the modified elements, which is not naturally translated by cost maps (often based on heuristic considerations).

The contributions of this thesis to the field of steganography are twofold:

- First of all, **a new approach has been developed allowing the sampling of modifications in the JPEG domain by guaranteeing their undetectability** in the framework of Natural Steganography. We have proposed in this thesis a new methodology for JPEG steganography based on the principle of cover-source switching, i.e. the fact that the embedding should mimic the transition from one cover-source to another in the JPEG domain. The presented schemes use the sensor

noise to model each source, and the embedding is performed by sampling a suited stego signal which enables the transition between the first and the second cover source's. This method in the case where the dependencies are analytically computed, provides good undetectability performances while proposing high embedding rates. However, it has to use RAW images as inputs and it uses an embedding step which is very costly from a computational point of view.

- Secondly, **we derived from the previous approach a mechanism to increase the empirical security of cost-based embedding schemes** whose distortion function is additive. The algorithm proposes a synchronization mechanism for JPEG steganography that can be used for classical additive cost-based embedding schemes. The proposed embedding algorithm makes it thus possible to switch from an additive to a non-additive embedding mechanism whose future modifications are shaped by the previous ones in order to preserve the correlations that exist between the DCT coefficients. This is achieved by employing synchronization mechanism that use on one hand, the initial costs provided by the additive embedding scheme, and on the other hand, a covariance matrix. This is done by leveraging the correlations between DCT coefficients after the development from RAW to DCT of an image composed of photonic noise. The embedding scheme requires the use of 8 lattices of disjoint DCT coefficients in order to synchronize one coefficient with potentially 6 coefficients of the same block and 24 coefficients belonging to adjacent horizontal or diagonal blocks. The correlations are taken into account by turning classical heuristic costs into marginal Gaussian distributions, and then building a multivariate Gaussian distribution associated with a covariance matrix that takes into the correlation between DCT coefficients.

The contribution mentioned in the introduction and detailed through the manuscript lead mainly to two methodologies whose motivations are similar but whose applications diverge.

The embedding approach in the JPEG domain (see 4) has shown an empirical security in the order of 35% w.r.t state-of-the-art steganalysis algorithms while maintaining a very high embedding rate. However, the use of this method must be weighed up because of the computational complexity which makes its use constraining (see 4.3). If this algorithm is more in the scope of the proof of concept, its contributions to the field of steganography are not minor. As a matter of fact, this approach named "J-Cov-NS" in the manuscript is the genesis of our synchronous modification approach for additive schemes, which resulted in an increase in empirical security on the two schemes on which it has been used.

Perspectives

Following this brief summary of the work presented in this manuscript, it seems reasonable to ask about the various possibilities for perfecting these two dissimulation processes. For this purpose, a number of directions could be explored, some of which are outlined below:

- **Image model (Chapter 3):** It would be important to look for a model of joint distribution that do not come from a linear model, i.e. are not Gaussian. In order to do so, we could approximate the dependencies using the multivariate Gaussian model, since it's a multivariate model, that is easy to deal with and then use a better model to then fit the marginal. Model fitting could be done using optimal transport.
- **J-Cov-NS (Chapter 4):** It seems conceivable to use GPUs to perform the matrix operations related to the computations related to the covariance matrices in parallel and faster.
- **J-Cov-NS (Chapter 4):** The use of the Cholesky algorithm can be improved here to avoid recalculating the whole Schur complement at each iteration. This would help to reduce the computing time. But, we did not started the exploitation of this runway.
- **J-Cov-NS (Chapter 4):** We would like to confirm our expectations on the use of color channel synchronization.
- **J-Cov-NS (Chapter 4):** In future works it will be interesting also to check if the approximation $\mu \approx x$ made in the section 1 of chapter 3 explains the fact that we do not reach a 50% error rate.
- **Cov-UERD/Cov-JUNI (Chapter 5):** Through this method, we used a quantization step q equal to 1, which corresponds to a quantization of a quality factor of 100. However, we have used this quantification step on quality factors other than 100, which certainly encourages changes in the "right direction" in the sense of preserving correlations, but the dynamics of the resulting costs are not correct. We would like to carry out these experiments using the right methodology and thus employ the right quantification factors.

To conclude, this thesis allowed to extend natural steganography to the JPEG domain (for a given development pipeline). Our observations and contributions in natural steganography have also nourished the development of a more practical approach, thus reinforcing the importance of research in the direction of model-based steganography and on taking into account the dependencies between samples of the cover content.

Appendices

The appendices present, for each DCT mode of each lattice, the list of correlated modes belonging to previous lattices.

Mode / Block	$(0,2)$	$(1,3)$	$(2,4)$	$(3,5)$	$(4,6)$	$(5,7)$	$(7,1)$	$(6,0)$	$(0,3)$	$(1,4)$	$(2,5)$	$(3,6)$	$(4,7)$	$(7,2)$	$(6,1)$	$(5,0)$
1	(0, 2)	(1, 3)	(2, 4)	(3, 5)	(4, 6)	(5, 7)	(7, 1)	(6, 0)	(0, 3)	(1, 4)	(2, 5)	(3, 6)	(4, 7)	(7, 2)	(6, 1)	(5, 0)
	(0, 0)	(3, 3)	(4, 4)	(5, 5)	(6, 6)	(7, 7)	(7, 7)	(0, 0)	(0, 1)	(1, 2)	(4, 5)	(5, 6)	(6, 7)	(7, 0)	(0, 1)	(5, 6)
	(2, 2)	(1, 1)	(2, 2)	(3, 3)	(4, 4)	(5, 5)	(1, 1)	(6, 6)	(2, 3)	(3, 4)	(2, 3)	(3, 4)	(4, 5)	(1, 2)	(6, 7)	(7, 0)

Table 6.1: Correlated modes for each mode of $\mathbf{\Lambda_1}$ w.r.t. coefficients belonging to the previous lattice.

Mode / Block	$(0,4)$	$(1,5)$	$(2,6)$	$(3,7)$	$(7,3)$	$(6,2)$	$(5,1)$	$(4,0)$	$(0,5)$	$(1,6)$	$(2,7)$	$(7,4)$	$(6,3)$	$(5,2)$	$(4,1)$	$(3,0)$
1	(0, 4)	(1, 5)	(2, 6)	(3, 7)	(7, 3)	(6, 2)	(5, 1)	(4, 0)	(0, 5)	(1, 6)	(2, 7)	(7, 4)	(6, 3)	(5, 2)	(4, 1)	(3, 0)
	(0, 2)	(1, 3)	(4, 6)	(5, 7)	(7, 5)	(6, 4)	(3, 1)	(2, 0)	(0, 3)	(5, 6)	(4, 7)	(7, 2)	(6, 5)	(3, 2)	(2, 1)	(5, 0)
	(0, 6)	(5, 5)	(6, 6)	(1, 7)	(7, 1)	(4, 2)	(1, 1)	(6, 0)	(0, 1)	(3, 6)	(6, 7)	(7, 6)	(6, 1)	(5, 4)	(4, 5)	(1, 0)
	(0, 0)	(1, 1)	(2, 4)	(7, 7)	(7, 7)	(2, 2)	(5, 5)	(4, 4)	(4, 5)	(1, 4)	(2, 3)	(7, 0)	(4, 3)	(1, 2)	(6, 1)	(3, 4)
	(4, 4)	(3, 5)	(2, 2)	(3, 5)	(1, 3)	(6, 6)	(5, 3)	(0, 0)	(6, 5)	(1, 2)	(2, 1)	(1, 4)	(2, 3)	(5, 6)	(4, 3)	(3, 6)
	(2, 4)	(7, 5)	(2, 0)	(3, 3)	(5, 3)	(0, 2)	(5, 7)	(4, 6)	(2, 5)	(1, 0)	(0, 7)	(5, 4)	(0, 3)	(7, 2)	(0, 1)	(7, 0)

Table 6.2: Correlated modes for each mode of $\mathbf{\Lambda_2}$ w.r.t. coefficients belonging to the previous lattices.

Mode / Block	$(0,6)$	$(1,7)$	$(7,5)$	$(6,4)$	$(5,3)$	$(4,2)$	$(3,1)$	$(2,0)$	$(0,7)$	$(7,6)$	$(6,5)$	$(5,4)$	$(4,3)$	$(3,2)$	$(2,1)$	$(1,0)$
1	(0, 6)	(1, 7)	(7, 5)	(6, 4)	(5, 3)	(4, 2)	(3, 1)	(2, 0)	(0, 7)	(7, 6)	(6, 5)	(5, 4)	(4, 3)	(3, 2)	(2, 1)	(1, 0)
	(0, 4)	(5, 7)	(7, 3)	(6, 2)	(3, 3)	(4, 4)	(5, 1)	(4, 0)	(4, 7)	(7, 2)	(6, 3)	(3, 4)	(4, 5)	(3, 4)	(4, 1)	(5, 0)
	(0, 2)	(3, 7)	(7, 1)	(6, 6)	(5, 5)	(2, 2)	(3, 5)	(6, 0)	(6, 7)	(7, 4)	(6, 1)	(5, 2)	(2, 3)	(5, 2)	(2, 5)	(3, 0)
	(4, 6)	(1, 3)	(7, 7)	(4, 4)	(1, 3)	(6, 2)	(1, 1)	(2, 4)	(0, 1)	(7, 0)	(4, 5)	(1, 4)	(6, 3)	(3, 6)	(6, 1)	(1, 4)
	(6, 6)	(7, 7)	(1, 5)	(2, 4)	(5, 1)	(4, 6)	(3, 3)	(2, 6)	(0, 3)	(3, 6)	(2, 5)	(5, 6)	(4, 1)	(1, 2)	(2, 3)	(1, 6)
	(0, 0)	(1, 1)	(3, 5)	(6, 0)	(5, 7)	(0, 2)	(7, 1)	(0, 0)	(2, 7)	(1, 6)	(0, 5)	(5, 0)	(0, 3)	(7, 2)	(2, 7)	(1, 2)
	(2, 6)	(1, 5)	(5, 5)	(0, 4)	(7, 3)	(4, 0)	(3, 7)	(2, 2)	(0, 5)	(5, 6)	(6, 7)	(7, 4)	(4, 7)	(3, 0)	(0, 1)	(7, 0)

Table 6.3: Correlated modes for each mode of $\mathbf{\Lambda_3}$ w.r.t. coefficients belonging to the previous lattices.

Mode / Block	$(0,0)$	$(1,1)$	$(2,2)$	$(3,3)$	$(4,4)$	$(5,5)$	$(6,6)$	$(7,7)$	$(0,1)$	$(1,2)$	$(2,3)$	$(3,4)$	$(4,5)$	$(5,6)$	$(6,7)$	$(7,0)$
1	(0, 0)	(0, 1)	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 1)	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 0)
	(1, 0)	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 0)
	(2, 0)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 0)
	(3, 0)	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 0)
	(4, 0)	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 0)
	(5, 0)	(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 0)
	(6, 0)	(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 0)
	(7, 0)	(7, 1)	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 1)	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 0)
2	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(6, 0)	(7, 0)	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(6, 0)	(7, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(6, 1)	(7, 1)	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(6, 1)	(7, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(6, 4)	(7, 4)	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(6, 4)	(7, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(6, 5)	(7, 5)	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(6, 5)	(7, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(6, 6)	(7, 6)	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(6, 6)	(7, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(6, 7)	(7, 7)	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(6, 7)	(7, 7)
3	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(6, 0)	(7, 0)	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(6, 0)	(7, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(6, 1)	(7, 1)	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(6, 1)	(7, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(6, 2)	(7, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(6, 3)	(7, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(6, 4)	(7, 4)	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(6, 4)	(7, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(6, 5)	(7, 5)	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(6, 5)	(7, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(6, 6)	(7, 6)	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(6, 6)	(7, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(6, 7)	(7, 7)	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(6, 7)	(7, 7)
4	(0, 0)	(0, 1)	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 1)	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 0)
	(1, 0)	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 1)	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 0)
	(2, 0)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 1)	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 0)
	(3, 0)	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 1)	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 0)
	(4, 0)	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 1)	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 0)
	(5, 0)	(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 1)	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 0)
	(6, 0)	(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 1)	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 0)
	(7, 0)	(7, 1)	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 1)	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 0)
0	(0, 0)	(1, 1)	(2, 1)	(3, 3)	(4, 4)	(5, 5)	(6, 6)	(7, 7)	(0, 1)	(1, 2)	(2, 3)	(3, 4)	(4, 5)	(5, 6)	(6, 7)	(7, 0)

Table 6.4: Correlated modes for each mode of $\mathbf{\Lambda_4}$ w.r.t. coefficients belonging to the previous lattices.

Mode / Block	$(0,2)$	$(1,3)$	$(2,4)$	$(3,5)$	$(4,6)$	$(5,7)$	$(7,1)$	$(6,0)$	$(0,3)$	$(1,4)$	$(2,5)$	$(3,6)$	$(4,7)$	$(7,2)$	$(6,1)$	$(5,0)$
1	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 1)	(0, 0)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 2)	(0, 1)	(0, 0)
	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 1)	(1, 0)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 2)	(1, 1)	(1, 0)
	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 1)	(2, 0)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 2)	(2, 1)	(2, 0)
	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 1)	(3, 0)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 2)	(3, 1)	(3, 0)
	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 1)	(4, 0)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 2)	(4, 1)	(4, 0)
	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 1)	(5, 0)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 2)	(5, 1)	(5, 0)
	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 1)	(6, 0)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 2)	(6, 1)	(6, 0)
	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 1)	(7, 0)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 2)	(7, 1)	(7, 0)
2	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(7, 0)	(6, 0)	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(7, 0)	(6, 0)	(5, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(7, 1)	(6, 1)	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(7, 1)	(6, 1)	(5, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(7, 2)	(6, 2)	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(7, 2)	(6, 2)	(5, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(7, 3)	(6, 3)	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(7, 3)	(6, 3)	(5, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(7, 4)	(6, 4)	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(7, 4)	(6, 4)	(5, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(7, 5)	(6, 5)	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(7, 5)	(6, 5)	(5, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(7, 6)	(6, 6)	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(7, 6)	(6, 6)	(5, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(7, 7)	(6, 7)	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(7, 7)	(6, 7)	(5, 7)
3	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(5, 0)	(7, 0)	(6, 0)	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(4, 0)	(7, 0)	(6, 0)	(5, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(5, 1)	(7, 1)	(6, 1)	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(4, 1)	(7, 1)	(6, 1)	(5, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(5, 2)	(7, 2)	(6, 2)	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(4, 2)	(7, 2)	(6, 2)	(5, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(5, 3)	(7, 3)	(6, 3)	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(4, 3)	(7, 3)	(6, 3)	(5, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(5, 4)	(7, 4)	(6, 4)	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(4, 4)	(7, 4)	(6, 4)	(5, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(5, 5)	(7, 5)	(6, 5)	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(4, 5)	(7, 5)	(6, 5)	(5, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(5, 6)	(7, 6)	(6, 6)	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(4, 6)	(7, 6)	(6, 6)	(5, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(5, 7)	(7, 7)	(6, 7)	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(4, 7)	(7, 7)	(6, 7)	(5, 7)
4	(0, 2)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 1)	(0, 0)	(0, 3)	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 2)	(0, 1)	(0, 0)
	(1, 2)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 1)	(1, 0)	(1, 3)	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 2)	(1, 1)	(1, 0)
	(2, 2)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 1)	(2, 0)	(2, 3)	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 2)	(2, 1)	(2, 0)
	(3, 2)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 1)	(3, 0)	(3, 3)	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 2)	(3, 1)	(3, 0)
	(4, 2)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 1)	(4, 0)	(4, 3)	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 2)	(4, 1)	(4, 0)
	(5, 2)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 1)	(5, 0)	(5, 3)	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 2)	(5, 1)	(5, 0)
	(6, 2)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 1)	(6, 0)	(6, 3)	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 2)	(6, 1)	(6, 0)
	(7, 2)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 1)	(7, 0)	(7, 3)	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 2)	(7, 1)	(7, 0)
0	(0, 2)	(1, 3)	(2, 4)	(3, 5)	(4, 6)	(5, 7)	(7, 1)	(6, 0)	(0, 3)	(1, 4)	(2, 5)	(3, 6)	(4, 7)	(7, 2)	(6, 1)	(5, 0)
	(0, 0)	(3, 3)	(4, 4)	(5, 5)	(6, 6)	(7, 7)	(7, 7)	(0, 0)	(0, 1)	(1, 2)	(4, 5)	(5, 6)	(6, 7)	(7, 0)	(0, 1)	(5, 6)
	(2, 2)	(1, 1)	(2, 2)	(3, 3)	(4, 4)	(5, 5)	(1, 1)	(6, 6)	(2, 3)	(3, 4)	(2, 3)	(3, 4)	(4, 5)	(1, 2)	(6, 7)	(7, 0)

Table 6.5: Correlated modes for each mode of $\mathbf{\Lambda_5}$ w.r.t. coefficients belonging to the previous lattices.

Mode / Block	$(0,4)$	$(1,5)$	$(2,6)$	$(3,7)$	$(7,3)$	$(6,2)$	$(5,1)$	$(4,0)$	$(0,5)$	$(1,6)$	$(2,7)$	$(7,4)$	$(6,3)$	$(5,2)$	$(4,1)$	$(3,0)$
1	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 3)	(0, 2)	(0, 1)	(0, 0)	(0, 5)	(0, 6)	(0, 7)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)
	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 3)	(1, 2)	(1, 1)	(1, 0)	(1, 5)	(1, 6)	(1, 7)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)
	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 3)	(2, 2)	(2, 1)	(2, 0)	(2, 5)	(2, 6)	(2, 7)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)
	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 3)	(3, 2)	(3, 1)	(3, 0)	(3, 5)	(3, 6)	(3, 7)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)
	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 3)	(4, 2)	(4, 1)	(4, 0)	(4, 5)	(4, 6)	(4, 7)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)
	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 3)	(5, 2)	(5, 1)	(5, 0)	(5, 5)	(5, 6)	(5, 7)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)
	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 3)	(6, 2)	(6, 1)	(6, 0)	(6, 5)	(6, 6)	(6, 7)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)
	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 3)	(7, 2)	(7, 1)	(7, 0)	(7, 5)	(7, 6)	(7, 7)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)
2	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(0, 0)	(1, 0)	(2, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(0, 1)	(1, 1)	(2, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(0, 2)	(1, 2)	(2, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(0, 3)	(1, 3)	(2, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(0, 4)	(1, 4)	(2, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(0, 5)	(1, 5)	(2, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(0, 6)	(1, 6)	(2, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(0, 7)	(1, 7)	(2, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)
3	(0, 0)	(1, 0)	(2, 0)	(3, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(0, 0)	(1, 0)	(2, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)
	(0, 1)	(1, 1)	(2, 1)	(3, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(0, 1)	(1, 1)	(2, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)
	(0, 2)	(1, 2)	(2, 2)	(3, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(0, 2)	(1, 2)	(2, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)
	(0, 3)	(1, 3)	(2, 3)	(3, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(0, 3)	(1, 3)	(2, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)
	(0, 4)	(1, 4)	(2, 4)	(3, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(0, 4)	(1, 4)	(2, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)
	(0, 5)	(1, 5)	(2, 5)	(3, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(0, 5)	(1, 5)	(2, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)
	(0, 6)	(1, 6)	(2, 6)	(3, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(0, 6)	(1, 6)	(2, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)
	(0, 7)	(1, 7)	(2, 7)	(3, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(0, 7)	(1, 7)	(2, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)
4	(0, 4)	(0, 5)	(0, 6)	(0, 7)	(0, 3)	(0, 2)	(0, 1)	(0, 0)	(0, 5)	(0, 6)	(0, 7)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)
	(1, 4)	(1, 5)	(1, 6)	(1, 7)	(1, 3)	(1, 2)	(1, 1)	(1, 0)	(1, 5)	(1, 6)	(1, 7)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)
	(2, 4)	(2, 5)	(2, 6)	(2, 7)	(2, 3)	(2, 2)	(2, 1)	(2, 0)	(2, 5)	(2, 6)	(2, 7)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)
	(3, 4)	(3, 5)	(3, 6)	(3, 7)	(3, 3)	(3, 2)	(3, 1)	(3, 0)	(3, 5)	(3, 6)	(3, 7)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)
	(4, 4)	(4, 5)	(4, 6)	(4, 7)	(4, 3)	(4, 2)	(4, 1)	(4, 0)	(4, 5)	(4, 6)	(4, 7)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)
	(5, 4)	(5, 5)	(5, 6)	(5, 7)	(5, 3)	(5, 2)	(5, 1)	(5, 0)	(5, 5)	(5, 6)	(5, 7)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)
	(6, 4)	(6, 5)	(6, 6)	(6, 7)	(6, 3)	(6, 2)	(6, 1)	(6, 0)	(6, 5)	(6, 6)	(6, 7)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)
	(7, 4)	(7, 5)	(7, 6)	(7, 7)	(7, 3)	(7, 2)	(7, 1)	(7, 0)	(7, 5)	(7, 6)	(7, 7)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)
0	(0, 4)	(1, 5)	(2, 6)	(3, 7)	(7, 3)	(6, 2)	(5, 1)	(4, 0)	(0, 5)	(1, 6)	(2, 7)	(7, 4)	(6, 3)	(5, 2)	(4, 1)	(3, 0)
	(0, 2)	(1, 3)	(4, 6)	(5, 7)	(7, 5)	(6, 4)	(3, 1)	(2, 0)	(0, 3)	(5, 6)	(4, 7)	(7, 2)	(6, 5)	(3, 2)	(2, 1)	(5, 0)
	(0, 6)	(5, 5)	(6, 6)	(1, 7)	(7, 1)	(4, 2)	(1, 1)	(6, 0)	(0, 1)	(3, 6)	(6, 7)	(7, 6)	(6, 1)	(5, 4)	(4, 5)	(1, 0)
	(0, 0)	(1, 1)	(2, 4)	(7, 7)	(7, 7)	(2, 2)	(5, 5)	(4, 4)	(4, 5)	(1, 4)	(2, 3)	(7, 0)	(4, 3)	(1, 2)	(6, 1)	(3, 4)
	(4, 4)	(3, 5)	(2, 2)	(3, 5)	(1, 3)	(6, 6)	(5, 3)	(0, 0)	(6, 5)	(1, 2)	(2, 1)	(1, 4)	(2, 3)	(5, 6)	(4, 3)	(3, 6)
	(2, 4)	(7, 5)	(2, 0)	(3, 3)	(5, 3)	(0, 2)	(5, 7)	(4, 6)	(2, 5)	(1, 0)	(0, 7)	(5, 4)	(0, 3)	(7, 2)	(0, 1)	(7, 0)

Table 6.6: Correlated modes for each mode of $\mathbf{\Lambda_6}$ w.r.t. coefficients belonging to the previous lattices.

Mode / Block	$(0,6)$	$(1,7)$	$(7,5)$	$(6,4)$	$(5,3)$	$(4,2)$	$(3,1)$	$(\mathbf{2},\mathbf{0})$	$(0,7)$	$(7,6)$	$(6,5)$	$(5,4)$	$(4,3)$	$(3,2)$	$(2,1)$	$(1,0)$
1	(0, 6)	(0, 7)	(0, 5)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)	(0, 7)	(0, 6)	(0, 5)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)
	(1, 6)	(1, 7)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)	(1, 7)	(1, 6)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)
	(2, 6)	(2, 7)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)	(2, 7)	(2, 6)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)
	(3, 6)	(3, 7)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)	(3, 7)	(3, 6)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)
	(4, 6)	(4, 7)	(4, 5)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)	(4, 7)	(4, 6)	(4, 5)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)
	(5, 6)	(5, 7)	(5, 5)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)	(5, 7)	(5, 6)	(5, 5)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)
	(6, 6)	(6, 7)	(6, 5)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)	(6, 7)	(6, 6)	(6, 5)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)
	(7, 6)	(7, 7)	(7, 5)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)	(7, 7)	(7, 6)	(7, 5)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)
2	(0, 0)	(1, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)	(2, 0)	(0, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)	(2, 0)	(1, 0)
	(0, 1)	(1, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)	(2, 1)	(0, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)	(2, 1)	(1, 1)
	(0, 2)	(1, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)	(2, 2)	(0, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)	(2, 2)	(1, 2)
	(0, 3)	(1, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)	(2, 3)	(0, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)	(2, 3)	(1, 3)
	(0, 4)	(1, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)	(2, 4)	(0, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)	(2, 4)	(1, 4)
	(0, 5)	(1, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)	(2, 5)	(0, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)	(2, 5)	(1, 5)
	(0, 6)	(1, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)	(2, 6)	(0, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)	(2, 6)	(1, 6)
	(0, 7)	(1, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)	(2, 7)	(0, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)	(2, 7)	(1, 7)
3	(0, 0)	(1, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)	(2, 0)	(0, 0)	(7, 0)	(6, 0)	(5, 0)	(4, 0)	(3, 0)	(2, 0)	(1, 0)
	(0, 1)	(1, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)	(2, 1)	(0, 1)	(7, 1)	(6, 1)	(5, 1)	(4, 1)	(3, 1)	(2, 1)	(1, 1)
	(0, 2)	(1, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)	(2, 2)	(0, 2)	(7, 2)	(6, 2)	(5, 2)	(4, 2)	(3, 2)	(2, 2)	(1, 2)
	(0, 3)	(1, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)	(2, 3)	(0, 3)	(7, 3)	(6, 3)	(5, 3)	(4, 3)	(3, 3)	(2, 3)	(1, 3)
	(0, 4)	(1, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)	(2, 4)	(0, 4)	(7, 4)	(6, 4)	(5, 4)	(4, 4)	(3, 4)	(2, 4)	(1, 4)
	(0, 5)	(1, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)	(2, 5)	(0, 5)	(7, 5)	(6, 5)	(5, 5)	(4, 5)	(3, 5)	(2, 5)	(1, 5)
	(0, 6)	(1, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)	(2, 6)	(0, 6)	(7, 6)	(6, 6)	(5, 6)	(4, 6)	(3, 6)	(2, 6)	(1, 6)
	(0, 7)	(1, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)	(2, 7)	(0, 7)	(7, 7)	(6, 7)	(5, 7)	(4, 7)	(3, 7)	(2, 7)	(1, 7)
4	(0, 6)	(0, 7)	(0, 5)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)	(0, 7)	(0, 6)	(0, 5)	(0, 4)	(0, 3)	(0, 2)	(0, 1)	(0, 0)
	(1, 6)	(1, 7)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)	(1, 7)	(1, 6)	(1, 5)	(1, 4)	(1, 3)	(1, 2)	(1, 1)	(1, 0)
	(2, 6)	(2, 7)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)	(2, 7)	(2, 6)	(2, 5)	(2, 4)	(2, 3)	(2, 2)	(2, 1)	(2, 0)
	(3, 6)	(3, 7)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)	(3, 7)	(3, 6)	(3, 5)	(3, 4)	(3, 3)	(3, 2)	(3, 1)	(3, 0)
	(4, 6)	(4, 7)	(4, 5)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)	(4, 7)	(4, 6)	(4, 5)	(4, 4)	(4, 3)	(4, 2)	(4, 1)	(4, 0)
	(5, 6)	(5, 7)	(5, 5)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)	(5, 7)	(5, 6)	(5, 5)	(5, 4)	(5, 3)	(5, 2)	(5, 1)	(5, 0)
	(6, 6)	(6, 7)	(6, 5)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)	(6, 7)	(6, 6)	(6, 5)	(6, 4)	(6, 3)	(6, 2)	(6, 1)	(6, 0)
	(7, 6)	(7, 7)	(7, 5)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)	(7, 7)	(7, 6)	(7, 5)	(7, 4)	(7, 3)	(7, 2)	(7, 1)	(7, 0)
0	(0, 6)	(1, 7)	(7, 5)	(6, 4)	(5, 3)	(4, 2)	(3, 1)	(2, 0)	(0, 7)	(7, 6)	(6, 5)	(5, 4)	(4, 3)	(3, 2)	(2, 1)	(1, 0)
	(0, 4)	(5, 7)	(7, 3)	(6, 2)	(3, 3)	(4, 4)	(5, 1)	(4, 0)	(4, 7)	(7, 2)	(6, 3)	(3, 4)	(4, 5)	(3, 4)	(4, 1)	(5, 0)
	(0, 2)	(3, 7)	(7, 1)	(6, 6)	(5, 5)	(2, 2)	(3, 5)	(6, 0)	(6, 7)	(7, 4)	(6, 1)	(5, 2)	(2, 3)	(5, 2)	(2, 5)	(3, 0)
	(4, 6)	(1, 3)	(7, 7)	(4, 4)	(1, 3)	(6, 2)	(1, 1)	(2, 4)	(0, 1)	(7, 0)	(4, 5)	(1, 4)	(6, 3)	(3, 6)	(6, 1)	(1, 4)
	(6, 6)	(7, 7)	(1, 5)	(2, 4)	(5, 1)	(4, 6)	(3, 3)	(2, 6	(0, 3)	(3, 6)	(2, 5)	(5, 6)	(4, 1)	(1, 2)	(2, 3)	(1, 6)
	(0, 0)	(1, 1)	(3, 5)	(6, 0)	(5, 7)	(0, 2)	(7, 1)	(0, 0)	(2, 7)	(1, 6)	(0, 5)	(5, 0)	(0, 3)	(7, 2)	(2, 7)	(1, 2)
	(2, 6)	(1, 5)	(5, 5)	(0, 4)	(7, 3)	(4, 0)	(3, 7)	(2, 2)	(0, 5)	(5, 6)	(6, 7)	(7, 4)	(4, 7)	(3, 0)	(0, 1)	(7, 0)

Table 6.7: Correlated modes for each mode of $\mathbf{\Lambda_7}$ w.r.t. coefficients belonging to the previous lattices. The 38 modes (depicted in figure 3.27) used to compute the conditional probability of mode $(2,0) \in \Lambda_7$ are colored in blue while and the red one correspond to itself.

Bibliography

[1] . *LibRaw raw image decoder*. http://www.libraw.org. 2010.

[2] Hasan Abdulrahman, Marc Chaumont, Philippe Montesinos, and Baptiste Magnier. "Color images steganalysis using rgb channel geometric transformation measures". In: *Security and communication networks* 9.15 (2016), pp. 2945–2956.

[3] Ross Anderson. "Stretching the limits of steganography". In: *International Workshop on Information Hiding*. Springer. 1996, pp. 39–48.

[4] European Machine Vision Association et al. "Standard for characterization of image sensors and cameras". In: *EMVA Standard* 1288 (2010).

[5] P. Bas, T. Filler, and T. Pevny. ""Break Our Steganographic System": The Ins and Outs of Organizing BOSS". In: *INFORMATION HIDING*. Vol. 6958/2011. Lecture Notes in Computer Science. Czech Republic, Sept. 2011, pp. 59–70. DOI: 10.1007/978-3-642-24178-9_15.

[6] Patrick Bas. "An embedding mechanism for Natural Steganography after down-sampling". In: IEEE ICASSP. 2017.

[7] Patrick Bas. "An embedding mechanism for Natural Steganography after down-sampling". In: *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2017, pp. 2127–2131.

[8] Patrick Bas. "Natural Steganography: Cover-source Switching For Better Steganography". working paper or preprint. July 2016. URL: https://arxiv.org/abs/1607.07824.

[9] Patrick Bas. "Steganography via cover-source switching". In: *2016 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE. 2016, pp. 1–6.

[10] Patrick Bas. "Steganography via Cover-Source Switching". In: IEEE Workshop on Information Forensics and Security (WIFS). 2016. URL: https://hal.archives-ouvertes.fr/hal-01360024.

[11] Solène Bernard, Tomás Pevny, Patrick Bas, and John Klein. “Exploiting Adversarial Embeddings for Better Steganography”. In: *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*. 2019, pp. 216–221.

[12] André Blanc-Lapierre and Robert Fortet. *Theory of random functions*. Gordon and Breach, 1965.

[13] Mehdi Boroumand, Mo Chen, and Jessica Fridrich. “Deep residual network for steganalysis of digital images”. In: *IEEE Transactions on Information Forensics and Security* 14.5 (2018), pp. 1181–1193.

[14] Jan Butora and Jessica Fridrich. “Reverse JPEG Compatibility Attack”. In: *IEEE Transactions on Information Forensics and Security* 15 (2019), pp. 1444–1454.

[15] Christian Cachin. “An information-Theoretic Model for Steganography”. In: *Information Hiding: Second International Workshop IHW'98*. Portland, Oregon, USA, 1998.

[16] Edward Chang, Shiufun Cheung, and Davis Y Pan. “Color filter array recovery using a threshold-based variable number of gradients”. In: *Sensors, Cameras, and Applications for Digital Photography*. Vol. 3650. International Society for Optics and Photonics. 1999, pp. 36–43.

[17] Chunhua Chen and Yun Q Shi. “JPEG image steganalysis utilizing both intrablock and interblock correlations”. In: *2008 IEEE International Symposium on Circuits and Systems*. IEEE. 2008, pp. 3029–3032.

[18] M. Chen, J. Fridrich, M. Goljan, and J. Lukás. “Determining image origin and integrity using sensor noise”. In: *Information Forensics and Security, IEEE Transactions on* 3.1 (2008), pp. 74–90.

[19] Mo Chen, Jessica Fridrich, Miroslav Goljan, and Jan Lukáš. “Source digital camcorder identification using sensor photo response non-uniformity”. In: *Security, steganography, and watermarking of multimedia contents IX*. Vol. 6505. International Society for Optics and Photonics. 2007, 65051G.

[20] Rémi Cogranne. “Selection-channel-aware reverse jpeg compatibility for highly reliable steganalysis of jpeg images”. In: *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. 2020.

[21] Rémi Cogranne, Patrick Bas, and Marc Chaumont. *STÉGANALYSE: Détection d'information cachée dans des contenus multimédias*. 2020.

[22] Remi Cogranne and Jessica Fridrich. “Modeling and extending the ensemble classifier for steganalysis of digital images using hypothesis testing theory”. In: *IEEE Transactions on Information Forensics and Security* 10.12 (2015), pp. 2627–2642.

[23] Rémi Cogranne, Quentin Giboulot, and Patrick Bas. “Steganography by Minimizing Statistical Detectability: The cases of JPEG and Color Images.” In: *ACM Information Hiding and MultiMedia Security (IH&MMSec)*. 2020.

[24] Rémi Cogranne, Quentin Giboulot, and Patrick Bas. “The ALASKA Steganalysis Challenge: A First Step Towards Steganalysis”. In: *Proceedings of the ACM Workshop on Information Hiding and Multimedia Security*. 2019, pp. 125–137.

[25] Rémi Cogranne and Florent Retraint. “An asymptotically uniformly most powerful test for LSB matching detection”. In: *Information Forensics and Security, IEEE Transactions on* 8.3 (2013), pp. 464–476.

[26] Rémi Cogranne and Florent Retraint. “Application of hypothesis testing theory for optimal detection of LSB matching data hiding”. In: *Signal Processing* 93.7 (2013), pp. 1724–1737.

[27] Rémi Cogranne, Vahid Sedighi, Jessica Fridrich, and Tomáš Pevny. “Is ensemble classifier needed for steganalysis in high-dimensional feature spaces?” In: *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*. IEEE. 2015, pp. 1–6.

[28] Rémi Cogranne, Cathel Zitzmann, Lionel Fillatre, Florent Retraint, Igor Nikiforov, and Philippe Cornu. “A cover image model for reliable steganalysis”. In: *International Workshop on Information Hiding*. Springer. 2011, pp. 178–192.

[29] Thomas M. Cover, Joy A. Thomas, John Wiley, et al. *Elements of information theory*. Vol. 6. Wiley Online Library, 1991.

[30] Tomáš Denemark, Patrick Bas, and Jessica Fridrich. “Natural Steganography in JPEG Compressed Images”. In: *Electronic Imaging*. San Francisco, United States, 2018. URL: `https://hal.archives-ouvertes.fr/hal-01687194`.

[31] Tomáš Denemark, Patrick Bas, and Jessica Fridrich. “Natural steganography in JPEG compressed images”. In: *Electronic Imaging* 2018.7 (2018), pp. 316–1.

[32] Tomáš Denemark and Jessica Fridrich. “Improving steganographic security by synchronizing the selection channel”. In: *Proceedings of the 3rd ACM Workshop on Information Hiding and Multimedia Security*. 2015, pp. 5–14.

[33] Tomáš Denemark and Jessica Fridrich. "Side-informed steganography with additive distortion". In: *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE. 2015, pp. 1–6.

[34] Tomáš Denemark, Jessica Fridrich, and Pedro Comesaña-Alfaro. "Improving selection-channel-aware steganalysis features". In: *Electronic Imaging* 2016.8 (2016), pp. 1–8.

[35] Tomas Denemark, Vahid Sedighi, Vojtech Holub, Rémi Cogranne, and Jessica Fridrich. "Selection-channel-aware rich model for steganalysis of digital images". In: *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE. 2014, pp. 48–53.

[36] Tomáš Denemark Denemark, Mehdi Boroumand, and Jessica Fridrich. "Steganalysis features for content-adaptive JPEG steganography". In: *IEEE Transactions on Information Forensics and Security* 11.8 (2016), pp. 1736–1746.

[37] Tomas Filler, Jan Judas, and Jessica Fridrich. "Minimizing additive distortion in steganography using syndrome-trellis codes". In: *Information Forensics and Security, IEEE Transactions on* 6.3 (2011), pp. 920–935.

[38] Alessandro Foi. "Clipped noisy images: Heteroskedastic modeling and practical denoising". In: *Signal Processing* 89.12 (2009), pp. 2609–2629.

[39] Alessandro Foi, Sakari Alenius, Vladimir Katkovnik, and Karen Egiazarian. "Noise measurement for raw-data of digital imaging sensors by automatic segmentation of nonuniform targets". In: *IEEE Sensors Journal* 7.10 (2007), pp. 1456–1461.

[40] Alessandro Foi, Mejdi Trimeche, Vladimir Katkovnik, and Karen Egiazarian. "Practical Poissonian-Gaussian noise modeling and fitting for single-image raw-data". In: *Image Processing, IEEE Transactions on* 17.10 (2008), pp. 1737–1754.

[41] Jessica Fridrich. *Steganography in digital media: principles, algorithms, and applications*. Cambridge University Press, 2009.

[42] Jessica Fridrich, Miroslav Goljan, and David Soukal. "Perturbed quantization steganography with wet paper codes". In: *Proceedings of the 2004 workshop on Multimedia and security*. ACM. 2004, pp. 4–15.

[43] Jessica Fridrich and Jan Kodovsky. "Rich models for steganalysis of digital images". In: *Information Forensics and Security, IEEE Transactions on* 7.3 (2012), pp. 868–882.

[44] Miroslav Goljan and Jessica Fridrich. “CFA-aware features for steganalysis of color images”. In: *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics. 2015, pp. 94090V–94090V.

[45] Miroslav Goljan, Jessica Fridrich, and Rémi Cogranne. “Rich model for steganalysis of color images”. In: *2014 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE. 2014, pp. 185–190.

[46] Miroslav Goljan, Jessica Fridrich, Rémi Cogranne, et al. “Rich model for steganalysis of color images”. In: *Parallel Computing Technologies (PARCOMPTECH), 2015 National Conference on*. IEEE. 2015, pp. 185–190.

[47] Linjie Guo, Jiangqun Ni, Wenkang Su, Chengpei Tang, and Yun-Qing Shi. “Using Statistical Image Model for JPEG Steganography: Uniform Embedding Revisited”. In: *IEEE Transactions on Information Forensics and Security* 10.12 (2015), pp. 2669–2680.

[48] Keigo Hirakawa and Thomas W Parks. “Adaptive homogeneity-directed demosaicing algorithm”. In: *IEEE Transactions on Image Processing* 14.3 (2005), pp. 360–369.

[49] Vojtěch Holub and Jessica Fridrich. “Low-complexity features for JPEG steganalysis using undecimated DCT”. In: *IEEE Transactions on Information Forensics and Security* 10.2 (2014), pp. 219–228.

[50] Vojtěch Holub and Jessica Fridrich. “Low-complexity features for JPEG steganalysis using undecimated DCT”. In: *IEEE Transactions on Information Forensics and Security* 10.2 (2015), pp. 219–228.

[51] Vojtěch Holub and Jessica Fridrich. “Phase-aware projection model for steganalysis of JPEG images”. In: *IS&T/SPIE Electronic Imaging*. International Society for Optics and Photonics. 2015, 94090T–94090T.

[52] Vojtech Holub and Jessica Fridrich. “Random projections of residuals for digital image steganalysis”. In: *Information Forensics and Security, IEEE Transactions on* 8.12 (2013), pp. 1996–2006.

[53] Vojtěch Holub, Jessica Fridrich, and Tomáš Denemark. “Universal distortion function for steganography in an arbitrary domain”. In: *EURASIP Journal on Information Security* 2014.1 (2014), p. 1.

[54] Andrew D Ker. “Steganographic strategies for a square distortion function”. In: *Electronic Imaging 2008*. International Society for Optics and Photonics. 2008, pp. 681904–681904.

[55] Andrew D Ker, Patrick Bas, Rainer Böhme, Rémi Cogranne, Scott Craver, Tomáš Filler, Jessica Fridrich, and Tomáš Pevny. "Moving steganography and steganalysis from the laboratory into the real world". In: *Proceedings of the first ACM workshop on Information hiding and multimedia security*. ACM. 2013, pp. 45–58.

[56] Andrew D Ker and Tomás Pevny. "Identifying a steganographer in realistic and heterogeneous data sets". In: *Media Watermarking, Security, and Forensics 2012*. Vol. 8303. International Society for Optics and Photonics. 2012, 83030N.

[57] Andrew D Ker, Tomáš Pevny, Jan Kodovsky, and Jessica Fridrich. "The square root law of steganographic capacity". In: *Proceedings of the 10th ACM workshop on Multimedia and security*. ACM. 2008, pp. 107–116.

[58] Matthias Kirchner and Rainer Böhme. ""Steganalysis in Technicolor" Boosting WS detection of stego images from CFA-interpolated covers". In: *2014 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2014, pp. 3982–3986.

[59] Jan Kodovský and Jessica Fridrich. "Calibration Revisited". In: *Proceedings of the 11th ACM Workshop on Multimedia and Security*. MM&Sec '09. Princeton, New Jersey, USA: Association for Computing Machinery, 2009, pp. 63–74. ISBN: 9781605584928. DOI: 10.1145/1597817.1597830. URL: https://doi.org/10.1145/1597817.1597830.

[60] Jan Kodovsky and Jessica Fridrich. "Steganalysis in high dimensions: Fusing classifiers built on random subspaces". In: *Media Watermarking, Security, and Forensics III*. Vol. 7880. International Society for Optics and Photonics. 2011, p. 78800L.

[61] Jan Kodovsky and Jessica Fridrich. "Steganalysis of JPEG images using rich models". In: *Media Watermarking, Security, and Forensics 2012*. Vol. 8303. International Society for Optics and Photonics. 2012, 83030A.

[62] Jan Kodovsky, Jessica Fridrich, and Vojtěch Holub. "Ensemble classifiers for steganalysis of digital media". In: *IEEE Transactions on Information Forensics and Security* 7.2 (2011), pp. 432–444.

[63] Jan Kodovsky, Tomáš Pevny, and Jessica Fridrich. "Modern steganalysis can detect YASS". In: *Media Forensics and Security II*. Vol. 7541. International Society for Optics and Photonics. 2010, p. 754102.

[64] Sarra Kouider, Marc Chaumont, and William Puech. "Adaptive steganography by oracle (ASO)". In: *Multimedia and Expo (ICME), 2013 IEEE International Conference on*. IEEE. 2013, pp. 1–6.

[65] Marc Lebrun, Miguel Colom, and Jean-Michel Morel. "The noise clinic: a blind image denoising algorithm". In: *Image Processing On Line* 5 (2015), pp. 1–54.

[66] Ji Soo Lee. "Photoresponse of CMOS Image Sensors". PhD thesis. University of Waterloo, Ontario, Canada, 2003.

[67] Bin Li, Ming Wang, Jiwu Huang, and Xiaolong Li. "A new cost function for spatial image steganography". In: *Image Processing (ICIP), 2014 IEEE International Conference on*. IEEE. 2014, pp. 4206–4210.

[68] Bin Li, Ming Wang, Xiaolong Li, Shunquan Tan, and Jiwu Huang. "A strategy of clustering modification directions in spatial image steganography". In: *Information Forensics and Security, IEEE Transactions on* 10.9 (2015), pp. 1905–1917.

[69] Weixiang Li, Weiming Zhang, Kejiang Chen, Wenbo Zhou, and Nenghai Yu. "Defining Joint Distortion for JPEG Steganography". In: *Proceedings of the 6th ACM Workshop on Information Hiding and Multimedia Security*. ACM. 2018, pp. 5–16.

[70] C kai Lin. "Pixel grouping for color filter array demosaicing". In: *25/05/2007)* (2006).

[71] Qingzhong Liu. "Steganalysis of DCT-embedding based adaptive steganography and YASS". In: *Proceedings of the thirteenth ACM multimedia workshop on Multimedia and security*. 2011, pp. 77–86.

[72] Jessica Fridrich Mehdi Boroumand. "Synchronizing Embedding Changes in Side-Informed Steganography". In: *Proc. IS&T, Electronic Imaging, Media Watermarking, Security, and Forensics 2020, San Francisco, CA* (2020).

[73] Richard von Mises. "Mathematical Theory of Probability and Statistics". In: *Mathematical Theory of Probability and Statistics, New York: Academic Press, 1964* (1964).

[74] Thomas Pevny, Patrick Bas, and Jessica Fridrich. "Steganalysis by Subtractive Pixel Adjacency Matrix". In: *Information Forensics and Security, IEEE Transactions on* 5.2 (2010), pp. 215 –224. ISSN: 1556-6013. DOI: 10.1109/TIFS.2010.2045842.

[75] Tomáš Pevny, Patrick Bas, and Jessica Fridrich. "Steganalysis by subtractive pixel adjacency matrix". In: *IEEE Transactions on information Forensics and Security* 5.2 (2010), pp. 215–224.

[76] Tomas Pevny and Jessica Fridrich. “Merging Markov and DCT features for multi-class JPEG steganalysis”. In: *Security, Steganography, and Watermarking of Multimedia Contents IX*. Vol. 6505. International Society for Optics and Photonics. 2007, p. 650503.

[77] Tong Qiao, Florent Retraint, Rémi Cogranne, and Thanh Hai Thai. “Source camera device identification based on raw images”. In: *2015 IEEE international conference on image processing (ICIP)*. IEEE. 2015, pp. 3812–3816.

[78] Phil Sallee. “Model-based steganography”. In: *International workshop on digital watermarking*. Springer. 2003, pp. 154–167.

[79] Vahid Sedighi, Rémi Cogranne, and Jessica Fridrich. “Content-adaptive steganography by minimizing statistical detectability”. In: *IEEE Transactions on Information Forensics and Security* 11.2 (2015), pp. 221–234.

[80] Xiaofeng Song, Fenlin Liu, Chunfang Yang, Xiangyang Luo, and Yi Zhang. “Steganalysis of adaptive JPEG steganography using 2D Gabor filters”. In: *Proceedings of the 3rd ACM workshop on information hiding and multimedia security*. ACM. 2015, pp. 15–23.

[81] Théo Taburet, Patrick Bas, Jessica Fridrich, and Wadih Sawaya. “Computing Dependencies between DCT Coefficients for Natural Steganography in JPEG Domain”. In: *IH-MMSec*. IH&MMSec’19 Proceedings of the ACM Workshop on Information Hiding and Multimedia Security. Paris, France, July 2019. DOI: 10.1145/3335203.3335715. URL: https://hal.archives-ouvertes.fr/hal-02165866.

[82] Théo Taburet, Patrick Bas, Wadih Sawaya, and Rémi Cogranne. “JPEG Steganography and Synchronization of DCT Coefficients for a Given Development Pipeline”. In: *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*. IH&MMSec ’20. Denver, CO, USA: Association for Computing Machinery, 2020, pp. 139–149. ISBN: 9781450370509. DOI: 10.1145/3369412.3395074. URL: https://doi.org/10.1145/3369412.3395074.

[83] Théo Taburet, Patrick Bas, Wadih Sawaya, and Remi Cogranne. *JPEG Steganography and Synchronization of DCT Coefficients for a Given Development Pipeline*. https://arxiv.org/abs/2003.10082v1. 2020. arXiv: 2003.10082 [cs.MM].

[84] Théo Taburet, Patrick Bas, Wadih Sawaya, and Jessica Fridrich. “A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain”. In: *Electronic Imaging*. Burlingame, United States, Jan. 2019.

[85] Théo Taburet, Patrick Bas, Wadih Sawaya, and Jessica Fridrich. “A Natural Steganography Embedding Scheme Dedicated to Color Sensors in the JPEG Domain”. In: *Electronic Imaging* 2019.5 (2019), pp. 542–1.

[86] Théo Taburet, Patrick Bas, Wadih Sawaya, and Jessica Fridrich. “Stéganographie naturelle pour images JPEG”. In: 2019.

[87] Théo Taburet, Louis Filstroff, Patrick Bas, and Wadih Sawaya. “An empirical study of steganography and steganalysis of color images in the JPEG domain”. In: *International Workshop on Digital Watermarking*. Springer. 2018, pp. 290–303.

[88] Théo Taburet, Louis Filstroff, Patrick Bas, and Wadih Sawaya. “An Empirical Study of Steganography and Steganalysis of Color Images in the JPEG Domain”. In: *IWDW, International Workshop on Digital Forensics and Watermarking*. Jeju, South Korea, Oct. 2018. URL: https://hal.archives-ouvertes.fr/hal-01904482.

[89] Weixuan Tang, Bin Li, Weiqi Luo, and Jiwu Huang. “Clustering steganographic modification directions for color components”. In: *IEEE Signal Processing Letters* 23.2 (2015), pp. 197–201.

[90] Weixuan Tang, Bin Li, Shunquan Tan, Mauro Barni, and Jiwu Huang. “CNN-based Adversarial Embedding for Image Steganography”. In: *IEEE Transactions on Information Forensics and Security* (2019).

[91] Weixuan Tang, Shunquan Tan, Bin Li, and Jiwu Huang. “Automatic steganographic distortion learning using a generative adversarial network”. In: *IEEE Signal Processing Letters* 24.10 (2017), pp. 1547–1551.

[92] Thanh Hai Thai, Remi Cogranne, and Florent Retraint. “Camera model identification based on the heteroscedastic noise model”. In: *IEEE Transactions on Image Processing* 23.1 (2013), pp. 250–263.

[93] Wadih Sawaya Théo Taburet Patrick Bas and Jessica Fridrich. “Natural Steganography in JPEG Domain with a Linear Development Pipeline”. In: *Transactions on Information Forensics & Security* (2020).

[94] Tomas Filler Tomas Pevny and Patrick Bas. “Using High-Dimensional Image Models to Perform Highly Undetectable Steganography”. In: *Information Hiding 2010*. Calgary, Canada, 2010.

[95] Andreas Westfeld. “F5—a steganographic algorithm”. In: *International workshop on information hiding*. Springer. 2001, pp. 289–302.

[96] Guanshuo Xu, Han-Zhou Wu, and Yun-Qing Shi. “Structural design of convolutional neural networks for steganalysis”. In: *IEEE Signal Processing Letters* 23.5 (2016), pp. 708–712.

[97] Jianhua Yang, Danyang Ruan, Jiwu Huang, Xiangui Kang, and Yun-Qing Shi. "An embedding cost learning framework using gan". In: *IEEE Transactions on Information Forensics and Security* 15 (2019), pp. 839–851.

[98] Jian Ye, Jiangqun Ni, and Yang Yi. "Deep learning hierarchical representations for image steganalysis". In: *IEEE Transactions on Information Forensics and Security* 12.11 (2017), pp. 2545–2557.

[99] Mehdi Yedroudj, Frédéric Comby, and Marc Chaumont. "Yedroudj-net: An efficient CNN for spatial steganalysis". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE. 2018, pp. 2092–2096.

[100] Weiming Zhang, Zhuo Zhang, Lili Zhang, Hanyi Li, and Nenghai Yu. "Decomposing joint distortion for adaptive steganography". In: *IEEE Transactions on Circuits and Systems for Video Technology* 27.10 (2016), pp. 2274–2280.

[101] Cathel Zitzmann, Rémi Cogranne, Florent Retraint, Igor Nikiforov, Lionel Fillatre, and Philippe Cornu. "Statistical decision methods in hidden information detection". In: *International Workshop on Information Hiding*. Springer. 2011, pp. 163–177.

Méthodes de stéganographie fondées sur la prise en compte du bruit de capteur:

La stéganographie est un terme désignant une communication secrète ou l'utilisateur cherche à cacher des messages dans des objets anodins afin de les transmettre à un ou plusieurs destinataires. Si la méthode de stéganographie est sûre, il sera impossible pour un système de détection de distinguer les objets anodins de ceux contenant un message.
Pour les images nous pouvons définir la perturbation induite par l'insertion d'un message secret sur l'image initiale (appelée l'image cover) comme l'ajout d'un signal spécifique. L'indétectabilité va alors reposer sur le fait que le signal ajouté ne perturbe pas les propriétés statistiques de l'image initiale. C'est en partant de ce principe que la Stéganographie Naturelle est née. Dans ce paradigme le message inséré cherche à imiter d'un bruit naturel (le bruit photonique du capteur utilisé) et l'image générée (l'image stego) dispose ainsi des propriétés d'une image anodine acquise à une sensibilité supérieure, ce qui garanti une sécurité pratique importante.
Ce manuscrit présente des contributions qui étendent cette méthode aux images JPEG et dérive également de celle-ci un schéma d'insertion plus générique.

Mots-clefs: Stéganographie, Stéganalyse, Sécurité

Steganography methods based on sensor noise estimation:

Steganography is a term referring to the concept of secret communication based on hiding messages in innocuous objects that would be communicated to the recipient(s): it should be impossible for a detection system to distinguish innocuous objects from those containing a message.
For images, the disturbance induced by the embedding of a secret message in the initial image (the cover image) can be defined as a specific signal, and the undetectability will be based on the fact that the added signal does not disturb the statistical properties of the initial image.
This principle gave birth to the paradigm of Natural Steganography : the message is embedded by mimicking a natural noise and the generated image (the stego image) thus displays the properties of an ordinary image, ensuring a high practical security.
This manuscript features contributions that extend this method to JPEG images and also derives from it a more generic insertion scheme.

Keywords: Steganography, Steganalysis, Security