

Notes et commentaires au sujet des conférences de S. Mallat du Collège de France (2024)

Apprentissage et génération par échantillonnage aléatoire

J.E Campagne*

Janv. 2024; rév. 25 mars 2024

*Si vous avez des remarques/suggestions veuillez les adresser à `jeaneric DOT campagne AT gmail DOT com`

Table des matières

1 Avant-propos	6
2 Séance du 17 Janv.	6
2.1 Introduction	6
2.2 Digressions autour de deux exemples emblématiques avec GPT-4	9
2.3 Quel est le cadre mathématique?	16
2.4 Lien entre entraînement supervisé et non-supervisé	18
2.5 Quelles sont les problématiques?	20
2.6 Panorama du domaine mathématique	21
2.6.1 La modélisation	21
2.6.2 L'échantillonnage	24
2.7 Quelques illustrations	26
2.8 Plan du cours	31
3 Séance du 24 Janv.	31
3.1 Modèles aléatoires: pourquoi?	32
3.2 La méthode de Monte Carlo	34
3.2.1 Calcul de volumes	36
3.2.2 Pourquoi ne pas faire un calcul déterministe?	38
3.2.3 Méthode de quasi-Monte Carlo	40
3.2.4 Limites de la méthode de Monte Carlo	41
3.3 Modélisation de probabilités, apprentissage/inférence	42
3.3.1 Petit exemple: détection de spams	42

3.3.2	Apprentissage: la meilleure approximation	43
3.3.3	L'Inférence	45
3.3.4	Quelle type de modélisation choisir? Exemples	45
4	Séance du 31 janv.	52
4.1	Schéma causal et non-causal	52
4.2	Champ de Markov	54
4.2.1	Définition et propriété	54
4.2.2	Deux exemples génériques	57
4.3	Propriétés d'indépendance	61
5	Séance du 7 Fév.	68
5.1	Introduction	68
5.2	Apprentissage de paramètres	70
5.2.1	Les cadres de Fisher et Shannon	70
5.2.2	Optimisation pour trouver θ^*	72
5.3	Le cas linéaire	74
5.4	Score Matching	77
5.5	Etude d'un exemple: potentiel ϕ^4	79
6	Séance du 14 Fév.	82
6.1	La distance en variation totale	82
6.2	Le Hessien de D_{KL} dans le cas linéaire	85
6.3	La divergence de Fisher (Score Matching)	87
6.4	Réécriture de $I(p, p_\theta)$	87

6.5	Exemple: le cas linéaire de l'énergie	89
6.6	Convergence du score matching	90
6.7	Un exemple illustratif du problème du Score Matching	92
6.8	Conditions d'usage du Score Matching	93
7	Séance du 28 Fév.	99
7.1	Introduction	99
7.2	Le cas à 1 dimension	101
7.3	Ergodicité d'une transformation déterministe	102
7.4	Mesure uniforme sur $[0, 1]$	104
7.5	Techniques de "Rejet"	109
7.5.1	Version simple	109
7.5.2	Version à partir de $q(x)$	109
7.5.3	Probabilité d'acceptance	111
7.6	Échantillonnage d'importance	112
7.7	Au-delà du "Rejet": la progression des idées	114
7.8	Chaîne de Markov	116
8	Séance du 6 Mars	118
8.1	Chaînes de Markov: exemple	119
8.2	Loi invariante	121
8.3	Conditions de réversibilité	122
8.4	Convergence vers la loi invariante	124
8.5	Point de vue d'analyse spectrale de la mesure invariante	128
8.6	Algorithme de Metropolis-Hasting	129

9 Séance du 13 Mars	134
9.1 Équation d'Ornstein-Uhlenbeck	135
9.2 Problème du débruitage	138
9.3 Réseau de débruitage	139
9.4 Transition entre mémorisation et généralisation	141
9.5 Ouverture vers quelques pistes de recherche	145
9.5.1 Retour sur le débruitage: bases d'ondelettes	145
9.5.2 Réseaux débruiteurs: que font-ils?	148
9.5.3 Résultats d'expérimentations numériques	150
9.5.4 Réflexions sur le modèle génératif par score diffusion	154
9.5.5 Étude de cas: la turbulence	155
9.6 Conclusion de cette année	158
10 NDJE. Quelques ajouts personnels	160
10.1 HMC: Hybride/Hamiltonian Monte Carlo	160
10.2 Normalizing Flows	166

1. Avant-propos

Avertissement: *Dans la suite vous trouverez mes notes au style libre prises au fil de l'eau et remises en forme avec quelques commentaires ("ndje" ou bien sections dédiées). Il est clair que des erreurs peuvent s'être glissées et je m'en excuse par avance. Vous pouvez utiliser l'adresse mail donnée en page de garde pour me les adresser. Je vous souhaite une bonne lecture.*

Veuillez noter que sur le site web du Collège de France vous trouverez toutes les vidéos des cours, des séminaires ainsi que les notes de cours non seulement de cette année mais aussi des années précédentes¹.

Je tiens à remercier l'ensemble de l'équipe du Collège de France qui réalise et monte les vidéos sans lesquelles l'édition de ces notes serait rendue moins confortable.

Notez également que S. Mallat² donne en libre accès des chapitres de son livre "*A Wavelet Tour of Signal Processing*", 3ème édition. ainsi que d'autres matériels sur son site de l'ENS.

Cette année 2024 c'est la septième du cycle de la chaire de la Science des Données de S. Mallat, le thème en est: **Apprentissage et génération par échantillonnage aléatoire**.

Enfin, dans le repository GitHub³, j'ai mis des notebooks d'applications numériques illustrant le cours depuis 2022. Autant que faire se peut, les notebooks peuvent être exécutés sur Google Colab.

2. Séance du 17 Janv.

2.1 Introduction

Le thème de cette année **Apprentissage et génération par échantillonnage aléatoire** colle parfaitement avec l'actualité des années précédentes qui ont été marquées par la mise

-
1. <https://www.college-de-france.fr/chaire/stephane-mallat-sciences-des-donnees-chaire-statutaire/events>
 2. <https://www.di.ens.fr/~mallat/CoursCollege.html>
 3. https://colab.research.google.com/github/jecampagne/cours_mallat_cdf

en ligne à partir de 2022 de l'assistant conversationnel ChatGPT⁴ conçu par la société américaine OpenAI spécialisée dans le domaine de l'intelligence artificielle. ChatGPT est conçu autour d'un **modèle de langage** encodé dans une architecture nommée **transformer**⁵ comptant un nombre colossal de paramètres ($> 10^{18}$).

Avec GPT-3.5, la version gratuite de ChatGPT, ce dernier a répondu à la requête "*Pouvez vous me faire une petite liste de ce que l'on peut faire avec votre version 3.5? Vous formulerez votre réponse en Latex en utilisant l'environnement itemize*" comme ceci⁶:

- **Répondre aux questions :** GPT-3.5 peut fournir des réponses à des questions générales en se basant sur les informations qu'il a apprises au cours de son entraînement.
- **Générer du texte créatif :** Il peut créer des histoires, des poèmes, des scripts, et d'autres formes de texte créatif en fonction des instructions que vous lui donnez.
- **Assistance à la rédaction :** Vous pouvez utiliser GPT-3.5 pour vous aider à rédiger des articles, des essais, des rapports, etc., en fournissant des informations de base.
- **Programmation :** Vous pouvez demander à GPT-3.5 de générer du code source pour des tâches de programmation spécifiques en lui fournissant des directives claires.
- **Traduction de texte :** Il peut aider à traduire des phrases ou des paragraphes d'une langue à une autre.
- **Simulations de conversation :** GPT-3.5 peut simuler des conversations avec des personnages fictifs ou même aider à pratiquer des dialogues pour des situations réelles.
- **Résolution de problèmes mathématiques :** Vous pouvez lui demander de résoudre des problèmes mathématiques ou de générer des étapes pour les résoudre.

4. <https://openai.com/blog/chatgpt>

5. NDJE: architecture liée au mécanisme d'attention dont l'implémentation date de 2017: <https://arxiv.org/pdf/1706.03762.pdf>

6. NDJE: j'ai enlevé le code Latex fournit pour ne donner que le rendu final.

- **Génération d'idées :** Il peut être utilisé pour générer des idées pour des projets créatifs, des campagnes marketing, etc.
- **Questions de trivia :** Vous pouvez lui poser des questions de trivia et il tentera de fournir des réponses basées sur les informations qu'il a apprises.
- **Soutien à la prise de décision :** GPT-3.5 peut aider à explorer différentes options et à fournir des informations pour faciliter la prise de décision.

Il est important de noter que GPT-3.5 fonctionne en générant des réponses basées sur des modèles linguistiques pré-entraînés et n'a pas une compréhension ou une conscience réelle. Il est essentiel de formuler des instructions claires pour obtenir les résultats souhaités.

NDJE: Maintenant, il faut avoir un regard critique sur ces fonctionnalités ainsi énoncées. En particulier, lorsqu'il "parle" de *Résolution de problèmes mathématiques*. Sans entrer dans les détails qui nous mèneraient à dévier du propos, quand on lui pose la question "*Bonjour, avez-vous quelques notions utiles sur les matrices unipotentes, svp?*", ChapGPT est très allant mais en voulant nous proposer 5 propriétés, 3 d'entre elles étaient manifestement fausses. De la "discussion" engagée pour lui montrer ses erreurs, nous constatons que ChapGPT essaye de les corriger en reformulant les démonstrations, mais finit par s'entremêler les "neurones" comme un élève qui n'a pas bien digéré son cours. Donc, on voit là une limitation du modèle sous-jacent où le rendu peut être revu par une personne experte. Bien d'autres expériences ont confirmé ces limites. Mais force est de constater que cette version gratuite à fait couler beaucoup d'encre. Cela a permis notamment à tout le monde d'utiliser un tel assistant, et d'en discuter les tenants et aboutissants théoriques, philosophiques, voire d'élaborer des schémas d'organisation du travail dans tous les domaines tels que l'éducation des enfants, la formation des étudiants, et les applications dans les services et le monde industriel. Une révolution est en marche.

La version payante (20\$/mois) de GPT-4 (Mars 2023) offre d'autres fonctionnalités encore plus époustouflantes telles que⁷:

- **Interpréter** des images et comprendre leurs contextes;
- **Générer** des images à partir d'un simple descriptif écrit en langage naturel;

7. NDJE: la liste n'est pas générée par ChatGPT-3.5 qui répond "*Je tiens à préciser qu'à ma connaissance (avec une dernière mise à jour de formation en janvier 2022), il n'y a pas de version GPT-4 disponible...*

- **Générer** des textes plus créatifs (chansons, scénarios de film...);
- **Résumer** des textes très longs jusqu'à 25 000 mots;
- **Créer** un site web ou une application à partir d'un simple croquis;
- **Générer** du code pour créer un jeu vidéo comme Snake...

Le terme **générer** est omniprésent et l'on voit bien le lien avec le cours de cette année. Cependant, outre cette capacité de générer des images, des textes, la capacité d'**interprétation** est toute aussi stupéfiante. Ces deux fonctionnalités posent des problèmes. Notons que la possibilité de génération permet des dérives bien illustrées dans les médias. Mais dans le cadre du cours, ces modèles de langage bousculent les idées reçues. Non seulement, on peut s'interroger de "comment" et "pourquoi" ces modèles sont aussi performants même s'ils peuvent faire des erreurs, mais aussi ils bousculent la façon dont on divise traditionnellement les tâches en apprentissage statistique.

En effet, il est habituel de distinguer (on peut se référer aux cours antérieurs comme 2018, 2019 par exemple):

- l'apprentissage **supervisé** où l'on a des **données** x , et l'on veut trouver une **réponse** y : ex. la classe d'un objet pour une tâche de **classification**, ou bien la valeur d'une grandeur pour une tâche de **régression**. La supervision réside dans le fait que pour réaliser une tâche, on dispose d'une base d'entraînement $\{x_i, y_i\}_{i \leq n}$ où l'on connaît la réponse y_i pour chaque donnée x_i .
- l'apprentissage **non-supervisé** pour lequel il s'agit de **construire un modèle des données** x . Nous verrons que l'on se pose alors la question de la modélisation de la **densité de probabilité** $p(x)$ qui permet de **générer** de nouvelles données par **échantillonnage**.

Or, on constate que ces deux types d'apprentissage tendent à être utilisés de manière complémentaire surtout dans l'approche des transformers.

2.2 Digressions autour de deux exemples emblématiques avec GPT-4

S. Mallat nous donne deux exemples de résultats de GPT-4 à la suite de sollicitations de sa part. Le premier exemple concerne une **génération d'image**. Il a soumis la requête (en français): *Image illustrant un cours au Collège de France sur: "Apprentissage*

et Génération par Échantillonnage Aléatoire". Le résultat est visible sur la figure 1. Force est de constater que l'image est de très bonne qualité et surtout qu'elle a **beaucoup de contenu sémantique indirecte**. On a clairement un modèle du monde qui particularisé par la requête donne selon ce modèle un cours au Collège de France. On peut y voir une sorte de temple de la connaissance avec des statues grecques, un amphithéâtre où l'on peut distinguer une certaine diversité du public avec des caractéristiques de personnes plus ou moins âgées (ex. couleur des cheveux) ce qui peut différer par rapport au public présent. Le genre masculin est clairement majoritaire ce qui par contre reflète assez bien la réalité. On constate qu'il y a un biais langagier puisque le titre du cours a été traduit en anglais ce qui n'était pas demandé, de plus on distingue clairement un tableau noir avec des formules/graphiques qui dénote un cours de mathématiques ce qui n'était pas non plus directement suggéré. Enfin, la personne qui donne le cours est un homme blanc portant des lunettes ce qui reflète assez la situation présente. On pourrait bien entendu détailler encore plus cette image qui est extrêmement riche en information sous-jacente (noter la mixité des intervenants derrière l'orateur). Comment de telles capacités sont possibles?

Le second exemple concerne la capacité d'**interpréter l'image** (Fig. 2) prise par un téléphone portable de 16M pixels (8-bits RGB) et téléchargée sur le site à partir de l'interface Dari et pour laquelle on donne la requête simple en langage naturel: "*Peux-tu me décrire cette image?*". La réponse faite est la suivante (*NDJE. sauf erreur(s) de retranscription*):

L'image montre une pièce bien aménagée avec des meubles en bois. Au centre, il y a une télévision éteinte posée sur un meuble bas avec des tiroirs de couleur claire. De chaque côté du meuble, des placards en bois sont intégrés dans le mur, peints en jaune avec des bordures orange et vertes. À droite, une bibliothèque remplie de livres est adossée au mur, et il semble y avoir une porte ou une armoire adjacente à cette bibliothèque. La pièce a l'air d'être éclairée par une lumière douce et on distingue un bout de couverture ou de drap, ce qui suggère que la photo a été prise depuis un lit.

A noter que ce type de tâche de décrire une image n'est pas nouvelle, elle date des premiers travaux sur la vision par ordinateur. Il y a 5-6 ans, nous dit S. Mallat, il y

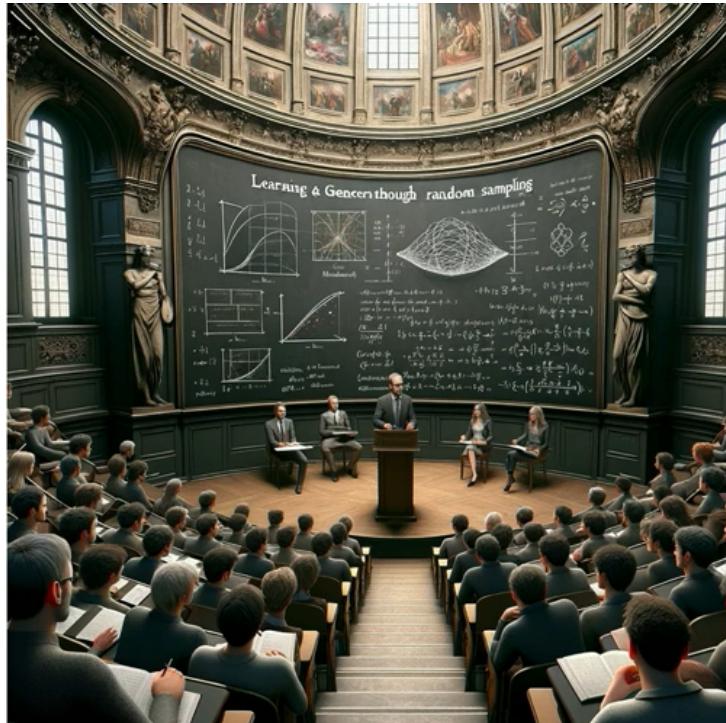


FIGURE 1 – Génération d'une image par Chat GPT-4 à partir d'une requête simple formulée en langage naturel. L'image est accompagnée d'un petit texte expliquant ce qu'est le Collège de France.



FIGURE 2 – Image dont on demande à ChatGPT-4: "Peux-tu me décrire cette image?".

avait pour de telle requête quelques labels qui étaient donnés comme réponse (ex. "image de pomme"). Là, avec GPT-4, on ne joue clairement pas dans la même catégorie, si on utilise un vocabulaire sportif. Non seulement la réponse formulée est grammaticalement correcte, mais ce qui est impressionnant c'est l'exactitude de tous les détails mentionnés, et la finesse de l'analyse avec le contexte sous-jacent pour formuler la dernière proposition.

Le modèle de langage entraîné est donc capable de générer et d'interpréter mais aussi de répondre à toute sorte de questions. C'est tout à fait impressionnant même s'il peut y avoir des erreurs, et l'on comprend les multiples questions que de telles capacités soulèvent, et la possibilité d'usages multiples. On constate un **écart considérable** entre d'un coté ce qui est produit par l'**industrie** qui utilise de plus en plus de tels systèmes qui se développent pour être adaptés aux problèmes spécifiques de tel ou tel domaine, et de l'autre la **recherche** qui se fait en parallèle sur les algorithmes et sur la compréhension des performances.

Coté industriel, pour entraîner de tels systèmes, il faut non seulement acquérir des données, les organiser, parfois les trier, éliminer, sélectionner, etc qui demande beaucoup de ressources humaines, mais aussi de part leur gigantisme, ces réseaux ne peuvent être entraîner que dans des centres spécialisés demandant des ressources matériels très impor-

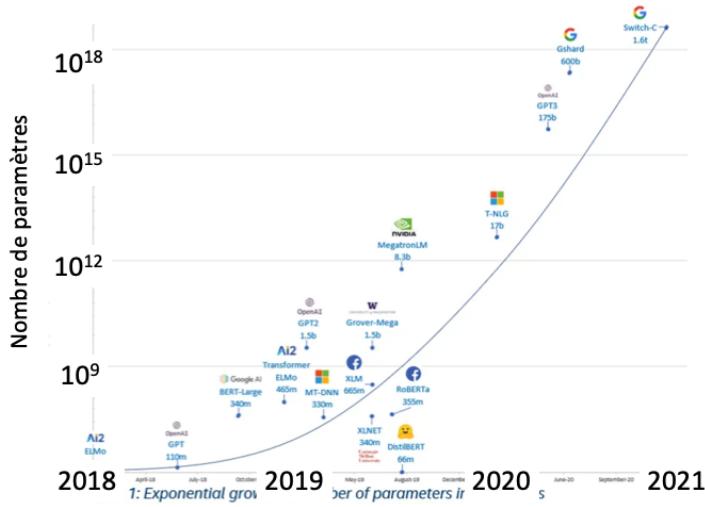


FIGURE 3 – Évolution du nombre de paramètres des architectures AI en fonction du temps.

tantes⁸ avec des coûts de fonctionnement tout aussi considérable. S. Mallat nous dit que l'on a même plus les chiffres car ils deviennent sensibles. Par exemple, Gemini⁹ de Google et GPT-4 sont constitués d'environ 1 trillon de paramètres soit 10^{18} (Fig 3). Et Serge Escalè du site ITSocial indique que "la durée d'apprentissage d'OpenAI est d'environ 6 mois. Ainsi les 10.000 GPU (processeurs graphiques) V100 de Nvidia qui servent au calcul consomment 7.200 Mwh", ce qui correspond à un coût électrique qui va avec, sans parler du coût environnemental.

Notons que d'un point de vue ingénierie (management), les sociétés développant de telles architectures se réorganisent pour optimiser leurs développements afin de garantir leur pérennité. Mais cela entraîne également un changement dans la façon dont les performances de ces modèles sont diffusées: voir par exemple le rendu de la comparaison de Gemini Ultra et GPT-4 par D. Hassabis (Pdg et co-fondateur de Google Deepmind) et S. Pichai (Pdg de Google et Alphabet) sur le site <https://blog.google/technology/ai/google-gemini-ai/#sundar-note>. Comment peut-on vérifier ce qui est avancé? Jusqu'à maintenant, nous dit S. Mallat, les algorithmes étaient dans le domaine public, non pas par esprit d'ouverture mais tout simplement parce que **la valeur résidait dans les données**,

8. NDJE: voir <https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8>

9. <https://deepmind.google/technologies/gemini>

donc il était intéressant de laisser se développer les algorithmes dans domaine public. A partir de maintenant, **la valeur est dans les algorithmes**. C'est ce qui se passe dans les modèles de langage (assistant conversationnel) où l'on a accès (payant pour les versions les plus récentes) à une simple interface. Le cas d'OpenAI est en quelque sorte exemplaire pour cette société qui comme son nom l'indique avait la volonté initiale de produire du logiciel libre profitant à toute l'humanité. Après mars 2019, il en va tout autrement car la société doit donner des dividendes aux actionnaires. Le jeu change donc du tout au tout, mais en quelque sorte c'est un développement naturel (*business model*) d'une société qui a besoin de capitaux afin de conserver son leadership et de supporter la concurrence.

Tout cela pose quand même des problèmes quand on fait de la recherche sur de telles architectures de modèle de langage. Primo, on n'a pas en général accès aux mêmes ressources (financières, humaines, matérielles), et secundo imaginons qu'un développement vous permette d'obtenir des résultats un peu meilleurs pour de la classification d'images de type ImageNet¹⁰, votre résultat passera totalement inaperçu, simplement parce que la problématique est bien moins difficile que celle mise en jeu dans les deux exemples précités. Donc, **quelle est la pertinence des recherches "académiques"**, quel est l'impact d'un chercheur, comment peut-il/elle faire avancer la connaissance? Ce sont des questions importantes, et la bonne nouvelle c'est qu'il y a beaucoup d'axes de recherche possibles.

Par exemple, en abordant des domaines où la ressource en données n'est pas aussi importante que celle utilisée pour entraîner les modèles de langage, et où les algorithmes doivent intégrer plus d'information *a priori*¹¹ et où les architectures sont beaucoup plus petites. Une autre axe de recherche est en quelque sorte un transfert de savoir faire. En effet, si les architectures des *transformers* sont adéquates pour traiter le langage, il est intéressant de les essayer dans d'autres domaines. Par exemple, la chimie afin de produire de nouvelles molécules, ou la pharmacologie afin d'obtenir de nouveaux médicaments, la physique des matériaux, la mécanique des fluides, la physique de l'infiniment petit et de l'infiniment grand et bien d'autres encore, tous ces domaines profitent des qualités de ces nouvelles architectures.

Enfin, il y a un domaine de recherche qui fait l'objet de la série de cours depuis 2017, c'est **d'essayer de comprendre**. D'où viennent les phénomènes émergents? Les systèmes

10. <https://www.image-net.org/>

11. NDJE: voir Cours 2020 dont le sujet était *l'Information a priori en grande dimension*.

à la ChatGPT/Gemini sont entraînés de manière non-supervisée. Typiquement avec les textes collectés sur le Web, on peut dans un premier temps, faire un entraînement de type "donne moi le mot manquant", puis "la phrase suivante", etc. Mais actuellement, l'entraînement est plus complexe avec des phases non-supervisées et des phases de renforcement, afin de contrôler des aspects liés à "l'éthique" et/ou la "conformité à la loi" par exemple. Mais, comprendre ce qu'il se passe dans ces architectures une fois entraînées est un sujet difficile. S. Mallat indique le cours de Benoit Sagot qui traite des aspects d'ingénierie des modèles de langage¹².

Le cours de cette année et de l'an prochain à vocation de construire les mathématiques qui aident à comprendre ce qui se passe lorsque l'on veut faire de la modélisation à des fins de génération de nouvelles données. Les mathématiques nous dit S. Mallat s'établissent dans le temps long donc patience.

NDJE. Avant de continuer, j'aimerai aborder un sujet qui bien entendu pourrait être développé à l'envie. Vous pouvez vous demander à ce stade: quelle est la pertinence de ces notes de cours? En effet, pourquoi ne pas faire visionner le cours de S. Mallat à GPT-4/Gemini et lui demander d'en faire une transcription, un résumé, en préciser/illustrer un point, etc. Je ne doute pas d'ailleurs qu'un jour cela sera fait. La valeur ajouté réside non pas dans la transcription écrite du cours purement et simplement. Il s'agit d'un rendu qui s'il me paraît clair à moi-même aura sans doute un écho pour la lectrice ou le lecteur. Par exemple, vous verrez parfois des articles mis en référence, des illustrations différentes de celles présentées, ou bien des digressions qui tendent d'expliquer tel ou tel point que S. Mallat n'a pas le temps d'évoquer. De plus, ayant écrit les notes depuis 2017, je mets en relation ces cours car S. Mallat ne peut chaque année revisiter ce qu'il a fait les années antérieures, ça n'aurait pas de sens. Le repository Github a été mis en œuvre dans le même esprit de donner cette fois des illustrations de notions/algorithmes sous forme de codes python avec des expérimentations numériques légères, en relation avec les points théoriques du cours. Il est clair qu'un modèle capable de pouvoir répondre à toutes les requêtes peut exactement faire le même travail. Mais le parallèle peut être fait entre avoir tous les livres de mathématiques possibles à disposition, et la valeur ajoutée d'un(e) professeur(e) qui pour élaborer son cours prend des morceaux choisis avec un certain parti pris, pour en faire une trame cohérente et ingérable par son public. Qui plus est la dépense énergétique

12. <https://www.college-de-france.fr/fr/chaire/benoit-sagot-informatique-et-sciences-numeriques-chaire-annuelle>.

d'un cerveau est de 20 à 40 Watts ce qui n'est rien en comparaison des usages des modèles actuels.

2.3 Quel est le cadre mathématique?

On se place dans un **cadre probabiliste** ce qui a des conséquences immédiates sur les outils utilisés. Cependant, pourquoi choisir un tel cadre? On peut se reporter aux discussions du Cours 2022 Sec. 2.2 et du Cours 2023 Sec. 3.1 et Sec. 3.2. En fait, dans le cas d'un apprentissage supervisé, la réponse y peut être vu comme le résultat d'une fonction déterministe des données ($y = f(x)$) et où $x \in \mathbb{R}^d$ peut même être confiné dans une variété de basse dimension qui tend à simplifier le problème. Donc, le cadre probabiliste est motivé par le choix de la **modélisation** avant tout (apprentissage non-supervisé). Mais en lui-même ce choix de modélisation, n'en est pas un pour les raisons suivantes.

Tout d'abord, on dispose de données pour lesquelles on veut établir un **modèle**, c'est-à-dire que naturellement on se place dans le cadre de l'**estimation statistique**, c'est-à-dire que l'on veut calculer des probabilités.

Cependant plus fondamentalement le point qui fait pencher la balance vers un cadre probabiliste est que l'on se place de part la nature des données en **très grande dimension**. On sait que d est de l'ordre de quelques millions pour des images courantes. Or, on veut estimer des probabilités de type $p(x)$ (ou $p(x|y)$) ce qui est "impossible" en général, à cause de la sacro-sainte **malédiction de la grande dimension**¹³. Pour mémoire elle se traduit par le fait que le nombre d'échantillons nécessaires n croît exponentiellement avec la dimension d . Pour que le problème soit abordable, il nous faut injecter de la connaissance *a priori* en se basant sur le fait que l'émergence doit être le fruit d'une **structuration sous-jacente**.

La notion omniprésente qui permet de s'en sortir est l'**indépendance** (cf. Cours 2022 et 2023). En effet, si une donnée x est un vecteur de dimension d , (x_1, \dots, x_d) et que chaque composante x_j suit une loi p_j alors

$$p(x) = p(x_1, \dots, x_d) = \prod_{j=1}^d p_j(x_j) \tag{1}$$

13. Voir spécifiquement le Cours 2018 mais c'est un leitmotiv des autres cours également.

c'est-à-dire que l'on a remplacé le problème de trouver une fonction à d variables, par la recherche de d fonctions à 1 seule variable. La décomposition du problème difficile en d problèmes plus faciles permet de combattre la malédiction de la dimension.

Ceci dit, cette "totale" indépendance est illusoire dans bien des cas pratiques: vous pouvez vous représenter par exemple les contours d'un objet dans une image, ils mettent naturellement en relation des pixels qui n'ont pas lieu d'être nécessairement des voisins; ou bien vous sentez sans doute actuellement l'influence sur nos vies de tous les jours (ex. prix du pétrole) de décisions prises part des dirigeants de pays qui nous sont très éloignés physiquement. Que faire dans ces cas de figure? On peut toujours écrire¹⁴

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\cdots = p(x_1) \prod_{j=2}^d p(x_j|x_1, \dots, x_{j-1}) \quad (2)$$

Ceci dit les probabilités conditionnelles sont en grande dimension dès que j devient proche de d . La question est de savoir si l'on peut émettre des hypothèses sur la dépendance de x_j vis-à-vis de toutes les variables (x_1, \dots, x_{j-1}) ? Ne peut-on pas invoquer une réduction? Vraisemblablement car par exemple: les pixels concernés par les contours d'un objet ne sont pas dépendant vis-à-vis de tous les pixels de l'image (et vice-versa); de même le prix du pétrole en première approximation dépend de décisions de quelques pays gros producteurs de couper plus ou moins le robinet, et du volume de la demande des pays consommateurs (plus nombreux certes). On sent donc que l'on peut faire des hypothèses sur le niveau d'interdépendance des variables x , ou bien on peut chercher des représentations $x \rightarrow x'$ telles que les variables x' soient moins en interaction que les variables initiales.

L'enjeu est donc dans chaque problème de comprendre les structures de probabilités conditionnelles. Ce qui est remarquable avec les gros modèles de langage précités, c'est qu'en somme ils ne calculent, ni plus ni moins, que des probabilités conditionnelles. Par exemple, pour générer l'image y de la figure 1 pour satisfaire la requête x (la phrase soumise à GPT-4), il a calculé tout d'abord $p(y|x)$ à partir de p trouvée par l'entraînement, puis il a échantillonné la dite probabilité conditionnelle pour donner la réponse. Naturellement, le calcul de la probabilité $p(y|x)$ ainsi que l'échantillonnage se faisant en très grande dimension, la tâche est naturellement très complexe. Donc, il faut comprendre

14. nb. pour alléger les notations, on enlève l'indice des probabilités.

le mécanisme qui permet la réalisation pratique et se poser la question: quelles sont les structures à différentes échelles (cf. Cours 2019 Sec. 3.7) des probabilités conditionnelles qui permettent de combattre la malédiction de la dimension.

Enfin, un point important qui sera discuter dans le cours, concerne **la complexité de calcul en grande dimension**. En effet, pour réaliser le calcul de moyennes, de marginales de probabilités conditionnelles etc, il nous faut procéder aux **calculs d'intégrales** en grande dimension. A priori, le nombre d'opérations doit croître exponentiellement avec la dimension d , sauf si on utilise des méthodes aléatoires dites de **Monte Carlo**¹⁵. Ces techniques consistent non pas à calculer ces intégrales sur des grilles de l'espace \mathbb{R}^d mais plutôt de tirer des échantillons aléatoirement et d'en faire une somme/moyenne. Nous verrons pourquoi on évite la malédiction de la dimension, même s'il n'en reste pas moins vrai que la génération peut être difficile dans des cas pratiques en grande dimension (ex. le cas de distributions multi-modales).

Nous avons donc passer en revue **pourquoi le cadre probabiliste est central et qu'il est directement relié au fait que l'on est contraint à évoluer en grande dimension**.

2.4 Lien entre entraînement supervisé et non-supervisé

Le lien en question n'est pas nouveau car il est découlé de la **perspective bayésienne** comme nous allons le voir.

Concernant l'apprentissage *supervisé* où l'on s'intéresse à obtenir la réponse y à partir de x , dans le cadre probabiliste on désire alors obtenir $p(y|x)$. Si l'on s'intéresse à un problème de classification, le classificateur de Bayes¹⁶ est celui qui en moyenne produit l'erreur minimum, il maximise $p(y|x)$

$$\hat{y} = \operatorname{argmax}_y p(y|x) \quad (3)$$

Par contre concernant l'apprentissage *non-supervisé* dans le cadre probabiliste, on s'intéresse plutôt à la modélisation de la probabilité $p(x)$. Mais il y a un lien avec ce qui

15. NDJE: en 2023 j'avais mis au point dans le notebook Python `Monte_Carlo_Sampling` des exemples de telles techniques. Le notebook est accessible ici https://github.com/jecampagne/cours_mallat_cdf/blob/main/cours2023/Monte_Carlo_Sampling.ipynb.

16. voir ex. Cours 2022 Sec. 2.2 ou/et Cours 2023 Sec. 3.4.2

précède car nous avons la formule de Bayes¹⁷:

$$p(y|x) = \frac{p(x|y)p(y)}{p(x)} \quad (4)$$

dans laquelle on reconnaît $p(y|x)$ la probabilité *a posteriori* (**posterior**), $p(x|y)$ la vraisemblance (**likelihood**), $p(y)$ la probabilité *a priori* (**prior**, ex. la probabilité *a priori* d'être dans une classe plutôt que dans un autre) et $p(x)$ l'**évidence**. Si l'on veut maximiser $p(y|x)$ selon y alors

$$\hat{y} = \operatorname{argmax}_y p(x|y)p(y) \quad (5)$$

mais $p(y)$ est plutôt à voir comme une donnée, et donc dans ce problème le point pratique et délicat est de maximiser la vraisemblance $p(x|y)$.

Schématiquement, on peut alors penser à procéder selon la suite d'opérations suivantes: prendre les données x , les regrouper dans les différentes classes y , et **modéliser chacune des classes**. C'est-à-dire qu'à la fin, on pourrait être en capacité de générer de nouveaux éléments des différentes classes. Mais, on sait qu'en pratique pour certains problèmes, il n'est pas nécessaire de faire ce travail et que l'on peut utiliser une autre approche. Pensez au cas de classifier une image en deux catégories selon qu'elle contient un camion de pompier ou une voiture noire. La modélisation de toutes les images qui comportent soit un camion de pompier soit une voiture noire est *a priori* extrêmement complexe étant donné la diversité des situations dans lesquelles évoluent ces véhicules. En contre partie, il est fort probable que le simple comptage du nombre de pixels rouge et noir suffise à fournir un bon indicateur de la classe de l'image.

En fait, l'**extraction d'éléments discriminants** (*features*) pour un problème posé est une première approche qui peut être très efficace, et participe naturellement à la réduction de dimension. Il est clair que l'approche de la modélisation n'était pas du tout celle mise en œuvre il y a quelques années encore, et donc il y avait deux philosophies/approches différentes. En quelque sorte calculer $p(x|y)$ pour distinguer deux classes peut être considéré comme une approche très brutale. Sauf que dans le cas d'une image, outre de savoir si elle contient un camion de pompier ou une voiture noire, on peut potentiellement poser bien d'autres questions, alors l'approche par modélisation devient beaucoup plus efficace.

C'est bien la nécessité de **répondre à toutes sortes de questions qui impose la**

17. Rappel: $p(x,y)$ la probabilité jointe est égale aux produits $p(y|x)p(x)$ et $p(x|y)p(y)$.

modélisation de $p(x)$, $p(y|x)$. C'est pourquoi les deux domaines d'apprentissage tendent à se rejoindre. **On est actuellement en train d'attaquer des domaines complexes où les raccourcis furent astucieux, comme la mise en œuvre de représentations en basse dimension, ne sont plus du tout efficaces.** Cependant, il faut garder à l'esprit que tous les problèmes ne sont pas de même nature, et dans bien des cas, par exemple ceux pour lesquels obtenir des données en grand nombre est difficile, ces représentations en basse dimension sont très utiles.

2.5 Quelles sont les problématiques?

On peut citer trois types de problèmes qui vont nous intéresser avec d'abord celui de la **modélisation**. Rappelons que nous avons à modéliser $p(x)$ ou bien $p(y|x)$. L'usage de **familles paramétrées** $\{p_\theta(x)\}_\theta$ est utilisé tout naturellement¹⁸.

Cependant, il nous faut choisir une famille de fonctions bien adaptées. C'est un **problème d'approximation**, car il faut pouvoir trouver le meilleur θ , noté θ^* et que p_{θ^*} soit une bonne approximation de la distribution réelle $p(x)$ (que l'on suppose *a priori* exister). La métrique utilisée durant l'apprentissage est la "distance" de Kullback-Leibler dont la définition est la suivante¹⁹:

$$D_{KL}(p||p_\theta) = \int p(x) \log \frac{p(x)}{p_\theta(x)} dx \geq 0 \quad (6)$$

(rappel: si elle est nulle alors $p = p_\theta$). Pour trouver θ^* , il nous faut minimiser $D_{KL}(p||p_\theta)$ sur toutes les valeurs de θ , c'est donc un **problème d'optimisation**. Plus précisément c'est un problème de **maximum de vraisemblance** tout à fait fondamental en estimation statistique introduit par **Ronald A. Fisher** en 1922 (ex. voir Cours 2022 Secs. 2.1 et 2.3).

Enfin, on peut utiliser ce schéma de modélisation et approximation/optimisation afin de réaliser de l'**inférence**. Par exemple dans le domaine du médical, nous avons typiquement des symptômes (les données x) comme toux, nausée, fièvre, vomissement, diarrhée, respiration difficile, palpitations cardiaques, etc, et l'on veut donner une réponse (y) en terme de diagnostic comme covid, grippe, pneumonie, tachycardie, insuffisance de toute

18. NDJE: Voir le Cours de 2022 avec l'approche de R. Fisher

19. Voir aussi le Cours de 2019 Sec. 7.2.3 par exemple.

nature, etc. Dans ce contexte, on peut disposer de bases de données, par exemple la Quick Medical Reference²⁰ qui décrit 573 *diagnostiques*, reconnaît 4 100 *observations de patients et comprend plus de 4 000 liens détaillant les relations causales, temporelles et probables entre les troubles*. Un médecin bien entendu veut pouvoir donner un diagnostique à partir de quelques symptômes de sa patiente ou son patient. Mettons que (x_1, x_2) soient les symptômes, il faut pouvoir calculer $p(y|x_1, x_2)$ et par exemple prendre le y donnant le maximum de probabilité. Maintenant, si l'on dispose de la probabilité conditionnelle $p(y|x_1, x_2, \dots, x_d)$ alors il faut réaliser l'intégrale (marginale)²¹

$$p(y|x_1, x_2) = \int p(y|x_1, x_2, \dots, x_d) \prod_{j=3}^d dx_j \quad (7)$$

Donc, faire une inférence à partir du modèle, va nécessiter de réaliser des calculs d'intégrale (en grande dimension) de probabilités conditionnelles, ce qui nous fait le lien avec les problématiques de tirages Monte Carlo. Ce faisant qui dit Monte Carlo dit effectuer un **échantillonnage**, et donc dit être capable de faire de la **génération**.

2.6 Panorama du domaine mathématique

2.6.1 La modélisation

Quand on aborde les problèmes cités dans les sections précédentes, il y a beaucoup de concepts différents, et nous allons prendre le temps de les aborder. Dans un premier temps, les techniques **Monte Carlo** nous occuperons. Ce n'est certes pas forcément évident, mais nous verrons que cela se fait bien. En revanche, ce qui est plus compliqué, c'est la **modélisation**.

Cependant, il est clair que la modélisation de $p(x)$ en 1D est beaucoup plus facile. On peut voir cela comme un problème d'interpolation. Dans ce contexte, le Cours de 2021 donne quelques points d'entrée. Par exemple à travers toute le domaine de l'analyse en basse dimension des bases orthogonales à choisir suivant les régularités de la fonction $p(x)$ à approximer. Le vecteur Θ est constitué par les coefficients dans la base choisie. Où

20. <https://www.nlm.nih.gov/research/umls/sourcereleasedocs/current/QMR/index.html>

21. nb. pour simplifier on a ordonné les variables $(x_i)_i$.

le problème devient compliqué c'est quand $x \in \mathbb{R}^d$. En effet, en essayant de considérer des fonctions dans l'espace $L^2(\mathbb{R}^d)$, nous en revenons à la malédiction de la dimension, à savoir que le nombre d'éléments de base devient beaucoup trop important. Il faut procéder autrement et pour se faire nous allons utilisé un outil déjà mis en œuvre surtout dans le cours de 2023, à savoir des **probabilités de Gibbs**:

$$p_\Theta(x) = Z_\Theta^{-1} \exp\left\{-\sum_k \theta_k \phi_k(x)\right\} = Z_\Theta^{-1} e^{-\Theta^T \Phi(x)} = Z_\Theta^{-1} e^{-U_\theta(x)} \quad (8)$$

avec $\Phi(x) = (\phi_k(x))_{k \leq K}$. La constante Z_Θ , **la fonction de partition de Gibbs**, assure la normalisation de la probabilité. On note par analogie avec la Physique Statistique, $U_\theta(x) = \Theta^T \Phi(x)$ l'**énergie interne** et $F_\theta = -\log Z_\theta$ l'**énergie libre**. **L'idée sous-jacente est qu'à travers les fonctions ϕ_k (et donc U_θ) on peut y injecter beaucoup d'information a priori qui rend compte de structures.**

Dans ce cadre, **les champs de Markov** vont être un sujet d'étude important, car c'est avec ces derniers on peut concevoir un premier type de modèles qui permettent d'appréhender la modélisation en grande dimension. En fait, avec les champs de Markov, on privilégie les interactions **locales** ce qui casse la malédiction de la dimensionnalité. Nous verrons le **théorème de Hammersley-Clifford** qui donne les conditions nécessaires et suffisantes sous lesquelles une distribution de probabilité strictement positive peut être représentée comme des événements générés par un champ de Markov. Ce théorème fait le lien entre l'indépendance conditionnelle et la façon d'écrire $p(x)$ sous forme d'une distribution de Gibbs, avec une expression particulière qui reflète cette indépendance. Ceci dit cette hypothèse de localité n'est pas suffisante en soit car nous avons cité précédemment des exemples où manifestement les interactions non-locales ne sont pas négligeables. Par contre, on va essayer de trouver des transformations qui vont rendre cette hypothèse de localité plus viable.

Au-delà des représentations qui nous placent dans un cadre d'interactions locales, il y a la notion de problèmes **multi-échelles** qui est importante (Fig. 4). En sous-jacent, l'hypothèse est celle d'une **organisation hiérarchique** du monde. C'est en autre assez vrai en Physique quand on regarde les structures à petites, moyennes et grandes échelles, c'est-à-dire de la Physique des Particules élémentaires à la Cosmologie en passant par les échelles intermédiaires du vivant. Il y a à toutes les échelles une organisation manifeste et

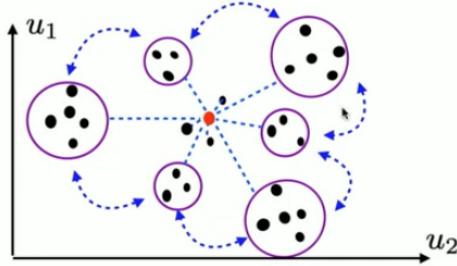


FIGURE 4 – Schématisation des interactions locales entre l’agent rouge et ses voisins, et la prise en compte des interactions à plus grandes échelles par regroupement d’agents. Par ce biais on obtient une modélisation en $O(\log d)$ termes.

l’on peut dans une certaine mesure étudier chaque échelle indépendamment au moins dans une première approximation. Mais, cette hiérarchie peut également servir d’approximation de l’évolution des agents d’un réseau social par exemple. Le concept fondamental qui a émergé de la Physique est le **Groupe de Renormalisation**²². Notons que nous n’aborderons pas ce domaine dans le cours de cette année.

Si on veut complexifier la modélisation de $U(x)$ plus avant, on peut utiliser les **réseaux de neurones**. Une remarque en passant: les réseaux de neurones s’inscrivent dans la démarche de modélisation. En cela les réseaux de neurones ne sont pas "à part" comme des intrus. Et s’ils sont efficaces, alors les idées mises en avant ci-dessus doivent ressortir d’une manière ou d’une autre. En fait, c’est l'**architecture** qui est porteuse des concepts comme la **hiérarchie** (ex. architecture U-Net), les **interactions locales** à travers des filtres convolutionnels de petites dimensions, et les **non-linéarités** renforcent la complexité de la modélisation.

Enfin, une idée supplémentaire qui a permis l’élaboration des grands modèles de langage, réside dans les **transformers** avec la notion d’**attention**. Quand on considère des champs de Markov, on regarde les interactions locales. Or, on sait que dans certains cas de figure, ce n’est pas suffisant car une information lointaine peut se propager et donner une interaction du même ordre de grandeur (voire plus grande). D’autres exemples que ceux déjà invoqués pour illustrer le propos: une petite paille dans un matériau peut se propager et briser la structure; une négation dans une phrase peut changer le sens d’un

22. NDJE: voir la digression Sec. 2.5 du Cours de 2023.

paragraphe entier (il peut en être également avec la place d'une virgule), etc. Donc, afin de capturer ce genre de phénomène, il est nécessaire de regarder au-delà des voisins. C'est en fait tout à fait naturel. S. Mallat nous décrit la façon dont le cerveau fournit une image d'une scène à partir des mouvements incessants et imperceptibles des yeux: nous réalisons en permanence des mouvements rapides de nos yeux qui permettent de placer les éléments importants de notre environnement à l'intérieur de la fovéa qui perçoit finement les détails d'une scène mais qui est très petite (1,5mm de diamètre au centre de la macula). Ces mouvements rapides ne sont pas aléatoires, se sont **les points d'attention** où il y a de l'information importante pour comprendre la structure de la scène. Donc, la notion d'attention a toujours été au cœur de la psychologie cognitive, ce qui est nouveau c'est que l'on a trouvé un moyen de l'implémenter et de la rendre effective. Ces transformes ont permis de réaliser en quelque sorte une **généralisation de la notion de voisinage** qui n'est plus un *a priori* (comme en fixant une grille et une distance) mais **qui s'apprend**.

2.6.2 L'échantillonnage

En parallèle de la modélisation des probabilités, nous avons vu dans les sections précédentes à quel point pouvoir générer des nouvelles données par **échantillonnage** est important.

De même que nous sentons que les méthodes pour modéliser $p(x)$ en 1D sont bien comprises, l'échantillonnage en 1D, même si ce n'est pas évident, n'est pas difficile. Dans un premier temps, on s'attache à créer une **variable aléatoire uniforme** à partir d'un **système chaotique**, ce qui fait appel à la notion d'**ergodicité** déjà abordée durant le Cours de 2023 (ex. Sec. 6.2). Puis, il faut transformer la distribution uniforme pour obtenir la distribution souhaitée. Dans ce cas, il y a des méthodes et algorithmes comme par exemple l'*inversion de la fonction de répartition*, la réjection-acceptation (*hit and miss*) ou encore l'*importance sampling*²³. Cette étape nous permettra dans le cours de poser des concepts importants.

Ceci dit, il nous faut aborder le cas difficile de la grande dimension. L'outil central que nous allons utiliser, également aborder dans le cours de 2023 (ex. Sec. 6.4 et Sec. 7), est constitué des **chaînes de Markov**. L'idée est de partir d'une distribution de probabilité p_0

23. NDJE: voir le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/cours2023/Monte_Carlo_Sampling.ipynb.

facile à générer (ex. distribution uniforme, gausienne,...), puis de faire opérer un **transport** (T) qui itéré doit nous mener à la distribution cible. Ainsi, à partir de x_0 un échantillon de p_0 , on itère pour obtenir $x_t = T^t(x_0)$, et si on a bien choisi T alors x_∞ est un échantillon de p .

En fait, si **la distribution p est un invariant** (point fixe²⁴) du transport et si la convergence est rapide, alors la génération d'échantillons selon $p(x)$ est réalisable en pratique. Nous étudierons un certain nombre d'algorithmes (**Metropolis**, **Metropolis-Hastings**²⁵ et **l'échantillonnage de Gibbs**,...) dont leurs convergences. Pour se faire, il est plus facile de voir t comme un *temps continu*. Ce passage du discret au continu nous permet en effet, d'utiliser beaucoup plus d'outils mathématiques. Nous obtiendrons ainsi des **équations différentielles stochastiques**, dont celle de Paul Langevin (1872-1946) et celle de Adriaan Fokker (1887-1972) et Max Planck (1858-1947), toutes deux élaborées pour comprendre le mouvement brownien.

Ces équations sont en quelque sorte des descentes de gradient de l'énergie ($\nabla_x U(x)$) auxquelles on ajoute un bruit. L'analyse nous apporte alors des résultats très jolis où la convergence est caractérisée par les constantes de "log-Sobolev". L'objet mathématique qui sort de cette analyse est le **Hessien**²⁶ de U . Ce n'est pas étonnant, en effet rappelons que l'on cherche à minimiser $U_\theta(x)$ (augmenter $p(x)$, cf. maximum likelihood), d'où la descente de gradient dont la convergence est d'autant plus rapide que le Hessien est bien conditionné²⁷.

Afin de se rapprocher des grands modèles génératifs, il y a les algorithmes de "**Score Diffusion**"²⁸. En bref, cette technique d'échantillonnage est une adaptation de celle de Langevin part une méthode homotopique, c'est-à-dire que l'on fait évoluer continument une distribution initiale vers une distribution finale (Fig. 5). Par exemple, on commence par une image contenant beaucoup de bruit que l'on réduit progressivement. Et au lieu de minimiser la divergence de Kullback-Leibler, on optimise l'**Information de Fisher** (Cours

24. Voir Cours 2023 Sec. 6.5.

25. Voir le notebook de la note de bas de page 23.

26. voir par ex. Cours 2022 Sec. 3.6.2 et Sec. 4.2.

27. Voir le théorème (Th. 4) exposé au Cours 2022 Sec. 3.6.2

28. NDJE: Voici deux références pour les personnes intéressées: J. Sohl-Dickstein, E. Weiss, N. Maheswaranathan, and S. Ganguli, "Deep unsupervised learning using nonequilibrium thermodynamics," in International Conference on Machine Learning. PMLR, 2015, pp. 2256–2265 (arXiv:1503.03585), et Chang, Ziyi; Koulieris, George Alex; Shum, Hubert P. H. (2023). "On the Design Fundamentals of Diffusion Models: A Survey". arXiv:2306.04542

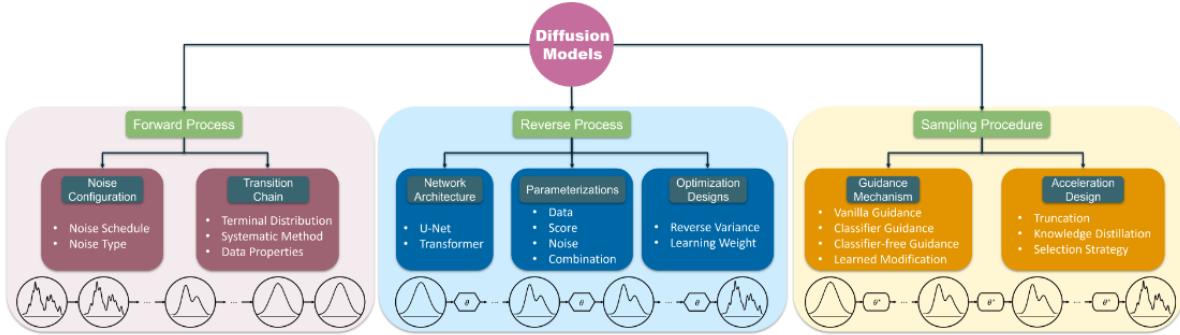


Fig. 1. The overview of diffusion models. The forward process, the reverse process, and the sampling procedure are the three core components of diffusion models, which are responsible for adding noise, training networks, and generating samples, respectively.

FIGURE 5 – Vue d’ensemble des modèles de diffusion. Le processus direct, le processus inverse et la procédure d’échantillonnage sont les trois composantes principales des modèles de diffusion qui sont respectivement chargées de l’ajout de bruit, de l’entraînement des réseaux et de la génération d’échantillons (*source arxiv:2306.04542*).

2022 Sec. 5.3) par la méthode du "**score matching**"²⁹. Il se trouve que cette méthode est beaucoup plus efficace et plus rapide mais cela ne marche que si U_θ satisfait certaines conditions.

2.7 Quelques illustrations

NDJE. A ce stade du cours S. Mallat nous présentent des résultats au video-projecteur. Pour certains d’entre-eux je vous renvoie à la vidéo mise en ligne sur le Collège de France (après 1:04:18).

Tout d’abord, S. Mallat nous fait entendre des trames sonores, ou voir des images de turbulence faisant le lien avec le Cours de 2023 Sec. 2.9. Cela illustre le fait que les **modèles gaussiens** ($U_\theta = \frac{1}{2}x^T C^{-1}x$) **ne sont pas capables de capturer des structures** (en audio il s’agit de l’intermittence). Ceci dit aller au-delà de ces modèles simples n’est pas évident. S. Mallat nous relate qu’il y a 30 ans la totalité des modélisations étaient gaussiennes de base et la question était d’étudier les petites déviations.

29. NDJE: le score étant $\nabla_\theta \log p_\theta(x)$.

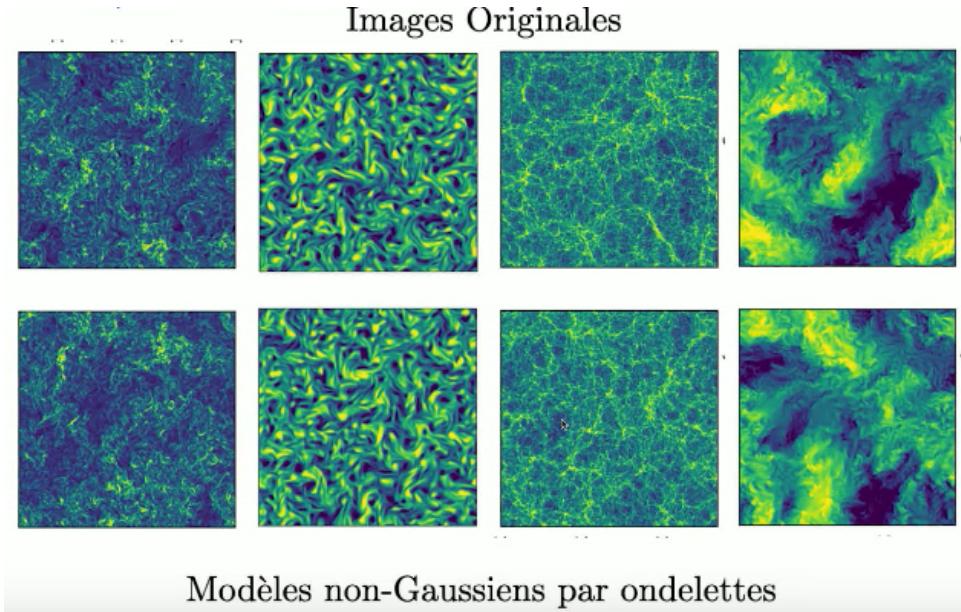


FIGURE 6 – Exemple de génération de champs non gaussiens par des modèles par ondelettes: *turbulence, cosmic web, etc.* (source arxiv:2306.17210)

Ensuite une fois que l'on a compris comment modéliser des interactions non-locales avec les corrélations entre échelles, on peut par exemple générer des images de champ non-gaussiens comme illustré sur la figure 6. On peut également mettre en œuvre des réseaux de scattering d'ordre 2 (Cours 2023 Sec. 9.9). Cependant, S. Mallat nous dit qu'il regarde ces résultats auxquels il a participé avec un certain sourire face aux résultats des modèles génératifs de type GPT-4. Mais, il nous dit qu'il y a besoin de comprendre vraiment ce qu'il se passe, et ce n'est pas la question de générer de belles images. Ce processus nécessite des mathématiques et de commencer avec des exemples simples (sans qu'ils soient pour autant simplistes).

Aller plus loin comme évoqué à la section précédente, consiste à effectuer une modélisation par les modèles de diffusion, en utilisant une équation différentielle stochastique (Langevin) que l'on va inverser³⁰. Dans un premier temps (Fig. 7), on part d'une image de visage sans bruit, puis progressivement à travers l'évolution (forward) de l'équation de Ornstein-Uhlenbeck³¹ le bruit est injecté jusqu'au point où l'on ne voit que du bruit. En-

30. NDJE: voir par exemple Yang Song et al. (2021) arXiv:2011.13456.

31. Leonard Salomon Ornstein (1880-1941) et George Eugene Uhlenbeck (1900-88) tous deux physiciens.

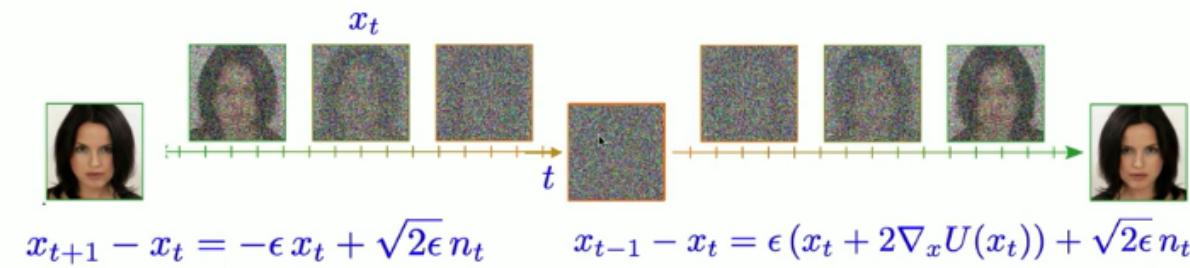


FIGURE 7 – Exemple d'évolution d'une image par le processus stochastique d'un modèle de diffusion: à gauche progression où l'on injecte du bruit (forward) et à droite où l'on opère un débruitage.

suite, on inverse le processus (backward) et on restaure l'image d'origine. Une fois connue l'expression de l'énergie $U(x)$, lors de la génération d'image (phase backward) à partir d'un bruit (blanc), le terme $\nabla_x U(x) = -\nabla_x \log p(x)$ est là naturellement pour maximiser la probabilité, et donner des exemples typiques de $p(x)$. Si les mathématiques semblent claires et établies depuis les années 1930 environ, le point que fait remarquer S. Mallat est que l'on pensait que l'estimation du score n'était possible. Mais la donne change avec les réseaux de neurones qui rendent possible le calcul du gradient de $U(x)$. L'optimisation du réseau se fait par "score matching".

Notez qu'à chaque nouvelle image de bruit x_0 correspond une nouvelle image issue de $p(x)$ au bout du processus de "débruitage", ce qui permet de générer de nouveaux portraits de "personnes qui n'existent pas", et de générer des scènes de paysages, d'intérieur de maison etc. Mais la question qui peut légitimement se poser est la suivante: imaginons que l'on ait une base de données de visages, puis que l'on entraîne un réseau de neurones selon la méthode du score matching, puis que l'on synthétise de nouveaux visages par diffusion backward, **en définitive n'obtient-on pas pour chaque nouveau visage une sorte de mélange de tous les visages de la bases de données?** Que se passe-t'il si la base de données d'entraînement est divisée en deux lots S1 et S2, que l'on fasse varier la taille de ces lots, et que l'on génère après l'entraînement de deux modèles de nouveaux visages à partir de la même image de bruit? **La question sous-jacente est celle de la généralisation.**

Notons au passage qu'ils sont tout comme Fokker d'origine néerlandaise.

La réponse à cette question est peut-être dans un article récent de Kadkhodaie et al. de 2023 auquel S. Mallat a contribué³², ce qui au passage est un exemple de recherche en train de se faire. La figure 8 tend à montrer qu'avec des bases d'entraînement de petites tailles ($N \leq 10$), les modèles ne font que "mémoriser" la base de données et ne sont pas capables de produire de nouveaux visages (overfitting). Si on initialise le processus de génération avec la même image de bruit, les images finales produites par les 2 modèles présentent des visages de leurs bases de données et sont naturellement différents par construction. Pour des bases de données comportant $O(100)$ visages différents, les modèles commencent à produire des "visages" nouveaux, mais ces derniers ne correspondent pas à la réalité d'un vrai visage bien formé. Ce n'est qu'à partir de bases de données dont la taille est de $O(10^4)$ que les modèles génèrent de nouveaux visages conformes à la réalité. Qui plus est, fait encore plus frappant, les modèles entraînés avec les lots S1 et S2 de cette taille qui rappelons le n'ont pas d'images en commun, donnent chacun un nouveau visage qui s'avèrent être des versions quasi-identiques si les deux modèles sont amorcés avec la même image de bruit. On a là vraiment une généralisation. **L'interprétation est que le transport du bruit blanc vers l'image finale appris par les deux modèles indépendants est le même, et qu'il s'agit de la même distribution $p(x)$ (ou de manière équivalente $U(x)$) qui est apprise, une fois que l'on dispose de suffisamment d'images d'entraînement.** On a une sorte de transition douce où l'on passe d'un état où les modèles sont de simples "perroquets", à un état où ils deviennent des "locuteurs" autonomes.

NDJE: je me permets un petit ajout. Il n'est pas rare, en apprentissage supervisé, de découper la base de données en plusieurs lots: par exemple un lot d'entraînement, un lot de test pour tuner les paramètres du modèle, et enfin un lot de validation pour en tirer in fine les valeurs de l'accuracy "top 5" dans le cas d'une tâche de classification par exemple. On peut de plus dans la même ligne avoir plusieurs modèles et des lots d'entraînements disjoints pour in fine comparer les performances. On peut d'ailleurs utiliser deux modèles d'architectures identiques mais dont les initialisations des paramètres sont différentes. Dans ce contexte, il est fréquent et de bon aloi de trouver que les performances de deux modèles sont statistiquement identiques, sinon on invoque un biais, une erreur systématique... Le résultat de Kadkhodaie et al. est de nature différente car il se place dans le cadre d'un apprentissage non-supervisé.

32. Kadkhodaie, Z., Guth, F., Simoncelli, E. P., Mallat, S. (2023). *Generalization in diffusion models arises from geometry-adaptive harmonic representation.* arXiv.2310.02557

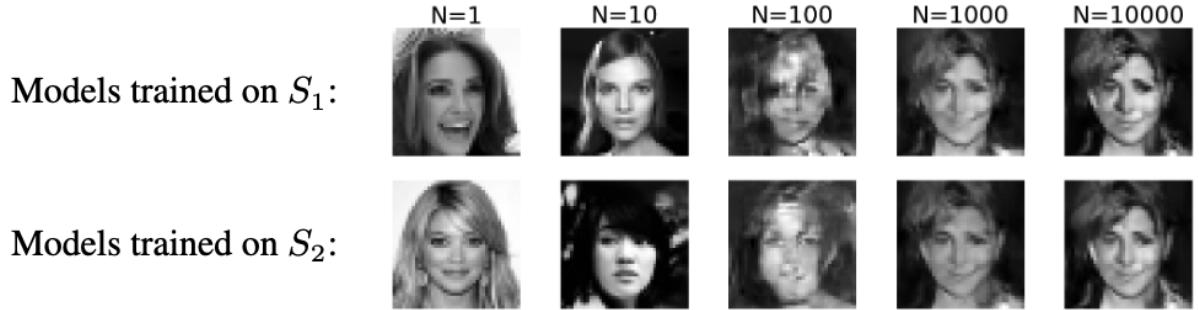


FIGURE 8 – Les modèles de diffusion sont entraînés sur des sous-ensembles distincts S_1 et S_2 d’un ensemble de données de visages. La taille des sous-ensembles N varie de 1 à 10^4 . Ensuite est généré un échantillon de chaque modèle avec une diffusion inverse déterministe initialisée avec la même image de bruit. Pour des ensembles d’entraînement de taille $N = 1$ et $N = 10$, les réseaux mémorisent, produisant des échantillons tirés de l’ensemble d’entraînement d’apprentissage. Pour $N = 100$, les échantillons générés ne correspondent à aucune image de l’ensemble d’apprentissage, mais sont fortement corrompus. Cela correspond à un régime où les réseaux passent de la mémorisation à la généralisation. Pour $N = 10^4$, les deux réseaux génèrent des images presque identiques.

La conséquence importante contrairement aux premières critiques exprimés par des "experts" nous dit S. Mallat, est que **les modèles de langages du type de GPT-4 ne sont pas de grosses mémoires** qui lors de nouvelles requêtes ne feraient qu'opérer un amalgame des différents éléments de la base de données. Il y a quelque chose de nettement plus profond.

2.8 Plan du cours

Pour terminer cette séance de présentation de la thématique de cette année, S. Mallat nous brosse rapidement le plan du cours:

- Nous commencerons pas les méthodes de Monte Carlo.
- On enchainera par le passage en revue de façon générale des notions de modélisation, d'apprentissage et d'échantillonnage.
- Puis nous approfondirons la modélisation par les champs de Markov,
- pour continuer sur l'échantillonnage d'abord en 1D, puis via les chaines de Markov.
- Pour aller plus loin, après le passage en temps continu, nous étudierons l'équation de Langevin.
- Puis nous aborderons la thématique du "score". Mais si le temps fait défaut cela ne sera que partie remise à l'an prochain.

Rappel: les séminaires sont importants pour donner un éclairage de ce qui se fait actuellement.

3. Séance du 24 Janv.

Durant cette séance, nous revenons sur les motivations qui nous amènent à utiliser des modèles aléatoires, puis nous abordons les méthodes de Monte Carlo pour effectuer des calculs en grande dimension. Finalement nous abordons le triptyque "Modélisation, Apprentissage, Inférence".

3.1 Modèles aléatoires: pourquoi?

Nous nous plaçons, comme illustrer à la dernière séance, en grande dimension. Mais rappelons qu'il n'est pas évident *a priori* qu'il faille se placer dans un cadre probabiliste. Voyons l'exemple du bruit blanc gaussien³³.

D'un point de vue probabiliste, le bruit blanc gaussien est une probabilité de d variables indécentes, dont chacune d'elles est distribuée selon une loi gaussienne, par exemple $x_i \sim \mathcal{N}(0, \sigma^2/d)$. Ainsi

$$p(x_1, x_2, \dots, x_d) = \prod_{i=1}^d p_i(x_i) = \prod_{i=1}^d \exp\left\{-\frac{x_i^2 d}{2\sigma^2}\right\} = \exp\left\{-\frac{\sum_{i=1}^d x_i^2 d}{2\sigma^2}\right\} = \exp\left\{-\frac{\|x\|^2 d}{2\sigma^2}\right\} \quad (9)$$

Or, $z = \sum_{i=1}^d x_i^2$ est une variable aléatoire. Si les x_i sont *iid* selon $\mathcal{N}(0, 1)$ on obtiendrait que $p(z)$ est la distribution du χ^2 à d degrés de liberté³⁴ dont l'espérance est $\mathbb{E}[z] = d$. Par changement de variables, on obtient alors

$$\|x\|^2 = \sum_{i=1}^d x_i^2 \xrightarrow{d \rightarrow \infty} \sigma^2 \quad (10)$$

et de même la variance du χ^2 à d degrés est $2d$, donc

$$Var(\|x\|^2) \xrightarrow{d \rightarrow \infty} \frac{2\sigma^4}{d} \quad (11)$$

Ainsi la localisation des échantillons d'un bruit blanc gaussien se fait sur une couche sphérique dont l'épaisseur tend vers 0 quand d augmente. On peut donc voir en grande dimension cette localisation comme une variété sphérique (Fig. 9). C'est assez général, **en grande dimension grâce à la propriété d'indépendance des variables, on assiste à des phénomènes de concentration sur des variétés.** On pourrait donc avoir un **regard purement déterministe**, et c'est en somme la perception usuelle que l'on se fait du monde. Notons nous avons montré durant le cours de 2023 comment en Physique statistique, à cause de la loi des grands nombres, émergent des phénomènes macroscopiques.

Cependant, cette vision déterministe nous amènerait à faire de la géométrie (sur

33. NDJE. voir aussi Cours 2023 Sec.2.6

34. NDJE. la loi $\chi^2(d)$ est $(1/2)^{d/2}/\Gamma(d/2)z^{d/2-1}e^{-z/2}$.

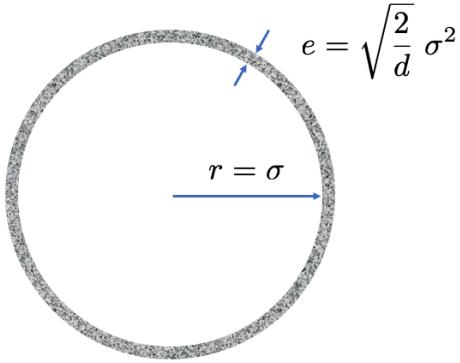


FIGURE 9 – Localisation de $\|x\|^2$ si toutes les composantes sont iid $x_i \sim \mathcal{N}(0, \sigma^2/d)$ en dimension d .

les variétés) en très grande dimension. Or, cela est très compliqué. Prenons le cas de la sphère de la figure 9. Imaginons un vecteur unitaire v émanant du centre de la sphère, et prenons un point x sur cette dernière. Examinons le produit scalaire $\langle v, x \rangle$:

$$z = \langle v, x \rangle = \sum_{i=1}^d x_i v_i \quad (12)$$

Les x_i étant des *v.a iid* gaussiennes, il vient que z est aussi une *v.a* gaussienne à savoir $\mathcal{N}(0, \sigma^2/d)$. Donc, quand d est grand, non seulement la moyenne est nulle, mais la variance du produit scalaire $\langle v, x \rangle$ tend également vers 0. Les x se trouvent donc comme s'ils étaient **concentrés sur l'équateur** perpendiculaire au vecteur v . Ainsi, ces phénomènes de concentrations ne sont pas forcément intuitifs.

Donc, ce n'est pas tant que le cadre déterministe ne serait pas envisageable, mais il serait très complexe à mettre en œuvre. Lors de la séance dernière, nous avions évoqué les raisons liées à la très grande dimension qui nous font adopter le cadre probabiliste. Par exemple, la nécessité de calculer des intégrales n'est rendu praticable qu'en utilisant la/les méthode(s) de Monte Carlo.

3.2 La méthode de Monte Carlo

Cette technique n'est pas très ancienne, on peut dater le début de son usage dans les années 40 lors du programme d'élaboration de la bombe nucléaire à Los Alamos. Le problème initial était décrit par un schéma déterministe: compter le nombre de diffusions d'un neutron avant qu'il ne vienne choquer un noyau d'atome dans un cœur de réacteur et donner une estimation de l'énergie perdue dans la collision. On peut en faire de même en théorie des jeux où les règles sont déterministes. C'est en quelque sorte un problème de dénombrement. Seulement, le problème pratique qui intéressait les physiciens atomistes était impossible à résoudre par ces techniques classiques. *NDJE. Les méthodes de simulation qu'on appelleraient maintenant bayésiennes en Physique des Particules & Cosmologie notamment, sont très naturelles. Prenons l'exemple des interactions du neutron ou de toute autre type de particules: on tire la probabilité d'occurrence de tel ou tel phénomène au fur et à mesure de la progression dans les différents matériaux que rencontre la particule. Ces techniques sont bien entendu à l'œuvre, par exemple pour étudier la progression des particules secondaires lors de collisions des protons des faisceaux dans le LHC du CERN dans les détecteurs Atlas, CMS, LHCb et Alice.*

Dans la suite des idées de Stanisław Marcin Ulam (1909-84) et John von Neumann (1903-57) à Los Alamos, il y a eu des articles notamment ceux de Nicholas Metropolis (1915-99) autour des chaînes de Markov qui ont permis les simulations aléatoires que sont les Monte Carlo Markov Chain (MCMC). On peut ainsi calculer des espérances par simulation. Rappelons au passage que l'expérience de Buffon³⁵ en 1733, calculait les décimales du nombre π également par un calcul d'espérance effectué par simulation.

En fait, calculer une espérance revient à calculer une intégrale:

$$\mathbb{E}_p[X] = \int x p(x) dx = \mu \quad (13)$$

Or, si l'on échantillonne $p(x)$ en produisant n valeurs $(x_i)_{i \leq n}$ alors on peut définir un estimateur de la moyenne:

$$\hat{\mu}_n = \frac{1}{n} \sum_{i=1}^n x_i \quad (14)$$

Il est non-biaisé à savoir $\mathbb{E}_p[\hat{\mu}_n]$ est égale à μ , si de plus les v.a $(x_i)_i$ sont *iid* de variance

35. Georges-Louis Leclerc, comte de Buffon (1707-88).

finie σ^2 alors la variance de $\hat{\mu}_n$ est donnée par

$$\sigma^2(\hat{\mu}_n) = \frac{\sigma^2}{n} \xrightarrow[n \rightarrow \infty]{} 0 \quad (15)$$

Enfin, la loi forte des grands nombres³⁶ nous indique que

$$\mathbb{P}\left[\hat{\mu}_n \xrightarrow[n \rightarrow \infty]{} \mathbb{E}_p[X]\right] = 1 \quad (16)$$

Comme les *v.a* $(x_i)_i$ sont *iid*, nous avons un peu plus via le théorème suivant:

Théorème 1 (*Central-limite*)

$$\sqrt{n}(\hat{\mu}_n - \mathbb{E}_p[X]) \xrightarrow[n \rightarrow \infty]{\text{cv. en loi}} \mathcal{N}(0, \sigma^2) \quad (17)$$

c'est-à-dire que l'on a

$$\lim_{n \rightarrow \infty} \mathbb{P}\left(\sqrt{n}(\hat{\mu}_n - \mathbb{E}_p[X]) \leq z\right) = \Phi(z/\sigma^2) \quad (18)$$

où $\Phi(z)$ est la fonction de répartition, ou fonction de distribution cumulative de la loi normale:

$$\Phi(z) = \frac{1}{\sqrt{2\pi}} \int_0^z e^{-x^2/2} dx = \frac{1}{2} \left(1 + \operatorname{erf}\left(\frac{z}{\sqrt{2}}\right)\right) \quad (19)$$

avec erf la fonction d'erreur de Gauss^a. On peut donc obtenir des intervalles de confiance asymptotiques. Notamment,

$$|\mu_n - \mathbb{E}_p[X]| \simeq \frac{\sigma}{\sqrt{n}} \quad (20)$$

a. Milton Abramowitz et Irene Stegun, Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables, section "Error Function and Fresnel Integrals".

Pourquoi cela nous intéresse-t'il? En fait, on peut toujours voir une intégrale de la façon suivante:

$$\int_{\mathcal{D}} f(x) dx = \int_{\mathbb{R}^d} \mathbf{1}_{\mathcal{D}}(x) f(x) dx \quad (21)$$

où $\mathbf{1}_{\mathcal{D}}$ est la fonction indicatrice de l'ensemble \mathcal{D} . Or, on peut toujours définir une densité de probabilité **uniforme** sur \mathcal{D} selon³⁷

$$p(x) = \frac{\mathbf{1}_{\mathcal{D}}(x)}{|\mathcal{D}|} \quad (22)$$

Ainsi,

$$\int_{\mathcal{D}} f(x) dx = |\mathcal{D}| \int_{\mathbb{R}^d} f(x) p(x) dx = |\mathcal{D}| \times \mathbb{E}_p[f[X]] \quad (23)$$

qui nous donne accès à **la valeur de l'intégrale via le calcul d'une espérance**. Soit U une *v.a* de **loi uniforme** sur \mathcal{D} , alors nous avons en tirant n échantillons $(u_i)_{i \leq n}$ de cette loi uniforme la convergence suivante:

$$(u_i)_{i \leq n} \text{ iid } U, \quad \frac{1}{n} \sum_{i=1}^n f(u_i) \xrightarrow{n \rightarrow \infty} \mathbb{E}_p[f[X]] \quad (24)$$

La convergence est assurée si la *v.a* $X = f(U)$ est de *variance bornée*. De plus la vitesse de convergence est directement reliée à la variance de X . Si par exemple la norme infinie est bornée

$$\|f\|_{\infty} = \sup_{x \in \mathcal{D}} |f(x)| < \infty \quad (25)$$

alors la variance de $f[X]$ est bornée³⁸. Alors, l'erreur d'approximation est bornée selon:

$$\left| \frac{1}{n} \sum_{i=1}^n f(u_i) - \mathbb{E}_p[f[X]] \right| \leq \frac{\|f\|_{\infty}}{\sqrt{n}} \quad (26)$$

Donc, en principe si l'on dispose de suffisamment d'exemples/échantillons, alors la moyenne donne un bon estimateur de l'espérance.

3.2.1 Calcul de volumes

Avec la technique de la moyenne, nous pouvons calculer par exemple des volumes:

$$|\Omega| = \int_{\mathcal{D}} \mathbf{1}_{\Omega}(x) dx \quad (27)$$

37. nb. Il est d'usage de noter le volume d'un ensemble \mathcal{A} selon: $|\mathcal{A}|$.

38. NDJE. On peut tenter une petite démonstration. D'une part, pour une *v.a* Y , $Var[Y] \leq \mathbb{E}[Y^2]$. En supposant que $f(\mathbb{E}(X)) = 0$ éventuellement en redéfinissant f , alors en prenant $Y = f(X)$, nous en déduisons que $Var[f(X)] \leq \mathbb{E}[f(X)]$ qui est manifestement plus petit ou égal à $\|f\|_{\infty}$.

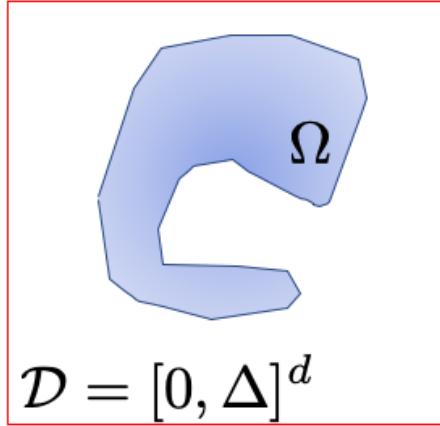


FIGURE 10 – Calcul du volume de $\Omega \subset \mathcal{D} = [0, \Delta]^d$.

où $\mathcal{D} = [0, \Delta]^d$. Ainsi en tirant des $(u_i)_i$ uniformément dans la boîte \mathcal{D} , on obtient

$$|\Omega| \approx \Delta^d \times \frac{1}{n} \sum_{i=1}^n \mathbf{1}_\Omega(u_i) \quad (28)$$

Il s'agit de la méthode hit & miss (ou acceptation-réjection). Nb. Il est sous-entendu qu'il faut être capable de pouvoir dire si u_i fait partie ou non du domaine Ω . Notons que l'on peut calculer $\pi/4$ en tirant uniformément des points dans le carré $[0, 1]^2$ et en comptant combien tombent en moyenne dans le quart de cercle de rayon 1.

Ce genre de calcul est omniprésent en ML par exemple pour le calcul des erreurs. On peut également considérer comment AlphaGo ou bien un simulateur d'échec calcule la "valeur" d'une position en considérant à partir de la position présente, la proportion de parties simulées qui sont *in fine* gagnantes. On note déjà avec ce genre d'exemples que l'on ne peut simuler toutes les parties possibles, et donc il faut être astucieux. Nous verrons comment de telles considérations sont à l'œuvre dans les méthodes MCMC. Car en effet, faire une exploration uniforme n'est pas forcément efficace³⁹

39. NDJE. Dans le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/cours2023/Monte_Carlo_Sampling.ipynb vous verrez comment l'*importance sampling* aide par exemple à résoudre des problèmes de calcul d'intégrale où il y a de grandes zones "mortes" si l'on considère un domaine \mathcal{D} englobant Ω mal adapté.

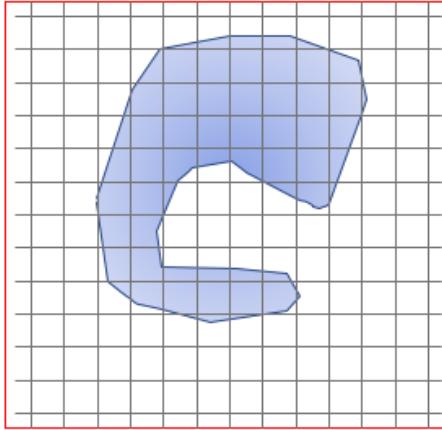


FIGURE 11 – Calcul de intégrale de $f(x)$ où $x \in \Omega \subset \mathcal{D} = [0, \Delta]^d$ en utilisant une grille uniforme.

3.2.2 Pourquoi ne pas faire un calcul déterministe?

On peut imaginer de faire un calcul d'intégral en utilisant une grille uniforme comme sur la figure 11, en utilisant la méthode de Riemann. L'erreur commise est alors

$$\varepsilon = \left| \frac{1}{|\mathcal{D}|} \int_{\mathcal{D}} f(x) dx - \frac{1}{n} \sum_{i=1}^n f(x_i) \right| \quad (29)$$

où les x_i sont les points de la grille. On peut découper le volume \mathcal{D} en petits volumes n'ayant chacun qu'un point x_i , ce qui permet d'écrire:

$$\int_{\mathcal{D}} f(x) dx = \sum_{i=1}^n \int_{\mathcal{V}_i} f(x) dx \quad (30)$$

d'où

$$\varepsilon = \left| \sum_{i=1}^n \left(\frac{1}{|\mathcal{D}|} \int_{\mathcal{V}_i} f(x) dx - \frac{1}{n} f(x_i) \right) \right| \quad (31)$$

En notant que $n|\mathcal{V}_i| = |\mathcal{D}|$ alors

$$\varepsilon = \frac{1}{|\mathcal{D}|} \left| \sum_{i=1}^n \int_{\mathcal{V}_i} (f(x) - f(x_i)) dx \right| \quad (32)$$

Le facteur $|\mathcal{D}|$ est l'équivalent du Δ^d vu précédemment et n'est pas essentiel pour la suite. Ce qui nous intéresse est de savoir si le comportement asymptotique du calcul déterministe est plus ou moins avantageux par rapport à une technique Monte Carlo.

Mettons que $f \in C^1$ (dérivée bornée) alors

$$\|f(x) - f(x_i)\| \leq \left(\sup_{x \in \mathcal{V}_i} |f'(x)| \right) \times |x - x_i| \quad (33)$$

Ainsi

$$\begin{aligned} \varepsilon &\leq \frac{1}{|\mathcal{D}|} \sum_i \int_{\mathcal{V}_i} |f(x) - f(x_i)| \, dx \\ &\leq \frac{1}{|\mathcal{D}|} \sum_{i=1}^d \left(\sup_{x \in \mathcal{V}_i} |f'(x)| \right) \times \left(\sup_{x \in \mathcal{V}_i} |x - x_i| \right) \times |\mathcal{V}_i| \end{aligned} \quad (34)$$

Or, $|\mathcal{V}_i| = |\mathcal{D}|/n$, donc

$$\varepsilon \leq \sup_{x \in \mathcal{D}} |f'(x)| \times \frac{1}{n} \sum_{i=1}^n \sup_{x \in \mathcal{V}_i} |x - x_i| \quad (35)$$

Cependant, si mettons que $|\mathcal{D}| = 1$, la distance max des points à l'intérieur de chaque petits volumes est de l'ordre de $n^{-1/d}$. Finalement, on obtient le résultat

$$\boxed{\varepsilon \leq \sup_{x \in \mathcal{D}} |f'(x)| \times C n^{-1/d}} \quad (36)$$

Donc, dans le cas du calcul **Monte Carlo on a un scaling en $n^{-1/2}$ indépendant de la dimension**, alors que **le calcul déterministe donne un scaling $n^{-1/d}$** qui pour d en grande dimension devient exponentiellement limitant. Notons pourtant que ce résultat est obtenu en mettant une contrainte plus forte sur la régularité de f . Donc, pour réaliser une erreur donnée, il faut un nombre d'échantillons colossal ($n \propto \varepsilon^{-d}$). **Le point de vue déterministe est confronté à la malédiction de la dimensionalité alors qu'en utilisant la méthode Monte Carlo, on s'en affranchi.**

Maintenant, pourquoi a-t-on cette différence de scaling entre les deux méthodes? **La raison profonde est qu'en grande dimension la distribution des points de la grille**

déterministe n'est pas du tout uniforme. Manifestement elle n'est pas du tout adaptée pour estimer une fonction dont on sait *a priori* uniquement quelle est régulière. Dans ce cas la distribution uniforme est bien mieux adaptée. Mais la question demeure cependant, peut-on de manière déterministe concevoir des distributions de points uniformes?

Déjà en prenant une grille "en quiconce" le scaling est en $n^{-2/d}$, mais peut-on obtenir un optimum?

3.2.3 Méthode de quasi-Monte Carlo

La question est de choisir d'une manière optimale les points x_i telle que l'erreur⁴⁰ ε (Eq. 29) soit la plus petite possible. Si les méthodes Monte-Carlo utilisent une suite de points aléatoires, les méthodes de quasi-Monte Carlo vont choisir une suite adaptée.

Théorème 2 (Quasi-Monte Carlo)

Soit la variation totale de la fonction, définie comme

$$\|f\|_{TV} = \int_{\mathbb{R}^d} \|\nabla f(x)\| dx \quad (37)$$

que l'on suppose finie. Sous cette hypothèse, on peut définir des $\{x_i\}_{i \leq n}$ tels que

$$\varepsilon \leq \|f\|_{TV} \times C_{n,d} n^{-1} \quad (38)$$

avec

$$C'_d (\log n)^{d-1} \leq C_{n,d} \leq C_d (\log n)^{d-1/2} \quad (39)$$

Donc, au mieux on a une borne en $n^{-1}(\log n)^{d-1/2}$ qui est **meilleure que celle obtenue avec la technique de Monte Carlo mais pour cela $n \geq e^d$ à cause de la dépendance des constantes "C" vis-à-vis de la dimension**, ce qui rend ces méthodes beaucoup moins efficaces que la méthode de Monte Carlo.

Certes la méthode quasi-Monte Carlo a l'air d'être efficace grâce à son scaling en n , mais les constantes sont énormes et l'on suppose de plus une hypothèse de régularité sur f assez forte, car elle sous-tend une forme de parcimonie. En pratique, sont utilisées

40. NDJE. En passant dans le cours $|\mathcal{D}|$ est pris implicitement égal à 1 dans les formules.

dans ce cadre des méthodes "mixtes" où l'on part d'une grille déterministe que l'on raffine aléatoirement afin d'améliorer le scaling des constantes. Mais il n'en reste pas moins vrai que le plus difficile est l'uniformité de la distribution des points de la grille. C'est pourquoi les calculs via Monte Carlo sont bien plus faciles à mettre en œuvre. Cependant, quelles sont leurs limites?

3.2.4 Limites de la méthode de Monte Carlo

En grande dimension comme évoqué auparavant, nous avons des **phénomènes de concentration**⁴¹. Donc, on va vouloir calculer des intégrales pour lesquelles la mesure du support Ω des fonctions est presque nulle. Si l'intégrale est non nulle (mettons que f est positive), alors cela implique que $f(x)$ a de grandes amplitudes de telle sorte que l'intégrale de $f^2(x)$ est très grande (variance), tout comme la $\|f\|_\infty$. Donc, la borne de l'erreur par Monte Carlo ε (Eq. 26) est très grande, ce qui nécessite beaucoup d'échantillons.

Une autre façon de voir le problème engendré par la concentration des données, est de considérer la méthode par réjection. Si on englobe par une "boîte" \mathcal{D} le domaine où se concentrent les données, l'efficacité de la méthode est gouvernée par le rapport $|\Omega|/|\mathcal{D}|$ qui est très petit. On a donc un problème pratique de générer beaucoup de points pour obtenir une erreur donnée.

Pour résoudre ce problème, **il faut que \mathcal{D} s'adapte⁴² au mieux à Ω** (Fig. 12). Pour cela, il est nécessaire de **trouver des distributions de probabilité qui s'adaptent à la géométrie**, et il faut pouvoir les échantillonner. En quelque sorte, on pourra alors faire le calcul suivant

$$\int_{\mathbb{R}^d} f(x) dx = \int_{\mathcal{D}} \underbrace{\frac{f(x)}{p(x)}}_{w(x)} p(x) dx \approx \frac{1}{n} \sum_{i=1}^n w(x_i) \quad x_i \sim p(x) \quad (40)$$

Donc, **le problème est de trouver des distributions de probabilité $p(x)$ qui peuvent approximer un domaine Ω arbitraire, et dont on peut tirer des échantillons.**

41. NDJE. Voir Cours 2022, 2023, la thématique des ensembles typiques de Shannon.

42. NDJE. D'où la méthode d'Importance Sampling dont un exemple est donné dans le notebook de la note en bas de page 39.

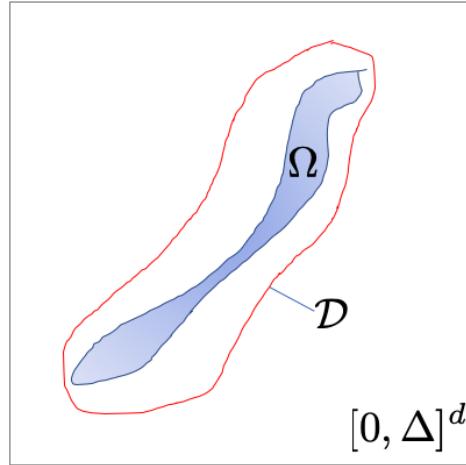


FIGURE 12 – Schématisation de l'ajustement à la géométrie de l'espace des données Ω , de l'espace englobant dans lequel on opère un tirage de points: passage de $[0, \Delta]^d$ à \mathbb{D} .

3.3 Modélisation de probabilités, apprentissage/inférence

On veut approximer une probabilité $p(x_1, \dots, x_d)$ tout comme $p(y|x_1, \dots, x_d)$ la probabilité conditionnelle de y connaissant des données x , en grande dimension. La section précédente nous invite à conclure qu'*a priori* c'est un problème difficile.

3.3.1 Petit exemple: détection de spams

Considérons le problème de détection de spams: recevant un mail constitué de mots de la langue française, on veut pouvoir dire si c'est un spam ($y = 1$) ou non $y = 0$. On code $x_i = 1$ si le mot i (du dictionnaire) est présent dans le mail (et $x_i = 0$ sinon). On a donc un problème à d variables binaires dont la combinatoire nous dit qu'il y a 2^d possibilités. Il n'est pas tenable de calculer toutes les probabilités correspondantes dès lors que d représente la taille du dictionnaire courant de la langue française. Il faut donc faire des approximations.

Dans le cas du problème de spam, on utilise l'**hypothèse de "naïveté bayésienne"**.

Selon la formule de Bayes, nous avons

$$p(y|x_1, \dots, x_d) = \frac{p(x_1, \dots, x_d|y)p(y)}{p(x_1, \dots, x_d)} \quad (41)$$

et l'hypothèse consiste à écrire

$$p(x_1, \dots, x_d|y) = \prod_{i=1}^d p(x_i|y) \quad (42)$$

c'est-à-dire que les mots sont indépendants les uns des autres sous l'hypothèse de faire partie d'un spam ou non. Pour obtenir le score du mail, on maximise la probabilité *a posteriori*, donc on veut obtenir

$$y^* = \operatorname{argmax}_y p(y) \prod_{i=1}^d p(x_i|y) \quad (43)$$

Dans ce cadre d'approximation pour estimer $p(x_i|y)$, il y a 4 états à considérer pour le couple (x_i, y) mais véritablement que 2 degrés de liberté (nb. $p(x|y=1) + p(x|y=0) = 1$). Finalement, **on a d probabilités à estimer ce qui fait nettement une différence.**

En pratique, cette méthode basée sur l'indépendance est assez efficace pour le problème de détection de spams. Par contre, lorsque l'on veut faire de la **classification d'images**, l'hypothèse de ne considérer que $p(x_i|y)$ revient à ce poser la question: considérant la valeur du pixel i (ie. x_i) a-t-on à faire avec une image de chien vs chat, voiture vs avion, etc. Il est clair que **l'hypothèse ne tient pas**. Il faut par contre pouvoir **analyser les corrélations** des pixels pour voir apparaître les structures.

3.3.2 Apprentissage: la meilleure approximation

Pour aborder les problèmes comme ceux de la génération/classification d'images, comme évoqué lors de la séance introductory, il nous faut utiliser des **familles paramétrées** $\{p_\theta(x)\}$ beaucoup plus sophistiquées, comme les **champs de Markov**. Nous verrons cela dans la suite.

Une fois définie une famille paramétrée de probabilités, il nous faut trouver le meilleur θ pour que $p_{\theta^*}(x)$ approxime au mieux $p(x)$. Il y a beaucoup d'approches possibles

qui consistent à minimiser une fonction d'objective, comme la divergence de Kullback-Leibler⁴³ telle que

$$D_{KL}(p||p_\theta) = \int p(x) \log \frac{p(x)}{p_\theta(x)} dx \geq 0 \quad (44)$$

et $D_{KL} = 0$ ssi $p = p_\theta$. On peut réécrire l'expression de $D_{KL}(p||p_\theta)$ comme suit

$$D_{KL}(p||p_\theta) = \underbrace{\int p(x) \log p(x) dx}_{-\mathbb{H}[p]} - \underbrace{\int p(x) \log p_\theta(x) dx}_{\mathbb{E}_p[\log p_\theta(x)]} \geq 0 \quad (45)$$

où $\mathbb{E}_p[\log p_\theta(x)]$ est la vraisemblance et $\mathbb{H}[p]$ l'entropie différentielles de p au sens de Cl. Shannon⁴⁴ qui est une constante vis-à-vis du problème d'optimiser la valeur de θ . L'idée revient à R. Fisher, car **maximiser la vraisemblance, c'est-à-dire l'aptitude de la distribution $p_\theta(x)$ à rendre compte de la distribution des données, minimise la divergence de Kullback-Leibler**. Si l'on dispose de données $(x_i)_{i \leq n}$ sensées être représentatives de la probabilité sous-jacente $p(x)$ alors

$$\mathbb{E}_p[\log p_\theta(x)] \approx \frac{1}{n} \sum_{i=1}^n \log p_\theta(x_i) \quad (46)$$

L'objectif est donc de maximiser cette quantité (où minimiser si l'on tient compte du signe "-").

L'hypothèse bayésienne ajoute une hypothèse sur les valeurs possibles de θ (*prior*) qui est une forme de régularisation en termes d'estimation. En fait, on aimerait maximiser la probabilité *a posteriori* $p(\theta|x)$ (le meilleur θ étant acquis les données x). En rappelant la formule de Bayes (Eq. 4):

$$p(\theta|x) = \frac{p(x|\theta)p(\theta)}{p(x)} \quad (47)$$

l'optimisation de $p(\theta|x)$ revient à optimiser le produit du *likelihood* ($p(x|\theta)$) et du *prior* ($p(\theta)$). Or, le likelihood est par construction ce que l'on a noté $p_\theta(x)$. Maintenant maximiser $p(\theta|x)$ revient à maximiser le log, et comme on a une base d'échantillons, on peut

43. NDJE. voir le Cours de 2019 Sec. 7.2.3 par exemple

44. NDJE. ex. voir Cours 2022 Sec. 5, et Cours 2023 Sec. 5.1

maximiser en moyenne. Il vient donc

$$\theta^* = \operatorname{argmax}_{\theta} \mathbb{E}_p[\log p(\theta|x)] = \operatorname{argmax}_{\theta} (\mathbb{E}_p[\log p_{\theta}(x)] + \log p(\theta)) \quad (48)$$

On ajoute donc une information *a priori* que l'on a sur les valeurs de θ . Cette information agit dans l'optimisation bayésienne comme le fait une régularisation⁴⁵ (en norme L2 *ridge regression*⁴⁶, ou en norme L1 pour augmenter la sparsité sur les paramètres) dans un algorithme déterministe de régression linéaire (ou SVM) ou de type optimisation des paramètres d'un réseau de neurones, pour **éviter l'overfitting**.

Une fois la fonction d'objective à optimiser définie, la méthode d'optimisation est usuellement une **descente de gradient** (nb. minimisation tenant compte du signe "-") que l'on a déjà abordé dans les cours des années précédentes⁴⁷.

3.3.3 L'Inférence

Considérons le problème de détection de spams, l'inférence consiste par exemple à calculer la probabilité qu'un mail contenant les mots k et j soit un spam ou non. Ainsi

$$p(y|x_k, x_j) = \frac{p(y, x_k, x_j)}{p(x_k, x_j)} = \frac{\int p(y, x_1, \dots, x_d) \prod_{i \neq k,j} dx_i}{\int p(x_1, \dots, x_d) \prod_{i \neq k,j} dx_i} \quad (49)$$

Les deux intégrales sont en grande dimension et on utilise donc des techniques de Monte Carlo. Comme nous le dit S. Mallat, dans ce domaine des mathématiques, "on passe sa vie à faire des calculs d'intégrales".

3.3.4 Quelle type de modélisation choisir? Exemples

C'est la partie la plus ouverte, toujours du domaine de la recherche, et la plus difficile: quelles sont les familles de probabilités qui sont les plus adaptées pour approximer des distributions de données. C'est dans cette problématique que **les réseaux de neurones ont fait la différence**.

45. NDJE. Voir Cours 2018 Sec. 7.3.2 et/ou Cours 2019 Sec. 7.2.4.2

46. Ex. si on cherche à minimiser $\theta^T \phi(x) + \lambda \|\theta\|^2$ cela revient à définir un prior de type $p(\theta) \propto e^{-\lambda \|\theta\|^2}$.

47. NDJE. voir par exemple Cours 2018 Sec. 10.1

Dans les années 2000, nous dit S. Mallat, à gros trait pour les applications, la modélisation se faisait par des **modèles gaussiens**. Dans ce type de modélisation on utilise l'expression⁴⁸ suivante, en supposant que $\mathbb{E}[x] = 0$ pour simplifier la notation:

$$p(x) = Z^{-1} \exp\left(-\frac{1}{2}x^T K x\right) \quad (50)$$

où K est un noyau (matrice $d \times d$ symétrique définie positive), et l'on a donc en terme d'énergie une fonction quadratique en x . L'on peut montrer⁴⁹ que K est relié à la matrice de covariance $C = \mathbb{E}_p(xx^T)$, c'est-à-dire⁵⁰ $C = K^{-1}$ (moments d'ordre 2 centrés), et ses éléments sont les paramètres θ à optimiser.

Un exemple particulièrement important utilisé par exemple en traitement du signal est celui où l'on fait l'**hypothèse de stationnarité** (invariance par translation). Si l'on index par i la coordonnée d'un pixel d'une image ou d'un échantillon d'une trame temporelle, selon cette hypothèse:

$$\forall k \geq 0, \forall \tau, \quad p(x_{i_1-\tau}, x_{i_2-\tau}, \dots, x_{i_k-\tau}) = p(x_{i_1}, x_{i_2}, \dots, x_{i_k}) \quad (51)$$

Cette hypothèse est très souvent vérifiée. S. Mallat par exemple l'illustre en prenant le cas d'une image sans point de référence; on peut également citer en Cosmologie que les propriétés du fond diffus cosmologique (CMB) sont avec une très bonne approximation invariantes selon la direction pointée par les instruments (translation dans l'espace ascension droite-déclinaison). Par contre, les images de visages centrées de type photographie de carte d'identité, ne sont pas du tout stationnaires.

Dans ce cadre stationnaire, nous avons la propriété suivante:

$$\mathbb{E}_p(x_i x_j^T) = \mathbb{E}_p(x_{i-j} x_0^T) = f(i - j) \quad (52)$$

c'est-à-dire que **la covariance ne dépend que du paramètre de translation**, c'est donc une matrice de Toeplitz (éléments des diagonales identiques).

Maintenant, comme $K = C^{-1}$ est symétrique définie positive, on peut regarder

48. NDJE. Voir par exemple Cours 2022 Sec. 2.6 et/ou Cours 2023 Sec.8.1

49. nb. ici $x = (x_1, \dots, x_d)^T$ donc x est de dimension $d \times 1$.

50. NDJE. *hint:* K est diagonalisable avec P une matrice orthogonale, telle que $K = P^T D P$, et l'on effectue un changement de variable $y = Px$.

les distributions dans la base qui la diagonalise. Il s'agit de la la base des **composantes principales** (PCA). **L'hypothèse stationnaire, nous indique que la base PCA est celle de Fourier**⁵¹. En effet, en ignorant les effets de bord, si e est un vecteur propre de C alors par définition:

$$Cx = \lambda x \Rightarrow \forall j, \sum_i C(i-j)e_i = \lambda e_j = \sum_k C(k)e_{k+j} \quad (53)$$

d'où la composante k d'un vecteur de la base est une sinusoïde $e_k(\omega) = e^{ik\omega}$ et la valeur propre positive λ peut se mettre sous la forme d'une variance σ_ω^2 , et n'est autre que $\hat{C}(\omega)$ soit le spectre de puissance (ou puissance spectrale)⁵².

Tout ceci est le modèle de base qui dit essentiellement que **les distributions de probabilité se trouvent concentrées - les ensembles typiques sont des ellipsoïdes - le long des axes principaux de la matrice de covariance.**

Cependant, **ce modèle ne suffit pas**, et pour l'illustrer prenons l'exemple de la turbulence⁵³. Sur la figure 13 nous avons un exemple à gauche d'une image d'un fluide turbulent⁵⁴, au centre son spectre de puissance, et à droite une réalisation d'un champ gaussien généré à partir de ce spectre de puissance. Pour se faire, il nous suffit de mesurer la fonction de corrélation à 2 points (transformée de Fourier du spectre de puissance), c'est-à-dire estimer la matrice de covariance. Si l'on voit de la granularité, **on a perdu toutes les structures** de l'image d'origine. Et pourtant notons que les tourbillons peuvent se produire n'importe où dans l'image, donc **on a bien à faire avec un champ stationnaire**. Donc, il manque quelque chose dans la modélisation. Mais quoi?

En fait, **le modèle gaussien correspond à un modèle d'entropie maximum**⁵⁵, c'est-à-dire une forme de désordre maximal moyennant les informations qu'il a sur les valeurs des coefficients de la matrice de covariance. Or, la turbulence est structurée et donc **les ensembles typiques ne sont pas des ellipsoïdes**.

En allant plus loin dans la modélisation, lors de l'introduction S. Mallat nous a

51. NDJE. Voir Cours 2021 Sec. 4.4

52. NDJE. En Cosmologie, le spectre de puissance est noté plutôt $P(k)$ où k est la norme du vecteur d'onde.

53. NDJE. c'est un exemple du Cours de 2022 Sec. 4.5, que j'ai mis à disposition dans la notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2023/gaussian_vs_turbulent_fow.ipynb.

54. image issue de l'article <https://phys.org/news/2015-10-key-features-transition-liquid-smooth.html>.

55. NDJE. Théorème de Gibbs vu dans le Cours 2023 Sec. 7.6

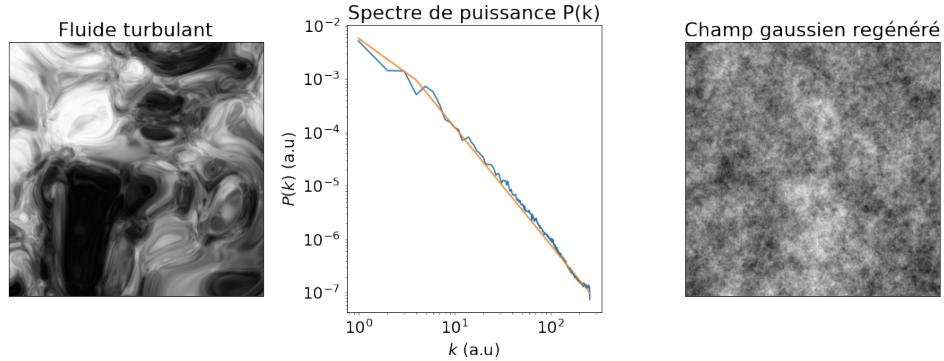


FIGURE 13 – A gauche une image d'un fluide turbulent dans une tuyau (Crédit: Piotr Siedlecki/public domain); au centre: le spectre de puissance issu de l'image ($\propto k^{-2.2}$) où k est ici la norme du vecteur d'onde qui est la notation consacrée en Cosmologie; à droite: un champ gaussien généré à partir de ce spectre de puissance.

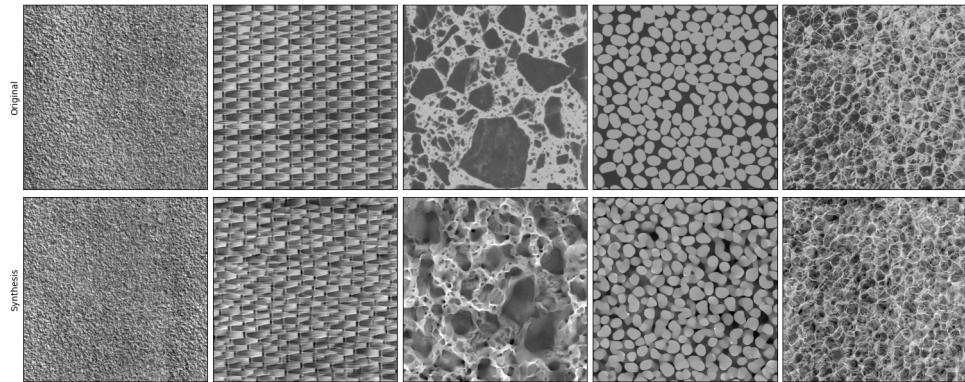


FIGURE 14 – Exemple de textures que l'on peut synthétiser en allant au delà de la modélisation par champ gaussien (réseaux d'ondelettes).

montré que l'on pouvait **synthétiser des champs stationnaires**, tels que des champs physiques (Fig. 6), ou bien des textures comme celles de la figure 14 en utilisant des réseaux d'ondelettes⁵⁶.

Le domaine de la recherche actuelle et de comprendre comment on peut **synthétiser des champs non stationnaires (non ergodiques)**. S. Mallat nous invite à regarder les séminaires de Marylou Gabrié qui parle des modèles dans le cadre de la Physique Statistique et les champs de Markov, et de Francis Bach qui lui discute des modélisations de visages et autres types d'images avec les modèles par Score Diffusion. **Les deux grosses différences résident dans la technique d'estimation, mais surtout dans la modélisation Markov vs Réseaux de neurones.**

S. Mallat, nous donne à voir une application des dernières modélisations, à savoir par exemple l'**inpainting** (Fig. 15) qui se présente ici comme la génération d'une image y selon la probabilité $p(y|x)$, où x est une image dégradée et p un modèle entraîné avec une très vaste base de données d'images. L'image y est une reconstitution de l'image x (nb. ce n'est pas forcément l'image d'origine mais les exemples choisis illustrent les possibilités). Le modèle a capturé toutes les formes de régularité. On peut également procéder au **débruitage** d'une image comme sur la figure 16. Il s'agit d'une cellule, et typiquement en imagerie X, on ne veut pas exposer les tissus à de fortes doses, et donc le rapport signal/bruit n'est pas optimal.

Malgré ces jolis résultats, il y a un gros hic voir "danger" nous dit S. Mallat. Les usages sont très précautionneux notamment dans l'imagerie médicale, à cause des **phénomènes d'hallucinations**. En effet, avec des modèles de *diffusion*, à la manière d'un débruitage, à partir d'un bruit on peut petit-à-petit faire apparaître une image qui se trouve géométriquement sur un espace typique (mais qui n'est pas un ellipsoïde) comme sur la figure 17. La question qui se pose est que dire de l'image générée? Dans le cas médical, on pourrait faire apparaître ou non des tumeurs par exemple. Le problème qui se pose est bien d'éviter ces phénomènes d'hallucinations.

S. Mallat nous projette ensuite (Fig. 18) un exemple de **séparation de composantes** dans l'audio qu'il nous avait présenté durant le cours de 2019 et qui datait de 2018⁵⁷. Il

56. NDJE. vous pouvez voir comment cela se fait via le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2023/TextureSynthesis.ipynb.

57. NDJE. voir Cours de 2019 Sec. 2.2.2, et pour le son je vous invite à visionner la vidéo du cours de 2024 vers 1:29:40.

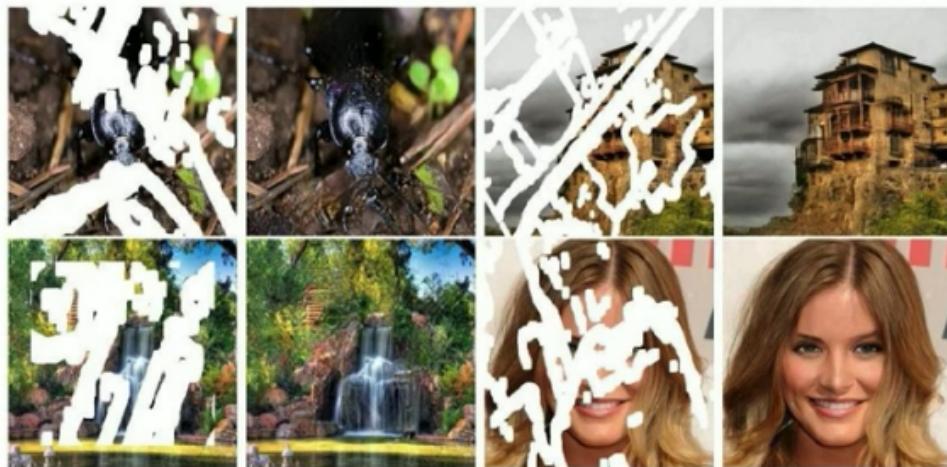


FIGURE 15 – Illustration de l'*inpainting*, où à partir d'un modèle entrainé avec une très vaste base de données d'images, il arrive à générer une image y à partir d'une image dont on a blanchi des parties.

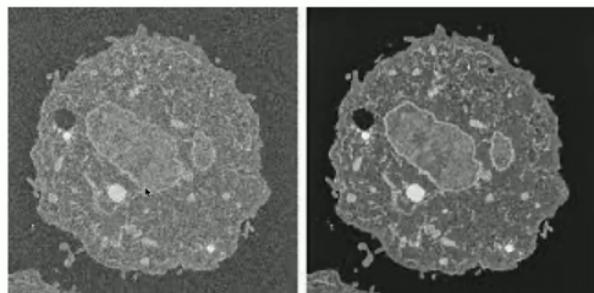


FIGURE 16 – Illustration du débruitage. On observe x qui est une superposition de y et du bruit, et l'on veut en tirer y , où plutôt un échantillon de $p(y|x)$.

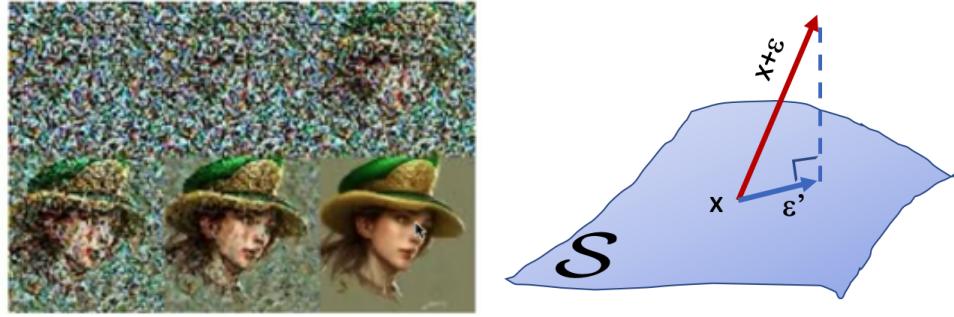


FIGURE 17 – Illustration de la génération par un modèle de diffusion qui ressemble à un débruitage mais pas tout à fait. Une fois entraîné par core matching, le modèle génère une image à partir d'un bruit. En quelque sorte il opère une projection d'une image totalement bruitée comme si elle était décomposable en une image "non-bruitée" x appartenant à un ensemble typique (S) et un bruit. Cependant que dire du x que va choisir le modèle?

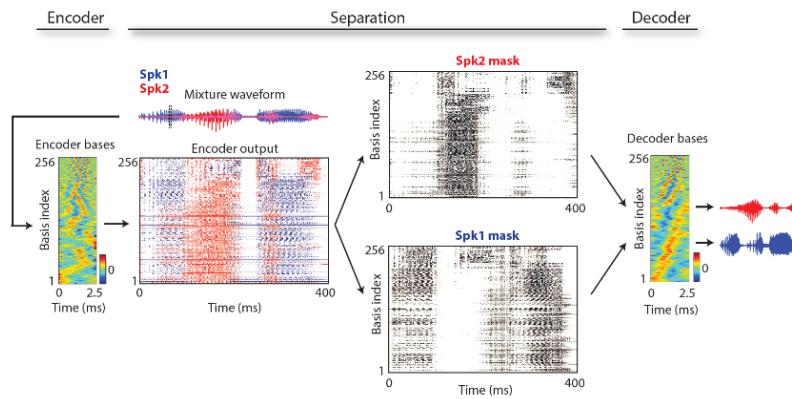


FIGURE 18 – Un résultat datant de 2018 (Voir Cours de 2019) qui montre comment on peut résoudre la séparation des discours de deux locuteurs ayant le même type de voix.

s'agit du "Cocktail Party challenge" pour lequel on essaye de séparer les discours prononcés en même temps par deux locuteurs ayant le même timbre de voix. L'apport des réseaux de neurones dans ce domaine a opéré une véritable révolution, car cela faisait grossièrement 30 ans, nous dit-il, que les gens essayaient de résoudre le problème en traitement du signal à partir des spectrogrammes et que cela ne donnait pas de résultats probants.

Pour finir et introduire le séminaire de Michele Sebag, S. Mallat nous indique que les modèles actuels de réseaux de neurones ne mettent pas en œuvre des graphes directionnels qui supporteraient une **modélisation causale**. En pratique, on ne sait pas pour le moment faire de l'apprentissage de telles structures. Donc, dans le cadre du cours, on va utiliser des modèles non-causaux.

4. Séance du 31 janv.

Durant cette séance nous allons aborder la *modélisation* et l'*approximation*⁵⁸ à travers la notion tout à fait fondamentale des **champs de Markov**. C'est l'outil qui permet de comprendre, avec des modèles de relative basse dimension, les interactions entre variables. Dans ce contexte, l'approximation et l'optimisation sont deux facettes de l'apprentissage. Rapelons, qu'il nous faut faire le choix d'un modèle paramétré qui approxime potentiellement la vraie distribution de probabilité, et il faut pouvoir en déterminer le meilleur jeu de paramètres.

Nous retrouvons le lien avec la Physique Statistique vu dans le cours de 2023. Le principe est de modéliser les **interactions entre les variables** tout en explicitant la notion d'**indépendance conditionnelle**.

4.1 Schéma causal et non-causal

Prenons un exemple simple de vote de quatre personnes (A,B,C,D) qui vont interagir et donc s'influencer mutuellement. Pour modéliser l'influence entre 2 personnes (X,Y) au moment du vote, on peut se dire que si les votes de X et Y sont identiques alors $\Phi(X, Y)$

58. NDJE. Finalement, S. Mallat n'aura pas eu le temps d'aborder cette thématique qui fera l'objet de la séance suivante.

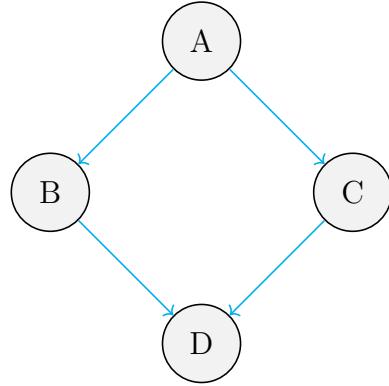


FIGURE 19 – Modèle directionnel.

qui représente l'interaction entre X et Y, est grande, et au contraire si les votes sont différents alors $\Phi(X, Y)$ est petite. Au final, ce que l'on cherche c'est la probabilité jointe $p(A, B, C, D)$, modélisée par exemple selon:

$$p(A, B, C, D) = \frac{1}{Z} \Phi(A, B) \Phi(B, C) \Phi(C, D)$$

avec Z une constante de normalisation. Avant le vote, les influences vont jouer leurs jeux, et on peut se dire qu'il y a une forme d'équilibre qui va advenir (supposant qu'il arrive avant le vote). Le vote le plus probable reflète les interactions entre les différents membres. Ceci dit, pour être capable de calculer $p(A, B, C, D)$, il faut pouvoir déterminer la constante Z . De plus, si l'on veut comprendre le vote de A, il faut comprendre l'échange de d'information entre les différents membres (phénomène de propagation). C'est plus compliqué qu'un modèle directionnel.

Un *modèle directionnel* (Fig. 19) est à l'œuvre par ex. si on sait que A influence B et C, lesquels influencent D. Dans ce cas, nous avons (*chain rule*)

$$p(A, B, C, D) = p(A|B, C, D)p(B, C|D)p(D)$$

Alors l'échantillonnage d'une configuration (A,B,C,D) se fait en trouvant une configuration de D seul, puis une configuration du couple (B,C) étant donnée celle de D, puis finalement une configuration de A tant donnée celle de (B,C,D). En quelque sorte en inversant le flux des influences, on peut échantillonner la probabilité jointe.

Le premier modèle est la typiquement un **champ de Markov non-directionnel** plus compliqué à échantillonner que le second représentant un **champ de Markov directionnel**. Cependant, établir des relations directionnelles est beaucoup plus compliqué: il est plus facile de savoir si deux variables sont corrélées que de connaître le schéma de causalité les concernant (voir la conférence de Michèle Sebag). **Tout le Machine Learning actuel est basé sur le cadre non-directionnel**, notamment les réseaux de neurones et même les très gros systèmes de langage.

4.2 Champ de Markov

4.2.1 Définition et propriété

La définition d'un champ de Markov (*Markov Random Field*) est la suivante

Définition 1 (Champ de Markov/MRF)

Un champ de Markov est une distribution de probabilité jointe définie sur un graphe G non-directionnel, dont les nœuds $(x_k)_{k \leq d}$ (v.a) suivent une densité de probabilité que l'on peut sous la forme:

$$p(x_1, x_2, \dots, x_d) = Z^{-1} \prod_{c \in \mathcal{C}} \Phi_c(x_c) \quad (54)$$

où $x_c = (x_{i_1}, \dots, x_{i_k})$ et $(i_1, \dots, i_k) \in c$ et "c" s'appelle une "clique". La constante Z assure la normalisation:

$$\int p(x_1, x_2, \dots, x_d) \prod_{i=1}^d dx_i = 1 \quad (55)$$

La densité de probabilité p est une distribution de Gibbs.

Le terme de *clique* en théorie des graphes désigne un sous-graphe totalement connecté (Fig. 20). La définition de \mathcal{C} permet d'opérer une **factorisation**. Cette factorisation est importante car qu'en bien même deux variables lointaines du graphe sont dépendantes (c'est-à-dire qu'il y a chemin qui les relie), en fait on va modéliser les interactions au sein des cliques de \mathcal{C} qui sont quant à elles **locales**. Donc, on cherche une **représentation de**

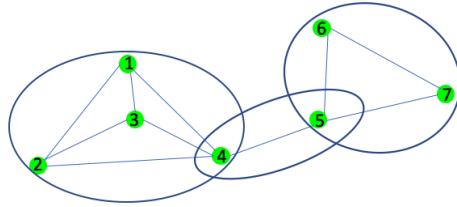


FIGURE 20 – Exemple de graphe non-orienté comprenant 7 "1-vertex" cliques (sommets), 10 "2-vertex" cliques (cotés), 5 "3-vertex" cliques, et 1 "4-vertex" clique. Les trois cliques entourées forment l'ensemble \mathcal{C} .

basse dimension centrée sur le caractère local des interactions (comme on le fait dans une modélisation multi-échelles).

Dans ce cadre de champs de Markov, nous avons une propriété fondamentale. Avant d'établir son énoncé, faisons un petit rappel⁵⁹:

- Soit 2 v.a x_1 et x_2 , elles sont **indépendantes en probabilité** ssi

$$p(x_1, x_2) = p(x_1)p(x_2)$$

- Soit 2 v.a x_1 et x_2 , et y une troisième v.a qui peut influencer les 2 premières, l'**indépendance conditionnelle** des variables x_1 et x_2 s'écrit alors

$$p(x_1, x_2|y) = p(x_1|y)p(x_2|y)$$

Par exemple on en déduit dans ce contexte:

$$p(x_1|x_2, y) = \frac{p(x_1, x_2, y)}{p(x_2, y)} = \frac{p(x_1, x_2|y)p(y)}{p(x_2|y)p(y)} \xrightarrow{\text{indep. cond.}} p(x_1|y)$$

c'est-à-dire que x_2 n'apporte pas d'information sur x_1 si on connaît déjà y . On note cette indépendance conditionnelle selon $x_1 \perp x_2 | y$.

La propriété de Markov nous donne alors:

59. NDJE. Voir cours 2022/2023

Propriété 1 (*Markov*)

Pour un champ de Markov, s'il n'existe pas de clique qui inclue deux variables (x_i, x_j) (c'est-à-dire il n'y a pas d'arête reliant directement les deux variables) alors la variable x_i à une indépendance conditionnelle vis-à-vis de x_j telle que

$$x_i \perp x_j | \{x_k\}_{k \neq i,j}$$

c'est-à-dire

$$p(x_i, x_j | \{x_k\}_{k \neq i,j}) = p(x_i | \{x_k\}_{k \neq i,j}) \times p(x_j | \{x_k\}_{k \neq i,j}) \quad (56)$$

Par exemple, dans la figure 20 les variables x_1 et x_6 sont dans la situation décrite par la propriété, et donc si l'on connaît toutes les autres variables (conditionnement), (x_1, x_6) sont indépendantes. Autrement dit, x_1 et x_6 ne s'influencent pas directement, et l'influence se fait via les autres variables du graphe.

Démonstration 1. Nous sommes en présence d'un graphe support d'un champ de Markov. L'ensemble \mathcal{C} peut être mis sous la forme d'union disjointe de sous-ensembles de cliques selon:

$$\mathcal{C} = \mathcal{C}_{i,j} \cup \mathcal{C}_{i,j'} \cup \mathcal{C}_{i,j} \cup \mathcal{C}_{i,j'} \quad (57)$$

où les sous-ensembles sont indiqués par le fait que x_i et/ou x_j appartiennent ou non à l'ensemble. L'hypothèse stipule alors que $\mathcal{C}_{i,j}$ est l'ensemble vide. Alors on peut écrire

$$\begin{aligned} p(x_i, x_j | \{x_k\}_{k \neq i,j}) &= \frac{p(\{x_k\}_{k \leq d})}{p(\{x_k\}_{k \neq i,j})} \\ &= \frac{\prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c) \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c) \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c)}{p(\{x_k\}_{k \neq i,j}) \int \prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c) \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c) \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c) \times dx_i dx_j \prod_{k \neq i,j} dx_k} \end{aligned} \quad (58)$$

On peut dispatcher les différents éléments de l'intégrale qui se factorisent, ainsi on fait apparaître d'une part:

$$p(\{x_k\}_{k \neq i,j}) \int \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c) \prod_{k \neq i,j} dx_k = \prod_{c \in \mathcal{C}_{i,j'}} \Phi_c(x_c) \quad (59)$$

qui se simplifie avec le terme correspondant du numérateur. D'autre part, les intégrales dépendant de x_i pour l'une et de x_j pour l'autre, se factorisent aussi. Donc

$$\begin{aligned} p(x_i, x_j | \{x_k\}_{k \neq i,j}) &= \frac{\prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c)}{\int \prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c) dx_i} \times \frac{\prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c)}{\int \prod_{c \in \mathcal{C}_{i,j}} \Phi_c(x_c) dx_j} \\ &= p(x_i | \{x_k\}_{k \neq i,j}) \times p(x_j | \{x_k\}_{k \neq i,j}) \end{aligned} \quad (60)$$

La dernière égalité s'obtenant en réinjectant dans chaque terme de la première ligne au numérateur le membre de droite de l'équation 59 et au dénominateur le membre de gauche de la même équation. En définitive donc on a obtenu l'équation qui indique l'indépendance conditionnelle de (x_i, x_j) connaissant les valeurs de $\{x_k\}_{k \neq i,j}$. ■

4.2.2 Deux exemples génériques

4.2.2.1 Processus Gaussien

Voyons quelques exemples d'applications, en premier lieu **le processus gaussien**. Notons au passage que pour un champ markovien on peut écrire

$$\log p(x) = -\log Z + \log \prod_{c \in \mathcal{C}} \Phi_c(x_c) \quad (61)$$

Or pour une distribution de Gibbs traditionnellement on a

$$\log p(x) = -\log Z - U(x) \quad (62)$$

On identifie donc **le terme des cliques comme le terme d'énergie** $-U(x)$. Dans le cas gaussien $U(x) = \frac{1}{2}x^T K x$. On sait que la matrice K est reliée à la matrice de covariance $K^{-1} = \mathbb{E}(xx^T)$. La propriété de Markov nous dit que si $K_{i,j} = 0$ alors $x_i \perp x_j | \{x_k\}_{k \neq i,j}$. On peut donc voir K directement comme la matrice d'interaction.

Un exemple utilisé notamment en Physique, concerne le cas où $K = \Delta$ (**Laplacien**). Mettons que x soient des pixels d'une image (grille en sous-jacent) alors le Laplacien

discret peut se représenter selon

$$\Delta = \begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{bmatrix} \quad (63)$$

qui représente les interactions d'un pixel avec ses voisins. **Ici seuls comptent ses plus proches voisins** en l'occurrence. Donc, maximiser la probabilité implique de minimiser l'énergie donc revient à maximiser l'interaction laplacienne ce qui favorise des formes de régularités, mais les valeurs des x vont fluctuer quand même. Maintenant, on peut réécrire le terme d'interaction selon⁶⁰ (moyennant des conditions aux limites) selon

$$-\frac{1}{2}x(u)^T \Delta_x x(u) = -\frac{1}{2} \int x(u) \nabla_u x(u) du = \frac{1}{2} \int |\partial_u x(u)|^2 du = \frac{1}{2} \sum_{i,j \in \mathbb{C}} |x_i - x_j|^2 \quad (64)$$

où les couples indices i, j sont tels que les j sont pris dans des cliques correspondant aux plus proches voisins des i (on peut noter cette propriété selon (i, j)). La probabilité devient

$$p(x) = Z^{-1} \exp \left\{ -\frac{1}{2} \sum_{(i,j)} |x_i - x_j|^2 \right\} \quad (65)$$

qui a l'**expression d'un mouvement brownien**⁶¹.

4.2.2.2 Théorie ϕ^4

Comme second exemple, S. Mallat nous montre une modélisation de ce qu'on appelle **la théorie ϕ^4** en théorie des champs qui est tirée **du modèle d'Ising**⁶². La théorie ϕ^4

60. NDJE. En Physique Statistique (th. des champs), en fait on fait l'inverse car le terme $\partial_\mu \phi(x) \partial^\mu \phi(x) = |\partial_\mu \phi(x)|^2$ n'est autre que le terme d'énergie cinétique.

61. NDJE. Dans sa forme simple, un mouvement brownien peut se simuler par un processus tel que $X_t = X_{t-1} + Z_t$ où Z_t est une $v.a$ qui suit une loi normale, donc $p(X_t, X_{t-1}) \propto \exp\{\frac{1}{2}(X_t - X_{t-1})^2\}$.

62. NDJE. Le problème que Lars Onsager (1903-76) a résolu exactement en 1944 est le depuis fameux modèle d'Ising 2D: ce modèle de spins en interaction a été introduit par Wilhelm Lenz (1888-1957) en 1920 et son étudiant Ernest Ising (1900-98) l'avait résolu en 1D uniquement et n'avait pu trouver de transition de phase. La résolution exacte de Onsager a permis d'en comprendre le sens et l'étude des exposants critiques et le développement en Mécanique Statistique de la Théorie des Équations du Groupe de Renormalisation qui mèneront à bien des résultats comme dans la Théorie du Modèle Standard des Particules Élémentaires (phénomène de Brout-Englert-Higgs nobélisé en 2013). La théorie ϕ^4 est une généralisation pour des spins à valeurs continues.

est issue des travaux de Vitaly Lazarevich Ginzburg (1916-2009) et Lev Davidovich Landau (1908-68). S. Mallat discute d'un modèle légèrement différent de celui classiquement étudié. Sur une grille $L \times L$ l'énergie est donnée par

$$U(x) = \frac{\beta}{2} \sum_{(i,j)} (x(i) - x(j))^2 + \sum_i (x(i)^2 - 1)^2, \quad p(x) \propto e^{-U(x)} \quad (66)$$

Dans ce cadre le potentiel est fixé, par contre ce qui varie c'est le facteur β de pondération du terme brownien par rapport au terme de potentiel. Selon la valeur de ce paramètre (l'inverse d'une sorte de température), et dans la limite où $L \rightarrow \infty$, il y a une compétition entre les deux termes. Une **transition de phase** apparaît quand $\beta = \beta_c \approx 0.68$. En effet,

- Quand $\beta \ll \beta_c$, le terme cinétique (Laplacien) devient négligeable (il disparaît de la modélisation quand $\beta = 0$) et seuls comptent les termes de potentiels qui sont séparables. Alors il y a indépendance entre les différentes variables x_i :

$$p(x) \propto \prod_i \exp\{-V(x(i))\} \quad (67)$$

le système est désordonné.

- Quand $\beta = \beta_c$, le terme brownien tend à favoriser des configurations smooths tandis que le terme de potentiel tend à faire en sorte que les x_i prennent leurs valeurs dans $\{-1, +1\}$. C'est un système où l'ordre se propage sur des échelles bien plus grandes que l'échelle locale (cela se matérialise également dans la forme du spectre de puissance pour $\beta = \beta_c$). Il s'agit d'une compétition ordre/désordre qui donne lieu à un phénomène de **transition de phases**.
- Enfin pour $\beta \gg \beta_c$, les configurations sont uniquement guidées par les interactions à courte portée. Si la configuration initiale comporte plus de 1, cette valeur tend à se propager à tout les x_i (et symétriquement si initialement il y a plus de -1). On a deux phases distinctes soit tous les $x_i = 1$ ou bien tous les $x_i = -1$. En fait, il y a des petites fluctuations locales autour de ces deux valeurs.

Des simulations des différentes situations selon la valeur de β vis-à-vis de β_c sont montrées sur la figure 21. **Alors que le modèle est purement local, on note les corrélations au-delà des plus proches voisins dans les simulations dans lesquelles $\beta \geq \beta_c$.** On assiste alors à l'extension de zones où les valeurs de x_i prennent les valeurs ± 1 .

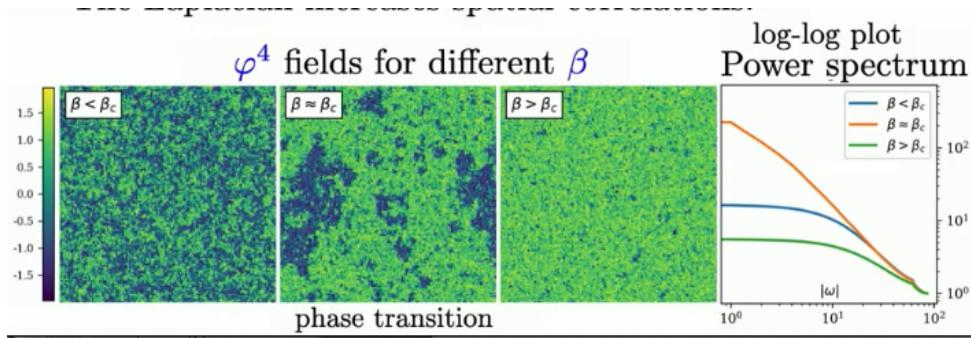


FIGURE 21 – Simulations de configurations du modèle Eq. 66.

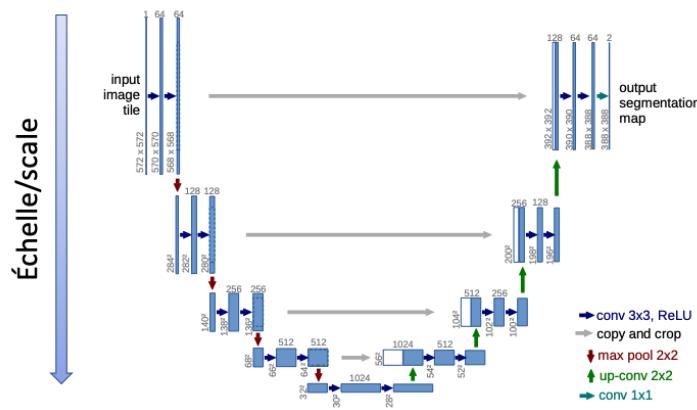


FIGURE 22 – Architecture multi-échelles d'un U-Net.

Le point important est que **pour comprendre l'émergence des structures qui sont la manifestation d'interactions entre champs à longue portée, il faut séparer les différentes échelles**, car il semble y avoir une organisation hiérarchique des interactions, dont le concept est apparu en Physique avec la notion de **Groupe de Renormalisation**⁶³. Ce qui est remarquable c'est que ces idées développées en Physique, on les retrouve dans les réseaux de neurones. Par exemple, un U-Net prend une image, la décompose à différentes échelles, et son architecture permet de modéliser les **interactions multi-échelles** (Fig. 22). Finalement, si les grands modèles génératifs qui manipulent des distributions de probabilités en grande dimension sont apprenables, c'est qu'**il y a en sous-jacent des factorisations à l'œuvre**. S. Mallat nous donne un exemple avec la génération de visages où l'on peut imposer des interactions locales entre pixels à cause des phénomènes multi-échelles du même type que ceux de la Physique.

4.3 Propriétés d'indépendance

Nous allons étudier la notion d'indépendance en prenant des groupes de variables comme illustré sur la figure 23. Typiquement, lorsque que l'on va modéliser une image, au lieu de considérer les interactions entre pixels, on s'intéresse aux interactions entre des blocs constitués des structures. Remarquez dans l'exemple que toutes les variables x_i du bloc X ne sont pas directement connectées (en interaction) avec les variables x_j du bloc Z .

Propriété 2

Dans un champ de Markov, si il n'existe pas d'arête entre X et Z , c'est-à-dire que

63. NDJE. Cette théorie a été initiée en Théorie des Champs en Physique de Particules en 1954 par Murray Gell-Mann (1929-2019) et Francis E. Low (1921-2007) dans le cadre de la QED (Quantum Electrodynamics), puis elle fut généralisée par Curtis Callan et Kurt Symanzik (1923-83) par l'établissement de ce que l'on appelle les équations de Callan-Symanzik. Les développements en Mécanique Statistique datent du Ph. D de Kenneth G. Wilson (1936-2013) obtenu sous la direction de Gell-Mann en 1961. Wilson fait le lien avec les développements en Théorie des Champs et développe la théorie des exposants critiques en lien avec les transitions de phases qui sera un thème de choix du domaine dans les années 70s comme le fameux "Les Houches Session XXVIII (1975): Methods in Field Theory" avec des contributions remarquables.

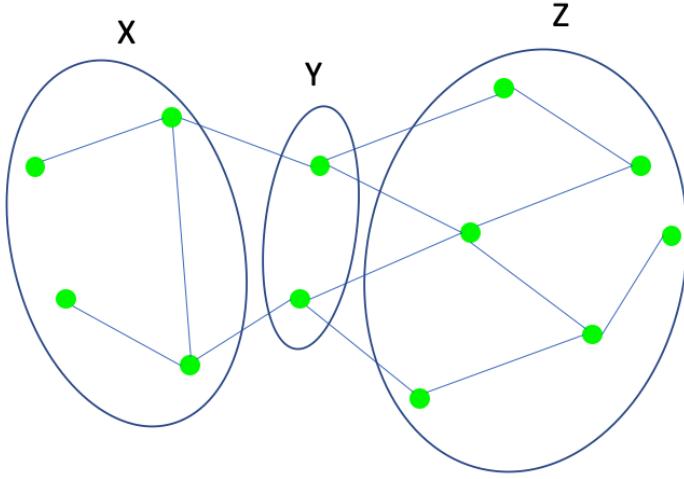


FIGURE 23 – Exemple de 3 variables (X, Y, Z) constituées de variables $(x_i)_i$ d'un graphe G (non-directionnel).

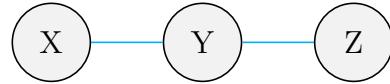


FIGURE 24 – Décomposition d'un graphe en 3 blocs (X, Y, Z) avec des connections non-directionnelles.

Y est une frontière entre X et Z, alors

$$X \perp Z|Y \quad (68)$$

telle que

$$p(X, Z|Y) = \frac{p(X, Y, Z)}{p(Y)} = \frac{f_1(X, Y)}{\sqrt{p(Y)}} \frac{f_2(Y, Z)}{\sqrt{p(Y)}} = p(X|Y) p(Z|Y) \quad (69)$$

La démonstration suit la même démarche que celle adoptée pour la propriété de Markov (Prop. 1) en considérant la décomposition en cliques en ayant $x_i \in X$ et $x_j \in Z$ et les $x_k \in Y$.

La réciproque est la suivante:

Propriété 3

Si dans un graphe G , il existe des blocs de variables X, Y, Z tels que

$$X \perp Z|Y \quad (70)$$

alors on peut écrire la distribution de probabilité sous forme d'un graphe dont la morphologie est celle de la figure 24. C'est-à-dire qu'il n'existe pas d'arête connectant directement les variables de X aux variables de Z .

Quand on écrit la probabilité jointe, cela se décompose facilement en utilisant la propriété d'orthogonalité conditionnelle:

$$\begin{aligned} p(X, Y, Z) &= p(X, Z|Y)p(Y) = p(X|Y)p(Z|Y)p(Y) \\ &= p(X|Y)\sqrt{p(Y)} \times p(Z|Y)\sqrt{p(Y)} = f_1(X, Y) \times f_2(Z, Y) \end{aligned} \quad (71)$$

qui indique que l'on a bien la factorisation souhaitée. On a donc une équivalence entre une indépendance conditionnelle et une factorisation de la distribution de probabilité en facteurs qui ne mettent pas en relation directe des variables. Notez bien que les blocs X et Z ne sont pas indépendants, c'est-à-dire que $p(X, Z) \neq p(X)p(Z)$, et **Y joue le rôle d'une variable cachée**. Attention, avec de tels types de graphes on ne peut rendre compte de la situation du graphe de la figure 25 où X et Z sont indépendants mais induisent Y (il y a une causalité). **Donc, les graphes de Markov n'épuisent pas toutes les possibilités d'indépendance.**

La généralisation de l'équivalence que nous venons de voir sur 3 groupes (X, Y, Z) est le résultat du **théorème de Hammersley-Clifford**. Tout d'abord énonçons une définition.

Définition 2 *On dit qu'un graphe G est une I-map (Information map/carte d'information) si toutes les propriétés d'indépendance conditionnelle sont dans la topologie*

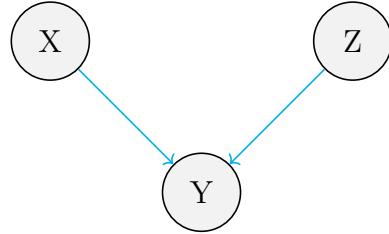


FIGURE 25 – Décomposition d'un graphe en 3 blocs (X, Y, Z) mais avec des connections directionnelles.

de G^a .

a. NDJE. c'est-à-dire que les indépendances conditionnelles de blocs se reflètent dans les connexions entre les noeuds.

Théorème 3 (Hammersley-Clifford)

Soit une probabilité positive $p(x_1, \dots, x_d) > 0$ où les x_i sont des v.a à valeurs dans χ , et soit un graphe G non-directionnel, alors G est une I-map sur p ssi p est une distribution de Gibbs sur le graphe.

Pour la démonstration, il faut d'abord montrer que l'on est capable d'écrire $p(x_1, \dots, x_d)$ sous forme d'un produit de facteurs sur des cliques (cf. champ de Markov), et de la donner sous forme d'indépendances conditionnelles entre blocs de variables. On a démontré que si $p(x)$ peut se mettre sous forme d'un produit de facteurs, alors il y a des indépendances conditionnelles entre certains blocs.

La réciproque demande à démontrer que si on a un ensemble de conditions de type " $X \perp Z|Y$ ", alors on peut construire sur G des cliques telles que la distribution de probabilité se factorise en produits selon la définition d'un champ de Markov (Def. 1). On l'a démontré dans le cas de 3 groupes (Prop. 3). Voici le principe de la généralisation. Pour chaque noeud du graphe (x_i fixé), on considère l'ensemble de ses voisins (ie. les noeuds en connexion avec lui)

$$Y_i = \{x_j \in G | x_j \text{ connexe à } x_i\}$$

Par construction, Y_i est la frontière entre x_i et tous les autres noeuds $x_k \notin Y_i$. Donc, on

peut écrire que

$$p(x_1, \dots, x_d) \propto \Phi_i^+(x_i, Y_i) \Phi_i^-(G \setminus \{x_i\})$$

où $G \setminus \{x_i\}$ est l'ensemble des noeuds de G différents de x_i . On peut faire ce genre de décomposition pour tout x_i . Il faut ensuite regrouper toutes ces factorisations pour obtenir une factorisation minimale. Ce processus correspond à effectuer des intersections entre les contraintes.

Voyons un exemple pour faire le lien avec les notions de multi-échelles: il s'agit du **mouvement brownien en 1D** (marche aléatoire⁶⁴). Soit $x_0 \sim \mathcal{N}(0, \sigma^2)$ par exemple, et le processus qui fait passer x_n à x_{n+1} ($n \in \mathbb{N}$) est

$$x_{n+1} = x_n + z_{n+1}, \quad z_n \stackrel{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2) \quad (72)$$

Nous avons $x_n = x_1 + z_1 + \dots + z_n$, ainsi

$$\begin{aligned} \mathbb{E}(x_n) &= 0 & \mathbb{E}(z_i z_j) &= \mathbb{E}(z_i) \mathbb{E}(z_j) = 0 \\ \mathbb{E}(x_n^2) &= (n+1)\sigma^2 & \mathbb{E}(x_n x_{n+k}) &= (n+1)\sigma^2 \end{aligned} \quad (73)$$

Donc, il y a de la corrélation à grande distance.

Maintenant, si on considère la chaîne $(x_0, x_1, x_2, x_3, \dots)$, on veut construire un arbre de dépendances 2-à-2. Pour cela, prenons (x_0, x_1) et construisons 2 nouvelles variables:

$$x_0^+ = \frac{x_0 + x_1}{\sqrt{2}} \quad x_0^- = \frac{x_0 - x_1}{\sqrt{2}} \quad (74)$$

et faisons la même chose pour (x_2, x_3) , (x_4, x_5) , etc. Toutes les paires (x_{2k}, x_{2k+1}) ($k = 0, 1, \dots, n/2$) sont ainsi transformées en paires $(+, -)$. Toutes les variables de type $-$ sont toutes indépendantes entre elles, par contre elles sont reliées aux variables de type $+$. Ces variables de type $+$, on peut de nouveau les regrouper 2-à-2: ex. (x_{2k}^+, x_{2k+2}^+) et procéder comme précédemment en effectuant leur somme et leur différence par exemple:

$$x_0^{+\pm} = \frac{x_0^+ \pm x_2^+}{\sqrt{2}} \quad x_4^{+\pm} = \frac{x_4^+ \pm x_6^+}{\sqrt{2}} \quad (75)$$

Or, $x_0^{+-} = f(z_0, z_1, z_2)$ et $x_4^{+-} = f(z_4, z_5, z_6)$ donc sont indépendantes. Et cela vaut pour

64. NDJE. Voir Cours 2023 Sec. 6.4.2.2

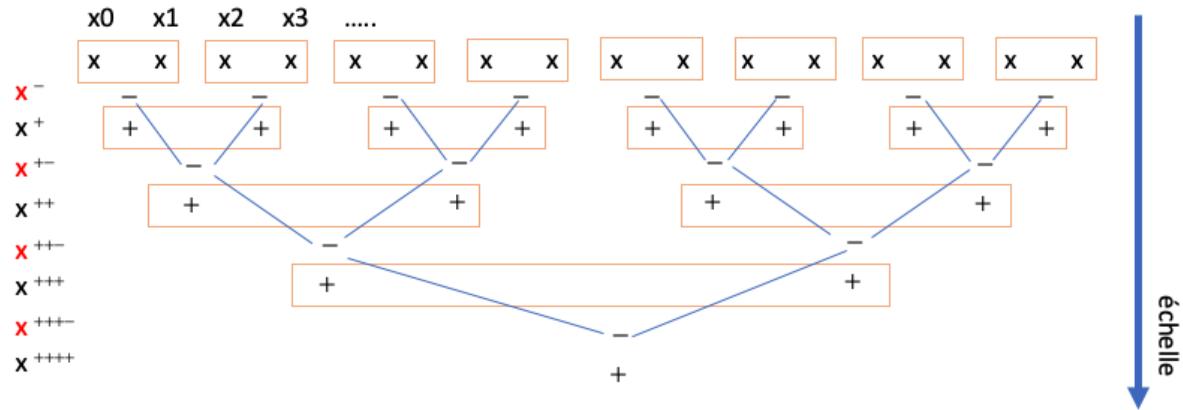


FIGURE 26 – Schématisation d’une cascade de transformations orthogonales opérées sur les variables initiales x_i (de la marche aléatoire) puis sur les variables de type $x^{++\dots+}$. Les variables $x^{++\dots-}$ sont indépendantes entre-elles à une étape donnée, mais dépendent de celles de l’étape précédente 2-à-2.

toutes les x_{2p}^{+-} . Remarquons que x_0^{+-} dépend de x_0^- et de x_2^- , de même x_4^{+-} dépend de x_4^- et x_6^- . Et ainsi de suite, à chaque itération, les variables de type $x_i^{++\dots+}$ sont orthogonalisées, avec les différences qui sont indépendantes entre-elles mais dépendent de deux différences de l’étape antérieure. On a donc une cascade de transformations orthogonales avec des dépendances comme illustré sur la figure 26.

Donc, on a représenté les **interdépendances à longue portée** entre les variables x_i , par des dépendances locales mais hiérarchiques. Concernant la marche aléatoire à 1D, on peut faire plus simple comme on le verra dans le cours dans une séance prochaine. Dans le cas d’une image cette hiérarchisation fonctionne très bien nous explique S. Mallat. De même, il nous brosse des cas où l’on peut faire des connexions entre les variables obtenues au même niveau de la cascade. Donc, on peut utiliser des graphes qui finalement reflètent des situations complexes qui pourtant n’ont pas tant de connexions que ça. Les relations entre personnes dans une entreprise peuvent être modélisées dans ce type de graphe.

Dans le cas de la marche aléatoire 1D ci-dessus, la cascade de transformations Eq. 74

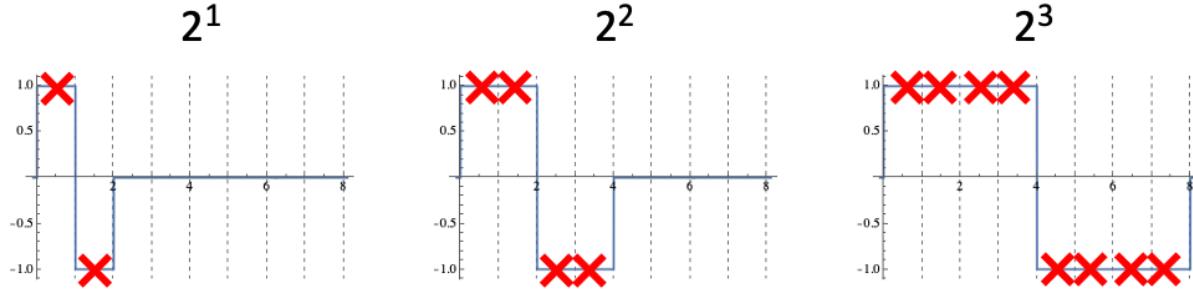


FIGURE 27 – Ondelette de Haar à différentes échelles $\Psi(x/2^n)$. Elle est utilisée implicitement dans la construction des variables $x_0^{++\cdots-}$ comme dans l'équation 76 en lisant les valeurs données par les 'croix rouges' à une échelle donnée affectées par le coefficient $1/2^{2/n}$.

correspond à la transformation de Haar⁶⁵. Par exemple, si l'on considère la variable x_0^{++-} on peut la voir sous deux angles qui sont des applications de l'ondelette de Haar à deux échelles (Fig. 27)

$$x_0^{++-} = \frac{x_0^{++} - x_4^{++}}{\sqrt{2}} = \frac{(x_0 + x_1 + x_2 + x_3) - (x_4 + x_5 + x_6 + x_7)}{2^{3/2}} \quad (76)$$

On obtient finalement une représentation orthogonale de la série des x_i d'origine, et alors que toutes les variables x_i sont en interaction, considérant **les variables aux différentes échelles, les interdépendances ne sont que locales**. Le champ de Markov n'est plus sur un seul axe "purement temporel" comme dans la marche aléatoire 1D (ou dans le cas du champ de la théorie ϕ^4), mais **il y a un nouvel axe qui apparaît, c'est celui des échelles qui permet de transformer les dépendances en ne conservant que des interactions de courte portée**. On peut appliquer de tels schémas dans beaucoup d'applications des réseaux de neurones.

Il est clair, nous dit S. Mallat, que si les grands modèles génératifs sont apprenables, il doit fondamentalement y avoir en sous-jacent une simplification qui opère. Sinon, s'il n'y a pas de source de simplification, un réseau de neurones ne peut pas fonctionner, à cause de

65. NDJE. voir Cours 2018 Sec. 6.3 et Cours 2021 Sec. 7.3.1

la malédiction de la dimension qui nécessiterait un nombre exponentiel de données. **Donc, vraiment l'enjeu est de trouver ces sources de simplifications.** C'est-à-dire trouver **des représentations des données** qui vont simplifier le schéma d'interdépendance, à le réduire à des **interactions d'extension limitée**. L'exemple de la marche aléatoire est en fait très générique parce qu'il fait émerger **la notion d'échelle qui est omniprésente** dans le monde qui nous entoure.

Finalement, toute la partie d'optimisation avec le maximum de vraisemblance est reportée à la séance prochaine, et nous entrerons dans la problématique de l'échantillonnage. En effet, s'il est naturel de concevoir des graphes de champs de Markov, échantillonner les probabilités est plus complexe ne serait ce qu'il faut pouvoir estimer la constante de normalisation qui est une intégrale en très grande dimension. Nous verrons alors en quoi les chaines de Markov⁶⁶ nous aident.

5. Séance du 7 Fév.

5.1 Introduction

S. Mallat nous invite à regarder des séminaires enregistrés des 1-2 Fev. 2024 qu'il a donnés à ICPT de Triestre devant un public de doctorants et de séniors physique théorique. Les enregistrements sont disponibles sur son site web⁶⁷. C'est un complément qui complète les cours avec une vision sur le domaine de la recherche actuelle.

Le lien entre Physique et Apprentissage Statistique est rappelé à l'aulne de l'*émergence de structures* qui nous renseignent sur les interactions entre les "atomes" des deux domaines. La Physique a un modèle basé sur une bien plus longue histoire. Et dans les années 70, que cela soit en Physique Statistique ou Physique des Particules, la Théorie du Groupe de Renormalisation (voir note de base de page 63) permet de comprendre l'évolution des structures à travers les échelles. En fait, cette idée de la *hiérarchisation* des structures, on la retrouve dans beaucoup de problèmes. S. Mallat nous remémore l'article de l'économiste Herbert Simon⁶⁸ qui propose une thèse selon laquelle si les organismes

66. NDJE. voir Cours 2023 Sec. 6.4.

67. NDJE. Les vidéos sont également disponibles ici www.ictp.it/home/salam-distinguished-lectures-series

68. NDJE. Voir Cours 2020 Sec. 3.2

perdurent, c'est qu'il y a une structure hiérarchique qui garantie la stabilité. Ceci dit, si la lecture d'un tel article est intellectuellement enrichissant, il n'y a cependant aucun cadre mathématique sous-jacent⁶⁹.

Dans la session précédente, nous avons vu que dans les arbres d'interactions, on peut modéliser non seulement des interactions horizontales, locales d'où l'usage des champ de Markov, et en même temps des interactions hiérarchiques (axe des échelles) qui peuvent rendre compte de phénomène d'interaction à longue portée. Le *Groupe de Renormalisation* fonctionne très bien en Physique, mais en Apprentissage Statistique on est confronté à des problèmes plus complexes, et dans ce cadre on constate que les réseaux de neurones fonctionnent très bien. La problème est Pourquoi? **C'est pour cette raison qu'il est intéressant de faire le pont entre ce que l'on sait faire en Physique sur des problèmes qui ne sont pas si simples, et ce que l'on sait faire avec les réseaux de neurones sur des problèmes beaucoup plus compliqués tels ceux que nous avons évoqués au début du cours de cette année.** Comprendre ces liens est un enjeu de la recherche actuelle.

Dans un cadre d'images ou de séries temporelles où l'on peut invoquer une *invariance par translation* (ex. reconnaître un verre dans une photo qui peut être prise selon n'importe quel point de vue), on peut dérouler le schéma suivant: qui dit *invariance par translation*, dit qu'il y a un *groupe* sous-jacent, qui dit *groupe* dit *transformée de Fourier* et dit alors toute l'**Analyse harmonique**. C'est-à-dire que dans ce cadre simple, il y a tout un pan des mathématiques sur lequel on peut s'appuyer pour définir notamment la notion d'*échelle*.

Quand on aborde des *problèmes de générations*, on s'attaque à des problèmes d'une autre complexité. Les techniques par **Score Diffusion** fonctionnent bien actuellement, et il y a besoin de faire le pont entre ces techniques et celles que l'on sait analyser dans un cadre mathématique solide. Notons que la notion de *diffusion* est reliée à celle de **débruitage** qui est un problème étudié au moins depuis les années 40 avec le filtre de Norbert Wiener (1894-1964). Il y a eu beaucoup de techniques qui se sont développées dans un cadre "classique" du débruitage depuis cette époque, mais quasiment du jour au lendemain, les réseaux de neurones ont amélioré considérablement l'efficacité. Étrangement, on a d'un côté pléthores de techniques que l'on contrôle parfaitement, et d'un autre un système non-linéaire qui fonctionne beaucoup mieux, mais dont "le pourquoi" reste un sujet. Sans

69. NDJE. entre les lignes S. Mallat fait référence aux articles de Fisher et Shannon dont les idées perdurent jusqu'à nos jours. Ils furent le sujet du cours de 2022.

doute nous dit S. Mallat, il faut comprendre le lien entre les deux sortes de techniques. Concernant la structure des réseaux à l'œuvre, on retrouve en partie la structure multi-échelles.

5.2 Apprentissage de paramètres

5.2.1 Les cadres de Fisher et Shannon

On se place dans le cadre où l'on a choisi une famille de densités de probabilités $\{p_\theta\}_\theta$. On prend ici une **modélisation de Gibbs** telle que

$$p_\theta(x) = Z_\theta^{-1} e^{-U_\theta(x)} \quad (77)$$

où $U_\theta(x)$ définit une sorte d'énergie paramétrée. Dans le cas d'un réseau de neurones, θ représente l'ensemble des poids du dit réseau. Et donc, ce qui nous importe à présent est de trouver le "meilleur" θ , noté θ^* pour approximer au mieux la vraie⁷⁰ distribution $p(x)$. La seule connaissance *a priori* que l'on a est celle d'une collection de données $\{x_i\}_{i \leq n}$ que l'on suppose être issues d'un tirage *iid* de p . La structure de $U_\theta(x)$ fait partie également de nos *a priori* sur le problème.

Il nous faut définir ce que l'on entend par " p_{θ^*} approche au mieux p ", c'est-à-dire la **métrique** qui nous servira pour l'**optimisation**. Ce problème a été posé par Ronald A. Fisher (1890–1962) en 1922, ce qui a fait le sujet du cours de 2022⁷¹. R. Fisher en donne une solution sous la forme du **maximum de vraisemblance**. L'idée est que l'on va choisir θ^* de telle façon que la probabilité de $p_{\theta^*}(x_i)$, sur les échantillons x_i , soit la plus grande possible. Donc, on veut maximiser en moyenne $p_\theta(x)$ ($x \sim p$) ou bien le $\log p_\theta(x)$ (principe du maximum de vraisemblance/*likelihood*, aussi noté MLE), c'est-à-dire:

$$\theta^* = \operatorname{argmax}_\theta \mathbb{E}_{x \sim p}(\log p_\theta(x)) \quad (78)$$

Il y a une autre façon de faire qui réinterprète ce schéma: c'est un point de vue de la

70. NDJE. rappelons qu'ici nous faisons une hypothèse à avoir que $p(x)$ existe.

71. NDJE. Cours 2022 Sec. 5, et l'article de Fisher <https://doi.org/10.1098/rsta.1922.0009> est disponible sur le site du cours <https://www.di.ens.fr/~mallat/CoursCollege.html>.

Théorie de l'Information de Claude Shannon (1916-2001) en 1948⁷², à travers la notion d'**Entropie** et celle de la divergence de Kullback-Leibler définie selon⁷³

$$D_{KL}(p||q) = \int p(x) \log \frac{p(x)}{q(x)} dx \quad (79)$$

On peut tout de suite voir le lien avec le MLE⁷⁴

$$D_{KL}(p||p_\theta) = \underbrace{\mathbb{E}_p(\log p)}_{-\mathbb{H}(p)} - \mathbb{E}_p(\log p_\theta(x)) \quad (80)$$

où l'on fait apparaître $\mathbb{H}(p)$, l'entropie de p , en l'occurrence indépendante de θ par définition. Donc, **maximiser la vraisemblance revient à minimiser la divergence de Kullback-Leibler**,

$$\theta^* = \operatorname{argmin}_{\theta} D_{KL}(p||p_\theta) \quad (81)$$

On peut se demander pourquoi $D_{KL}(p||p_\theta)$ est une quantité "naturelle" à considérer pour notre problème (en soit on a bien vu pourquoi Fisher introduit le MLE)? En définitive, D_{KL} supporte une **notion de similarité** entre les deux distributions de probabilité. Rappelons deux propriétés fondamentales⁷⁵:

Propriété 4 (**Kullback-Leibler**)

- $D_{KL}(p||q) \geq 0$, et = 0ssi $p = q$;
- si $p(x) = \prod_{i=1}^d p_i(x_i)$ manifestant l'indépendance des v.a x_i ($x = (x_1, \dots, x_d)$),
idem pour $q(x)$, alors

$$D_{KL}(p||q) = \sum_{i=1}^d D_{KL}(p_i||q_i) \quad (82)$$

Notons au passage une interprétation en termes de codage. Si on veut coder la distribution p avec la distribution q , le code n'est pas optimal, et le nombre de bits perdus est donné

72. C. E. Shannon, The Bell System Technical Journal, Vol. 27, pp. 379–423, 623–656, July, October. Également disponible sur le site du cours.

73. NDJE. Je garde la notation des années passées.

74. NDJE. Pour les notations vous verrez que $\mathbb{E}_p(f(x))$ implicitement stipule que $x \sim p$.

75. Voir aussi le Cours de 2019 Sec. 7.2.3 par exemple.

par la divergence de Kullback-Leibler.

La démonstration⁷⁶ de la première propriété utilise l'inégalité de Jensen. La démonstration de la seconde propriété est simple en prenant deux variables (x_1, x_2) par un calcul direct en se servant de la propriété du log à transformer un produit en somme, puis en généralisant à un nombre quelconque de variables.

Ce point de vue de l'usage de D_{KL} est celui des réseaux de neurones tout simplement parce qu'on essaye de modéliser une probabilité: ex. en apprentissage supervisé la classification⁷⁷ peut s'interpréter comme maximiser la probabilité conditionnelle de la classe étant donné l'échantillon x_i , et en apprentissage non-supervisé on retombe sur le même type de problématique.

On est donc toujours dans le schéma de Fisher, ce qui a changé concerne l'expression de p_θ . Elle est passée d'une simple gaussienne, dans l'article original, dont la covariance dépend de θ à des formes plus complexes dans les réseaux de neurones.

5.2.2 Optimisation pour trouver θ^*

Il nous faut maximiser la vraisemblance $\log p_\theta$ ou bien minimiser son opposé $-\log p_\theta$. Ainsi, on considère la fonction de coût (*loss*) $\ell(\theta)$

$$\ell(\theta) = -\mathbb{E}_{x \sim p}(\log p_\theta(x)) = \log Z(\theta) + \int p(x)U(x; \theta) dx \quad (83)$$

si l'on note $U(x; \theta)$ l'argument de l'exponentiel dans l'expression Eq. 77. L'outil pour minimiser $\ell(\theta)$ est la **descente de gradient**⁷⁸. Sous ce vocable, il y a beaucoup d'algorithmes et d'implémentations⁷⁹. Nous avons en préliminaire ce résultat:

Lemme 1

$$\nabla_\theta \ell(\bar{\theta}) = \mathbb{E}_{x \sim p}(\nabla_\theta U(x; \bar{\theta})) - \mathbb{E}_{x \sim p_{\bar{\theta}}}(\nabla_\theta U(x; \bar{\theta})) \quad (84)$$

76. NDJE. Elle peut être trouvée par exemple dans le cours de 2022 Th. 13.

77. NDJE. voir par exemple Cours 2019 Secs. 7.2.3 et 7.3.2.

78. NDJE. Voir Cours 2018 Sec. 10, Cours 2019 Sec.4.2.2, Cours 2022 Sec. 3.6.2

79. NDJE. voir par exemple l'article de S. Ruder, "An overview of gradient descent optimization algorithms", <https://arxiv.org/abs/1609.04747>.

La démonstration est simple il suffit de se rappeler que

$$Z(\theta) = \int e^{-U(x;\theta)} dx \quad (85)$$

qui donne lors de la dérivation du premier terme de la définition Eq. 83

$$\frac{\nabla_{\theta} Z(\bar{\theta})}{Z(\bar{\theta})} = - \int \nabla_{\theta} U(x; \bar{\theta}) p_{\bar{\theta}}(x) dx = -\mathbb{E}_{x \sim p_{\bar{\theta}}}(\nabla_{\theta} U(x; \bar{\theta})) \quad (86)$$

La conséquence immédiate de ce lemme est que

$$\nabla_{\theta} \ell(\theta^*) = 0 \Leftrightarrow \mathbb{E}_{x \sim p}(\nabla_{\theta} U(x; \theta^*)) = \mathbb{E}_{x \sim p_{\theta^*}}(\nabla_{\theta} U(x; \theta^*)) \quad (87)$$

D'une manière générale on appelle **moment** une quantité

Définition 3 (**moment**)

$$\mathbb{E}_{x \sim p}(f(x)) = \int f(x)p(x) dx \quad (88)$$

Dans l'expression 87, $f(x) = \nabla_{\theta} U(x; \theta^*)$ est un vecteur. On a donc égalité entre deux vecteurs (familles) de moments. **Les moments** $\mathbb{E}_{x \sim p}(\nabla_{\theta} U(x; \theta^*))$ **sont des contraintes**⁸⁰ **qui doivent être satisfaites pour la distribution** p_{θ} . On appelle cela une *projection de moments*.

Maintenant la *descente de gradient* en tant que telle est une façon de modifier au fur et à mesure la valeur de θ pour obtenir θ^* . A l'étape $t + 1$, connaissant les valeurs à l'itération t , nous avons

$$\theta_{t+1} - \theta_t = -\varepsilon \nabla_{\theta} \ell(\theta_t) = \varepsilon \left(\mathbb{E}_{x \sim p_{\theta_t}}(\nabla_{\theta} U(x; \theta_t)) - \mathbb{E}_{x \sim p}(\nabla_{\theta} U(x; \theta_t)) \right) \quad (89)$$

On modifie la valeur de θ dans la direction de l'erreur des moments. **Le problème principal de la méthode réside dans le calcul de ces moments.**

80. NDJE. Dans le Cours de 2023 concernant le Th. de Gibbs (Th. 17 Sec. 7.6) ce sont les espérances de fonctions $\phi_k(x)$ ($\mathbb{E}_{x \sim p}(\phi_k(x))$) qui sont les contraintes pour définir p_{θ} . En l'occurrence on montre que p_{θ} prend la forme d'une distribution de Gibbs.

Concernant l'espérance avec la distribution p , on utilise les données initiales $(x_i)_i$ que l'on suppose être des échantillons *iid* de p :

$$\mathbb{E}_{x \sim p}(\nabla_{\theta} U(x; \theta_t)) = \int \nabla_{\theta} U(x; \theta_t) p(x) dx \approx \frac{1}{n} \sum_{i=1}^n \nabla_{\theta} U(x_i; \theta_t) \quad (90)$$

Par contre, concernant l'espérance avec la distribution avec p_{θ_t} , **il faut se construire des échantillons *iid***. Les méthodes MCMC vont nous donner le moyen, mais la mise en œuvre n'est pas facile *a priori*. Notez cependant que d'une part on doit procéder à la construction d'une chaîne suffisamment longue pour que la moyenne converge raisonnablement vers l'espérance, et d'autre part cette étape de génération doit être répétée pour chaque itération de la descente de gradient. **En d'autres termes, même si la méthode paraît claire à mettre en œuvre, elle peut être très lente.** Historiquement, depuis en gros les travaux de Fisher et ceux de Shannon, le paradigme n'avait pas changé jusqu'à 2005 où l'idée du **score matching** a émergé (Aapo Hyvärinen⁸¹). Nous verrons pourquoi cette technique permet d'accélérer le processus, mais il n'y a pas de miracle il y aura un prix à payer.

Maintenant, le paradigme général de choisir une expression de p_{θ} , d'utiliser une fonction à minimiser pour trouver le meilleurs θ par descente de gradient est sans aucun doute ce que l'on a envie de faire, nous dit S. Mallat. On peut vouloir remplacer la divergence de Kullback-Leibler par une autre fonction à minimiser pour accélérer le processus, mais avant il nous faut mieux comprendre les propriétés de cette "métrique".

5.3 Le cas linéaire

Il s'agit d'un cas particulier très important de forme de l'énergie telle que⁸²

$$U(x; \theta) = \sum_{k=1}^K \theta_k \phi_k(x) = \theta^T \Phi(x) \quad (91)$$

81. NDJE. S. Mallat fait allusion à l'article Hyvarinen, A., (2005), "Estimation of non-normalized statistical models by score matching", Journal of Machine Learning Research, Vol 6(Apr), pp. 695–709. <https://jmlr.org/papers/volume6/hyvarinen05a/hyvarinen05a.pdf>

82. NDJE. Voir par ex. Cours 2022 Sec 4.2, Cours 2023 Sec. 7.6.

où $\theta = \{\theta_k\}_{k \leq K}$ les fonctions $\Phi = \{\phi_k\}_{k \leq K}$ sont connues à l'avance. C'est une décomposition linéaire sur un espace linéaire

$$U(x; \theta) \in \mathcal{V} = \text{Vect}\{\phi_k\}_{k \leq K} \quad (92)$$

Ce type de paramétrisation est typique de situations rencontrées en Physique par exemple. Notez qu'en Machine Learning Φ est un *feature vector*. Les θ_k peuvent être vus comme des coefficients de couplage.

Dans ce cadre linéaire, le gradient est facile à calculer puisque:

$$\nabla_{\theta} U(x, \bar{\theta}) = \Phi(x) \quad (93)$$

de plus il est indépendant de θ . Les contraintes données par l'égalité entre moments (Eq. 87) se traduisent selon

$$\forall \theta, \quad \mathbb{E}_{x \sim p_{\theta}}(\Phi(x)) = \mathbb{E}_{x \sim p}(\Phi(x)) \quad (94)$$

Nous avons alors la propriété suivante

Théorème 4 (**Entropie Maximum**)

Dire que $D(p||p_{\theta^})$ est minimum, c'est-à-dire que θ^* est optimal (MLE), est équivalent à dire que p_{θ^*} est la distribution qui maximise l'entropie*

$$\mathbb{H}(q) = -\mathbb{E}_q(\log q) = -\int p(x) \log p(x) \, dx \quad (95)$$

et t.q

$$\mathbb{E}_{x \sim q}(\Phi(x)) = \mathbb{E}_{x \sim p}(\Phi(x)) \quad (96)$$

Ce théorème nous dit que l'on peut donc oublier en quelque sorte **le problème paramétrique, en le substituant par la recherche d'une distribution contrainte par des moments et qui satisfait au Principe d'Entropie Maximum**⁸³. C'est une réinterprétation en termes de Théorie de l'Information du problème paramétrique.

83. NDJE. concernant le Principe d'Entropie Maximum voir aussi le Cours 2022 Sec. 7.5

Démonstration 4. On a un problème d'optimisation d'une fonction concave ($\mathbb{H}(q)$) où l'inconnue est $q(x)$ satisfait la collection d'égalités (K contraintes)

$$\forall k, \quad \mathbb{E}_q(\phi_k(x)) = \int q(x)\phi_k(x) dx = \mathbb{E}_p(\phi_k(x)) = \mu_k \quad (97)$$

De plus, q étant une probabilité, $\int q(x)dx = 1$. Or, ce type de problème se résout par les multiplicateurs de Lagrange⁸⁴. Comme $\mathbb{H}(q)$ est strictement concave, la solution au problème est unique.

Donc, soit le lagrangien

$$\mathcal{L}(q, \theta) = \mathbb{H}(q) + \sum_{i=1}^K \theta_k c_k(q) + \theta_0 \left(\int q(x)dx - 1 \right) \quad (98)$$

avec $c_k(q) = \mu_k - \int q(x)\phi_k(x) dx$. NDJE. La démonstration est identique à celle du Cours de 2023 Th. 17 (Gibbs), en requérant que la dérivée de $\mathcal{L}(q, \theta)$ par rapport à la distribution q est nulle, on trouve alors que la distribution optimale s'écrit

$$p^*(x, \theta) = Z(\theta)^{-1} \exp \left\{ - \sum_{k=1}^K \theta_k \phi_k(x) \right\} \quad Z(\theta) = e^{\theta_0 - 1} = \int \exp \left\{ - \sum_{k=1}^K \theta_k \phi_k(x) \right\} dx \quad (99)$$

et les multiplicateurs de Lagrange θ_k sont tels que

$$\int p^*(x, \theta) \phi_k(x) dx = \mu_k \quad (100)$$

on note alors θ^* leurs valeurs. On montre de plus

$$D_{KL}(p||p_{\theta^*}) = \mathbb{H}(p_{\theta^*}) - \mathbb{H}(p) \quad (101)$$

84. NDJE. Voir Cours 2018 Sec. 8.3 Kuhn & Tucker.

En effet

$$\begin{aligned}
 D_{KL}(p||p_{\theta^*}) &= \int p(x) \log \frac{p(x)}{p_{\theta^*}(x)} dx \\
 &= -\mathbb{H}(p) - \int p(x) \left(-\log Z(\theta^*) - \sum_k \phi_k(x) \theta_k^* \right) dx \\
 &= -\mathbb{H}(p) + \log Z(\theta^*) + \sum_k \mathbb{E}_p(\phi_k(x)) \theta_k^*
 \end{aligned} \tag{102}$$

Or, pour la distribution optimale obtenue ci-dessus, $\mathbb{E}_p(\phi_k(x)) = \mathbb{E}_{p_{\theta^*}}(\phi_k(x))$, donc

$$\begin{aligned}
 D_{KL}(p||p_{\theta^*}) &= -\mathbb{H}(p) - \underbrace{\int p_{\theta^*}(x) \left(-\log Z(\theta^*) - \sum_k \theta_k^* \phi_k(x) \right) dx}_{\log p_{\theta^*}} \\
 &= \mathbb{H}(p_{\theta^*}) - \mathbb{H}(p)
 \end{aligned} \tag{103}$$

■

Donc, à partir des contraintes μ_k que l'on estime à partir des $\phi_k(x_i)$ où x_i sont les échantillons initiaux, on impose le critère d'entropie maximum pour trouver la distribution p_{θ^*} . L'entropie maximale va garantir l'uniformisation de la distribution sur des ensembles typiques⁸⁵. En quelque sorte, on ne privilégie pas des échantillons particuliers/configurations particulières dans ces ensembles. En Phys. Statistique, on invoquerait une **statistique micro-canonique** à la Boltzmann. Dans ce contexte, la **divergence de Kullback-Leibler est une mesure de l'excès d'entropie de la distribution p_{θ^*}** car $D_{KL}(p||q) \geq 0$. S'il y a trop d'entropie (ie. $D_{KL}(p||q) > 0$), la question qui se pose est: **quelle est la bonne représentation de Φ ? Le meilleur modèle est celui qui a la plus petite entropie, tout en satisfaisant toutes les contraintes et le Principe d'Entropie Maximale.**

5.4 Score Matching

Nous avons vu lors de la descente de gradient (Eq 89) qu'il faut calculer à chaque étape t l'espérance $\mathbb{E}_{x \sim p_{\theta_t}}(\nabla_{\theta} U(x; \theta_t))$ qui est très couteuse car il faut échantillonner la

⁸⁵. NDJE. Cours de 2022 Sec. 7.5

distribution p_{θ_t} . Mais si l'on revient à l'expression de la *loss* (Eq. 83) que nous remettons ici par commodité:

$$\ell(\theta) = -\mathbb{E}_{x \sim p}(\log p_\theta(x)) = \log Z(\theta) + \int p(x)U(x; \theta) dx$$

on sait que lors du calcul du gradient, le second terme n'est pas un problème, car il fait intervenir $\mathbb{E}_{x \sim p}(\nabla_\theta U(x; \theta_t))$ qui peut s'approximer avec la moyenne sur les échantillons $(x_i)_i$. Donc le problème se concentre en fait dans le calcul du gradient du 1er terme qui nous donne une espérance avec la probabilité p_{θ_t} et qu'il est difficile à échantillonner. **Donc, tout serait plus simple si nous n'avions pas à gérer une constante de normalisation.** Car c'est une intégrale de grande dimension et on doit invoquer de nouveau des MCMC. Comment s'en affranchir?

On peut remarquer que

$$D_{KL}(p||p_\theta) = \int p(x) (\log p(x) - \log p_\theta(x)) dx \quad (104)$$

qui fait apparaître la différence des log. Pour éviter $Z(\theta)$ du second terme, il "suffit" de considérer le gradient par rapport à x , et de faire en sorte d'avoir une quantité positive à minimiser. Ainsi, on regarde la quantité suivante

$$I(p, p_\theta) = \frac{1}{2} \int p(x) \|\nabla_x \log p(x) - \nabla_x \log p_\theta(x)\|^2 dx \quad (105)$$

qui utilise

$$\nabla_x \log p_\theta(x) = \nabla_x (-\log Z(\theta) - U(x; \theta)) = -\nabla_x U(x; \theta) \quad (106)$$

et fait disparaître $Z(\theta)$. La quantité $I(p, p_\theta)$ a été introduite par R. Fisher (déjà en 1923), c'est la **divergence de Fisher**. *NDJE. Attention: le score qui intervient dans l'information de Fisher invoque $\nabla_\theta \log p_\theta(x)$ (Cours. 2022 Sec. 5.3). Or ici notez ∇_x des log des probabilités. Il y a une raison historique⁸⁶ pour laquelle on appelle score les deux formulations,*

⁸⁶. NDJE. L'Information de Fisher est définie (1D) selon

$$I(\theta) = \mathbb{E}_{p_\theta} \left[\left(\frac{\partial \log p_\theta(x)}{\partial \theta} \right)^2 \right] = \int_{\mathbb{R}} p_\theta(x) \left(\frac{\partial \log p_\theta(x)}{\partial \theta} \right)^2 dx$$

mais donc il faut faire attention.

Donc, le Score Matching consiste au lieu de minimiser $D_{KL}(p||p_\theta)$ à minimiser $I(p, p_\theta)$. Le gros avantage c'est que cela se calcule bien plus facilement. Ceci dit $D_{KL}(p||p_\theta)$ est la "métrique naturelle", ce qui en faisait le choix *a priori*. En fait, l'idée d'utiliser la divergence de Fisher n'est venue sur le devant de la scène parce que la communauté a été confrontée aux problèmes de calculs pratiques en grande dimension. Cette métrique est utilisée dans les grands modèles génératifs.

Il nous faut étudier le lien entre $D_{KL}(p||p_\theta)$ et $I(p, p_\theta)$ qui n'est pas *a priori* le choix optimal de métrique. Les questions qui se posent sont: obtient-on la même distribution p_{θ^*} ? et est-ce que cela va converger avec un nombre d'échantillons raisonnables? C'est cette seconde question qui est la plus critique en fait. Dans ce contexte vont apparaître les constantes de *log-Sobolev* qui font un lien avec un chapitre en Analyse qui étudie les équivalences entre métriques sur des fonctions.

5.5 Etude d'un exemple: potentiel ϕ^4

On reprend l'exemple tiré de la Physique vu au paragraphe 4.2.2.2. Pour mémoire le terme d'énergie prend la forme Eq. 66 suivante:

$$U(x) = \frac{\beta}{2} \sum_{(i,j)} (x(i) - x(j))^2 + \sum_{i=1}^N V(x(i)), \quad p(x) \propto e^{-U(x)} \quad (107)$$

qui contient un terme d'énergie cinétique (nb. discrétisation du Laplacien après intégration par partie et (i, j) désigne des couples de spins plus proches voisins sur une grille $N \times N$)

avec $p_\theta(x)$ une pdf paramétrée par θ . A partir de n'importe quelle pdf $p(x)$, on peut définir une forme paramétrique $p(x - \theta)$. Dériver par rapport à θ ou par rapport à x est équivalent. Alors

$$I(\theta) = \int_{\mathbb{R}} p(x - \theta) \left(\frac{\partial \log p_\theta(x - \theta)}{\partial \theta} \right)^2 dx$$

or, par changement de variable cette quantité est indépendante de θ et peut s'écrire

$$I[p] = \int_{\mathbb{R}} p(x) \left(\frac{\partial \log p(x)}{\partial x} \right)^2 dx$$

qui peut s'interpréter comme l'Information de Fisher de la distribution de probabilité p . La notion de divergence suit.

et un terme de potentiel qui pour la théorie ϕ^4 est en forme de 2 puits attracteurs $x = \pm 1$ ($V(t) = (t^2 - 1)^2$).

On veut trouver la loi sous-jacente alors que l'on ne dispose que d'exemple $\{x_i\}_{i \leq n}$ (nb. attention l'indice i indique l'échantillon $x_i(k)$ désigne le pixel/spin k de ce dernier). On ne connaît pas *a priori* la forme 107 de l'énergie. Cette situation est typique des problèmes en Apprentissage Statistique. Ce qui est différent de la Physique où le problème a été étudié depuis longtemps pour avoir une idée de $U(x)$. Donc, que faire? D'après ce que l'on vu dans les sections précédentes, on se définit un modèle, c'est-à-dire ici *une famille d'énergies* $U_\theta(x)$. Mettons que l'on ait *une information a priori* qu'il y ait deux termes:

$$U_\theta(x) = \frac{1}{2}x^T K x + \sum_i V_\gamma(x(i)) \quad (108)$$

le premier est la forme d'un *modèle gaussien*, et le second est une forme d'un *potentiel scalaire paramétrisé* (γ). Si on impose une stationnarité, donc invariance par translation, alors K est un *opérateur de convolution*⁸⁷ de support finit:

$$\begin{aligned} x^T K x &= \sum_i x(i)(K * x)(i) = \sum_{i,j} x(i)K(i-j)x(j) \\ &= \sum_{i,\tau} x(i)K(\tau)x(i-\tau) = \sum_\tau K(\tau) \sum_i x(i)x(i-\tau) \\ &= \sum_\tau K(\tau)(x * \tilde{x})(\tau) = K^T(x * \tilde{x}) \end{aligned} \quad (109)$$

où $\tilde{x}(i) = x(-i)$. On ne connaît pas vraiment le support, donc il nous faut déterminer $K(\tau)$ pour $|\tau| < C$ avec C inconnu. Concernant le terme de potentiel dont la forme est inconnue, on peut juste invoquer une paramétrisation dans une base de fonctions (ex. fonctions linéaires par morceaux):

$$V_\gamma(t) = \sum_{k=1}^K \gamma_k \rho_k(t) = \gamma^T \rho(t) \quad (110)$$

avec les vecteurs $\gamma = (\gamma_k)_k$ et $\rho = (\rho_k)_k$. On peut par exemple prendre $\rho_k(t)$ un rectificateur linéaire (ReLU) translaté par un facteur dépendant de k . En effet avec 2 ReLus on peut construire une fonction triangle (Fig. 28) à la base de la décomposition en fonctions

87. NDJE. Voir Cours 2023 Sec. 9.3

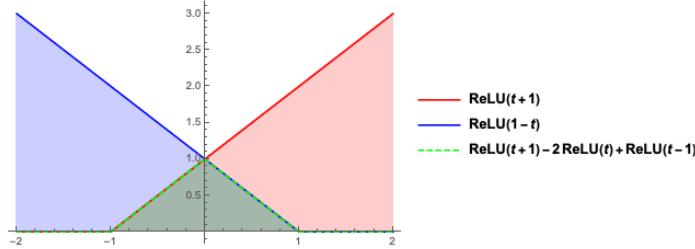


FIGURE 28 – Avec une combinaison linéaire de ReLU on peut obtenir une fonction "triangle" (en vert) (NDJE. extrait Cours 2019 Sec. 5.3.2).

linéaires par morceaux comme conséquence du théorème d'échantillonnage⁸⁸. Mais, on peut tout aussi prendre des combinaisons linéaires de polynômes, etc. On peut écrire le terme de l'énergie correspond au potentiel scalaire comme:

$$\sum_i V_\gamma(x(i)) = \gamma^T \sum_i \rho(x(i)) = \gamma^T \Gamma(x) \quad (111)$$

où $\Gamma_k(x) = \sum_i \rho_k(x(i))$. Ainsi, l'énergie peut s'écrire selon l'expression suivante:

$$U_\theta(x) = \frac{1}{2} K^T(x * \tilde{x}) + \gamma^T \Gamma(x) = \Theta^T \Phi(x) \quad (112)$$

où l'on identifie les paramètres $\theta = (K, \gamma)$ et le *feature vector* $\Phi(x) = \{1/2(x * \tilde{x}), \Gamma(x)\}$. Si on pense à x en tant qu'image, le premier terme est la convolution de l'image avec elle-même, tandis que le second terme peut être la moyenne de ReLUs sur toute l'image.

A présent, ayant identifiée l'expression paramétrique de l'énergie $U_\theta(x)$, on peut dérouler le formalisme décrit dans les sections précédentes. C'est-à-dire trouver θ tel que $\mathbb{E}_{p_\theta}(\Phi(x))$ soit égal aux mesures $\mathbb{E}_p(\Phi(x))$ estimées avec les données $(x_i)_i$. On utilise la descente de gradient avec une métrique (divergence de Kullback-Leibler ou de Fisher). Au final, on dispose d'un modèle de distribution de probabilité décrivant les données. Ce que l'on a envie de faire par la suite, c'est de générer de nouveaux échantillons x iid selon p_{θ^*} . Pour cela il nous faut faire de l'échantillonnage.

Avant cela nous avons à comprendre la différence entre les métriques.

88. NDJE. Cours 2018 Sec. 5.2.4

6. Séance du 14 Fév.

Nous allons étudier un peu plus en détails les différences entre **la divergence de Kullback-Leibler** qui est la "métrique" naturelle pour estimer la meilleure approximation de la distribution des données ($p(x)$), alors que pour des raisons techniques, notamment dans les grands modèles génératifs, on en est venu à utiliser **la divergence de Fisher** (notion de **Score matching**). Par exemple, il nous faut comprendre quand est-ce que la minimisation du score nous donne une bonne approximation de $p(x)$?

6.1 La distance en variation totale

Comme nous l'avons vu à la section 5.2.1, la détermination du jeu de paramètres optimaux θ^* se fait en minimisant la divergence $D_{KL}(p||p_\theta)$ qui décrit la similarité entre la distribution $p(x)$ des données, et un élément $p_\theta(x)$ d'une famille paramétrée. Ainsi,

$$\theta^* = \operatorname{argmin}_\theta D_{KL}(p||p_\theta) \quad (113)$$

Nous avions vu deux propriétés de cette divergence (Prop. 4). Nous allons en donner une autre en relation avec la distance en variation totale⁸⁹

Définition 4 (*distance en variation totale*)

Soit p, q deux densités de probabilité sur \mathcal{E} ensemble mesurable (mettons \mathbb{R}^d), on définit la distance en variation totale selon

$$d_{VT}(p, q) = 2 \sup_{\mathcal{A} \subset \mathcal{E}} |p(\mathcal{A}) - q(\mathcal{A})| = \int_{\mathcal{E}} |p(x) - q(x)| dx = \|p - q\|_{L1} \leq 2 \quad (114)$$

avec $d_{VT}(p, q) = 2$ si les supports de p et q sont disjoints.

La distance en variation totale est définie dans un cadre plus large de la théorie des mesures de probabilité, par la suite on la notera $\|p - q\|_{VT}$.

89. NDJE. je l'ai particularisé pour de *pdf* mais elle est définie pour des mesures au sens plus large.

Pour démontrer l'égalité entre les deux définitions, on peut procéder ainsi. Soit l'ensemble

$$\mathcal{A}_+ = \{x / p(x) > q(x)\} \quad (115)$$

et son complémentaire dans \mathcal{E}

$$\mathcal{A}_+^c = \{x / p(x) \leq q(x)\} \quad (116)$$

Pour $\mathcal{A}_+ \subset \mathcal{E}$ quelconque⁹⁰, nous avons la double inégalité suivante

$$m = p(\mathcal{A}_+^c) - q(\mathcal{A}_+^c) \leq p(A) - q(A) \leq p(\mathcal{A}_+) - q(\mathcal{A}_+) = M \quad (117)$$

On peut démontrer par exemple la seconde inégalité ainsi:

$$\begin{aligned} p(\mathcal{A}) - q(\mathcal{A}) &= \underbrace{p(\mathcal{A} \cap \mathcal{A}_+) - q(\mathcal{A} \cap \mathcal{A}_+)}_{>0} + \underbrace{p(\mathcal{A} \cap \mathcal{A}_+^c) - q(\mathcal{A} \cap \mathcal{A}_+^c)}_{\leq 0} \\ &\leq p(\mathcal{A} \cap \mathcal{A}_+) - q(\mathcal{A} \cap \mathcal{A}_+) \\ &\leq p(\mathcal{A}_+) - q(\mathcal{A}_+) \end{aligned} \quad (118)$$

La démonstration de la première inégalité procède d'une méthode similaire. Maintenant, on peut remarquer que $\mathcal{A}_+ \subset \mathcal{E}$, donc le *supremum* est atteint pour $\mathcal{A} = \mathcal{A}_+$, ainsi

$$\|p - q\|_{VT} = 2|p(\mathcal{A}_+) - q(\mathcal{A}_+)| = 2M \quad (119)$$

Or $p(\mathcal{E}) = q(\mathcal{E}) = 1$, donc $m + M = p(\mathcal{E}) - q(\mathcal{E}) = 0$ d'où $m = -M$. Ainsi

$$\|p - q\|_{VT} = 2M = M - m = \int_{\mathcal{A}_+} |p(x) - q(x)| dx + \int_{\mathcal{A}_+^c} |p(x) - q(x)| dx = \int_{\mathcal{E}} |p(x) - q(x)| dx \quad (120)$$

La divergence $D_{KL}(p||q)$ contrôle la valeur de cette distance selon l'inégalité suivante établie par Mark Semenovich Pinsker (1925-2003):

Théorème 5 (*Inégalité de Pinsker*)

$$\|p - q\|_{VT}^2 \leq 2D_{KL}(p||q) \quad (121)$$

Démonstration 5.

La démonstration suit un schéma typique dans ce domaine. Reprenons le cas des densités

^{90.} NDJE. je simplifie les notations, il faudrait considérer les Boréliens de $mathcal{E}$, et on utilise la mesure de Lebesgue.

de probabilités, et oublions dans les notations l'ensemble \mathcal{E} sur lequel se font les intégrales. On se place également dans les mêmes conditions de validité de $D_{KL}(p||q)$. Ainsi,

$$\|p - q\|_{VT} = \int |p(x) - q(x)| dx = \int q(x) \left| \frac{p(x)}{q(x)} - 1 \right| dx = \mathbb{E}_q[|f(x)|] \quad (122)$$

si $f(x) = \frac{p(x)}{q(x)} - 1 \geq -1$. Notons au passage que $\mathbb{E}_q(f(x)) = 0$. D'un autre coté, on peut écrire

$$\begin{aligned} D_{KL}(p||q) &= \int p(x) \log \frac{p(x)}{q(x)} dx = \mathbb{E}_q[(1 + f(x)) \log(1 + f(x))] \\ &= \mathbb{E}_q[(1 + f(x)) \log(1 + f(x)) - f(x)] \end{aligned} \quad (123)$$

On peut montrer⁹¹ de plus que $\forall y > -1$,

$$(1 + y) \log(1 + y) - y \geq \frac{1}{2} \frac{y^2}{1 + y/3} \quad (124)$$

Donc

$$D_{KL}(p||q) \geq \frac{1}{2} \mathbb{E}_q \left[\frac{f(x)^2}{1 + f(x)/3} \right] \quad (125)$$

Pour conclure, nous utilisons le lemme⁹² suivant sur les v.a X et Y , supposant que $Y \geq 0$

$$\mathbb{E}(|X|^2/Y) \geq \frac{E(|X|)^2}{E(Y)} \quad (126)$$

Finalement, on aboutit à la relation

$$D_{KL}(p||q) \geq \frac{1}{2} \frac{\mathbb{E}_q[|f(x)|]^2}{\mathbb{E}_q[1 + f(x)/3]} = \frac{1}{2} \mathbb{E}_q[|f(x)|]^2 = \frac{1}{2} \|p - q\|_{VT}^2 \quad (127)$$

c'est ce que nous cherchions à démontrer. ■

91. NDJE. *hint:* dans un premier temps on peut faire un développement en série en $y = 0$ des deux fonctions de part et d'autre de l'inégalité pour comprendre le choix de fonction du membre de droite. Ensuite en définissant une fonction f comme différence la fonction du membre de gauche et celle du membre de droite, en la dérivant deux fois, on trouve que f'' est strictement positive sur $(-1, \infty)$. La fonction f' est strictement croissante de $-\infty$ à ∞ et son unique zéro est en $y = 0$. La fonction f est donc strictement décroissante de $(-1, 0]$ et strictement croissante de $[0, \infty)$ avec son minimum en $y = 0$ où elle vaut 0.

92. NDJE. Utiliser Cauchy-Schwartz avec les 2 v.a $|X|/\sqrt{Y}$ et \sqrt{Y} .

Ce théorème nous dit que **la divergence de Kullback-Leibler** est certes asymétrique et n'est pas une distance, elle **contrôle néanmoins une distance en variation totale qui est assez forte**. En particulier minimiser $D_{KL}(p||q)$ contrôle bien le fait que 2 distributions de probabilités vont converger l'une vers l'autre.

6.2 Le Hessien de D_{KL} dans le cas linéaire

A nouveau notre objectif est de trouver la meilleure distribution $p_\theta(x)$ approximant celle des données (notée $p(x)$) et nous avons identifier dans la section 5.2.1 que la "métrique" naturelle est $D_{KL}(p||p_\theta)$. De plus, selon les relations 80 et 83, nous avons

$$D_{KL}(p||p_\theta) = \mathbb{H}[p] + \ell(\theta) \quad (128)$$

mettant en avant le log de la vraisemblance⁹³

$$\ell(\theta) = -\mathbb{E}_p(\log p_\theta(x)) \quad (129)$$

La minimisation par descente de gradient, nous a permis d'établir le lemme 1, telle que la condition $\nabla_\theta \ell(\theta) = 0$ donne

$$\mathbb{E}_{x \sim p}(\nabla_\theta U(x; \theta^*)) = \mathbb{E}_{x \sim p_{\theta^*}}(\nabla_\theta U(x; \theta^*)) \quad (130)$$

C'est-à-dire que la meilleure distribution p_θ est celle qui rend compte des moments ici définis comme le gradient de l'énergie par rapport à θ . Le problème du point de vue calculatoire réside dans le terme de droite comme nous l'avons mis en évidence à la fin de la section 5.2.2. Et dans la section 5.4, nous avons identifié le point réellement dure, à savoir le calcul de la constante de normalisation $Z(\theta)$. C'est ce qui motive l'usage du *score matching*.

Mais avant de voir cela, est-ce que la descente de gradient converge? Nous avons vu,

93. NDJE. Attention au signe par rapport de la définition de la *loss* que j'ai prise.

dans le cas exponentiel où l'énergie a une expression linéaire (Sec. 5.3) de la forme

$$U(x; \theta) = \theta^T \Phi(x) \quad (131)$$

où $\Phi(x)$ est un *features vector* que

$$\nabla_\theta D_{KL}(p||p_\theta) = \mathbb{E}_p(\Phi(x)) - \mathbb{E}_{p_\theta}(\Phi(x)) \quad (132)$$

Cependant, si on veut un problème facile à résoudre avec une descente de gradient, il est souhaitable de se trouver dans un cas convexe, c'est-à-dire qu'il faut étudier le Hessien. Soit alors le théorème suivant:

Théorème 6

Le Hessien par rapport à θ de $D_{KL}(p||p_\theta)$ dans un cas où l'énergie à une forme linéaire $U(x; \theta) = \theta^T \Phi(x)$ ($\Phi = \{\phi_k\}_{k \leq K}$) est tel que

$$H_\theta[D_{KL}(p||p_\theta)] = \text{Cov}_{p_\theta}(\Phi(x)) = \mathbb{E}_{p_\theta}(\Phi(x)\Phi(x)^T) - \mathbb{E}_{p_\theta}(\Phi(x))\mathbb{E}_{p_\theta}(\Phi(x)^T) \geq 0 \quad (133)$$

La démonstration ce fait facilement en considérant l'expression du Hessien $H_\theta = \nabla_\theta \nabla_\theta^T$ et en se servant de l'expression Eq. 86. Donc, comme le Hessien est toujours positif, on sait que le problème est plus simple, la descente de gradient converge vers la solution. Cependant, deux points émergent:

- comme déjà dit, il y a un problème de calcul de $Z(\theta)$ qui nécessite beaucoup d'échantillons;
- le Hessien peut être très mal conditionné, c'est-à-dire que les valeurs propres extrêmes peuvent être très différentes⁹⁴ ce qui obère la vitesse de convergence.

Malgré ces deux points durs cette approche a été utilisée et n'a été remplacée récemment par le score matching. La méthode en gros marchait parce que l'on faisait face à des problèmes de relative basse dimension. Le problème devient de plus en plus aigu au fur et mesure que la dimensionnalité augmente.

94. NDJE. voir Cours 2019 Sec. 8.2.2

6.3 La divergence de Fisher (Score Matching)

La solution trouvée en 2005 par Aapo Johannes Hyvärinen (1970-) (cf. note de bas de page 81) est d'utiliser une nouvelle métrique (Sec. 5.4). On va remplacer

$$D_{KL}(p||p_\theta) = \mathbb{E}_p[\log p(x) - \log q(x)] \quad (134)$$

par la quantité suivante appelée **divergence de Fisher** dans la littérature, reliée à l'Information de Fisher (cf. note de bas de page 86)

$$I(p, p_\theta) = \frac{1}{2} \mathbb{E}_p[\|\nabla_x \log p(x) - \nabla_x \log p_\theta(x)\|^2] \quad (135)$$

qui permet comme on l'a vu de se **débarrasser de la constante $Z(\theta)$ car on prend le gradient par rapport à x** , la norme garantissant la positivité. Donc, l'idée est de minimiser $I(p, p_\theta)$ par descente de gradient:

$$\theta_{SM} = \operatorname{argmin}_\theta I(p, p_\theta) \quad (136)$$

(θ_{SM} est alors le θ du score matching).

La première remarque est que le calcul de la descente de gradient est plus facile, mais la seconde remarque concerne en quoi cette méthode est-elle aussi précise que celle obtenue avec $D_{KL}(p||p_\theta)$? C'est ce dernier point qui est plus délicat. Car, on peut se trouver dans un cas où la méthode utilisant la divergence de Fisher est plus beaucoup lente, et en cela on est confronté au problème de contrôle d'une métrique par une autre. Ce pan d'Analyse (eg. les constantes de log-Sobolev) a été particulièrement développé dans les 30 dernières années, nous dit S. Mallat.

6.4 Réécriture de $I(p, p_\theta)$

Etudions d'abord la partie facile de la convergence par ce théorème:

Théorème 7 (Hyvärinen-2005)

Soit l'hypothèse de régularité $\sup_x \|\nabla_x U_\theta(x)\| < \infty$ alors on peut réécrire $I(p, p_\theta)$ selon

$$I(p, p_\theta) = \mathbb{E}_p \left[\Delta_x U_\theta(x) + \frac{1}{2} \|\nabla_x U_\theta(x)\|^2 \right] + C(p) = J(\theta) + C(p) \quad (137)$$

où $C(p)$ est une constante dépendant de la distribution p .

Nous voyons le gain potentiel dans le fait que $I(p, p_\theta)$ **fait intervenir une espérance selon la distribution** p qui va être facile d'accès via les données $\{x_i\}_{i \leq n}$ représentatives d'échantillons *iid* de la distribution p :

$$I(p, p_\theta) \approx \frac{1}{n} \sum_{i=1}^n \left(\Delta_x U_\theta(x_i) + \frac{1}{2} \|\nabla_x U_\theta(x_i)\|^2 \right) \quad (138)$$

et l'on connaît les formules analytiques de $U_\theta(x)$ donc on peut calculer le gradient et le Laplacien sans souci⁹⁵. Il n'y a plus besoin de méthode de MCMC. Dans le cas d'une forme linéaire de la paramétrisation de l'énergie, on peut obtenir une formule analytique de $I(p, p_\theta)$. Cette méthode qui a l'air miraculeuse marche souvent nous dit S. Mallat, et rappelons que c'est sur cette formulation que sont basés tous les algorithmes de synthèse/génération.

Démonstration 7. Le démonstration suit un procédé classique, si l'on écrit $p(x) = Z^{-1} e^{-U(x)}$ le remplacement du $\log p$ nous donne $-U(x)$ (idem pour p_θ), donc

$$\begin{aligned} I(p, p_\theta) &= \frac{1}{2} \mathbb{E}_p [\|\nabla_x U(x) - \nabla_x U_\theta(x)\|^2] \\ &= \underbrace{\frac{1}{2} \mathbb{E}_p [\|\nabla_x U(x)\|^2]}_{C(p)} + \frac{1}{2} \mathbb{E}_p [\|\nabla_x U_\theta(x)\|^2] - \mathbb{E}_p [(\nabla_x U(x))^T \nabla_x U_\theta(x)] \end{aligned} \quad (139)$$

95. NDJE. qui plus est avec les softwares de calcul automatique des dérivées.

Or,

$$\begin{aligned}
-\mathbb{E}_p[(\nabla_x U(x))^T \nabla_x U_\theta(x)] &= - \int p(x) (\nabla_x U_\theta(x))^T \nabla_x U(x) dx \\
&= - \int \nabla_x U_\theta(x))^T \underbrace{Z^{-1} \nabla_x U(x) e^{-U(x)}}_{\Delta_x U_\theta(x)} dx \\
&= \int (\nabla_x U_\theta(x))^T \nabla_x p(x) dx \\
&\stackrel{IPP}{=} \int p(x) \Delta_x U_\theta(x) dx
\end{aligned} \tag{140}$$

Donc,

$$I(p, p_\theta) = C(p) + \frac{1}{2} \mathbb{E}_p[\|\nabla_x U_\theta(x)\|^2] + \mathbb{E}_p[\Delta_x U_\theta(x)] \tag{141}$$

ce qui est le résultat escompté, si $p(x) \nabla_x U_\theta(x)$ converge vers 0 quand $\|x\| \rightarrow \infty$ qui est bien vérifié grâce à la condition de régularité choisie. ■

6.5 Exemple: le cas linéaire de l'énergie

Ce résultat du Th. 7 est remarquable, rappelons le, car il élimine toutes les constantes de normalisation. Voyons ce que cela donne dans le cas d'une expression de l'énergie linéaire en fonction des paramètres. On sait que la minimisation de la divergence de Kullback-Leibler va parfaitement fonctionner. Dans ce cas, $U_\theta(x) = \theta^T \Phi(x)$, ainsi en omittant le terme $C(p)$, on peut se concentrer sur le terme dépendant de θ

$$J(\theta) = \theta^T \mathbb{E}_p[\Delta_x \Phi(x)] + \frac{1}{2} \mathbb{E}_p[\|\theta^T \nabla_x \Phi(x)\|^2] \tag{142}$$

qui est une expression quadratique en θ convexe donc il n'y a aucune difficulté à la minimiser. Le gradient de J donne

$$\nabla_\theta J(\theta) = \mathbb{E}_p[\Delta_x \Phi(x)] + \mathbb{E}_p[\nabla_x \Phi(x)(\nabla_x \Phi(x))^T] \theta \tag{143}$$

qui donne la solution pour $\nabla_\theta J(\theta) = 0$

$$\theta_{SM} = -\mathbb{E}_p[\nabla_x \Phi(x)(\nabla_x \Phi(x))^T]^{-1} \mathbb{E}_p[\Delta_x \Phi(x)]$$

(144)

Ainsi, on est passé d'un problème qui pouvait s'avérer long à calculer à chaque étape de la descente de gradient, à un problème analytique. Ceci dit, que le *feature* vecteur Φ est de grande dimension, il vaut mieux utiliser la descente de gradient en utilisant à l'étape t l'expression ci-dessus de $\nabla_\theta J(\theta)$ pour $\theta = \theta_t$.

6.6 Convergence du score matching

Tout d'abord on peut se poser la question: est-ce que l'on converge vers un θ identique à celui que l'on obtiendrait par la minimisation de la divergence de Kullback-Leibler? La réponse brève à cette question est "oui, si le modèle est le bon...".

Théorème 8 (*Consistance du score matching*)

Imaginons que $p(x)$ s'écrit comme un élément de la famille $\{p_\theta\}_\theta$ du modèle dans lequel on opère le programme de recherche de la meilleure approximation. Soit donc θ^ tel que $p = p_{\theta^*}$ avec $p_{\theta^*} > 0$ (eg. famille exponentielle), et on suppose que θ^* est unique. Dans ces conditions, $\theta_{SM} = \theta^*$ atteint quand $I(p, p_\theta) = 0$.*

Démonstration 8. On sait que par descente de gradient, on arrive au minimum de $J(\theta)$ qui est atteint quand $I(p_\theta, p_{\theta^*}) = 0$. Donc $\forall x$

$$\nabla_x \log p_\theta(x) = \nabla_x \log p_{\theta^*}(x) \quad (145)$$

qui par intégration donne

$$\log p_\theta(x) = \log p_{\theta^*}(x) + cte \Rightarrow p_\theta(x) = cte \times p_{\theta^*}(x) \quad (146)$$

Or, comme les deux sont des *pdf*, la constante est égale à 1. Ainsi, si la probabilité $p(x)$ appartient à la famille, on trouve le θ qui lui correspond en minimisant la divergence de Fisher. ■

Le problème qui se pose n'est pas tant que cette méthode est plus sujette ou non au fait que p peut ne pas être un élément de la famille de probabilité, **mais a-t-on bien une**

vitesse de convergence plus grande par exemple que part la minimisation de Kullback-Leibler?

S. Mallat nous fait un parallèle avec le théorème qui semble miraculeux d'approximation universelle de fonctions par des réseaux de neurones à 1-couche cachée⁹⁶. En effet, ce théorème finalement est totalement inefficace, car il est contraint par la malédiction de la dimensionnalité: en effet on peut approximer n'importe quelle fonction, mais le prix à payer et qu'il nous faut un nombre de paramètres croissant exponentiellement avec la dimensionnalité du problème. Ce qui est totalement hors sujet pour $d = 10^6$ dans le cas de génération d'images par exemple.

Donc, la question de la vitesse de convergence n'est pas anodine, et **dans le cas du score matching, nous allons voir que cette vitesse est régie par le nombre n d'échantillons et donc la précision que l'on veut avoir pour le calcul de l'espérance.** C'est-à-dire que l'on use de la convergence $((x_i)_i$ échantillons *iid* de $p(x)$)

$$\frac{1}{n} \sum_{i=1}^n F(x_i) \xrightarrow{n \rightarrow \infty} \mathbb{E}_p[F(x)] \quad (147)$$

pour laquelle on sait que les fluctuations en fonction de n sont essentiellement gaussiennes grâce au théorème central limite.

Si l'on revient sur l'expression de θ_{SM} dans le cas linéaire (Eq. 144), on peut par exemple en déduire que

$$\boxed{\sqrt{n}(\theta - \theta_{SM}) \xrightarrow{n \rightarrow \infty} \mathcal{N}(0, C_{SM})} \quad (148)$$

avec C_{SM} une covariance qui dépend de la métrique choisie, ie. la divergence de Fisher. De même, dans le cas du calcul du maximum de vraisemblance (MLE) à l'aide de la minimisation de la divergence de Kullback-Leibler, grâce à la borne de Crameér-Rao⁹⁷ d'un estimateur non-biaisé, on a également que

$$\boxed{\sqrt{n}(\theta - \theta_{KL}) \xrightarrow{n \rightarrow \infty} \mathcal{N}(0, C_{KL})} \quad (149)$$

96. NDJE. Voir Cours 2019 Sec. 5.3

97. NDJE. Voir Cours 2022 Sec. 5.3

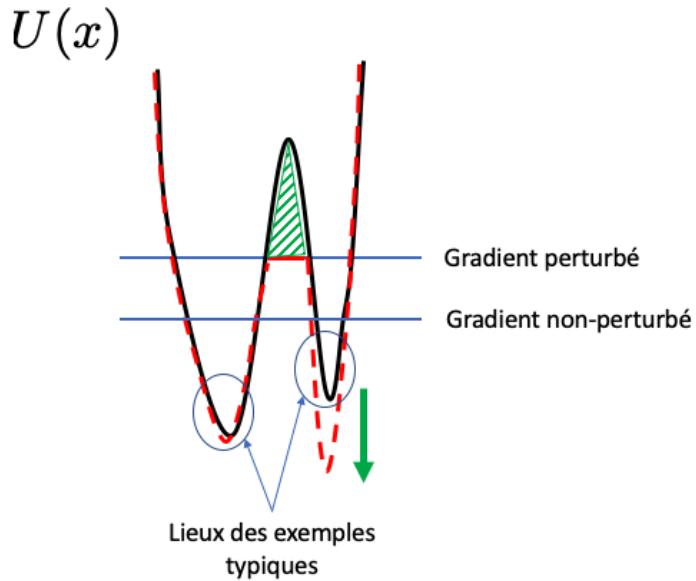


FIGURE 29 – Transformation de $U(x)$ par une coupure (hachure) et une translation du pic secondaire.

avec $C_{KL} = 1/I_1(\theta)$ l'inverse de l'information de Fisher obtenue avec 1 échantillon.

La question qui se pose est quel est le rapport entre C_{SM} et C_{KL} ? Voyons d'abord un exemple avant d'appréhender l'analyse proprement dite du problème.

6.7 Un exemple illustratif du problème du Score Matching

En fait on essaye de contrôler une log-probabilité par son gradient: en effet pour déterminer p_θ ou $-\log p_\theta = Z_\theta + U_\theta$, en utilisant la méthode du score matching via $I(p, p_\theta)$, on élimine la constante de normalisation et on opère un contrôle du gradient $\nabla_x U_\theta$.

Imaginons une fonction $U(x)$ avec des minima locaux comme sur la figure 29, c'est un problème non-convexe. La question est de savoir si l'on peut opérer une transformation de U sans trop perturber son gradient, tout en changeant complètement le minimum? Ce que l'on peut faire c'est d'opérer la transformation qui donne le graphe en rouge pointillé

qui fait une coupure de la barrière de potentielle (zone hachurée) et une descente (par une constante) du second minimum. Il y a d'une part des zones où les gradients sont peu voire pas changés, et d'autre part la zone de coupure qui perturbe les gradients. Mais cette dernière n'est pas le lieu des événements typiques qui se concentrent plutôt dans les minima. Donc, les évaluations des espérances ne sont pas perturbées par le profil modifié de $U(x)$. Ou en d'autres termes **pour apprécier la différences entre le profil de $U(x)$ initial ou sa forme modifiée, il faut énormément d'échantillons pour explorer les queues de distributions et explorer les lieux où les gradients ont été modifiés sensiblement.**

Ainsi pour des problèmes non convexe, ce qui est le cas en règle générale, on a une perte de connaissance du profil exact de $U(x)$ à cause du non-usage de la constante de normalisation. On a un risque de ne pas aller vers la bonne valeur de θ . On peut le dire autrement: la variance C_{SM} risque d'être très grande. La question qui vient alors est: peut-on éviter ce genre de situation?

6.8 Conditions d'usage du Score Matching

Pour que la méthode fonctionne, il nous faut contrôler la divergence de Kullback-Leibler par la divergence de Fisher. C'est dans ce cadre qu'apparaît la notion de **constante de log-Sobolev**⁹⁸ définie de la façon suivante:

Définition 5 (log-Sobolev)

Soit deux densités de probabilité p et q strictement positives, la constante de log-Sobolev $c(p)$ est la plus petite constante telle que

$$\forall q, \quad D_{KL}(q||p) \leq c(p) I(p, q) \quad (150)$$

Notez au passage l'on aurait plutôt eu l'idée de contraindre $D_{KL}(p||q)$ (rappel: D_{KL} n'est pas symétrique) qui nous sert quand $q = p_\theta$. Mais nous allons voir en quoi cette constante $c(p)$ peut nous servir. Notons aussi que **la divergence de Fisher est symétrique**.

98. NDJE. Les inégalités de log-Sobolev ont été découvertes en 1967 par le mathématicien américain Leonard Gross (1931-) lors de ses recherches sur la construction de bases solides de la Théorie Quantique des Champs, notamment l'interaction mutuelle de champs bosoniques qui intervient en Chromosynétique Quantique, et plus généralement dans les théories de jauge non-abéliennes de Yang-Mills. Les publications paraissent en 1975.

Pour voir pourquoi nous avons ce type d'inégalité, nous allons la réécrire comme ceci (usage de la symétrie de $I(p, q)$):

$$D_{KL}(q||p) = \int q(x) \log \frac{q(x)}{p(x)} dx \leq c(p) \frac{1}{2} \int \left\| \nabla_x \log \frac{q(x)}{p(x)} \right\|^2 q(x) dx \quad (151)$$

Soit alors la fonction

$$f(x) = \sqrt{\frac{q(x)}{p(x)}} \quad (152)$$

on a immédiatement que $\int f^2(x)p(x)dx = 1$, et donc f est un élément de l'ensemble des fonctions à carré sommable, normalisée par rapport à la mesure $p(x)dx$ (noté $L^2(p(x)dx)$). Ainsi, on peut réécrire l'inégalité selon

$$\int f^2(x) \log f^2(x) p(x) dx \leq \frac{c(p)}{2} \int \|\nabla_x \log f^2(x)\|^2 q(x) dx = 2c(p) \int \frac{\|\nabla_x f(x)\|^2}{\|f(x)\|^2} q(x) dx \quad (153)$$

Comme $q(x)/\|f(x)\|^2 = p(x)$, l'inégalité initiale se transforme alors en l'inégalité suivante⁹⁹

$$\boxed{\int f^2(x) \log f^2(x) p(x) dx \leq 2c_{LS}(p) \int \|\nabla_x f(x)\|^2 p(x) dx} \quad (154)$$

On est alors plongé dans un autre monde des mathématiques, celui des fonctions dont le carré du gradient est intégrable selon la mesure $p(x)dx$. L'intégrale du membre de droite est une norme de Sobolev qui assure une forme de régularité, le membre de gauche étant une mesure entropique. L'enjeu est alors de contrôler la constante.

Remarquons que pour des applications pratiques de Machine Learning, **ces inégalités sont assez critiques**, au sens que **si on se trouve dans un cas de figure qui contrarie ces inégalités, alors le score matching ne va pas marcher du tout!** Il faut donc vraiment comprendre la nature de ces constantes, et avoir conscience du pourquoi on en est arrivé à utiliser le score matching/divergence de Fisher en lieu et place du MLE/divergence de Kullback-Leibler. Dans **les applications de génération par Score Diffusion**, on utilise non pas un modèle exponentiel avec une énergie linéaire en θ , mais un réseau de neurones dont la sortie est $s_\theta(x)$. Il est entraîné pour calculer $\nabla_x \log p(x)$, mais **la question est de**

99. NDJE. je note la constante c_{LS} car plus loin nous allons voir une autre constante

savoir de combien d'échantillons a-t-on besoin? La réponse à cette question est largement influencée par ces constantes de log-Sobolev.

Pour ce donner une idée de ces constantes $c(p)$, on va faire le lien avec une autre inégalité, plus classique, à savoir l'**inégalité de Poincaré**¹⁰⁰ qui relie la norme de Sobolev à la variance.

Théorème 9 (Inégalité de Poincaré)

Soit une fonction g , dont on regarde la variance en considérant la mesure $p(x)dx$ et dont on note \bar{g} la moyenne:

$$\text{Var}_p(g) = \int (g(x) - \bar{g})^2 p(x)dx \quad \bar{g} = \int g(x) p(x)dx \quad (155)$$

L'inégalité de Poincaré stipule que la constante de Sobolev (notée ici $c_P(p)$) satisfait:

$$\text{Var}_p(g) \leq c_P(p) \int \|\nabla_x g(x)\|^2 p(x)dx \quad (156)$$

Typiquement, on se dit que la variance de g autour de sa moyenne est reliée à la moyenne de ses variations dont le carré de la norme de son gradient est un proxis.

Pour faire le lien entre l'inégalité 154 et celle ci-dessus, posons $f(x) = 1 + g(x).\varepsilon$ avec $\bar{g} = 0$. Notons que

$$f(x)^2 \log(f^2(x)) \approx 2(g(x).\varepsilon) + 3(g(x).\varepsilon)^2 + \dots$$

donc

$$\int f^2(x) \log f^2(x) p(x)dx \approx 3\varepsilon^2 \text{Var}_p(g) \quad (157)$$

De même, $\|\nabla_x f(x)\|^2 = \varepsilon^2 \|\nabla_x g(x)\|^2$, donc si on note $c_P(p)$ la constante de Poincaré et $c_{LS}(p)$ la constante de log-Sobolev de l'expression 154 alors on a une relation du type¹⁰¹

$$c_P(p) = 2/3 c_{LS}(p) \quad (158)$$

100. NDJE. Henri Poincaré (1854-1912)

101. NDJE. On trouve dans la littérature que $c_{LS}(p)$ intègre le facteur $1/2$ dans l'inégalité Eq. 154 alors la relation dans ce cas serait $c_P(p) = 1/3 c_{LS}(p)$. Dans un cas plus général, en gardant cette définition, on trouve une inégalité du type $c_P(p) \leq 1/2 c_{LS}(p)$. Voir par exemple <https://djalil.chafai.net/blog/2023/01/12/log-sobolev-and-bakry-emery/>.

L'inégalité de Poincaré est très importante de le calcul variationnel¹⁰².

Voyons des applications. Prenons **le cas gaussien**, où $g(x)$ est linéaire. Soit e un vecteur de norme 1:

$$g(x) = \langle e, x \rangle = \sum_{i=1}^d e(i)x(i) \quad (159)$$

le gradient de g , $\|\nabla_x g\|^2 = \|e\|^2 = 1$. Regardons l'inégalité de Poincaré, elle nous dit

$$\forall e, tq. \|e\| = 1, \quad \text{Var}_p(g) = \text{Var}_p(\langle e, x \rangle) \leq c_P(p) \quad (160)$$

On peut choisir le pire cas de figure où le vecteur unitaire e réalise le maximum de variance. Quel est le lien avec la matrice de covariance $\text{Cov}(p)$ de $p(x)$? En fait¹⁰³

$$\text{Var}_p(\langle e, x \rangle) = e^T \text{Cov}(p) e \quad (161)$$

Donc, le vecteur propre de $\text{Cov}(p)$ qui a la plus grande valeur propre (notée λ_{max}) réalise le pire cas de figure. Donc,

$$c_P(p) \geq \lambda_{max} \quad (162)$$

Si on revient au problème du potentiel ϕ^4 (Sec. 5.5) non-convexe, au moment de la transition il y a des corrélations de longue portée qui apparaissent. Comme on a affaire à un processus stationnaire, la matrice est diagonalisable dans une base de Fourier. Les valeurs propres constituent la puissance spectrale¹⁰⁴ qui a un comportement typique en $1/|\omega|^\eta$ (Fig. 30) avec $\eta \approx 2$ à cause des discontinuités¹⁰⁵ à petites échelles. Donc, à basses fréquences la matrice de covariance est mal conditionnée. C'est le reflet que sur des zones d'iso-niveaux de gris (couleur) les pixels sont très corrélés, donc il y a des corrélations à longue portée (grande échelle \leftrightarrow petite valeur de ω), et beaucoup de puissance. **Ainsi, la constante $c(p)$ est grande par nature et donc on ne va pas contraindre efficacement la divergence de Kullback-Leibler: "les choses se passent mal!".**

Comment se fait-il cependant que les réseaux de neurones arrivent à s'en sortir?

102. NDJE. Pour les lecteurs intéressés par les relations mathématiques entre les inégalités de log-Sobolev dite aussi de Gross et de Poincaré voir <https://hal.science/hal-00012428v1>

103. NDJE. C'est un résultat que l'on peut tirer de la Sec. 9.3 du Cours 2023.

104. NDJE. Cours 2023 Sec. 9.3

105. NDJE. il y a une coquille dans la légende de la figure 10. Il y a un comportement en $1/k$ à petits k soit à grandes échelles spatiales, et en $1/k^2$ à grands k soit petites échelles.

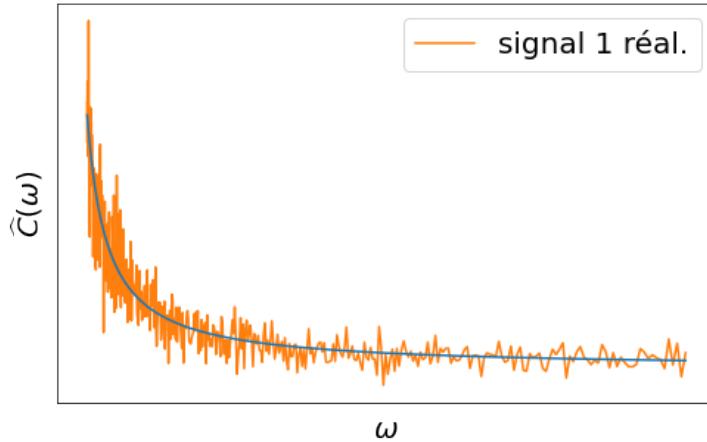


FIGURE 30 – Evolution typique de la puissance spectrale par exemple dans les images.
(*NDJE. ex Cours 2022 Figs. 10 et 11*)

Où comment contourner ce problème de mauvais conditionnement de la matrice de covariance? En quelque sorte les hautes fréquences (petites échelles) sont masquées par les basses fréquences (grandes échelles). *NDJE. Dans le Cours de 2023 Sec. 9.3 nous avons vu qu'une manière est de considérer des fenêtres en ω sur lesquelles le rapport des valeurs propres min/max est bien plus favorable (en fait on s'attachait alors à trouver un estimateur consistant des moments d'ordre 2).* On peut alors effectuer, à toutes les étapes du réseau, des opérations de **BatchNorm**¹⁰⁶ qui vont (re)conditionner le problème pour en quelques sortes faire ressortir les hautes fréquences.

Il y a deux inégalités inverses pour borner supérieurement les constantes de Sobolev $c(p)$. L'enjeu est de pouvoir dire dans quels cas ces constantes ne sont pas trop grandes. Il y a 2 cas dans lesquels on arrive à faire ces calculs. Le premier concerne le **cas convexe** de $U(x)$ et l'on veut contrôler le Hessien par rapport à x . Soit alors le théorème de Dominique Bakry et Michel Émery:

Théorème 10 (Bakry-Emery, 1985) Soit $H_x[U(x)]$ le Hessien de U . Si $\lambda_{min} > 0$ la

106. NDJE. Cours 2019 Sec. 8.2.3

plus petite valeur propre de ce hessien est telle que

$$\forall x, \quad H_x[U(x)] \geq \lambda_{min} \text{ Id} \quad (163)$$

alors

$$c_{LS}(p) \leq 2/\lambda_{min} \quad (164)$$

Ce premier résultat permet de montrer que dans le cas gaussien (g linéaire), alors la constante de log-Sobolev (Eq. 162) du théorème de Poincaré satisfait l'égalité

$$c_P(p) = \lambda_{max} \quad (\text{cas gaussien}) \quad (165)$$

(rappel. où λ_{max} est la plus grande valeur propre de la covariance) et il en est de même¹⁰⁷ pour $c_{LS}(p)$ de l'expression 154. Dans le cas non-gaussien, $U(x)$ n'est pas convexe et le théorème précédent ne s'applique pas.

Le second cas où on a un résultat, c'est **le cas de l'indépendance des composantes** $(x(i))_{i \leq d}$ de x . Alors on sait que l'on peut factoriser $p(x)$ selon

$$p(x) = \prod_{i=1}^d p_i(x(i)) \quad (166)$$

Dans ce cas la constante de Sobolev satisfait l'inégalité suivante:

$$c_{LS}(p) \leq \max_{i \leq d} c_{LS}(p_i) \quad (\text{cas v.a indép.}) \quad (167)$$

En quoi cela aide-t'il à résoudre la plupart des problèmes? Nous avons vu que les problèmes arrivent quand la dimension d augmente. **En fait, cela se traduit par l'augmentation de ces constantes de log-Sobolev avec d . C'est la malédiction de la dimensionalité.** Mais, dans le cas d'indépendance des variables, elles n'interagissent pas, et donc pour sortir des minima locaux en d dimensions, il suffit de sortir selon 1 dimension seulement. *Donc, les phénomènes de non-convexité qui entraînent des constantes plus instables, deviennent un peu plus gérables en 1D.*

Mais encore une fois dans la plupart des problèmes en grande dimension, quand d

^{107.} NDJE. voir Eq.1.4 <https://perso.math.univ-toulouse.fr/cattiaux/files/2013/11/CFG-21-06-22.pdf> avec le changement de définition de $c_{LS}(p)$.

augmente d'une certaine façon les minima deviennent de plus en plus profonds ce qui dégrade sévèrement les constantes comme le disent par exemple C. Domingo-Enrich et A. Pooladiani¹⁰⁸: "[...], pour les distributions multimodales telles que les mélanges de gaussiennes, c_{LS} croît de manière exponentielle en fonction de la hauteur de la barrière potentielle entre les modes...".

Nous reviendrons sur ces considérations car **ces constantes de log-Sobolev définissent la vitesse asymptotique de l'algorithme d'échantillonnage de Langevin**. Cet algorithme effectue une descente de gradient (selon x , cf. le score matching) tout en ajoutant du bruit. Donc, quand il se trouve dans un minimum local, il peut "sauter" la barrière de potentiel par une fluctuation du bruit. Mais bien entendu, plus la barrière est haute plus la probabilité de transition est faible. Vient alors les $c_{LS}(p)$ pour quantifier ces phénomènes et contrôler la vitesse de convergence de l'algorithme.

7. Séance du 28 Fév.

7.1 Introduction

Nous allons aborder dans cette séance et les suivantes, les problématiques de l'échantillonnage et de la génération. Les outils mathématiques sont de natures très différentes de celles vues jusqu'à présent.

On suppose déjà acquise la détermination de la probabilité $p(x)$ sous-jacente aux échantillons donnés au départ. Et pour être concret, on suppose que p est une distribution de Gibbs à savoir

$$p(x) = Z^{-1} e^{-U(x)} \quad (168)$$

Le problème à présent est de **générer des réalisations de cette distribution**. Soit donc ces nouveaux¹⁰⁹ échantillons $\{x_i\}_{i \leq n}$. Si l'on calcule des valeurs moyennes à partir de ces

108. NDJE. l'extrait tiré de l'article de Carles Domingo-Enrich et Aram-Alexandre Pooladiani publié dans Transactions on Machine Learning Research (06/2023) "*An Explicit Expansion of the Kullback-Leibler Divergence along its Fisher-Rao Gradient Flow*", <https://arxiv.org/abs/2302.12229>.

109. NDJE. nb. on prend par facilité la même notation que celles des échantillons initiaux qui ont amené possiblement à la détermination de l'expression de $p(x)$.

échantillons, on doit retrouver les valeurs des espérances obtenues avec $p(x)$. Soit alors la *distribution empirique* $p_n(x)$ forgée à partir de ces échantillons:

$$p_n(x) = \frac{1}{n} \sum_{i=1}^n \delta_{x_i} \quad (169)$$

on s'attend alors à ce que pour toute fonction intégrable

$$\int f(x) p_n(x) dx = \frac{1}{n} \sum_{i=1}^n f(x_i) \xrightarrow{n \rightarrow \infty} \int f(x) p(x) dx = \mathbb{E}_{x \sim p}[f(x)] \quad (170)$$

L'intérêt de pouvoir générer des échantillons est de pouvoir approximer des intégrales en grande dimension par des moyennes empiriques. Donc, le problème est de générer $\{x_i\}_{i \leq n}$ tels qu'ils parcourrent tout l'espace de configuration χ en se concentrant sur l'espace où la probabilité est grande¹¹⁰. L'idée centrale pour réaliser le programme de génération est de créer des **systèmes dynamiques** où les itérés $x(t)$ sont les $\{x_i\}_{i \leq n}$, c'est-à-dire en quelque sorte où l'indice i est un "temps". Les mathématiques sous-jacentes font appel à la notion d'**ergodicité**¹¹¹.

Le cas à 1 dimension est un peu plus simple, car on peut définir **des transformations à partir d'une mesure uniforme** sur $[0, 1]$ afin d'obtenir la densité de probabilité cible. Il faut cependant, définir la capacité d'échantillonner une mesure uniforme sur $[0, 1]$ ce qui nous fait apprécier la conception d'un **système dynamique qui crée de l'aléatoire à partir du déterministe**, et qui satisfait la propriété d'ergodicité. A côté (voire aux côtés) de la voie du "système dynamique", nous verrons **l'algorithme du rejet** (ou de "rejection sampling", ou "hit & miss", etc). En fait, il arrive que la voie du "système dynamique" donne une distribution $q(x)$ différente de celle que l'on souhaite, là l'algorithme du rejet permet la transformation de $q(x)$ vers $p(x)$. C'est une idée simple mais fondamentale que l'on retrouve aussi bien en basse qu'en grande dimension. Enfin, nous aborderons **l'échantillonnage d'importance**¹¹² (ou "importance sampling") qui fait parti des algorithmes de base d'échantillonnage lequel permet également à partir d'une distribution $q(x)$ de restituer la distribution $p(x)$ cherchée.

Maintenant, passer à **la grande dimension** nous fait redouter d'être confronté à

110. NDJE. Dans le cas de l'existence de modes multiples, la géométrie de cet espace est plus complexe.

111. NDJE. voir Cours 2023 Sec. 6.2

112. NDJE. voir la note en bas de page 23.

la malédiction: appliquer brutalement les techniques ci-dessus ne fonctionne pas. Il faut trouver le moyen d'une façon où d'une autre de faire en sorte que $q(x)$ soit très bien adaptée. Le système dynamique adéquate est celui **des chaînes de Markov**, et l'algorithme de **Metropolis-Hastings** est un algorithme de rejet appliqué sur celles-ci. Ces deux ingrédients forment la base des algorithmes de **Monte Carlo Markov Chain** (MCMC).

Pour finir, nous aborderons les **algorithmes de Score Diffusion/Denoising** pour voir comment cela s'inscrit dans le schéma précédent. L'idée est que l'on va de nouveau définir des chaînes de Markov. Mais on s'intéresse plutôt à $-\nabla_x U(x) = \nabla_x \log p(x)$ qui n'est autre que le score (Sec. 5.4). Nous verrons alors le lien avec le débruitage (*denoising*). Les réseaux de neurones interviennent pour approximer $\nabla_x \log p(x)$ via une fonction $s_\theta(x)$. Si tout est sous contrôle, alors on est capable à partir d'un bruit blanc, de générer des échantillons par chaîne de Markov selon la bonne probabilité $p(x)$.

7.2 Le cas à 1 dimension

Supposons que l'on dispose de la v.a U de densité uniforme sur $[0, 1]$ (notée $\mathcal{U}(0, 1)$). On veut pouvoir échantillonner $p(x)$ avec $x \in \mathbb{R}$. Soit la **fonction de partition** $F(x)$ telle que

$$F(x) = \int_{-\infty}^x p(u)du \quad (171)$$

A partir de là, on utilise la méthode suivante:

Lemme 2 (méthode de la Transformation Inverse)

Soit F définie $\mathbb{R} \longleftrightarrow [0, 1]$, strictement monotone, et soit $U \sim \mathcal{U}(0, 1)$, alors

$$X = F^{-1}(U) \quad (172)$$

est une v.a dont la fonction de répartition est $F(x)$ et donc de densité $p(x) = F'(x)$.

Démonstration 2. Envisageons la fonction de répartition de X :

$$\mathbb{P}(X = F^{-1}(U) \leq x) = \mathbb{P}(U \leq F(x)) = F(x) \quad (173)$$

en effet $F(x) \in [0, 1]$ et pour tout $a \in [0, 1]$, U étant de densité uniforme $\mathbb{P}(U \leq a) = \int_0^a 1 dx = a$. Donc, nous avons le résultat du lemme. Le côté "strict" de la monotonie peut être relaxé, il faut alors définir $F^{-1}(x)$ telle que

$$F^{-1}(x) = \inf\{x \text{ tq. } F(x) \leq u\} \quad (174)$$

■

Notez que $p(x)$ peut être complexe mais il suffit alors d'en obtenir des approximations (ex. splines) pour rendre l'inversion de $F(x)$ facile. Le point réellement difficile en l'espèce est d'échantillonner $\mathcal{U}(0, 1)$. Le théorème central est celui de **Birkhoff**¹¹³ et la notion d'**ergodicité**.

7.3 Ergodicité d'une transformation déterministe

Définition 6 (Transformation ergodique)

Soit un ensemble probabilisé (χ, \mathcal{B}, μ) constitué d'un ensemble d'états χ , d'une mesure μ et d'un ensemble des ensembles mesurables relativement à μ (noté \mathcal{B}) qui est une σ -algèbre^a. On suppose que μ est une mesure de probabilité sur χ , notamment $\mu(\chi) = 1$. Une transformation T est ergodique si:

- $\forall A \in \mathcal{B}, \quad \mu(T^{-1}(A)) = \mu(A) \quad (\text{mesure invariante}) \quad (175)$

- $A \in \mathcal{B}, \text{ tq. } T^{-1}(A) = A \Rightarrow \begin{cases} \text{soit } \mu(A) = 0 \\ \text{soit } \mu(A) = 1 \end{cases} \quad (176)$

a. NDJE. ex. les ouverts de \mathbb{R}^d

NDJE. Pour comprendre la première propriété, considérons par exemple $A \in \mathcal{B}$, sa mesure par rapport à μ s'écrit

$$\mu(A) = \int_{\chi} \mathbf{1}_A(x) d\mu(x)$$

113. NDJE. Voir Th 11 du Cours 2023 Sec. 6.2

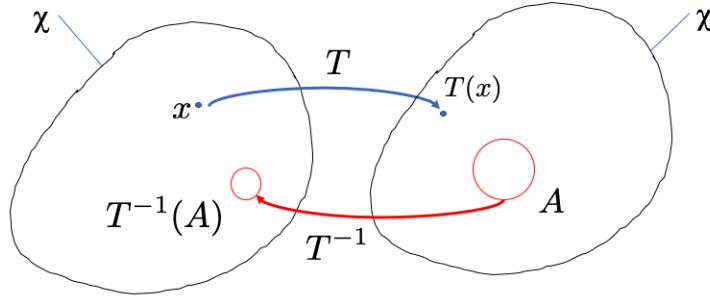


FIGURE 31 – Schéma qui montre pourquoi $T^{-1}(A)$ apparaît dans la définition de la mesure invariante.

Soit $T(x)$ le transformé de x par l'application T , on peut se demander ce que représente:

$$\int_{\chi} \mathbf{1}_A(T(x)) d\mu(x)$$

C'est la mesure de l'ensemble B contenant tous les transformés des éléments de χ qui se retrouve dans A (Fig. 31), donc $B = T^{-1}(A)$ et l'intégrale ci-dessus peut s'écrire

$$\int_{\chi} \mathbf{1}_A(T(x)) d\mu(x) = \int_{\chi} \mathbf{1}_B(x) d\mu(x) = \mu(B) = \mu(T^{-1}(A))$$

La notion de mesure invariante par μ , s'identifie alors comme la propriété énoncée à savoir $\mu(A) = \mu(T^{-1}(A))$ pour tout A . En théorie de la mesure on définit la notion de tiré en avant de la mesure par T , noté $T_*\mu$, et on note alors

$$\int_{\chi} \mathbf{1}_A(T(x)) d\mu(x) = \int_{\chi} \mathbf{1}_A dT_*\mu$$

L'invariance s'écrit alors $T_*\mu = \mu$.

La seconde propriété nous indique qu'il n'existe pas d'ensemble A différent de l'ensemble tout entier χ ou réduit à des points isolés, tel que prenant un élément x et procédant aux itérés $x_n = T^n(x)$, on resterait coincer dans cet ensemble.

Le théorème central est celui **d'ergodicité de Birkhoff** qui s'énonce ainsi

Théorème 11 (Birkhoff)

Si T est ergodique alors pour toute fonction mesurable $f \in L^1(\chi, \mu)$, $\int_{\chi} |f(x)|d\mu(x) < \infty$, considérant presque tout $x_0 \in \chi$

$$\frac{1}{n} \sum_{k=1}^n f(T^k(x_0)) \xrightarrow{n \rightarrow \infty} \int_{\chi} f(x) d\mu(x) = \mathbb{E}_{\mu}[f(X)] \quad (177)$$

L'hypothèse d'ergodicité nous dit que la moyenne empirique calculée sur les points transformés $(x_k = T^k(x_0))_{k \leq n}$ converge vers l'espérance relative à la mesure μ de la v.a X sous tendue par la génération des $(x_k)_k$. C'est-à-dire que les $(x_k)_k$ vont parcourir tout l'espace et la concentration de ces points dans χ est le reflet de la densité de probabilité $d\mu(x)$.

Un cas particulier important est celui où $f = \mathbf{1}_A$, le théorème nous indique (rappel $\mu(\chi) = 1$)

$$\frac{1}{n} \sum_{k=1}^n \mathbf{1}_A(T^k(x_0)) \xrightarrow{n \rightarrow \infty} \mathbb{E}_{\mu}[\mathbf{1}_A] = \mu(A) \quad (178)$$

c'est-à-dire que **le comptage des retours dans l'ensemble A** en partant de x_0 donne la mesure de A .

Le programme est alors le suivant: étant donnée une mesure (ex. la densité de probabilité $p(x)$), trouver une transformation T qui laisse cette mesure invariante, et telle que les itérés parcourent tout l'espace, c'est-à-dire que ses seuls ensembles invariants sont soit l'ensemble χ soit des points isolés. Comme on l'a dit en introduction, s'il n'est pas possible d'obtenir p comme mesure invariante d'une transformation, il faudra trouver une mesure invariante q proche de p , que l'on transformera par des algorithmes de type "rejet".

7.4 Mesure uniforme sur $[0, 1]$

Pour commencer, il nous obtenir des échantillons d'une loi uniforme sur $[0, 1]$ notée $\mathcal{U}(0, 1)$. Pour cela, on peut tenter de trouver une transformation T qui transforme un point du cercle unité vers un autre point de ce cercle. On aimerait alors que la densité des points soit uniforme sur le cercle.

Soit l'expression suivante (Fig. 32):

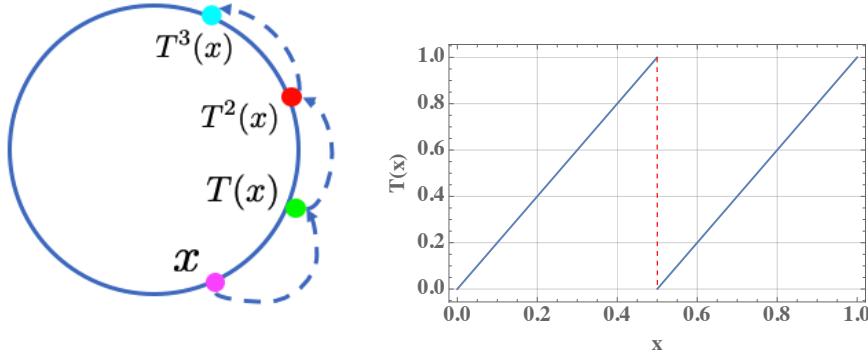


FIGURE 32 – Itérés successifs par transformation T d'un point x sur le cercle. A droite la transformation dyadique Eq. 179.

$$T(x) = 2x \bmod 1 \quad (179)$$

qui est un décalage en notation binaire (*bit shift map*, ou Bernoulli ou *dyadic map*). Si on part d'un x irrationnel, en binaire la liste de ses bits n'est pas périodique, ainsi le décalage produit un nombre différent à chaque itération, et donc il n'y a pas de périodicité. Il faut cependant démontrer que la mesure invariante est la mesure uniforme¹¹⁴. Notons que pour deux nombres irrationnels proches, les itérés au bout d'un certain rang, vont diverger. Cela tient au fait que dans la décomposition des deux nombres, les N premiers bits sont

114. NDJE. hints: Si (b_0, b_1, b_2, \dots) est une suite infini de bits $(0, 1)$, alors la transformation T donne la suite (b_1, b_2, \dots) , et tout nombre réel sur $[0, 1]$ peut s'écrire comme

$$x = \sum_{k=0}^{\infty} \frac{b_k}{2^{k+1}}$$

Un nombre irrationnel est caractérisé par une suite de bits non périodique. Ce qui manifeste le fait que T parcourt tout le cercle en partant d'un nombre irrationnel. Maintenant, si on prend une densité de points de départ $\rho(x)$, une itération transforme cette densité. Pour trouver l'expression de la nouvelle densité, on cherche les points $y = T^{-1}(x)$ (argument déjà abordé plus haut). Pour obtenir la densité, on utilise alors la formule de la dérivée sachant qu'il faut prendre en compte les 2 intervalles $[0, 1/2]$ et $[1/2, 1]$ mis en jeu par le modulo. On trouve alors que la transformation de la densité s'écrit

$$\rho(x) \xrightarrow{T} \frac{1}{2}\rho(x/2) + \frac{1}{2}\rho((x-1)/2)$$

On note alors que $\rho(x) = 1$ (loi uniforme) est invariante par cette transformation. Maintenant, si l'on ne dispose que d'une représentation finie des nombres irrationnels, à terme la série des itérés finie par tomber à partir d'un certain rang sur le point fixe de la transformation, à savoir $x = 0$.

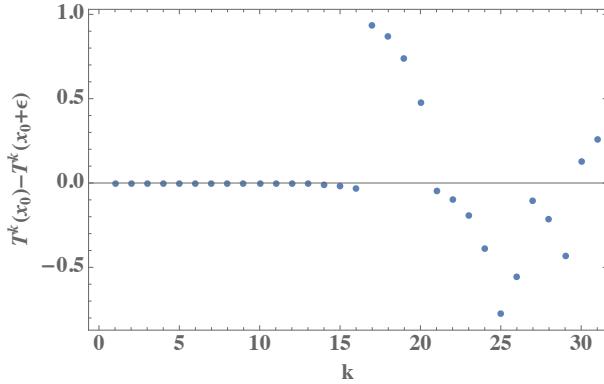


FIGURE 33 – Itérés successifs par transformation T dyadique de $x_0 = 1/\sqrt{2}$ et de $x_1 = x_0 + 10^{-6}$.

identiques (voir note de bas de page 114), mais ensuite ils diffèrent. Or, à la N -ième itération seuls comptent ces bits qui sont différents ce qui entraîne la divergence. Voir un exemple sur la figure 33. **C'est une forme de chaos qui s'instaure au bout d'un certain rang d'itération de la transformation dyadique.** Notons que l'on peut concevoir une transformation ergodique ayant la loi uniforme comme invariant en utilisant l'expression suivante pour T

$$T(x) = x - \tau \mod 1 \quad \tau \notin \mathbb{Q} \quad (180)$$

Elle satisfait tous les critères d'ergodicité et la mesure uniforme est invariante. Mais, cette transformation n'est pas chaotique ce qui aura un inconvénient comme nous le verrons plus loin.

En fait, dans la plupart des cas, on veut générer des échantillons $(x_i)_i$ selon p qui sont **indépendants**. Notons que pour le théorème de Birkhoff, ce n'est pas nécessaire. Si à partir d'un *v.a* notée X_n , on définit la *v.a* X_{n+1} selon

$$X_{n+1} = T(X_n) \quad (181)$$

il est clair que X_{n+1} dépend de X_n (nb. T est déterministe). Mais au bout de k itérés

$$X_{n+k} = T^k(X_n) \quad (182)$$

peut-il y avoir un mécanisme menant à la quasi-indépendance de X_{n+k} vis-à-vis de X_n ?

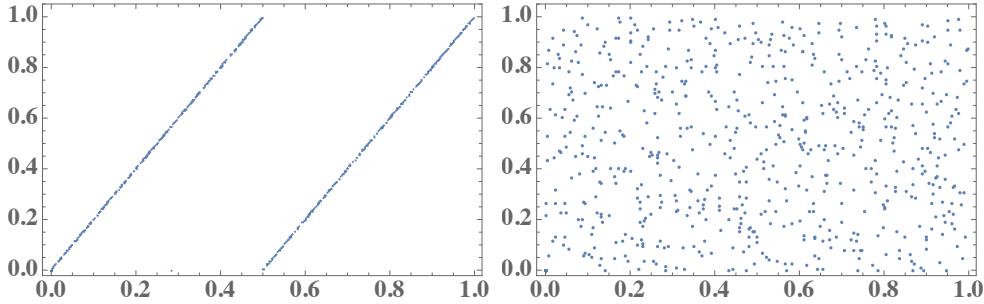


FIGURE 34 – Scatter plot de (X_n, X_{n+k}) pour $k = 1$ (gauche) et $k = 10$ (droite) et $X_0 = 1/\sqrt{2}$ et $n \leq 1000$. On voit que quand k devient suffisamment grand on approche une du cas où (X_n, X_{n+k}) sont indépendants.

C'est là où le chaos permet cette "dé-corrélation". Une illustration de ce phénomène de quasi-décorrélation est montrée sur la figure 34 pour la transformation dyadique. Pour vérifier l'indépendance de X_n et X_{n+k} , on réalise des tests statistiques comme par exemple le découpage de l'intervalle $[0, 1]$ en sous intervalles de longueur identique et compter le nombre de x_n dans chaque sous intervalle. En principe, si tout est bien, on doit obtenir un histogramme plat (aux fluctuations statistiques de Poisson près). Mais il y a des tests plus sophistiqués¹¹⁵.

Les deux transformations citées ne sont pas très bonnes en réalité¹¹⁶ , en voici une

115. NDJE. On peut trouver une discussion à la fois de quelques algorithmes et tests ici <https://www.omsccs-notes.com/simulation/generating-uniform-random-numbers/>. Une analyse fouillée sur des aspects technologiques est faite ici https://theses.hal.science/tel-00759976v1/file/These_MathildeSoucarros.pdf.

116. NDJE. Dans les années 90, les grandes expériences du LEP au CERN, on eu besoin d'augmenter significativement le nombre de simulations de leurs détecteurs (DELPHI, ALICE, OPAL, L3). De plus, pour profiter de plusieurs centres de calcul en dehors du CERN (ex. le CCIN2P3 en France), il fallait trouver des générateurs (PRGN) suffisamment résistants aux tests statistiques pour rendre ces simulations indépendantes. C'est dans ce contexte que Martin Luescher a proposé RANLUX en 1993 (voir <https://arxiv.org/pdf/hep-lat/9309020.pdf>) dont la période est de l'ordre de 10^{171} , ce qui permettait de distribuer des graines aux différents centres et de faire les générations indépendantes désirées. Sans cela ni les expériences LEP, ni plus tard celles du LHC n'auraient pu envisager d'étudier les événements rares nécessitant des simulations les plus précises possibles. Qui plus est RANLUX est solide théoriquement. Maintenant, C++ ou Python/Numpy utilisent MT19937 qui est un PRGN de type Mersenne Twister développé par Makoto Matsumoto en 1997 qui a des atouts également, sa période est colossale $2^{19937}-1 \approx 4 \cdot 10^{6001}$, mais il a des défauts en particulier concernant la distribution de graines pour des tirages Monte Carlo indépendants. Ceci étant la cryptographie a été la pourvoyeuse de nouveaux algorithmes qui sont au coeur par exemple de la génération de nombres aléatoires de libraries comme JAX. Voir par exemple: John K. Salmon, Mark A. Moraes, Ron O. Dror, and David E. Shaw. (2011). *Parallel random*

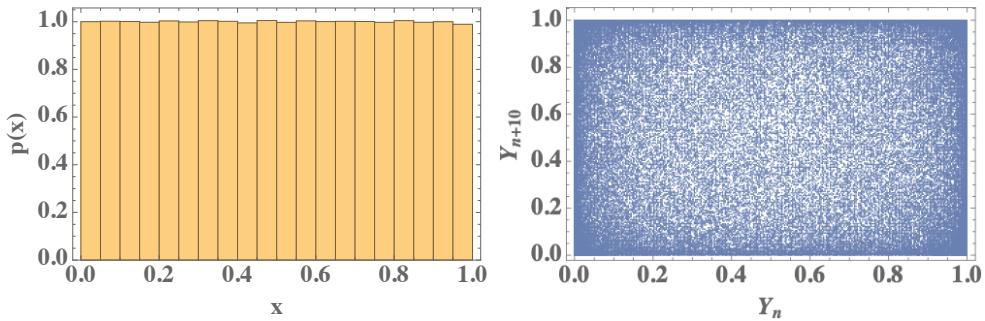


FIGURE 35 – A gauche: Histogramme des Y_n générés par la transformation de la suite logistique (Eq. 183) par la fonction de répartition de sa mesure invariante. On a utilisé $X_0 = 0.1$ et généré 10^6 échantillons Y_n . A droite: le biplot (Y_n, Y_{n+10}) pour 10^5 échantillons. La répartition est quasi-uniforme, on voit l'effet des points fixent $x = 0$ et $x = 1$ de $T(x)$.

meilleure¹¹⁷ qui ne repose pas sur la manipulation de bits:

$$T(x) = 4x(x - 1) \mod 1 \quad (183)$$

on l'appelle la *logistic map*. Les itérés de cette transformation donne lieu à un chaos par dédoublement de fréquence dont la mesure invariante est $q(x) = \frac{1}{\pi\sqrt{x(x-1)}}$. Ce n'est pas la loi uniforme mais on peut utiliser sa fonction de répartition $F(x)$ à cette fin (on renverse le lemme 2). On obtient alors, $F(x) = 2 \arcsin(\sqrt{x})/\pi$. Ainsi, on peut définir la suite $X_{n+1} = T(X_n)$ puis $Y_{n+1} = F(X_n)$ pour obtenir une suite de nombre uniformément répartis dans $[0, 1]$ comme le montre la figure 35.

Maintenant, le problème qui va nous occuper, c'est le passage à la grande dimension. Mais dès $d = 2$ on a un problème basique, car la fonction de répartition F est définie $\mathbb{R}^2 \rightarrow \mathbb{R}$ donc ne peut être inversible. Il nous faut donc envisager d'autres techniques.

numbers: as easy as 1, 2, 3. In Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC '11). Association for Computing Machinery, New York, NY, USA, Article 16, 1–12. <https://doi.org/10.1145/2063384.2063405>, accessible ici <http://www.thesalmons.org/john/random123/papers/random123sc11.pdf>. Donc, le message est: avant de ce lancer dans des programmes qui demandent énormément de tirages aléatoires, mieux vaut regarder quel est le PRGN de base utilisé.

117. NDJE. Je corrige les typo faites au tableau.

7.5 Techniques de "Rejet"

7.5.1 Version simple

Il s'agit de méthodes génériques où l'idée centrale est de mettre $p(x)$ à l'intérieur d'une boîte. C'est-à-dire à la variable $x \in \text{support}(p(x)) \subset \mathbb{R}^d$, on ajoute une variable $u \in [0, \max_x p(x)] \subset \mathbb{R}^+$. Les bornes sur x et u forment les dimensions de la boîte \mathcal{D} . Soit alors l'ensemble \mathcal{A} définit selon

$$\mathcal{A} = \{(x, u) / 0 \leq u \leq p(x)\} \subset \mathcal{D} \quad (184)$$

Soit, le couple de *v.a* $Z = (X, U)$ *iid* selon la mesure uniforme sur la boîte \mathcal{D} . si on note p_Z la densité de probabilité de Z , alors la loi marginale selon X est égale à

$$p_X(x) = \int_{\mathcal{A}} p_Z(x, u) du = \int_0^{p(x)} 1 du = p(x) \quad (185)$$

On retrouve alors que x suit la loi $p(x)$. Donc l'algorithme s'écrit:

Algorithm 1 Rejection sampling (1)

- 1: Shoot uniformly $(x, u) \in \mathcal{D}$
 - 2: **if** $u > p(x)$ **then** reject
 - 3: **else** accept
-

Les échantillons (x, u) sont rejetés s'ils ne sont pas dans \mathcal{A} autrement dit s'ils sont éléments de \mathcal{A}^C . Ils sont acceptés dans le cas contraire. Voir une illustration où $d = 1$ sur la figure 36.

Ce type d'algorithme est extensible au cas où la "boîte" \mathcal{D} est remplacée par un ensemble qui épouse au mieux la forme de la distribution $p(x)$.

7.5.2 Version à partir de $q(x)$

Soit une distribution de probabilité $q(x)$ que l'on sait échantillonnée, et on trouve M tel que

$$p(x) \leq M q(x) \quad (186)$$

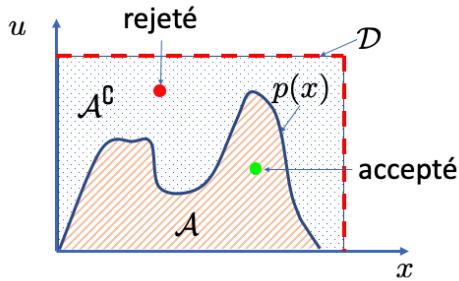


FIGURE 36 – Illustration de la méthode du rejeté-accepté (rejection sampling) avec $p(x)$ en 1D.

L'algorithme de rejet est alors le suivant (voir Fig. 37 pour une version en 1D):

Algorithm 2 Rejection sampling (2)

```

1: for  $i : 1, \dots, n$  do
2:   ok = False
3:   while not ok do
4:     Shoot  $x_i \sim q(x)$  and  $u \sim \mathcal{U}(0, 1)$ 
5:     if  $u \leq p(x_i)/(Mq(x_i))$  then ok = True
6:   keep  $x_i$ 
7: return  $(x_i)_{i \leq n}$ 
```

Le principe général est donc l'introduction d'une variable auxiliaire u indépendante des tirages des x_i qui réduit la probabilité dans le rapport souhaité pour obtenir un échantillon de $p(x)$ ¹¹⁸.

Maintenant, le problème de cet algorithme est que M ne doit pas être trop grand¹¹⁹. Cela se voit sur l'efficacité η de cet algorithme défini en utilisant le rapport du nombre

118. NDJE. la marginale s'écrit dans ce cas $f_X(x) = \int_0^1 \Theta(p(x)/(Mq(x)) - u) Mq(x)du = p(x)$ avec $\Theta(x)$ la fonction d'Heaviside.

119. NDJE. Voir le cas de figure étudié dans le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/cours2023/Monte_Carlo_Sampling.ipynb d'une distribution $p(x)$ avec 2 modes dont l'un est bien concentré.

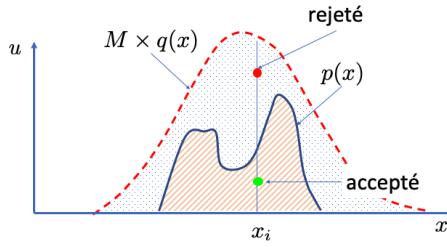


FIGURE 37 – Illustration de la méthode du rejeté-accepté (rejection sampling) à partir d'une distribution $q(x)$ que l'on sait échantillonnée pour obtenir avec $p(x)$.

d'échantillons x_i acceptés sur le nombre total d'essais (n)

$$\eta = 1 - \frac{\#\text{reject}}{\#\text{total}} = \frac{\#\text{accept}}{\#\text{total}} \quad (187)$$

7.5.3 Probabilité d'acceptance

Il nous faut calculer¹²⁰

$$\mathbb{E}_{x \sim q} \left[\mathbb{P} \left(u \leq \frac{p(x)}{Mq(x)} | x \right) \right] = \mathbb{E}_{x \sim q} \left[\frac{p(x)}{Mq(x)} \right] = \int \frac{p(x)}{Mq(x)} q(x) dx = 1/M \quad (188)$$

Donc, si M doit être grand pour s'adapter à la valeur de $\max_x p(x)$ alors l'algorithme perd en efficacité. Or, en grande dimension nous avons des phénomènes de concentration de probabilité (Sec. 3.2.4) qui entraîne mécaniquement de grandes valeurs de $p(x)$ donc de M . **Typiquement, on s'attend**¹²¹ à ce que $M \propto e^d$. On voit par exemple par cet argument que l'on ne peut en pratique générer des "visages" à partir d'une distribution gaussienne. En fait, il faut pouvoir obtenir un $q(x)$ qui soit aussi près que possible de $p(x)$. Avant de voir comment procéder, voyons un autre type d'échantillonnage.

120. NDJE. nb. pour $\alpha \in [0, 1]$, $\mathbb{P}(u \leq \alpha) = \alpha$ pour une loi uniforme sur $[0, 1]$.

121. NDJE. Un exemple est donné dans le notebook mentionné dans la note de bas de page 39 dans le cas de l'évaluation de la variance de la mesure uniforme sur la boule de rayon unité en dimension d . On revisite la figure 9.

7.6 Échantillonnage d'importance

L'idée est de pouvoir adapter $q(x)$ pour calculer une intégrale par méthode de Monte Carlo. Mettons que l'on sache non seulement calculer $p(x)$ et $q(x)$, mais on sait également échantillonner selon $q(x)$ (nb. on suppose que $q(x)$ ne s'annule pas sur le support de $p(x)$). Soit alors l'intégrale suivante

$$I(f) = \int f(x) p(x) dx = \int f(x) \underbrace{\frac{p(x)}{q(x)}}_{w(x) \geq 0} q(x) dx = \mathbb{E}_q[f(x)w(x)] \quad (189)$$

Soit alors des échantillons indépendants $(x_i)_{i \leq n}$ tirés selon $q(x)$ alors

$$I_n(f) = \frac{1}{n} \sum_{i=1}^n f(x_i) w(x_i) \xrightarrow{n \rightarrow \infty} I(f) \quad (190)$$

$I_n(f)$ est un estimateur non biaisé grâce à l'indépendance des échantillons $(x_i)_i$, en effet on a facilement que

$$\mathbb{E}_q[I_n(f)] = \frac{1}{n} \times n \times \mathbb{E}_q[f(x)w(x)] = I(f) \quad (191)$$

La question est alors de savoir quelle est la vitesse de convergence? On sait que par l'indépendance des v.a $f(x_i)w(x_i)$ on a

$$Var(I_n(f)) = \frac{1}{n} Var(f(x)w(x)) \quad (192)$$

On aimerait que le terme $Var(f(x)w(x))$ ne soit pas trop grand. Tentons de le borner:

$$Var(f(x)w(x)) = \mathbb{E}_q[|f(x)w(x) - I(f)|^2] = \mathbb{E}_q[|f(x)w(x)|^2] - I(f)^2 \leq \mathbb{E}_q[|f(x)w(x)|]^2 - I(f)^2 \quad (193)$$

l'égalité étant acquise *ssi* $|f(x)|w(x) = c$. On aimerait donc que $q(x)$ satisfasse

$$|f(x)| \frac{p(x)}{q(x)} = c \Rightarrow q(x)c = |f(x)|p(x) \Rightarrow c = \int |f(x)|p(x) dx \quad (194)$$

d'où le "meilleur" $q(x)$ a pour expression

$$q(x) = \frac{|f(x)|p(x)}{\int |f(x)|p(x)dx} \quad (195)$$

Ce que cela nous dit, c'est que pour calculer $I(f)$, on peut faire mieux que d'échantillonner selon $p(x)$. Mais remarquons que le calcul initial implique l'intégrale de $f(x)p(x)$ et que la constante de normalisation de $q(x)$ en est très proche. Donc, on a certes l'expression du meilleurs $q(x)$ mais en pratique on a un problème. On procèdera alors à l'usage d'un $q(x)$ non optimal.

Retenons que dans ce type de méthode, on ne fait pas de rejet des échantillons tirés de $q(x)$, on calcule directement la moyenne empirique Eq. 190. En fait, on calcule une moyenne avec **une mesure empirique pondérée**:

$$\tilde{p}_n = \frac{1}{n} \sum_{i=1}^n \delta_{x_i} w(x_i) \quad (196)$$

Cette méthode peut être envisagée dans le cas où $p(x) = Z_p^{-1}\tilde{p}(x)$ et $q(x) = Z_q^{-1}\tilde{q}(x)$ et où Z_p et Z_q sont les constantes de normalisation très difficiles à estimer (ex. énergie de Gibbs). Car, alors il suffit de considérer $w(x) = \tilde{p}(x)/\tilde{q}(x)$ et

$$I_n(f) = \sum_i \frac{f(x_i)w(x_i)}{\sum_i w(x_i)} \quad (197)$$

(nb. on peut appliquer la même méthode si bien entendu on sait calculer $q(x)$ directement).

La limitation de ce genre de méthode est du même type que celle rencontrée pour la méthode du rejet, à savoir pour avoir un algorithme de bonne efficacité, il faut partir d'un $q(x)$ quasi-optimal. Et cela devient de plus en plus critique en grande dimension à cause de la concentration de la $p(x)$ sur ses ensembles typiques. **Il faut faire en sorte que $q(x) \propto |f(x)|p(x)$ avec une constante de proportionnalité qui n'explose pas.**

Pour ce faire, **il faut pouvoir adapter** $q(x)$ et on ne peut le faire en piochant dans des fonctions de typologie connue *a priori*, ce qui revient *grosso modo* à prendre une gaussienne multivariée.

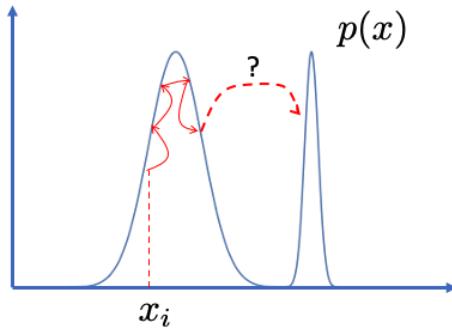


FIGURE 38 – Illustration de la méthode d'exploration autour d'un échantillon x_i à forte probabilité. Attention néanmoins, il faut pouvoir opérer un grand step pour pouvoir explorer l'autre (ou les autres) mode(s) de $p(x)$.

7.7 Au-delà du "Rejet": la progression des idées

Si on inspecte la méthode du rejet (Algo. 2), à chaque itération pour obtenir un nouveau échantillon x_i , on ne se sert pas de la connaissance que l'on a eu dans les tirages antérieurs. En particulier, pour $k < i$ on a obtenu des échantillons représentatifs de $p(x)$. Donc, **au fur et à mesure, on devrait pouvoir tenir compte de l'apprentissage de la distribution cible.**

Notons qu'à cause de la concentration de $p(x)$ (Fig. 12), la plus part des x ont une probabilité quasi-nulle. Mais mettons, que l'on ait obtenu un x_i avec $p(x_i)$ notablement non-nul, on a potentiellement un bon candidat, et on aimerait alors voir ce que donne des échantillons autour de ce dernier. Il est clair que l'algorithme de rejet perd cette mémoire. Au contraire, à **partir de x_i on va tenter un petit step** $x_i \rightarrow x_i + \delta$. **Le problème** que l'on sent poindre immédiatement, c'est que dans le cas où $p(x)$ **a plusieurs modes**, on peut se retrouver à n'explorer que le mode dans lequel x_i se trouve, si jamais δ est trop petit par rapport à la séparation entre modes. Donc, **il faut avoir une probabilité non nulle de faire un grand step** (Fig. 38).

Donc, il faut pouvoir établir une carte $T(x)$ déterministe qui donne la possibilité de rendre compte de la façon de "bouger" et dont la mesure invariante est $p(x)$, sachant que l'on peut partir d'un *a priori* à savoir une estimation grossière de $p(x)$. C'est l'idée

des **chaînes de Markov**: bouger localement ou avec un grand saut est formalisé par la probabilité de transition de x_n à x_{n+1} , c'est-à-dire que l'**on modélise les probabilités conditionnelles** $p(x_{n+1}|x_n)$. La modèle au départ est sans doute "mauvais" et donc **il va falloir faire évoluer ces probabilités conditionnelles**. Comment? par l'**algorithme de rejet**. C'est l'idée de l'algorithme de **Metropolis-Hastings**¹²² qui utilise l'algorithme de rejet sur les probabilités de transitions.

Maintenant, si $p(x)$ est une probabilité de Gibbs, là où $p(x)$ est maximum $U(x)$ est minimum et vice-versa. Donc, ayant un échantillon x_n , il est clair que se diriger vers le minimum de U est guidé par une **descente de gradient!** Donc, la probabilité de transition aurait la bonne idée de tenir compte de $-\nabla_x U(x)$. Mais comme en ML, on sait qu'**ajouter du bruit** aide. Mais ici ce n'est pas tant de s'affranchir des points-selles (points-cols, ou *saddle points*) qui nous motive, mais plutôt la nécessité d'explorer tous les modes éventuels de $p(x)$ afin de pouvoir par la suite produire de nouveaux échantillons¹²³. C'est l'idée de l'**algorithme de Langevin**. Ceci dit si l'amplitude du bruit est trop grande, on sait que l'on ne peut converger vers les minima, et s'il est trop petit on se trouve confiné, donc **il faut faire un ajustement**.

Mais, effectuer ces ajustements a ces limites, pourrait-on alors **remplacer le $U(x)$ multi-modale par un $U(x)$ mono-mode** qui n'a donc plus de problème vis-à-vis de la descente de gradient. Ça paraît miraculeux. Et pourtant, c'est l'idée derrière les algorithmes de **Score Denoising** (débruitage). On va lisser petit-à-petit $U(x)$ pour en faire une fonction convexe que l'on sait échantillonner facilement. On a donc une série de $(U_i)_{i \leq n}$ avec $U_1 = U$ multi-modale et U_n mono-mode qui sont proches les unes des autres entre deux étapes (paramètre d'homotopie sous-jacent). Puis, à rebours, on échantillonne selon

122. Nicholas Metropolis (1915-99), Wilfred Keith Hastings (1930-2016), et il y a d'autres auteurs associés à cet algorithme comme Arianna W. Rosenbluth (1927-2020), Marshall Rosenbluth (1927-2003), Augusta H. Teller (1909-2000) and Edward Teller (1908-2003) tous associés au projet Manhattan (Sec. 3.2).

123. NDJE. Il y a des problématiques différentes de la descente de gradient selon son usage. Ça a été pendant longtemps un problème de compréhension des résultats des réseaux de neurones: quand on essaye d'envisager de faire de la descente de gradient dans un univers à grande dimension, on se dit qu'il y a sans doute beaucoup de minima locaux de la fonction de coût à minimiser. En quelque sorte, on a aucune garantie qu'un algorithme de type SGD/Adam trouve le minimum global. Or, force est de constater que des réseaux entraînés différemment notamment concernant les graines d'initialisation des poids, ont les mêmes propriétés concernant les statistiques réalisées sur des échantillons de test/validation. On a une sorte d'universalité des minima accessibles par ces algorithmes qui opèrent une régularisation de la loss. Donc, peu importe d'explorer tout l'espace. La problématique est totalement différente concernant l'obtention d'échantillons de $p(x)$ où tous les minima de $U(x)$ sont à explorer.

$p_n = Z_n^{-1} e^{-U_n}$ et on utilise l'algorithme du rejet pour obtenir des échantillons de p_{n-1} et ainsi de suite. Le processus de création des U_i se fait par convolution par des gaussiennes. Or, rendre U_n comme une forme quadratique de type $x^T K x$ c'est avoir pour p_n une distribution gaussienne, c'est-à-dire du bruit! Les échantillons de p_n sont donc un bruit. Et donc le processus est d'abord d'ajouter du bruit progressivement par exemple dans une base de donnée d'images, et ensuite de procéder à partir d'une réalisation de pur bruit, à un débruitage pour obtenir une nouvelle image. **Tout cela se fait par une équation différentielle stochastique que l'on sait inverser.**

7.8 Chaîne de Markov

Définition 7 (*Chaine de Markov*)

Un processus (X_1, \dots, X_n) est une chaîne de Markov si $\forall (x_i)_{i \leq n+1} \in \chi^{n+1}$

$$\mathbb{P}(X_{n+1} = x_{n+1} | X_n = x_n, \dots, X_1 = x_1) = \mathbb{P}(X_{n+1} = x_{n+1} | X_n = x_n) \quad (198)$$

Donc, pour produire X_{n+1} (futur) on a besoin uniquement de la connaissance de X_n (présent), ou en terme plus radical "le passé ne compte pas". Soit alors la propriété/définition suivante:

Définition 8 (*stationnaire/homogène*)

Une chaîne de Markov est stationnaire (ou homogène) si $\forall (x, y) \in \chi^2$

$$\forall n, \quad \mathbb{P}(X_{n+1} = y | X_n = x) = \mathbb{P}(X_n = y | X_{n-1} = x) = P_{x,y} \quad (199)$$

donc les probabilités de transition ne dépendent pas de n , que l'on peut noter $P_{x,y}$ (attention à l'ordre, pensez "on passe de x à y ").

En quoi cela va nous servir? En particulier, on espère par là opérer une transition d'une distribution $p_0(x)$ vers la distribution $p(x)$ recherché, que l'on notera par la suite $\pi(x)$, en utilisant la matrice de transition pour transformer un/des échantillon(s) de $p_0(x)$ (x_0), progressivement en x, x_2, \dots c'est-à-dire des échantillons des distributions $p_1(x), p_2(x), \dots$

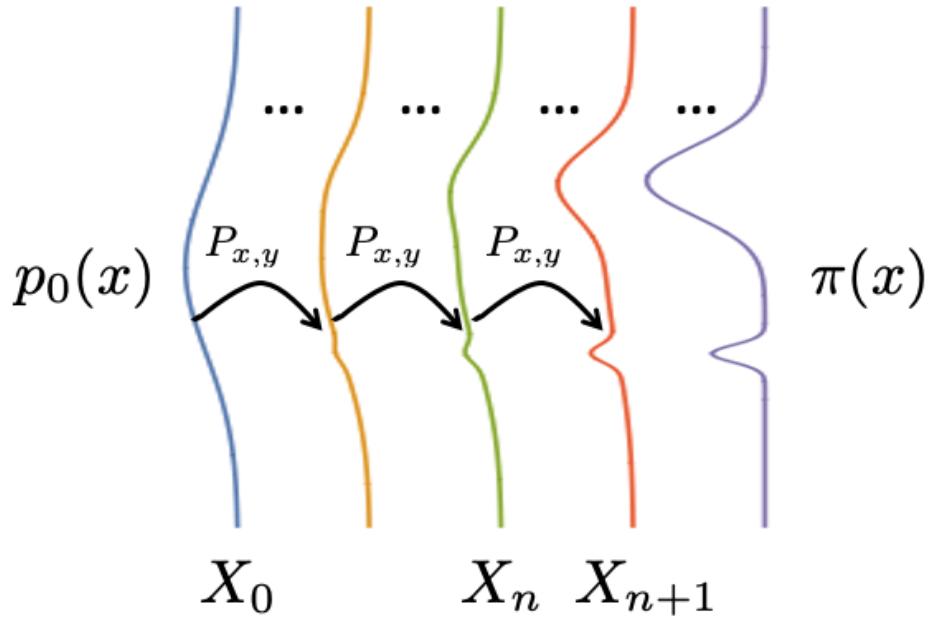


FIGURE 39 – Évolution de la distribution initiale $p_0(x)$ vers la distribution cible $\pi(x)$ par application de la matrice de transition $P_{x,y}$ de la chaîne de Markov.

des v.a X_1, X_2, \dots (Fig. 39). **Tout le problème est de déterminer la matrice $P_{x,y}$ adéquate pour réaliser ce programme.**

Nous avons les propriétés suivantes de la matrice $P_{x,y}$:

Propriété 5

Dans le cas d'un alphabet (ie. χ est discret) alors $(P_{x,y})_{(x,y) \in \chi^2}$ est une matrice stochastique de transition qui a les propriétés suivantes:

- *primo*

$$\boxed{\forall x \in \chi, \sum_{y \in \chi} P_{x,y} = 1} \quad (\text{matrice stochastique}) \quad (200)$$

qui vient du fait que $p(y|x) = p(x,y)/p(x)$ et $\sum_y p(x,y) = p(x)$.

- *secundo, en notant que^a*

$$\boxed{\forall y \in \chi, \mathbb{P}(X_{n+1} = y) = \sum_{x \in \chi} \mathbb{P}(X_n = x) P_{x,y}} \quad (201)$$

on peut utiliser une forme matricielle en regroupant sous forme d'un vecteur colonne μ_n l'ensemble des probabilités à l'étape n , ainsi

$$\mu_n := (\mathbb{P}(X_n = x))_{x \in \chi} \implies \mu_{n+1} = P^T \mu_n \quad (202)$$

a. on le voit facilement en exprimant que $\mathbb{P}(X_{n+1} = y)$ est une marginale de la probabilité jointe de (X_{n+1}, X_n) .

Donc, quand on itère on a alors notant $P = P_{x,y}$

$$\mu_{n+k} = (P^T)^k \mu_n \quad (203)$$

et pour savoir s'il y a convergence, on est naturellement invité à faire l'étude des **valeurs propres de cette matrice P** , ce qui va nous renseigner sur l'existence éventuelle d'une **mesure invariante** que l'on espère être $\pi(x)$.

8. Séance du 6 Mars

Concernant les chaînes de Markov abordées à la section précédente, on suppose réalisée l'étape de modélisation des données par exemple selon une distribution de Gibbs que l'on note dans ce cadre markovien $\pi(x)$. La problématique se concentre sur la génération (échantillonnage) de nouveaux échantillons. Donc, on ne dispose comme information initiale uniquement de $\pi(x)$ qui est en général une distribution en grande dimension. **On part d'une distribution $q(x)$, facile à échantillonner, et l'on construit une transformation qui transporte les échantillons de q vers des échantillons de π .**

Si on fait une petite incise sur les algorithmes de Score Diffusion, le problème est différent car on part un cran avant, en disposant de données $\{x_i\}_{i \leq n}$ on veut à la fois estimer la distribution $\pi(x)$ et réaliser un échantillonnage. Néanmoins, on a en sous-jacent des chaînes de Markov. On inverse en quelque sorte le chemin, on part de $\pi(x)$ et l'on

opère un bruitage progressif des échantillons $\{x_i\}_i$ (équation stochastique de Ornstein-Uhlenbeck) pour aboutir à des échantillons de $q(x)$ constituée d'un bruit blanc gaussien (c'est-à-dire des échantillons de $q = \mathcal{N}(0, \Sigma)$). L'ensemble des échantillons obtenus aux étapes intermédiaires forment des chaînes de Markov. Ensuite, on renverse le processus par un débruitage de nouvelles réalisations de pur bruit issues de $q(x)$, et l'on récupère au bout du processus de échantillons de $\pi(x)$. L'estimation via le *score* se fait à l'aide d'un réseau de neurones profond durant la dernière étape d'inversion du processus stochastique.

Donc en commun aux deux approches, nous avons le *transport de distributions de probabilités*, le *calcul de probabilités de transition* et la *notion d'inversion du temps*, c'est-à-dire la *réversibilité*. Une différence entre les deux méthodes concerne l'invariance de la probabilité de transition dans la première méthode et son évolution au contraire dans la méthode du Score diffusion.

8.1 Chaînes de Markov: exemple

Nous continuons la section 7.8 où nous l'avions laissée. Un rappel: $P_{x,y}^T$ est la matrice de transport du vecteur μ_n de l'ensemble des probabilités $(\mathbb{P}(X_n = x))_{x \in \chi}$ (voir Fig. 39). Les exemples de chaînes de Markov qui vont nous intéresser sont celles obtenues par addition de *v.a* indépendantes: ce sont **des marches aléatoires généralisées** où intervient la notion de **variables "cachées"**.

Théorème 12 (*Marches aléatoires*)

Soit $\{Z_i\}_{i \geq 1}$ des *v.a iid* et indépendantes des $(X_j)_{j \geq 1}$, l'état à l'étape n dépend d'une fonction récurrente telle que

$$X_{n+1} = f(X_n, Z_{n+1}) \quad (204)$$

C'est typique des équations différentielles stochastiques où Z_n est une sorte de bruit qui induit des transitions aléatoires. Alors (X_1, \dots, X_n) est une chaîne de Markov stationnaire.

NDJE. La démonstration a été faite dans le Cours de 2023 Sec. 6.4.2.2 Th. 13. Concernant des exemples déjà étudiés en 2023, le notebook <https://github.com/jecampagne/>

cours_mallat_cdf/blob/main/cours2023/randomwalk.ipynb décrit le processus $X_{n+1} = \rho X_n + Z_{n+1}$ en 1D où Z_{n+1} v.a de Rademacher. Le cas $\rho = 1$ donne la marche aléatoire brownienne. Le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/cours2023/urne_Ehrenfest.ipynb simule un gaz de N particules dans une boîte avec Z_{n+1} une v.a qui prend ses valeurs dans $\{-1, +1\}$ comme pour Rademacher, mais cette fois $\mathbb{P}(Z_{n+1} = -1 | X_n = x) = x/N$. On y découvre la notion d'irréversibilité.

Un exemple qui concerne la méthode de Score Diffusion est le **processus d'Ornstein-Uhlenbeck** (1930 environ) qui simule une particule soumise à des frictions. La variable X_n est typiquement la vitesse à un instant n discrétisée, et nous avons¹²⁴

$$\underbrace{X_{n+1} - X_n}_{\text{accélération}} = \underbrace{-\alpha X_n}_{\text{friction}} + \underbrace{Z_{n+1}}_{\text{bruit brownien}} \quad \alpha \in [0, 1] \quad (205)$$

La solution peut s'écrire

$$X_{n+1} = (1 - \alpha)^{n+1} X_0 + \sum_{i=1}^n (1 - \alpha)^i Z_{n+1-i} \quad (206)$$

Quand n tend vers l'infinie, le terme dépendant des conditions initiales X_0 tend vers 0, et si $Z_{n+1} \sim \mathcal{N}(0, \sigma^2)$ alors

$$X_n \sim \mathcal{N}(0, \sigma_n^2) \quad \sigma_n^2 = \frac{\sigma^2}{\alpha} (1 - (1 - \alpha)^n) \xrightarrow{n \rightarrow \infty} \frac{\sigma^2}{\alpha} \quad (207)$$

De ce petit exemple, on illustre deux points fondamentaux:

- si on se trouve dans de bonnes configurations, **on peut oublier les conditions initiales** (X_0) pour des n suffisamment grands
- et surtout **on converge vers une mesure fixe** (ici la loi normale définie par les paramètres σ et α).

Donc, de manière générale de la distribution $\pi(x)$ potentiellement complexe (multi-modales), on converge vers une loi normale multivariée par exemple (Fig 40). Quelles sont les conditions pour que la chaîne de Markov converge?

124. NDJE. Le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2024/Ornstein_Uhlenbeck.ipynb donne un exemple en 1D avec en plus un terme de moyenne qui tire l'évolution de X_n .

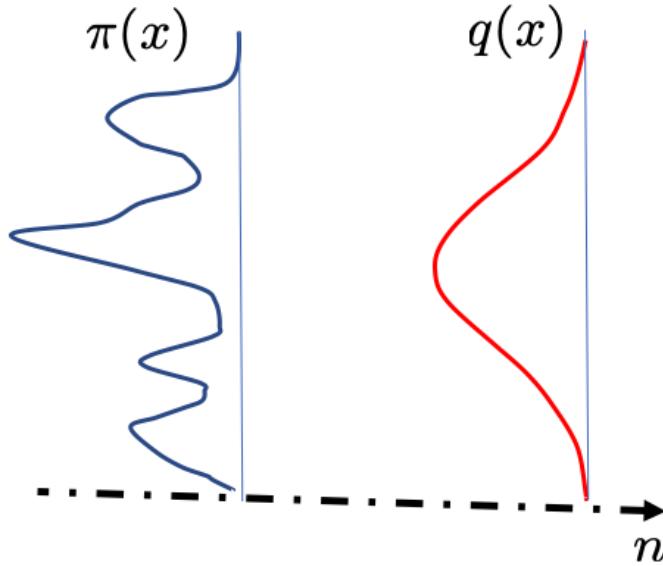


FIGURE 40 – Évolution de la distribution initiale $\pi(x)$ vers une distribution $q(x)$ (loi normale) par le processus de d'Ornstein-Uhlenbeck.

8.2 Loi invariante

La convergence en loi de la chaîne de Markov passe par les conditions d'existence d'une **loi stationnaire invariante**.

Définition 9 (loi invariante stationnaire)

Une loi π est stationnaire invariante, est une mesure de probabilité telle que dans les conditions des propriétés 5, nous avons

$$\boxed{\pi = P^T \pi} \quad (208)$$

c'est-à-dire que π est un vecteur propre de P^T de v.p égale à 1, et

$$\pi(y) = \sum_{x \in \chi} \pi(x) P_{x,y} \quad (209)$$

Ce que l'on veut c'est donc définir $P_{x,y}$ (sa transposée) pour que $\pi(x)$ (la donnée) soit bien

une vecteur propre associé à la valeur propre 1. Au passage nous avons le petit lemme suivant:

Lemme 3

Si $\pi(x)$ est invariante, et que $\mu_0 = \pi$ alors $\forall n, \mu_n = \pi$.

Ce qui va plutôt nous intéresser c'est la notion d'**inversion du temps** qui est l'étape de génération dans l'algorithme de Score Diffusion.

8.3 Conditions de réversibilité

Rappelons que dans le processus de fabrication de la chaîne de Markov, il y a non seulement les échantillons qui évoluent dans le temps (n) mais aussi leurs densités de probabilité. Donc, on passe de (X_0, μ_0) à (X_1, μ_1) , etc qui potentiellement vont converger vers un *v.a* associée à la mesure invariante $\mu_\infty = \pi$. Concernant les mesures le transport de μ_0 à π est *déterministe* (processus lié à P^T), alors que concernant les X_i , il s'agit d'un transport stochastique de variables aléatoires. Peut-on inverser ces transformations?

On sait que par définition de la matrice $P_{x,y}$

$$P_{x,y} = \mathbb{P}(X_{n+1} = y | X_n = x) \quad (210)$$

On s'intéresse alors à $\mathbb{P}(X_n = y | X_{n+1} = x)$. La formule de Bayes nous dit alors

$$\begin{aligned} \mathbb{P}(X_n = y | X_{n+1} = x) &= P(A|B) = \frac{P(B|A)P(A)}{P(B)} \\ &= \frac{\mathbb{P}(X_{n+1} = x | X_n = y)\mathbb{P}(X_n = y)}{\mathbb{P}(X_{n+1} = x)} \\ &= P_{y,x} \times \frac{\mathbb{P}(X_n = y)}{\mathbb{P}(X_{n+1} = x)} = Q_{x,y} \end{aligned} \quad (211)$$

Si, on a le bon goût de commencer le processus avec la mesure invariante π , d'après le lemme 3, alors

$$Q_{x,y} = \mathbb{P}(X_n = y | X_{n+1} = x) = P_{y,x} \times \frac{\pi(y)}{\pi(x)}$$

(212)

Définition 10 (Matrice inverse)

Soit un processus stationnaire (Déf. 5) avec une matrice $P_{x,y} = \mathbb{P}(X_{n+1} = y | X_n = x)$ ($\forall (x,y) \in \chi^2$). S'il existe une loi stationnaire $\pi(x) \neq 0$, c'est-à-dire que tous les x sont accessibles, alors on définit la matrice $Q_{x,y}$ suivante

$$Q_{x,y} = P_{y,x} \times \frac{\pi(y)}{\pi(x)} \quad (213)$$

La matrice Q est stochastique, $\sum_y Q_{x,y} = 1$.

(La propriété résulte que $\pi = P^T\pi$ et donc $\pi(x) = \sum_y \pi(y)P_{y,x}$).

Un cas particulier mais très pratique est celui de chaînes de Markov réversibles:

Définition 11 (Chaine de Markov réversible)

On dit qu'une chaîne de Markov stationnaire de matrice de transition P est réversible relativement à une distribution invariante π , si étant dans les conditions de la définition 10 alors $Q = P$, ce qui se traduit par

$$\pi(x)P_{x,y} = \pi(y)P_{y,x} \quad (\text{balance détaillée}) \quad (214)$$

(nb. la balance globale est donnée par l'équation de la mesure invariante $\pi = P^T\pi$).

On a alors le petit schéma suivant

$$X_1 = x \sim \pi \xrightleftharpoons[P_{y,x}]{P_{x,y}} X_2 = y \sim \pi$$

Les marches aléatoires ainsi que l'urne d'Erhenfest sont des exemples de processus inversibles. Cette notion d'égalité des flux de $x \rightarrow y$ et de $y \rightarrow x$ concerne le cas des processus de physique (statistique) à l'équilibre¹²⁵. On en déduit alors une propriété d'existence d'une mesure invariante:

125. NDJE. notée que la marche vers l'équilibre elle est irréversible en grande dimension, voir Cours 2023

Propriété 6 (existence d'une mesure invariante) Si P est réversible (balance détaillée) par rapport à la mesure π , alors π est invariante.

La démonstration est immédiate: on sait que $\forall x, y, \pi(x)P_{x,y} = \pi(y)P_{y,x}$, sommons sur y de part est d'autre

$$\sum_y \pi(y)P_{y,x} = \sum_y \pi(x)P_{x,y} = \pi(x) \underbrace{\sum_y P_{x,y}}_{=1 \text{ (mat. stoch.)}} = \pi(x) \quad (215)$$

qui n'est rien d'autre que $\pi = P^T\pi$, donc π est bien la distribution invariante.

Donc, par rapport à l'objectif de créer une chaîne de Markov qui a comme mesure invariant la distribution qui nous intéresse, cela va être de **construire des matrices de transition $P_{x,y}$ qui satisfont la condition de balance détaillée**. C'est l'objectif de l'algorithme de Metropolis-Hastings.

Maintenant, les propriétés de convergence sont reliées à la notion d'**ergodicité** (Déf. 6) et au **théorème de Birkhoff** (Th. 11). En effet, avec la matrice P , on propage une graine X_0 et la question est de **savoir est-ce que l'on parcours tout l'espace afin d'explorer en l'espèce les lieux où la probabilité $\pi(x)$ est grande**. Dans ce cas X_0 n'est pas important. En somme, il ne faut pas rester "coincer" dans des régions de l'espace des X (ex. des minima locaux des énergies dans le cas de probabilités de Gibbs). On peut traduire les propriétés de stationnarité et d'ergodicité de π : pour tout espace mesurable A (relatif à χ doté de la mesure π)

- $\pi(T^{-1}A) = \pi(A)$ (la mesure est stationnaire/invariante)
- $T^{-1}A = A$ alors soit $\pi(A) = 0$ (A est réduit à des points singuliers) soit $\pi(A) = 1$ (A est l'ensemble tout entier).

8.4 Convergence vers la loi invariante

Nous allons revisiter le théorème de Birkoff (Th. 11) dans le cas des chaînes de Markov. Premièrement il faut garantir que l'on explore tout l'ensemble χ :

Définition 12 (chaîne irréductible)

Une chaîne de Markov est irréductible si on peut passer de n'importe état x à n'importe quel état y (et vice-versa), c'est-à-dire

$$\forall (x, y) \in \chi^2, P_{x,y} > 0 \quad (216)$$

La seconde propriété, concerne le temps de retour:

Définition 13 (temps de premier retour)

Pour un état $X_0 = x$, on définit le "temps" T_x

$$T_x = \inf\{n; X_n = x\} \quad (217)$$

($T_x = +\infty$ si on ne peut atteindre x)

Comme X_n est une v.a, il en est de même de T_x , soit alors la définition suivante

Définition 14 (récurrent positif)

Un état x est dit récurrent positif si l'espérance de son temps de retour est fini.

$$\mathbb{E}_x[T_x] < \infty \quad (218)$$

Par extension, si tous les états sont récurrents, alors on dit que la chaîne est récurrente.

Si tel est le cas, imaginons que l'on soit dans un ensemble infini, la chaîne partant de $X_0 = x$ reviendra toujours en ce point en un temps fini. De plus, concernant la mesure invariante π nous avons pour tout $x \in \chi$:

$$\pi(x) = \frac{1}{\mathbb{E}_x[T_x]} \quad (219)$$

La dernière notion dont on a besoin concerne le caractère apériodique de la chaîne de

Markov. Soit le schéma de la figure 41: si l'on part de l'état "1" en $n = 0$, on y retourne en $n = 3, 6, \dots$ et $P_{1,1}^n = 1$ pour $n = 0 \pmod{3}$.

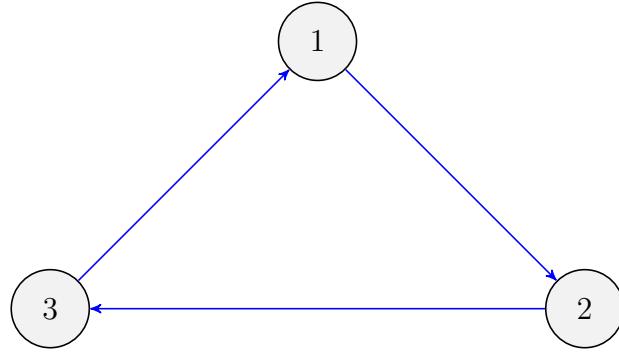


FIGURE 41 – Processus markovien de matrices de transition avec des transitions positives entre les états.

Soit alors la définition de la période d'un état:

Définition 15 (période d'un état, d'une chaîne)

Soit, l'ensemble des temps de retour à un état x

$$\tau_x = \left\{ n \geq 1 \text{ tq. } P_{x,x}^n > 0 \right\} \quad (220)$$

La période d'un état x est le PGCD de $\tau(x)$. Si la période est égale à 1, alors il est dit apériodique.

Par extension, une chaîne de Markov est dite apériodique si tous les états $x \in \chi$ sont apériodiques.

Cette notion est importante car elle garantie que l'on peut oublier les conditions initiales (c'est-à-dire le premier état X_0 de la chaîne). Une condition suffisante est que

$$\forall x, \quad P_{x,x} > 0 \quad (221)$$

car alors en 1 étape on peut revenir à l'état initial. Dans le schéma 41, il faudrait rajouter des flèches qui bouclent d'un état sur lui-même (ex. avec un poids 1/2).

Donc, pour récapituler, les considérations nous dirigent vers des chaînes irréductibles, récurrentes et apériodiques. On peut alors énoncé sans démonstration, le théorème fondamental des chaînes de Markov¹²⁶

Théorème 13 (Ergodicité)

Soit $(X_n)_n$ une chaîne de Markov, irréductible et récurrente positive, alors

1. la mesure définie par

$$\pi(x) = \frac{1}{\mathbb{E}_x[T_x]} \quad (222)$$

est l'unique mesure invariante de la chaîne.

2. pour n'importe quelle fonction f , tq. $\sum_x |f(x)|\pi(x) < \infty$, alors le théorème de Birkhoff nous dit que

$$\frac{1}{n} \sum_{k=1}^n f(X_k) \xrightarrow[n \rightarrow \infty]{p.s} \sum_x f(x)\Pi(x) = \mathbb{E}_{\pi}[f(x)] \quad (223)$$

3. Si de plus la chaîne est apériodique alors

$$X_n \xrightarrow[n \rightarrow \infty]{loi} \pi \quad (224)$$

autrement dit

$$\mathbb{P}(X_n = x) \xrightarrow{n \rightarrow \infty} \pi(x) \quad (225)$$

La propriété 2, nous sert en pratique pour calculer des intégrales par Monte Carlo en utilisant les échantillons itérés de la chaîne de Markov, tandis que la propriété 3 nous permet d'oublier la condition initiale de la chaîne¹²⁷.

Donc, la problématique est la suivante: si on se donne $\pi(x)$ (estimée dans une phase de modélisation), il faut trouver $P_{x,y}$ dont π est la mesure invariante (balance détaillée) et qui va générer une chaîne de Markov qui doit être irréductible récurrente positive pour pouvoir se servir des propriétés du théorème d'ergodicité.

126. NDJE. Primo, notez de petites différences par rapport à la version de 2023. Secundo, il y a une coquille dans la version de 2023 le point 3) que je rectifie ici, et les notes sont mises à jour sur le mon site github.

127. NDJE. voir Cours 2022 Sec. 5.5 la petite discussion sur la terminologie des convergences.

8.5 Point de vue d'analyse spectrale de la mesure invariante

Nous avons vu que la mesure invariante est le vecteur propre de P^T ayant une valeur propre égale à 1 (voir Déf. 9). Or, l'inconvénient du théorème 13 est qu'il ne renseigne pas sur la vitesse de convergence.

Soit alors le théorème d'analyse classique

Théorème 14 (**Perron-Frobenius**)

Soit $P_{x,y}$ une matrice stochastique (Eq. 200) et dont toutes les entrées sont strictement positives (chaîne irréductible Déf. 12) qui a une mesure invariante, alors toute valeur propre λ de $P_{x,y}$ est telle que $|\lambda| \leq 1$, et la valeur propre 1 est simple.

En conséquence, il existe une base de vecteurs propres $(\pi_k)_k$ dont les valeurs propres notées $(\lambda_k)_k$ ($P^T \pi_k = \lambda_k \pi_k$) sont telles que $|\lambda_k| \leq 1$. Ainsi, la mesure initiale μ_0 (que suit la v.a X_0) peut être décomposée sur cette base selon

$$\mu_0 = \sum_k \alpha_k \pi_k \quad (226)$$

Par transport successif, nous avons

$$\mu_n = (P^T)^n \mu_0 = \sum_k (\alpha_k \lambda_k^n) \pi_k \quad (227)$$

Quand n tend vers l'infini, toutes les composantes telles que $\lambda_k < 1$ disparaissent, et ne subsistent alors que la composante associée à la mesure invariante, mettons π_0 . A quelle vitesse la somme converge-t-elle? **Elle est dominée par la valeur propre dont le module est le plus grand inférieur à 1.** On appelle *le saut spectral* la différence $1 - \max_{k \neq 0} |\lambda_k|$, il gouverne la décroissance exponentiel des composantes différentes de la mesure invariante.

Le problème, c'est quand grande dimension, le contrôle du saut spectral est difficile et **on est très souvent dans la situation où les valeurs propres s'accumulent vers 1 (en module) et la convergence exponentielle devient compromise.**

8.6 Algorithme de Metropolis-Hasting

La méthode¹²⁸ développée d'abord en 1949 par Nicholas C. Metropolis (1915-99) et Stanisław Ulam (1909-84) fut reprise plus en détails en 1953 par Metropolis et collaborateurs puis étendue en 1970 par Wilfred Hastings (1930-2016). On la baptise l'algorithme de Metropolis-Hastings (MH) même si plusieurs auteurs y ont contribué. C'est une application de toutes les notions décrites dans les sections précédentes.

On veut que la densité de probabilité invariante soit $\pi(x)$ (c'est la donnée du problème). Rappel: on sait calculer $\pi(x)$, et nous verrons que l'on peut se passer de savoir calculer la constante de normalisation, ce qui en pratique est très appréciable. Si on arrive à concevoir une **matrice réversible** $P_{x,y}$ (Déf. 11), équation de **balance détaillée**, alors π est la mesure invariante, et avec un peu d'astuce on peut se trouver dans les conditions du **théorème d'ergodicité** (Th. 13).

Afin de connaître $P_{x,y}$, on fait une **proposition** notée $Q_{x,y}$ qui sera la matrice de transition¹²⁹ de $x \rightarrow y$; on va la choisir irréductible (rappel: à partir de n'importe quel x , on puisse atteindre n'importe quel y) et strictement positive: par exemple

$$Q(x, y) = (2\pi\sigma^2)^{-d/2} e^{-\frac{1}{2}\frac{\|x-y\|^2}{\sigma^2}} \quad (228)$$

Maintenant, comment construire $P_{x,y}$? L'idée est d'écrire

$$P_{x,y} = \rho(x, y) \times Q(x, y) \quad (229)$$

où $\rho(x, y)$ est une fonction que l'on va adapter. C'est une **procédure de "rejet"** (Sec. 7.5) sur la probabilité de transition proposée. En l'espèce $\rho(x, y) \in [0, 1]$.

L'objectif de la balance détaillée se traduit littéralement par

$$\rho(x, y)Q(x, y)\pi(x) = \rho(y, x)Q(y, x)\pi(y) \quad (230)$$

128. NDJE. Afin d'introduire des éléments pour utiliser un notebook, j'avais introduit cet algorithme en 2023.

129. NDJE. notée parfois $Q(y|x)$, et il y a d'autres notations. Attention à l'ordre, vérifier la façon dont $P_{x,y}$ est décomposée selon le produit de la distribution simple Q et la fonction d'acceptation/rejet.

Soit alors la définition suivante de $\rho(x, y)$

$$\boxed{\rho(x, y) = \min \left(1, \frac{Q(y, x)\pi(y)}{Q(x, y)\pi(x)} \right)} \quad (Metropolis - Hastings) \quad (231)$$

La balance détaillée est respectée et π est bien la mesure invariante.

En effet, supposons que $Q(y, x)\pi(y) > Q(x, y)\pi(x)$, alors $\rho(x, y) = 1$ et

$$\rho(y, x) = \frac{Q(x, y)\pi(x)}{Q(y, x)\pi(y)}$$

donc

$$P_{x,y} = Q(x, y) \quad P_{y,x} = Q(y, x) \times \frac{Q(x, y)\pi(x)}{Q(y, x)\pi(y)} = \frac{Q(x, y)\pi(x)}{\pi(y)} = \frac{P_{x,y}\pi(x)}{\pi(y)}$$

qui montre bien que la balance détaillée est satisfaite et donc π est la mesure invariante de la chaîne de Markov associée à la matrice de transition $P_{x,y}$. Le cas symétrique de l'hypothèse se traite de la même façon.

Notez comme annoncé que le calcul de $\rho(x, y)$ est un rapport de probabilités où les constantes de normalisation en sont absentes. Ceci est très important en pratique, car nous avons vu qu'estimer les normalisations des distributions de Gibbs sont des intégrales en grande dimension.

L'algorithme se déroule ainsi

Algorithm 3 Metropolis-Hastings

Require: $Q(x, y)$ une distribution facile à échantillonner pour obtenir x

- 1: Shoot $x_0 \sim \mu_0$ (ex. $Q(x, 0)$)
 - 2: **for** $i : 1, \dots, n$ **do**
 - 3: Shoot $x_{prop} \sim Q(., x_{i-1})$ and $u \sim \mathcal{U}(0, 1)$
 - 4: Compute $r = \rho(x_{i-1}, x_{prop}) = \min \left(1, \frac{Q(x_{prop}, x_{i-1})\pi(x_{prop})}{Q(x_{i-1}, x_{prop})\pi(x_{i-1})} \right)$
 - 5: **if** $r = 1$ OR $u \leq r$ **then** $x_i = x_{prop}$
 - 6: **else** $x_i = x_{i-1}$
 - 7: Keep x_i
 - 8: return $(x_i)_{i \leq n}$
-

Cet algorithme définit la matrice de transition $P_{x,y}$ suivante:

$$P_{x,y} = \rho(x, y)Q(x, y) + \alpha(x)\mathbf{1}(y = x) \quad \alpha(x) = 1 - \sum_{z \in \chi} \rho(x, z)Q(x, z) \quad (232)$$

(on vérifie aisément que $\sum_y P_{x,y} = 1$).

Maintenant, la question pratique est la suivante: **l'algorithme converge-t'il?** On sait que la mesure invariante est la bonne par construction, car la **balance détaillée est assurée** comme on l'a démontré ci-dessus. Donc, la question est de savoir si les hypothèses du théorème d'ergodicité sont satisfaites?

On se limite aux états $x \in \chi$ où $\pi(x) > 0$ (ceux de probabilité nulle ne sont pas à considérer). Or, $Q(x, y) > 0$ (par choix), donc $\rho(x, y) > 0$, ainsi $P_{x,y} > 0$ pour tout couple d'états, donc la chaîne est **irréductible**. Concernant l'apériodicité, pour que $P_{x,x} = 0$, nécessiterait que $\alpha(x) = 0$ donc que $\sum_z \rho(x, z)Q(x, z) = 1$ c'est-à-dire que $\rho(x, z) = 1$ (on accepte toujours) ce qui ferait de Q la distribution de probabilité recherchée. Donc, $P_{x,x} > 0$ ce qui est une condition suffisante d'**apériodicité** de la chaîne (nb. notons que $P_{x,x} > 0$ est la conséquence algorithmique que durant le processus il y a des cas où la proposition est rejetée, et donc $x_{i+1} = x_i$). Comme $\pi(x) > 0$, on en déduit que $\mathbb{E}_\pi[T_x] < \infty$ donc la chaîne est **récurrente positive**. Ainsi, **les conditions du théorème d'ergodicité sont réunies et donc on a bien la convergence vers la distribution souhaitée.**

Quid de la vitesse de convergence¹³⁰? Poser cette question n'est pas un détail, car c'est là où il y a beaucoup de pratiques basées sur l'expérimentation¹³¹. Le point crucial est que **la convergence peut être très lente**. Surtout à cause de la faiblesse du *saut spectral* en grande dimension qui peut complètement obérer la convergence en un temps fini. En fait, il faut pouvoir choisir $Q(x, y)$ judicieusement.

La première chose que l'on peut voir immédiatement qui va nous aider à comprendre

130. NDJE. pour les personnes qui voudraient aller plus loin voir par exemple l'article de Guanyang Wang de 2021 "*Exact Convergence Rate Analysis of the Independent Metropolis-Hastings Algorithms*" <https://arxiv.org/pdf/2008.02455v6.pdf>.

131. NDJE. Notamment, au fil de la pratique ont émergé le concept de *burning* pour ne prendre en compte que les échantillons à partir d'un certain rang, et de *thinning* algorithme opérant après cette phase de *burning*. Le plus simple des algorithmes de *thinning*, considère que chaque k-ème itération est conservée et le reste écarté. L'objectif et la volonté de réduire la corrélation positive entre les états restants, et donc de réduire la variance asymptotique des estimateurs. Cependant, cette pratique peut s'avérer très délicate.

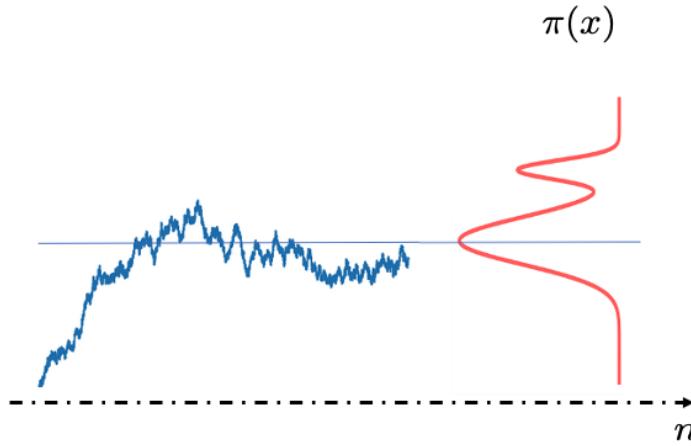


FIGURE 42 – Cas d'évolution d'une chaîne produite par l'algorithme de Metropolis avec les échantillons ne scrutant qu'un seul mode de la distribution $\pi(x)$.

le problème, c'est le cas particulier où $Q(x, y)$ est symétrique (Algorithme de Metropolis original). Dans ce cas, il vient

$$\boxed{\rho(x, y) = \min \left(1, \frac{\pi(y)}{\pi(x)} \right)} \quad (\text{Metropolis}) \quad (233)$$

Or, comme déjà dit en l'absence d'information *a priori* sur $\pi(x)$, en général on choisit $q(x) = Q(., y)$ une gaussienne multivariée de moyenne y et donc les rôles de x et y sont interchangeables. Mais, si $\pi(x)$ a deux modes (Fig. 42)¹³². La production de la chaîne commence, et supposons que le sigma de la gaussienne soit petit, si on se retrouve à accumuler des échantillons ayant une forte probabilité à cause de la présence d'un mode, alors le rapport $\pi(y)/\pi(x)$ va être proche de 1, on va toujours accepter les échantillons qui vont échantillonner uniquement ce mode. C'est-à-dire que l'on a quasiment aucune chance de visiter le second mode.

Pour remédier à cela, il faut d'une manière ou d'une autre faire en sorte que le support de $\pi(x)$ soit échantillonné: par exemple, on peut pour μ_0 utiliser une distribution avec une largeur plus importante que $Q(x, y)$ et faire évoluer en parallèle plusieurs chaînes

¹³². NDJE. Image issue du notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2023/Monte_Carlo_Sampling.ipynb.

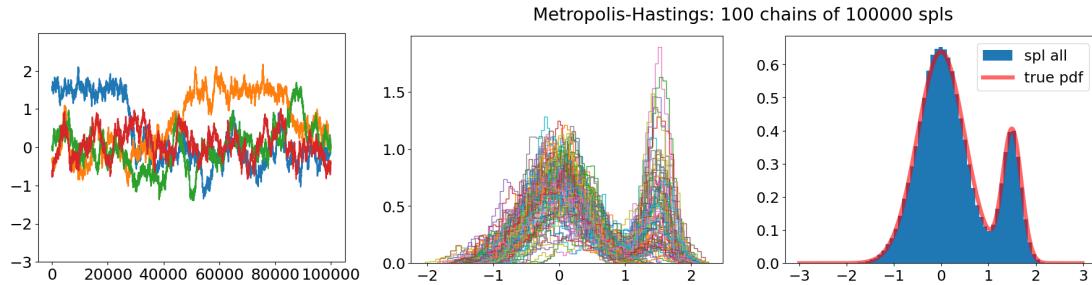


FIGURE 43 – Pour le même type de problème de la figure 42, on a utilisé une distribution assez large pour obtenir le premier échantillon et l'on a fait évoluer 100 chaînes en parallèle (on a utilisé 100,000 échantillons par chaîne après avoir enlever les 25,000 premiers.). On explore alors bien les deux modes de $\pi(x)$. A gauche: évolution des 4 premières chaînes. Au milieu: les histogrammes superposés des 100 chaînes. A droite: histogrammes de tous les échantillons cumulés et comparé à la distribution $\pi(x)$.

$(k \leq K)$ de graines $x_0^{(k)} \sim \mu_0(x)$. C'est le parti pris pour réaliser la figure 43.

Cependant, le problème principal est quand grande dimension le support de π est concentré dans une petite fraction de l'espace et beaucoup de chaînes peuvent passer du temps dans des zones où $r \leq 1$ qui fait baisser mécaniquement le taux d'acceptation de l'algorithme (c'est-à-dire son efficacité). **En fait, l'exploration d'une zone est conditionnée par la largeur de la distribution de proposition et son ajustement est toute la difficulté du problème. A moins qu'on se tourne vers des algorithmes qui orientent la recherche.** Par exemple, en utilisant des méthodes dites HMC pour Hybride Monte Carlo (ou parfois dit Hamiltonian Monte Carlo)¹³³.

Donc, la moralité si l'on peut dire, c'est qu'il faut un peu regarder ces théorèmes de convergence avec un certain recul. **Le passage à la grande dimension pose problème.** Il y a une façon très élégante que nous verrons à la prochaine séance, d'approcher le problème différemment avec les algorithmes de **Diffusion de Score**. La grande différence est que l'on va se servir des données pour modifier d'une certaine manière la distribution

133. NDJE. Voir une implémentation simple dans le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2003/Monte_Carlo_Sampling.ipynb, et le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2003/Monte_Carlo_Sampling_2.ipynb en montre l'usage à l'aide d'une librairie. Voir également l'article de Matthew D. Hoffman et Andrew Gelman (2014) pour une variante nommée No-U-Turn <https://jmlr.org/papers/volume15/hoffman14a/hoffman14a.pdf>.

de propositions de manière à garantir une convergence plus rapide y compris en grande dimension.

9. Séance du 13 Mars

Durant cette dernière séance nous allons aborder l'échantillonnage selon la méthode de la **Diffusion de Score** qui est à la base des algorithmes mis en oeuvre par exemple dans les modèles de langage tels que GPT et Gemini (Sec. 2.1). Rappelons que l'enjeu est d'échantillonner une distribution de type Gibbs en grande dimension:

$$p(x) = Z^{-1}e^{-U(x)} \quad x \in \mathbb{R}^d \quad (234)$$

Durant le cours, on a vu les expressions de $U(x)$ donnant les **distributions gaussiennes sans structure**, puis nous avons vu le cas de distributions plus complexes en **Physique Statistique** où l'on a utilisé des **champs de Markov pour lesquels les dépendances conditionnelles sont locales** (ex. théorie ϕ^4 et Ising). Tout cela est avant tout bien compris en Physique, et l'on arrive également à maîtriser avec les réseaux de neurones. **La grande surprise est que des très grands modèles ont le pouvoir de générer des images de visages, d'environnements au sens large etc. L'enjeu est de comprendre les mathématiques à l'œuvre derrière ces modèles génératifs.** Il y a certes des algorithmes que l'on comprend assez bien, mais la question est de savoir, est-on comme pour le cas de la Phys. Stat. en train d'estimer une distribution de probabilité sous-jacente aux visages, d'environnements etc? si tel est le cas **quelle est la nature de ces distributions?** Car en grande dimension, à cause de la malédiction de la dimensionalité, si l'on peut générer des visages, c'est qu'il **y a une forme de simplification de la structure des corrélations.** Notons que l'on peut estimer ces corrélations certes avec beaucoup d'exemples, mais ils sont peu nombreux comparativement à ce que l'on aurait pu s'attendre s'il y avait eu une explosion exponentielle avec la dimension.

L'idée de l'algorithme de Score Diffusion a été vue dans les chaînes de Markov (Fig. 44): on part de la distribution $p(x)$ qui est potentiellement très complexe, et par **transport** (indiqué par $t \in [0, T]$) on va la simplifier jusqu'à une distribution simple à échantillonner. Dans le cas de l'algorithme de Score Diffusion, **la distribution $p_T(x)$ est**

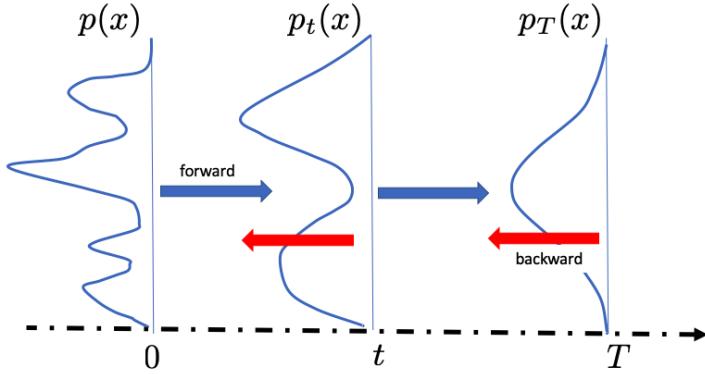


FIGURE 44 – Évolution de la distribution $p(x)$ au cours des étapes de transformation indiquées par t dans le sens d'ajouter du bruit (flèches bleues), et si l'on peut inverser le processus (flèches rouges) alors à partir d'un échantillon de $p_T(x)$ on obtient un échantillon nouveau de $p(x)$.

une gaussienne. Et l'enjeu ensuite, est d'**inverser le processus de transport** pour qu'à partir d'un échantillon de $p_T(x)$ on obtienne un échantillon de $p(x)$. **On va caractériser $p(x)$ par d'une part $p_T(x)$ et toutes les probabilités conditionnelles intervenant durant le transport inverse.**

Donc, le cadre est celui des **chaînes de Markov** (1906) mais tel quel ce cadre est trop général. En particulier, quel type de chaîne faut-il prendre? Que désigne la chaîne "forward" dans la figure 44. Le cadre qui va nous guider est celui du **Groupe de Renormalisation** (voir note 63) des années 1970 en Physique Théorique (Statistique et des Particules) où l'axe du "temps" t est un **axe d'échelle**. En **IA générative** par Score Diffusion (2020), l'axe du "temps" est celui de l'**addition d'un bruit**.

9.1 Équation d'Ornstein-Uhlenbeck

Il s'agit de l'évolution par transport ("aller"/*forward*) de la distribution $p(x)$ (Fig. 44). Soit l'équation de la *v.a* x_t (qui suit $p_t(x)$) qui s'écrit selon

$$dx_t = -x_t dt + \sqrt{2} dB_t \quad t \in [0, T] \quad (235)$$

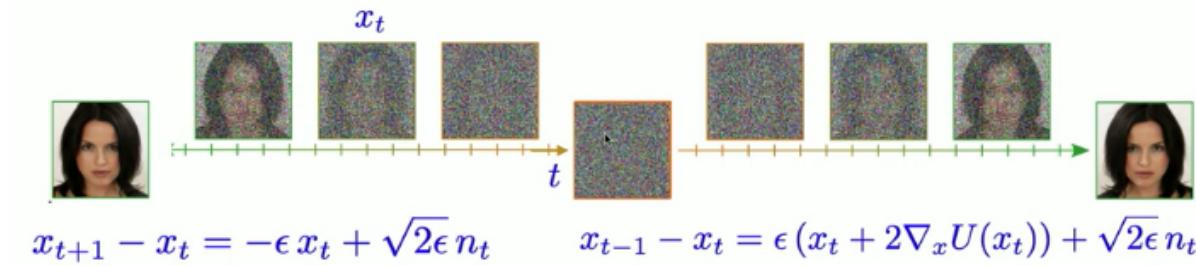


FIGURE 45 – Exemple d'évolution d'une image par le processus stochastique d'un modèle de diffusion: à gauche progression où l'on injecte du bruit (*forward*) et à droite où l'on opère un débruitage.

où dB_t est un mouvement brownien. Si on l'a discrétise avec un pas de temps δ :

$$x_{t+\delta} - x_t = -x_t \delta + \sqrt{2\delta} z \quad z \stackrel{iid}{\sim} \mathcal{N}(0, Id) \quad (236)$$

C'est une chaîne de Markov¹³⁴ que nous avons déjà étudiée (Sec. 8.1) qui s'écrit

$$x_{t+\delta} = (1 - \delta)x_t + \tilde{z} \quad \tilde{z} \stackrel{iid}{\sim} \mathcal{N}(0, 2\delta Id) \quad (237)$$

et quand $\delta \rightarrow 0$ la solution au temps t prend la forme compacte

$$x_t = e^{-t}x_0 + (1 - e^{-2t})^{1/2}z \quad z \stackrel{iid}{\sim} \mathcal{N}(0, Id) \quad (238)$$

La démonstration est immédiate. On remarque "l'oubli de la distribution initiale", et la convergence se fait de manière exponentielle vers le bruit blanc gaussien. L'illustration de cette évolution de $p(x)$ est montrée sur la figure 45 (c'est la même de la section 2.7 reproduite ici pour facilité la lecture).

Quelle est l'expression de la distribution de x_t ? Dans le mode "forward" on sait que l'on converge vers un bruit blanc gaussien. Soit la renormalisation suivante de x_t qui fait

^{134.} NDJE. voir notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2024/Ornstein_Uhlenbeck.ipynb

apparaître l'explosion en variance au fur et à mesure de l'évolution temporelle

$$\tilde{x}_t = e^t x_t = x_0 + \underbrace{(e^{2t} - 1)^{1/2}}_{\sigma_t} z \quad (239)$$

On a une équation du type où le "signal bruité" est la somme du "signal original" et du "bruit" dont la variance croît exponentiellement. À tout "instant" t

$$p_t(\tilde{x}_t) = \int p(\tilde{x}_t, x_0) dx_0 = \int p_t(\tilde{x}_t | x_0) p_0(x_0) dx_0 \quad (240)$$

où

$$p_t(\tilde{x}_t | x_0) = \frac{1}{(2\pi\sigma_t^2)^{d/2}} e^{-\frac{\|\tilde{x}_t - x_0\|^2}{2\sigma_t^2}} = g_{\sigma_t}(\tilde{x}_t - x_0) \quad (241)$$

On constate que l'on a une **convolution**

$$p_t(\tilde{x}_t) = \int p_0(x_0) g_{\sigma_t}(\tilde{x}_t - x_0) dx_0 = (p_0 * g_{\sigma_t})(\tilde{x}_t) \quad (242)$$

On procède donc à un lissage progressif de la distribution initiale p_0 avec une gaussienne.

Maintenant, **il nous faut inverser le processus** en calculant les probabilités conditionnelles inversées. Dans le sens *forward* on a besoin $p(x_{t+\delta} | x_t)$ et dans le sens *backward*, c'est $p(x_t | x_{t+\delta})$ qu'il nous faut, sachant que pour une évolution continue par eq. différentielle, $\delta \rightarrow 0$. Si l'existence de la chaîne inverse ne pose pas de problème en soit, c'est son calcul qui retient notre attention. On va donner des arguments justifiant que la variable x_{T-t} suit l'équation suivante:

$$dx_{T-t} = (x_{T-t} + 2\nabla_x \log p_{T-t}(x_{T-t})) dt + \sqrt{2} dB_t \quad t \in [0, T] \quad (243)$$

il s'agit d'une **équation de Langevin amortie** où l'on voit apparaître le **score** (Sec. 5.4). Pour comprendre cette structure d'équation¹³⁵, on va prendre le point de vue du débruitage qui n'a pas été le point de vue ce ceux qui ont fait émerger la solution via les eq. différentielles stochastiques¹³⁶.

135. NDJE. pour les personnes intéressées voir D. McAllester *On the Mathematics of Diffusion Models*, <https://arxiv.org/pdf/2301.11108.pdf>.

136. Yang Song et al. (2020) "Score-Based Generative Modeling through Stochastic Differential Equa-

9.2 Problème du débruitage

Le problème classique est celui de l'observation y d'un signal x entaché d'un bruit additif z tel que

$$y = x + z \quad z \stackrel{iid}{\sim} \mathcal{N}(0, \sigma^2 Id) \quad (244)$$

L'enjeu est de retrouver au mieux x à partir de y . Pour ce faire on a un estimateur de x , noté $\hat{x}(y)$, qui va minimiser l'erreur quadratique en moyenne, c'est-à-dire minimiser

$$\mathbb{E}_{x,z} \|x - \hat{x}(y)\|^2 \quad (245)$$

où les x et z suivent leur lois respectives.

Petit rappel: soit x une *v.a*, si l'on cherche une constante μ qui approxime au mieux x en erreur quadratique, la solution est

$$\min \mathbb{E}_x [\|x - \mu\|^2] \Leftrightarrow \mu = \mathbb{E}_x[x] \quad (246)$$

Dans notre cas du débruitage, on a une information en plus à savoir la connaissance de y (l'observation) donc:

$$\hat{x}(y) = \mathbb{E}_x[x|y] = \int x p(x|y) dx \quad (247)$$

Le problème du calcul de cet estimateur optimal a essentiellement débuté dans les années 1940 avec les travaux de Wiener¹³⁷. Le problème en grande dimension est le calcul de $p(x|y)$, on donc a commencé par simplifier la recherche, par exemple en essayant de trouver un estimateur linéaire en y , tel que $\hat{x}(y) = L.y$ et où l'opérateur L est optimisé. On a par la suite utilisé d'autres classes d'opérateurs.

Mais peut-on caractériser la solution optimale qui est celle de l'équation 247? Là il y a une proposition qui va faire le lien avec la section précédente qui est la suivante:

tions", <https://arxiv.org/pdf/2011.13456.pdf>.

137. Norbert Wiener (1894-1964)

Théorème 15 (*Tweedie, Robbins, Miyasawa-1956-61*)

La solution $\hat{x}(y)$ s'écrit

$$\hat{x}(y) = y + \sigma^2 \nabla_y \log p(y) \quad (248)$$

où apparaît le score.

Démonstration 15. La distribution $p(y)$ s'écrit en reprenant le calcul de l'équation 242

$$p(y) = \int p(x) g_\sigma(y - x) dx \quad (249)$$

Donc, il vient naturellement et grâce à la gaussienne g_σ

$$\nabla_y p(y) = \int p(x) \left(-\frac{y-x}{\sigma^2} \right) g_\sigma(y - x) dx == \frac{1}{\sigma^2} \int (x-y) p(y, x) dx \quad (250)$$

Ainsi, en calculant le score, on a

$$\sigma^2 \nabla_y \log p(y) = \int (x-y) \frac{p(x|y)p(y)}{p(y)} dx = \int x p(x|y) dx - y = \hat{x}(y) - y \quad (251)$$

■

Avec ce théorème, on comprend pourquoi **le score de la distribution bruitée apparaît dans le processus backward** (Eq. 243) qui est par essence une forme de débruitage. Mais, comment estimer le score surtout que l'on se trouve en grande dimension? La grosse surprise est que l'on y arrive avec un réseau de neurones.

9.3 Réseau de débruitage

L'enjeu est d'estimer le terme $\nabla_x \log p_{T-t}(x_{T-t})$ (le score) dans l'équation 243. L'idée est d'utiliser un réseau de débruitage. En effet, si on est capable d'obtenir l'estimateur optimal $\hat{x}(y)$, c'est-à-dire celui qui minimise le risque quadratique et donc donne accès à x , alors on estime aussi le score (Eq. 248).



FIGURE 46 – Schéma d'un débruiteur constitué par un réseau de neurones dont il faut déterminer les paramètres θ . Comme sous-produit le réseau fournit un estimateur du score via $\hat{x}(y) - y$ (au facteur de variance de bruit près).

Donc, l'idée est de construire un débruiteur sous forme d'un réseau de neurones (Fig. 46). De manière classique, partant de y l'observable bruitée, au passage à travers le débruiteur dépendant des paramètres θ , on recueille $\hat{x}_\theta(y)$. Afin de déterminer la valeur optimale de θ , on est naturellement amené à utiliser le risque quadratique que l'on minimise avec des échantillons $\{x_i, y_i\}_{i \leq n}$ où $y_i = x_i + z_i$ (z_i bruit blanc gaussien indépendant de x_i de variance σ^2):

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \frac{1}{n} \sum_{i \leq n} \|\hat{x}_\theta(y_i) - x_i\|^2 \quad (252)$$

Ainsi un estimateur du score est obtenu par

$$\hat{S}(y) = \frac{1}{\sigma^2} (\hat{x}_{\theta^*}(y) - y) \approx \nabla_y \log p(y) \quad (253)$$

La question que l'on peut se poser dans le cas de l'estimation de $\nabla_x \log p_{T-t}(x_{T-t})$ (pour tout $t \in [0, T]$), c'est quels sont les échantillons d'entraînement? En fait, **on va se servir de la phase forward** pour entraîner un débruiteur "universel" c'est-à-dire qu'il puisse débruiter sans connaître a priori la valeur de σ_t . On constate que ce type de réseau fonctionne mieux que si on l'entraînait à ne débruiter que pour un unique niveau de bruit. Peut-on expliquer ce phénomène?

Imaginons que le signal x "vive" sur une variété (en grande dimension) potentiellement très irrégulière, alors la densité de probabilité $p_0(x)$ est une mesure dont le support est concentré sur cette variété. On a vu¹³⁸ que $p_t(x_t)$ est une forme de lissage de $p_0(x)$ par une gaussienne g_{σ_t} (Eq. 240). Dans le mode *backward*, au départ $t = T$ la valeur de σ_T est grande, ce qui donne à $p_T(x_T)$ une forme extrêmement régulière. Au fur et à mesure, le σ_t diminuant, on commence à voir les détails de p_0 . Donc, au fur et à mesure, si on initialise

^{138.} nb. on l'a fait avec \tilde{x}_t mais cela vaut pour x_t .

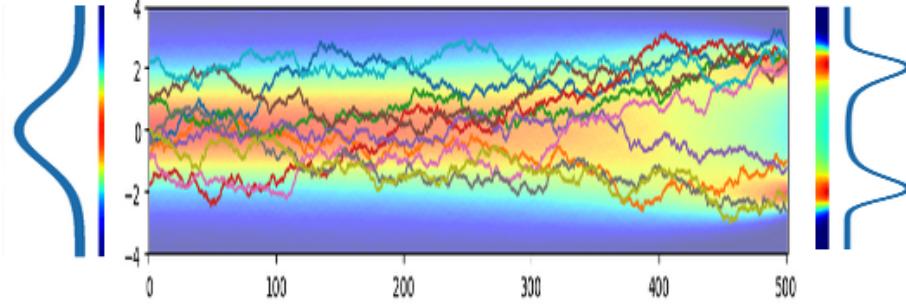


FIGURE 47 – Évolution de chaînes de Markov dans le mode backward partant d'une distribution gaussienne vers une distribution à 2 modes.

une chaîne de Markov avec $p_T(x_T)$ très lisse, l'évolution de $p_t(x_t)$ va donc piloter la chaîne vers les zones de forte probabilité. Ceci est illustré¹³⁹ par un exemple simple sur la figure 47. C'est une déformation homotopique de la distribution $p_T(x_T)$ qui est un bruit blanc vers la distribution $p_0(x_0)$ potentiellement très complexe.

9.4 Transition entre mémorisation et généralisation

S. Mallat nous décrit succinctement l'architecture typique de réseaux convolutionnels utilisés dans ce cadre de débruitage que sont les U-Nets¹⁴⁰ (Fig. 22). Une première partie du réseau effectue des opérations de convolution et sous-échantillonnage tandis que la seconde procède aux opérations de déconvolution et sur-échantillonnage pour redonner l'image d'origine. Il nous donne également des images issues de modèles génératifs que tout à chacun a vu ici ou là dans la presse et qui donne par exemple des visages de célébrités qui "n'existent pas" comme celles de la figure 48. D'autres types d'images comme des chambres-à-coucher sont également bien générées après entraînement. Enfin, nous avons les images générées à partir d'une requête formulée en langage naturel telle que celle de la figure 1 de la section 2.2.

139. NDJE. extrait du notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2024/ScoreDiffusionGene.ipynb.

140. nb. Olaf Ronneberger, Philipp Fischer et Thomas Brox (2015) ont développé une telle architecture pour un problème de segmentation d'images médicales. Voir <https://arxiv.org/abs/1505.04597>.



FIGURE 48 – Quelques exemples d’images de visages issues d’un modèle génératif entraîné avec une grande base de données à l’aide d’une méthode de diffusion de score.

La question qui se pose réellement quand on voit de telles images produites par ces modèles génératifs, est la suivante: **est-on en train d’échantillonner une densité de probabilité?** En d’autres termes le modèle a-t’il appris $p(x)$ avec x une image de visage, et quelle est la nature de cette distribution p ? L’alternative serait probablement que les nouvelles images soient des sortes de patchworks sophistiqués des images de la base de données? *NDJE. S. Mallat nous a donné quelques éléments de la réponse dans sa première séance de cette année (Sec. 2.7, Fig. 8), il nous les reformule dans cette séance.* L’étude consiste à entraîner deux modèles génératifs avec deux lots disjoints de taille N d’une même base de données d’images de visages. Puis, on fait générer une image par les deux modèles à partir de la même image initiale de bruit blanc gaussien. Le résultat est le suivant: quand la taille N des lots d’entraînement augmente on passe d’un mode de *mémorisation* à un mode de *génération*. Ceci est illustré¹⁴¹ dans la figure 49. **On a bien appris le transport inverse déterministe** qui permet à partir de toute nouvelle réalisation d’un bruit blanc de générer une nouvelle image de visage indépendante de la base de donnée initiale¹⁴². Si on réitère la synthèse d’image par les deux modèles entraînés avec 100,000 images, en leur fournissant la même image initiale de bruit à chaque fois,

141. NDJE. source: Zahra Kadkhodaie, Florentin Guth, Eero P. Simoncelli et Stéphane Mallat (2024) *Generalization in diffusion models arises from geometry-adaptive harmonic representations* <https://arxiv.org/pdf/2310.02557.pdf>.

142. NDJE. du moins dans une certaine mesure car si l’on ne prend qu’un seul type de visage de célébrités hollywoodiennes de peau blanche, on ne va pas générer un visage d’un homme ou femme asiatique, africaine, etc.

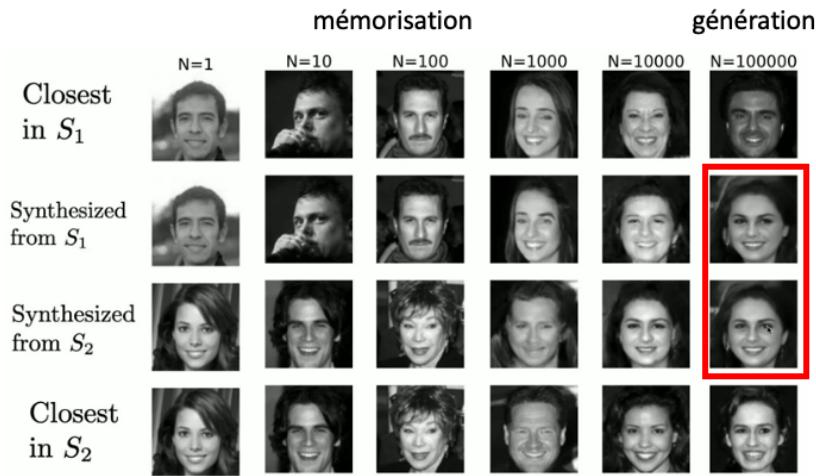


FIGURE 49 – Deux modèles génératifs S_1 et S_2 sont entraînés avec deux bases de données sans recouvrement et chacune de taille N . On donne ensuite la même image de bruit aux deux modèles afin de synthétiser une nouvelle image (80×80). On peut trouver l'image de la base de données qui approcherait au mieux cette image synthétisée. Tant que $N < O(10,000)$, S_1 et S_2 produisent des images différentes et qui ressemblent d'autant plus à une image de la base de données que N est petit. Dès que $N = 100,000$, les images synthétisées par S_1 et S_2 sont d'une part (quasi-)identiques et d'autre part différentes des images les plus proches des deux bases de données. On assiste bien à une transition vers un modèle qui apprend une distribution de probabilité.

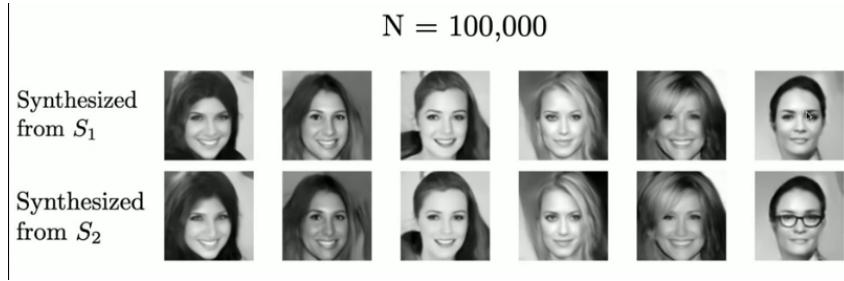


FIGURE 50 – Images synthétisées des modèles S_1 et S_2 (Fig. 49) entraînés avec $N = 100,000$ images et pour lesquels on donne à chaque fois la même image de bruit blanc gaussien en début de génération. La quasi-totalité des images synthétisées sont identiques à quelques instabilités près.

on obtient les mêmes images générées, sauf à des détails près ce qui est le reflet des différences des deux réseaux optimisés (Fig. 50). Le même type d’expérience a été menée avec d’autres types d’images (ex. chambres-à-coucher). Les conclusions sont les mêmes.

La transition mémorisation versus génération aux alentours de $N = 100,000$ dépend de la taille du réseau de neurones qui apprend le score et de la résolution des images: dans les expériences menées ci-dessus, le réseau était un U-Net à $7 \cdot 10^6$ paramètres et les images étaient de taille 80×80 pixels. Si on a des images de taille 40×40 , on a besoin d’un réseau plus petit, et la généralisation est effective dès lors que $N = O(10,000)$. Somme toute on est dans une explication classique d’interpolation: plus on a de paramètres plus on a besoin d’images d’entraînement. Ce qui est plus surprenant, c’est que si on estime la taille de la variété où se situent les chambres-à-coucher, on intuite une dimension de l’ordre de $\log_2(10^5) \approx 17$ ce qui est relativement modeste en comparaison du nombre de pixels ($80^2 = 6,400$). Donc, on se dit que finalement les chambres-à-coucher sont à peu près toutes de composition identique à des détails près. Ce qui amène S. Mallat à avoir une réflexion d’ordre général sur la perception que l’on peut avoir sur la diversité du monde: en quelque sorte **ne surestime-t-on pas la complexité du monde?** Et si tout compte fait, le monde est plus simple qu’il n’y paraît, alors cela permettrait de comprendre en fait pourquoi les réseaux de neurones arrivent à capturer les corrélations qui structurent ces images.

9.5 Ouverture vers quelques pistes de recherche

S. Mallat développe des axes de recherche et continue sa séance en nous projetant des diapositives (*NDJE. vers 56:15 après le début de la vidéo.*). Ce sont des thèmes qu'il abordera en 2025.

9.5.1 Retour sur le débruitage: bases d'ondelettes

Le problème du débruitage, comme déjà évoqué, date des années 1940. *NDJE. Notons également que le cours de 2021 abordait le point de vue classique des représentations parcimonieuses qui débouchait sur la compression de données, mais aurait tout aussi bien pu couvrir l'aspect du débruitage.* En fait, prenons une base orthonormée $\{\psi_k\}_k$ et décomposons le signal bruité $x_t = x + z$ ($z \sim \mathcal{N}(0, \sigma_t^2 Id)$):

$$x_t = \sum_k \langle x_t, \psi_k \rangle \psi_k \quad (254)$$

Comment enlever la composante du bruit? En fait la décomposition étant linéaire

$$\langle x_t, \psi_k \rangle = \langle x, \psi_k \rangle + \langle z, \psi_k \rangle \quad (255)$$

Or, z étant une *v.a* gaussienne, il en est de même de $\langle z, \psi_k \rangle$ et cela va générer des petits coefficients (tout du moins si σ^2 n'est pas trop grand par rapport au signal). Donc, l'application d'un **seuillage par une non-linéarité (ReLU)** (Fig. 51) permet de ne garder que les grands coefficients que l'on espère être suffisants pour reconstruire une bonne approximation du signal:

$$\hat{x} = \sum_k \rho_T(\langle x_t, \psi_k \rangle) \psi_k \quad \text{avec} \quad \rho_T(u) = \max(u - T, 0) \quad (256)$$

Typiquement, on choisit $T \approx (3 \div 5)\sigma$.

Il s'agit d'un **algorithme de débruitage adaptatif** développé dans les années 80 décrits dans le cours de 2021. Il est *adaptatif* car il s'adapte à décomposition du signal, en comparaison le débruitage de Fourier mettrait une coupure sur $k < k_{max}$ qui en contexte serait un indice de fréquence.

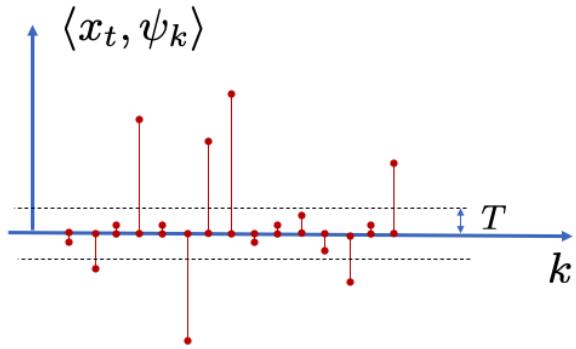


FIGURE 51 – Exemple de seuillage des produits scalaires pour éliminer au mieux le bruit qui génère des petits coefficients en comparaison des quelques grands coefficients qui suffisent à reconstruire une bonne approximation du signal.

L’enjeu est de trouver **la bonne base de représentation** pour avoir une décomposition du signal la plus parcimonieuse possible. Dans ce cadre, ont été développées dans les années 90 les **analyses multi-résolutions à bases d’ondelettes**¹⁴³. L’intérêt des ondelettes est d’obtenir des représentations parcimonieuses du signal dont les coefficients sont grands uniquement aux zones de discontinuité. Imaginons, une image d’un objet tel qu’un vase sur un fond uni, les coefficients se localisent aux bords du vase. Idem pour les contours d’un visage, les yeux dans un visage, etc. Et ceci à toutes les échelles spatiales et toutes les zones d’une image. On a donc une représentation bien adaptée pour la reconnaissance de formes et leur identification même en présence de bruit. Un exemple de débruitage sur un signal 1D est montré¹⁴⁴ sur la figure 52. Remarquer que le débruitage n’a pas consisté à faire un moyennage par exemple sur une fenêtre glissante ou bien en éliminant les composantes de Fourier de haute fréquence: **on a conservé toutes les discontinuités du signal non bruité**. Les artefacts résiduels aux endroits des discontinuités ont une forme de phénomène de Gibbs mais très amoindri par rapport à son pendant en Fourier. La généralisation en 2 dimensions des bases orthonormales d’ondelettes est quasi-immédiate et l’on peut également effectuer un seuillage des coefficients pour obtenir un débruitage

143. NDJE. Je vous invite à consulter le cours de 2021 qui couvre ce que S. Mallat nous dit rapidement dans ces diapositives de cette année

144. NDJE. J’ai utilisé les fonctions `swt/iswt` librairie `pywavelet`, avec `level=6` pour un signal de 2048 échantillons et "db2" comme ondelette.

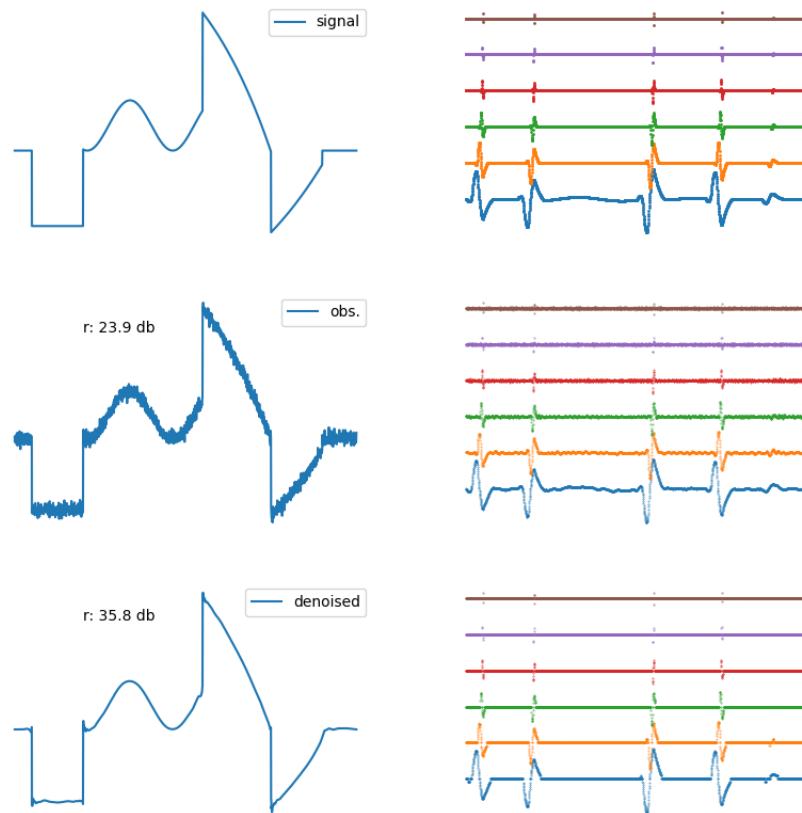


FIGURE 52 – Exemple de seuillage de débruitage par simple seuillage à $T = 5\sigma$ sur les coefficients d'ondelettes. A gauche: le signal sans bruit (haut), le signal auquel on a ajouté un bruit blanc gaussien $\sigma = 3$ (milieu), le signal reconstruit après seuillage. A droite: les coefficients de détails des trois signaux à différentes échelles (se raffinant de plus selon les ordonnées croissantes). La métrique utilisée est $r = 10 \log_{10}(\|x_r\|^2 / \|x - x_r\|^2)$ où x_r est le signal sans bruit pris comme référence.

tout aussi efficace.

9.5.2 Réseaux débruiteurs: que font-ils?

Quel est le lien avec ce que font les réseaux de neurones? On a vu qu'effectuer un débruitage optimal (Th. 15) c'est estimé le score d'une distribution

$$\hat{x}(x_t) = x_t + \sigma_t^2 \underbrace{\nabla_{x_t} \log p(x_t)}_{\hat{s}(x_t)} \quad (257)$$

On va pour cela utiliser un réseau débruiteur universel optimisé lors de la phase forward (Sec. 9.3) tel que

$$\hat{x}(x_t) = \hat{x}_{\theta^*}(x_t) \quad (258)$$

Or, si le réseau de neurones n'utilise que des ReLU et que tous les termes de biais ont été supprimés¹⁴⁵ alors il est *localement homogène de degré 1* et d'après le théorème d'Euler¹⁴⁶ il vient

$$\hat{x}_{\theta^*}(x_t) = \nabla_x(\hat{x}_{\theta^*}(x_t)) x_t \quad (259)$$

où apparaît le Jacobien de \hat{x}_{θ^*} . On peut le diagonaliser¹⁴⁷ dans une base orthonormale des vecteurs propres notés $\{\psi_k\}_k$ associés aux valeurs propres $(\lambda_k)_k$. Projetons x_t sur cette base, on peut alors écrire

$$\hat{x}(x_t) = \sum_k \lambda_k \langle x_t, \psi_k \rangle \psi_k \quad (260)$$

Remarquons alors que si $\lambda_k \approx 0$ le coefficient $\langle x_t, \psi_k \rangle$ est annulé. **On effectue un seuillage des coefficients de la décomposition de x_t via les valeurs propres.** Or, contrairement au

145. NDJE. réseau "bias-free": sont supprimées toutes les constantes additives des opérations de convolution et des opérations de normalisation par lots (c'est-à-dire que la normalisation par lots ne soustrait pas la moyenne).

146. NDJE. Une fonction homogène de degré k de \mathbb{R}^d vers \mathbb{R} se caractérise par

$$f(tx_1, tx_2, \dots, tx_d) = t^k f(x_1, x_2, \dots, x_d)$$

et le théorème d'Euler dit que

$$kf(x_1, x_2, \dots, x_d) = \sum_{i=1}^f x_i \frac{\partial f}{\partial x_i}((x_1, x_2, \dots, x_d))$$

147. NDJE. usage d'une décomposition en valeurs singulières, SVD.

cas de débruitage par seuillage des coefficients d'ondelettes, ici **la base orthonormale n'est pas choisie a priori, elle vient de la diagonalisation du Jacobien, elle va donc changer en fonction du signal d'entrée x_t** .

On peut à ce stade se poser un certain nombre de questions face au problème d'estimation de probabilité:

- la première est **la dépendance vis-à-vis des données**, a-t-on une invariance ou pas du résultat face au jeu de données (d'entraînement)? Il semble d'après l'expérience numérique sur la génération des images de visages et de chambres-à-coucher que **l'on a une certaine indépendance** vis-à-vis de la base de données d'entraînement (*NDJE. au bémol près que l'on ne génère pas tous les types de faciès, ni tous les types de chambres existantes dans le monde*).
- Mais utilisons un argument (trivial) suivant: on peut toujours générer une image remplie de 0 quelques soient les données, alors le résultat est par construction indépendant des données. Ainsi, il n'est pas suffisant de dire que le résultat est indépendant de la base de données. il faut donc **questionner la précision du résultat**.

S. Mallat parle de "variance" pour la première et de "biais" pour la seconde. On peut comprendre la variance ainsi: si le résultat est dépendant de la base de données, en changer modifie le résultat, on a donc une fluctuation du résultat induite par le choix de base de données. Qui dit fluctuation dit variance. Concernant le biais, il s'agit de la précision avec laquelle vous aller obtenir le résultat¹⁴⁸.

Donc, la question est **de savoir si on a bien estimé** $\nabla_x \log p(x) = -U(x)$? Dans l'expérience de génération des visages/chambres il semble à vue d'œil que les échantillons générés ressemblent bien à ce que l'on s'attend. Peut-on avoir une confirmation mathématique de cette impression visuelle? Là il faut pouvoir trouver/extraire une question d'ordre mathématique qui capture l'essence du problème que l'on cherche à démontrer sans que ce cela soit trop simple mais que cela soit abordable néanmoins.

148. NDJE. Il faut bien se placer dans le contexte où le vocabulaire est utilisé. Cette terminologie peut être sentie différemment. Lorsque l'on effectue en Physique une mesure, on indique l'erreur statistique (variance) qui est due au nombre d'échantillons récoltés pour effectuer la mesure, et l'erreur systématique (biais) liée à la façon d'effectuer la mesure, d'estimer les efficacités, d'estimer les calculs théoriques d'extractions de la quantité souhaitée, etc (voir on peut identifier plusieurs types de systématiques dont celles qui dépendent de la statistique par exemple des simulations pour estimer une efficacité de détection).

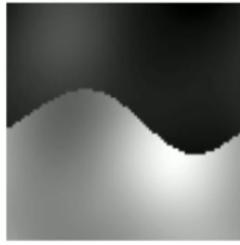


FIGURE 53 – Image type C^α où l'on a une courbe régulière sur un fond régulier où la régularité est contrôlée par α . Par exemple $\alpha = 4$.

9.5.3 Résultats d'expérimentations numériques

En l'espèce on va utiliser des images plus simples comme celle de la figure 53 où le fond a α dérivées tout comme la ligne qui sépare deux régions. Ce type d'images, nous dit S. Mallat, ont été beaucoup étudiées dans les années 2000, car à cette époque l'enjeu était de montrer que les techniques de débruitage par ondelettes "traditionnelles" n'étaient pas optimales, à cause d'une incapacité à s'adapter à **la géométrie du problème**. Disons simplement que les supports des ondelettes certes localisées sont des sortes de pavés dont les déformations ne s'adaptent pas géométrie de la courbe de transition. **On aimerait des ondelettes dont le support peut épouser les contours.** Cela à déboucher sur une panoplie de travaux dont les bases de "bandelettes"¹⁴⁹. L'idée est donc de comprendre comment construire des estimateurs optimaux du signal bruité? Donc, à partir de ces images x , on applique un bruit $z \sim \mathcal{N}(0, \sigma^2)$, et on note $x_t = x + z$. Il est démontré que l'estimateur optimal \hat{x} a une erreur telle que

$$\|\hat{x} - x\|^2 \sim \sigma^{-2\alpha/(\alpha+1)} \quad (261)$$

quand on a effectué un seuillage des coefficients dans la base de bandelettes. A l'époque, nous dit S. Mallat, arriver à obtenir l'optimum était difficile, et on n'y arrivait pas tout à fait (il y avait un facteur $\log \sigma$). Mais cela donne un cadre mathématique bien circonscrit. La question est: que va faire le réseau de neurones de type **U-Net, est-il capable d'obtenir l'optimum?**

¹⁴⁹. NDJE. voir par ex. Ch. Dossal, E. Le Pennec et S. Mallat (2011) <https://www.di.ens.fr/~mallat/papiers/2011-SigPro-DLPM.pdf> ainsi que les travaux faits avec G. Peyré.

Les résultats¹⁵⁰ sont montrés sur la figure 54. Tout d'abord (figure du haut), on peut regarder les vecteurs de la base orthogonale du Hessien. Ils sont indicés par leurs valeurs propres et ce que l'on constate c'est que 1) l'on observe des sortes de fonctions oscillantes dans les 2 zones de l'image (haut/bas) de moyennes nulles de part et d'autre de la frontière 2) on peut distinguer une petite bande qui suit la frontière. Il est important de noter que les fonctions oscillantes calculent des variations sur des zones régulières et non pas des variations à travers la frontière car alors cela donnerait des grands coefficients à cause des discontinuités qui viendraient contaminer tous les autres coefficients¹⁵¹. **Tout cela nous dit que ces vecteurs propres forment bien une base optimale** que l'on connaissaient mathématiquement et que le réseau a appris avec le débruitage des images. Ensuite (figure du bas), on peut comparer la vitesse du débruitage: c'est-à-dire l'évolution de la fonction qui donne le PSNR¹⁵² de l'image débruitée en fonction du PSNR de l'image bruitée. On ne maîtrise que la pente théorique. On constate que **le réseau se comporte d'une manière également optimale**.

Un autre exemple est montré sur la figure 55: l'image de base est un disque de rayon et position variables, tout comme les couleurs (niveaux de gris) à l'intérieur et à l'extérieur du disque. On a donc une variété de dimension 5 de l'espace tangent dans lequel baignent les images. On peut en calculer les 5 vecteurs propres (figure du haut). Sur la figure du bas, on voit que le réseau apprend également 5 vecteurs propres qui ressemblent très fortement à ceux calculés, il doit cependant ajouter d'autres composantes sous optimales, mais elles respectent bien les caractéristiques déjà invoquées.

Donc, il semble bien que pour des cas où l'on sait calculer les vecteurs propres optimaux, on peut attester que le réseau **débruite de manière optimale**, donc **il estime le score et donc la probabilité de manière optimale**. Que fait-il avec des images de visages? Un exemple est donné sur la figure 56. **Ce qui est tout à fait remarquable est l'adaptation de la base orthogonale à la géométrie de l'image avec des formes oscillantes dans les zones uniformes et qui respectent les contours.**

150. NDJE. extrait de Z. Kadkhodaie, F. Guth, E. P. Simoncelli, S. Mallat (2023-24) <https://arxiv.org/abs/2310.02557>.

151. NDJE. C'est une précision que S. Mallat m'a donné après son cours.

152. NDJE. c'est une métrique qui estime la similarité entre 2 images, *Peak Signal to Noise Ratio*.

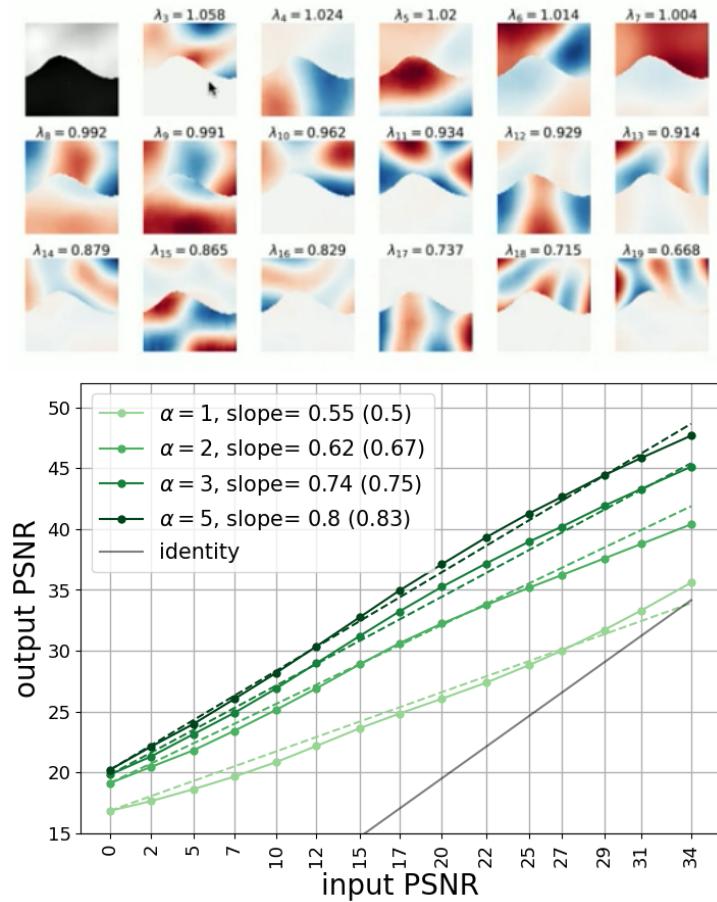


FIGURE 54 – En haut: l'image 53 (80×80) et les vecteurs propres de plus grandes valeurs propres. En bas: comparaison des courbes théoriques (traits tirés) et des mesures expérimentales (points) de l'évolution du PSNR de l'image débruitée (output) en fonction du PSNR de l'image bruitée (input).

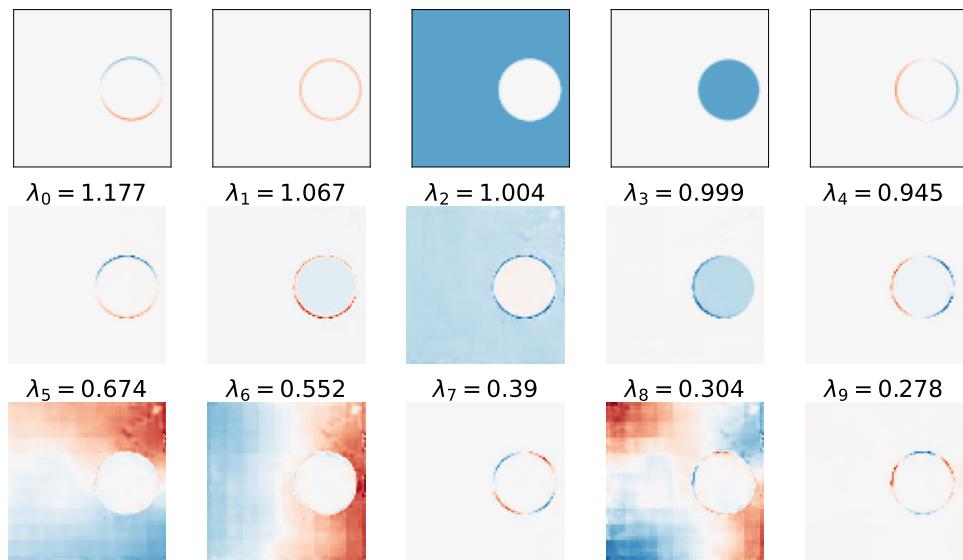


FIGURE 55 – En haut: les 5 vecteurs propres de l'espace des images constituées d'un disque de rayon et position variables, tout comme les couleurs (niveaux de gris) à l'intérieur et à l'extérieur du disque. En bas: les 5 premiers vecteurs propres du Hessien obtenus lors du débruitage qui sont identiques à ceux attendus, et ensuite quelques composantes trouvées par le réseau afin de compléter la base dans l'espace des images qu'il a utilisé lors de l'entraînement.

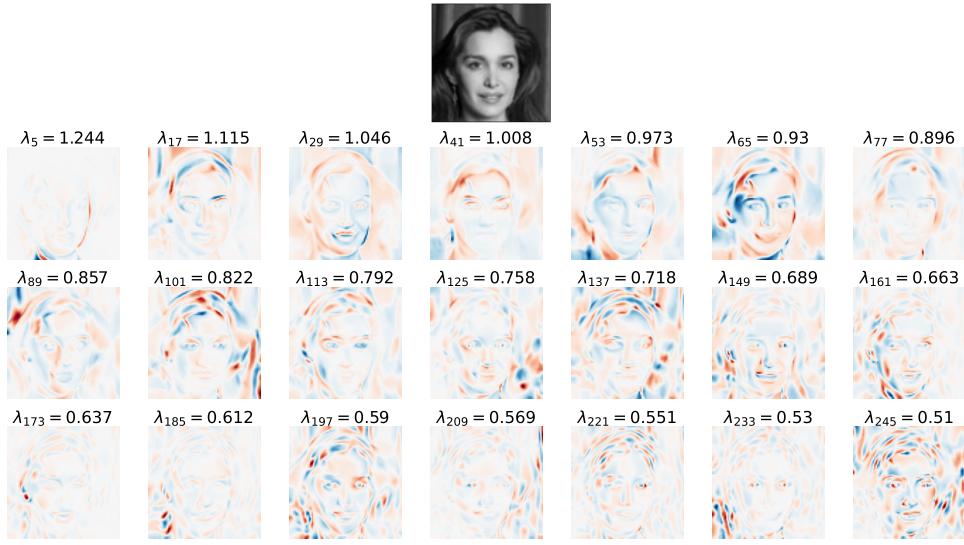


FIGURE 56 – Vecteurs propres du Hessian trouvés par le réseau entraîné avec des images du visage bruitées.

9.5.4 Réflexions sur le modèle génératif par score diffusion

On peut constater que le réseau de neurones (ex. U-Net) permettant d'estimer le score, permet d'estimer la probabilité sous-jacente aux images de visages, chambres-à-coucher, etc. Il est **capable de généraliser**, mais comme montré à la section 9.4, **il faut beaucoup d'exemples**. Par exemple, dans les expérimentations numériques montrées dans les sections précédentes, il a fallu 100,000 images pour être certain que le modèle générera réellement de nouvelles images 80×80 . Donc on a un problème, soit dans le cas où il faut pouvoir disposer de réalisations de plus grande taille, soit dans le cas où on ne dispose pas de beaucoup d'images furent-elles de petites dimensions. En particulier, dans le deuxième cas, on va mécaniquement aboutir à un **overfitting**, c'est-à-dire que l'on est dans le registre de la **mémorisation**, ce qui n'est pas le but recherché.

Maintenant, même s'il faut beaucoup de données, leur nombre ne suit pas une loi exponentielle de la dimension, le modèle a en quelque sorte absorbé la structure des données. Quelle est-elle? Pour répondre à la question, il faut ouvrir la "boîte noire" (Fig. 22) et **comprendre l'architecture** qui est chargée de sens. Cette introspection est d'autant plus importante, nous dit S. Mallat, que lorsque l'on est confronté à un problème où les

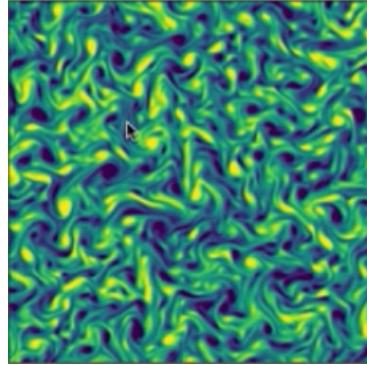


FIGURE 57 – Exemple d'image d'un fluide turbulent.

données sont limitées, il nous faut construire des modèles de basse dimension¹⁵³, et pour cela il faut comprendre les architectures.

9.5.5 Étude de cas: la turbulence

Le problème que nous propose S. Mallat est celui de la caractérisation de la turbulence. C'est un sujet dont A. N. Kolmogorov a jeté les bases en 1941 mais qui reste encore ouvert de nos jours. En fait la turbulence est le résultat de la dissipation à petites échelles de l'énergie injectée à grande échelle (effets de la viscosité à grand nombre de Reynolds). On veut donc pouvoir estimer la densité de probabilité sous forme d'énergie de Gibbs en considérant le phénomène comme s'il était à l'équilibre stationnaire (ce qui n'est pas à proprement parler exact). Quoiqu'il en soit la question est de savoir qu'elle est l'énergie d'un fluide turbulent en 2D? Lorsque l'on regarde une image d'un fluide turbulent (Fig. 57), on remarque les nombreuses volutes, filaments, et tourbillons, etc: **il y a de la géométrie**.

Maintenant par l'étude de ce phénomène, la Physique donne une piste **de réduction possible du nombre de degrés de liberté**: on est en présence d'une **structure multi-échelles** du type de celle de la figure 4. On peut donc réduire de d interactions directes à $O(\log d)$ termes multi-échelles. Le hic est que les groupes à chaque niveau d'échelles interagissent entre-eux, ce qui fait de ce problème, un problème non trivial.

153. NDJE. Voir par ex. les réflexions de la section 2.1.3 du cours de 2020.

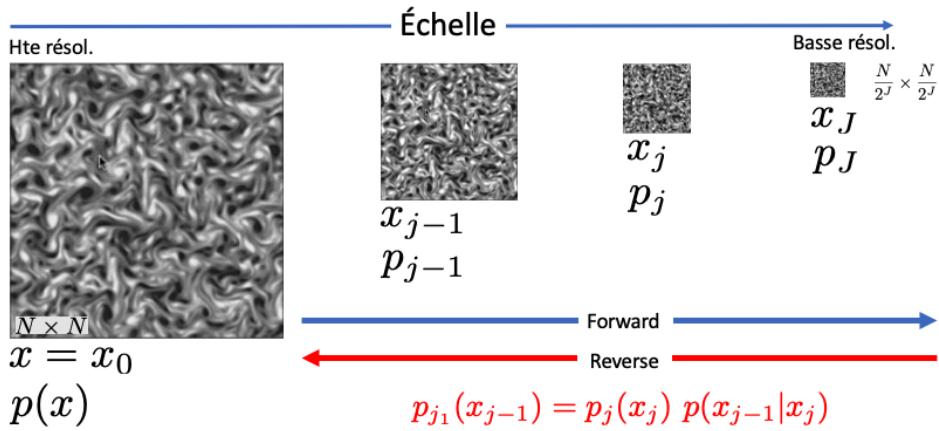


FIGURE 58 – Schéma d'évolution à travers les échelles: dans le sens "forward" où l'on moyenne et sous-échantillonne progressivement l'image d'origine, et dans le sens "reverse" où l'on inverse la chaîne de Markov en commençant la chaîne par un échantillon de la probabilité de basse dimension (facile), et en utilisant la relation entre les probabilités.

Pour résoudre ce type de problème, la Physique (1970) a développé le schéma suivant (Fig.58):

Forward : dans un premier temps l'image (ou le champ d'une manière générale) x est progressivement moyennée et sous-échantillonnée pour obtenir des versions x_j à différentes échelles de résolution $j \in 0, J$. On peut en même temps regarder l'évolution de la distribution de probabilité $p_j(x_j)$ (par exemple en ajustant un modèle linéaire en énergie discuté à la section 6.5).

Reverse : ensuite on veut générer de nouvelles images à l'échelle originale. Pour cela, on remarque qu'à l'échelle J , la taille de l'image est réduite à $(N/2^J)^2$ qui peut être de quelques pixels seulement. Donc, en basse dimension il est *a priori* facile d'échantillonner une telle distribution. Puis, ayant une cascade à l'étape "forward", on peut calculer la cascade inverse. En fait, on a

$$p_{j-1}(x_{j-1}) = p_j(x_j) p(x_{j-1}|x_j) \quad (262)$$

Au lieu d'avoir une chaîne de Markov qui évolue avec le niveau de bruit comme pour la méthode de Score Diffusion, ici la chaîne évolue selon les échelles. C'est

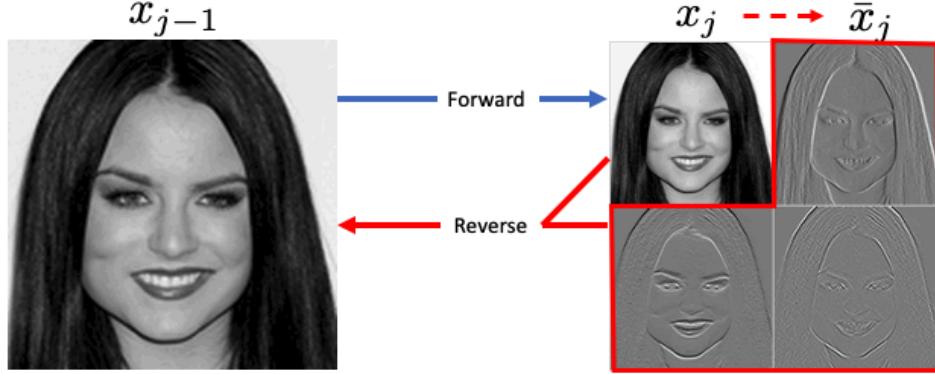


FIGURE 59 – Quand passe une image x_{j-1} à travers une transformation en ondelettes 2D (mode forward) on obtient non seulement l'image moyennée et sous-échantillonée x_j mais aussi les composantes de "détails" notées \bar{x}_j . En mode "reverse" où à partir d'une image x_j on veut produire une image x_{j-1} de plus grande résolution, il faut pouvoir construire les composantes de "détails".

un peu la philosophie derrière le U-Net.

Le point à éclaircir est comment on va de l'échelle $j - 1$ à l'échelle j ? C'est là où la transformée en ondelettes va nous guider, car pour construire une image de résolution $j - 1$ (plus haute) à partir d'une image de résolution j (plus faible), il nous faut **rajouter des "détails"** notés \bar{x}_j (Fig. 59). Or, la décomposition de x_{j-1} en (x_j, \bar{x}_j) est orthogonale, donc on va partir du principe que

$$p(x_{j-1}|x_j) = p(\bar{x}_j|x_j) \quad (263)$$

Et tout comme nous avions vu pour le mouvement brownien (Sec. 4.3, Fig. 26) que l'on peut obtenir **des interdépendances à longue portée entre les variables par des dépendances locales mais hiérarchiques (multi-échelles)**, il en va de même dans le cas présent. Maintenant, pour modéliser $p(\bar{x}_j|x_j)$ on peut utiliser un modèle d'énergie de Gibbs linéaire (Sec. 6.5):

$$p_{\theta_j}(\bar{x}_j|x_j) = Z_j^{-1} e^{\theta_j^T \Phi(\bar{x}_j, x_j)} \quad (264)$$

où $(\theta_j)_j$ est de basse dimension. Cette modélisation est un enjeu de recherche tout à fait ouvert nous dit S. Mallat.

Une fois l'optimisation faite le chemin est tout tracé. On part d'un échantillon x_J de p_J , puis on génère les imagettes de détails \bar{x}_J selon $p(\bar{x}_J|x_J)$, on reconstruit alors x_{J-1} . De nouveau on génère les imagettes \bar{x}_{J-1} selon $p(\bar{x}_{J-1}|x_{J-1})$, et l'on peut reconstruire x_{J-2} , et ainsi de suite. Comme cela on aboutit à un échantillon x_0 (ou x) de la probabilité

$$p(x) = p_{\theta_J}(x_J) \prod_{j=J}^1 p_{\theta_j}(\bar{x}_j|x_j) \quad (265)$$

Avec de telles modélisations on peut procéder¹⁵⁴ à la génération de champs non gaussiens par des modèles par ondelettes comme ceux de la figure 6 (*turbulence, cosmic web, etc.*).

9.6 Conclusion de cette année

L'enjeu est de **comprendre la structure**, car on a envie de savoir pourquoi se fait-il que les réseaux de neurones arrivent à calculer des scores et donc à générer des images de visages, chambres-à-coucher etc, et dans bien d'autres domaines. Pour ces problèmes, l'on a pas la moindre base d'un modèle théorique comme on peut l'obtenir avec des modèles de Physique où l'on a un voire plusieurs siècles de réflexions qui ont donné ces bases sur lesquelles on peut s'appuyer. Mais si les réseaux arrivent à capturer ces structures, comme déjà énoncé c'est que peut être il y a un espoir que cette structure n'est pas si complexe que cela, et **il y a peut être quelques principes directeurs** qui pourraient nous mettre sur le chemin pour **attaquer le problème mathématiquement**.

Dans le cas de la **génération par diffusion de score**, on tente à montrer que l'on peut effectivement être dans un mode de généralisation (et non pas uniquement de la mémorisation des données). Mais, les algorithmes reposent sur une "**boîte noire**" à savoir de réseau débruiteur (ex. U-Net). Donc si on veut aller plus loin dans la compréhension, il faut ouvrir la-dite boîte.

Probablement, un des principes directeurs, nous dit S. Mallat, c'est sans sans doute la **notion de hiérarchie**. Dans le cas d'une image comme montré dans la section précédente,

154. NDJE. voir Sihao Cheng, Rudy Morel, Erwan Ally, Brice Ménard, Stéphane Mallat (2023), <https://arxiv.org/abs/2306.17210>, ainsi que Florentin Guth, Etienne Lempereur, Joan Bruna, Stéphane Mallat (2023) <https://arxiv.org/abs/2306.00181>, et les travaux référencés.

il s'agit des échelles. **Dans le cas d'une image**, c'est beaucoup plus simple, car la topologie est bien définie, la notion de dilatation/contraction est bien définie, ce qui donne accès à **la notion d'échelle**. Par contre, imaginons le **cadre d'une entreprise** où les acteurs interagissent pour faire aboutir un projet, il n'y a pas de notion d'échelle naturelle, mais certainement **la notion de hiérarchie est présente**. Dans ce cadre l'article de H. Simons des années 60¹⁵⁵ donne une sorte d'explication à cette universalité des systèmes hiérarchiques: il y a une recherche dynamique de stabilité. Mais, ce faisant une structure hiérarchique ce n'est pas une structure d'arbre simple vertical avec des niveaux horizontaux déconnectés qui correspondraient au stade de décomposition en ondelettes par exemple. Mais, pour reprendre le cas d'une entreprise, **on a envie qu'à chaque niveau de la hiérarchie il y ait de la communication**. De même, dans les images de visages (turbulence, etc) on voit qu'il y a des **interactions à longue portée à toutes les échelles**.

Ces connexions posent à nouveau des questions ouvertes en mathématique. Et nul doute que des pistes pour comprendre les principes qui sous-tendent ces structures hiérarchiques "horizontales" sont nécessaires. Car les systèmes hiérarchiques sous forme d'arbre simple ont été testés dans tout un tas de domaines: ex. en linguistique avec les grammaires de Noam Chomsky¹⁵⁶ mais force est de constater que cela ne marche pas et ce n'est pas cela qui a débouché sur les grands modèles de langage (voir Introduction de cette année); de même en imagerie, les analyses multi-résolutions (ondelettes) permettent de réaliser des traitements mais fondamentalement ce n'est pas cela qui a permis les générateurs d'images, idem en audio etc.

D'une certaine manière si on exquise un parallèle, les structures auto-similaires sont des formes simples de fractals mais il y a des objets bien plus complexes dans la famille des fractals, il en va de même des hiérarchies: celles en arbre simple nous ont permis de faire un pas vers la compréhension des structures hiérarchiques, mais il y a encore du chemin que nous continuerons à explorer l'an prochain.

155. NDJE. Voir Cours 2020 Sec. 3.2

156. NDJE. Voir Cours 2019 Sec. 2.3.1

10. NDJE. Quelques ajouts personnels

Dans cette section, je vais ajouter quelques éléments en lien avec le cours:

- au sujet d'une technique très utilisée d'échantillonnage, nommé HMC.
- au sujet de ce que l'on appelle les *Normalizing Flows*

10.1 HMC: Hybride/Hamiltonian Monte Carlo

Dans cette section, je vais introduire une méthode Monte Carlo (notée HMC) qui veut tenter de répondre à la question: peut-on réduire le temps nécessaire pour qu'une chaîne de Markov puisse donner un lot d'échantillons indépendants issues d'une distribution cible $\pi(x)$. Dans la suite je ne fait qu'effleurer quelques notions et je renvoie par exemple vers l'article de Michael Betancourt¹⁵⁷ (2018) pour plus de détails à ce sujet.

On a vu au cours de cette année que l'on cherche à échantillonner des probabilités telles que $\pi(x) > 0$, et que l'on peut les modéliser sous forme de distribution de Gibbs:

$$\pi(x) = Z^{-1} e^{-U(x)} \Leftrightarrow -\log(\pi(x)) = U(x) + Cte \quad (266)$$

On supposera par la suite que l'on sait calculer le score, soit $\partial U(x)/\partial x$ ¹⁵⁸.

Dans la technique HMC, on ajoute une variable conjuguée à x , que l'on note opportunément p et l'on forme la fonction (hamiltonien)

$$H(x, p) = U(x) + E_c(p) \quad (267)$$

avec $E_c(p) = \|p\|_2^2/2$ représentant une sorte d'*énergie cinétique* du système quand $U(x)$ en représente l'énergie potentielle. Mais il ne faut pas pousser plus loin l'analogie. Les équations dynamiques du système sont alors identiques à celle de la Mécanique classique

157. Michael Betancourt, *A Conceptual Introduction to Hamiltonian Monte Carlo*, <https://arxiv.org/pdf/1701.02434.pdf>

158. Notons au passage que les codes de différentiation automatique (tensorflow/torch/JAX) utilisés pour le ML permettent d'obtenir des codes efficaces et stables si l'expression de $\nabla_x U(x)$ n'est pas connue analytiquement.

de Hamilton:

$$\dot{x} = \frac{\partial H}{\partial p} = \frac{\partial E_c}{\partial p} = p \quad \dot{p} = -\frac{\partial H}{\partial x} = -\frac{\partial U}{\partial x} \quad (268)$$

où la notation \dot{a} signifie une dérivée "temporelle" de la variable a ($\dot{a} = da/dt$), et quand le temps est discrétisé $\dot{a} = a(t+1) - a(t)$.

L'algorithme HMC se décline autour de la version simplifiée suivante pour discuter des ingrédients. Partant d'un état x_i d'une chaîne pour obtenir l'état x_{i+1} , on procède ainsi:

Étape 1: tirage aléatoire de $p_i \sim \mathcal{N}(0, 1)$; puis l'on initialise $p_{new} = p_i$ et symétriquement $x_{new} = x_i$;

Étape 2: itération de n_{steps} étapes d'intégration des équations du mouvement suivant la méthode dite de l'algorithme *leapfrog* en anglais ou *sauter-mouton* en français:

$$\begin{cases} p_{new} &= p_{new} - \varepsilon \times \frac{1}{2} \frac{\partial U(x)}{\partial x} |_{x=x_{new}} \\ x_{new} &= x_{new} + \varepsilon \times p_{new} \\ p_{new} &= p_{new} - \varepsilon \times \frac{1}{2} \frac{\partial U(x)}{\partial x} |_{x=x_{new}} \end{cases} \quad (269)$$

Notons que $-\nabla_x U(x)$ agit comme une "force", et donc si ε est une sorte d'élément infinitésimal de temps dt , alors on comprends que $dp = -dt \times \nabla_x U(x)$ (le 1/2 est là pour tenir compte du fait que l'on découpe l'intervalle en temps en deux parties égales. De même p (si la masse est unité) est une vitesse, donc $dx = dt \times p$. Donc, on comprends bien que l'on intègre les équations du mouvement. On note la mise à jour du gradient en cours de route, ce qui interviendra plus tard;

Étape 3: puis, on renverse le moment $p_{new} = -p_{new}$ (pas nécessaire en pratique, nous verrons);

Étape 4: enfin, on procède comme pour l'algorithme de Metropolis selon la probabilité d'acceptation du nouvel état (x_{new}, p_{new}) :

$$p_{acc} = \min \left\{ 1, r = \frac{\tilde{P}(x_{new}, p_{new})}{\tilde{P}(x_i, p_i)} \right\} \quad (270)$$

avec

$$\tilde{P}(x, y) = \exp\{-H(x, p)\} = \exp\{-U(x)\} \exp\{-E_c(p)\} \quad (271)$$

Cela se fait par tirage uniforme d'un nombre u dans l'intervalle $[0, 1]$. Si $u < r$ on

accepte x_{new} comme valeur pour x_{i+1} , sinon on prend x_i . Remarquons le découplage qu'il y a entre les deux variables x et p . Notons aussi, que l'on ne garde pas p_{new} par la suite.

La méthode soulève la question suivante: pourquoi utiliser des équations d'Hamilton? Les arguments sont les suivants:

1. la *réversibilité* temporelle de la dynamique hamiltonienne est importante pour la réversibilité de la chaîne de Markov (voir après);
2. ensuite la *conservation* de l'Hamiltonien au cours du temps fait qu'en principe $r = 1$, et donc favorise l'acceptation d'un nouvel état de la chaîne. Cependant, en pratique la conservation de H n'est pas exacte ne serait-ce que par les imperfections de l'algorithme d'intégration;
3. la proposition du nouvel état est guidée par p_{new} , or ce dernier pointe dans la *direction de plus forte densité* de probabilité, ce qui va dans le bon sens;
4. enfin, la dynamique hamiltonienne *conserve les volumes dans l'espace des phases*, c'est le théorème de Liouville. Cette propriété est cruciale pour pouvoir utiliser la méthode de Metropolis (et non la version Hastings) (étape 4) afin d'accepter ou non un nouvel état de la chaîne.

Maintenant, deux questions viennent naturellement à l'esprit: pourquoi l'algorithme (HMC) converge vers $\pi(x)$, et pourquoi a-t'on cette structure étrange de l'algorithme de saute-mouton, notamment le renversement du moment à la fin? La première question réfère à la notion de *detailed balance* qui compare les probabilités de processus inverses $A \rightarrow B$ et $B \rightarrow A$ (Voir Sec. 11)¹⁵⁹. Donc, voyons pourquoi nous avons l'égalité suivante qui garantira que $\pi(x)$ est la mesure invariante de l'algorithme:

$$\pi(x)p_t(x \rightarrow y) = \pi(y)p_t(y \rightarrow x) \quad (272)$$

Ici, $p_t(x \rightarrow y)$ est la probabilité de transition de la position x vers la position y . Que vaut cette probabilité de transition dans le cas de la méthode HMC?

159. Notons qu'il existe des versions de HMC qui ne satisfont pas ce critère, car il n'est pas nécessaire pour obtenir une distribution stable vers laquelle l'algorithme converge: voir par exemple <https://arxiv.org/pdf/1409.5191.pdf>.

Voyons cela: si l'on part d'un état (x_0, p_0) au bout de T étapes (à la manière de discréétisation du temps) on aboutit à l'état (x_T, p_T) . Cependant, la dynamique hamiltonienne est déterministe, donc il existe deux fonctions f_1 et f_2 telles que

$$x_T = f_1(x_0, p_0) \quad p_T = f_2(x_0, p_0) \quad (273)$$

Or, la *réversibilité* des équations dynamiques nous dit que si l'on part de l'état $(x_T, -p_T)$ alors en T étapes on retombe sur l'état initial (x_0, p_0) . Cela impose alors que

$$x_0 = f_1(x_T, -p_T) \quad p_0 = f_2(x_T, -p_T) \quad (274)$$

ce qui confère alors une correspondance bi-univoque entre (x_0, p_0) et (x_T, p_T) d'une part et $(x_T, -p_T)$ d'autre part. Supposons à présent qu'il n'y ait qu'un unique moment $p^{(x,y)}$ tel que l'on ait entre les positions x et y le mapping suivant $y = f_1(x, p^{(x,y)})$, et en même temps notons $p_y = f_2(x, p^{(x,y)})$. La réversibilité nous dit alors que $x = f_1(y, -p_y)$ et $p^{(x,y)} = f_2(y, -p_y)$. Ainsi, il n'y a qu'un moment $p^{(y,x)}$ qui relie $(x, p^{(x,y)})$ et $(y, p^{(y,x)})$ c'est $p^{(y,x)} = -p_y$. Donc, $p_t(x \rightarrow y) = \Pi(p^{(x,y)})$ ($\Pi(p) \propto e^{-E_c(p)}$) car seule la donnée du moment fait bouger la valeur de la position, et d'une manière similaire on identifie $p_t(y \rightarrow x) = \Pi(-p_y)$ ¹⁶⁰. Maintenant, on peut démontrer la relation 272:

$$\begin{aligned} \pi(x)p_t(x \rightarrow y) &= \pi(x)\Pi(p^{(x,y)}) \\ &= \frac{1}{Z}e^{-U(x)-E_c(p^{(x,y)})} = Z^{-1}e^{-H(x,p^{(x,y)})} \\ &= Z^{-1}e^{-H(y,p_y)} \quad (\text{H} = \text{Cte}) \\ &= Z^{-1}e^{-U(y)-E_c(p_y)} \\ &= \pi(y)\Pi(p_y) \\ &= \pi(y)\Pi(-p_y) \quad (\text{E}_c(\text{p}) \text{ sym. } \text{p} \leftrightarrow -\text{p}) \\ &= \pi(y)p_t(y \rightarrow x) \end{aligned} \quad (275)$$

Ayant cette propriété de balance détaillée, on sait que $\pi(x)$ est la mesure invariante vers laquelle l'algorithme va converger.

Maintenant concernant la seconde question, inspectons la partie de l'algorithme

160. J'en conviens cela beaucoup de p , π , etc.

correspondant à l'étape 2. C'est une forme discrétisée des équations du mouvement, elles rappellent l'algorithme d'Euler par exemple. Le plus intrigant est l'étape 3, à quoi sert-elle? Reprenons les étapes d'une itération dans le sens + de la progression de (x_0, p_0) vers un nouvel état (l'étape intermédiaire est noté avec 1/2):

$$\begin{aligned}\hat{p}_{1/2}^+ &= p_0^+ - \varepsilon/2 \nabla V(x_0^+) \\ \hat{x}_1^+ &= x_0^+ + \varepsilon \hat{p}_{1/2}^+ \\ \hat{p}_1^+ &= \hat{p}_{1/2}^+ - \varepsilon/2 \nabla U(x_1^+)\end{aligned}\tag{276}$$

et finalement avec l'étape 3, on passe donc de (x_0^+, p_0^+) à $(\hat{x}_1^+, -\hat{p}_1^+)$. Si l'on reprend la même démarche mais en partant de $(x_0^-, p_0^-) = (\hat{x}_1^+, -\hat{p}_1^+)$ dans le sens inverse alors il vient pour le premier demi-step du moment:

$$\begin{aligned}\hat{p}_{1/2}^- &= p_0^- - \varepsilon/2 \nabla U(x_0^-) \\ &= -\hat{p}_1^+ - \varepsilon/2 \nabla U(\hat{x}_1^+) \\ &= -\left(\hat{p}_{1/2}^+ - \varepsilon/2 \nabla U(x_1^+)\right) - \varepsilon/2 \nabla U(x_1^+) \\ &= -\hat{p}_{1/2}^+\end{aligned}\tag{277}$$

On va bien dans le sens inverse... Concernant la position

$$\begin{aligned}\hat{x}_1^- &= x_0^- + \varepsilon \hat{p}_{1/2}^- \\ &= \hat{x}_1^+ - \varepsilon \hat{p}_{1/2}^+ \\ &= x_0^+ + \varepsilon \hat{p}_{1/2}^+ - \varepsilon \hat{p}_{1/2}^+ \\ &= x_0^+\end{aligned}\tag{278}$$

donc on revient à la position originale. Finalement, en effectuant le second demi-step du moment, il vient

$$\begin{aligned}\hat{p}_1^- &= p_{1/2}^- - \varepsilon/2 \nabla U(x_1^-) \\ &= -\hat{p}_{1/2}^+ - \varepsilon/2 \nabla U(x_0^+) \\ &= -p_0^+ + \varepsilon/2 \nabla U(x_0^+) - \varepsilon/2 \nabla U(\hat{x}_0^+) \\ &= -p_0^+\end{aligned}\tag{279}$$

Ainsi, après l'étape du renversement du moment, on retrouve également la valeur originale p_0^+ . Donc, l'étape de renversement du moment permet de rendre compte de la réversibilité des lois de la dynamique hamiltonienne qui est à la base de la propriété de convergence de l'algorithme. Cependant, en pratique l'expression de l'énergie cinétique étant quadratique en p , changer le signe n'a aucune incidence sur la probabilité d'acceptation de la nouvelle position, donc on peut en pratique supprimer l'étape 3.

Voilà, vous comprenez un peu mieux, je l'espère, la mécanique de l'algorithme HMC. Cependant, il y a deux (hyper)-paramètres n_{steps} et ε dont les valeurs relèvent de la pratique et influencent expérimentalement la vitesse de convergence de l'algorithme. Le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2003/Monte_Carlo_Sampling.ipynb donne une implémentation simple en python (`numpy`) de cet algorithme. On sent bien intuitivement que ε doit être petit, car il caractérise le pas de discrétisation des équations de Hamilton. Mais le nombre de steps (`n_steps`) reste un problème: trop petit la chaîne a une évolution de type marche aléatoire (trop de stochasticité), trop grand on utilise des ressources inutilement, mais surtout cela peut faire en sorte que la trajectoire de la chaîne boucle sur elle-même ce qui génère des échantillons "proche" les uns des autres comme si la chaîne stagnait, voire même on peut faire perdre tout simplement les propriétés de convergence de la chaîne (non-ergodicité). Donc la détermination de n_{steps} est délicate et demande des simulations préparatoires et beaucoup de pratique. Ce point dur rendait la pratique de la méthode HMC difficile, enfin avant la mise au point de l'algorithme NUTS¹⁶¹ en 2011.

NUTS, pour No-U-Turn Sampler, construit une procédure qui assure la distance maximale entre la position x_i et la potentielle nouvelle position x_{new} . On réalise que si l'on part de x_i , la dynamique hamiltonienne nous donne que la variation de $\|x_i - x(t)\|^2$ satisfait l'équation

$$C(t) = \frac{1}{2} \frac{d}{dt} \|x(t) - x_i\|^2 = (x(t) - x_i) \cdot \frac{dx(t)}{dt} = (x(t) - x_i) \cdot p(t) \quad (280)$$

Ainsi, on peut se rendre compte au fur et à mesure de la progression dans l'algorithme saute-mouton si le terme de droite devient négatif, car alors cela indiquerait un point de rebroussement de la trajectoire. Cependant, si on ne simule que la partie où $C(t) >$

161. Matthew D. Hoffman et Andrew Gelman (2011), *The No-U-Turn Sampler: Adaptively Setting Path Lengths in Hamiltonian Monte Carlo*, arXiv:1111.4246, <https://arxiv.org/abs/1111.4246>

0 on a un problème avec la réversibilité. Pour palier ce problème, l'algorithme NUTS introduit une variable auxiliaire et construit un arbre binaire de mouvements forward (direction + utilisé plus haut) et backward (−) afin de donner sur une profondeur 2^n des propositions de mouvement. Mais l'algorithme complet est complexe donc je vous renvoie à l'article des auteurs. Pour un utilisateur, c'est là où la librairie `numpyro`¹⁶² va être utile. Notons que `numpyro` n'est pas la seule librairie sur le marcher pour traiter de la génération de chaînes de Markov par la méthode HMC/NUTS. L'usage d'autres librairies est par exemple discuté dans ce post datant de 2018: <https://mattpitkin.github.io/samplers-demo/pages/samplers-samplers-everywhere/>.

Le notebook https://github.com/jecampagne/cours_mallat_cdf/blob/main/2003_Monte_Carlo_Sampling_2.ipynb montrent des usages de NUTS tout comme HMC avec `numpyro`.

10.2 Normalizing Flows

Dans le cours nous avons vu la méthode du Score Diffusion qui est élaborée autour d'une équation différentielle stochastique d'Ornstein-Uhlenbeck que l'on sait inverser. Dans son mode forward, on transforme la *v.a* X qui suit la distribution $p(x)$ dont on a des échantillons $(x_i)_i$, progressivement en une *v.a* X_T qui suit la distribution $p_T(x)$ en ajoutant du bruit blanc gaussien pour faire en sorte que p_T soit un pur bruit. En même temps on apprend le score $\nabla_x \log p_t(x)$ avec le bruit σ_t . Ce score est utilisé lors du mode backward afin de générer de nouveaux échantillons de $p(x)$ à partir de tirages de $p_T(x)$ simple à réaliser.

Ceci dit, dans la littérature vers 2018 a été utilisée une autre technique appelée *Normalizing Flows*. Je ne vais pas faire l'étude détaillée ici vous pouvez vous référer par exemple à la revue de Papamakarios et al. (2019-21)¹⁶³.

Le principe général est relativement simple car il s'agit ni plus ni moins d'opérer un

162. Site web: <http://num.pyro.ai>; Du Phan, Neeraj Pradhan and Martin Jankowiak, *Composable Effects for Flexible and Accelerated Probabilistic Programming in NumPyro* <https://arxiv.org/pdf/1912.11554.pdf>

163. <https://arxiv.org/abs/1912.02762>

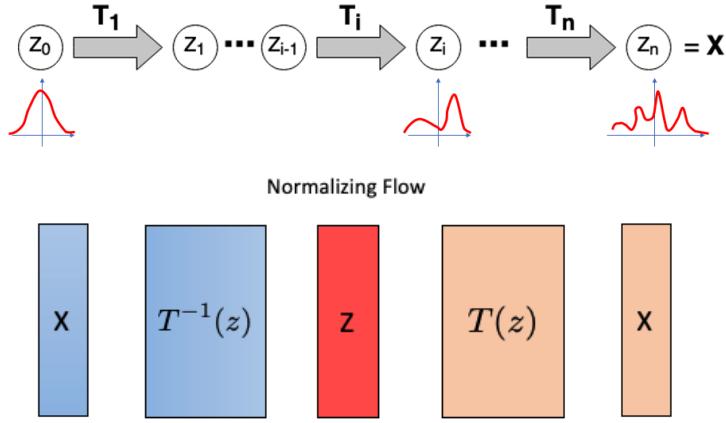


FIGURE 60 – Haut: Schéma qui montre qu'à partir d'une distribution simple de Z_0 en opérant des transformations T_i inversibles et différentiables, l'on aboutit à un la distribution de X laquelle est beaucoup plus complexe (par ex. multimodale). Bas: Schéma qui met en lumière une architecture en deux parties.

changement de variable tel que si Z est *pdf* $p_z(Z)$:

$$X = T(Z) \quad \text{avec} \quad Z \sim p_z(Z) \quad (281)$$

La *pdf* $p_x(X)$ de X s'obtient alors selon¹⁶⁴

$$Z = T^{-1}(X); \quad p_x(X) = p_z(Z) |\det J_T(Z)|^{-1} = p_z(T^{-1}(X)) |\det J_{T^{-1}}(X)| \quad (282)$$

Ceci n'est possible que si T est **inversible** et si T et T^{-1} sont des **applications différenciables**, il faut en effet pouvoir calculer le Jacobien J qui mesure le changement de volume. Au passage, **cela fixe la dimensionnalité de Z à celle de X** . Dans les librairies (et la littérature), la transformation T est souvent appelée un **bijector**, avec son mode **forward** et son mode **backward** quand on applique respectivement T ou T^{-1} .

^{164.} $p_x(a)$ signifie que la *pdf* de x est évaluée avec la variable a , ex. $p_x = \mathcal{N}(\mu, \sigma^2)$ alors $p_x(a) \propto \exp\{-(a - \mu)^2 / 2\sigma^2\}$.

L'intérêt est que l'on peut composer les transformations de telle sorte que

$$T = T_1 \circ T_2 \circ \cdots \circ T_n \quad (283)$$

avec toutes les T_i inversibles et différentiables. Primo l'inverse T^{-1} est facilement calculable selon $T^{-1} = T_n^{-1} \circ T_{n-1}^{-1} \circ \cdots \circ T_1^{-1}$ et secundo le déterminant du jacobien de T est le produit des déterminants des jacobiens des n transformations ($Z_0 = Z$ et $Z_n = X$):

$$Z_i = T_i(Z_{i-1}) \quad \log |\det J_T| = \sum_i \log |\det J_{T_i}(z_{i-1})| \quad (284)$$

On retrouve la notion de transport de probabilité dont il est question dans le cours. On peut ainsi à partir d'une distribution de base $p_z(z)$ simple (ex. une gaussienne multivariée de d variables $\mathcal{N}(0, \mathbf{1}_d)$) pour modéliser des distributions de X complexes comme schématisé sur la figure 60. Notons que le calcul des déterminants (Eq. 284) peut être couteux sauf si les transformations sont spéciales, en particulier si leurs jacobiens sont des **matrices triangulaires**.

Connaissant p_z et disposant d'un lot des échantillons $(x_i)_{i \leq N}$, que l'on suppose être issues de tirages *iid* de la loi p_x , alors seule la transformation T (et ses composantes dans un schéma itératif) est inconnue. Mettons que T soit un élément d'une famille de transformations dont il faut déterminer les paramètres ϕ afin de la définir pleinement, alors on procède en *maximisant la vraisemblance* que les $(x_i)_i$ soit effectivement issus de la loi p_x , c'est-à-dire en minimisant la *loss*

$$\mathcal{L}(\phi) = - \sum_{i=1}^N \log p_x(x_i) \quad (285)$$

En fait:

- pour *l'entraînement*, on a uniquement besoin de T^{-1} car on utilise la seconde égalité de l'équation 282 pour calculer $p_x(X)$ et $\mathcal{L}(\phi)$.
 - si l'on veut *générer de nouveaux échantillons* de X à partir de p_x obtenue, il vaudrait mieux avoir accès à la transformation T directement pour utiliser les relations 281.
- Évidemment, on cherche des familles de transformations qui permettent (1) de calculer la *pdf* p_x et (2) de générer des nouveaux échantillons le plus efficacement possible, en particulier sans avoir à inverser la transformation.

L'article de Papamakarios et al. donne une série de méthode pour implémenter ces transformations comme les *modèles auto-régressifs* qui partent de l'équation 2

$$p(x) = p(x_1)p(x_2|x_1)p(x_3|x_1, x_2)\cdots = p(x_1) \prod_{j=2}^d p(x_j|x_1, \dots, x_{j-1}) \quad (286)$$

pour modéliser

$$p(x_i|x_{1:i-1}) = p(x_i; h_i) \quad h_i = C_i[x_{1:i-1}] \quad (287)$$

avec des réseaux de neurones.

Donc, la question qui vient à l'esprit est: quelle est la différence avec la méthode de Score Diffusion? *La différence principale¹⁶⁵ avec un normalizing flow classique est que le chemin entre les données et la gaussienne est fixé, ce n'est pas une fonction arbitraire que l'on apprend mais il est donné par l'équation d'Ornstein-Uhlenbeck avec le score. Cela permet d'avoir un objectif pour chaque temps: on apprend le chemin petit bout par petit bout, au lieu d'avoir uniquement une loss sur le point d'arrivée comme l'entraînement par maximum-likelihood des normalizing flows.* Et force est de constater que s'il a existé des modèles génératifs à base de normalizing flows¹⁶⁶, les grands modèles génératifs sont à base de score.

165. Extrait d'un échange avec F. Guth.

166. Voir par ex. Kingma et Dhariwal (2018) <https://arxiv.org/pdf/1807.03039.pdf> (tous deux d'OpenAI).