COLLEGE MATHS, TELECOMS
DOCTORAL INFORMATIQUE, SIGNAL
BRETAGNE SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

THÈSE DE DOCTORAT DE

L'INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE RENNES

ÉCOLE DOCTORALE N° 601
Mathématiques, Télécommunications, Informatique, Signal, Systèmes, Électronique
Spécialité : *AST*

Par
Taha YASSINE

Hybrid approaches for data processing

Bridging machine learning and analytical models

Thèse présentée et soutenue à Rennes, le 09 avril 2024
Unité de recherche : IETR
Thèse N° : 24ISAR 05 / D24 - 05

Rapporteurs avant soutenance :

Lina MROUEH — Professeure des Universités, Institut Supérieur d'Electronique de Paris (ISEP)
Maxime GUILLAUD — Directeur de recherche INRIA, Laboratoire CITI, Lyon

Composition du Jury :

Président :	Raphaël COUTURIER	Professeur des Universités, IUT de Belfort-Montbéliard, Université de Franche-Comté
Examinateurs :	Cédric HERZET	Chargé de recherche INRIA, Professeur associé à l'ENSAI, Rennes
	Charlotte LANGLAIS	Directrice d'Etudes, Institut Mines-Télécom Atlantique (IMT-A)
	Lina MROUEH	Professeure des Universités, Institut Supérieur d'Electronique de Paris (ISEP)
	Maxime GUILLAUD	Directeur de recherche INRIA, Laboratoire CITI, Lyon
Dir. de thèse :	Matthieu CRUSSIÈRE	Professeur des Universités, INSA Rennes
Co-dir. de thèse :	Luc LE MAGOAROU	Maître de Conférences, INSA Rennes

Invité(s) :

Stéphane PAQUELET — Responsable du laboratoire Intelligence Artificielle, IRT b<>com



O Son of Adam, you are nothing but a number of days. Whenever each day passes, then a part of you has gone.

Al-Hasan Al-Basri رحمه الله

ACKNOWLEDGEMENTS

I would like to express my gratitude to all those who have supported and guided me throughout these four years.

First and foremost, I would like to thank my two advisors, Luc Le Magouaro and Matthieu Crussière, for their invaluable support and encouragement throughout this journey. Luc introduced me to the field of wireless communications when it was still uncharted territory for me, and I am truly thankful for his patience and guidance. Matthieu assisted me in navigating the challenges of academic research and provided reassurance whenever it was needed. I sincerely appreciate the time they dedicated to mentoring me and the freedom they allowed me to explore and develop my ideas.

I extend my thanks to the jury members Cédric Herzet, Charlotte Langlais, Lina Mroueh, Maxime Guillaud, and Raphaël Couturier for their thorough evaluation and validation of this manuscript, and for their constructive comments.

I am grateful to my colleagues at b<>com who were all welcoming and kind. I would like to particularly thank my lab manager, Stéphane Paquelet, for providing an intellectually stimulating environment, for the long discussions we had that enriched this research, and for the continuous feedback he provided. I would also like to thank the other members in my lab: Mathieu, Baptiste, Matthieu, Amine, Pierre, Malo, and many others I met during my time there. I'm also grateful to the colleagues I met at IETR during the brief time I spent there.

My sincere thanks go to all my friends who supported me throughout this journey. I'm grateful to my friends—then PhD students, now doctors—Antoine, Julien, Enzo, Olivier, Gaël, and William for the shared experiences and mutual support that have made this challenging path more enjoyable. To Charafdine, Anis, Yassir, Aymane, Nezar, and Mouad, thank you for the

long discussions and the emotional support. And to Ilyas, thank you for your support since our school days.

I would also like to thank my friends whom I consider my second family, my soul brothers, who have welcomed me with open arms: Hicham, Houdaifa, Gwendal, Soufyan, Éliaz, Joseph, Adil, Mohamed, Mohamed R, Ivan, Idriss, Younes, and many others. I'm truly fortunate to have met you, and my spiritual journey was elevated by your presence. And a very special thanks to Adil, who welcomed me when I first arrived in France, and continued to support me, and without whom I would not have been able to be where I am today.

Finally, I am deeply grateful to my wonderful family. My grandparents have been a blessing to many, myself included. My uncles, aunts, and cousins have offered constant support and care. Above all, I thank my parents, who dedicated their lives to their children, for their unconditional love, and my brother for his continual presence and encouragement.

SYNTHÈSE EN FRANÇAIS

Dans les années 1890, Guglielmo Marconi, inventeur italien et ingénieur en électricité, réalisa une série d'expériences et de démonstrations avant-gardistes dans le domaine de la transmission sans fil. Notamment, en 1899, il réussit à établir une liaison sans fil entre l'Angleterre et la France passant au-dessus de la Manche. Ses travaux lui valurent de nombreux honneurs et prix internationaux, dont un prix Nobel en 1909. Ces expériences initiales donnèrent la cadence pour plus d'un siècle d'innovations continues, façonnant la trajectoire de la technologie et des communications de manière inédite. Les découvertes de cette époque permirent d'établir les bases du monde interconnecté dans lequel nous vivons aujourd'hui, avec les technologies sans fil au cœur de notre vie moderne.

À la fin du 20ème siècle, nous assistâmes à la naissance de la première génération de téléphones mobiles, ou 1G, qui permettaient de passer des appels en itinérance. Le passage à la 2G fut le passage aux communications dites numériques et l'introduction des messages par texte, révolutionnant ainsi notre manière de communiquer. L'introduction de la 3G quant à elle permit l'accès à Internet, nous donnant la possibilité de naviguer sur le web et d'accéder aux courriels depuis nos appareils portables. La 4G, qui a suivi, a encore accéléré les vitesses de transfert de données, rendant le streaming vidéo de haute qualité et l'utilisation d'applications sur les smartphones monnaie courante.

Aujourd'hui, au 21ème siècle, nous sommes entrés dans l'ère de la 5G, la cinquième génération de technologie de communication mobile. La 5G, actuellement en cours de déploiement dans le monde entier, promet des vitesses de données sans précédent, une ultra-faible latence et la capacité de connecter une vaste gamme d'appareils et de systèmes, inaugurant l'ère de l'Internet des objets (IoT) et transformant encore davantage notre manière de vivre, de tra-

vailler et de communiquer.

Ces générations successives de systèmes de communication sans fil, toutes construites sur les fondations posées par les expériences initiales de Marconi, ont redéfini notre façon de nous connecter les uns aux autres et au monde. Elles ont rendu possible un monde où l'information est instantanément accessible, et nos appareils sont interconnectés de manières autrefois inimaginables, tout cela grâce à l'esprit d'innovation constant dans les technologies sans fil.

Mais l'histoire ne s'arrête pas là, et les prochaines générations de systèmes sans fil sont déjà à l'horizon, promettant des capacités encore plus transformatrices et des opportunités pour un avenir connecté avec des applications dans la robotique, l'e-santé, la réalité augmentée et autre. Les systèmes à entrées multiples et à sorties multiples (MIMO) constituent un exemple de technologie habilitante. Ils consistent en des dispositifs de communication équipés de plusieurs antennes travaillant ensemble pour transmettre et recevoir de l'information. Bien que certains produits commerciaux utilisent déjà cette technologie, par exemple la majorité des routeurs Wifi modernes, c'est en augmentant le nombre d'antennes à des dizaines, voire des centaines, que les vrais bénéfices deviennent clairs. De tels systèmes sont appelés systèmes MIMO massifs. Leur potentiel réside dans leur capacité à desservir un plus grand nombre d'appareils tout en maintenant ou même en augmentant le débit de données grâce au multiplexage spatial. Ces systèmes sont également loués pour leur efficacité énergétique nettement améliorée. En effet, en exploitant le principe de superposition de la propagation des ondes dans une technique appelée « beamforming », l'énergie peut être concentrée dans une direction particulière au lieu d'être rayonnée dans toutes les directions, comme c'est le cas avec les systèmes d'antennes classiques. Cependant, ces avantages sont accompagnés de défis, le plus important étant l'estimation du canal, un sujet exploré en détail dans ce manuscrit.

La communication sans fil n'est pas le seul domaine qui a connu un développement fulgurant ; le domaine de l'Intelligence Artificielle (IA) a également vécu progrès extraordinaire. L'IA, la science de la conception de machines capables de simuler l'intelligence humaine, a considérablement évolué ces dernières décennies. En 1997, Deep Blue a battu Gary Kasparov, le champion du monde

d'échecs en titre. En 2016, Lee Sedol, un joueur de classe mondiale, a perdu contre un ordinateur, AlphaGo, dans une partie de Go. En 2020, AlphaFold 2 a réalisé un score sans précédent dans la compétition de prédiction de pliage de protéine CASP. Ce ne sont là que quelques-unes des nombreuses occasions où les machines ont su démontrer leurs incroyables capacités.

L'IA est passée d'un concept théorique à une réalité pratique. L'apprentissage automatique, l'apprentissage profond et les réseaux de neurones ont ouvert la voie à des machines capables de comprendre, de raisonner et de prendre des décisions, souvent avec une précision étonnante. Ces avancées ont trouvé des applications dans divers secteurs, allant de la santé et de la finance aux véhicules autonomes et au traitement naturel du langage.

L'un des développements les plus remarquables en IA est l'émergence d'algorithmes puissants basés sur les données capables d'analyser de vastes quantités d'informations, de reconnaître des modèles et de faire des prédictions avec une précision remarquable. Cette progression a permis la création d'assistants virtuels intelligents, de systèmes de recommandation et de robots autoapprenants, améliorant l'efficacité et la commodité dans notre vie quotidienne. De plus, l'IA a trouvé sa place dans la recherche de pointe, allant de la découverte de médicaments et de la génomique à la modélisation du climat et à l'exploration spatiale. Un nombre croissant d'experts s'accordent à dire qu'atteindre un niveau d'intelligence équivalent à celui des humains pourrait devenir une réalité dans un avenir proche, suscitant des discussions sur l'éthique de l'IA, le remplacement d'emplois et les limites des capacités de l'IA.

Le progrès de l'IA est lié à de nombreux facteurs, mais a initialement été entravé par un manque de ressources informatiques suffisantes. Prenons l'exemple des réseaux de neurones : bien qu'ils existent depuis des décennies en tant que concept théorique, les ordinateurs de l'époque n'étaient pas assez puissants pour démontrer leurs réelles capacités d'apprentissage. De plus, les données nécessaires à leur entraînement étaient rares et l'espace de stockage limité. Avec le temps, le domaine de la microélectronique a rattrapé la demande et les machines sont devenues de plus en plus puissantes. Au début des années 2010, les premiers signes du véritable potentiel de l'apprentissage automatique ont commencé à se manifester, et le domaine a connu une renaissance complète. Aujourd'hui, le manque de ressources informatiques en-

trave une fois de plus les progrès, mais cette fois à une échelle totalement différente. Mais la solution consiste-t-elle toujours à courir derrière un matériel plus puissant mais également plus gourmand en énergie ? Ou existe-t-il un moyen d'exploiter la puissance des machines apprenantes sans céder à leurs besoins incessants en données et en énergie ? Dans un monde aux ressources limitées et où le changement climatique fait fureur, trouver des réponses à ces questions est essentiel.

L'apprentissage profond basé sur des modèles, en tant que paradigme émergent, est une tentative de résoudre certains de ces défis. Il découle de l'observation que, alors que la résolution classique de problèmes repose sur des modèles mathématiques artisanaux de complexité limitée, les réseaux de neurones contournent cette limitation en apprenant les règles complexes sous-jacentes à un problème donné à partir de quantités massives de données. Les deux approches se situent aux extrémités opposées d'un spectre performance contre besoins en données. Est-il possible d'atteindre un juste milieu ? Est-il possible de concevoir des systèmes capables d'apprendre à résoudre des problèmes complexes à partir d'une quantité limitée de données ? L'apprentissage profond basé sur des modèles propose de guider la conception de réseaux de neurones en s'appuyant sur la connaissance humaine du problème à traiter. Par exemple, si un algorithme classique fonctionne bien pour une configuration donnée du problème mais peine lorsque cette configuration devient plus complexe, alors au lieu de remplacer l'ensemble de l'algorithme par un réseau de neurones générique, l'approche basée sur des modèles consisterait à s'inspirer de l'algorithme original pour concevoir un réseau doté d'une architecture plus efficace. Bien sûr, la question qui demeure est comment procéder ? De nombreux travaux récents explorent ce concept et offrent des réponses préliminaires (Shlezinger et al. 2020).

Dans le contexte des communications sans fil, l'émergence de nouvelles technologies s'accompagne de nouveaux défis que l'IA pourrait aider à résoudre (Ali et al. 2020). Actuellement, les systèmes sans fil dépendent fortement de modèles mathématiques pour concevoir les différents blocs du réseau, dont certains sont sous-optimisés et inefficaces. L'IA pourrait aider à remplacer ou à augmenter ces blocs pour répondre aux besoins des prochaines générations de systèmes de communication. En outre, l'IA pourrait aider à

tirer parti du potentiel offert par les nouvelles technologies et le matériel pour dévoiler de nouvelles capacités, comme le positionnement avancé et les tâches en découlant pouvant en bénéficier.

Ce manuscrit est structuré en deux parties distinctes, chacune comprenant plusieurs chapitres. Voici ici un résumé de chacun de ces chapitres.

Première partie : État de l'Art

La partie I introduit les différentes notions et outils nécessaires à la compréhension de chaque contribution.

Chapitre 1 : Modélisation du canal

Dans le chapitre 1, nous voyons comment la propagation des ondes électromagnétiques, qui est à la base des systèmes modernes de communication sans fil, peut être modélisée mathématiquement. Différents phénomènes physiques en jeu (par exemple, l'affaiblissement, la propagation multitrajet, etc.) peuvent être intégrés dans le modèle selon le besoin de précision ou de simplicité. Les modèles qui tentent d'être réalistes paient le prix en complexité et tendent à nécessiter des considérations non triviales. Le concept de canal de propagation est également introduit. Il permet d'écrire le modèle du système (MIMO) comme une relation entrée/sortie entre le signal d'entrée $\mathbf{X}$ et la sortie $\mathbf{Y}$:

$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{N},$$

où $\mathbf{N}$ représente le bruit. Toute la modélisation mathématique de l'environnement de propagation est contenue dans $\mathbf{H}$. Pour simuler un scénario de propagation, deux approches prévalent : les modèles déterministes et les modèles statistiques. Des exemples des deux sont donnés. Enfin, dans un contexte de communication, l'estimation du canal est généralement une étape critique. Certains des algorithmes d'estimation classiques sont présentés. Ceux-ci se sont historiquement avérés robustes, mais à mesure que la dimension du canal augmente et que les environnements de propagation évoluent vers des configurations plus complexes, la tâche d'estimation devient difficile.

Chapitre 2 : Apprentissage machine

Puisque l'élaboration artisanale de modèles de canaux devient fastidieuse, et parfois même irréaliste, s'appuyer sur des approches basées sur les données a gagné en popularité au sein de la communauté. Un intérêt croissant pour l'apprentissage automatique dans les communications a été observé. Le chapitre 2 sert d'introduction à la matière. Le principe général est d'apprendre une fonction désirée en minimisant une mesure d'erreur donnée sur l'ensemble d'entraînement. Ces différents composants peuvent être implémentés de nombreuses manières, comme démontré dans les exemples donnés. Une manière en particulier se démarque : l'apprentissage profond qui a connu un grand succès ces dernières années. Au moment de la rédaction de ce manuscrit, le monde a les yeux rivés sur les modèles de langage de grande taille pas si anciens comme ChatGPT et Gemini pour leurs impressionnantes capacités à imiter les humains. Comme toute application d'apprentissage profond, la clé de leur succès a été l'énorme quantité de données disponibles et les ressources de calcul. Les algorithmes d'apprentissage automatique tentent d'apprendre les relations sous-jacentes qui structurent les données. L'hypothèse de la variété que nous explorons, par exemple, suggère que les données que l'on retrouve dans la nature peuvent généralement être expliquées par un nombre plus réduit de variables. Les méthodes de réduction de la dimensionnalité visent à exploiter cette hypothèse pour extraire une représentation compressée des données, distillant finalement leurs caractéristiques les plus saillantes. La dernière partie du chapitre explore l'utilisation de l'apprentissage profond pour l'estimation du canal.

Chapitre 3 : Cartographie du canal

Le chapitre 3, où le concept de cartographie du canal est introduit, se situe à la croisée des deux premiers chapitres. Bien que cela ne soit pas immédiatement évident, apprendre une carte est finalement une manière d'extraire les caractéristiques les plus importantes d'un canal de propagation. Dans ce contexte, l'hypothèse de la variété suggère que, dans un environnement (quasi-)statique, le canal est régi par quelques paramètres relatifs à l'emplacement de l'émetteur. Puisque ces paramètres sont généralement de très faible di-

mension, il est logique d'essayer de les extraire en utilisant les méthodes de réduction de la dimensionnalité présentées dans le chapitre 2. Parmi le grand nombre d'approches explorées dans la littérature, celles basées sur des mesures de distance dans l'espace des vecteurs du canal sont particulièrement intéressantes. En effet, un certain nombre d'algorithmes d'apprentissage de la variété sont basés sur la matrice de distance des entrées. Trouver une mesure de distance adéquate est donc d'une importance cruciale.

Deuxième partie : Contributions

La partie II entre dans les détails de chaque contribution.

Chapitre 4: Estimation du canal basée sur un modèle

Dans le chapitre 4, mpNet est présenté. En tant que réseau neuronal basé sur un modèle, il est entraîné pour apprendre une correspondance entre un signal reçu et le vecteur du canal associé. En particulier, l'estimation LS facile à calculer est utilisée en entrée. De cette manière, mpNet peut être vu comme un modèle de débruitage peu coûteux. En effet, par rapport à d'autres réseaux génériques, le modèle est conçu pour être léger en calcul. Cela est dû à sa structure qui est obtenue par le dépliage de l'algorithme *matching pursuit*. mpNet est initialisé à partir de la connaissance initiale imparfaite des paramètres du système et est particulièrement efficace pour apprendre à se corriger lui-même. Par exemple, nous démontrons qu'il peut s'adapter avec le temps à des anomalies telles que des antennes endommagées ou le vieillissement.

Chapitre 5 : Cartographie du canal basée sur une modèle

Un autre réseau basé sur un modèle est présenté dans le chapitre 5, cette fois dans le contexte de la cartographie du canal. Il est basé sur la distance insensible à la phase introduite dans (Le Magoarou 2021a) et prend la forme d'un encodeur avec une structure hybride. Ce modèle bénéficie de trois idées principales : une initialisation intelligente, le calcul de la distance implicite-

ment dans le cadre des calculs du réseau, et l'entraînement en mode triplet. Ce dernier exploite le fait que les observations du canal sont généralement accompagnées d'horodatages et que deux échantillons proches dans le temps ont de fortes chances d'être proches dans l'espace également. Nous montrons comment la mise en œuvre de chaque idée individuellement améliore la qualité de la carte apprise.

Chapitre 6 : Paramétrisation système pour la cartographie du canal

Nous avons remarqué que la distance insensible à la phase a certaines limites. Dans le chapitre 6, nous identifions et étudions ces limites dans le contexte des systèmes MIMO OFDM. Nous voyons qu'elles proviennent de la nature périodique et oscillatoire de la distance. La notion d'identifiabilité est proposée et décrit une mesure de distance qui peut être utilisée de manière fiable pour apprendre une carte sans ambiguïtés. Sur la base de cette définition, des conditions sont dérivées sur les paramètres du système pour créer une configuration où la distance peut être utilisée de manière fiable. Lorsqu'elles sont mises en pratique, ces conditions améliorent considérablement la qualité de la carte finale.

Chapitre 7 : Application de la cartographie du canal

Dans le chapitre 7, la cartographie du canal est considérée du point de vue de l'application. Une fois qu'une carte est disponible, à quoi peut-elle servir ? Le beamforming est pris comme exemple. Dans (Le Magoarou, Yassine, et al. 2022b), un réseau de neurones est utilisé pour apprendre le précodeur optimal correspondant à une position d'entrée. La même approche est testée lors du remplacement des positions réels par des pseudo-positions sur une carte apprise. Nous montrons que les précodeurs appris sont de bonne qualité, même dans des conditions NLoS. Le "channel mapping" est également considéré. En effet, nous montrons également que la projection d'une observation du canal dans une BS donnée d'abord, puis la transmission de la pseudo-position à une BS différente pour le calcul du précodeur fonctionne bien. Cela peut être vu comme un moyen de compresser l'information et de ne transmettre qu'une

version distillée du canal aux BS cibles, éliminant le besoin d'une estimation du canal redondante.

Conclusion

Nous montrons que mpNet peut être vu comme un autoencodeur. Cela signifie que l'encodeur projette les vecteurs du canal dans un espace de faible dimension. Cet espace est cependant non structuré et la nature dynamique du réseau n'aide pas. Puisque le but est de produire une estimation du canal, la forme de l'espace latent n'est pas importante. Cependant, nous pouvons supposer que, en règle générale, avoir un espace "lisse" et régulier aiderait dans le processus de décodage, indépendamment de l'application qui s'ensuit. C'est le postulat de la cartographie du canal. Sachant que le canal est un produit direct de l'emplacement de l'émetteur, il est fort probable que l'espace de faible dimension que nous recherchons est lié à celui-ci, conformément à l'hypothèse de la variété. La cartographie du canal vise à intégrer cette connaissance a priori dans l'encodage. En somme, au lieu d'essayer d'apprendre une correspondance qui va directement de l'espace des vecteurs du canal à une sortie souhaitée, il est préférable de passer d'abord par un espace latent sous la forme d'une carte du canal, surtout lorsque la qualité d'une telle carte peut être quantifiée et améliorée.

Perspectives

Voici quelques perspectives qu'il serait intéressant d'explorer et qui permettraient de poursuivre le travail de cette thèse.

Apprentissage en ligne L'entraînement d'un modèle d'apprentissage automatique pour apprendre une tâche nécessite un ensemble de données d'entraînement. Alors que l'objectif principal de l'apprentissage non supervisé est d'éliminer cette exigence en trouvant des moyens d'apprendre uniquement à partir des données d'entrée, il permet également au modèle d'apprendre de manière continue. En effet, un système pourrait réutiliser les données qu'il collecte

pendant son fonctionnement pour réajuster les paramètres du modèle afin de s'adapter à un changement de distribution. Cela est démontré dans le chapitre 4. Ce principe peut être étendu à la plupart des applications de l'apprentissage non supervisé. Bien que l'idée de s'entraîner sur de nouvelles données soit simple, il n'est pas clair comment la mettre en œuvre de la manière optimale. L'entraînement doit-il être effectué dès que de nouveaux échantillons sont collectés ? Ou peut-être est-il préférable de ré-entraîner périodiquement le modèle ? De plus, comment gérer les anciennes données ? Le ré-entraînement doit-elle être limité uniquement aux nouvelles données ? Ou peut-être est-il préférable de maintenir l'ensemble des données à une taille fixe et de considérer un modèle premier entré premier sorti (FIFO) ? Toutes ces questions méritent d'être étudiées. Au-delà de l'entraînement traditionnel basé sur la descente de gradient, les modèles hybrides comme celui introduit dans le chapitre 5 bénéficient d'une initialisation basée sur les données disponibles à la création du modèle. En particulier, les échantillons retenus pour l'initiasation se sont avérés avoir un impact significatif sur la performance du modèle. Avec le temps qui passe, l'environnement change et les échantillons nouvellement collectés reflètent mieux ce changement. Il serait intéressant d'étudier si le remplacement progressif des éléments de la matrice par de nouveaux résulterait en une meilleure adaptation du système. Un article récent (Taner, Guillaud, et al. 2023) donne quelques premiers éléments de réponse sur cette question.

Entraînement de bout en bout Comme mentionné auparavant, la tâche d'apprendre une carte du canal en soi n'a pas beaucoup de sens si considérée seule. C'est lorsqu'elle est associée à une application en aval que les avantages de la carte deviennent clairs. Dans l'exemple que nous voyons où la cartographie du canal est appliquée au beamforming, le modèle est appris en deux étapes. D'abord, le modèle de cartographie du canal est entraîné, puis le décodeur de beamforming est entraîné sur les pseudo-positions obtenues. Bien que cette approche fonctionne bien, il pourrait être intéressant d'essayer un entraînement bout-à-bout. Cela pourrait être fait de deux manières : soit en le faisant pour toute la phase d'entraînement, soit en le considérant comme partie d'une étape ultérieure de "fine-tuning". Ce dernier semble plus adéquat

car il donne au modèle de cartographie l'opportunité d'apprendre un espace latent qui correspond à la définition d'une carte du canal. Cette approche devrait également être essayée sur différentes tâches en aval pour évaluer comment leur nature influence la performance de l'encodeur et sa carte résultante.

Compression CSI Bien que l'aspect dépendant de la localisation de la cartographie du canal soit important, il peut être étendu à un domaine d'application plus large. Comme nous le voyons, apprendre une carte revient à apprendre à réduire la dimensionnalité des vecteurs du canal. Autrement dit, cela peut être vu comme une tâche de compression. Cela peut être utile pour réduire la surcharge de retour d'information CSI dans l'estimation du canal (Yin et al. 2022) par exemple. Explorer les capacités de compression des modèles de cartographie et les comparer aux approches de compression pure pourrait apporter des informations cachées qui pourraient améliorer les deux mondes. Peut-être y a-t-il un compromis à trouver entre compresser le canal et conserver suffisamment d'informations pour la tâche ultérieure. Une autre possibilité est que compresser par la cartographie pourrait permettre de meilleurs niveaux de compression sans dégrader les performances sur l'objectif final.

Inspiration d'autres domaines Bien que l'apprentissage automatique et l'apprentissage profond aient le vent en poupe depuis un certain temps, force est de reconnaître que la communauté des communications sans fil tarde à s'y mettre. À première vue, cela peut sembler être une situation désavantageuse, mais en réalité, c'est une incroyable opportunité de s'inspirer du travail effectué dans les autres domaines qui ont une longueur d'avance. Par exemple, l'architecture des transformeurs proposée à l'origine pour la traduction de séquence à séquence (Vaswani et al. 2017) a été appliquée avec succès à un certain nombre de tâches différentes, même en dehors des tâches de modélisation du langage. Le mécanisme d'attention est particulièrement intéressant car il permet de comparer différentes parties de l'entrée, appelées "tokens", entre elles. Cette information est exploitée par le réseau de neurones pour décider efficacement de ce qu'il faut produire. Après un examen plus approfondi, il apparaît que l'encodeur hybride introduit dans le chapitre 5 présente

des similitudes avec le transformateur. En effet, la première étape consistant au calcul de la corrélation entre un vecteur du canal d'entrée h et la colonne de la matrice D pourrait être vue comme le calcul de la corrélation entre les clés et les requêtes du transformeur. Ensuite, l'opération de seuillage dur est une alternative à l'opération softmax et enfin la somme pondérée des éléments de Z est similaire à la somme pondérée des valeurs. Avec cela à l'esprit, certains autres astuces du transformeur pourraient être intéressantes à essayer sur l'encodeur hybride. La première idée qui vient à l'esprit est d'essayer d'avoir plusieurs têtes avec différentes matrices D et Z pour chaque tête. Ce n'est qu'un exemple de la façon dont il peut être bénéfique de s'inspirer de ce qui a été fait dans les autres domaines de l'apprentissage automatique. La littérature est une mine d'or d'idées pour aborder des problèmes complexes, et il est important d'en tirer parti.

TABLE OF CONTENTS

Synthèse en français **iii**

Table of Contents **xv**

Introduction **1**

Context 1
Thesis outline 6
Contributions 8

I State of the art **11**

1 MIMO Channel modelling and estimation **12**

1.1 MIMO Channel models 13
1.1.1 General considerations 13
1.1.2 Input/output model 15
1.1.3 Deterministic models 22
1.1.4 Statistical models 24
1.2 MIMO Channel estimation 26
1.2.1 Least squares channel estimation 27
1.2.2 LMMSE channel estimation 28
1.2.3 Sparse recovery for channel estimation 29
1.3 Conclusion 31

2 Machine learning **32**

2.1 General principle 33
2.2 Popular algorithms 35
2.2.1 Linear regression 35
2.2.2 k-nearest neighbors 36

2.2.3 Support vector machine 37
2.3 Deep learning 39
2.3.1 Neural networks 40
2.3.2 Gradient descent 43
2.3.3 Backpropagation 44
2.3.4 Model-based deep learning 44
2.4 Dimensionality reduction 46
2.4.1 Nonparametric DR 47
2.4.2 Parametric DR 55
2.5 Machine learning-based channel estimation 60
2.6 Conclusion 61

3 Channel charting **62**
3.1 Manifold hypothesis for channel charting 63
3.2 Problem formulation 64
3.3 Performance measures 65
3.3.1 Trustworthiness 66
3.3.2 Continuity 66
3.3.3 Kruskal's stress 67
3.4 Channel charting approaches 67
3.4.1 General approaches 67
3.4.2 Distance-based approaches 69
3.5 Applications 73
3.5.1 Localization 73
3.5.2 Pilot allocation 74
3.5.3 Beam management 75
3.6 Conclusion 75

II Contributions **77**

4 Model-based deep learning for channel estimation **78**
4.1 Problem formulation 79
4.1.1 System settings 79
4.1.2 Physical model 81

4.2 Motivation: imperfect models . . . 82
4.3 Proposed approach: mpNet . . . 84
4.3.1 Unfolding matching pursuit . . . 84
4.3.2 Training mpNet . . . 87
4.3.3 Computational complexity . . . 88
4.3.4 Choosing the stopping criterion . . . 90
4.4 Experimental validation . . . 91
4.4.1 Fixed SNR . . . 92
4.4.2 Varying SNR . . . 95
4.4.3 Anomaly detection and recovery . . . 96
4.4.4 From ULA to UPA . . . 102
4.5 Conclusion and perspectives . . . 103

5 Model-based deep learning for channel charting 106
5.1 Problem formulation . . . 107
5.2 Distance measure . . . 107
5.3 Hybrid encoder . . . 108
5.4 Triplet loss . . . 109
5.5 Experiments . . . 111
5.5.1 Scenarios . . . 111
5.5.2 Triplet dataset construction . . . 111
5.5.3 Results . . . 112
5.6 Conclusion and perspectives . . . 116

6 System parametrization for channel charting 118
6.1 Problem formulation . . . 119
6.1.1 System and channel model . . . 119
6.1.2 Channel charting . . . 120
6.1.3 Identifiability . . . 121
6.2 Limitations of channel charting with a ULA . . . 124
6.3 UCA are better than ULA for channel charting . . . 130
6.4 Practical consequences of the theoretical findings . . . 136
6.5 Experiments . . . 137
6.5.1 Practical influence of system parameters . . . 137
6.5.2 ULA vs. UCA . . . 138

6.5.3 Realistic scenario 140
6.6 Conclusion and perspectives 144

7 Channel charting for cell-free precoding 145
7.1 Problem formulation 146
7.1.1 Setting 146
7.1.2 Objective 146
7.2 Proposed method 147
7.2.1 Inference 147
7.2.2 Training 148
7.2.3 Potential applications 150
7.3 Experiments 151
7.3.1 Simulation settings 151
7.3.2 Implementation details 152
7.3.3 Compared variants 153
7.3.4 Results 153
7.4 Conclusion and outlook 154

Conclusion and Perspectives 157
7.5 Conclusion 157
7.6 Perspectives 160

III Appendices 165

A Chapter 6's appendix 166
A.1 Proof of Lemma 6.1 166
A.2 Proof of Lemma 6.2 166
A.3 Proof of Lemma 6.3 167

Bibliography 171

INTRODUCTION

Context

In the 1890s, Guglielmo Marconi, an Italian inventor and electrical engineer, made a series of pioneering experiments and demonstrations of wireless transmission. Notably, in 1899, he established a wireless link between England and France across the English Channel. His work earned him numerous international honors and awards, including a Nobel prize in 1909. These initial experiments set the pace for more than a century of continuous innovations, shaping the trajectory of technology and communications in unprecedented ways. The discoveries made during this time laid the foundation for the interconnected world we now inhabit, with wireless technology at the forefront of modern life.

In the late 20th century, we witnessed the birth of the first generation of mobile phones, or 1G, which allowed people to make calls on the move. The leap to 2G brought digital communication and text messaging, revolutionizing the way we communicate. The subsequent introduction of 3G brought mobile internet access, enabling us to browse the web and access emails on our handheld devices. 4G, which followed, further accelerated data transfer speeds, making high-quality video streaming and app usage on smartphones commonplace.

Today, in the 21st century, we have entered the era of 5G, the fifth generation of mobile communication technology. 5G, currently in the process of being deployed worldwide, promises unprecedented data speeds, ultra-low latency, and the ability to connect an extensive range of devices and systems, ushering in the era of the Internet of Things (IoT) and further transforming the way we live, work, and communicate.

These successive generations of wireless communication, all building upon

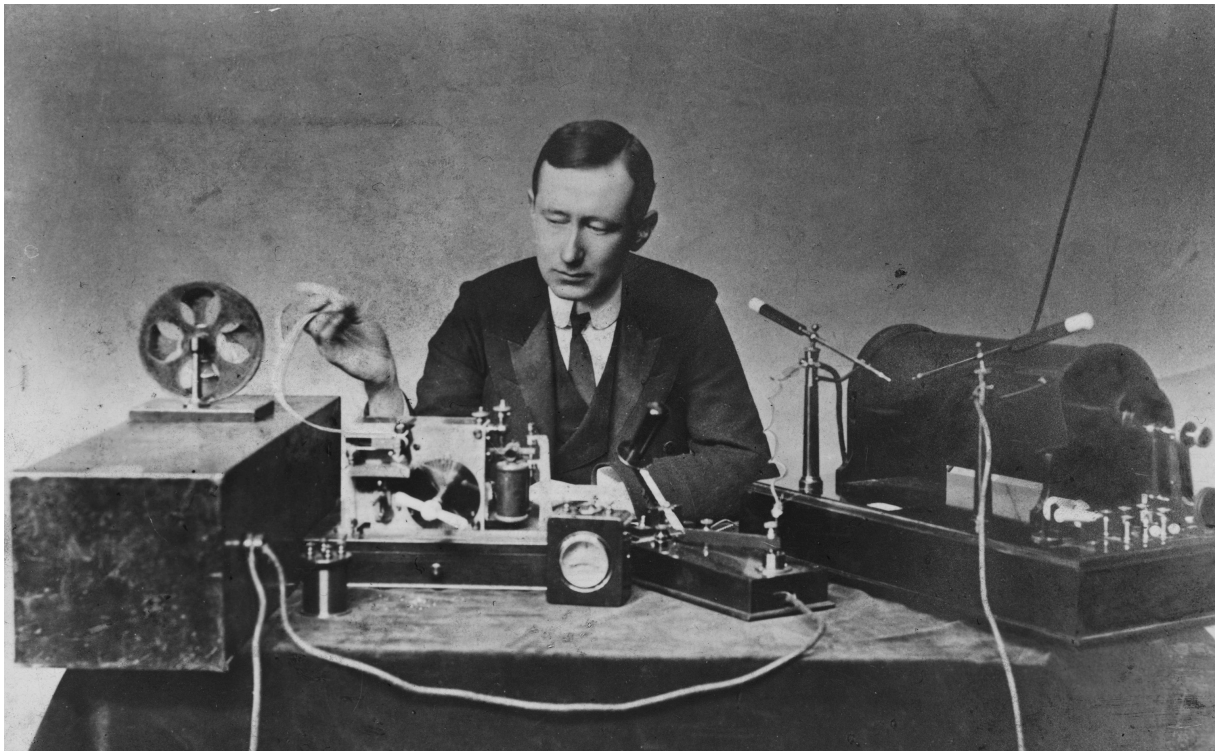


Figure 1 – Guglielmo Marconi demonstrating equipment he used in his first long-distance radio transmissions in the 1890s. Source: Wikipedia.

the foundation set by Marconi's initial experiments, have redefined the way we connect with each other and the world. They have enabled a world where information is instantly accessible, and our devices are interconnected in ways that were once unimaginable, all thanks to the ongoing spirit of innovation in wireless technology.

But the story doesn't stop here, and the next generations of wireless systems are already on the horizon, promising even more transformative capabilities and opportunities for a connected future with applications to robotics, eHealth, augmented reality, and more. One such example of enabling technologies is multiple input multiple output (MIMO) systems that consist of having communication devices equipped with multiple antennas working together to transmit and receive information. Although some commercial products are already making use this technology, e.g., the majority of modern Wifi routers, it is when scaling up the number of antennas to dozens or even hundreds that the real benefits become clear. Such systems are referred to as massive

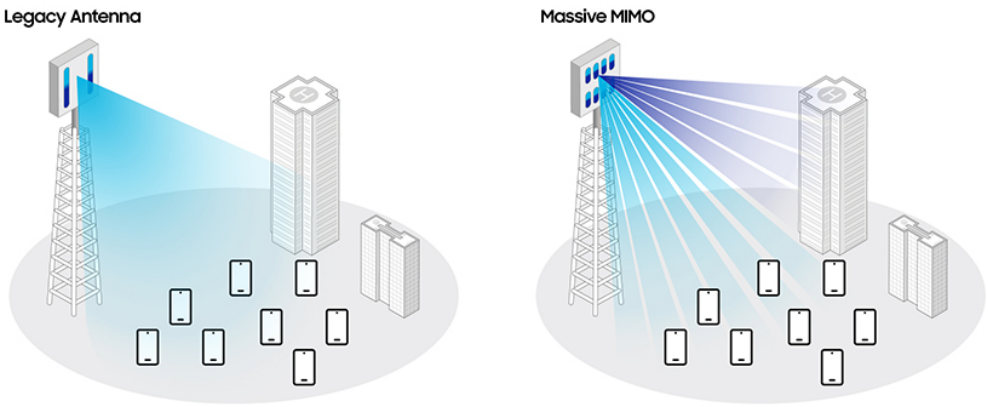


Figure 2 – Traditional radio transmission compared to beamforming. Source: (Samsung Electronics 2020)

MIMO systems. Their potential lies in their ability to serve a larger number of devices while maintaining or even increasing the data rate thanks to spatial multiplexing. Such systems are also praised for their significantly improved energy efficiency. Indeed, by exploiting the superposition principle of wave propagation in a technique called *beamforming*, energy can be focused towards a particular direction instead of radiated in every direction, as is the case with classical antenna systems (figure 2). However, these benefits come with challenges, the most significant being channel estimation, a topic that will be further explored in the rest of this manuscript.

Wireless communications is not the only field that has witnessed a fulgurant development; the realm of Artificial Intelligence (AI) has also experienced an extraordinary journey of progress. AI, the science of creating machines that can simulate human intelligence, has evolved significantly in recent decades. In 1997, Deep Blue defeated Kasparov, the reigning chess world champion. In 2016, Lee Sedol, a world-class player, lost to a computer, AlphaGo, in a game of Go. In 2020, AlphaFold 2 achieved an unprecedented score in the CASP protein folding prediction competition. These are a few of the many times machines showed their incredible capabilities.



Figure 3 – Lee Sedol playing against AlphaGo. Source: The Guardian.

AI has gone from being a theoretical concept to a practical reality. Machine learning, deep learning, and neural networks have paved the way for machines to understand, reason, and make decisions, often with astonishing accuracy. These advancements have found applications in various sectors, from healthcare and finance to autonomous vehicles and natural language processing.

One of the most remarkable developments in AI is the emergence of powerful, data-driven algorithms that can analyze vast amounts of information, recognize patterns, and make predictions with remarkable precision. This progress has enabled the creation of intelligent virtual assistants, recommendation systems, and self-learning robots, enhancing efficiency and convenience in our daily lives. Moreover, AI has found its place in cutting-edge research, from drug discovery and genomics to climate modeling and space exploration. A growing number of experts even believe that human-level intelligence could become a reality in a near future, sparking discussions about AI ethics, job displacement, and the boundaries of AI capabilities.

The progress of AI is tied to many factors, but it was initially hindered by a lack of sufficient computational resources. If we take the example of neural networks, although they have been around for decades as a theoretical concept, early computers were not powerful enough to demonstrate their real learning capabilities. Along with that, the data to train them was scarce and neither was the storage available. With time, the field of microelectronics caught up with the demand and machines became more and more powerful. By the early 2010s, the first signs of the real potential of machine learning started showing, and the field saw a complete rebirth. Today, the lack of computational resources is once again hindering progress, only this time on a whole new scale. But is the solution to always keep pushing for more powerful but also power hungry hardware? Or is there a way to harness the power of the learning machines without succumbing to their never-ending demands in data and energy? In a world with limited resources and where climate change is roaring, finding answers to these questions is essential.

Model-based deep learning as an emerging paradigm is an attempt at solving some of these challenges. It stems from the observation that while classical problem solving relies on hand-crafted mathematical models of limited complexity, neural networks bypass that limitation by learning the complex rules underlying a given problem from massive amounts of data. Both approaches are at opposite ends of a performance versus data needs spectrum. Is a middle ground attainable? Is it possible to design systems capable of learning to solve complex problems from a limited amount of data? Model-based deep learning proposes to guide the design of neural networks by relying on human acquired knowledge of the problem at hand. For example, if a classical algorithm works well for a given configuration of the problem but struggles when that configuration becomes more complex, then instead of replacing the whole algorithm with a generic neural network, the model-based approach would be to take inspiration from the original algorithm to conceive a network with a more efficient architecture. Of course, the question that remains is how to do so? Many recent works explore this concept and offer preliminary answers (Shlezinger et al. 2020).

In the context of wireless communications, the emergence of new technologies comes with new challenges that ML could help solve (Ali et al. 2020).

Currently, wireless systems heavily rely on mathematical models to design the different network blocks, some of which are underoptimized and inefficient. ML could help replace or augment such blocks to meet the needs of the next generations of communication systems. In addition, ML could help leverage the potential offered by new technologies and hardware to unveil new capabilities, like advanced positioning and subsequent tasks that can benefit from it.

In this manuscript, we explore the potential of model-based deep learning in wireless communications. While it can be applied to reshape and improve different parts of a wireless system, we first focus on channel estimation. The idea is to use domain knowledge to design a learning algorithm capable of accurate estimation of the channel without the need for a large amount of training data and compute resources. While channel estimation is a good candidate for model-based deep learning, we find that a more holistic approach can be more beneficial. We investigate the potential of model-based deep learning in the context of channel charting, a new paradigm that can be described as user localization in an unsupervised way. Indeed, since the propagation channel is intrinsically tied to the transmit location, channel charting aims at recovering a pseudo-location solely based on the former with no prior training on input-label pairs. We see that doing so leads to a compressed form of the channel that can be exploited in downstream location-dependent tasks like learning a precoder.

Thesis outline

The manuscript is structured into two distinct parts each comprised of a number of chapters.

— First part: State of The Art

  — Chapter 1 introduces the mathematical foundations of channel modelling along with some examples of usual channel models and datasets found in the literature. An introduction to channel estimation is also given, and classical estimation algorithms are developed and explained.

- Chapter 2 presents machine learning in general with examples of popular algorithms. The chapter then focuses on deep learning, introducing neural networks, how they work and what makes them so powerful. The final part of the chapter delves into the subject of dimensionality reduction which will be useful in subsequent chapters. Machine learning-based channel estimation is also briefly addressed.
- Chapter 3 explains the concept of channel charting as a pseudo-localization approach and how it relates to dimensionality reduction. Approaches found in the literature are explained and examples of applications are given.

— Second part: Contributions

- Chapter 4 introduces mpNet, a deep unfolded model-based neural network used for channel estimation. It is obtained from the unfolding of a sparse recovery algorithm and is capable of accounting for errors in the initial knowledge of system parameters like antenna gains and positions. Consequently, it is also capable of recovering from anomalies like broken antennas and aging.
- Chapter 5 presents another model-based network for channel charting this time. It is a hybrid model that takes the from of an encoder that combines some ideas from the literature to guide its design and training. In particular, it has an architecture that is based on a phase-insensitive channel distance, it is initialized in a smart way, and it is trained as a triplet network exploiting the timestamps that come with channel observations.
- Chapter 6 is dedicated to the analysis of the phase-insensitive distance and its limitations. A formal mathematical framework is developed in order to propose solutions to eliminate these limitations. The proposed solutions are of the form of guidelines on choosing adequate system parameters for channel charting.
- Chapter 7 presents an application of channel charting to beamforming. It is based on work on location based beamforming. The idea is to use pseudo-locations obtained from a learned channel chart instead of real locations (usually hard to obtain) to compute the optimal precoder. This can be useful for channel mapping, which con-

sists of separating the channel estimation from its subsequent exploitation so they can be conducted at different BSs and/or different frequencies for example.

To ease the reading experience, each contribution chapter can be read independently of the rest of the manuscript. Within each chapter, the addressed problem is succinctly recalled for clarity.

Contributions

Journal articles

— Taha Yassine and Luc Le Magoarou (July 2022), « mpNet: variable depth unfolded neural network for massive MIMO channel estimation », *in: IEEE Transactions on Wireless Communications* 21.7 (99), pp. 5703–5714

— Taha Yassine, Luc Le Magoarou, et al. (Oct. 2024), « Optimizing Multicarrier Multiantenna Systems for LoS Channel Charting », *in: IEEE Transactions on Wireless Communications* 23.10, pp. 14702–14714

Conference articles

— Taha Yassine, Luc Le Magoarou, Stéphane Paquelet, et al. (July 2022), « Leveraging triplet loss and nonlinear dimensionality reduction for on-the-fly channel charting », *in: 2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, Oulu, Finland, Oulu, Finland: IEEE, pp. 1–5

— Luc Le Magoarou, Taha Yassine, et al. (May 2022b), « Deep Learning for Location Based Beamforming with NLoS Channels », *in: ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, Singapore, Singapore, Singapore: IEEE, pp. 8812–8816

— Luc Le Magoarou, Taha Yassine, et al. (Oct. 2022a), « Channel charting based beamforming », *in: 2022 56th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, Pacific Grove, CA, USA: IEEE, pp. 1185–1189

— Mattia Merluzzi et al. (2023), « The Hexa-X Project Vision on Artificial Intelligence and Machine Learning-Driven Communication and Computation Co-Design for 6G », *in*: *IEEE Access* 11, pp. 65620–65648

— Taha Yassine, Luc Le Magoarou, Baptiste Chatelier, et al. (Aug. 2023), « Cartographie du canal par réduction de dimension et réseaux triplets », *in: 29° Colloque sur le traitement du signal et des images*, 2023-1381, Grenoble: GRETSI - Groupe de Recherche en Traitement du Signal et des Images, pp. 1165–1168

— Taha Yassine, Baptiste Chatelier, et al. (Oct. 2023), « Model-Based Deep Learning for Beam Prediction Based on a Channel Chart », *in: 2023 57th Asilomar Conference on Signals, Systems, and Computers*, IEEE, pp. 1636–1640

Deliverables

— Hamed Farhadi et al. (Apr. 2023), *Hexa-X Deliverable D4.3 - AI-driven communication & computation co-design solutions*

PART I

State of the art

CHAPTER 1

MIMO CHANNEL MODELLING AND ESTIMATION

In modern wireless communication systems, electromagnetic waves are used to carry information from a transmitter (TX) to a receiver (RX). During their travel, these waves are subject to many phenomena. They propagate in free-space in different directions and in straight lines. They bounce over the first obstacle they meet, lose energy, change direction and then continue on there journey until the next obstacle. In the end, some of the initial waves successfully attain the RX, and the resulting signal is thus a combination of many copies of the original signal with different amplitudes and phase shifts. In practice, all these phenomena are summed up in the concept of the *channel*. It represents the effects and the interactions that the propagation environment has over the electromagnetic waves in their journey from the TX to the RX. Its complexity is consequently directly tied to the complexity of the environments. In real world scenario, these can be highly diverse and multifaceted, presenting challenges in understanding and optimizing wireless communication systems. In urban areas, for instance, the channel is influenced by buildings, streets, and various structures that can cause signal reflections, diffraction around corners, and scattering off surfaces. In rural or open environments, natural features like hills, trees, and bodies of water also play a significant role in shaping the channel characteristics. Moreover, dynamic factors such as movement of objects, weather conditions, and interference from other electronic devices further contribute to the channel's variability.

To effectively design and operate wireless communication systems, it's essential to model and understand the behavior of the channel. This knowledge enables engineers and researchers to develop techniques that mitigate its ef-

fects and enhance the reliability and performance of wireless links. Various modeling approaches, including empirical models derived from measurement data and deterministic models based on physical principles, are employed to capture the channel's intricacies and reproduce realistic scenarios.

Knowing how to model the channel is only one step in designing comprehensive communication systems. Channel estimation (CE) comes next in line, as it is crucial to be able to acquire the channel state to ensure reliable communication. Plenty of CE algorithms exist, yet innovations are needed to address the challenges imposed by modern hardware and scenarios of growing complexity.

1.1 MIMO Channel models

1.1.1 General considerations

Free-space pathloss

In their simplest form, waves travel in free-space, that is, in a straight line from the TX to the RX. When considering isotropic antennas, energy conservation dictates that the energy radiated must be conserved uniformly in all directions, which corresponds to a spherical distribution. This means that the energy must be conserved over the entire surface of a sphere centered at the TX antenna. As the spherical wavefront expands, the energy per unit area decreases with the square of the distance from the source, maintaining the principle of energy conservation. This reasoning can be extended to include any antenna type and pattern. The receive power as a function of the distance r and the frequency f is given by Friis' law (Molisch 2022)

$$P_R = P_T G_T G_R \left(\frac{\lambda}{4\pi r}\right)^2, \tag{1.1}$$

where λ is the wavelength and is defined as $\lambda = \frac{c}{f}$. P_T corresponds to the transmit power. In the case of anisotropic antennas, the gains G_T and G_R in this equation depend on their radiation pattern seen in the direction of propagation at TX and RX respectively.

Fading

When considering the influence of the propagation environment, different factors come into play. Two significant ones to take into account are *small-scale fading* and *large-scale fading* in multipath signal propagation.

Small-scale fading, also known as fast fading, is primarily caused by the constructive and destructive interference of multipath components (MPCs). It results in rapid, short-term variations in the received signal strength due to the varying phase relationships and delays of individual MPCs. Small-scale fading can lead to signal drops, or deep fades, which can be problematic for maintaining a reliable wireless connection, especially in high-mobility scenarios.

On the other hand, large-scale fading, or shadowing, is a phenomenon associated with the attenuation of the signal as a result of obstructions. It typically leads to fluctuations of the signal's strength on a larger scale than small-scale fading. Large-scale fading is influenced by factors such as the type of the environment, its characteristics, and obstacles, called *scatterers*, in the signal's path. With shadowing, the receive power decays at a rate much faster than the r^{-2} given in (1.1).

Multipath propagation

The concept of MPCs plays a critical role in channel modeling. They refer to the various paths that electromagnetic waves can take from the TX to the RX due to reflections, diffractions, and scattering from objects and surfaces in the environment. These MPCs have different arrival times, amplitudes, and phase shifts, and they can either interfere constructively or destructively at the RX. Understanding and characterizing MPCs are vital for optimizing signal reception and mitigating issues such as signal fading and interference. Figure 1.1 shows a simplified 2D example of propagation in an urban scenario between a TX and a RX. The channel at the latter is the sum of the contributions of each individual component.

Highlight 1.1: Reflection, scattering, diffraction

When an electromagnetic wave meets a scatterer, it interacts with it. The type and the characteristics of the interactions are governed by the material and the geometry of the scatterer. If its surface is smooth, part of the wave is *reflected* and part is absorbed. Otherwise, if the surface is rough, the wave is diffusely *scattered*. Finally, if the wave meets an edge, it is *diffracted*. All these factors contribute to shadowing.

Channel modelling is the task of determining the various parameters that characterize the propagation channel. These include receive power, phase shifts, delays, etc.

1.1.2 Input/output model

Simple system

Let's now consider a simple wireless system. The RX symbol y at frequency f is given as a function of the TX symbol x and the channel h by

$$y = hx + n, \tag{1.2}$$

where n is zero-mean additive white Gaussian noise (AWGN), i.e., $n \sim \mathcal{CN}(0, \sigma^2)$. In the simple case where only a single communication path of length r exists, the channel is given by

$$h_{\text{single-path}} = \alpha g_R g_T e^{-j\frac{2\pi}{\lambda}r} = \alpha g_R g_T e^{-j\frac{2\pi}{c}fr}, \tag{1.3}$$

where α is the attenuation coefficient taking into account the pathloss, and g_R and g_T are the TX and RX gains respectively. When taking into account the different MPCs, the resulting channel is the sum of the channels corresponding to each component:

$$h = g_R g_T \sum_l \alpha_l e^{-j\frac{2\pi}{\lambda}r_l}, \tag{1.4}$$

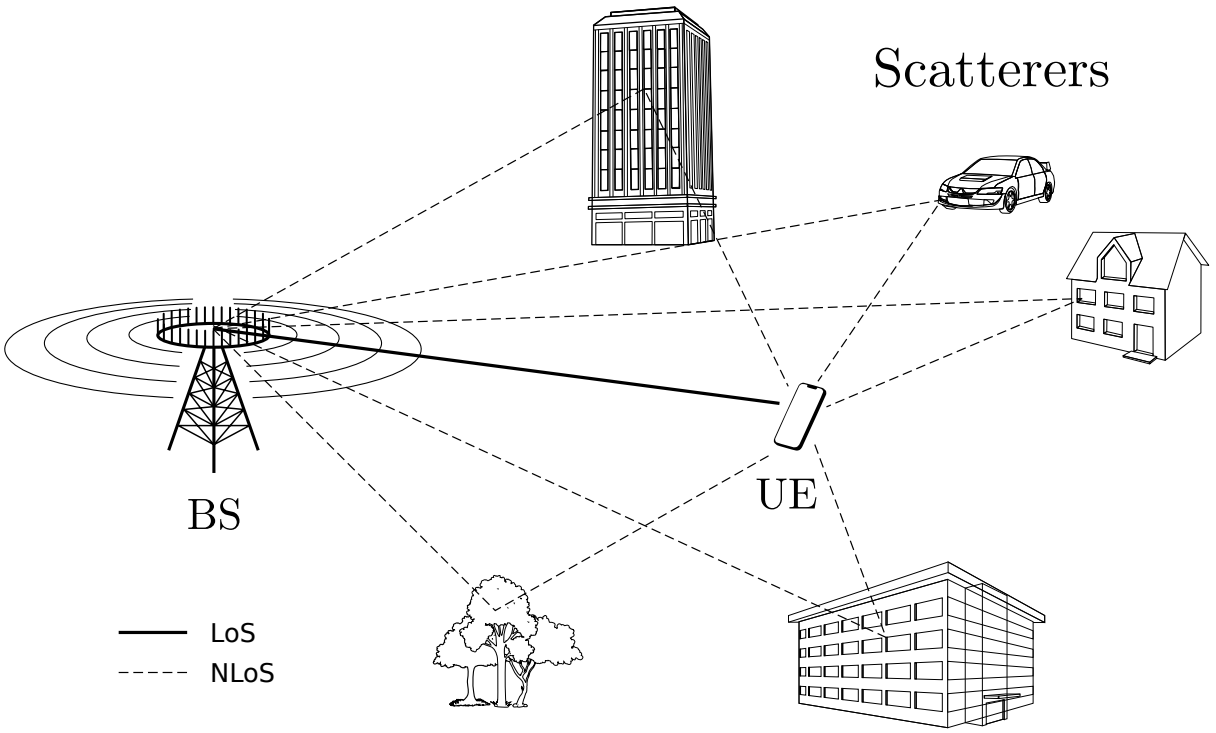


Figure 1.1 – Visualization of multipath propagation.

where the sum is taken over the components. This time, the attenuations and travel distances are specific to each component since different components take different paths to reach the RX. In addition, the attenuations for the non-line-of-sight (NLoS) components include pathloss and TX and RX gains, as well as the potential interactions with the scatterers previously discussed.

MIMO system

Modern systems have moved to using multiple TX and RX antennas in the form of antenna arrays. These types of systems are called multiple input multiple output (MIMO) systems. They offer several significant advantages, including:

— Improved Data Throughput: MIMO systems can transmit and receive multiple data streams simultaneously, which significantly increases the data throughput. This allows for faster data transfer rates and better utilization of available bandwidth.

— Enhanced Range: MIMO technology can extend the communication range

without increasing the transmit power. This is particularly beneficial for long-distance or outdoor communication links.

— Enhanced Reliability: MIMO systems can improve signal reliability by using spatial diversity. Even if some of the signal paths are obstructed or experience interference, others can still provide a reliable connection. This makes MIMO systems less susceptible to fading and interference.

— Enhanced Resistance to Interference: MIMO systems can spatially filter out interference by using beamforming techniques. This enables better performance in environments with high levels of interference caused by a high number of devices for example.

— Lower Transmit Power: MIMO can achieve the same communication quality with lower transmit power compared to single-antenna systems. This can lead to significant energy savings.

All in all, MIMO wireless systems offer substantial advantages, including higher data rates, increased range, better reliability, improved spectral efficiency, and adaptability to various communication scenarios. These advantages have made MIMO a fundamental technology in modern wireless communication systems, such as Wi-Fi, 4G, and 5G networks, and it continues to be at the center of attention in current state-of-the-art work.

The channel in the case of a MIMO system can be written in matrix form, where each element corresponds to a TX and RX pair. Similarly to (1.2), the RX symbol $\mathbf{y}$, in vector form this time, can be written as a function of the TX symbol $\mathbf{x}$ and the channel

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}. \tag{1.5}$$

$\mathbf{H}$ is the channel matrix, also called *channel state information* (CSI). Each of its elements represent a unique propagation channel between the ith RX antenna and the jth TX antenna:

$$\mathbf{H} = \begin{bmatrix} h_{1,1} & \cdots & h_{1,N_T} \\ \vdots & \ddots & \vdots \\ h_{N_R,1} & \cdots & h_{N_R,N_T} \end{bmatrix}. \tag{1.6}$$

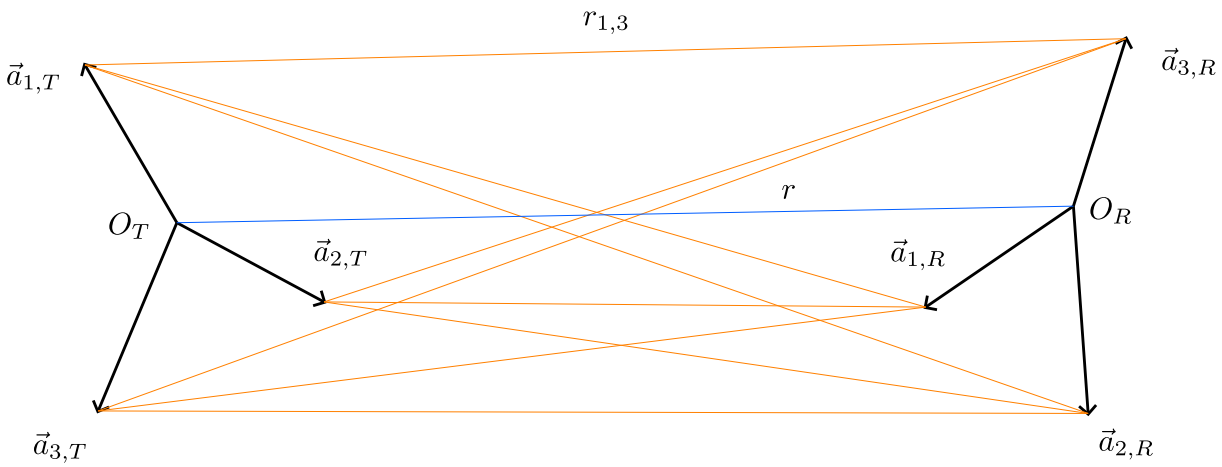


Figure 1.2 – Visualization of the antenna geometries of arbitrary arrays and the distances between each element.

Each element of $\mathbf{H}$ can be expressed in a similar manner to (1.3) as

$$h_{i,j} = \alpha g_{i,R} g_{j,T} e^{-j\frac{2\pi}{\lambda} r_{i,j}}, \tag{1.7}$$

where $r_{i,j}$ is the distance between the ith RX antenna and the jth TX antenna, and $g_{i,R}$ and $g_{j,T}$ are the gains of the ith RX and jth TX antennas respectively. The RX (or TX equivalently) is supposed to be in the far field so that the plane wave assumption can be considered, in which case, the geometries of the antenna arrays play an important role. For TX and RX antenna arrays of sizes N_T and N_R, respectively, define the positions of their antenna elements relative to their centroids O_T and O_R as $\{\vec{a}_{1,T}, \ldots, \vec{a}_{N_T,T}\}$ and $\{\vec{a}_{1,R}, \ldots, \vec{a}_{N_R,R}\}$ as shown on figure 1.2. Consequently, the distance between each antenna pair is replaced by its first-order approximation where $r_{i,j} \approx r - \vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R$, r is the distance between O_T and O_R, and $\vec{u}_T$ and $\vec{u}_R$ are unit vectors pointing at the direction of departure (DoD) and the direction of arrival (DoA), respectively. Figure 1.3 shows a geometrical explanation of this approximation. Under the plane wave assumption, equation (1.7) becomes

$$h_{i,j} = \alpha g_{i,R} g_{j,T} e^{-j\frac{2\pi}{\lambda}(r - \vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R)}. \tag{1.8}$$

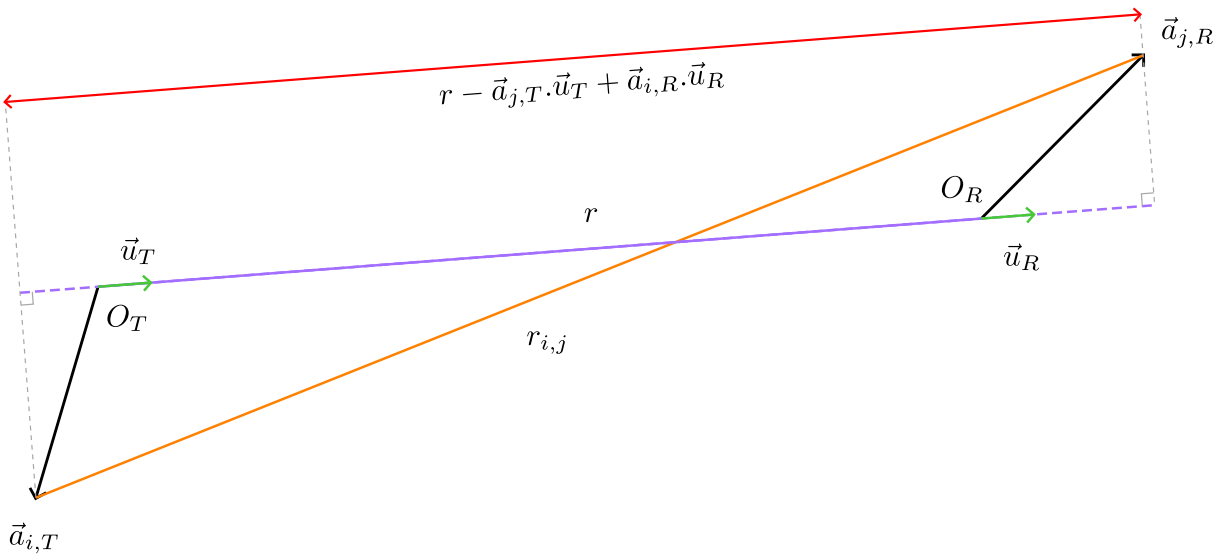


Figure 1.3 – Geometrical visualization of the plane wave approximation. The orange distance is approximated by the red one.

Reusing (1.8) in (1.6) leads to this nice expression

$$\mathbf{H} = \sqrt{c_T c_R}\alpha e^{-j\frac{2\pi}{\lambda}r}\mathbf{a}_R(\vec{u}_R)\mathbf{a}_T(\vec{u}_T)^{\mathsf{H}}, \tag{1.9}$$

where the phase-shift contributions of the displacement of each antenna element are organized in vectors called *steering vectors* (or *array response vectors*) and defined as

$$\mathbf{a}_X(\vec{u}_x) \triangleq \frac{1}{\sqrt{c_X}} \begin{pmatrix} g_{1,X}e^{-j\frac{2\pi}{\lambda}\vec{a}_{1,X}.\vec{u}_X} \\ \vdots \\ g_{N_X,X}e^{-j\frac{2\pi}{\lambda}\vec{a}_{N_X,X}.\vec{u}_X} \end{pmatrix} \text{ with } X \in \{T, R\}. \tag{1.10}$$

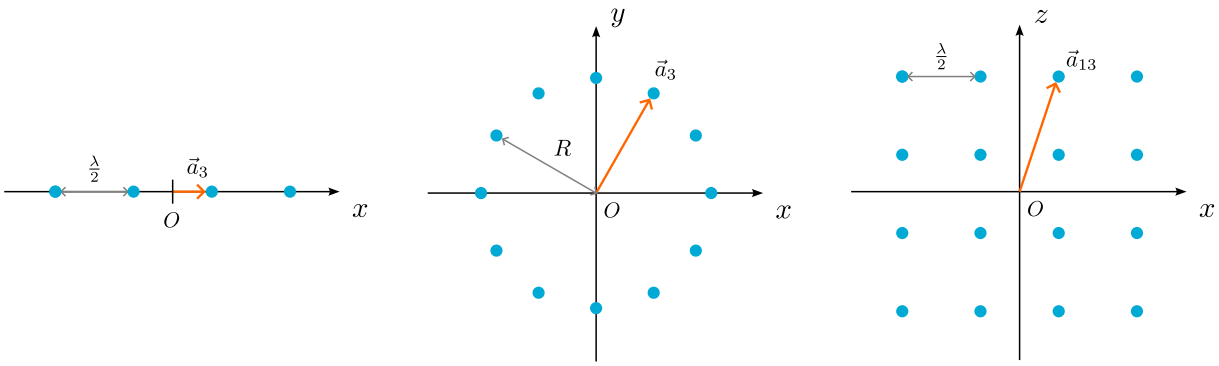
where $c_X = \sum_{i=1}^{N_X} |g_{i,X}|^2$ [1]. Once again taking into account the MPCs leads to

$$\mathbf{H} = \sqrt{c_T c_R}\sum_l \alpha_l e^{-j\frac{2\pi}{\lambda}r_l}\mathbf{a}_R(\vec{u}_{l,R})\mathbf{a}_T(\vec{u}_{l,T})^{\mathsf{H}}. \tag{1.11}$$

Here, in addition to a specific attenuation and travel distance, each component is characterized by a DoD and a DoA as well.

Some examples of the antenna positions of usual antenna array geometries

1. In the case of unitary gains, $a_X = N_X$.



(a) ULA (b) UCA (c) UPA

Figure 1.4 – Different antenna arrays.

are shown on figure 1.4 and are obtained using the following:

— *Uniform linear array* (ULA)—aligned with the x axis:

$$\vec{a}_i = \left((i - \frac{N_X + 1}{2}) \frac{\lambda}{2}, \, 0, \, 0 \right); \tag{1.12}$$

— *Uniform circular array* (UCA)—on the xy-plane:

$$\vec{a}_i = \left(R \cos \frac{2\pi(i-1)}{N_X}, \, R \sin \frac{2\pi(i-1)}{N_X}, \, 0 \right), \tag{1.13}$$

where R is the radius of the array;

— *Uniform planar array* (UPA)—on the xz-plane:

$$\vec{a}_{ij} = \left((i - \frac{\sqrt{N_X} + 1}{2}) \frac{\lambda}{2}, \, 0, \, (j - \frac{\sqrt{N_X} + 1}{2}) \frac{\lambda}{2} \right). \tag{1.14}$$

Here, the positions are defined using double indices to align with the 2D nature of UPAs. The resulting steering vector is thus in matrix form. A vectorization operation can be used to convert it to a vector.

Multiple frequencies

While MIMO corresponds to sampling the signal over different spatial points, a similar strategy can be applied to the frequency domain where the signal is sampled at different points, with each point corresponding to a pure frequency

subcarrier. Consequently, symbols are transmitted over multiple subcarriers of frequencies $\{f_1, \ldots, f_{N_s}\}$, where N_s is their number. Combining MIMO with multiple frequencies is a common practice in modern systems. However, since the TX and RX antennas are already taking up the two dimensions of matrix $\mathbf{H}$, an additional dimension for frequencies will not fit. A solution is to switch to tensor notation. Each element of the resulting tensor can be obtained from (1.8) by substituting the right frequency term

$$h_{i,j,s} = \alpha g_{i,R} g_{j,T} e^{-j\frac{2\pi}{c} f_s (r - \vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R)} \tag{1.15}$$

$$= \alpha g_{i,R} g_{j,T} e^{-j\frac{2\pi}{c}\left(f_c(r - \vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R) + (f_s - f_c) r - \cancel{(f_s - f_c)(\vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R)}\right)}, \tag{1.16}$$

where $f_c \triangleq \frac{1}{N_s}\sum_s f_s$ is the central frequency and s, i and j are the subcarrier, TX antenna and RX antenna indices, respectively. Tensor notation is however not very suitable for expressing the input/output equation. Nonetheless, there exists a workaround involving a rewriting of the latter and under the reasonable assumption that the last term is negligible[2]. First, define the *frequency signature vector* $\mathbf{f}(r)$, which is analogous to the steering vectors but this time considering the phase-shift contributions of the different frequency components.

$$\mathbf{f}(r) \triangleq \frac{1}{\sqrt{N_s}} \begin{pmatrix} e^{-j\frac{2\pi}{c} r (f_1 - f_c)} \\ \vdots \\ e^{-j\frac{2\pi}{c} r (f_{N_s} - f_c)} \end{pmatrix}. \tag{1.17}$$

The input/output equation can then be rewritten as

$$\mathbf{Y} = \sqrt{N_s} \underbrace{\sqrt{c_T c_R}\alpha e^{-j\frac{2\pi}{\lambda} r} (\mathbf{a}_R(\vec{u}_R)\mathbf{a}_T(\vec{u}_T)^{\mathsf{H}})}_{\mathbf{H} \text{ from (1.9)} = \mathbf{H}_{\text{MIMO}}} \mathbf{X} \underbrace{\operatorname{diag}(\mathbf{f}(r))}_{\mathbf{H}_{\text{OFDM}}} + \mathbf{N}, \tag{1.18}$$

where $\mathbf{Y} \in \mathbb{C}^{N_R \times N_s}$, $\mathbf{X} \in \mathbb{C}^{N_T \times N_s}$ and $\mathbf{N} \in \mathbb{C}^{N_R \times N_s}$. Now the trick to restore this equation to a form involving a single channel matrix is to vectorize the whole.

2. This is because, in real-world systems, $\frac{f_s - f_c}{c}(\vec{a}_{j,T}.\vec{u}_T + \vec{a}_{i,R}.\vec{u}_R) \ll 1$. This approximation essentially says that for narrowband systems, the phase shift associated with each antenna element of the arrays (TX and RX) is almost constant over the whole bandwidth and can be directly computed from the central frequency f_c. Note that it is only necessary when combining MIMO and multiple subcarriers. It is irrelevant in scenarios involving one or the other.

Consequently, the new relationship becomes

$$\operatorname{vec}(\mathbf{Y}) = \operatorname{vec}(\mathbf{H}_{\text{MIMO}}\mathbf{X}\mathbf{H}_{\text{OFDM}}) + \operatorname{vec}(\mathbf{N}) \tag{1.19}$$

$$= \underbrace{(\mathbf{H}_{\text{OFDM}} \otimes \mathbf{H}_{\text{MIMO}})}_{\mathbf{H}_{\text{MIMO-OFDM}}} \operatorname{vec}(\mathbf{X}) + \operatorname{vec}(\mathbf{N}), \tag{1.20}$$

which is of the same form as (1.5). In other words, the idea consists of stacking the antenna and frequency dimensions. The resulting channel matrix is block diagonal of the form

$$\mathbf{H}_{\text{MIMO-OFDM}} = \sqrt{N_s c_T c_R}\, \operatorname{diag}(\mathbf{f}(r)) \otimes \alpha e^{-j\frac{2\pi}{\lambda}r}(\mathbf{a}_R(\vec{u}_R)\mathbf{a}_T(\vec{u}_T)^{\mathsf{H}}). \tag{1.21}$$

Additionally, the multipath version of the channel is

$$\mathbf{H} = \sqrt{N_s c_T c_R} \sum_l \alpha_l e^{-j\frac{2\pi}{\lambda}r_l}\, \operatorname{diag}(\mathbf{f}(r_l)) \otimes (\mathbf{a}_R(\vec{u}_{l,R})\mathbf{a}_T(\vec{u}_{l,T})^{\mathsf{H}}). \tag{1.22}$$

While the presented model of the multiple frequencies channel isn't restricted to any particular distribution of frequencies, modern systems use a specific waveform called orthogonal frequency division multiplexing (OFDM). It consists of dividing the available bandwidth into orthogonal subcarriers, each carrying a different symbol. It can be viewed as transmitting many narrowband signals where the channel is locally flat. The orthogonality requires that the subcarrier spacing Δ_f is $\geq \frac{1}{T_s}$ where T_s is the symbol duration. OFDM has the advantage of greatly simplifying the design of the TX and RX radio chains as the subcarriers can be separated using the fast Fourier transform (FFT) and it no longer requires a different filter for each one.

To simulate the behavior of real systems, channel models typically rely on either *deterministic* or *statistical* techniques, or a combination of both in some more involved cases. In the following, an overview of these techniques is provided.

1.1.3 Deterministic models

Deterministic models strive to accurately estimate the parameters of the channel by simulating the propagation environment from a given TX location

to an RX one. They usually rely on ray tracing, which is a technique that traces the paths of individual rays as they interact with the surrounding objects and surfaces. By meticulously modeling the interactions and reflections of these rays, deterministic models provide detailed insights into how electromagnetic waves propagate through a specific environment. The environment itself is usually a 3D reproduction of its real-world equivalent. Ray tracing solutions however are known for being computationally intensive, requiring a lot of computations to best approximate real-world propagation. Fortunately, the incredible progress that the area of microelectronics have seen in the recent years has led to the emergence of powerful machines capable of meeting the computational demands of many intensive programs, including ray tracers. In the scientific literature, ray tracers have been extensively used to generate datasets of realistic channel observations in custom scenarios in order to verify the validity of novel ideas, whereas collecting real-world data would have been impractical, too expensive or often simply impossible.

DeepMIMO (A. Alkhateeb 2019) is one such example. It is a framework for generating datasets of millimeter wave (mmWave) MIMO channel measurements in different virtual scenarios based on the Remcom Wireless InSite ray tracer (Remcom n.d.). It is particularly aimed at machine learning applications which usually require great amounts of data. DeepMIMO has the advantage of being parametric, meaning that some parameters such as the number of antennas and the number of subcarriers can be adjusted and adapted to each application. The positions of the radio devices are however predetermined. Figure 1.5 shows a 3D view of the "O1" scenario consisting of two intersecting streets surrounded by buildings. The channel measurements are provided for different frequencies including 3.5 GHz and 28 GHz.

Sionna RT (Hoydis, Ait Aoudia, et al. 2023) is another framework for generating deterministic channel measurements. It is part of the bigger Sionna library (Hoydis, Cammerer, et al. 2022) that was recently introduced as a featureful toolbox that aims at facilitating and accelerating AI-based research in communication systems. It comes with a house-made open source ray tracer that is capable of simulating wave propagation through any 3D environment it is provided with. Such environments can be easily generated from geographic databases such as OpenStreetMap (OpenStreetMap contributors

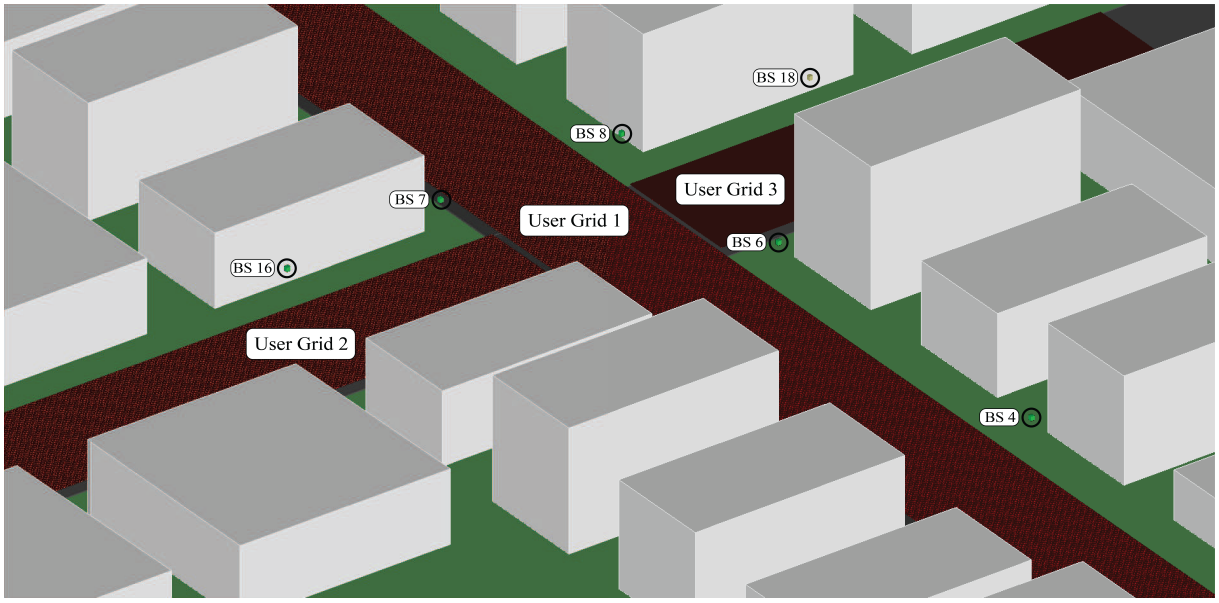


Figure 1.5 – The "O1" scenario from DeepMIMO. The UEs are located on the roads while the BSs are above the ground. The different gray blocks represent the surrounding buildings. Source: (A. Alkhateeb 2019).

2017). The positions of the radio devices and their characteristics can be customized along with other parameters like the electromagnetic properties of the scatterers. Figure 1.6 show a 3D reproduction of an area in Munich along with some propagation paths using Sionna.

1.1.4 Statistical models

Contrary to deterministic models, statistical ones rely on the observation that the channel parameters usually follow some probabilistic distributions. The goal of these models becomes one of determining an adequate family of distributions as well as estimating their parameters. It is worth noting that the considered parameters of study are what determine the nature of the model. Indeed, the simplest models consider the whole channel matrix as random, the goal becomes to model each entry as a random variable. More advanced models study the distribution of the underlying parameters such as the attenuations and the DoAs, etc. Channel observations are then generated by drawing the parameters from the distributions.

The simplest example of this approach is the *Rayleigh-fading model*. It

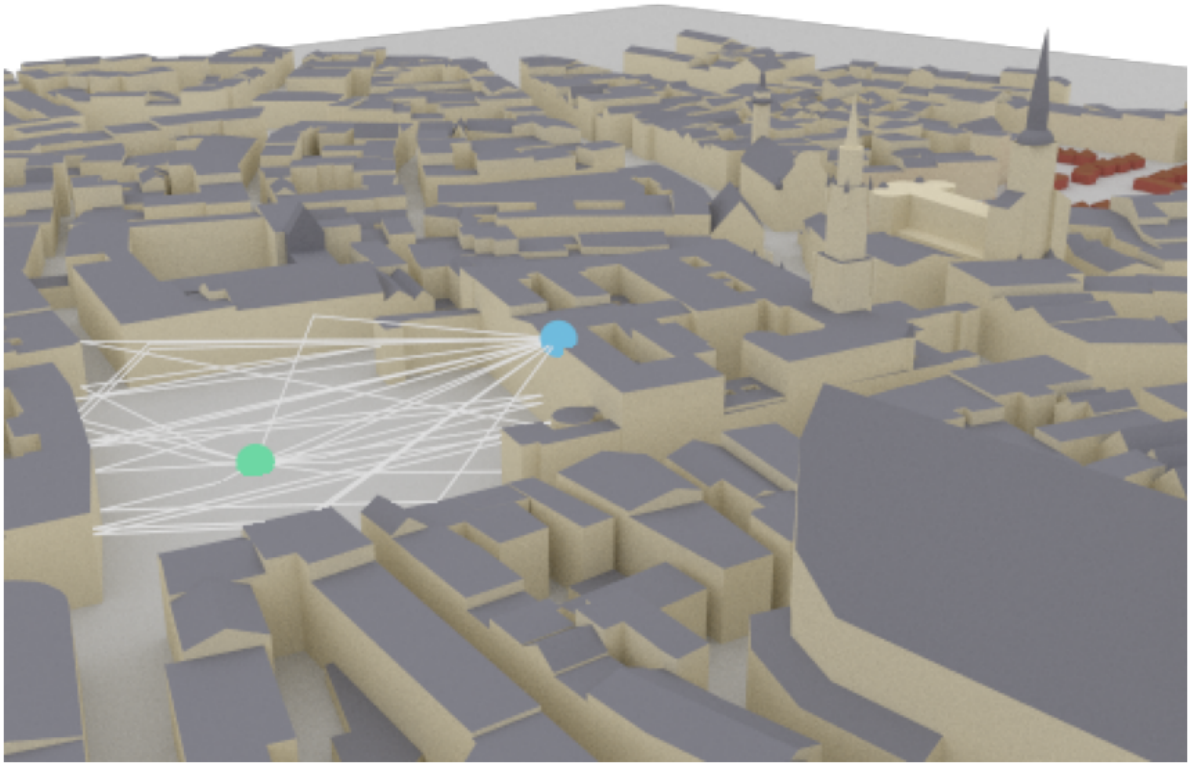


Figure 1.6 – Example of a ray tracing simulation generated with the Sionna library. The scene contains the area around the Frauenkirche in Munich. The blue sphere is the transmitter and the green one is a receiver. The white paths correspond to the different MPCs.

considers the case where there is no LoS component and where all components arise from a high number of distant scatterers distributed around the RX, which is a reasonable enough assumption in many scenarios. In the absence of a dominant component, the contributions of the components in (1.4) can be modeled as i.i.d. random variables. When their number grows large enough, the Central Limit Theorem applies and the resulting channel follow a Gaussian distribution $h \sim \mathcal{CN}(0, \sigma^2)$. The amplitude of such a channel follows a Rayleigh distribution, which explains the name of the model. This model also applies to MIMO channels where $\mathrm{vec}(\mathbf{H}) \sim \mathcal{CN}(\mathbf{0}, \mathbf{R})$. The simplest case is to consider i.i.d. channel coefficients, i.e., $\mathbf{R} = \sigma^2\mathbf{I}$.

The statistical spatial channel model (SSCM) (Samimi and T. S. Rappaport 2016) is an example of a more sophisticated model designed for MIMO sys-

tems. It is based on extensive propagation measurements and is suitable for generating synthetic mmWave (28 GHz and 73 GHz) LoS and NLoS observations corresponding to urban scenarios. The model consists of a multi-step process describing how to generate the effective channel vectors and involves drawing a series of realizations from estimated distributions as explained in the original paper. The results are realistic and have the same properties as the real-world measurements.

QuaDRiGa (Jaeckel et al. 2014) is another example of a statistical model worth mentioning. More precisely, it belongs to the sub-category of geometry-based statistical models. Indeed, the "QuaD" in its name stands for QUAsi Deterministic, meaning that parts of it are stochastic while others are deterministic. It is widely adopted within the research community thanks to the multitude of features it offers and the adaptability to different scenarios it allows.

1.2 MIMO Channel estimation

In a real communication scenario, the channel matrix is unknown. *Channel estimation* (CE) is the task of computing an estimate of the channel from a set of reference signals known to both the transmitter and the receiver, called *pilots*. They are usually transmitted alongside data symbols in a periodic manner and over different frequencies to allow for continuous CE as the channel changes and evolves. While pilot design and allocation is important for fast and reliable channel estimation, it is a subject of its own and is outside the scope of this manuscript, as is *blind* CE which considers estimating the channel without relying on pilot symbols. Consider N_p pilot symbols, with stacked frequency and antenna dimensions as previously explained, transmitted and organized in a matrix $\mathbf{P} = (\mathbf{p}_1, \ldots, \mathbf{p}_{N_p}) \in \mathbb{C}^{N_T N_s \times N_p}$. Formally, CE consist of estimating $\mathbf{H} \in \mathbb{C}^{N_R N_s \times N_s N_T}$ in

$$\mathbf{Y} = \mathbf{H}\mathbf{P} + \mathbf{N}, \tag{1.23}$$

from the knowledge of $\mathbf{Y} \in \mathbb{C}^{N_R N_s \times N_p}$ and $\mathbf{P}$. $\mathbf{N} = (\mathbf{n}_1, \ldots, \mathbf{n}_{N_p}) \in \mathbb{C}^{N_R N_s \times N_p}$ is the matrix of AWGN vectors.

Some very well-known strategies for estimating $\mathbf{H}$ from the knowledge of $\mathbf{P}$ (Biguesh and Gershman 2006) are recalled in the sequel.

1.2.1 Least squares channel estimation

The least squares (LS) estimate is the solution to the minimization problem

$$\underset{\hat{\mathbf{H}}}{\operatorname{argmin}}\, \mathcal{L}_{\text{MSE}}(\mathbf{Y}, \hat{\mathbf{H}}\mathbf{P}) = \underset{\hat{\mathbf{H}}}{\operatorname{argmin}}\, \|\hat{\mathbf{H}}\mathbf{P} - \mathbf{Y}\|_2^2. \tag{1.24}$$

In other terms, the solution minimizes the squared error across pilot symbols. This problem is convex, and the minimum can be obtained by finding the root of the derivative. It can be shown that

$$\frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \hat{\mathbf{H}}}(\mathbf{Y}, \hat{\mathbf{H}}\mathbf{P}) = -2\mathbf{Y}\mathbf{P}^{\mathsf{H}} + 2\hat{\mathbf{H}}\mathbf{P}\mathbf{P}^{\mathsf{H}}. \tag{1.25}$$

It follows that

$$-2\mathbf{Y}\mathbf{P}^{\mathsf{H}} + 2\hat{\mathbf{H}}_{\text{LS}}\mathbf{P}\mathbf{P}^{\mathsf{H}} = \mathbf{0} \Rightarrow \mathbf{Y}\mathbf{P}^{\mathsf{H}} = \hat{\mathbf{H}}_{\text{LS}}\mathbf{P}\mathbf{P}^{\mathsf{H}}. \tag{1.26}$$

Finally, the LS estimate is given by

$$\hat{\mathbf{H}}_{\text{LS}} = \mathbf{Y}\mathbf{P}^{\mathsf{H}}(\mathbf{P}\mathbf{P}^{\mathsf{H}})^{-1}. \tag{1.27}$$

Recall that the LS estimate also corresponds to the maximum likelihood estimate (MLE) under the AWGN assumption, i.e., $\mathbf{n}_p \sim \mathcal{CN}(\mathbf{0}, \sigma^2\mathbf{I})$. Consider the log-likelihood

$$\log p(\mathbf{Y}|\mathbf{P}, \hat{\mathbf{H}}) = \underbrace{\cancel{-N_p N_R(\sigma^2 + \log \pi)}}_{\text{Doesn't depend on } \hat{\mathbf{H}}} - \frac{1}{\sigma^2}\sum_{p=1}^{N_p}(\mathbf{y}_p - \hat{\mathbf{H}}\mathbf{p}_p)^{\mathsf{H}}(\mathbf{y}_p - \hat{\mathbf{H}}\mathbf{p}_p) \tag{1.28}$$

$$= -\frac{1}{\sigma^2}\operatorname{tr}\left((\mathbf{Y} - \hat{\mathbf{H}}\mathbf{P})^{\mathsf{H}}(\mathbf{Y} - \hat{\mathbf{H}}\mathbf{P})\right) \tag{1.29}$$

$$= -\frac{1}{\sigma^2}\|\hat{\mathbf{H}}\mathbf{P} - \mathbf{Y}\|_2^2. \tag{1.30}$$

Minimizing the log-likelihood is thus equivalent to (1.24) and leads to the same solution.

1.2.2 LMMSE channel estimation

The minimum mean squared error (MMSE) estimator is the solution to the minimization problem

$$\underset{\hat{\mathbf{H}}}{\operatorname{argmin}}\, \mathbb{E}\left[\mathcal{L}_{\text{MSE}}(\hat{\mathbf{H}}, \mathbf{H})\right] = \underset{\hat{\mathbf{H}}}{\operatorname{argmin}}\, \mathbb{E}\left[\|\hat{\mathbf{H}} - \mathbf{H}\|_2^2\right], \tag{1.31}$$

where the expectation is taken over $\mathbf{H}$ conditioned on $\mathbf{Y}$. Solving this problem can be difficult. A solution is to consider a first order approximation by considering the linear relationship $\hat{\mathbf{H}} = \mathbf{YW}$. The goal becomes to find the $\mathbf{W}$ that minimizes (1.31) so that

$$\underset{\mathbf{W}}{\operatorname{argmin}}\, \mathbb{E}\left[\|\mathbf{YW} - \mathbf{H}\|_2^2\right] = \underset{\mathbf{W}}{\operatorname{argmin}}\, \operatorname{tr}\left(\mathbb{E}\left[(\mathbf{HPW} + \mathbf{NW} - \mathbf{H})^{\mathsf{H}}(\mathbf{HPW} + \mathbf{NW} - \mathbf{H})\right]\right) \tag{1.32}$$

$$= \underset{\mathbf{W}}{\operatorname{argmin}}\, \operatorname{tr}(\mathbf{R}) - \operatorname{tr}(\mathbf{RPW}) - \operatorname{tr}\left(\mathbf{W}^{\mathsf{H}}\mathbf{P}^{\mathsf{H}}\mathbf{R}\right) + \tag{1.33}$$

$$\operatorname{tr}\left(\mathbf{W}^{\mathsf{H}}(\mathbf{P}^{\mathsf{H}}\mathbf{RP} + \sigma\mathbf{I})\mathbf{W}\right) + \tag{1.34}$$

$$\underbrace{\operatorname{tr}\left((\mathbf{W}^{\mathsf{H}}\mathbf{P}^{\mathsf{H}} - \mathbf{I})\mathbb{E}\left[\mathbf{H}^{\mathsf{H}}\mathbf{N}\right]\mathbf{W}\right) + \operatorname{tr}\left(\mathbf{W}^{\mathsf{H}}\mathbb{E}[\mathbf{N}^{\mathsf{H}}\mathbf{H}](\mathbf{PW} - \mathbf{I})\right)}_{\text{Because } \mathbb{E}[\mathbf{H}^{\mathsf{H}}\mathbf{N}]=0} \tag{1.35}$$

where $\mathbf{R} = \mathbb{E}[\mathbf{H}^{\mathsf{H}}\mathbf{H}]$ and $\mathbb{E}[\mathbf{N}^{\mathsf{H}}\mathbf{N}] = \sigma\mathbf{I}$ are the correlation matrices of $\mathbf{H}$ and $\mathbf{N}$ respectively[3]. Similarly to the LS case, the minimum can be obtained by finding the root of the derivative.

$$\frac{\partial \mathcal{L}_{\text{MSE}}}{\partial \mathbf{W}}(\mathbf{YW}, \mathbf{H}) = 2(\mathbf{P}^{\mathsf{H}}\mathbf{RP} + \sigma\mathbf{I})\mathbf{W} - 2\mathbf{P}^{\mathsf{H}}\mathbf{R}. \tag{1.36}$$

It follows that

$$2(\mathbf{P}^{\mathsf{H}}\mathbf{RP} + \sigma\mathbf{I})\mathbf{W} - 2\mathbf{P}^{\mathsf{H}}\mathbf{R} = \mathbf{0} \Rightarrow \mathbf{W} = (\mathbf{P}^{\mathsf{H}}\mathbf{RP} + \sigma\mathbf{I})^{-1}\mathbf{P}^{\mathsf{H}}\mathbf{R} \tag{1.37}$$

$$\Rightarrow \hat{\mathbf{H}}_{\text{LMMSE}} = \mathbf{Y}(\mathbf{P}^{\mathsf{H}}\mathbf{RP} + \sigma\mathbf{I})^{-1}\mathbf{P}^{\mathsf{H}}\mathbf{R} \tag{1.38}$$

3. These correlation matrices are different from the traditional ones which are usually defined for a given random matrix $\mathbf{A}$ as $\mathbb{E}[\operatorname{vec}(\mathbf{A})^{\mathsf{H}} \operatorname{vec}(\mathbf{A})]$

Interestingly, if $\mathbf{R} = \mathbf{I}$ and if the noise level is negligible compared to the receive power, the LMMSE estimator $\hat{\mathbf{H}}_{\text{LMMSE}}$ becomes equivalent to the LS estimator $\hat{\mathbf{H}}_{\text{LS}}$.

1.2.3 Sparse recovery for channel estimation

In some cases, the channel can be sparse both in the temporal and spatial domains, meaning that it consists of a small number of MPCs. This is the case for example for channels in high frequency bands such as millimeter waves (mmWave) (Akdeniz et al. 2014; Samimi and T. S. Rappaport 2016). This property can be exploited in the design of estimation algorithms. Indeed, sparse recovery approaches (D. Donoho 2006) has been considered for CE (Cotter and B. Rao 2002; Bajwa et al. 2010). The literature on the subject is diverse, and there are as many approaches as there are sparse recovery algorithms and sparse channel domains (Choi et al. 2017).

As an example, consider sparsity in the angular domain of MISO single-carrier channels. A first estimate of the channel is obtained through the LS method, i.e., $\hat{\mathbf{h}}_{\text{LS}} = \mathbf{Y}\mathbf{p}^{\mathsf{H}}(\mathbf{p}\mathbf{p}^{\mathsf{H}})^{-1}$. From (1.9), the channel can be written as a linear combination of a few (since the channel is sparse) steering vectors with the DoAs of the corresponding MPCs. Consider a dictionary of steering vectors (called atoms in this context) $\mathbf{D} = (\mathbf{a}_R(\vec{u}_{R,1}), \ldots, \mathbf{a}_R(\vec{u}_{R,N_D})) \in \mathbb{C}^{N_R \times N_D}$ where the DoAs can be chosen to be uniformly distributed around the receiver, or to have a specific distribution if prior knowledge is available. A general formulation of sparse recovery problems is

$$\underset{\mathbf{w}}{\operatorname{argmin}} \|\mathbf{w}\|_0 \quad \text{such that } \mathbf{D}\mathbf{w} = \hat{\mathbf{h}}_{\text{LS}}, \tag{1.39}$$

where $\|.\|_0$ is the l_0-norm corresponding to the number of non-zero elements in a vector.

Instead of trying to directly minimize this problem, greedy algorithms, a subclass of sparse recovery algorithms, try to iteratively recover the linear combination of atoms of $\mathbf{D}$ that best matches $\hat{\mathbf{h}}_{\text{LS}}$. Take *matching pursuit* (MP) (Mallat and Z. Zhang 1993) for example, the idea is to compute at each iteration, in a greedy fashion, the steering vector that correlates the most with the channel estimate. The latter is then projected on that vector, and its contri-

bution is subtracted from the residual that becomes the input of the following iteration. This is best explained by algorithm 1.

Algorithm 1 MP & OMP

Input: Dictionary of atoms $\mathbf{D}$, input $\mathbf{h}$
$\mathbf{r} \leftarrow \mathbf{h}$
if OMP **then**
  $\Lambda \leftarrow \emptyset$
end if
while Stopping criterion not met **do**
  Find the most correlated atom: $s \leftarrow \underset{i}{\text{argmax}} \ |\mathbf{a}_i^{\mathsf{H}}\mathbf{r}|$
  if MP **then**
    Update the residual: $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{a}_s\mathbf{a}_s^{\mathsf{H}}\mathbf{r}$
  else
    if OMP **then**
      Update the set of atoms: $\Lambda \leftarrow \Lambda \cup \{s\}$
      Update the residual: $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{D}_\Lambda \operatorname{argmin}_{\mathbf{x}} \|\mathbf{D}_\Lambda\mathbf{x} - \mathbf{h}\|_2$
    end if
  end if
end while
Output: $\hat{\mathrm{h}} \leftarrow \mathrm{h} - \mathbf{r}$

The number of iterations corresponds to the sparsity of the channel[4]. It can be determined manually, or by deciding on a stopping criterion (e.g., depending on the SNR). Finally, this example can be seen as refining the LS estimate where the projections on the different atoms serve as a denoising process.

It is worth noting that MP has been declined into an orthogonal version, *orthogonal matching pursuit* (OMP), where a single atom is selected at each step, but instead of computing its projection coefficient alone, all the coefficients of the atoms selected up until that iteration are recomputed to minimize the l_2-norm, resulting in an orthogonal projection of the input on the subspace spanned by that set of atoms at each step. Consequently, OMP shares the same base as MP, but differs in the way atoms are updated as shown in algorithm 1.

4. This is true only when an atom can be selected at most once throughout the iterations.

1.3 Conclusion

This chapter was a brief overview of channel modeling and estimation. While the principal of wave propagation appears simple, the different interactions with the surrounding environment and the resulting effects on the channel are not trivial. The chapter has introduced basic modeling principals starting from the simplest setting and moving to more complex ones involving multiple antennas and frequencies. To obtain realistic simulation data, two main families of models exist: deterministic ones and statistical ones. Popular examples of both were given.

When it comes to channel estimation, many algorithms exist. The chapter introduced the most popular ones that are based on the LS estimator and the LMMSE estimator. More advanced ones exploiting the sparsity of the channel were also introduced involving the use of MP, OMP or other sparse recovery algorithms.

The next chapter focuses on machine learning, and on deep learning in particular. It also introduces the principal of dimensionality reduction which is at the core of the contributions of this manuscript as we will see.

CHAPTER 2

MACHINE LEARNING

Natural data represent a phenomenal source of knowledge that us humans have been gathering for centuries. Through their observation and collection, we came to understand many of the laws of the universe we live in. With the advancement of electronics, the amount of data that we were able to capture grew exponentially. Natural data has the interesting property of being subject to a particular structure. Indeed, contrary to random noise, knowledge can be extracted by comparing different data points and conclusions can be drawn about the phenomenon at study. This observation gave rise to a not-so-old scientific paradigm, called *machine learning* (ML). It is the ability of manipulating and processing data in an automated way in order to find the answer to a particular question. The automation part of this concept is what differentiates it from classical approaches. Indeed, the machine performing the task is expected to "learn" to solve the problem from the data.

ML concepts are not new, but only recently have we seen a real global adoption. This can be attributed to two main factors: the phenomenal increase in computational power offered by modern hardware and the abundance of data.

At the core of ML are algorithms that are able to effectively learn and adjust to data. From pure statistical models to modern deep learning networks, these algorithm have been developed and refined for the last decades. Today, their potential no longer needs to be proven. Some of these algorithms are explored in this chapter, starting with the basic linear regression and moving to more complex algorithms with some involving different flavors of neural networks. Some of these algorithms are investigated in the context of dimensionality reduction, which is at the code of some of the work presented in this manuscript. Although all these methods are based on the same principle of

learning from data, they differ in ways that make them suitable for different applications as we will see.

In the context of wireless communications, they present a great opportunity to improve existing systems on all levels. A great example is how they can be used to improve the performance of channel estimation.

2.1 General principle

ML is in reality an umbrella term that encompasses a large set of algorithms capable of learning to perform a specific task without being explicitly programed to do so, from the data that are presented to them (called the training set). The ultimate goal of a learning algorithm (or model) is to be able to correctly perform the task it was trained for when presented with a piece of data that it has never seen before. In other terms, its performance is evaluated on its ability to generalize to unseen data (called the test set). The task that the algorithm learns to perform is usually too complex to be solved by hand-crafted algorithms.

ML can take many forms depending on the task at hand. For example, in *supervised learning*, the learning algorithm is presented with a training set consisting of inputs and their corresponding expected answers, called labels. During the training phase, the algorithm tries to predict the correct label and adjusts itself whenever its prediction is wrong. Formally, training an algorithm amounts to finding a function $f^\star$ so that

$$f^\star = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{N} \sum_{(x,y)\in\mathcal{T}} \mathcal{L}(y, f(x)). \tag{2.1}$$

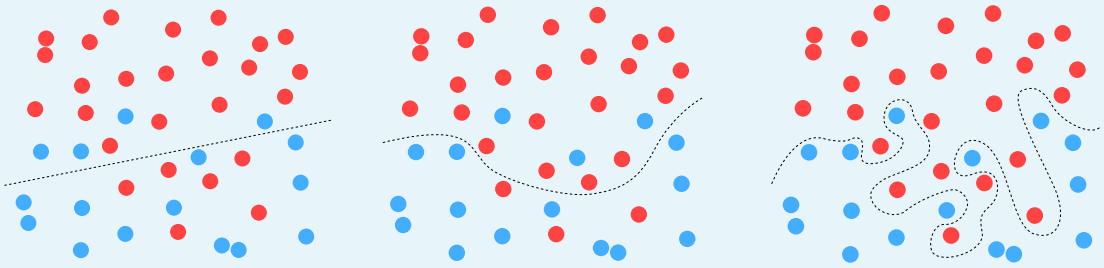
where $\mathcal{F}$ is the search space, $\mathcal{T}$ is the training set of size N consisting of pairs of inputs and their corresponding labels, and $\mathcal{L}$ is the error measuring function, also called *loss*. In general, this optimization problem is non-convex and a closed-form solution is seldom available. Iterative optimization methods are thus used to find a suitable solution in the search space that achieves acceptable performance, a global optimum being difficult, if not impossible, to attain. Note that a solution close to the global optimum doesn't necessarily align with the initial generalization objective. Indeed, the algorithm can overfit

to the training data which is analogous to learning by heart all the solutions to the already seen inputs while still struggling with unseen ones. Solving this problem is a field of study of its own and solutions usually consist of a combination of regularization techniques that include but are not limited to having a big and diverse enough training set, using a validation set, adding a regularization term to the loss, etc.

Highlight 2.1: Underfitting and overfitting

Underfitting happens when the model is too simple and not flexible enough (e.g., using a linear model for a non-linear problem) to successfully learn the relationship between the inputs and their labels, resulting in a high loss on both the training and the test set.

On the other hand, overfitting happens when the model is too flexible and learns to perfectly associate the training inputs to their labels, but fails to do so on the test set.



(a) Underfitting (b) Optimal fitting (c) Overfitting

Figure 2.2

Unsupervised learning is another form of ML where the algorithm has access to a training set of inputs deprived of their labels. Indeed, in many scenarios, labels are difficult to obtain and for many reasons. In such cases, approaches not requiring them are of particular interest. The algorithm is then expected to learn on its own by unveiling the hidden structure of the data and finding patterns that it could exploit to predict the correct answers.

Among the tasks that an algorithm can be trained to perform, *classification* and *regression* are certainly the most popular. In the former, the expected outputs are limited to a countable set of possibilities. Usually, the complexity

of a classification task is correlated with the number of possibilities, called classes, that the algorithm has to decide between. A textbook example of a classification task is that of determining if an image is that of a cat or a dog. This special case is called binary classification since there are only two classes to decide between and is of limited complexity. Had there been more classes, the complexity would've been greater.

On the other hand, regression can be seen as the limit case when the number of classes is infinite. Indeed, it consists of predicting an output in a continuous space. Following the logic above, a regression task is more challenging than a classification one.

ML models can also be classified into *parametric* and *nonparametric* models. Parametric models usually have a fixed number of parameters independent from the data and define a mapping of a given form that connects the inputs to the outputs. Non-parametric models on the other hand are more flexible as the learned mapping is not predefined and the number of parameters grows with the size of the training set.

2.2 Popular algorithms

In this section, a review of three of the most popular ML algorithm is given.

2.2.1 Linear regression

Linear regression is the simplest regression algorithm. It amounts to computing a scalar output $\hat{y} \in \mathbb{R}$ as a linear combination of the elements of the corresponding vector input $\mathbf{x} \in \mathbb{R}^{n+1} = (x_0, \ldots, x_n)$ with $x_0 = 1$.

$$\hat{y} = \mathbf{w}^\top \mathbf{x}, \tag{2.2}$$

where $\mathbf{w} \in \mathbb{R}$ is the vector containing the parameters of the model, it is thus a parametric one. The objective is to find the configuration that minimizes the square error between $\hat{y}$ and the corresponding label y, called residual, on the

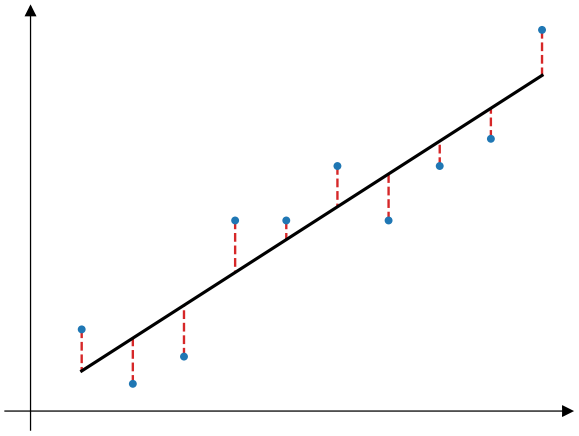


Figure 2.3 – Example of a linear regression model fitted on the blue points. The dashed lines correspond to the residuals.

training set. The optimization problem is defined as

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \mathcal{L}_{\text{MSE}}(y, \hat{y}) = \underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{N} \sum_{i=0}^{N} (\underbrace{\hat{y}_i - y_i}_{\text{residual}})^2. \tag{2.3}$$

This form of the loss is called the *mean squared error* (MSE). It can be shown that this problem is convex and has a closed-form solution $\mathbf{w}^\star$, called the *least squares* (LS) estimator [1]. Suppose that the training inputs and are organized as rows of a matrix $\mathbf{X} \in \mathbb{R}^{N \times n+1}$ and the training label as the elements of a vector $\mathbf{y} \in \mathbb{R}^N$. Then

$$\mathbf{w}^\star = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}. \tag{2.4}$$

2.2.2 k-nearest neighbors

The k-nearest neighbors (k-NN) algorithm is a simple and straightforward non-parametric learning algorithm where the prediction associated with a

1. The LS channel estimator of section 1.2.1 is based of the generalization to the multidimensional case.

given input depends on the k closest elements in the training set. Although it can be used for regression, it is usually used as a classification model. Accordingly, for a given dataset of input points and corresponding labels representing the classes, the k closest elements of a new unseen input are first retrieved and the resulting class is computed from a simple majority vote.

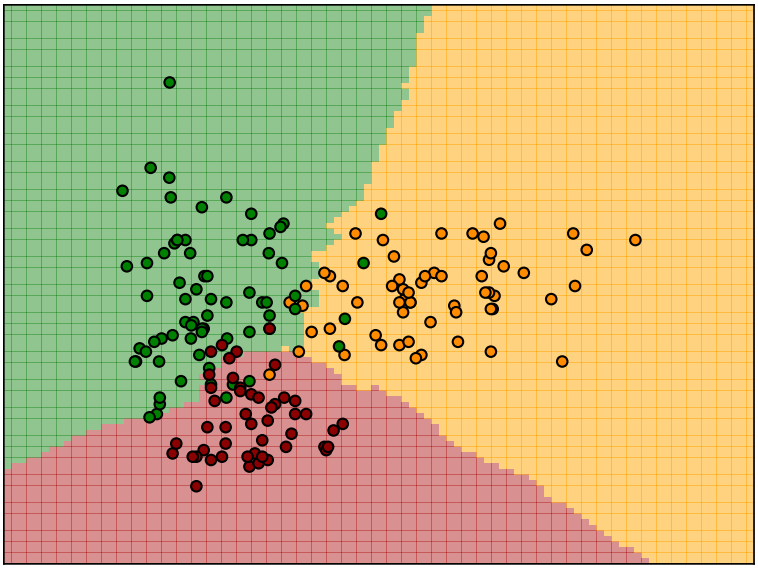


Figure 2.4 – Example of a k-NN classification model fitted on points classified into three classes. The colored regions show the decision boundaries of the model.

A variant of the k-NN algorithm is the fixed-radius near neighborhoods algorithm that, instead of considering the k closest neighbors, rather considers the neighbors in the training set within a given fixed distance from the input.

2.2.3 Support vector machine

Another prominent algorithm is *support vector machine* (SVM). Like k-NN, it can be used for both classification and regression. In classification problems, it aims at finding a decision boundary that maximizes the margin between the

different classes. This amounts to constructing the hyperplane in the input space that has the largest distance to the nearest training points. The points that lie on (or within for non-linearly separable problems) the margin are called *support vectors*. The most basic version of the algorithm (called hard margin) is defined for the binary classification case for linearly separable problems. For inputs $\mathbf{x}_i$ and corresponding labels $y_i \in \{-1, 1\}$, the decision $\hat{y}$ for an unseen input $\mathbf{x}$ is given by

$$\hat{y} = \begin{cases} -1 & \text{if } \mathbf{w}^\top \mathbf{x} + b < 0, \\ 1 & \text{if } \mathbf{w}^\top \mathbf{x} + b \geq 0 \end{cases}, \tag{2.5}$$

where $\mathbf{w}$ and b are the parameters of the model. The optimization problem that SVM solves is

$$\begin{aligned} &\underset{\mathbf{w},b}{\operatorname{argmin}} && \frac{1}{2}\mathbf{w}^\top \mathbf{w} \\ &\text{subject to} && y_i(\mathbf{w}^\top \mathbf{x}_i + b) \geq 1 \text{ for } i = 1, \ldots, N \end{aligned} \tag{2.6}$$

This quadratic programming problem can be solved using usual solvers.

The basic version of SVM can be extended to problems that aren't linearly separable either by allowing a small number of training points to be falsely classified (called soft margin), or by using the kernel trick (highlight 2.2). In addition, extension to the multi-class case can be done by using a "one-versus-all" strategy where the decision is made based on multiple binary SVM classifiers trained each time by considering points belonging to one class versus the rest of the points.

Highlight 2.2: The kernel trick

The linear version of SVM can be extended to be non-linear using a so-called kernel trick. It stems from the fact that the basic formulation of the problem can be re-written as

$$\mathbf{w}^\top \mathbf{x} + b = b + \sum_{i=1}^{N} \alpha_i \mathbf{x}^\top \mathbf{x}_i, \tag{2.7}$$

where α_i are the dual coefficients related to (2.6). This form reveals the underlying dot products between the input and the training points. The trick consists in replacing it with a kernel $K(\mathbf{x}, \mathbf{x}_i) = \phi(\mathbf{x})^\top \phi(\mathbf{x}_i)$ for a given feature mapping ϕ. Doing so is analogous to first computing the features for all training then applying standard SVM. In this case, the decision function becomes $b + \sum_{i=1}^{N} \alpha_i K(\mathbf{x}, \mathbf{x}_i)$ which is non-linear with respect to $\mathbf{x}$. Some of the popular kernels are the polynomial kernel (i.e., $K_{\text{poly}}(\mathbf{x}, \mathbf{x}_i) = (\mathbf{x}^\top \mathbf{x}_i + c)^d$) and the radial basis function (RBF) kernel (i.e., $K_{\text{RBF}}(\mathbf{x}, \mathbf{x}_i) = \exp(-\gamma \|\mathbf{x} - \mathbf{x}_i\|_2^2)$).

2.3 Deep learning

Deep learning (DL) is a subfield of ML that has gathered a lot of attention in the recent years, and for good reason. It has revolutionized many fields such as healthcare, autonomous vehicles, finance, and beyond. Deep neural networks (NNs), which are models inspired by the structure and functioning of the human brain, are its backbone.

One of the most remarkable aspects of DL is its ability to automatically learn hierarchical features and representations from data. In traditional machine learning, feature engineering often requires substantial manual effort, where domain experts craft features to feed into algorithms. However, DL models, such as convolutional neural networks (CNNs), recurrent neural networks (RNNs) and Transformers, can automatically learn relevant features directly

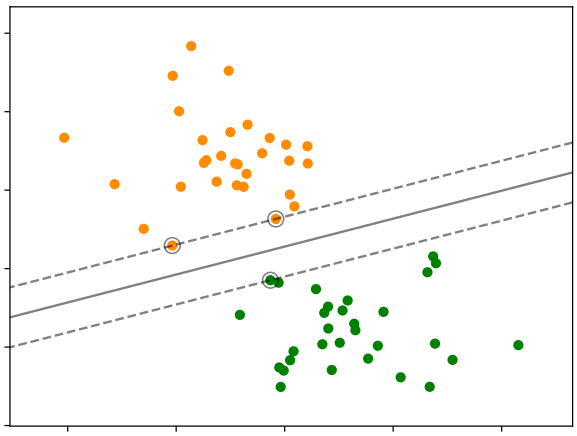


Figure 2.5 – Example of an SVM classifier fitted on a 2D binary problem. The continuous line represent the decision boundary while the dashed one represent the margins. The circled points are the support vectors.

from raw data. This automation significantly reduces the need for domain-specific knowledge and time-consuming feature engineering.

2.3.1 Neural networks

Deep NNs consist of compositions of alternating linear and nonlinear mappings. Essentially, each linear mapping f_i is followed by a nonlinear mapping a_i, also called activation function. The composition of the two $a_i \circ f_i$ constitute the ith layer. Typically, the number of layers L is high, which is why the models are said to be deep, as it was shown that they achieve better performance than their shallower counterparts (Simonyan and Zisserman 2015). The linear mappings[2] $f_1, \ldots, f_L$ are defined as

$$f_i(\mathbf{x}) = \mathbf{W}_i^\top \mathbf{x} + \mathbf{b}_i, \tag{2.8}$$

where $\mathbf{W}_i$ and $\mathbf{b}_i$ are the learnable parameters of the layer, also called *weights*. In addition, the size of $\mathbf{b}_i$ is called the width of the layer. The role of activation

2. "Linear mappings" is an abuse of terminology to refer to mappings that are in reality affine.

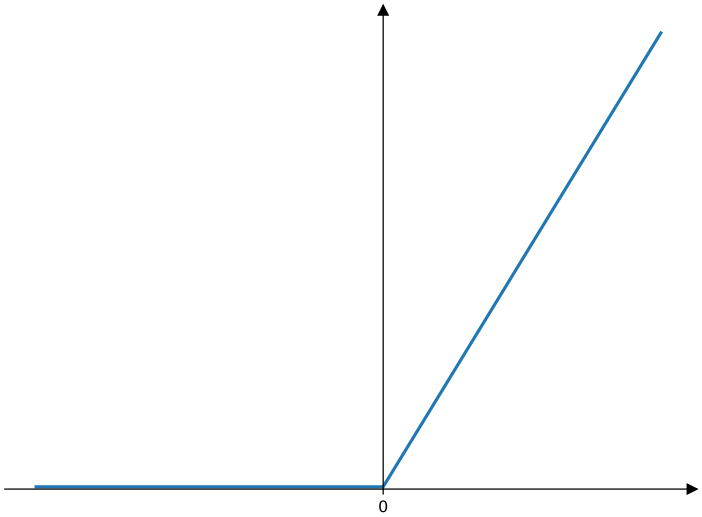


Figure 2.6 – Plot of the ReLU activation function.

functions become clear when considering layers consisting of only the linear parts. Indeed, it can be easily shown that a chaining of linear functions is a linear function in itself, making it useless. Activation functions serve the role of introducing a nonlinear aspect to the network. The choice of adequate ones is greatly discussed in the literature (Dubey, Singh, and Chaudhuri 2022). The rectified linear unit (ReLU) (Nair and Geoffrey E. Hinton 2010) is probably the most popular one as most modern NN architectures rely on it (or at least variations of it). It is defined as $\sigma_{\text{ReLU}}(\mathbf{x}) \triangleq \max(0, \mathbf{x})$ where $\max$ is taken piecewise. Figure 2.6 shows its plot.

The last layer L of the network is called the output layer and is in charge of computing the result of the model. The remaining layers are called hidden layers and are in charge of computing the intermediate results of the network.

It is worth noting that NNs are often represented as graphs as it facilitates the understanding of certain concepts. It is also a closer representation of how we visualize the interconnections of neurons in the brain. As a consequence, artificial neurons constitute the nodes of the graph and each one corresponds to an element of the output vector of a given layer. On the other hand, the edges connecting the neurons correspond to the elements of the multiplication matrix. Figure 2.7 shows an example of a layer of a NN.

Similarly to the ML algorithms introduced above, the training consists of

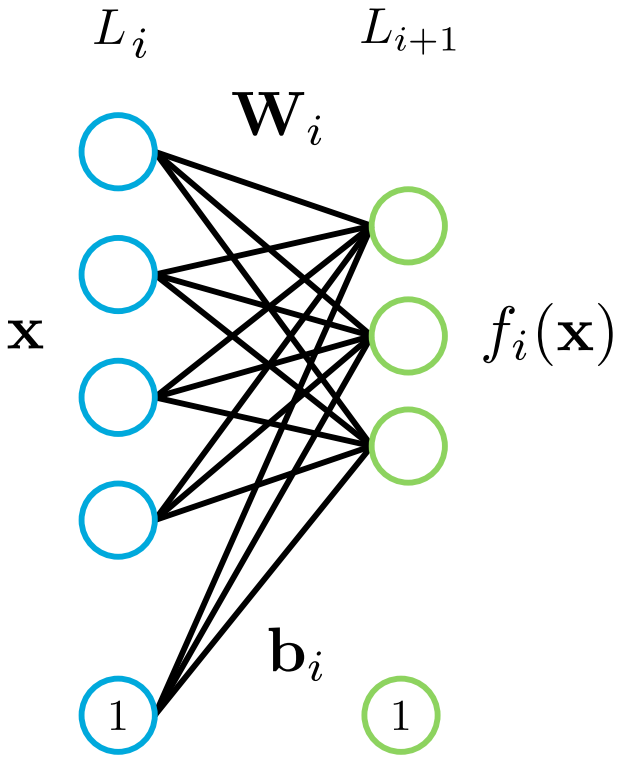


Figure 2.7 – Graph representation of an NN layer.

minimizing a loss between the output and the corresponding label by tweaking the model's weights. This is done by combining two techniques: *gradient descent* and *backpropagation*. The former is an iterative optimization algorithm and the latter is what enable the training of weights of the hidden layers. Both are described in the following sections. Before continuing, it's important to understand what makes NNs so special compared to other algorithms and what explains there great success. This could be greatly attributed to the fact that they are *universal approximators* (Hornik, Stinchcombe, and White 1989; Cybenko 1989). It means that provided some conditions, they can approximate virtually any function.

Highlight 2.3: The universal approximation theorem

In a simple and non-formal way, the universal approximation theorem states that a NN with at least one hidden layer can approximate any continuous function to a reasonable degree of accuracy given a wide hidden layers and certain conditions on the activation functions.

This theorem, however, doesn't explain how NNs should be trained in order to achieve the desired accuracy for a given function, and rather only guaranties the existence of such a network.

2.3.2 Gradient descent

NNs do not have the luxury of having a closed-form solution to their minimization problem as the loss is usually non-convex with respect to the learnable parameters. A different approach from the ones described in the previous sections should thus be considered. Gradient descent is an iterative optimization technique where in each iteration the algorithm updates the learnable parameters of the model to be closer to the global optimum. It does so by relying on the derivative of the loss with respect to the parameters. Indeed, for a minimization problem, the negative gradient indicates its "direction of fastest decrease" locally, meaning that taking a small step in that direction leads to a smaller value of the loss. In other words, for a loss function $\mathcal{L}$ and parameters θ, an iteration of the algorithm corresponds to computing

$$\theta_{n+1} = \theta_n - \gamma \nabla \mathcal{L}(\theta_n), \tag{2.9}$$

where θ_n is the result of the algorithm at the nth iteration, and γ is the learning rate (i.e., what controls the size of the step taken). Consequently, $\mathcal{L}(\theta_{n+1}) \leq \mathcal{L}(\theta_n)$ for a small enough γ. Successive iterations of the algorithm should thus lead to a local minimum.

Gradient descent is not limited to deep learning and is a rather popular algorithm for all sorts of optimization problems. However, NNs rely on a specific variant called *stochastic gradient descent* (SGD). Since the optimization prob-

lems considered in deep learning often consider the loss across all the training set, the gradient computation has to be taken for each individual input. This rapidly becomes a problem for large dataset since the computational complexity for one step of the algorithm becomes proportional to its size. A solution is to only consider a small subset of the whole dataset, called a *minibatch*, for each iteration. This works because averaging the gradient over a minibatch can be seen as approximating its expectation and is a reasonable enough assumption. The quality of the approximation is directly tied to its size, so a compromise should be sought.

Note that modern architectures don't use SGD as is, but rather rely on more advanced versions that try to solve some of its limitations that are out of the scope of this manuscript. Some examples of such algorithms are ADAM (Kingma and Welling 2014), AdaGrad (Duchi, Hazan, and Singer 2011) and RMSProp.

2.3.3 Backpropagation

Backpropagation (Goodfellow, Bengio, and Courville 2016, Chapter 6) refers to the algorithm used by NNs to compute the gradient of each layer. While analytically computing each one individually is possible, it is not an efficient way of doing so. A more efficient way of handling the problem however stems from the observation that since a NN is a composition of multiple functions, one can apply the chain rule to compute the derivatives up to the desired point in the chain. The idea is to work in a reverse order, starting from the output layer and moving backward through the network, computing the gradient of the loss with respect to the output of each layer. This iterative process efficiently propagates errors backward through the network. By applying the chain rule, the gradients of the composed functions within each layer are efficiently computed and then reused for the preceding layer.

2.3.4 Model-based deep learning

Although DL helps reduce the need for feature engineering and domain-specific knowledge, it comes at the cost of complex models often computationally intensive and data hungry. Handcrafted methods on the other hand are

often designed to be computationally frugal at the cost of greater approximations, leading to less precise models. Model-based deep learning (Shlezinger et al. 2020) is a new paradigm with the goal of finding a compromise between pure model-based approaches and pure DL approaches. It does so by incorporating prior knowledge of the problem at hand into the design of NNs. The obtained hybrid networks are thus specialized and as a consequence smaller in size than their model-agnostic counterparts. A training phase is still needed, but the amount of data that is needed to achieve acceptable performance is often smaller when compared to traditional DL approaches.

Model-based DL takes many forms. It can be implemented in many ways and for a large set of problems. Nevertheless, certain techniques can often be reused across many scenarios that are governed by similar principles. *Deep unfolding* (or sometimes *deep unrolling*) in particular has seen a lot of success recently. The main idea behind it is to convert an iterative algorithm into a NN by making each layer or group of layers reproduce the computations of a single iteration. The act of stacking these layer one in front of the other is called unfolding, and passing an input through the resulting network is equivalent to executing successive iterations of the original algorithm. The benefits of employing this technique are on multiple levels:

— The weights of the NN correspond to the parameters of the original iterative algorithm. They can thus benefit from an optimal initialization when one is available.

— These parameters, otherwise fixed, are now “free”, meaning that they can be learned by training the NN on some data.

— The parameters of different layers are allowed to be different, meaning that, when considering the NN in its iterative form, two successive iterations will always compute the same operations but can have different parameters.

— The obtained networks are much more interpretable than generic black box ones, offering an enlightening view into the otherwise hermetic models.

Deep unfolding was originally proposed in (Gregor and LeCun 2010) where the iterative sort thresholding algorithm (ISTA) was unfolded for sparse cod-

ing. The idea has since been adopted by several subsequent works for many different applications (see (Monga, Li, and Eldar 2021) and references therein).

2.4 Dimensionality reduction

With the abundance of data comes another challenge. When the dimensionality of data becomes high, traditional low-dimensional methods tend to fail. The idea of dimensionality reduction comes from the so-called *manifold hypothesis*.

> **Highlight 2.4: Manifold hypothesis**
>
> High dimensional real-world data are organized close to a lower-dimensional manifold (a space that can be locally approximated by a subspace at any point), called the embedding space or latent space. Their degrees of freedom are in reality reduced to a small limited set representing their variation directions.

Dimensionality reduction is closely connected to the concept of manifold learning. Although a manifold is a well defined mathematical concept, it tends to be loosely defined as a set of points lying on a “curved surface”. One such example of a low-dimensional manifold embedded in a higher-dimensional space is the famous Swiss roll toy example represented on figure 2.8. It can be observed that although its composed of points in a 3-dimensional space, they all lie on 2-dimensional "rolled sheet". Unrolling that sheet and representing it in its true underlying dimensionality is the ultimate objective of dimensionality reduction and manifold learning (Goodfellow, Bengio, and Courville 2016, Section 5.11.3) in particular.

In practice, doing so helps reduce the complexity of a given problem. If a high-dimensional measured quantity can in fact be explained by a small number of variables, then surely the phenomenon it stems from can be explained by a simple enough model. Inspired by this observation, a lot of data processing techniques try to approximate the underlying manifold (Carlsson 2009; Peyré 2009).

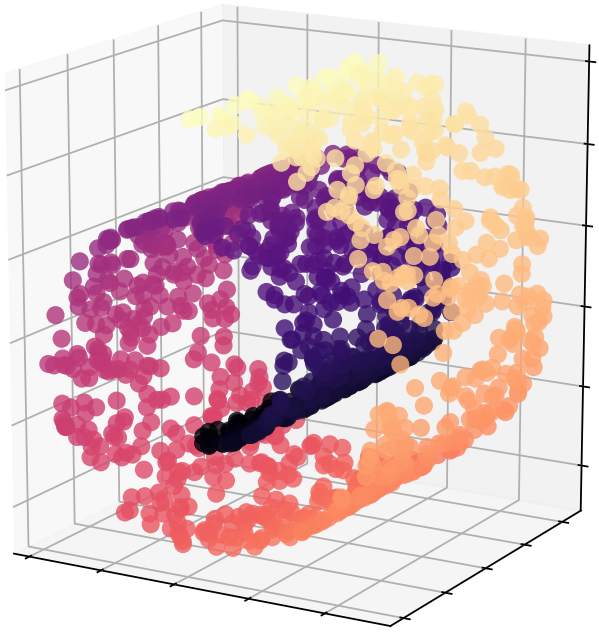


Figure 2.8 – Swiss roll.

In the following, an overview of popular parametric and nonparametric dimensionality reduction and manifold learning methods is given.

2.4.1 Nonparametric DR

PCA

Principal components analysis (PCA) is maybe the most widespread and straightforward dimensionality reduction method. It is a linear one as it only requires linear algebra operations and is useful for tasks like data compression and data visualization. It has found applications in many fields such as biology, chemistry and engineering. The gist of it is to project high-dimensional

data on a lower-dimensional space while retaining the most useful information.

Formally, consider a matrix $\mathbf{X} \in \mathbb{R}^{n \times D}$ containing n data points of dimension D with zero column sample mean. The objective of PCA is to find its *principal components*, i.e., orthogonal directions maximizing its variance, or alternatively minimizing the residuals between the original data and their projection. The first step is to find the 1^{st} component, the one with the largest variance. The following component is the one maximizing the variance on the remaining directions. Subsequent directions are extracted in a similar manner by maximizing the variance on directions remaining from the preceding components. When the total number of components is equal to D, then applying PCA amounts to a change of basis thanks to their orthogonality. It can be mathematically written as

$$\mathbf{X}_{\text{PCA}} = \mathbf{X}\mathbf{W}, \tag{2.10}$$

where $\mathbf{W} \in \mathbb{R}^{D \times D}$ is the orthogonal column matrix of principal components and $\mathbf{X}_{\text{PCA}}$ is the set of point projected onto the new basis. Let's denote $\mathbf{w}_k$ the kth component. Note that components are chosen to be of unit norm.

First principal component The first component $\mathbf{w}_1$ is the one maximizing variance across the data points, thus

$$\mathbf{w}_1 = \underset{\mathbf{w}}{\operatorname{argmax}} \|\mathbf{X}\mathbf{w}\|_2^2 = \underset{\mathbf{w}}{\operatorname{argmax}}\, \mathbf{w}^\top \mathbf{X}^\top \mathbf{X}\mathbf{w},$$

where $\mathbf{X}^\top\mathbf{X}$ is the covariance matrix. It can be proven that to maximize this quantity under the constraint $\|\mathbf{w}\|_2 = 1$, $\mathbf{w}_1$ should be chosen to be the eigenvector corresponding to the largest eigenvalue.

Subsequent principal components The kth component is the one maximizing the variance from the remaining directions

$$\mathbf{w}_k = \underset{\mathbf{w}}{\operatorname{argmax}} \left\| \left(\mathbf{X} - \sum_{i=1}^{k-1} \mathbf{X}\mathbf{w}_i \right) \mathbf{w} \right\|_2^2.$$

Similarly to the first principal component, it can be shown that the vector maximizing this quantity under $\|\mathbf{w}\|_2 = 1$ corresponds to the eigenvector

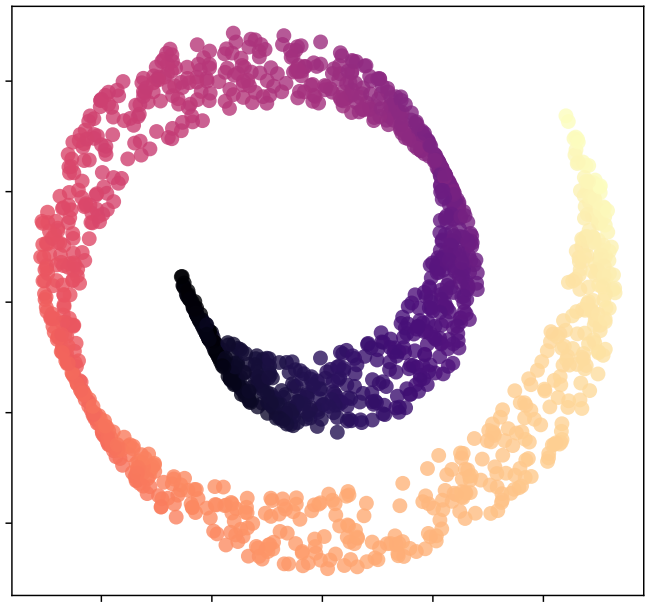


Figure 2.9 – PCA applied to the Swiss roll data.

associated with the kth eigenvalue.

All in all, *PCA amounts to projecting the data onto the basis of eigenvectors of their covariance matrix*. If information is indeed concentrated on a small set of directions, and since the principal components are organized in a decreasing order with regards to the variance, it makes sense to only keep the d first components resulting in d-dimensional points. The choice of the dimension of the embedding space d depends on the desirable compromise between data compression and dimensionality reduction. Figure 2.9 shows the result of applying PCA to the Swiss roll. Only two dimensions are kept from the original 3D data. The first principal component corresponds to the x-axis and the second to the y-axis.

MDS

Another well known dimensionality reduction method is multidimensional scaling (MDS). The main idea behind it is to find a space of lower dimension than the original space that preserves as much as possible the pairwise distances of data points, or more generally their dissimilarities. Formally, for a

set of n points organized in a matrix $\mathbf{X} \in \mathbb{R}^{n \times D}$, MDS tries to learn a matrix $\mathbf{Y} \in \mathbb{R}^{n \times d}$ of their lower-dimensional projections (i.e., $d < D$) so that

$$d_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}_j) \approx d_{\mathbf{y}}(\mathbf{y}_i, \mathbf{y}_j), \tag{2.11}$$

where $\mathbf{x}_i, \mathbf{x}_j$ and $\mathbf{y}_i, \mathbf{y}_j$ are rows of $\mathbf{X}$ and $\mathbf{Y}$ respectively. $d_{\mathbf{x}}(.,.)$ is a distance measure defined on the original space while $d_{\mathbf{y}}(.,.)$ is usually taken to be the Euclidean distance. Many variants of the algorithm exist. The classical one for instance is analogous to PCA, where the dissimilarity measure is chosen to be the covariance. Here, a more general version, namely metric MDS, is presented. It consists of finding $\mathbf{Y}$ such that

$$\mathbf{Y} = \underset{\mathbf{Y}}{\operatorname{argmin}} \sqrt{\sum_{i,j} \left(d_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}_j) - \|\mathbf{y}_i - \mathbf{y}_j\|_2\right)^2}, \tag{2.12}$$

Since no closed-form solution exists for this general form of the problem, iterative optimization algorithms like gradient descent or Newton's method can be used. This makes metric MDS a nonlinear method, contrary to its classical version and thus PCA. Figure 2.10 shows the result of applying MDS to the Swiss roll using the scaling by majorizing a complicated function (SMACOF) algorithm (Leeuw 1977) to solve the optimization problem.

Sammon's mapping (SM) is a special case of MDS. The function it tries to minimize is a weighted version of the one in (2.12) and is defined as

$$\mathbf{Y} = \underset{\mathbf{Y}}{\operatorname{argmin}} \frac{1}{\sum_{i,j} d_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}_j)} \sqrt{\sum_{i,j} \frac{\left(d_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}_j) - \|\mathbf{y}_i - \mathbf{y}_j\|_2\right)^2}{d_{\mathbf{x}}(\mathbf{x}_i, \mathbf{x}_j)}}. \tag{2.13}$$

Isomap

Another special case of MDS is Isomap (Tenenbaum, Silva, and Langford 2000). It tries to approximate the geodesic distances on the manifold of data. It builds on the idea that distances on manifolds can be locally approximated by the Euclidean distance. This approximation only holds for relatively small neighborhoods around a given point.

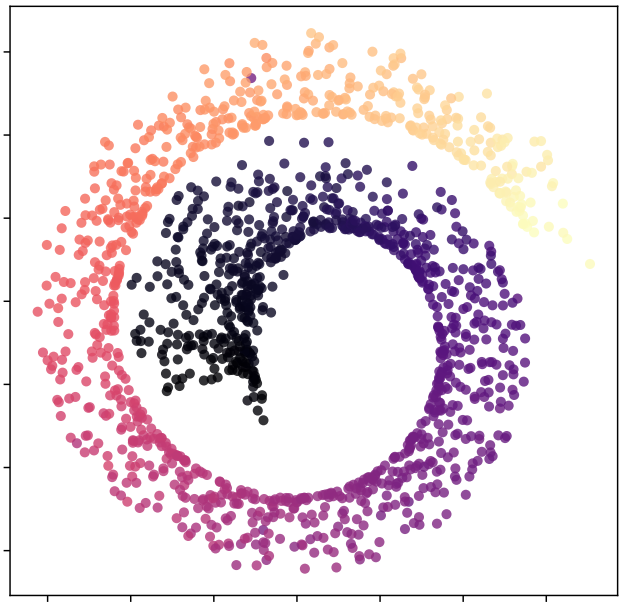


Figure 2.10 – MDS applied to the Swiss roll data.

Highlight 2.5: Geodesic

A geodesic is the shortest path between two points on a manifold such that the path lies on it.

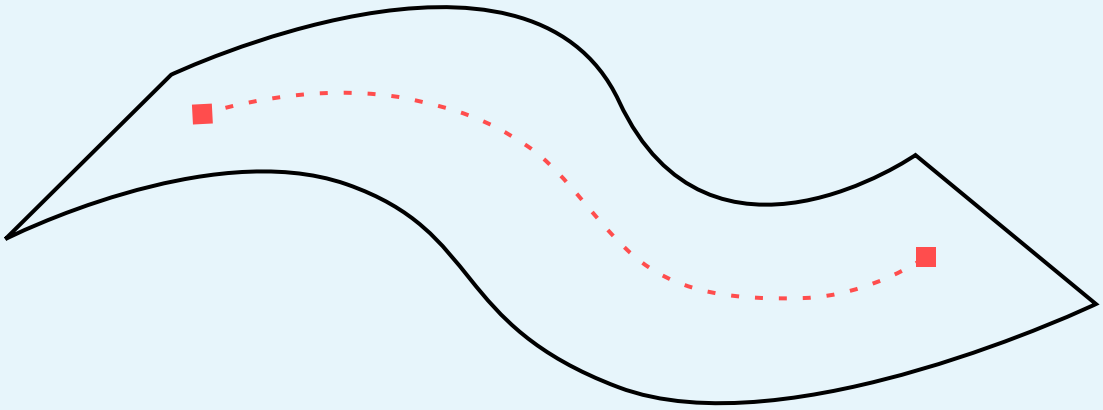


Figure 2.11 – The dashed line represent the geodesic between the square points on the manifold.

Figure 2.12 shows the difference between the geodesic and the Euclidean

distance in the original space. When the two points are close on the manifold, their Euclidean distance is a good approximation to their geodesic one, as show in green. Conversely, the Euclidean distance between points far away from each other on the manifold are usually bad approximations, as shown in red.

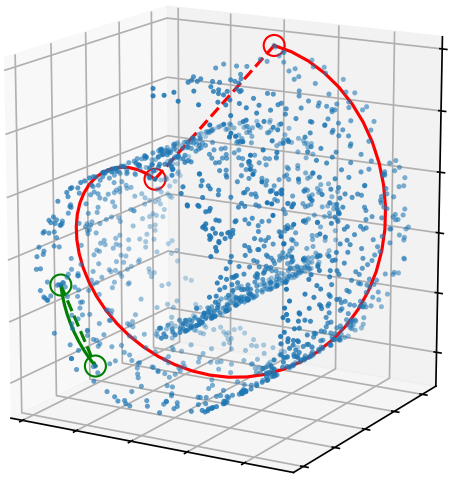
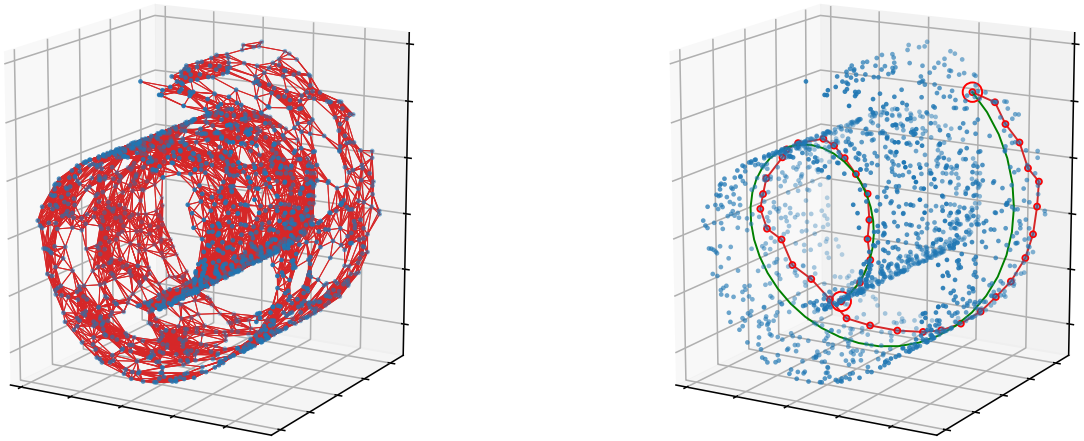


Figure 2.12 – Visualization of the geodesic (solid line) and the Euclidean distance (dashed line) between two close points and two far points on the Swiss roll.

Like MDS, Isomap starts by computing the pairwise distance matrix. However, instead of keeping all distances, it only keeps the k-smallest ones for each entry. This way, high distances representing points outside of their immediate neighborhoods, and thus poorly estimated, are eliminated. The obtained sparse distance matrix is a representation of a k-NN graph[3]. Figure 2.13a shows as visualization of the graph computed on the Swiss roll. The next step is to estimate the geodesic distances of the eliminated entries. This is done by following the shortest path on the graph between two given points by using traditional algorithms like Dijkstra's. A visualization of the shortest path

3. Alternatively, a fixed-radius near neighbors graph could also be used.



(a) (b)

Figure 2.13 – (a) Visualization of the k-NN graph on the Swiss roll. (b) Visualization of the geodesic and its estimation following the shortest path on the graph between two points on the Swiss roll.

between two points on such a graph along with the true geodesic is shown on figure 2.13b. The shown path was chosen to be particularly long and, although not exact, it is still a much better approximation than the Euclidean distance in the high-dimensional space.

The final step is to apply MDS to the newly constructed geodesic distance matrix to obtain the low-dimensional embeddings. Figure 2.14 shows the results of applying Isomap to the Swiss roll. Compared to MDS, it successfully unrolls the underlying "sheet" on which the data points lie.

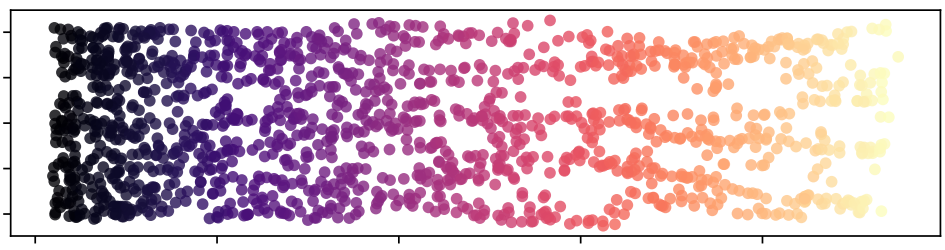


Figure 2.14 – Isomap applied to the Swiss roll data.

t-SNE

t-distributed stochastic neighbor embedding (t-SNE) (Maaten and G. Hinton 2008) is another nonlinear dimensionality reduction method that aims at projecting high-dimensional data into a 2D or 3D space[4]. It frames the problem of preserving the distances in the embedding space as a probabilistic one. It is composed of two main steps, the first of which constructs a joint probability distribution of the original points as well as the embedding space such that close pairs are assigned a higher probability. The second step tries to solve an optimization problem where to goal is to minimize the Kullback-Leibler (KL) divergence of the joint probabilities between the original and embedding space.

Consider a set of n points organized in a matrix $\mathbf{X} \in \mathbb{R}^{n \times D}$. Define the joint probability distribution on the set of original points such that

$$p(\mathbf{x}_i, \mathbf{x}_j) = \frac{1}{2D} \left(\frac{e^{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2}{2\sigma_i^2}}}{\sum_{k \neq i} e^{\frac{-\|\mathbf{x}_i - \mathbf{x}_k\|_2^2}{2\sigma_i^2}}} + \frac{e^{\frac{-\|\mathbf{x}_j - \mathbf{x}_i\|_2^2}{2\sigma_j^2}}}{\sum_{k \neq j} e^{\frac{-\|\mathbf{x}_j - \mathbf{x}_k\|_2^2}{2\sigma_j^2}}} \right), \tag{2.14}$$

where σ_i and σ_j are the variances of the Gaussians centered on $\mathbf{x}_i$ and $\mathbf{x}_i$ respectively[5], and $p(\mathbf{x}_i, \mathbf{x}_i) = 0$. It is a symmetrized conditional probability modeled as a Gaussian distribution. Note that $\sum_{i,j} p(\mathbf{x}_i, \mathbf{x}_j) = 1$. Similarly, define the joint probability distribution on the set of embeddings $\mathbf{Y} \in \mathbb{R}^{n \times d}$ such that

$$q(\mathbf{y}_i, \mathbf{y}_j) = \frac{(1 + \|\mathbf{y}_i - \mathbf{y}_j\|_2^2)^{-1}}{\sum_{k \neq l} (1 + \|\mathbf{y}_l - \mathbf{y}_k\|_2^2)^{-1}}, \tag{2.15}$$

and $q(\mathbf{y}_i, \mathbf{y}_i) = 0$. The chosen distribution is a Student t-distribution with a single degree of freedom.

Finally, the minimization problem is defined as

$$\mathbf{Y} = \underset{Y}{\operatorname{argmin}} \operatorname{KL}(P \| Q) = \sum_{i \neq j} p(\mathbf{x}_i, \mathbf{x}_j) \log \frac{p(\mathbf{x}_i, \mathbf{x}_j)}{q(\mathbf{y}_i, \mathbf{y}_j)}. \tag{2.16}$$

Using the KL divergence is the most intuitive way of representing the dissim-

4. This is a limitation of the Barnes-Hut implementation of t-SNE.
5. The method for determining the variances involve a binary search to find the solutions that lead to a value of the entropy determined by the user. More details can be found in the original paper.

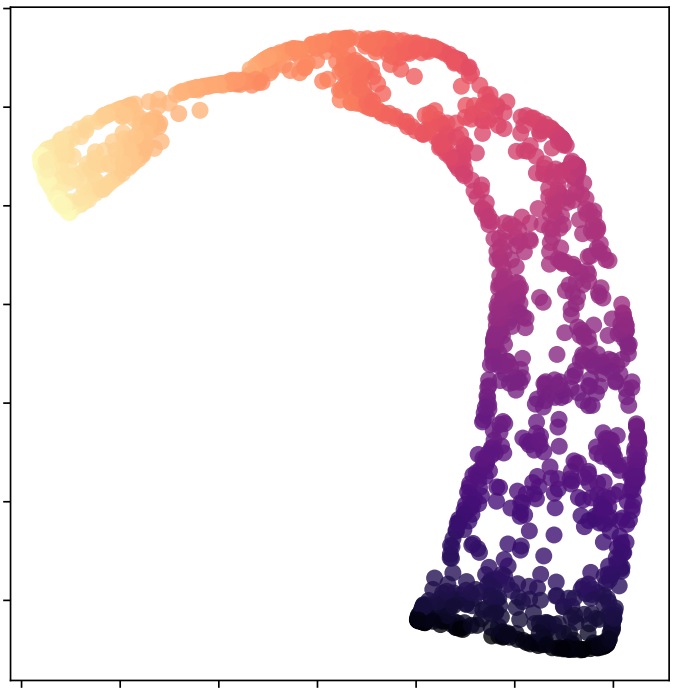


Figure 2.15 – t-SNE applied to the swiss roll data.

ilarity between the two spaces from a probabilistic point a view. However, this time again the problem is non-convex, meaning that iterative optimization methods are once again needed such as gradient descent which is what was used when the method was initially introduced. Figure 2.15 shows the result of applying t-SNE to the Swiss roll.

2.4.2 Parametric DR

Autoencoders

Autoencoders are NNs with an architecture particularly suitable for dimensionality reduction. They can be decomposed into two sub-networks: an *encoder* and a *decoder*. The goal of the encoder is to learn a mapping from the input space to a latent space, while the decoder tries to learn the inverse mapping, i.e., going from the latent space to the output space. Autoencoders were originally used in the context of unsupervised learning where the training goal is to reconstruct the original input. While it might seem useless at first since a

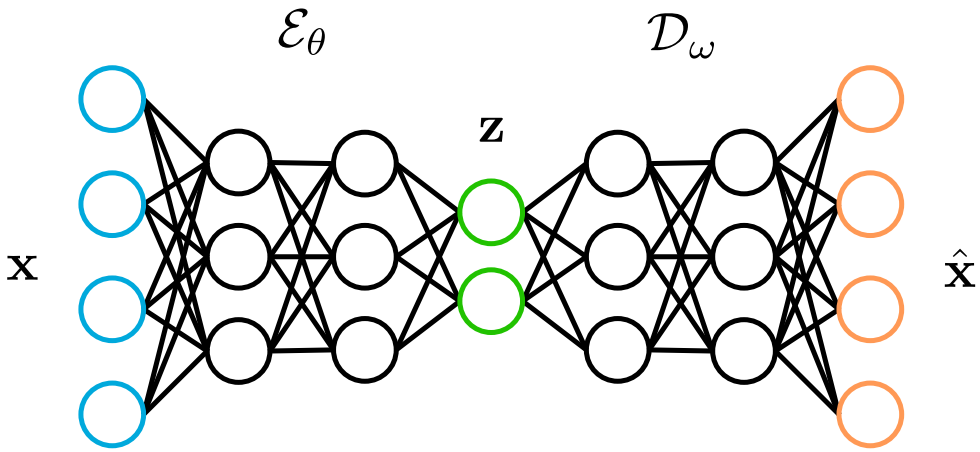


Figure 2.16 – Autoencoder.

simple identity mapping allows for a perfect reconstruction, it is when looking at the constraints imposed on the architecture of the model that the benefits become clear. Indeed, the latent space, as suggested by its name, is taken to be of lower-dimension than the input space. Such a bottleneck forces the model to concentrate information on a small number of variables through the encoding process. The decoder then tries to reconstruct the input from the projection on the latent space.

Formally, an autoencoder consists of two parametric mappings $\mathcal{E}_\theta$ and $\mathcal{D}_\omega$ where

$$\mathcal{D}_\omega(\mathcal{E}_\theta(\mathbf{x})) = \mathcal{D}_\omega(\mathbf{z}) = \hat{\mathbf{x}}, \tag{2.17}$$

where $\mathbf{z}$ is the latent projection of $\mathbf{x}$, and $\hat{\mathbf{x}}$ is the reconstructed input. The autoencoder is trained to minimize

$$\mathcal{L}(\mathbf{x}, \mathcal{D}_\omega(\mathcal{E}_\theta(\mathbf{x}))) = \|\mathbf{x} - \mathcal{D}_\omega(\mathcal{E}_\theta(\mathbf{x}))\|_2^2, \tag{2.18}$$

on all training example. The decoder is often a mirrored version of the encoder. Nonetheless, the weights are not tied.

Autoencoders exist in many flavors. From k-sparse autoencoders (Makhzani and Frey 2014) where a constraint on the l_0-norm of the latent vector is imposed, to stochastic autoencoders like variational autoencoders (VAEs) (Kingma and Welling 2014).

Siamese networks

Traditional learning approach generally rely on evaluating the error between an input and a target. The error is quantified by computing a loss function that takes as input the output of the model and a target. However, this approach cannot work if a target is not available, or if it's available in a form that can't be exploited directly. A different learning paradigm, called *contrastive learning* (Jaiswal et al. 2020), suggests a workaround that is especially useful for representation learning. The goal is to learn a mapping to an output space where data pairs that are similar stay close, while dissimilar ones are far apart. The similarity between pairs is what guides training. For example, when the training data consists of input vectors classified into a number of classes, contrastive learning can be used to learn low-dimensional representations where instances of the same class are close together and far away from the rest.

Siamese networks are an example of a contrastive learning approach. It consists of a an NN that computes the output of a pair of inputs, resulting in a pair of outputs[6]. A dissimilarity measure is computed from the pair, and is given as input to a loss function. Its other input if the "target" dissimilarity, i.e., the one obtained from the data. Training the networks consists in minimizing the error between the computed dissimilarity and the target one.

Siamese network were first proposed for handwritten signature verification (Bromley et al. 1993). Subsequent work showed that they can be successfully applied to a wide range of problems (Chicco 2020) such as human re-identification (Varior et al. 2016), text similarity learning (Neculoiu, Versteegh, and Rotaru 2016) and chromosome classification (Swati et al. 2017).

Triplet networks

Triplet networks are another instance of contrastive learning. It reiterates the same idea of producing multiple outputs and comparing them through a dedicated loss function. In this case, elements of a triplet of outputs computed from a triplet of inputs are compared. The input triplet consist of an anchor

6. An alternative view is to consider two neural networks sharing parameters. This is usually how siamese networks are depicted.

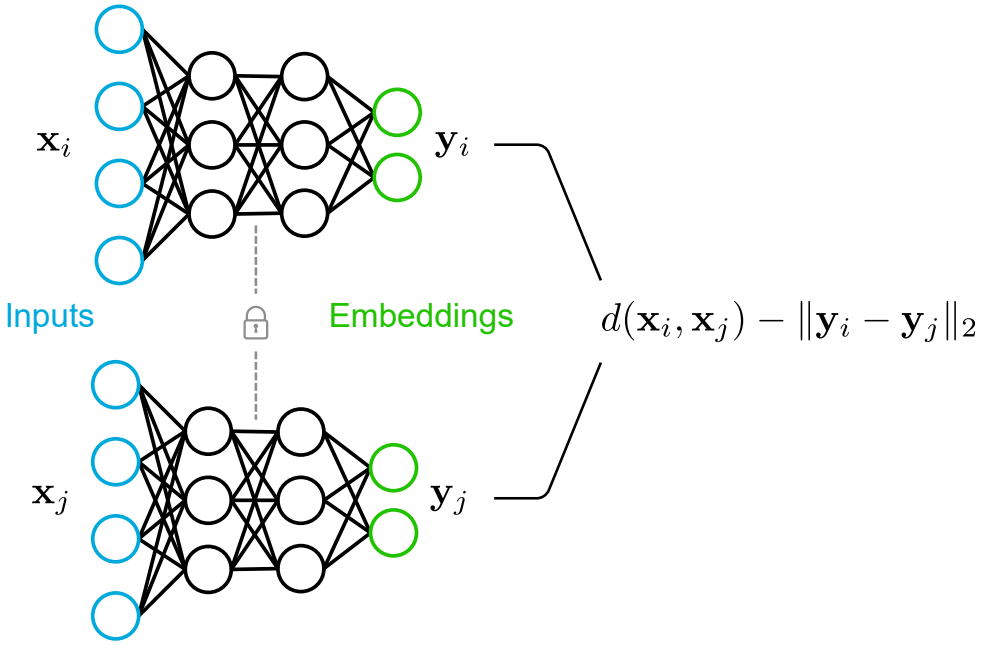


Figure 2.17 – Siamese network.

sample, a positive one and a negative one. The dissimilarities between the anchor and the close sample, and the anchor and the far sample are both computed. The network is then trained to minimize the first one while maximizing the other. Like in siamese networks, the model tries to learn representations where similar samples are close together. The difference here is that it is also simultaneously and explicitly trained to push away dissimilar samples. The loss function enforcing these constraints is called the triplet loss.

FaceNet (Schroff, Kalenichenko, and Philbin 2015) was the first proposed triplet networks model. It was used to learn face recognition of the same person at different poses and angles. Since then, many other works explored their use for a broad range of applications.

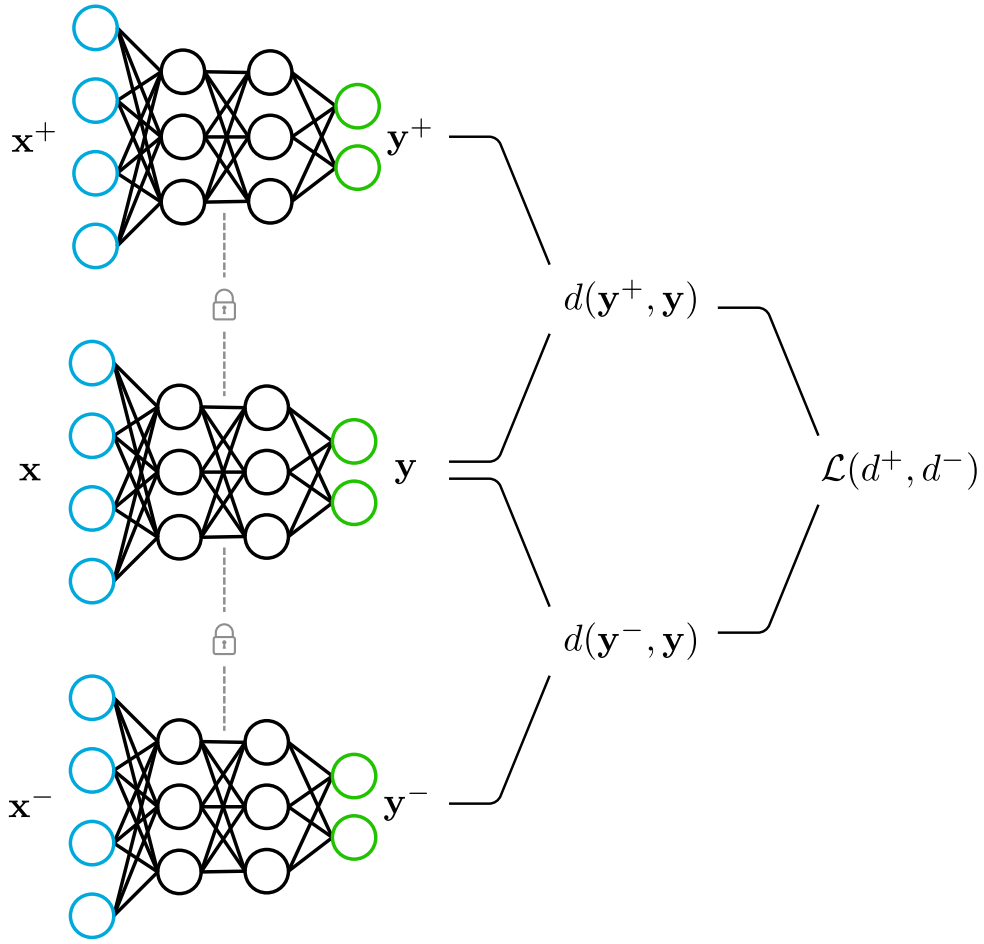


Figure 2.18 – Triplet network.

Highlight 2.6: Parametric versus nonparametric

The nonparametric approaches presented in this section all suffer from a common problem: they are unable to project new data on the fly. Indeed, to compute the projection of a new input, the optimization step needs to include all existing data as well. While out-of-sample extensions have been explored (Bengio et al. 2003), they are usually less performant than their parametric counterparts (Pai et al. 2019).

2.5 Machine learning-based channel estimation

ML holds promise for wireless communication (O'Shea and Hoydis 2017; T. Wang et al. 2017; Qin et al. 2019). More specifically, since the physics of propagation provides pretty accurate analytical models, model-driven ML approaches (He, Jin, et al. 2019) seem particularly suited.

In the context of massive MIMO channel estimation, the literature is full of promising ideas, but no prominent approach has yet stood out. For example, it has recently been proposed to use adaptive data representations using dictionary learning techniques (Ding and Bhaskar D Rao 2018). However, classical dictionary learning employing algorithms such as K-SVD (Aharon, Elad, and Bruckstein 2006), as proposed in (Ding and Bhaskar D Rao 2018), is very computationally heavy.

Deep unfolding has also been considered by communications researchers (Balatsoukas-Stimming and Studer 2019). It has mainly been used for symbol detection, unfolding projected gradient descent (Samuel, Diskin, and Wiesel 2017) or approximate message passing (AMP) (He, Wen, et al. 2018a) algorithms.

Regarding MIMO channel estimation, it has been proposed in (He, Wen, et al. 2018b) to unfold a sparse recovery algorithm named denoising-based AMP (DAMP) (C. A. Metzler, Maleki, and R. G. Baraniuk 2016). However, the method is directly adapted from image processing (C. Metzler, Mousavi, and R. Baraniuk 2017) and does not assume any prior. A recent work also proposes to use deep unfolding of the DAMP algorithm for MIMO channel estimation (Wei, Hu, and Dai 2019), using a physical model to optimize the shrinkage functions used by DAMP. However, these previously proposed methods all require collecting a database of clean channels and an offline training phase, due to their intrinsic supervised nature. This may hinder their practical applicability.

Finally, deep learning has also been applied to channel estimation in an orthogonal frequency-division multiplexing (OFDM) context, which is mathematically very close to the one studied here. In (Gao et al. 2018), a neural network is used as a post-treatment of the LS channel estimates in order to denoise them, and included in a joint channel estimation and detection framework. In (Soltani et al. 2019), the noisy time-frequency response of the

channel is viewed as an image and is denoised using classical denoising neural networks. Once again, these approaches are supervised and are of high complexity compared to classical methods (van Lier et al. 2020).

In (Balevi, Doshi, and Andrews 2020), a low complexity neural network based on the deep image prior network is used as a pre-treatment of the LS channel estimates and can be seen as a denoiser.

2.6 Conclusion

This chapter served as an introduction to ML concepts. While learning from data is its fundamental idea, it can be manifested in a variety of implementations and applications. We have seen how algorithms can be categorized, for example, according to the way they are trained, to the nature of the task they are trying to solve, or to the way their learned parameters are used. Classical algorithms like SVMs, k-NN, and PCA have been introduced, while a more focused overview of deep learning and its inner workings was given. Dimensionality reduction and manifold learning were also discussed as a way of compressing data and discovering its hidden structure. The chapter ends with an quick overview of how DL has been used in the literature for channel estimation.

The next chapter is dedicated to channel charting. While its general purpose is to learn to recover pseudo-locations from channel vectors, we will see how this problem can be framed as a dimensionality reduction one, and how it can be exploited to ease downstream tasks.

Chapter 3

Channel charting

User positioning has long been a challenging task. Although dedicated positioning systems like global navigation satellite systems (GNSS) are capable of accurately locating users, they require dedicated hardware, and integration in traditional communication systems is not always feasible nor desired (Studer et al. 2018). They are also known for their bad performance in indoor scenarios, making them not as versatile as needed. Pure radio-based approaches where established communications systems are used for user positioning are thus more attractive. Such approaches are often model-based and usually rely on angle-of-arrival (AoA), time-difference-of-arrival (TDoA) and round-trip-time (RTT) measurements (Dwivedi et al. 2021). These rather basic models work well for simple scenarios, e.g., line-of-sight (LoS) propagation, but struggle when tested on complex environments, e.g., non-line-of-sight (NLoS) propagation. They constitute a great step towards accurate and precise positioning but still fall behind in terms of the performance requirements of the next generations of wireless systems.

The widespread adoption of massive MIMO systems and large bandwidths means higher resolvability in both the temporal and angular domains. The channel, being dependent almost exclusively on the transmit position, now contains enough information to retrieve it. Nonetheless, hand-crafting algorithms capable of this becomes unrealistic as propagation scenarios become more complex. On the other hand, data-driven methods were successfully exploited to learn to locate UEs with great precision (X. Wang et al. 2015; Vieira et al. 2017; Decurninge et al. 2018; Ferrand, Decurninge, and Guillaud 2020). However, these methods come with their own drawbacks as they usually require a database of labelled data often hard to acquire. This becomes even more challenging as a database needs to be provided for every new en-

vironment to train the model. This being said, although many components of a communication system may benefit from the knowledge of UEs' locations, most of them would only require a *relative* positioning of the users.

Recently, *Channel charting* (CC) (Studer et al. 2018; Ferrand, Decurninge, Ordoñez, et al. 2021; B. Rappaport et al. 2021) has emerged as an unsupervised alternative to user positioning, relying solely on channel data to pseudo-locate UEs relatively to each other. It consists in learning a mapping between channel measurements and a low dimensional space, giving rise to a chart. A mathematical framework to formally introduce the problem is given in the first part of this chapter. Points on the chart are akin to pseudo-locations in that they preserve the structure of small spatial neighborhoods rather than the absolute locations. To measure their quality, performance metrics derived from the field of dimensionality reduction can be used as we will see. To learn to produce a chart, different methods have been proposed, ranging from PCA, the simplest form of dimensionality reduction, to more involved deep learning-based ones. While most of them were discussed in the last chapter, we will present how they can be specialized for CC. Although less informative than absolute locations, pseudo-locations often convey enough information for some location-based applications such as SNR prediction (Kazemi, Al-Tous, et al. 2020), beam management (Ponnada et al. 2021) and pilot allocation (Ribeiro, Leinonen, et al. 2020) among others. Some of these applications will be discussed at the end of this chapter.

3.1 Manifold hypothesis for channel charting

In a static propagation environment, the channel is mainly determined by the position and the nature of the scatterers, as well as the TX and RX locations. A perfect knowledge of these parameters would allow for a perfect reconstruction of the channel. It would also allow for the retrieval of the TX location of any given channel [1]. More specifically, a BS would be able to know the exact position of UEs communicating with it with the only knowledge of their channels. On the other hand, the set of "observable" channel measure-

1. Although two identical channel measurements can in fact correspond to different TX locations, it is considered a sufficiently rare case to be reasonably ignored.

ments (i.e., measurements that can indeed occur) is a subset of the whole channel space. More specifically, it is a low-dimensional subspace embedded in the higher dimensional one. The manifold hypothesis introduced in section 2.4 applies for propagation channels. In this case, it states that these high-dimensional vectors can in fact be summarized by a smaller set of parameters. In a static propagation environment where everything is fixed, the location of an UE is supposedly the only parameter needed for determining the channel. Varying this parameter[2] would allow to navigate on the manifold of all possible channel measurements.

While in reality such a knowledge of the propagation environment is not attainable, channel models (section 1.1) postulate a propagation model and try to estimation its parameters. This can be seen as approaching the real channel manifold through a parametric model whose parameters control its shape. The accuracy of the approximation depends on both the complexity of the assumed model and that of the propagation environment. With the increasing trend of massive MIMO systems, channel vector dimensions continue to grow, posing a challenge for classical models that rapidly reach their limits. In contrast, CC aims at jointly learning the channel model as well as its parameters by directly learning the channel manifold through dimensionality reduction techniques (section 2.4). These techniques make use of the considerable amount of available data to learn their intricate underlying patterns. Moreover, the unsupervised aspect of CC is also very important as it shows that this knowledge can be retrieved without any side information.

3.2 Problem formulation

Consider N UEs placed on a map around a BS with associated locations $\{\mathbf{p}_i\}_{i=1}^N$ and channels $\{\mathbf{h}_i\}_{i=1}^N$ (in vector form). Alternatively, a single UE can be considered where the different locations and channels correspond to its progress through time. Let us denote $M = N_T N_R N_s$ the considered channel dimension. CC aims at learning a function $\mathcal{F}$, usually called the *charting function*, that maps the channel vectors of UEs in an area of interest A to a

2. This parameter is in reality 2 or 3-dimensional

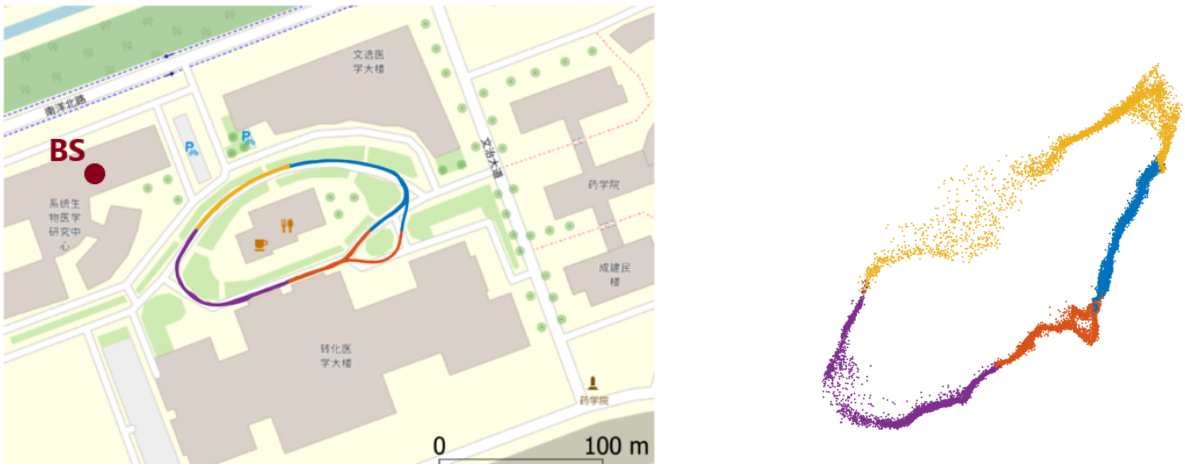


Figure 3.1 – The chart (right) is obtained from the channel measurements of UEs distributed as shown on the map (left). From (Ferrand, Decurninge, Ordoñez, et al. 2021).

low-dimensional space of dimension $D \ll M$,

$$\begin{aligned} \mathcal{F} : \mathbb{C}^M &\to \mathbb{R}^D \\ \mathbf{h} &\mapsto \mathcal{F}(\mathbf{h}) = \mathbf{z}, \end{aligned}$$

where $\mathbf{z}$ is the chart location (or *pseudo-location*) associated to $\mathbf{h}$. CC methods are assessed on their ability to preserve spatial neighborhoods, that is, how well the original spatial distances match the distances on the learned chart for close points

$$\|\mathbf{p}_i - \mathbf{p}_j\|_2 \approx \|\mathbf{z}_i - \mathbf{z}_j\|_2, \tag{3.1}$$

The ability to preserve the global geometry can also be considered. Figure 3.1 shows an example of a chart obtained from channel measurements of UEs distributed as shown on the map. Note that the use of colors is standard in CC as it allows for a better understanding of the structure and shape of the chart and how it relates to the original locations.

3.3 Performance measures

Evaluating the performance of a channel chart by comparing it to the true geographical locations based on visual inspection has its limits. Although this

surely helps get a sense of the global structure of the learned chart and gives insight into the way it was constructed, drawing conclusions is highly subjective and not reliable as a consequence. Instead, it is better to rely on commonly used performance metrics that objectively measure different desirable properties a chart has to conform to.

3.3.1 Trustworthiness

Trustworthiness (TW) measures how much channels yielding nearby points on the learned charts have TX locations that are also nearby in the original geographical space. The obtained score is between 0 (worst) and 1 (best). Formally, for a given neighborhood of size K, TW is expressed as

$$\text{TW}(K) = 1 - \frac{2}{NK(2N-3K-1)} \sum_{i=1}^{N} \sum_{j \in \mathcal{U}_K^i} \max(0, (r(i,j) - K)), \tag{3.2}$$

where $\mathcal{U}_K^i$ is the set of K closest neighbors to the ith point in the learned chart, and $r(i,j)$ is the rank of j in the neighborhood of i in the original geographical space in terms of the Euclidean distance separating them.

3.3.2 Continuity

Continuity (CT) is very similar to trustworthiness. It measures how much channels associated with nearby TX locations yield points on the chart that are also nearby. The obtained score is also between 0 (worst) and 1 (best). Formally, for a given neighborhood of size K, CT is expressed as

$$\text{CT}(K) = 1 - \frac{2}{NK(2N-3K-1)} \sum_{i=1}^{N} \sum_{j \in \mathcal{V}_K^i} \max(0, (\hat{r}(i,j) - K)), \tag{3.3}$$

where $\mathcal{V}_K^i$ is the set of K closest neighbors to the ith point in the original geographical space, and $\hat{r}(i,j)$ is the rank of j in the neighborhood of i in the learned chart in terms of the Euclidean distance separating them.

3.3.3 Kruskal's stress

Kruskal's stress (KS) measures how much distances between points in the original space match distances on the learned chart. It is closely related to the stress functions used in MDS and SM (section 2.4.1) and can be expressed as

$$\mathrm{KS} = \sqrt{\frac{\sum_{i,j} \left(\|\mathbf{p}_i - \mathbf{p}_j\|_2 - \beta \|\mathbf{z}_i - \mathbf{z}_j\|_2 \right)^2}{\sum_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|_2^2}}, \tag{3.4}$$

where β is defined as

$$\beta = \sqrt{\frac{\sum_{i,j} \|\mathbf{p}_i - \mathbf{p}_j\|_2 \|\mathbf{z}_i - \mathbf{z}_j\|_2}{\sum_{i,j} \|\mathbf{z}_i - \mathbf{z}_j\|_2^2}}. \tag{3.5}$$

The lower the KS the better.

3.4 Channel charting approaches

Although CC is a relatively novel area of study, many approaches have already been proposed in the literature. Since it is based on dimensionality reduction, it makes sense to also categorize CC approaches into parametric and nonparametric. The latter produce a chart at a one-shot based on available data, while the former learns a mapping that allows for *on-the-fly* computation of a pseudo-location.

3.4.1 General approaches

The seminal paper (Studer et al. 2018) introduced the problem of CC as one of dimensionality reduction. Their initial observation was that raw channel observations are not very suitable for the task and need some preprocessing, mainly because of problems related to scaling and to the global phase. The first step was to consider a set of channel features that would ease the task. The authors considered features that preserve the large-scale fading properties derived from the empirical raw second moment of channel vectors, i.e., $\mathbf{R} = \frac{1}{T} \sum_{t=1}^{T} \mathbf{h}_t \mathbf{h}_t^{\mathsf{H}}$ where the sum is taken over T observations collected over a timeframe well below the coherence time. The outer product effectively can-

cels out the global phase-shift and thus mitigates the effect of small-scale fading. Based on these features, three dimensionality reduction methods were tested and compared against each other, namely, PCA (section 2.4.1), SM (section 2.4.1) and a simple autoencoder (section 2.4.2). PCA, being a linear method, performs the worst as expected. Sammon's mapping and the autoencoder give relatively good results.

Following work further explored the use of neural networks for CC. In (Lei et al. 2019), the authors proposed to use siamese networks (section 2.4.2) to learn the chart. Their idea was inspired by SM and can be seen as it model-based deep learning counterpart. Indeed, the loss function to minimize is exactly that of SM, only this time the parameter space is that of the weights of the siamese network instead of the chart locations directly. The obtained model is thus a parametric version of SM, allowing for data-driven optimization.

Inspired by this work, authors of (Ferrand, Decurninge, Ordoñez, et al. 2021) proposed to use yet another contrastive learning approach, namely triplet networks (section 2.4.2). Their strategy for harvesting positive (closely) and negative (far) samples was to rely on a timestamp-based approach. Indeed, each channel observation is accompanied by a timestamp corresponding to the exact moment at which it was collected by the BS. It is then legitimate to consider two observations close in time to correspond, with high probability, to TX locations spatially close to each other. An inner temporal threshold T_c can then be set to define an interval around the anchor sample's timestamp; close samples lie inside the interval and far samples outside of it. In addition, an outer temporal threshold T_f can be defined to set the limits of the interval inside of which far samples lie.

On another note, CC can be extended to a semi-supervised setting where side information is incorporated in the training, for example in the form of exact knowledge for a small set of training points, to turn the learned chart into a map of real approximate locations (P. Huang et al. 2019; Lei et al. 2019).

Many of the CC approaches can also be extended to the multipoint case, that is, in scenarios where many BSs collaborate to produce the chart. The procedure usually involves a chart fusion. The first work to exploit this idea was (Deng et al. 2018). Their method was to compute a dissimilarity matrix

at each BS, and then combine the matrices into a global one by computing a sum weighted by a reliability factor derived from the estimated SNRs of each corresponding entry. Subsequent work explored different charting and fusion strategies like using NNs or relying on visibility conditions (Q. Zhang and Saad 2021; Pihlajasalo et al. 2020; Al-Tous, Ponnada, et al. 2020).

3.4.2 Distance-based approaches

Since CC aims at preserving the relative locations of the UEs (locally first, then globally if possible), many of the recent work focuses on improving the considered channel distance measure.

In (Agostini, Utkovski, and Stańczak 2020), an Euclidean distance matrix completion (EDMC) approach is presented. It is based on the raw second moment features. The covariance distance measure (CMD) given as

$$d_{\text{CMD}}(\mathbf{R}_i, \mathbf{R}_j) = 1 - \frac{\text{tr}(\mathbf{R}_i^{\text{H}}\mathbf{R}_j)}{\|\mathbf{R}_i\|_2\|\mathbf{R}_j\|_2} \tag{3.6}$$

is used to identify close clusters of UEs. This is because this distance is only approximately accurate for positions that are almost colinear. A neighborhood graph is constructed from the computation of this distance and only connections under a certain threshold are kept. The edges are however populated with the original Euclidean distance in the feature space, i.e., $\|\tilde{\mathbf{R}}_i - \tilde{\mathbf{R}}_j\|_2$ where $\tilde{\mathbf{R}}$ is a scaled version of $\mathbf{R}$. Finally, the chart obtained by solving a graph realization problem through an EDMC algorithm.

The author of (Le Magoarou 2021a) proposed a distance that, contrary to other methods, doesn't rely on the sometimes expensive[3] estimation of the second moments. The distance is instead evaluated on the raw channel vectors. It stems from the observation that the Euclidean distance on the space of channel vectors is not suitable. The proposed phase-insensitive channel distance eliminates the ill-effect of the global phase. It is the solution to the

3. Expressive not exclusively from the point of view of computational complexity but also when considering the size (or dimension) of the resulting features.

minimization problem

$$d_{\mathrm{PI}}(\mathbf{h}_i, \mathbf{h}_j) = \min_{\phi \in [0, 2\pi]} \left\| \frac{\mathbf{h}_i}{\|\mathbf{h}_i\|_2} - e^{j\phi} \frac{\mathbf{h}_i}{\|\mathbf{h}_i\|_2} \right\|_2 . \tag{3.7}$$

The closed-form solution is given as

$$d_{\mathrm{PI}}(\mathbf{h}_i, \mathbf{h}_j) = \sqrt{2 - 2\frac{|\mathbf{h}_i^{\mathsf{H}} \mathbf{h}_j|}{\|\mathbf{h}_i\|_2 \|\mathbf{h}_j\|_2}}. \tag{3.8}$$

Using this distance, a chart can be obtained through a manifold learning algorithm like Isomap (section 2.4.1). This distance proved useful for user positioning (Le Magoarou 2021b) as well. It is however important to mention that it needs phase synchronization to be carried out.

Highlight 3.1: Channel distance challenges

Usual geometric distances are not very suitable for channel vectors. For example, the Euclidean distance between two channel measurements corresponding to distant locations (w.r.t. the BS) will be smaller than the distance between two channel measurements corresponding to nearby locations, even if the real geometric distances are the same. This is because of large-scale fading that is reflected in the norm of the channel vectors.

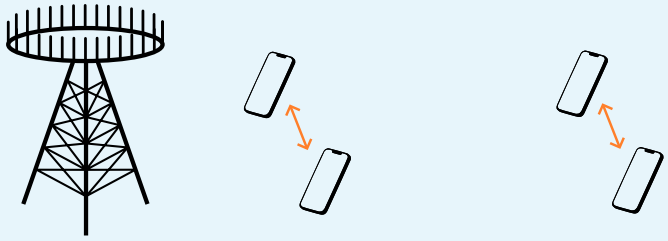


Figure 3.2 – Although the UEs are separated by the same distance, their Euclidean channel distances depend on their proximity to the BS.

Normalizing the vectors is not enough neither because of the effect of the global phase. Indeed, two channel measurements corresponding to locations aligned with the BS and separated by half a wavelength will appear maximally distant.

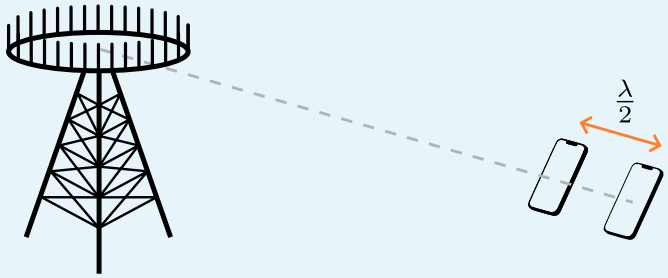


Figure 3.3 – UEs aligned with the BS and separated by $\frac{\lambda}{2}$ appear maximally distant under the normalized Euclidean channel distance.

The idea of first deriving a distance measure that is locally approximately proportional to the real Euclidean distance and then generalizing to a global

distance through an estimation of the geodesics has been reused and adapted in many of the subsequent work. For example, the authors of (Stahlke et al. 2023) proposed to use a local distance that is based on an approximation of the time of flight difference between the channel measurements corresponding to two locations that share the same MPCs and measured at one BS or more. It takes as input the channel vectors in the time domain and is given as

$$d_{\text{CIR}}(\mathbf{h}_i, \mathbf{h}_j) = \sum_{b=1}^{B} \sum_{l=1}^{L} \sum_{t=1}^{T} ||\mathbf{h}_i[t] - \mathbf{h}_j[t]||_2 \,, \tag{3.9}$$

where B is the number of considered BSs, L the number of MPCs and T the number of time instants. The precision of the approximation is proportional to B. Similarly to (Le Magoarou 2021a), a chart is constructed by applying Isomap on an initial set of measurements to learn the global distance. Finally, a model-based extension of Isomap is proposed in the same fashion as (Lei et al. 2019) where the network learns to approximate the geodesic distance learned by the manifold learning algorithm.

(Stephan, Euchner, and Brink 2023) introduces two novel distance measures. The first one, called the angle-delay profile (ADP) distance, is inspired by both d_{PI} and d_{CIR}. It is given as

$$d_{\text{ADP}} = \sum_{b=1}^{B} \sum_{t=t_{\min}}^{t_{\min}} \left(1 - \frac{|\mathbf{h}_i^{\mathsf{H}} \mathbf{h}_j|^2}{\|\mathbf{h}_i\|_2 \|\mathbf{h}_j\|_2^2}\right), \tag{3.10}$$

where $t_{\min}$ and $t_{\max}$ are chosen to exclude weak parts of the received signal. It exploits the fact that channel vectors are sparse in the time domain and that most of the power is concentrated around the LoS component. Second, the authors suggest to directly train a NN to learn the distance measure. Such a model would take as input two channel features derived from channel measurements $\mathbf{h}_i$ and $\mathbf{h}_j$ (autocorrelation matrices in this case), and predict the distance between them. A distance matrix produced this way can then be further exploited to learn a parametric model capable of projecting a single input on the chart. The network is trained to minimize the error between the predicted distance and a temporal distance based on the timestamp of each measurement.

The article also shows that many of the ideas of preceding work can be

combined together to achieve better performance. For example, it was shown that combining two or more distances computed independently in the form of a weighted sum leads to a chart of better quality, especially when the distances rely on different sources of knowledge. Indeed, such distances may individually fail to perform well on all scenarios, but compensate for the weaknesses of each other once combined together (e.g., channel measurements and timestamp-based distances).

A final example of channel distance measures is the one presented in (Al-Tous, Kazemi, et al. 2022). The approach therein is resembles the one in (Deng et al. 2018) and considers extracting the angle-delay-power profile $\{(\alpha_1, \tau_1, p_1), \ldots, (\alpha_{L_i}, \tau_{L_i}, p_{L_i})\}_i$ for each channel vector to construct corresponding features using superresolution algorithms. To tackle the challenge that different numbers and ordering of MPCs presents, a solution is borrowed from the world of optimal transport. Indeed, it is proposed to use an earth-mover distance (EDM, or Wasserstein distance) on the superresolution features, the details of which can be found in the original paper.

3.5 Applications

Many radio resource management (RRM) applications only require a locally coherent chart where pseudo-locations are consistent with local neighborhoods. Others are sensitive to the global geometry as well. In all cases, CC can be exploited in many downstream tasks to either improve existing techniques or enable novel use cases (Ferrand, Guillaud, et al. 2023). Some examples are listed below.

3.5.1 Localization

UE localization is one of the most straightforward applications of CC. It can be achieved with relative ease, provided that a partially annotated dataset of locations is available. Points for which both the channel measurements and the associated locations are available are referred to as *anchors*. A channel chart can be viewed as a distorted version of the map of real locations. The objective then becomes to learn the transformation, implicitly or explicitly, to

apply to the charting function so that it maps channel vectors to real locations. In this context, many approaches have been proposed.

Deep learning-based CC approaches generally proceed by adding penalty terms to the loss function (P. Huang et al. 2019; Lei et al. 2019), or by dedicating a separate training phase to the anchors (Q. Zhang and Saad 2021). This way, constraints on the learned charting function are enforced so that the pseudo-locations of the anchors match their real locations. In doing so, the hope is that non-anchor points benefit from these constraints as well, and for their locations and pseudo-locations to be close.

A simpler approach is to assume that a simple affine transformation is sufficient to go from the chart to the real map, as proposed by (Stahlke et al. 2023; Stephan, Euchner, and Brink 2023). The transformation's coefficients are found by solving a simple LS problem involving the anchors.

In the multipoint case, an alternative is to exploit the knowledge of BSs' locations if available instead of relying on the partial knowledge of UEs locations as proposed in some work (Taner, Palhares, and Studer 2023; Pihlajasalo et al. 2020).

3.5.2 Pilot allocation

As mentioned in section 1.2, pilot symbols are an essential part of CE. Unfortunately, many estimation algorithms rely on the orthogonality of the set of pilots assigned to the UEs. This condition becomes harder to fulfill in dense communication scenarios like the ones envisioned in 5G systems (e.g., communicating with a large number of IoT devices). Allocating a unique pilot per UE becomes challenging. A different strategy consists of allocating orthogonal pilots to UEs close together, and reusing pilots for UEs far away from each others. However, this strategy is prone to pilot contamination, which is the interference caused by colliding pilot sequences. To mitigate this problem, (Ribeiro, Leinonen, et al. 2020) proposed to exploit a channel chart to allocate pilot symbols based on the proximity of pseudo-locations which reflect UEs of close DoAs. A greedy algorithm attributes pilot symbols to points on the chart in a cyclic way, minimizing interferences. An extension to the multipoint case was later presented in (Ribeiro, Leinonealn, et al. 2022) where chart fusion

was accomplished in a similar manner to (Deng et al. 2018).

3.5.3 Beam management

Beam management is another task for which CC has proven useful. The angular dependency of beams makes it a perfect candidate.

For example, a beam prediction algorithm based on CC was proposed in (Ponnada et al. 2021). The objective was to determine the optimal beam pairs between a BS and UEs, all equipped with antenna arrays and thus capable of beamforming. Instead of fully relying on conventional beam sweeping, usually time-consuming, for all, the authors proposed to only annotate an initial training set with the best beams, and learn to predict beams for subsequent data. This was achieved by associating the annotation to pseudo-locations on a learned chart. NNs, SVMs and k-NN were considered and compared for learning the final mapping from the chart to the best beam. The NN was capable of achieving the same performance for both true locations and pseudo-locations, proving that CC can indeed replace the need for complementary position systems.

CC was also considered for beam SNR prediction (Kazemi, Ponnada, et al. 2021) and beam tracking (Kazemi, Al-Tous, et al. 2022).

Learning an optimal charting function is still a relatively young research problem and many applications are yet to be discovered. Beyond the ones listed above, examples of other instances where CC could prove useful include: *CSI compression*, *channel modelling*, *device grouping*, *handover prediction*, etc.

3.6 Conclusion

This chapter was dedicated to CC. While it is closely related to user localization, it differs in that the model s expected to learn to produce a spatially coherent chart on its own, without relying on labelled data. In doing so, the hope is that it learns an internal model of the channel in the form of a mapping to its underlying manifold—a manifold of the same dimension or close to that of the UEs' locations. To do so, CC methods rely on dimensionality reduction and manifold learning algorithms like Isomap and contrastive NNs. We've

seen that CC comes with a flexibility that allows for a diverse range of implementations to exist. For example, it is possible to use multiple BSs at once to produce a chart. It is also possible to produce more accurate charts at the cost of some partial labeling by training in a semi-supervised way. Distance-based CC methods are of particular interest as we showed how using an adequate distance measure is a close proxy problem. Relying on a distance measure can also be considered as a way of incorporating domain knowledge into CC methods, which is in line with the concept of model-based deep learning (section 2.3.4) when working with NNs. Finally, we've seen that CC is merely a preprocessing step to ease the way for downstream location-dependent tasks like pilot allocation and beam management.

The next part of this manuscript goes over the different contributions of this thesis. We will see how the model-based principals introduced in this part can be used to solve and improve existing techniques when it comes to channel estimation and channel charting.

PART II

Contributions

Chapter 4

Model-based deep learning for channel estimation

Channel estimation is the first main topic of this thesis. We have seen that modern MIMO communication systems lead to channels of very high dimensions comprising hundreds or even thousands of complex numbers, making channel estimation difficult. But despite their high dimension, their sparsity makes it possible to approximate them by only a few dominant propagation paths. While physical channel models allow injecting strong a priori knowledge based on solid principles, they necessarily make simplifying assumptions and require knowing exactly the system configuration (positions of the antennas, gains, etc.). Such requirements being unrealistic in practice, the massive MIMO channel estimation task could perfectly benefit from the flexibility offered by model-based deep learning approaches in general, and by deep unfolding in particular.

This chapter introduces mpNet, an unfolded neural network specifically designed for massive MIMO channel estimation. The unfolded algorithm is MP (Mallat and Z. Zhang 1993), taking advantage of the massive MIMO channel sparsity. The objective stays the same (section 1.2.3), with the added flexibility offered by deep learning. The weights of mpNet are initialized using an imperfect physical model. The main concerns while designing the method were to make it:

— *Unsupervised*: no need for clean channels to train mpNet, the method uses the LS channel estimates as training data and a reconstruction loss function. It is thus trained as an autoencoder (Rumelhart, G. E. Hinton, and Williams 1986). In addition, and unlike the LMMSE estimator (section 1.2.2), mpNet does not require knowledge of the channel correlation

matrix.

— *Online*: the traditional offline training phase is replaced by an intelligent initialization using an imperfect physical model. mpNet is then trained incrementally using online gradient descent (Bottou 1998).

— *SNR adaptive*: no need for several networks trained at different signal-to-noise ratio (SNR) levels, mpNet automatically adapts its depth to incoming data.

— *Computationally efficient*: backpropagation through mpNet is cheaper than classical channel estimation using a greedy sparse recovery algorithm (which corresponds to the forward pass of mpNet).

Such a method is particularly suited to imperfectly known or non-calibrated systems. Indeed, starting from an imperfect physical channel model, our method allows a BS to automatically correct its channel estimation algorithm based on incoming data. While most ML-based channel estimation methods require a training dataset and are often computationally intensive, the method we propose can be seen as an instance of the model-driven approach and is designed to be computationally efficient.

This chapter starts by introducing the considered system configuration and formally describes the estimation problem and its relationship with the LS estimator. The physical model the method is based on is also formally defined. To demonstrate the negative impact of an imperfect knowledge of the physical parameters, a small experiment is conducted. mpNet is then introduced. Its structure and training are detailed, and its computational complexity is studied. Finally, the method is evaluated in a number of configurations involving fixed and varying SNR, anomaly detection and recovery, and the use of different antenna structures.

4.1 Problem formulation

4.1.1 System settings

We consider in this work a massive MIMO system, also known as multi-user MIMO (MU-MIMO) system (Rusek et al. 2013; Larsson et al. 2014; Lu

et al. 2014), in which a BS equipped with N_R antennas communicates with N single antenna users ($N < N_R$). For simplicity, transmission on a single subcarrier is considered (i.e., $N_s = 1$), even though everything presented here can obviously be generalized to the multicarrier case. The system operates in time division duplex (TDD) mode, so that channel reciprocity holds and the channel is estimated in the uplink: each user sends a pilot sequence $\mathbf{p}_n \in \mathbb{C}^T$ of duration T (orthogonal to the sequences of the other users, $\mathbf{p}_n^{\mathsf{H}}\mathbf{p}_m = \delta_{nm}$ where δ_{nm} is the Kronecker delta) for the BS to estimate the channel. For simplicity, the transmit power is set to 1 throughout this chapter, and the signal to noise ratio (SNR) is determined only by the channel and noise powers. The received signal is thus expressed

$$\mathbf{Y} = \sum_{n=1}^{N} \mathbf{h}_n \mathbf{p}_n^{\mathsf{H}} + \mathbf{N}, \tag{4.1}$$

where $\mathbf{Y} \in \mathbb{C}^{N_R \times T}$, $\mathbf{h}_m$ is the channel corresponding to the mth user and $\mathbf{N} \in \mathbb{C}^{N_R \times T}$ is Gaussian noise. After correlating the received signal with the pilot sequences, and assuming no pilot contamination from adjacent cells for simplicity, the BS gets noisy measurements of the channels of all users, taking the form

$$\mathbf{x}_m \triangleq \mathbf{Y}\mathbf{p}_m = \underbrace{\sum_{n=1}^{N} \mathbf{h}_n \mathbf{p}_n^{\mathsf{H}} \mathbf{p}_m}_{\mathbf{h}_m} + \underbrace{\mathbf{N}\mathbf{p}_m}_{\mathbf{n}_m}, \tag{4.2}$$

for $m = 1, \dots, N$, with $\mathbf{n}_m \sim \mathcal{CN}(0, \sigma^2 \mathbf{Id})$, $\forall n$. In order to simplify notations in the remaining parts of the chapter, we drop the user index and denote such measurements with the canonical expression

$$\mathbf{x} = \mathbf{h} + \mathbf{n}, \tag{4.3}$$

where $\mathbf{h}$ is the channel of the considered user and $\mathbf{n}$ is noise, with $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2 \mathbf{Id})$. Such a dropping of the user index does not harm the description of our approach, since it treats the channels of all users indifferently. Note that $\mathbf{x}$ is already an unbiased estimator of the channel, obtained by solving a LS estimation problem, so that we call it the LS estimator in the sequel.

Its performance can be assessed by the input SNR

$$\text{SNR}_{\text{in}} \triangleq \frac{\|\mathbf{h}\|_2^2}{N_R \sigma^2}.$$

However, one can get better channel estimates using a physical model which allows to denoise the LS estimate, as is explained in the next section.

Highlight 4.1: LS estimate

The formula of the LS estimate derived in section 1.2.1 and adapted for this scenario becomes

$$\mathbf{h}_{\text{LS}} = \mathbf{Y}\mathbf{p}^*(\mathbf{p}^{\text{H}}\mathbf{p})^{-1}. \tag{4.4}$$

Since $\mathbf{p}^{\text{H}}\mathbf{p} = 1$, then the expression matches that of (4.2), and $\mathbf{x} = \mathbf{h}_{\text{LS}}$.

4.1.2 Physical model

Let us denote $\{g_1, \dots, g_{N_R}\}$ the complex gains of the BS's antennas, $\{\vec{a_1}, \dots, \vec{a_{N_R}}\}$ their positions with respect to the centroid of the antenna array and λ the considered wavelength. According to (1.9), the channel resulting from a single propagation path with DoA $\vec{u}$ is $\mathbf{h} = \beta \mathbf{a}(\vec{u})$, with $\beta = \sqrt{N_R}\alpha e^{-j\frac{2\pi}{\lambda}r}$. In this case, a sensible estimation strategy (Sayeed 2002; Bajwa et al. 2010; Le Magoarou and Paquelet 2018) is the one presented in section 1.2.3 and recalled in the following. It consists of building a dictionary of steering vectors corresponding to N_D potential DoAs and to compute a channel estimate with the procedure

$$\vec{v} = \text{argmax}_{\vec{u_i}} \ |\mathbf{a}(\vec{u_i})^{\text{H}}\mathbf{x}|, \tag{4.5a}$$

$$\hat{\mathbf{h}} = \mathbf{a}(\vec{v})\mathbf{a}(\vec{v})^{\text{H}}\mathbf{x}. \tag{4.5b}$$

The first step, (4.5a), of this procedure amounts to find the atom of the dictionary the most correlated with the observation $\mathbf{x}$ (LS channel estimate) to estimate the DoA $\vec{v}$, while the second step, (4.5b), amounts to project the observation on the corresponding steering vector $\mathbf{a}(\vec{v})$. The SNR at the output of

this procedure reads

$$\mathrm{SNR}_{\mathrm{out}} \triangleq \frac{\|\mathbf{h}\|_2^2}{\mathbb{E}\left[\|\mathbf{h}-\hat{\mathbf{h}}\|_2^2\right]},$$

and we have at best $\mathrm{SNR}_{\mathrm{out}} = N_R \mathrm{SNR}_{\mathrm{in}}$ (neglecting the discretization error), if the selected steering vector is collinear to the actual channel. This is a direct consequence of the Cauchy-Schwarz inequality, and is intuitively explained by the fact that from the N_R complex dimensions of $\mathbf{x}$ corrupted by noise, only one is kept when projecting on the best steering vector, so that the effective noise variance is divided by N_R. The potential gain of using such a physical model can be huge, especially for massive MIMO systems in which the number of antennas N_R is large.

Moreover, this strategy can be generalized to estimate sparse multipath channels of the form

$$\mathbf{h} = \sum_{l=1}^{L} \beta_l \mathbf{a}(\vec{u_l}), \tag{4.6}$$

where L is the number of paths, by iterating the procedure (4.5) until some predefined stopping criterion is met.

4.2 Motivation: imperfect models

Basing an estimation strategy on a physical model, as suggested in the previous section, requires knowing precisely the physical parameters of the system (in particular the positions and gains of the antennas) in order to build an appropriate dictionary. Then, even in the case of perfect system knowledge, some simplifying hypotheses (such as the plane wave assumption considered in the previous section) have to be made in order to keep the model mathematically tractable. Consequently, every model, regardless of its sophistication, is necessarily imperfect. Such a situation is well-known and summarized by the aphorism *"All models are wrong"* (Box 1976).

In the context of MIMO channel estimation, what is the impact of an imperfect knowledge of the physical parameters and/or of the invalidity of some hypotheses? In order to address this question, a simple experiment is conducted. Consider an antenna array of $N_R = 64$ antennas at the BS, whose known *nominal* configuration is a ULA of unit gain antennas separated by

half-wavelengths and aligned with the x-axis. This nominal configuration corresponds to gains and positions $\{\tilde{g}_i, \tilde{\vec{a}}_i\}_{i=1}^{N_R}$. Now, suppose the knowledge of the system configuration is imperfect, meaning that the unknown *true* configuration of the system is given by the gains and positions $\{g_i, \vec{a}_i\}_{i=1}^{N_R}$, with

$$\begin{aligned} g_i &= \tilde{g}_i + n_{g,i},\ n_{g,i} \sim \mathcal{CN}(0, \sigma_g^2), \\ \vec{a}_i &= \tilde{\vec{a}}_i + \lambda \mathbf{n}_{p,i},\ \mathbf{n}_{p,i} = \begin{pmatrix} e_{p,i}, & 0, & 0 \end{pmatrix}^T, e_{p,i} \sim \mathcal{N}(0, \sigma_p^2). \end{aligned} \tag{4.7}$$

This way, σ_g (resp. σ_p) quantifies the uncertainty about the antenna gains (resp. spacings). Moreover, let

$$\tilde{\mathbf{a}}(\vec{u}) \triangleq \frac{1}{\sqrt{\sum_{i=1}^{N_R} |\tilde{g}_i|^2}} (\tilde{g}_1 \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\tilde{\vec{a}}_1 . \vec{u}}, \ldots, \tilde{g}_{N_R} \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\tilde{\vec{a}}_{N_R} . \vec{u}})^T \tag{4.8}$$

be the nominal steering vector and $\tilde{\mathbf{D}} \triangleq \left(\tilde{\mathbf{a}}(\vec{u_1}), \ldots, \tilde{\mathbf{a}}(\vec{u_{N_D}})\right)$ be a dictionary of nominal steering vectors. The experiment consists in comparing the estimation strategy of (4.5) using the true (perfect but unknown) dictionary $\mathbf{D}$ with the exact same strategy using the nominal (imperfect but known) dictionary $\tilde{\mathbf{D}}$. To do so, we generate measurements according to (4.2) with channels of the form $\mathbf{h} = \mathbf{a}(\vec{u})$ where $\vec{u}$ corresponds to azimuth angles chosen uniformly at random, and $\mathrm{SNR}_{\mathrm{in}}$ is set to $10\,\mathrm{dB}$. Then, the dictionaries $\mathbf{D}$ and $\tilde{\mathbf{D}}$ are built by choosing $N_D = 32N_R$ directions corresponding to evenly spaced azimuth angles. Let $\hat{\mathbf{h}}_{\mathbf{D}}$ be the estimate obtained using $\mathbf{D}$ in (4.5), and $\hat{\mathbf{h}}_{\tilde{\mathbf{D}}}$ the estimate obtained using $\tilde{\mathbf{D}}$. The SNR loss (performance decrease) caused by using the imperfect but known dictionary $\tilde{\mathbf{D}}$ instead of the perfect but unknown dictionary $\mathbf{D}$ is measured by the quantity

$$\frac{\|\hat{\mathbf{h}}_{\tilde{\mathbf{D}}} - \mathbf{h}\|_2^2}{\|\hat{\mathbf{h}}_{\mathbf{D}} - \mathbf{h}\|_2^2}.$$

Results in terms of SNR loss, in average over 10 antenna array realizations and 1000 channel realizations per antenna array realization are shown on figure 4.1. From the figure, it is obvious that even a relatively small uncertainty about the system configuration can cause a great SNR loss. For example, an uncertainty of 0.03λ on the antenna spacings and of 0.09 on the antenna gains leads to an SNR loss of more than $10\,\mathrm{dB}$, which means that the mean squared

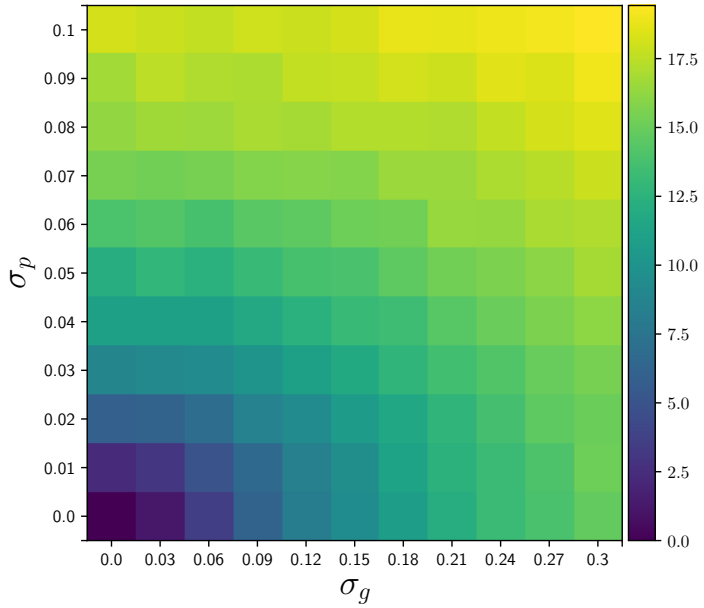


Figure 4.1 – SNR loss in decibels (dB) due to imperfect knowledge of the system.

error undergoes a more than tenfold increase.

This short experiment simply highlights the fact that using imperfect models can severely harm MIMO channel estimation performance. Note that we took here as an example of imperfection the uncertainty about antenna positions and gains, but many other sources of imperfection will impact real world practical systems. The main contribution of this work is to propose a method that takes into account and corrects to some extent imperfect physical models using ML, and more specifically deep unfolding.

4.3 Proposed approach: mpNet

A strategy based on deep unfolding is now proposed, allowing to correct a channel estimation algorithm based on an imperfect physical model incrementally, via online learning.

4.3.1 Unfolding matching pursuit

The MP and OMP algorithms presented in section 1.2.3 are recalled here

Algorithm 2 MP & OMP

Input: Dictionary of atoms $\mathbf{D}$, input $\mathbf{x}$

$\mathbf{r} \leftarrow \mathbf{x}$
if OMP **then**
  $\Lambda \leftarrow \emptyset$
end if
while Stopping criterion not met **do**
  Find the most correlated atom: $s \leftarrow \underset{i}{\text{argmax}} \ |\mathbf{a}_i^{\mathsf{H}}\mathbf{r}|$
  if MP **then**
    Update the residual: $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{a}_s\mathbf{a}_s^{\mathsf{H}}\mathbf{r}$
  else
    if OMP **then**
      Update the set of atoms: $\Lambda \leftarrow \Lambda \cup \{s\}$
      Update the residual: $\mathbf{r} \leftarrow \mathbf{r} - \mathbf{D}_\Lambda \operatorname{argmin}_{\boldsymbol{\alpha}} \|\mathbf{D}_\Lambda\boldsymbol{\alpha} - \mathbf{x}\|_2$
    end if
  end if
end while

Output: $\hat{\mathbf{h}} \leftarrow \mathbf{x} - \mathbf{r}$

for better readability. The estimation strategy relying on MP can be unfolded as a neural network taking the observation $\mathbf{x}$ as input and outputting a channel estimate $\hat{\mathbf{h}}$. Indeed, the first step in the while loop amounts to perform a linear transformation (multiplying the input by the matrix $\mathbf{D}^{\mathsf{H}}$) followed by a nonlinear one (finding the inner product of maximum amplitude and setting all the others to zero) and the second step corresponds to a linear transformation (multiplying by the matrix $\mathbf{D}$). Such a strategy is parameterized by the dictionary of steering vectors $\mathbf{D}$. In the case where the optimal dictionary $\mathbf{D}$ is unknown (or imperfectly known), we propose to learn the dictionary matrix used in (4.5) directly on data via backpropagation (Rumelhart, G. E. Hinton, and Williams 1986).

Neural network structure This is done by considering the dictionary matrix as weights of the neural network we introduce, called mpNet, whose forward pass is given in algorithm 3. The notation HT_1 refers to the hard thresholding operator which keeps only the entry of greatest modulus of its input and sets all the others to zero. The parameters of this neural network are the weights $\mathbf{W} \in \mathbb{C}^{N_R \times N_D}$, where N_D is an hyperparameter denoting the number of con-

sidered atoms in the dictionary. Note that complex weights and inputs are handled classically by stacking the real and imaginary parts for vectors and using the real representation for matrices. The forward pass of mpNet can be seen as a sequence of K iterations whose schematic description is shown on figure 4.2. The stopping criterion determining the number K of replications of the aforementioned structure, which corresponds to the number of estimated paths, is studied in section 4.3.4, with the objective of making the depth of mpNet adaptive to the SNR.

Algorithm 3 Forward pass of mpNet

Input: Weight matrix $\mathbf{W} \in \mathbb{C}^{N_R \times N_D}$, input $\mathbf{x}$

1: $\mathbf{r} \leftarrow \mathbf{x}$
2: **while** Stopping criterion not met **do**
3: $\quad \mathbf{r} \leftarrow \mathbf{r} - \mathbf{W}\,\mathrm{HT}_1(\mathbf{W}^{\mathsf{H}}\mathbf{r})$
4: **end while**

Output: $\mathtt{FW}(\mathbf{W}, \mathbf{x}) \leftarrow \mathbf{x} - \mathbf{r}$

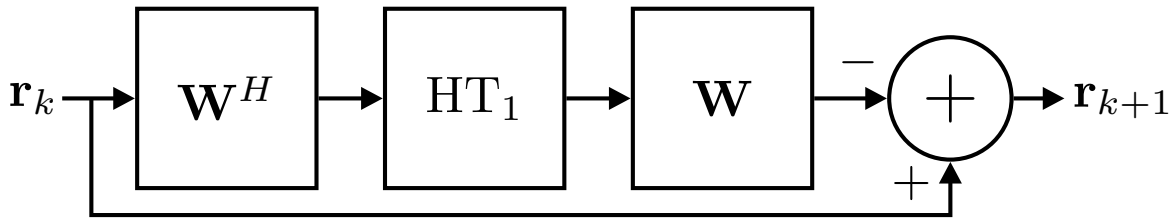


Figure 4.2 – One layer of mpNet.

Highlight 4.2: mpNet as an autoencoder

mpNet can also be seen as an autoencoder such that the last matrix multiplication represents the decoder, while the encoder consists of all anterior steps. Figure 4.3 shows an example with three blocks structured in a way to highlight its autoencoding nature. The latent vector contains all the coefficients selected by the network. In the example, it contains at most three non-zero elements.

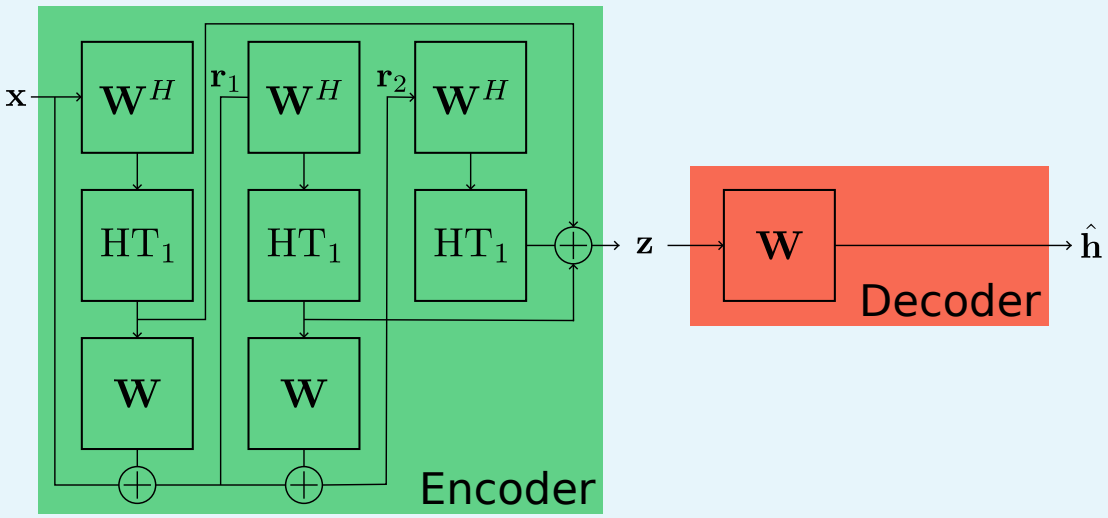


Figure 4.3 – mpNet as an autoencoder.

4.3.2 Training mpNet

The method we propose to jointly estimate channels while simultaneously correcting an imperfect physical model is summarized in algorithm 4. Note that mpNet is fed with normalized inputs of the form $\frac{\mathbf{x}}{\|\mathbf{x}\|_2}$, since we noticed it improved its performance. The output of the network is then multiplied by $\|\mathbf{x}\|_2$ in order to compensate for this initial normalization to get channel estimates of correct norm (see line 5 of algorithm 4). The training strategy amounts to initialize the weights of mpNet with a dictionary of nominal steering vectors $\tilde{\mathbf{D}}$ and then to perform a minibatch gradient descent (Bottou 2010) on

the weights $\mathbf{W}$ to minimize the risk

$$R \triangleq \mathbb{E}\left[\frac{1}{2}\frac{\|\mathbf{x}-\hat{\mathbf{h}}\|_2^2}{\|\mathbf{x}\|_2^2}\right]. \tag{4.9}$$

Denoting $\mathcal{B}$ the current minibatch of size B, the expectation involved in the risk is approximated by computing an average over the minibatch observations, leading to the loss function

$$\mathcal{L} \triangleq \frac{1}{2B}\sum_{\mathbf{x}\in\mathcal{B}}\frac{\|\mathbf{x}-\hat{\mathbf{h}}\|_2^2}{\|\mathbf{x}\|_2^2}. \tag{4.10}$$

Note that this loss function evolves with time, since all minibatches are made of different observations, thus allowing real-time adaptation of mpNet to changes in the channel distribution. The network is trained to minimize the average discrepancy between its inputs and outputs, exactly as a classical autoencoder. It operates *online*, on streaming observations $\mathbf{x}_t$, $t = 1, \ldots, \infty$ of the form (4.3) acquired over time (coming from all users indifferently). Note that, as opposed to the classical unfolding strategies (Gregor and LeCun 2010; Hershey, Roux, and Weninger 2014; Kamilov and Mansour 2016), the proposed method is totally *unsupervised*, meaning that it requires only noisy channel observations and no database of clean channels to run. In all the experiments performed in this work, we use minibatches of $B = 200$ observations and the Adam optimization algorithm (Kingma and Welling 2014). By abuse of notation, we denote $\texttt{Adam}(\mathbf{W}, \frac{\partial\mathcal{L}}{\partial\mathbf{W}}, \gamma)$ the update of the weights $\mathbf{W}$ by the Adam algorithm on $\mathcal{L}$ using the learning rate γ.

4.3.3 Computational complexity

Let us denote K the number of times line 3 of algorithm 3 is executed, which corresponds to the number of estimated channel paths. This number depends on the chosen stopping criterion, which is studied in section 4.3.4. The forward pass of mpNet costs $\mathcal{O}(KN_RA)$ arithmetic operations, its complexity being dominated by the multiplication of the input by the matrix $\mathbf{W}^{\mathsf{H}}$ (first block of figure 4.2). The backpropagation step costs only $\mathcal{O}(KN_R)$ arithmetic operations (N_D times less), since the error flows through only one columns of

Algorithm 4 Online training of mpNet

Input: Nominal dictionary $\tilde{\mathbf{D}} \in \mathbb{C}^{N \times N_D}$, minibatch size B, learning rate γ

Initialize the weights: $\mathbf{W} \leftarrow \tilde{\mathbf{E}}$
Initialize the cost function: $C \leftarrow 0$
for $t = 1, \ldots, \infty$ **do**
  Get observation $\mathbf{x}_t$ following (4.3)
  Estimate the channel (forward pass):
  $\hat{\mathbf{h}}_t \leftarrow \|\mathbf{x}_t\|_2 \, \texttt{FW}\left(\mathbf{W}, \frac{\mathbf{x}_t}{\|\mathbf{x}_t\|_2}\right)$
  Increment the cost function:
  $\mathcal{L} \leftarrow \mathcal{L} + \frac{1}{2B\|\mathbf{x}_t\|_2^2}\|\mathbf{x}_t - \hat{\mathbf{h}}_t\|_2^2$
  if $t \mod B = 0$ **then**
    Update the weights (backward pass):
    $\mathbf{W} \leftarrow \texttt{Adam}(\mathbf{W}, \frac{\partial \mathcal{L}}{\partial \mathbf{W}}, \gamma)$
    Reset the cost function: $\mathcal{L} \leftarrow 0$
  end if
end for

the weight matrix $\mathbf{W}$ at each step, due to the hard thresholding operation done during the forward pass. This short complexity analysis means that jointly estimating the channel (forward pass) and learning the model (backward pass) is done at a cost that is overall the same order as the one of simply estimating the channel with a greedy algorithm (MP or OMP), without adapting the model at all to data (which corresponds to computing only the forward pass). This very light computational cost makes the method particularly well suited to online learning, as opposed to previously proposed channel estimation strategies based on deep learning. Indeed, (He, Wen, et al. 2018b) and (Wei, Hu, and Dai 2019), for example, are both based on modified versions of AMP where certain steps (e.g., shrinkage function) were replaced by trainable neural networks. If AMP ((D. L. Donoho, Maleki, and Montanari 2009)) were to be used in similar settings to ours, its computational complexity would be $\mathcal{O}(K' N_R A)$ which is of the same order as that of MP. Knowing that AMP needs more iterations K' to converge than MP (i.e., $K' > K$), this means that it is indeed more computationally heavy. Moreover, modified versions of AMP could be even slower if their modified parts are particularly computationally demanding. In fact, the denoiser neural network used in (He, Wen, et al. 2018b) as a replacement for the shrinkage function has alone way more trainable parameters than our

network (20 convolutional layers 18 of which use 64 different $3 \times 3 \times 64$ filters). Finally, the fully-connected denoiser neural network used in (van Lier et al. 2020) as a post-treatment of the LS channel estimate has a reasonably low computational complexity when considering only the forward pass. However, when taking account of the offline training overhead, the complexity becomes higher and unsuitable for online channel estimation in a continuously varying environment.

4.3.4 Choosing the stopping criterion

mpNet is the unfolded version of the MP algorithm, and its number K of layers corresponds to the number of iterations of the said algorithm. It also represents the number of estimated channel paths. In fact, K is nothing else but a hyperparameter that needs to be optimized. But how can we determine this number appropriately? A naive approach would be to test different values of K and choose the one yielding the best results in terms of relative error in average over channel observations, by cross-validation. In this case, the number of estimated paths would be the same for every channel observation. For practical systems, this strategy is suboptimal. Indeed, the number of estimated paths should ideally depend on the SNR: the higher the SNR the more paths can be estimated reliably. Since the SNR depends on the distance separating the users and the BS on one hand, and the path loss and gains of the different propagation paths on the other hand, the depth of mpNet should be allowed to vary in order to estimate a number of paths adapted to each channel observation.

In order to do so, let us take advantage of the adaptive stopping criteria proposed for greedy sparse recovery algorithms. In (Cai and L. Wang 2011; Wu, W. Huang, and Chen 2012), the authors show that OMP with the stopping criterion

$$\mathtt{SC}_1 : \|\mathbf{r}\|_2^2 \leq \tilde{\sigma}^2 (N_R + 2\sqrt{N_R \log N_R}), \tag{4.11}$$

with $\tilde{\sigma}^2 \triangleq \frac{\sigma^2}{\|\mathbf{x}\|_2^2}$, is optimal in a support recovery sense. Moreover, for a small number of iterations and an incoherent enough dictionary, MP and OMP give very close results. Hence, we propose to use $\mathtt{SC}_1$ as stopping criterion for mpNet. Implementing this stopping criterion requires knowing the noise variance σ^2

or at least having an estimate $\hat{\sigma}^2$. Fortunately, the noise variance can be estimated quite reliably in MIMO systems (Das and Bhaskar D. Rao 2012). In the sequel, we assume that a perfect noise variance estimate is available ($\hat{\sigma}^2 = \sigma^2$).

Note that practical channels $\mathbf{h}$ and learned weights $\mathbf{W}$ do not follow exactly the generative model used in (Cai and L. Wang 2011; Wu, W. Huang, and Chen 2012) to derive the optimal stopping criterion, so that $\texttt{SC}_1$ may lose its optimality in the case studied here. For this reason, we also propose to use the simpler and more intuitive stopping criterion

$$\texttt{SC}_2 : \|\mathbf{r}\|_2^2 \leq \tilde{\sigma}^2 N_R. \tag{4.12}$$

From a neural network perspective, using the stopping criteria $\texttt{SC}_1$ and $\texttt{SC}_2$ means training a neural network whose depth is adaptive and dynamically adjusted during learning and inference. The structure of figure 4.2 is indeed replicated until the chosen stopping criterion is met. The two stopping criteria $\texttt{SC}_1$ and $\texttt{SC}_2$ are empirically compared on realistic synthetic channels in section 4.4.

4.4 Experimental validation

Let us now assess mpNet on realistic synthetic channels. To do so, we consider the SSCM (section 1.1.4) in order to generate NLoS sparse channels at $28\,\text{GHz}$ (see (Samimi and T. S. Rappaport 2016, table IV)) corresponding to all users. This amounts to generating:

1. the total number of paths L,
2. the DoAs $\{\vec{u_1}, \ldots, \vec{u_L}\}$ associated with each path ,
3. the complex gains $\{\vec{\beta_1}, \ldots, \vec{\beta_L}\}$ associated with each path.

Once generated, those parameters are used to compute the actual channels using equation (4.6). We consider the same setting as in section 4.2, namely a BS equipped with a ULA of 64 antennas, with an half-wavelength nominal spacing and unit nominal gains used to build the imperfect nominal dictionary $\tilde{\mathbf{D}}$ (with $N_D = 8N_R$) which serves as an initialization for mpNet. The actual

antenna arrays are generated the same way as in section 4.2, using (4.7). The generated physical parameters are kept fixed for the whole duration of each experiment, except for section 4.4.3 where we describe how they evolve.

An important consideration is that the way we generate channels corresponds to a setting in which each channel observation is associated with a different user, simulating the initial access of users to the BS. In this way, our model does not need to be aware to which user the channel it is estimating corresponds. This is possible because the dictionary we consider at the BS is common to all users and each of its columns corresponds to the channel response to an excitation in a particular spatial direction (which is the same irrespective of which user is considered). This is in contrast with bayesian channel estimators such as the LMMSE, which require several channel observations of the same user in order to estimate its individual distribution to be injected as prior knowledge.

4.4.1 Fixed SNR

For the first set of experiments, we consider two model imperfections: $\sigma_p = 0.05$, $\sigma_g = 0.15$ (small uncertainty) and $\sigma_p = 0.1$, $\sigma_g = 0.3$ (large uncertainty) to build the unknown ideal dictionary $\mathbf{D}$. The input SNR is fixed during the experiment and takes the values $\{5, 10\}$ dB.

We compare the performance of various configurations of mpNet, namely the version with a fixed number of iterations K, with K set to 6 for the SNR of 5 dB and to 8 for the SNR of 10 dB (determined by cross validation) and the versions with an adaptive stopping criterion, using criteria $\texttt{SC}_1$ and $\texttt{SC}_2$ described in section 4.3.4. In addition, the proposed method is compared to the LS estimator and to the OMP algorithm using the stopping criterion $\texttt{SC}_2$, with either the imperfect nominal dictionary $\tilde{\mathbf{D}}$ or the unknown ideal dictionary $\mathbf{D}$. Finally, in order to demonstrate the advantages of using the imperfect model initialization, we compare it to using the well-known Xavier random initialization (Glorot and Bengio 2010) which assumes no prior knowledge. This baseline corresponds to a classical online dictionary learning method (Mairal et al. 2010).

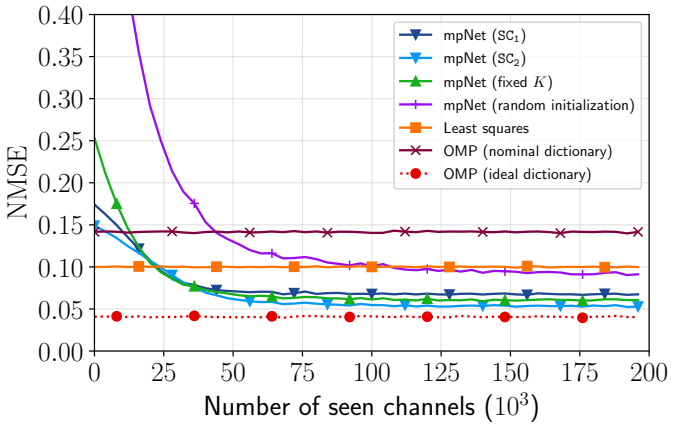
Note that a quantitative comparison with the MMSE estimator is absent

from our experiments as it is not directly possible. Ideally, MMSE requires exact knowledge of the channel correlation matrix to apply. In the absence of that knowledge, an estimation procedure should be conducted which requires a very large number of time coherent samples to be satisfying. On the other hand, our method does not require prior information on the channel statistics since the prior information is entirely based on physics, which is an advantage compared to MMSE.

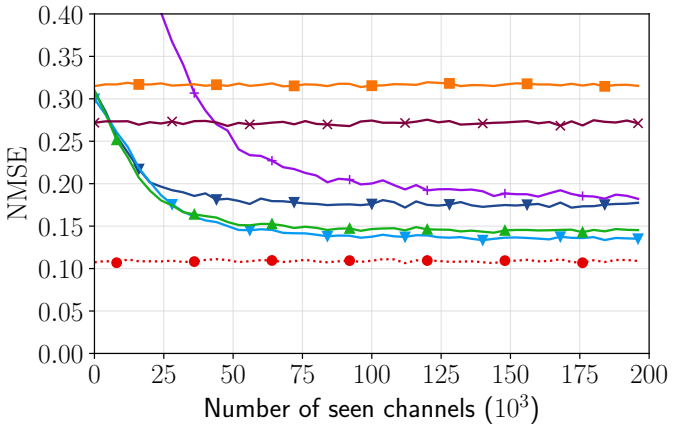
Results The results of this experiment are shown on figure 4.4 as a function of the number of channels of the form (4.3) seen by the BS over time. The performance measure is the normalized mean squared error (NMSE $= \|\hat{\mathbf{h}} - \mathbf{h}\|_2^2 / \|\mathbf{h}\|_2^2$) averaged over minibatches of 200 channels.

Several comments are in order:

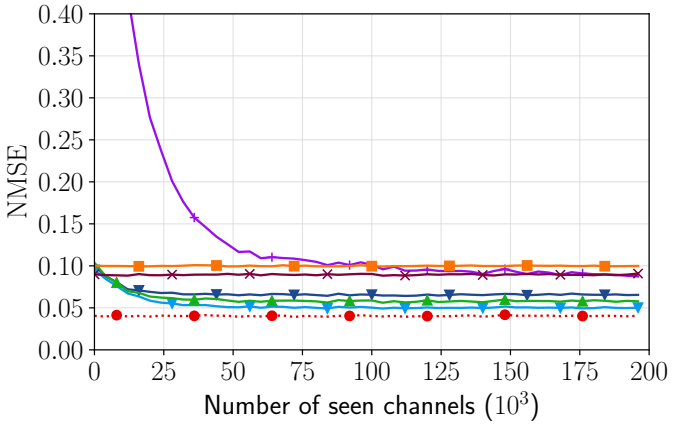
1. The imperfect model is shown to be well corrected by mpNets, the green and the two blue curves being very close to the red one (OMP with ideal unknown dictionary) after a certain amount of time. This is true both for a small uncertainty and for a large one and at all tested SNRs. Moreover, this shows that mpNet, although based on MP, is capable of attaining a performance level close to that of OMP once it converges. Note that using the nominal dictionary (initialization of mpNet) may be even worse than the LS method, showing the interest of correcting the model, since mpNet always ends up outperforming the LS, thanks to the learning process.
2. Comparing figures 4.4a and 4.4b, it is interesting to notice that learning is faster and the attained performance is better with a large SNR (the green and the two blue curves get closer to the red one faster), which can be explained by the better quality of data used to train the model.
3. Comparing the figures 4.4a and 4.4c, it is apparent that a smaller uncertainty, which means a better initialization since the nominal dictionary is closer to the ideal unknown dictionary, leads to a faster convergence, but obviously also to a smaller improvement.
4. Looking at the purple curve on all figures, it appears that initialization matters. Indeed, the randomly initialized network takes longer to converge and performs much worse than the one initialized with the nominal dictionary.



(a) $\mathrm{SNR}_{\mathrm{in}} = 10\,\mathrm{dB}$, $\sigma_p = 0.1$, $\sigma_g = 0.3$, $K = 8$



(b) $\mathrm{SNR}_{\mathrm{in}} = 5\,\mathrm{dB}$, $\sigma_p = 0.1$, $\sigma_g = 0.3$, $K = 6$



(c) $\mathrm{SNR}_{\mathrm{in}} = 10\,\mathrm{dB}$, $\sigma_p = 0.05$, $\sigma_g = 0.15$, $K = 8$

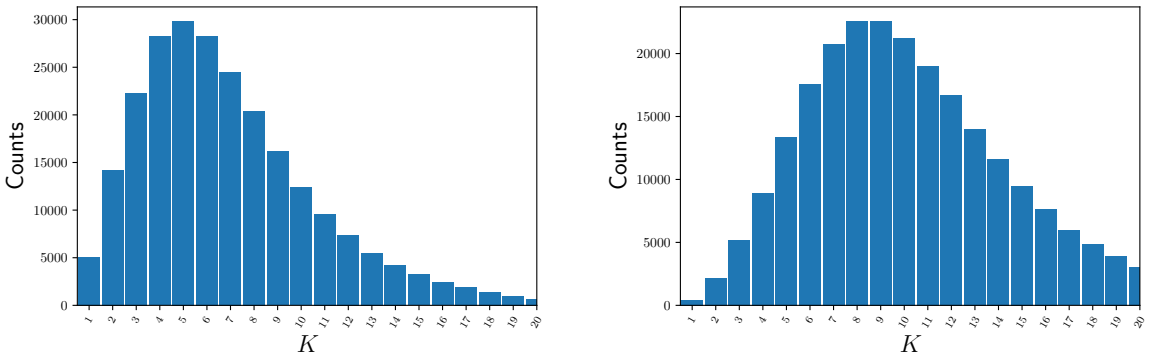
Figure 4.4 – Channel estimation performance on synthetic realistic channels for various SNRs and model imperfections.

5. The green and the two blue curves are all close to each other. In terms of performance, the light blue curve (corresponding to the use of $\texttt{SC}_2$ as a stopping criterion) always leads to the best performance, followed by the green curve (corresponding to a fixed depth) and finally the dark blue curve (corresponding to the use of $\texttt{SC}_1$). Figure 4.4b shows that with a lower SNR, the gap between the dark blue and the rest of the mpNet curves is more pronounced. To further understand the difference between $\texttt{SC}_1$ and $\texttt{SC}_2$, figure 4.5 shows histograms of the selected depths for the experimental settings corresponding to the leftmost figure in figure 4.4. It can be seen that, for $\texttt{SC}_1$, the distribution is centered at $K = 5$ which is 3 iterations behind the network with the best results at a fixed depth. However, for $\texttt{SC}_2$, the distribution is centered at $K = 8$ which is the optimal value for a fixed depth. Moreover, the $\texttt{SC}_2$ version outperforming the fixed depth one is a result of its adaptability to each channel observation.

These results are very promising and highlight the applicability of the proposed method.

4.4.2 Varying SNR

In practical scenarios, the SNR is not fixed, and rather depends on the distance between the transmitter and the receiver as well as propagation conditions. This variability has to be taken into account for the learning algorithm to be efficiently optimized. Using $\texttt{SC}_2$, which takes into account both the noise level and the signal intensity, suggests that our model is capable of automatically adapting to the SNR. To verify this claim, we consider the same SSCM channel model described in the previous section, but this time, the distance between the transmitter and the receiver is sampled from a uniform distribution ranging from $60\,\text{m}$ to $200\,\text{m}$. Our model is then compared to its fixed depth versions for a selected set of values of K ($\{3, 6, 8, 14\}$), as well as to the other previously presented estimation methods. The results are shown in figure 4.6. While all versions of mpNet are capable of learning over time, the adaptive one is clearly outperforming the others throughout the whole experiment, starting once again at the same error level as OMP equipped with the nominal dictionary (the brown curve). Furthermore, compared to the precedent experiment



(a) $\mathtt{SC}_1$ (b) $\mathtt{SC}_2$

Figure 4.5 – Histograms of depths selected by mpNet when equipped with $\mathtt{SC}_1$ and $\mathtt{SC}_2$.

at a fixed SNR, the gap between the fixed depth and the adaptive version is more pronounced. The performance of mpNet with $\mathtt{SC}_2$ is indeed very close in terms of performance in this more realistic scenario to what is achievable with a perfect knowledge of the physical parameters (here represented by the red curve).

Finally, figure 4.7 shows how the SNR of the generated channel observations is distributed. Note that we choose not to consider observations with a low SNR ($< 1\,\text{dB}$) as the model struggles to learn from very noisy instances. In practice, this could be achieved by setting a threshold on the intensity of received signals as a way to filter out observations heavily corrupted. This leads to a truncated normal distribution of the SNR centered at $10\,\text{dB}$.

4.4.3 Anomaly detection and recovery

One of the benefits of online learning is the continuous adaptation of the model to incoming data. If this data is disturbed, the impact would be observed on the loss function used by the system. Indeed, the distribution of the new data would be different from the one on which the system has learned so far. The speed at which the error increases would be proportional to the rate at which the distribution of the new data shifts from its original state. Thankfully, this increase in the error is simultaneously compensated by the training that is done on the network. This behavior may prove useful for detecting and recovering from anomalies. Anomalies could occur for many reasons in a

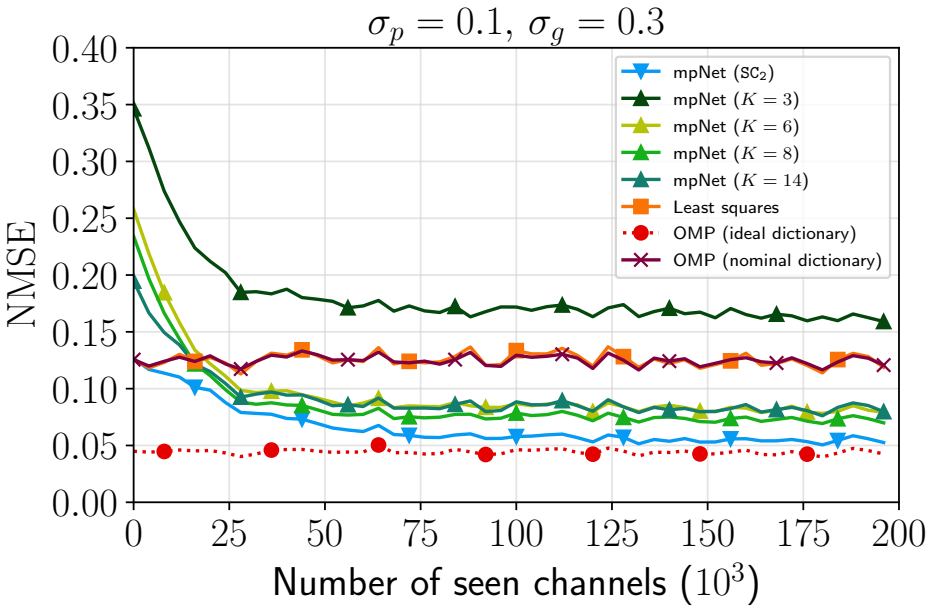


Figure 4.6 – Channel estimation performance on synthetic realistic channels for a varying SNR.

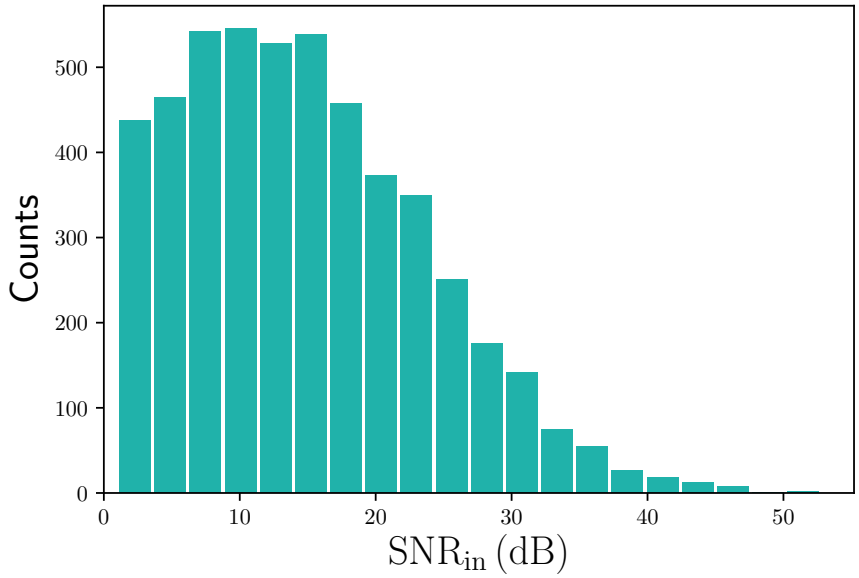
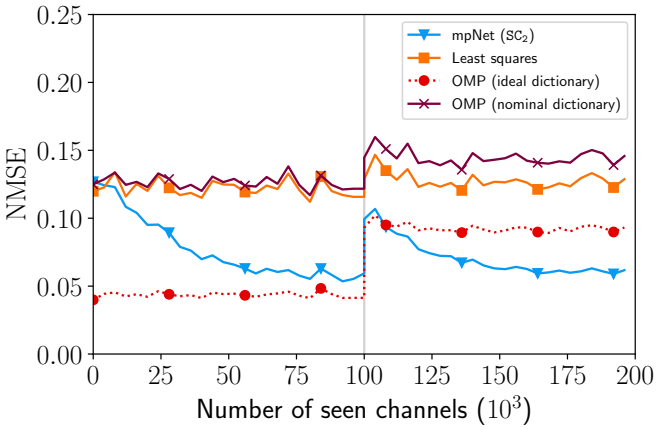
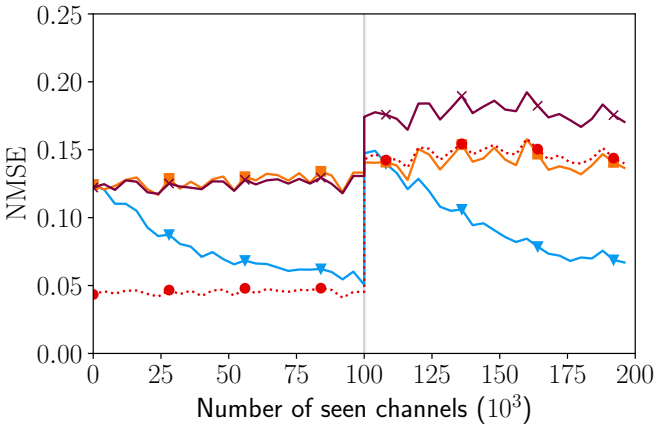


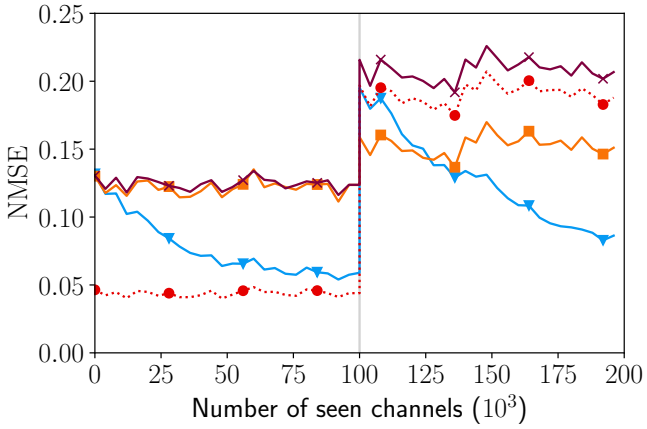
Figure 4.7 – SNR distribution over 5000 generated channel observations.



(a) $\sigma_p = 0.1$, $\sigma_g = 0.3$, broken antennas $= 10\%$



(b) $\sigma_p = 0.1$, $\sigma_g = 0.3$, broken antennas $= 30\%$



(c) $\sigma_p = 0.1$, $\sigma_g = 0.3$, broken antennas $= 50\%$

Figure 4.8 – Adaptation to antenna damage. The horizontal bar at the middle marks the moment at which the break happens.

massive MIMO system, including:

— Bent or broken antennas due to natural causes

— Improperly adjusted antennas after an intervention

— Disoriented array due to wind

In this section, we propose to test the ability of our model to detect and adapt to various types of anomalies. Note that we assume that the antennas' parameters are only known to the BS at the beginning of its life and that any shift from the initial state cannot be detected directly.

Out of order antennas

Let us first simulate antennas that go out of order, in a BS equipped with 64 antennas. This is done by setting some of the antenna gains to zero at a certain point of time during training. The BS will nonetheless still receive noise on out of order antennas. Three scenarios are considered: 10%, 30% and 50% of broken antennas (chosen uniformly at random). Similarly to previous experiments, the channels are generated following the SSCM at a variable SNR (figure 4.7) and the adaptive version of mpNet with $\mathtt{SC}_2$ is compared to the other estimation methods. Figure 4.8 shows the results. It can be seen that the training starts as usual with our model slowly approaching the performance of OMP with the optimal dictionary. The anomaly can be observed at the middle of training when the number of seen channels reaches $\sim$100 000 channels. All the estimation methods see their error jump, with the exception of LS where no change is observed. The amount by which the error increases is proportional to the number of broken antennas. The error starts decreasing again for mpNet as the training resumes, but naturally stays the same for OMP based methods that do not correct the dictionary they use. By the end of the experiment, and depending on the number of broken antennas, mpNet can completely recover from the damage and its error reaches once again the level it successfully attained right before the anomaly.

The LS estimation method does not depend on any physical parameter, which explains the stable error level it maintained throughout the experiment. OMP methods, however, see their error increase because the physical parameters they are based on become less precise and thus induce a bigger error.

Those methods are hence capable of detecting the exact moment where the damage happens but are incapable of adapting. On the other hand, and for the same reason, mpNet is also capable of detecting the anomaly but rapidly adapts its parameters (dictionary). In practice, the detection could be implemented via a simple threshold.

Note that the anomaly recovery only concerns the channel estimation performance. Indeed, a BS with broken antennas will always be less efficient than a fully functional one, especially in terms of channel capacity. In summary, the model does the best it can on the channel estimation task with the available means.

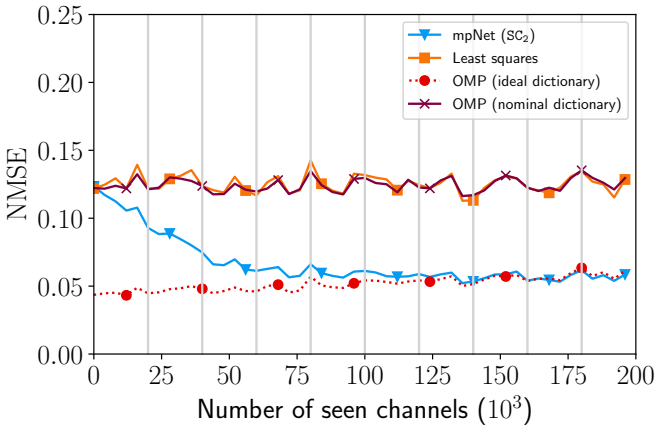
Aging antenna array

The second type of anomaly we consider is antenna aging. It corresponds to antenna gains slowly shifting away from their initial values, it could be due for example to a slow change of temperature throughout the seasons, or a change in weather conditions more generally.

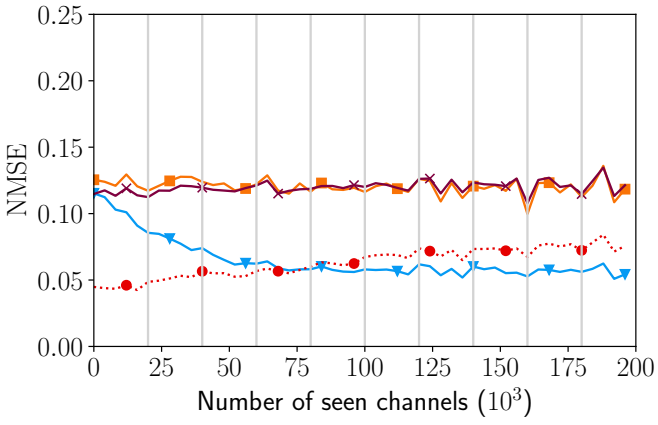
Considering antenna aging, using the initially ideal dictionary will lead to an increase of the error over time since the physical parameters are less and less precisely known. To simulate this phenomenon, we consider the same settings as for the precedent experiment. We then iteratively add noise to the antenna gains over the course of training. This is done for 10 iterations. Antenna gains at iteration t are thus expressed as

$$g_{i,t} = g_{i,t-1} + n_i,\ n_i \sim \mathcal{CN}(0, \sigma_a^2), \tag{4.13}$$

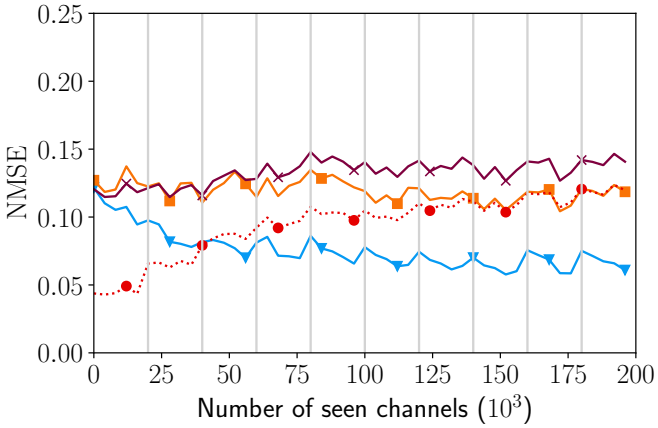
where σ_a could be seen as a measure of the severity (speed) of aging. We consider 3 levels of aging: $\sigma_a = 0.05$ (mild), $\sigma_a = 0.1$ (medium) and $\sigma_a = 0.2$ (severe). Again, channels are generated following the SSCM at a variable SNR (figure 4.7) and the adaptive version of mpNet with $\mathtt{SC}_2$ is compared to the other estimation methods. Results are shown on figure 4.9. We observe that both OMP based estimations see their error progressively increase at a rate proportional to the severity of aging, but performance worsens more rapidly when starting with an ideal dictionary. On the other hand, the neural network continues to learn and the impact of aging is barely noticeable. mpNet's



(a) $\sigma_p = 0.1$, $\sigma_g = 0.3$, $\sigma_a = 0.05$



(b) $\sigma_p = 0.1$, $\sigma_g = 0.3$, $\sigma_a = 0.1$



(c) $\sigma_p = 0.1$, $\sigma_g = 0.3$, $\sigma_a = 0.2$

Figure 4.9 – Adaptation to antenna aging. The horizontal bars mark the different moments at which the aging happens.

online learning compensates for the error induced by the continuous change of physical parameters. Finally, no noticeable change is observed on the LS estimation method.

4.4.4 From ULA to UPA

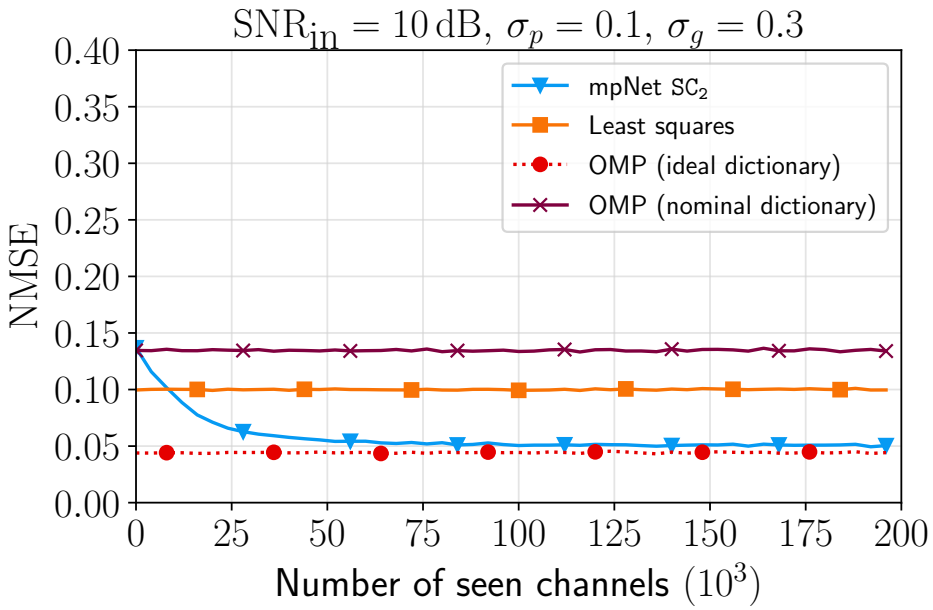


Figure 4.10 – Channel estimation performance on synthetic realistic channels for a BS equipped with a UPA.

The physical model on which is based mpNet is structure-agnostic, meaning that it is meant to work with any antenna array structure. This suggests that our model, which was initially tested on ULAs, is capable of working with any structure as well. To verify this claim, we propose to adapt it to uniform planar arrays (UPAs). A change in the way steering vectors are generated is required to take into account the rotational symmetry that was verified for ULAs and that no longer holds for UPAs. Therefore, the dictionary of steering vectors has to be built from DoAs sampled from the whole 3D half space, instead of a half-plane in the case of ULAs.

We conducted an experiment similar to the ones described in section 4.4. We consider an UPA consisting of a square grid of 8×8 antennas ($N_R = 64$) separated by half-wave lengths and placed on the xz-plane . Channels are

generated at a fixed SNR of $10\,\text{dB}$. To take into account the additional dimension of UPAs, ideal antenna positions are this time given by

$$\vec{a_i} = \tilde{\vec{a_i}} + \lambda \mathbf{n}_{p,i},\ \mathbf{n}_{p,i} = \begin{pmatrix} e_{x,p,i}, & 0, & e_{z,p,i} \end{pmatrix}^T, \tag{4.14}$$

with $e_{x,p,i}, e_{z,p,i} \sim \mathcal{N}(0, \sigma_p^2)$. We compare the adaptive version of mpNet equipped with $\texttt{SC}_2$ to other estimation methods. The results are shown on figure 4.10. Once again, the model successfully learns over time reducing its estimation error while the other classical methods maintain their performances.

This small experiment can be seen as a sanity check to show that, indeed, the model is capable of accepting any antenna structure with minimal changes in the implementation.

4.5 Conclusion and perspectives

In this chapter, we introduced mpNet: a neural network allowing adding flexibility to physical models used for MIMO channel estimation. It is based on the deep unfolding strategy that views a classical algorithm (MP in this case) as a neural network, whose parameters can be trained. The proposed method was shown to incrementally correct (via online learning) an imperfect or imperfectly known physical model in order to make channel estimation as efficient as if the unknown ideal model were known. It is trained in an unsupervised manner as an autoencoder, and mpNet can be seen as a denoiser for channel observations. Training mpNet thus does not necessitate a database of clean channels, nor an offline training phase, which makes it particularly attractive for practical systems and unalike previously proposed methods.

We have shown that initializing the network with a dictionary of imperfect steering vectors (as opposed to using a random initialization) improves performance considerably. In the presented experiments, we simulated the model imperfection by introducing uncertainties on the antenna gains and positions, but it is important to highlight the fact that the method could in principle correct many other model imperfections (such as uncertainties about the antenna diagrams, couplings between neighboring antennas, etc.).

Moreover, we introduced a stopping criterion, inspired by previous work on

the OMP algorithm, to dynamically select the optimal depth of mpNet. This was shown to be particularly convenient when working on observations with a varying SNR level, which more accurately resembles real world channel observations. Evaluated on realistic synthetic data, this approach showed great results compared to other methods.

In addition, online learning enabled us to exploit the observed change in data distribution following an anomaly occurrence to detect and recover from it. We simulated two types of anomalies: antenna damage and aging. In both cases, our model was capable of efficiently recovering from the decrease in performance over time.

Finally, we proved that our model is capable of adapting to any antenna structure with no apparent drop in performance or change in behavior. In particular, we showed that a simple change in the steering vector generation process was required to adapt the model, previously based on a ULA structure, to a UPA structure.

This work opens many perspectives, one of which would be to explore the unfolding of more sophisticated sparse recovery algorithms (such as iterative soft thresholding (Daubechies, Defrise, and De Mol 2004) or AMP (D. L. Donoho, Maleki, and Montanari 2009)) using the same strategy in a way that would mitigate their relatively high computational complexity, making them suitable for online training. In addition, other stopping criteria could be integrated to the model and tested. Also, the isotropic antennas assumption could be dropped and to consider anisotropic antennas where complex gains would depend on the DoA, in which case signals arriving at DoAs where gain is stronger are to be prioritized. The proposed method could also be extended quite readily to handle multi-antenna users by structuring the dictionary used within mpNet as a Kronecker product of steering vectors dictionaries. Another avenue worth exploring is the use of our model together with MMSE estimation. One could imagine using mpNet to obtain cleaner channels than LS estimates when a user first connects to the BS in order to form an empirical covariance matrix that could be used within the framework of MMSE estimation later. Finally, it is worth noting that the deep unfolding of MP initialized with a dictionary based on an imperfect model is by no means limited to the MIMO channel estimation task and could be exploited for other tasks, as long

as an initial model is available.

While this chapter's focus was on turning a classical algorithm (namely MP) into a neural network for channel estimation as an implementation of model-based deep learning, the next chapter will focus on doing the same with existing channel charting methods, although following a different strategy.

Chapter 5

Model-based deep learning for channel charting

Channel charting is the second main topic of this thesis. Approaches proposed in the literature (section 3.4) usually rely on dimensionality reduction to learn a suitable chart. Some use neural networks while others exploit an adequate distance measure. But the task remains challenging. However, no work seems to consider the problem from the point of view of model-based deep learning, that is, combining the acquired knowledge of the problem (e.g., in the form of a distance measure) with the learning capabilities of neural networks.

In this chapter, a hybrid approach to channel charting that builds on earlier ideas (Ferrand, Decurninge, Ordoñez, et al. 2021; Le Magoarou 2021a; Le Magoarou 2021b) is explored. A model-based neural network (Shlezinger et al. 2020) with few parameters is structured as in (Le Magoarou 2021b), allowing a smart initialization of its weights based on the adapted distance measure of (Le Magoarou 2021a) jointly exploited with Isomap. The network is then trained using a triplet loss as in (Ferrand, Decurninge, Ordoñez, et al. 2021).

The idea behind the smart initialization is to exploit as much prior information as possible in order to put the model in the best configuration prior to any optimization. This initialization naturally arises from the structure imposed on the network. Training is expected to gradually lead the network to configurations yielding even better results. To do so, the triplet loss relies on the way the channels are collected, bringing another source of information into the mix. The expected result is a reduction in the computational cost of both training and inference when compared to existing models, in an attempt to compute channel charts on the fly.

This chapter start by recalling the channel charting problem. The proposed hybrid approach is then detailed in three parts: the distance measure, the hybrid encoder and the triplet loss. Finally, to assess the performance of the model, experiments are conducted on synthetic datasets that are detailed and the results are analyzed.

5.1 Problem formulation

We consider a BS equipped with an antenna array of N_R antennas, and one or more single antenna UEs placed on a map and sending signals to the BS on N_s evenly spaced subcarriers and a central frequency f_c. We denote $\mathbf{h}_i \in \mathbb{C}^M$ the ith transmitted uplink channel vector, with $M = N_R N_s$, and $\mathbf{p}_i \in \mathbb{R}^D$ the corresponding transmit location, where D corresponds to the number of retained spatial dimension (2 or 3 typically). A database $\{\mathbf{h}_i\}_{i=1}^N$ of N channel vectors is constructed through the collection of channel observations from different locations. As explained in chapter 3, channel charting aims at learning a function $\mathcal{F}$ that takes as input the channel vectors $\mathbf{h}_i$ and produces low-dimensional representations denoted $\mathbf{z}_i$.

$$\begin{aligned} \mathcal{F} \colon \mathbb{C}^M &\to \mathbb{R}^D \\ \mathbf{h}_i &\mapsto \mathcal{F}(\mathbf{h}_i) = \mathbf{z}_i. \end{aligned}$$

Note that the learned representations $\{\mathbf{z}_i\}_{i=1}^N$ are not necessarily of same dimension as the UE locations and are allowed to have a different number of dimensions, but are typically chosen to be two-dimensional.

5.2 Distance measure

The first step in our approach is to compute the distance matrix of the collected channel vectors. The adopted distance measure needs to correctly reflect the local spatial neighborhoods of channel observations. In addition to the reasons detailed in section 3.4.2, the traditional Euclidean distance is inadequate to the task because of its unusual behavior in high-dimensional spaces (Aggarwal, Hinneburg, and Keim 2001). A more adequate distance

measure is the phase-insensitive channel distance introduced in (Le Magoarou 2021a) and presented in chapter 3. As a reminder, it is defined as follows:

$$d_{\text{PI}}(\mathbf{h}_i, \mathbf{h}_j)^2 = 2 - 2\frac{|\mathbf{h}_i^{\text{H}}\mathbf{h}_j|}{\|\mathbf{h}_i\|_2\|\mathbf{h}_j\|_2}. \tag{5.1}$$

Based on the plane wave physical model, this distance measure removes the sensitivity of channel vectors to the fast variations of the global phase, making it reliable and suitable for retrieval of local neighborhoods. In addition, a normalization factor is introduced to mitigate the detrimental effect of signal intensity.

5.3 Hybrid encoder

Now that a distance measure has been defined, one could use a generic manifold learning algorithm to project the channel vectors into the low-dimensional space. For instance, (Le Magoarou 2021a) relies on Isomap to obtain the channel chart. However, this approach suffers from two main:

(1) Manifold learning algorithms like Isomap are incapable of directly performing out-of-sample projections, meaning that projecting new unseen samples would require recomputing the projection for the whole dataset which is very inefficient and cumbersome.

(2) These methods depend on the accuracy of the distance measure and are prone to errors due to its approximative nature.

To fix the first issue, we propose the following strategy (summarized in algorithm 5) that resembles the one introduced in (Le Magoarou 2021b). We begin by selecting a small subset of N_{init} collected channel observations that we organize in a matrix $\mathbf{D} \in \mathbb{C}^{M \times N_{\text{init}}}$ as column vectors. The more representative it is of the data distribution the better. We then compute its distance matrix using (5.1) and feed it to Isomap to produce an initial channel chart $\mathbf{Z}$ according to the method proposed in (Le Magoarou 2021a). For a given new observation $\mathbf{h}$ that we would like to project, we compute the modulus of its correlation to each one of the channel vectors in $\mathbf{D}$. We keep the k largest elements of the obtained vector using the hard thresholding nonlinear operator

denoted $\mathrm{HT}_k(.)$. We normalize the result using an l_1-norm so that the vector elements sum up to 1. Finally, we multiply the normalized vector by the matrix $\mathbf{Z}$ which amounts to performing a weighted average of the projections of the k most correlated channel vectors to the input vector $\mathbf{h}$.

Algorithm 5 Proposed encoder

Input: Subset of channel vectors $\mathbf{D}$, corresponding initial channel chart $\mathbf{Z}$, channel $\mathbf{h}$ to locate on the chart

1: Correlation: $\mathbf{a} \leftarrow \mathbf{D}^{\mathsf{H}}\mathbf{h}$
2: Modulus: $\mathbf{b} \leftarrow |\mathbf{a}|$
3: Hard thresholding: $\mathbf{c} \leftarrow \mathrm{HT}_k(\mathbf{b})$
4: Normalization: $\mathbf{d} \leftarrow \mathbf{c}/\|\mathbf{c}\|_1$
5: Weighted sum: $\mathbf{z} \leftarrow \mathbf{Z}\mathbf{d}$

Output: $\mathbf{z} = \mathcal{F}(\mathbf{h})$

This strategy makes it possible to project individual channel observations independently of the rest of the dataset. Besides being initialized from a small dataset in conjunction with Isomap, it comprises two matrix multiplications separated by nonlinear operations which makes it easily laid out as a neural network as depicted in figure 5.1.

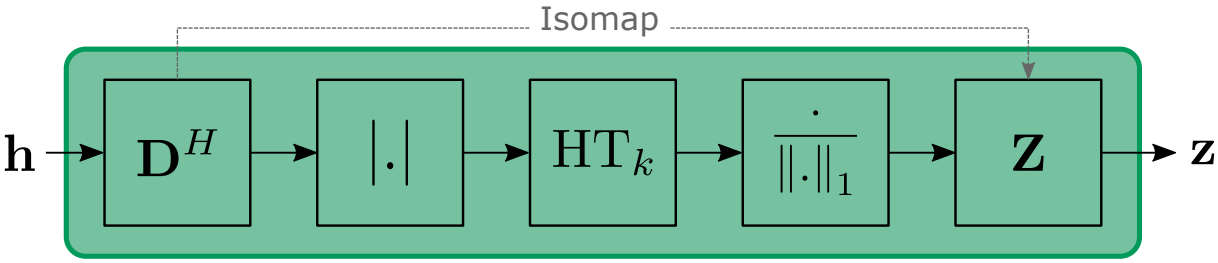


Figure 5.1 – Hybrid encoder.

The second issue is mitigated with the help of training and is tackled in the next section. Indeed, the weight matrices $\mathbf{D}$ and $\mathbf{Z}$ will be allowed to vary during the training in order to optimize the whole model.

5.4 Triplet loss

Inspired by the work in (Ferrand, Decurninge, Ordoñez, et al. 2021), we investigate the use of triplet networks (section 2.4.2) to train our hybrid en-

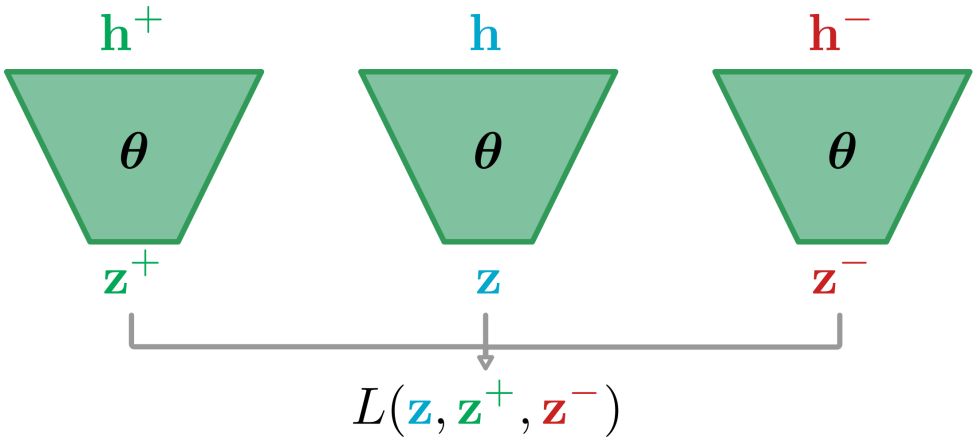


Figure 5.2 – Triplet network structure. $\boldsymbol{\theta}$ is the set of parameters (i.e. weights) that the neural network learns throughout training.

coder in an unsupervised way. In this case, a single channel triplet comprises an anchor sample $\mathbf{h}$, a close sample $\mathbf{h}^+$ and a far sample $\mathbf{h}^-$. The network produces their corresponding projections into the channel chart $\mathbf{z}$, $\mathbf{z}^+$ and $\mathbf{z}^-$ respectively as depicted on figure 5.2. The loss function is defined as

$$\mathcal{L}(\mathbf{z}, \mathbf{z}^+, \mathbf{z}^-) = \max(0, d^+ - d^- + m), \tag{5.2}$$

where $d^+ = \|\mathbf{z} - \mathbf{z}^+\|_2$, $d^- = \|\mathbf{z} - \mathbf{z}^-\|_2$ and m is a margin parameter. The network is trained to minimize it.

Instead of using a generic MLP as in (Ferrand, Decurninge, Ordoñez, et al. 2021), we propose to use the hybrid encoder presented in section 5.3 initialized with a small subset of channel observations following the proposed strategy. The parameters (i.e., weights) of the network $\boldsymbol{\theta} = \{\mathbf{D}, \mathbf{Z}\}$ are then optimized through the training of the encoder in the triplet configuration. The learned charting function is denoted $\mathcal{F}_{\boldsymbol{\theta}}$.

It is important to note that constructing triplets obviously requires knowledge of positive and negative samples for a given anchor sample. In the unsupervised setting, this knowledge has to come from the dataset of inputs itself. Section 5.5.2 describes how channel triplets are extracted from such a dataset.

5.5 Experiments

5.5.1 Scenarios

DeepMIMO

To assess the performance of the proposed method, we rely on synthetic data generated from the DeepMIMO « O1 » outdoor ray-tracing scenario (A. Alkhateeb 2019). A total of $N = 5910$ channels are generated and their transmit locations are geographically distributed along a path representing the route of a pedestrian (figure 5.3). The pedestrian starts their journey at the lower rightmost point of the path and navigates counterclockwise. The pedestrian walks at a speed of $v = 1.4$ m/s. The sampling rate f_s is set at 7 samples/s which means that consecutive samples are separated, on average, by $1.4/7 = 20$ cm along the path. In addition, mild noise is added perpendicularly to the direction of travel to better emulate the walking of a real person that would not follow a perfectly straight line.

A single BS (BS 16 in the original scenario) equipped with a UPA of $N_R = 64$ half-wavelength spaced antennas collects the uplink channels at a central frequency of $f_c = 3.5$ GHz, with $N_s = 16$ evenly spaced subcarriers spanning a band of 20 MHz. The resulting complex channels are of dimension $M = 1024$.

Sionna

Another experiment is conducted on a scenario generated using the Sionna framework. The considered environment is the campus of INSA Rennes modeled in 3D. The data for this modeling were generated using the OpenStreetMap (OpenStreetMap contributors 2017) open-source tool. The BS is equipped with an 8×8 UPA and is located 10 meters above the roof of a building. Communication operates at a carrier frequency of $f_c = 3.5$ GHz using OFDM, considering 16 sub-carriers.

5.5.2 Triplet dataset construction

Special care should be taken when constructing the triplet dataset as it is a critical component of the model. In the unsupervised setting, mining close and

far examples w.r.t each anchor sample can be challenging in the absence of meaningful information to guide the process. However, as shown in (Ferrand, Decurninge, Ordoñez, et al. 2021), the time-correlated nature of channel observations can be exploited (section 3.4.1). Indeed, each channel observation is accompanied by a timestamp corresponding to the exact moment at which it was collected by the BS. The DeepMIMO dataset not providing timestamps, we rely on both the sampling rate and the temporal thresholds to deduce a threshold expressed in terms of the number of preceding and following samples w.r.t. the anchor sample. The inner and outer thresholds are expressed as $S_c = T_c \times f_s$ and $S_f = T_f \times f_s$ respectively. For example, with $f_s = 7$ samples/s and $T_c = 100\,\text{s}$, S_c amounts to 700 samples, meaning that the 700 preceding samples and the 700 following ones, distributed over a total distance of $2 \times v \times T_c = 280$ m, are considered close samples for the triplet construction. The same procedure is used to determine valid far candidates. For the Sionna scenario, $f_s = 28$ samples/s, $T_c = 100\,\text{s}$ and T_f is not considered.

Knowing that samples are indexed in the order that their corresponding locations appear on the path, a single triplet (i, j, k), where i, j and k are the indices of the anchor sample, the close sample and the far sample respectively, is constructed so that $i \in [0, N-1]$, $j \sim \mathcal{U}_{[i-S_c, i[\cup]i, i+S_c]}$ and $k \sim \mathcal{U}_{[i-S_f, i-S_c[\cup]i+S_c, i+S_f]}$.

Note that we can generate multiple triplets for each anchor sample simply by sampling different close and far samples, allowing us to greatly expend the dataset with no added cost. For example, 20 triplets are generated for each sample for the Sionna scenario.

5.5.3 Results

The method is first assessed on the DeepMIMO dataset. The evaluation is done based on the contribution of three aspects: *initialization*, *structure* and *training*. An experiment is conducted for each aspect. The base model consists of the hybrid triplet network introduced in sections 5.3 and 5.4. The matrix D is initialized using a subset of $N_{\text{init}} = 100$ channel observations randomly sampled from $\{\mathbf{h}_i\}_{i=1}^{N}$. The matrix $\mathbf{Z}$ is computed using Isomap set to keep 5 neighbors per sample. The hard thresholding nonlinearity HT_k is set to pick the $k = 5$ largest elements. The thresholds are set to $T_c = 100$ s and

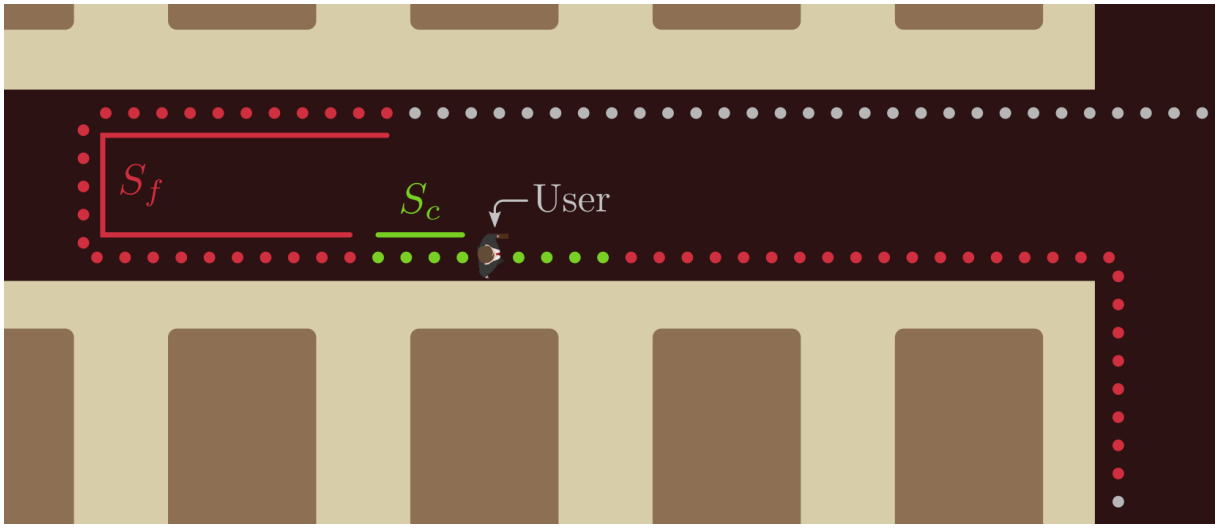


Figure 5.3 – Locations corresponding to a triplet's close (green) and far (red) candidates. The anchor location is where the user is placed.

$T_f = 290$ s, and 5 triplets are generated for each column in D. The margin parameter in the triplet loss is set to $m = 1$. All models are trained on 70% of the dataset for 30 epochs. Performance of the base model in particular is evaluated over 100 runs with different columns in D each time, as we have observed results of high variance compared to the other configurations. The evaluation is done on the 30% unseen samples. This is in contrast to previous work where performance is evaluated on training data.

First, the base model is compared to a model with the same structure with the difference that the weight matrices are randomly initialized using the well-known Xavier initialization (Glorot and Bengio 2010). Figure 5.4 shows the performance of both models in terms of TW and CT (section 3.3). We observe that the smartly initialized model, on average, achieves performance levels comparable to those of the randomly initialized one. However, results vary highly between runs, especially when it comes to TW. This means that choosing the right combination of columns to keep in D is critical and is something to be explored in future work. It is interesting to note that at $K = 5\%$, the CT score begins to degrade for both models. This behavior has to be linked to the thresholds S_c and S_f imposed during the triplet selection process.

Next, the base model is compared to an MLP composed of 6 hidden layers of sizes 1024, 512, 256, 128, 64 and 2 (output). All layers but the last one

are followed by a ReLU nonlinearity. Figure 5.4 shows that the base model performs much better both in terms of TW and CT. Another advantage that the base model has over the MLP is the low number of trainable parameters. The MLP has $2\,793\,600$ trainable parameters while the hybrid model only has $409\,800$ parameters. Therefore, the proposed structure is suitable for applications where computing power and time constraints are critical, which is often the case in wireless communication systems. Indeed, this model may be considered for computing channel charts on the fly to be used by subsequent tasks (e.g., handover). An additional advantage of using light models such as the one presented in this work is the ability to train the network on the fly (or online) on the continuous stream of collected channel observations to adapt to the changing propagation environment. This aspect is yet to be explored.

Finally, the base model's performance after training is compared to its performance before training. It is clear that the training plays a significant role in improving the results for both TW and CT. Interestingly, the figure clearly shows that CT's variance in particular is reduced after training. Another important observation in favor of the smartly initialized model is that its untrained version still performs better than the trained MLP. In addition, this untrained model naturally performs better than its untrained randomly initialized equivalent (not shown on the figures, near 0.5 for both TW and CT), meaning that less training is required to achieve a desired level of performance when the model is smartly initialized.

Figure 5.5 shows the learned channel charts of the different models compared to the real physical locations on the path that the UE follows. Although chart visualizations are subject to personal interpretation, we can nonetheless observe that local neighborhoods are more or less preserved depending on the model. For example, it appears that the chart learned by the smartly initialized model better preserves the structure of long segments of the path (blue, green, orange, brown and pink points) while points belonging to short segments are more easily mixed with the others (purple, olive and red points).

Figure 5.6 shows the learned channel charts of the smart model before and after training on the Sionna dataset. While both are of good quality, it is once again clear that training helps improve and refine the result.

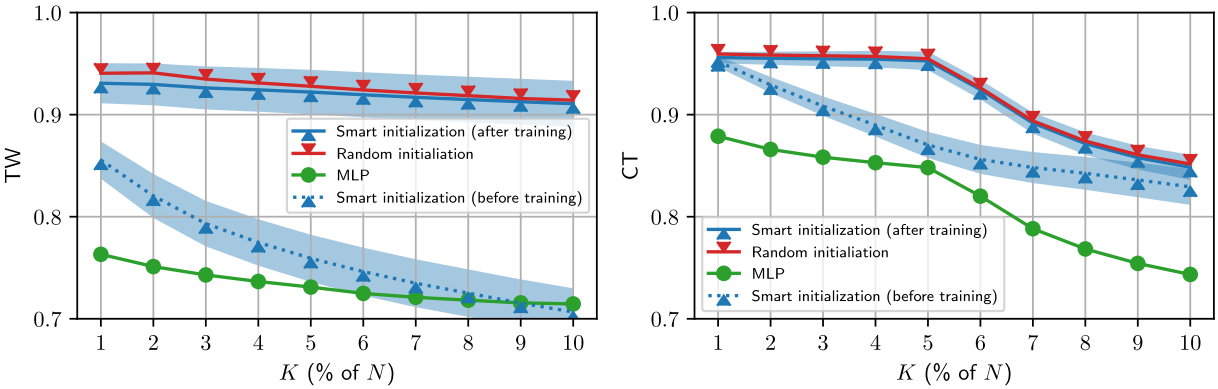
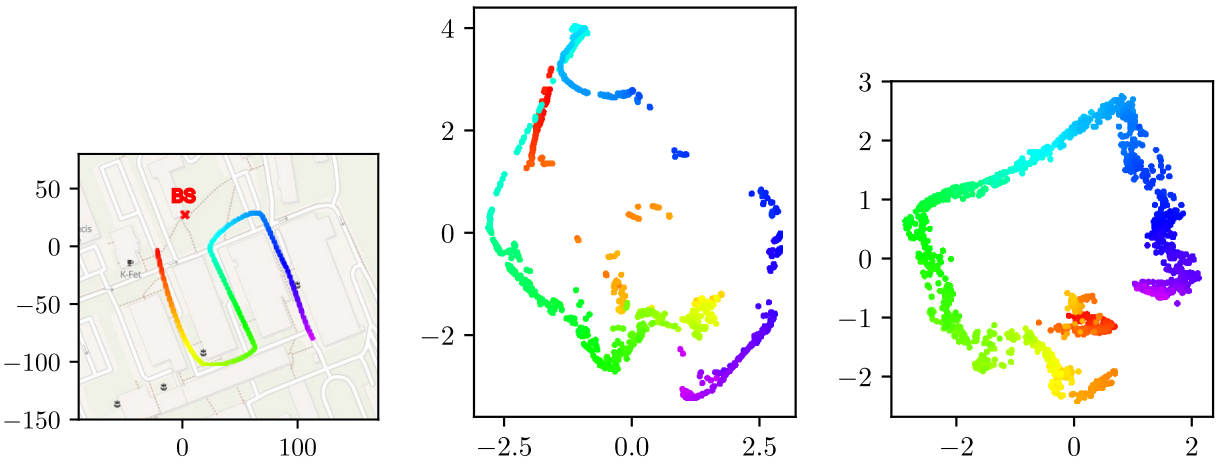


Figure 5.4 – Comparison of the models in terms of TW and CT. The K parameter represents the size of the neighborhood considered for computing both metrics. Regarding the smartly initialized model, the lines represent the means over 100 runs, and the shaded regions represent the one-standard deviation bounds. Note that the lines of the smartly initialized model after training and the randomly initialized one overlap.



(a) UE's locations (b) Smart (c) Random (d) MLP

Figure 5.5 – Channel charts of the different models compared to the ground-truth locations. DeepMIMO dataset.



(a) UE's locations (b) Before training (c) After training

Figure 5.6 – Channel charts before and after training compared to the ground-truth locations. Sionna dataset.

5.6 Conclusion and perspectives

In this chapter, a triplet hybrid network was proposed for the task of channel charting with the aim of reducing the computation cost, hereby allowing for on-the-fly computation and update in portable devices. The model benefits from smart initialization using a small set of channel observations, an adapted distance measure and a manifold learning algorithm like Isomap. Training the network using the triplet loss further improves performance. Furthermore, the model was shown to require fewer parameters compared to a classical MLP.

The results are encouraging and show great potential for improvement. First, initialization could be improved by judiciously choosing the channel vectors to keep in the initial subset. Second, the database could be virtually augmented at no cost by constructing multiple triplets for each anchor sample by sampling different close and far samples. However, not all triplets are created equal, and some are more meaningful than others (e.g. ones containing hard-negative samples). Finding a way of selecting the optimal triplets for each sample could help with the training. Finally, extensive hyperparameter tuning could help discover a better configuration of the model.

As we have seen in this chapter, the channel distance measure is an important part of the hybrid model. While the training should compensate for

its shortcomings, it is still important to understand its limitations in order to come up with measures to mitigate them. This is what the next chapter is about.

CHAPTER 6

SYSTEM PARAMETRIZATION FOR CHANNEL CHARTING

Among CC methods, some explicitly rely on a distance measure between channels to build the chart (section 3.4.2), like the one presented in the previous chapter. For them to be successful, the considered channel distance must allow to reliably detect if any two channel measurements in a given area were taken close to each other or not. In other words, the spatial neighborhoods within the considered area should be *identifiable* by the channel distance measure. A distance with such characteristics is not trivially achievable due to the nature of wave propagation. The PI distance, for example, suffers from ambiguities that degrade charting performance as we will see. One can then raise the following two questions: *What distance is the most adequate for identifiability? And how can a multicarrier MIMO system be calibrated and parameterized to guarantee it?*

In this chapter, MIMO OFDM systems are considered for the task of CC in LoS conditions. First, the notion of identifiability of an area by a channel distance is rigorously defined and its importance for CC is discussed. Then, the PI distance introduced in (Le Magoarou 2021a) and used for CC in (Le Magoarou, Yassine, et al. 2022a; Yassine, Le Magoarou, et al. 2022) is thoroughly analyzed, highlighting its dependency on system parameters such as antenna positions and subcarrier distribution. Conditions to achieve identifiability of an area that link it to these system parameters using the PI distance are given. Special system configurations easing identifiability and consequently particularly suited for CC are proposed, and practical guidelines are provided. Finally, experiments are conducted on both synthetic and realistic channels, validating all the theoretical findings of the study.

Although the communication capabilities of the designed system aren't explored in this work, it could nonetheless be framed as an ISAC problem (Wymeersch et al. 2021). Indeed, the focus here is on adapting an existing communication system to CC, which can be seen as a form of sensing. Similarly to (Mateos-Ramos et al. 2022), the approach presented here could be easily extended to account for a trade-off between communication and CC.

6.1 Problem formulation

In this section, the considered setting is introduced, the CC task is recalled and the notion of identifiability is properly defined and discussed.

6.1.1 System and channel model

Let us consider a BS equipped with N_R antennas, located at positions $\{\vec{a}_1, \dots, \vec{a}_{N_R}\}$ relative to the barycenter of the antennas, which is considered at the origin. The BS communicates with single antenna UEs over N_s evenly spaced OFDM subcarriers $f_1, \dots, f_{N_s}$. Let Δ_f denote the subcarrier spacing, $B \triangleq N_s \Delta_f$ denote the total bandwidth, $f_c \triangleq \frac{1}{N_s} \sum_i f_i$ the central frequency and $\lambda \triangleq \frac{c}{f_c}$ the central wavelength.

In order to make the theoretical analysis of this chapter clear and interpretable, let us consider a two dimensional (2D) configuration in which the BS antennas and the users are at the same height (all in the same plane) and free space propagation. In this case and based on the general model of section 1.1.2, the channels have a single LoS path, so that the uplink channel between a user at location $\vec{x} = (r, \theta)$ (in polar coordinates) and the nth antenna of the BS on the mth subcarrier can be expressed as

$$h_{mn}(\vec{x}) = h_{mn}(r, \theta) = \frac{1}{r} \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\left(r\frac{f_m}{f_c} - \vec{a}_n \cdot \vec{u}(\theta)\right)}, \tag{6.1}$$

where $\vec{u}(\theta) = (\cos(\theta), \sin(\theta))$ (in cartesian coordinates) is the unit vector pointing towards the location $\vec{x}$. The complete channel considering all BS antennas and

subcarrier can then conveniently be expressed in vector form as

$$\mathbf{h}(\vec{x}) = \frac{\sqrt{N_R N_s}}{r} \mathrm{e}^{-\mathrm{j}2\pi \frac{r}{\lambda}} \mathbf{f}(r) \otimes \mathbf{a}(\theta) \in \mathbb{C}^{N_R N_s}, \tag{6.2}$$

where $\mathbf{f}(r)$ and $\mathbf{a}(\theta)$ are the frequency signature vector and the steering vector respectively, defined as

$$\mathbf{f}(r) \triangleq \frac{1}{\sqrt{N_s}} \left(\mathrm{e}^{-\mathrm{j}2\pi \frac{r}{c}(f_1 - f_c)}, \ldots, \mathrm{e}^{-\mathrm{j}2\pi \frac{r}{c}(f_{N_s} - f_c)} \right)^\top, \tag{6.3}$$

and

$$\mathbf{a}(\theta) \triangleq \frac{1}{\sqrt{N_R}} (\mathrm{e}^{\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_1 . \vec{u}(\theta)}, \ldots, \mathrm{e}^{\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_{N_R} . \vec{u}(\theta)})^\top. \tag{6.4}$$

For a ULA in 2D, $\vec{a}_n$ is given as $(0, (n-1)\Delta_r \lambda)$ where Δ_r is the normalized antenna separation. Similarly, for a UCA, $\vec{a}_n$ is given as $(R\cos(\frac{2\pi(n-1)}{N_R}), R\sin(\frac{2\pi(n-1)}{N_R}))$ where R is the radius of the array.

6.1.2 Channel charting

Let us denote $M = N_R N_s$ the considered channel dimension. As introduced in chapter 3, CC aims at learning a function $\mathcal{F}$ that maps the channel vectors of UEs in an area of interest A to a low-dimensional space of dimension $D \ll M$,

$$\begin{aligned} \mathcal{F} : \ & \mathbb{C}^M \to \mathbb{R}^D \\ & \mathbf{h} \mapsto \mathcal{F}(\mathbf{h}) = \mathbf{z}, \end{aligned}$$

where $\mathbf{z}$ is the chart location associated to $\mathbf{h}$. CC methods are assessed on their ability to preserve spatial neighborhoods, that is, how well the original spatial distances match the distances on the learned chart for close points

$$\|\vec{x}_i - \vec{x}_j\|_2 \approx \|\mathbf{z}_i - \mathbf{z}_j\|_2. \tag{6.5}$$

The ability to preserve the global geometry can also be considered. Performance measures such as CT, TW and KS are used to assess the quality of such charts (see section 3.3 for more details). They are calibrated using a

database of channel measurements at different locations

$$\{\mathbf{h}_i\}_{i=1}^N = \{\mathbf{h}(\vec{x}_i)\}_{i=1}^N,$$

N being the number of measured channels (training samples), and the two notations can be used indifferently, depending on the wish for brevity or explicitness.

Similarly to methods presented in section 3.4.2, this work focuses on learning a chart based on a distance measure $d : \mathbb{C}^M \times \mathbb{C}^M \to \mathbb{R}_+$ between pairs of channel vectors (or features derived from it) that tries to match the Euclidean distances between the real locations as closely as possible such that

$$\|\vec{x}_i - \vec{x}_j\|_2 \approx d(\mathbf{h}_i, \mathbf{h}_j). \tag{6.6}$$

Such a distance is then used to produce the chart. In general, a distance matrix between all pairs of training channels is computed:

$$\mathbf{D} \in \mathbb{R}^{N \times N}, \text{ with } d_{ij} = d(\mathbf{h}_i, \mathbf{h}_j),$$

and subsequently used as the input to a dimensionality reduction algorithm, such as Isomap (section 2.4.1), capable of producing the final chart.

6.1.3 Identifiability

Using an appropriate distance measure is necessary to build good charts. In particular, the distance should allow to detect spatial neighborhoods. In order to make this statement more precise, let us define the identifiability of a set of channel observations by a distance.

Definition 6.1: Strong Identifiability

A set of channel observations and their corresponding locations $S \subset \mathbb{R}^2$ are said to be strongly identifiable by the channel distance d iff $\forall \vec{x}, \vec{y}, \vec{z} \in S$

$$d(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x})) > d(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x})) \Leftrightarrow \|\vec{y} - \vec{x}\|_2 > \|\vec{z} - \vec{x}\|_2$$

This definition is described as *strong* because it amounts to finding a channel distance bijective with the Euclidean distance. Since achieving this is often challenging, a *weak* version of the problem is presented.

Definition 6.2: Weak Identifiability

A set of channel observations and their corresponding locations $S \subset \mathbb{R}^2$ is said to be weakly identifiable by the channel distance d iff $\forall \vec{x}, \vec{y}, \vec{z} \in S$

$$d(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x})) > d(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x}))$$
$$\Rightarrow |r_y - r_x| > |r_z - r_x| \text{ or } |\theta_y - \theta_x| > |\theta_z - \theta_x|$$

This weak version of identifiability is defined in terms of the polar coordinates to better suit the mathematical model underlying the propagation channels. Obviously, strong identifiability implies weak identifiability, so that weak identifiability is necessary for strong identifiability.

For a CC method based on a distance d to perform well on an area A, identifiability is an important pre-requisite. Indeed, it is impossible for the method to preserve spatial neighborhoods (which is its objective) if the channel distance measure it is based on already fails to do so. In the subsequent parts of this chapter, the phase-insensitive distance measure introduced in (Le Magoarou 2021a) is analyzed, leading to the statement of system setting guidelines and a thresholding procedure in order to favor weak identifiability. For two UEs communicating with the BS with corresponding channel vectors $\mathbf{h}_i$

and $\mathbf{h}_j$, this distance is defined as

$$d^\star(\mathbf{h}_i, \mathbf{h}_j) \triangleq \sqrt{2 - 2s^\star(\mathbf{h}_i, \mathbf{h}_j)}, \tag{6.7}$$

where $s^\star(\mathbf{h}_i, \mathbf{h}_j) \triangleq \frac{|\mathbf{h}_i^\mathsf{H}\mathbf{h}_j|}{\|\mathbf{h}_i\|_2\|\mathbf{h}_j\|_2}$ is a similarity measure.

The first step of the analysis of the distance is to notice that it can nicely be decomposed into a radial and an angular term. Indeed, when injecting (6.2) one can highlight the following decomposition:

$$\begin{aligned}
\frac{|\mathbf{h}_i^\mathsf{H}\mathbf{h}_j|}{\|\mathbf{h}_i\|_2\|\mathbf{h}_j\|_2} &= \left|\left(\mathbf{f}(r_i) \otimes \mathbf{a}(\theta_i)\right)^\mathsf{H} \left(\mathbf{f}(r_j) \otimes \mathbf{a}(\theta_j)\right)\right| \\
&\overset{(a)}{=} \left|\left(\mathbf{f}(r_i)^\mathsf{H} \otimes \mathbf{a}(\theta_i)^\mathsf{H}\right) \left(\mathbf{f}(r_j) \otimes \mathbf{a}(\theta_j)\right)\right| \\
&\overset{(b)}{=} \left|\left(\mathbf{f}(r_i)^\mathsf{H}\mathbf{f}(r_j)\right) \otimes \left(\mathbf{a}(\theta_i)^\mathsf{H}\mathbf{a}(\theta_j)\right)\right| \\
&= \underbrace{\left|\left(\mathbf{f}(r_i)^\mathsf{H}\mathbf{f}(r_j)\right)\right|}_{\bar{f}(r_i, r_j)} \times \underbrace{\left|\left(\mathbf{a}(\theta_i)^\mathsf{H}\mathbf{a}(\theta_j)\right)\right|}_{\bar{a}(\theta_i, \theta_j)},
\end{aligned} \tag{6.8}$$

where (a) and (b) are because of the transpose property and the mixed-product property of the Kronecker product respectively. This leads to a nice expression of $s^\star(\mathbf{h}_i, \mathbf{h}_j)$ as the product of two terms: $\bar{f}(r_i, r_j)$ and $\bar{a}(\theta_i, \theta_j)$. This decomposition in terms of radial and angular component explains why weak identifiability is studied here.

Even if such distance measure is phase insensitive, it is still subject to ambiguities that prohibit the establishment of unique local neighborhoods. Indeed, when moving away from a given reference point in an arbitrary direction, the value of the distance oscillates, meaning that points on that line appear successively farther then closer then farther again with varying amplitudes, creating aliases. In particular, these ambiguities arise from two phenomena: a long-range one and a short-range one. Combined, they create false neighborhoods that affect the trustworthiness of the measure. On one hand, the long-range ambiguities are caused by the periodic nature of the function, meaning that local neighborhoods are duplicated indefinitely in a periodic manner. A simple solution to go around this problem is to only consider users in a area smaller or equal to one period of the function. System parameters could consequently be adjusted to obtain an area of the desired

size. One the other hand, the short-range ambiguities are caused by the oscillatory nature of the function inside one period. Fortunately, the amplitudes of the oscillations decrease monotonically, meaning that applying a threshold to only keep the main lobe is feasible and would allow the elimination of the remaining aliases. These considerations are made mathematically rigorous in the next section assuming a ULA for the antenna array.

6.2 Limitations of channel charting with a ULA

In the classical case of a base station equipped with a ULA, it is possible to explicit further the terms of the decomposition (6.8). This allows to highlight the limitations of ULAs for channel charting. Note that UPAs could almost similarly be analyzed as an extension to the 3D case but are not considered here.

Before proceeding, a review of some classical results is necessary as proposed in the following lemmas.

Lemma 6.1

In an OFDM system with evenly spaced subcarriers, the radial term is of the form

$$\bar{f}(r_i, r_j) = \left| D_{N_s}\left(\frac{2\pi B(r_i - r_j)}{c}\right)\right|, \tag{6.9}$$

where D_{N_s} is the Dirichlet kernel defined as $D_N(x) = \frac{1}{N}\frac{\sin(\frac{x}{2})}{\sin(\frac{x}{2N})}$. It is periodic of period $D_{\bar{f}} = \frac{c}{\Delta_f}$ provided that the absolute delays of both channel observations (i.e., τ_i and τ_j) are known. Additionally, it contains a main lobe of width $L_{\bar{f}} = \frac{2c}{B}$ centered at $r_i = r_j$.

Proof. See Appendix A.1. □

Lemma 6.2

With a ULA, the angular term is of the form

$$\bar{a}(\theta_i, \theta_j) = \left| D_{N_R}(2\pi\Delta_r N_R(\Theta_i - \Theta_j)) \right|, \tag{6.10}$$

where $\Theta = \sin\theta$. It is of period $D_{\bar{a}} = \frac{1}{\Delta_r}$ w.r.t. Θ. It contains a main lobe centered at $\Theta_i = \Theta_j$ of width $\frac{2}{\Delta_r N_R}$. In addition, it is symmetrical w.r.t. $\theta = \frac{\pi}{2}\ [\pi]$.

Proof. See Appendix A.2. □

When taken w.r.t. θ, the $\sin$ in (6.10) makes the periodic pattern of varying width in the intervals $[-\frac{\pi}{2}, \frac{\pi}{2}] + \pi\mathbb{Z}$. When the pattern is of width π, it covers the whole interval, leading to the following corollary:

Corollary 6.1

If the ULA antennas are separated by half the wavelength (i.e., $\Delta_r = \frac{1}{2}$), $\bar{a}$ contains a single main lobe $\forall\theta_i, \theta_j \in [-\frac{\pi}{2}, \frac{\pi}{2}]$. For a given $\theta_i \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, the main lobe extends from $\arcsin(\sin\theta_i - \frac{1}{N_R\Delta_r})$ to $\arcsin(\sin\theta_i + \frac{1}{N_R\Delta_r})$ when $-1 + \frac{1}{N_R\Delta_r} \leq \sin(\theta_i) \leq 1 - \frac{1}{N_R\Delta_r}$, otherwise it is split in two. For $\theta_i = 0$, its width is equal to $L_a = 2\arcsin(\frac{1}{N_R\Delta_r})$ (in rad).

Proof. The proof follows from lemma 6.2 where the $\arcsin$ is applied to the width of the main lobe to convert it to the angular domain. □

In the case where the main lobe is split in two:

1. if $\sin(\theta_i) \leq -1 + \frac{1}{N_R\Delta_r}$, one part extends over the span of $[-\frac{\pi}{2}, \arcsin(\sin\theta_i + \frac{1}{N_R\Delta_r})]$ and the other over the span of $[\arcsin(\sin(\frac{\pi}{2} - \theta_i) - \frac{1}{N_R\Delta_r}), \frac{\pi}{2}]$;
2. if $1 - \frac{1}{N_R\Delta_r} \leq \sin(\theta_i)$, one part extends over the span of $[-\frac{\pi}{2}, \arcsin(\sin(-\frac{\pi}{2} + \theta_i) + \frac{1}{N_R\Delta_r})]$ and the other over the span of $[\arcsin(\sin\theta_i - \frac{1}{N_R\Delta_r}), \frac{\pi}{2}]$.

In the remainder of this chapter, ULAs with half a wavelength separated antennas are considered.

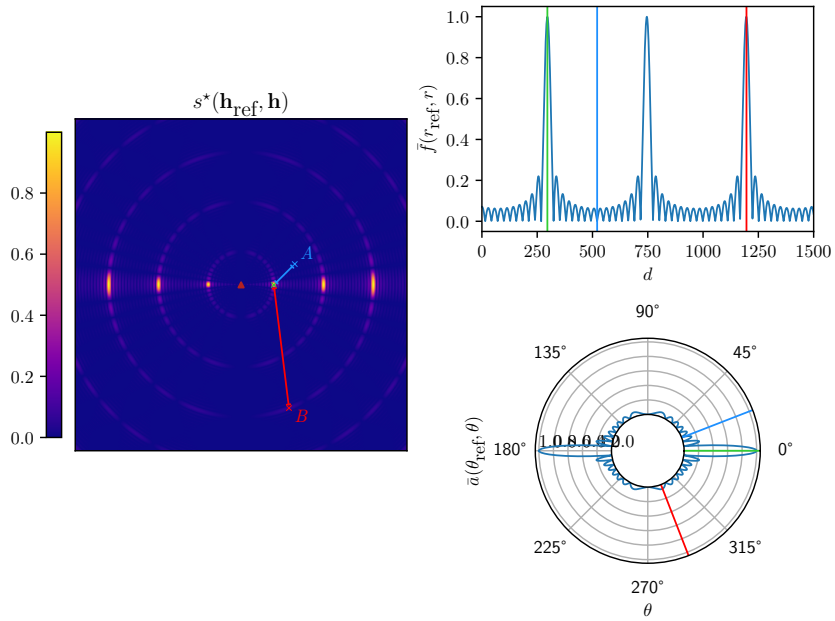


Figure 6.1 – The left figure shows a plot of the similarity $s^\star(\mathbf{h}_{\text{ref}}, \mathbf{h})$ between the reference user (green cross and lines) and the rest of the points on the map. The BS is placed at its center. The top right figure shows a plot of $\bar{f}(r_{\text{ref}}, r)$ and the bottom right one shows a plot of $\bar{a}(\theta_{\text{ref}}, \theta)$. User A (blue cross and lines) is spatially closer to the reference user than user B (red cross/lines), but the distance measure makes the latter appear closer than the former.

Figure 6.1 shows a 2D color plot of (6.8) along with plots of $\bar{f}$ and $\bar{a}$ for some arbitrary system parameters and for a reference user with associated channel $\mathbf{h}_{\text{ref}}$ and position $(r_{\text{ref}}, \theta_{\text{ref}})$. It highlights the limitations of the current form of the distance measure that is, *from its point of view, 2 UEs far away from each other can appear closer due to its periodic and oscillatory nature*. Specifically, for a given reference UE, a UE *A* located on a local minimum of $s^\star$ will be considered farther than a UE *B* located on a local maximum even if *A* is closer than *B* in reality (i.e. according to the Euclidean distance between the UEs' locations).

To deal with the ambiguities mentioned above, system parameters should be chosen guided by the results of lemma 6.1 and corollary 6.1 with regards to definition 6.2.

Proposition 6.1: Necessary identifiability condition for ULA

A necessary condition for UEs in an area A to be weakly identifiable by $d^\star$, is:

1. the radial size R of A should satisfy $R \leq c(\frac{1}{\Delta_f} - \frac{1}{B})$,
2. for a given $\theta \in [\arcsin(-\frac{1}{N_R}), \arcsin(\frac{1}{N_R})]$, the angular spread of A should remain in the interval $[\arcsin(\sin(\theta) + \frac{N_R-1}{N_R}), \arcsin(\sin(\theta) - \frac{N_R-1}{N_R})]$,

Proof. Considering that $s^\star$ is the product of two terms parametrized by orthogonal parameters, the area of identifiable channels is characterized by deriving ranges on these parameters independently. Essentially, this amounts to determining intervals on r_i, r_j and θ_i, θ_j where only the central main lobes of $\bar{f}$ and $\bar{a}$, respectively, remain.

Since the period of $\bar{f}$ is $D_{\bar{f}} = \frac{c}{\Delta \bar{f}}$, it is straightforward to conclude that for a given r representing the radial center of the area, r_i and r_j should remain in the range $[r-(\frac{D_{\bar{f}}}{2} - \frac{L_{\bar{f}}}{4}), r+(\frac{D_{\bar{f}}}{2} - \frac{L_{\bar{f}}}{4})]$. Half the width of the main lobes (i.e., $\frac{L_{\bar{f}}}{2}$) is subtracted at the extremities to leave them out of the area and avoid resulting ambiguities. This results in an area of radial width of $2 \times (\frac{D_{\bar{f}}}{2} - \frac{L_{\bar{f}}}{4}) = c(\frac{1}{\Delta_f} - \frac{1}{B})$.

Regarding $\bar{a}$, consider (6.10) where the period is $D_{\bar{a}} = \frac{1}{\Delta_r}$ w.r.t. Θ. For a given θ representing the angular center of the area where $sin(\theta) = \Theta$, Θ_i and Θ_j should remain in the range $[\Theta-(\frac{D_{\bar{a}}}{2} - \frac{L_{\bar{a}}}{4}), \Theta+(\frac{D_{\bar{a}}}{2} - \frac{L_{\bar{a}}}{4})] = [\Theta-\frac{N_R-1}{N_R}, \Theta+\frac{N_R-1}{N_R}]$ (with $\Delta_r = \frac{1}{2}$). When converted to the angular domain, the angular range is given by $[\arcsin(\sin(\theta) - \frac{N_R-1}{N_R}), \arcsin(\sin(\theta) + \frac{N_R-1}{N_R})]$. To respect the definition domain of $\arcsin$, θ should remain in the interval $[\arcsin(-\frac{1}{N_R}), \arcsin(\frac{1}{N_R})]$. □

Proposition 6.1 is illustrated in figure 6.2. Notice how the width of the area is composed of two terms: $\frac{c}{\Delta_f}$ and $\frac{c}{B}$. When the number of subcarriers N_s is high enough, the second one becomes negligible compared to the first, meaning that the radial size of the area is mainly controlled by the subcarrier spacing Δ_f. On the other hand, when the number of antennas is high enough, the angular range of the area tends to $[-\frac{\pi}{2}, \frac{\pi}{2}]$, meaning that it covers the whole area in front of the ULA. With this in mind, for UEs in an area of radial width

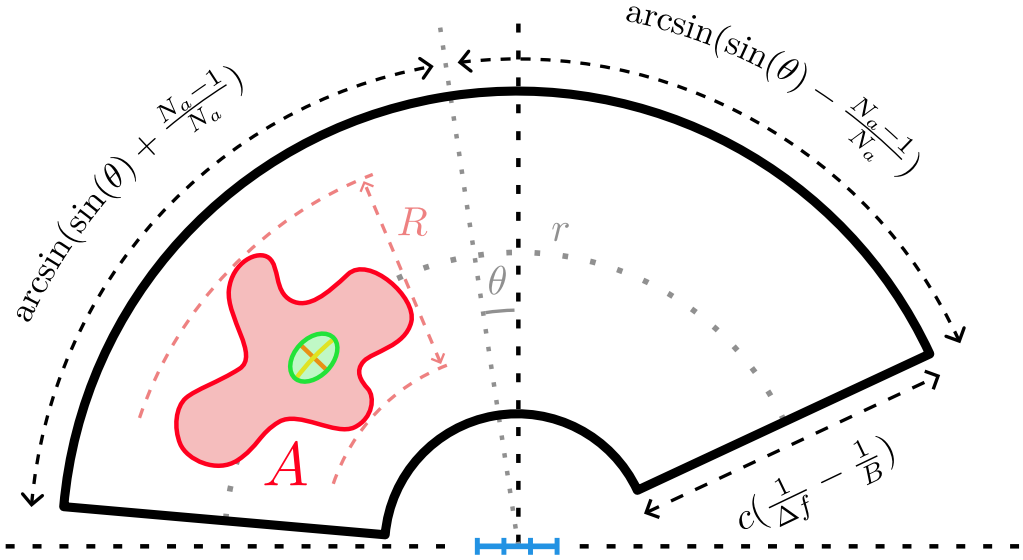


Figure 6.2 – The identifiable area's outline. Any region A of any shape included in there (e.g., the red region) verifies the necessary condition for identifiability. The green patch represents the identifiable neighborhood of the channel at its center. The orange segment is its radial axis and the yellow arc is its angular axis. The blue segment represents the ULA at the location of the BS. r is the radial center of the area and θ is its angular center.

R, the subcarrier spacing should be fixed so that $\Delta_f \lessapprox \frac{c}{R}$. Another important remark is that an identifiable area, by definition, should not cross the line of the ULA.

Proposition 6.1 is necessary because it is required that any channel set identifiable by $d^\star$ according to definition 6.2 is included in a well defined region controlled by system parameters. It is defined as the smallest region where a single main lobe appears wherever the reference user is taken, effectively eliminating long-range ambiguities. However, it is not sufficient because short-range ambiguities still subsist.

Proposition 6.2: Sufficient identifiability condition for ULA

A sufficient condition for UEs in an area A to be weakly identifiable by $d^\star$ is, *in addition to verifying the necessary identifiability condition*, for their similarity value $s^\star$ to be above or equal to a threshold $t_s \triangleq \max(|D_{N_s}(3\pi)|, |D_{N_R}(3\pi)|)$.

Proof. Given a reference channel $\mathbf{h}_{\text{ref}}$ in the identifiable area, to cut out the secondary lobes of $s^\star(\mathbf{h}_{\text{ref}}, .)$, only values above their maximum should be retained for the distance measure. Consequently, corresponding thresholds $t_{\bar{f}}$ and $t_{\bar{a}}$ are derived for the frequency and angular terms respectively. The thresholds are deduced from the amplitude of the second highest extremum (because of the absolute value) of the Dirichlet kernel. The first one is given as $t_{\bar{f}} \triangleq |D_{N_s}(3\pi)| = \frac{1}{N_s \sin(\frac{3\pi}{2N_s})}$. Similarly, the second threshold is given as $t_{\bar{a}} \triangleq |D_{N_R}(3\pi)| = \frac{1}{N_R \sin(\frac{3\pi}{2N_R})}$. Since $s^\star = \bar{f} \times \bar{a}$ and knowing that $0 \leq \bar{f}, \bar{a} \leq 1$, a sufficient condition for both terms to be above their thresholds is to impose $s^\star(\mathbf{h}_{\text{ref}}, .) \geq t_s \triangleq \max(t_{\bar{f}}, t_{\bar{a}})$. The set of channels verifying this condition, called the *identifiable neighborhood* of $\mathbf{h}_{\text{ref}}$, is a subset of the set of all identifiable channels.

For $\forall \vec{x}$ in the identifiable region and $\forall \vec{y}, \vec{z}$ in its identifiable neighborhood

$$d^\star(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x})) > d^\star(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x})) \tag{6.11}$$

$$\Leftrightarrow \quad |\bar{f}(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x}))| \times |\bar{a}(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x}))| > \tag{6.12}$$

$$|\bar{f}(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x}))| \times |\bar{a}(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x}))| \tag{6.13}$$

$$\Rightarrow \quad |\bar{f}(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x}))| > |\bar{f}(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x}))| \tag{6.14}$$

$$\text{or } |\bar{a}(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x}))| > |\bar{a}(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x}))| \tag{6.15}$$

Since the identifiable neighborhood corresponds to the main lobes of $\bar{f}$ and $\bar{a}$, this means that they are strictly monotonic in that interval w.r.t. their parameter r and θ respectively. As a consequence,

$$\begin{aligned} & d^\star(\mathbf{h}(\vec{y}), \mathbf{h}(\vec{x})) > d^\star(\mathbf{h}(\vec{z}), \mathbf{h}(\vec{x})) \\ \Rightarrow & |r_y - r_x| > |r_z - r_x| \text{ or } |\theta_y - \theta_x| > |\theta_z - \theta_x|. \end{aligned}$$

□

Each identifiable channel has its own identifiable neighborhood. In particular, the spatial region associated to it has a shape that resembles an ellipse with a radial linear axis and an angular curved one, corresponding to the radial and angular lobes respectively, as shown on figure 6.2. Consequently, any channel which location belongs to that region is in the identifiable neighborhood

of the reference channel. This results in a neighborhood graph that can be exploited by other methods to construct the low-dimensional channel chart. A consequence of the thresholding operation involved in the construction of this graph is that the main lobes' widths are changed. The new width of the main lobe of the frequency component, noted $L'_{\bar{f}}$, is the solution to the equation $\left|D_{N_s}(\frac{\pi B L'_{\bar{f}}}{c})\right| = t_s$. It can be computed through the inversion of D_{N_s} in the interval $[0, 2\pi]$ using classical root finding algorithms. A similar method can be used to derive the new width of the main lobe of the angular component, noted $L'_{\bar{a}}$ (in rad). It is given this time as the difference between the two roots of the equation $|D_{N_R}(2\pi\Delta_r N_R(\sin\theta_0 - \sin\theta))| = t_s$ in the interval $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Hence, the radial axis of an identifiable neighborhood is of length $L'_{\bar{f}}$ while its angular axis is of length $L'_{\bar{a}} \times r_{\text{ref}}$.

Although the necessary and sufficient conditions guarantee the weak identifiability of d^* in the case of a ULA, it can still suffer from the varying geometrical size of the identifiable neighborhoods. Indeed, the width of the main lobe of $\bar{a}(\theta_{\text{ref}}, .)$ is not constant and depends on θ_i. In addition, a ULA is limited to the half-space in front of it because of the symmetry at $\theta = \frac{\pi}{2}$ $[\pi]$ (the axis of the ULA) causing ambiguities. The next section dives into how using a UCA can help mitigate these limitations.

6.3 UCA are better than ULA for channel charting

In this section, the use of UCAs as a replacement for ULAs is explored. It is shown that UCAs are better for charting than ULAs because they do not exhibit axial symmetry and their angular resolution is constant (does not depend on the azimuth), which allows to derive a relation between the bandwidth and the radius of the UCA that limits the warping behavior of CC.

Lemma 6.3

With a UCA, a good approximation of $\bar{a}$ is of the form

$$\tilde{a}(\theta_i, \theta_j) = \left| J_0 \left(\frac{4\pi}{\lambda} R \left| \sin \frac{\theta_i - \theta_j}{2} \right| \right) \right|, \tag{6.16}$$

where J_0 is the Bessel integral of order 0. It contains a single main lobe in the interval $[-\pi, \pi]$ centered at $\theta_i = \theta_j$ of width $L_{\tilde{a}} = 4 \arcsin(\min(\frac{\lambda}{4\pi R} \times j_{0,1}, 1))$ (in rad) where $j_{0,1} = 2.4048$ is the first root of the Bessel integral of order 0 (Weisstein 2020).

Proof. See Appendix A.3. □

Based on this approximation, a new distance function is defined,

$$\tilde{d}^\star(\mathbf{h}_i, \mathbf{h}_j) \triangleq \sqrt{2 - 2\tilde{s}^\star(\mathbf{h}_i, \mathbf{h}_j)}, \tag{6.17}$$

where $\tilde{s}^\star(\mathbf{h}_i, \mathbf{h}_j) = \bar{f}(r_i, r_j) \times \tilde{a}(\theta_i, \theta_j)$. Figure 6.3 shows a 2D color plot of (6.8) along with plots of $\bar{f}$ and $\tilde{a}$ for some arbitrary system parameters and for a reference user with associated channel $\mathbf{h}_{\text{ref}}$ and location $(r_{\text{ref}}, \theta_{\text{ref}})$. Contrary to $s^\star$, this time only one angular main lobe appears.

Proposition 6.1′: Necessary identifiability condition for UCA

For UEs in an area A to be weakly identifiable by $\tilde{d}^\star$, its radial size R should satisfy $R \leq c(\frac{1}{\Delta_f} - \frac{1}{B})$.

Proof. The first part of the proof is the same as the one involving $\bar{f}$ in the proof of proposition 6.1.

Regarding $\tilde{a}$, the function contains a single main lobe in its domain of definition according to lemma 6.3 and is thus not subject to long-range ambiguities that can be caused by the period nature of a a function. □

Proposition 6.1′ is illustrated figure 6.4. It highlights the fact that with a

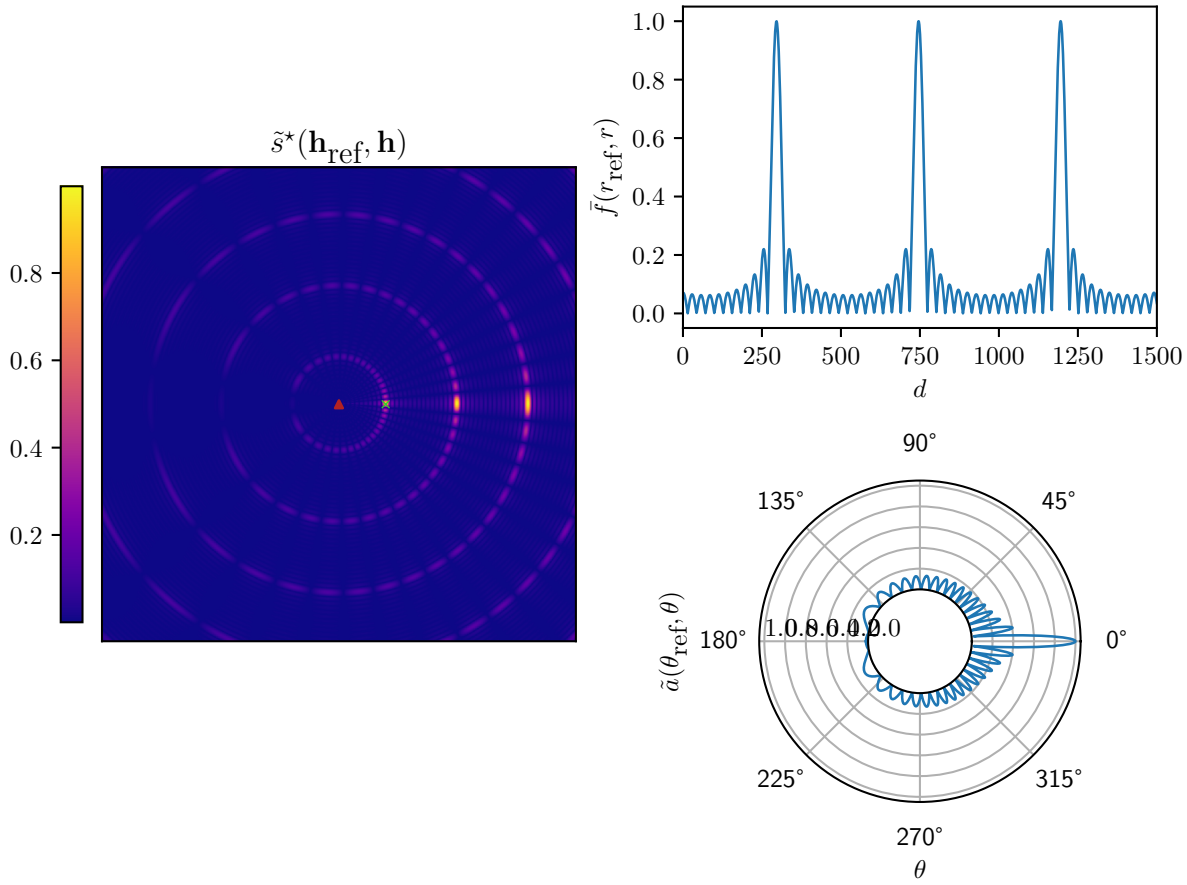


Figure 6.3 – The left figure shows a plot of the similarity $\tilde{s}^\star(\mathbf{h}_{\text{ref}}, \mathbf{h})$ between the reference user (green cross) and the rest of the points on the map. The BS is placed at its center. The top right figure shows a plot of $\bar{f}(r_{\text{ref}}, r)$ and the bottom right one shows a plot of $\tilde{a}(\theta_{\text{ref}}, \theta)$.

UCA, the identifiable area is no longer angularly restricted and is extended to include the whole angular domain (i.e., $[-\pi, \pi]$) with no ambiguities.

Proposition 6.2′: Sufficient identifiability condition for UCA

A sufficient condition for UEs in an area A to be weakly identifiable by $\tilde{d}^\star$ is, *in addition to verifying the necessary identifiability condition and for* $N_s > 2$, for their similarity value $\tilde{s}^\star$ to be above or equal to a threshold $t_{\tilde{s}} \triangleq 0.403$.

Proof. The proof follows the same logic as that of proposition 6.2 with the difference that this time the threshold of the angular term is changed and is derived from the amplitude of the second highest extremum of the Bessel integral. It is straightforwardly given as $t_{\tilde{a}} \triangleq |J_0(j'_{0,1})| = 0.403$ where $j'_{0,1} = 3.8317$ is the first root of the derivative of the Bessel integral of order 0 (Weisstein 2020). The global threshold is given as $t_{\tilde{s}} \triangleq \max(t_{\bar{f}}, t_{\tilde{a}})$. Finally, knowing that $t_{\bar{f}}$ is strictly decreasing with N_s, and that $t_{\bar{f}} < t_{\tilde{a}}$ for $N_s > 2$, it follows that $t_{\tilde{s}} = t_{\tilde{a}}$ for $N_s > 2$. □

This time again, thresholding has an impact on the widths of the main lobes. Since $t_{\tilde{s}}$ has now a constant value, a closed-form expression of $L'_{\bar{f}}$ can be derived through inversion of $\left|D_{N_s}(\frac{\pi B L'_{\bar{f}}}{c})\right| = t_{\tilde{s}}$. Although the root depends on N_s, its actual value varies negligibly with it and is thus derived for $N_s \to \infty$ as an acceptable approximation. As a result, the width of the radial main lobe after thresholding is given by $L'_{\bar{f}} = \frac{c \times 4.238}{\pi B}$. Similarly, the new width of $\tilde{a}$ (in rad), noted $L'_{\tilde{a}}$, is given as the solution to the equation $\left|J_0(\frac{4\pi}{\lambda} R \left|\sin \frac{L'_{\tilde{a}}}{4}\right|)\right| = t_{\tilde{s}}$. It can be computed through the inversion of J_0 in the interval $[0, j_{0,1}]$. It's value is given by $L'_{\tilde{a}} = 4 \arcsin(\frac{\lambda}{4\pi R} \times 1.692)$.

Proposition 6.3: Constant angular spread

When using the thresholded channel distance, the UCA (as opposed to the ULA) yields neighborhoods whose angular spread do not depend on the azimuth.

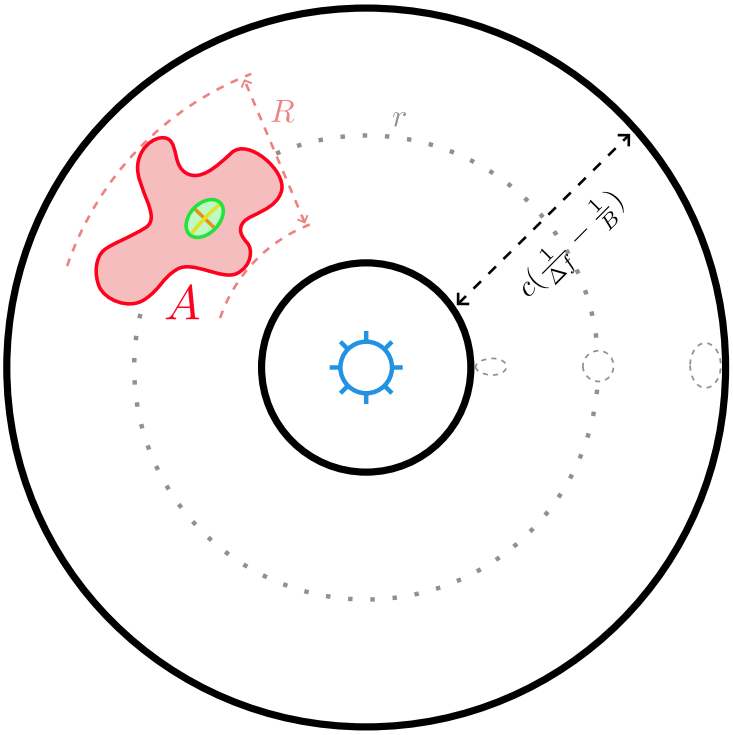


Figure 6.4 – The identifiable area's outline. Any region A of any shape included in there (e.g., the red region) verifies the necessary condition for identifiability. The green patch represents the identifiable neighborhood of the channel at its center. The orange segment is its radial axis and the yellow arc is its angular axis. The blue circle represents the UCA at the location of the BS. r is the radial center of the area.

Proof. The proof directly follow from the fact that $L'_{\tilde{a}}$, which represents the angular spread of the obtained neighborhoods, doesn't depend on θ. □

This is a very interesting property since it implies that the angular distance perceived by the system is independent of the azimuth, thus avoiding warping in the obtained channel chart, as illustrated in section 6.5.2. However, the size of the resulting identifiable neighborhoods are still not constant everywhere. To ensure that the distance reflects the true geometrical neighborhoods without distortion, the identifiable neighborhoods should be as round as possible. This is measured by the ratio of both their axis, i.e. $\gamma \triangleq \frac{L'_{\tilde{f}}}{L'_{\tilde{a}} \times r_{\text{ref}}} \approx \frac{c}{B r_{\text{ref}}} \frac{1.06}{\pi \arcsin\left(\frac{\lambda}{4\pi R}.1.692\right)}$.

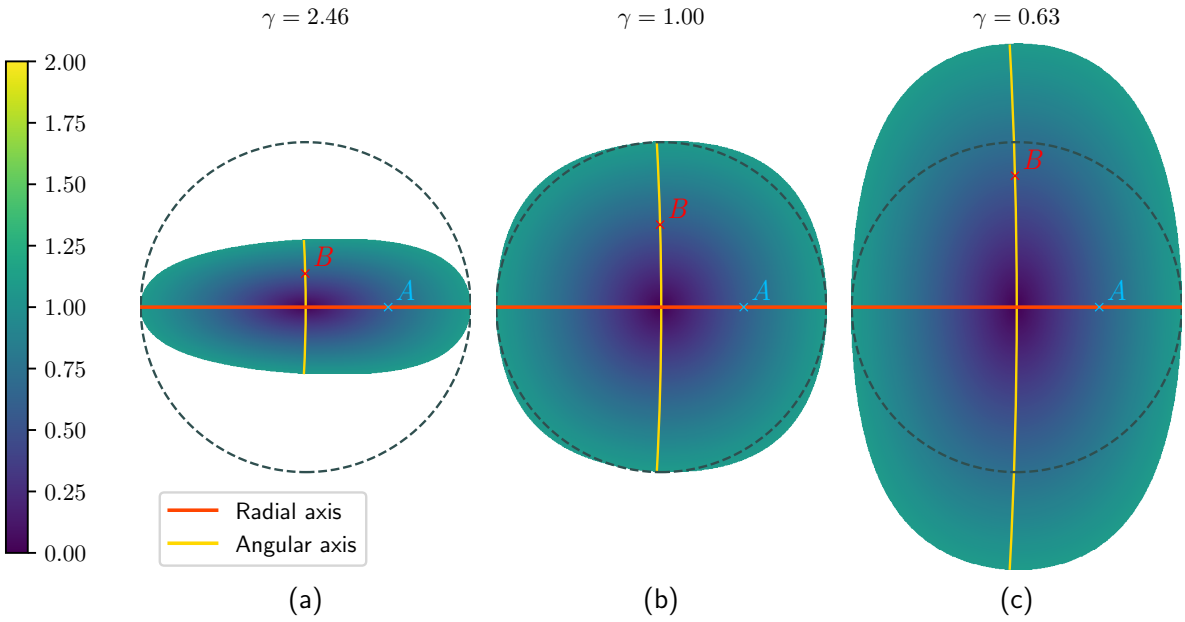


Figure 6.5 – Visualization of the distance measure for identifiable neighborhoods at (a) the inner, (b) center and (c) outer edge of the identifiable area as pictured in figure 6.4. The circle outline helps visualize how their shapes deviate from being perfectly round. UEs A and B are considered at the same distance from the point of view of the PI distance.

Proposition 6.4: Round neighborhoods

For a BS equipped with a UCA of radius R, a bandwidth B and an area considered for charting whose center is at a distance r_0 from the BS, the neighborhoods identified by the thresholded PI distance are round at the center of the area if and only if $1 = \gamma \approx \frac{c}{Br_0} \frac{1.06}{\pi \arcsin\left(\frac{\lambda}{4\pi R}.1.692\right)}$.

Proof. Since an identifiable neighborhood has the shape of an ellipse approximately, it is round when its axes' lengths, $L'_{\tilde{f}}$ and $L'_{\tilde{a}} \times r_0$, are equal. As a consequence, $\gamma = 1$. □

This result allows to determine the optimal bandwidth B as a function of the distance r_0 and the radius R as $B_{\text{opt}} = \frac{c}{r_0} \frac{1.06}{\pi \arcsin\left(\frac{\lambda}{4\pi R}.1.692\right)}$. It also allows to determine the optimal radius R as a function of the distance r_0 and the

bandwidth B as $R_{\text{opt}} = \frac{\lambda.1.692}{4\pi \sin\left(\frac{c}{Br_0}\frac{1.06}{\pi}\right)}$.

Figure 6.5 shows the shape of the identifiable neighborhoods when the reference user is at the inner edge of the identifiable area, at it's center and at it's outer edge, from left to right respectively. Because of distortion, users A and B will be considered at the same distance from the reference user as measured by (6.8), although it's only true for the center identifiable neighborhood. Furthermore, its area can be approximated with the area of the circle of diameter L'_f as $\frac{\pi L'^2_f}{4}$, as shown on the figure. Since for better performance a minimum number of neighboring users k_{min} should be located inside the area defined by the patch for each reference user, the minimum density of users is given as $\frac{4k_{\text{min}}}{\pi L'^2_{\tilde{f}}}$. In reality, this density should be a bit higher to account for the distortion of the neighborhood going towards the inner part of the identifiable area. In practice, the number of neighbors inside of the identifiable neighborhoods will vary for each reference user, but there should be at least one to connect it to the neighborhood graph constructed by Isomap.

6.4 Practical consequences of the theoretical findings

As a summary of the previous propositions, let's consider an area A of radial size R and center r_0 (figure 6.4). To be able to identify all users within the area without ambiguities, a MIMO OFDM system should have the following characteristics:

— To be equipped with a UCA of a given radius R_{UCA} (proposition 6.3);

— To have a bandwidth $B = \frac{c}{r_0}\frac{1.06}{\pi \arcsin\left(\frac{\lambda}{4\pi R_{\text{UCA}}}.1.692\right)}$ to ensure having the roundest neighborhoods possible (proposition 6.4);

— To have an subcarrier spacing $\Delta f \leq \frac{cB}{RB+c}$ (proposition 6.1′);

— The computed PI distance should be thresholded to only keep values below $\sqrt{2-2\times t_{\tilde{s}}} = 1.093$ (proposition 6.2′).

6.5 Experiments

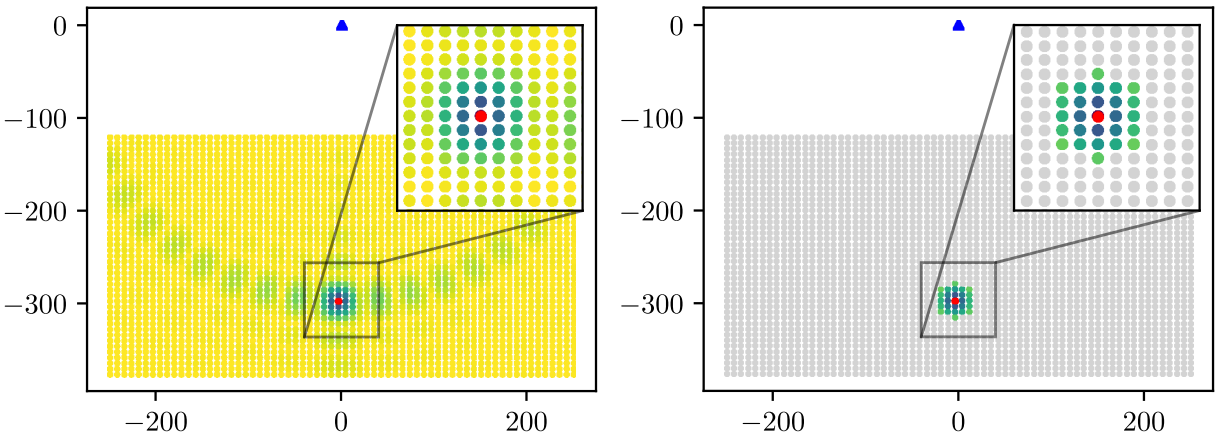
In this section, the theoretical claims and system setting guidelines of section 6.2 and section 6.3 are empirically validated both on synthetic and realistic channel data.

6.5.1 Practical influence of system parameters

A first simple experiment is considered, in which the effect of system parameters on the obtained channel chart is evidenced, showing the practical importance of the theoretical claims of the previous sections. The base scenario consists of a BS equipped with a UCA of 64 antennas and of radius $R = 42$ cm operating at $f_c = 3$ GHz. It transmits over a bandwidth of $B = 10$ MHz and $N_s = 16$ subcarriers. A total of $N = 2838$ channel observations of UEs located in front of the BS is generated according to (6.2). These parameters are used to compute and characterize the identifiable area, the identifiable neighborhoods as well as the optimal distance for the radial center r_0 according to the analysis of the previous sections. The identifiable area's width is of 422 m (proposition 6.1′) and its radial center should be placed at a distance of 296 m from the BS to limit neighborhood distortion (proposition 6.4). Figure 6.6 shows a visualization of the distance between the UE at the center of the area (i.e., $r_{\text{ref}} = r_0$) and the rest of the UEs before and after thresholding. The base scenario is tested against 3 variants:

1. not applying the threshold;
2. halved width of the identifiable area by doubling the subcarrier spacing Δ_f and keeping the same bandwidth so that some users end up outside of it;
3. non-round neighborhoods at the center by reducing the bandwidth B.

Figure 6.7 shows the learned charts for the base scenario as well as for each variant. The word "b<>com" helps visualize how the global structure of the locations is preserved on it. The chart obtained with the first variant looks disorganized. Although long-range ambiguities are eliminated, short-range ones remain as no threshold is applied, causing a degradation in performance in particular in terms of TW and KS. The chart obtained with the



(a) Before thresholding (b) After thresholding

Figure 6.6 – The computed PI distance between a reference UE (red point) and the rest of the UEs before (left) and after (right) thresholding. The greyed points represent the points that are outside the identifiable neighborhood and thus discarded.

second variant is warped as the parts of the considered area that are outside of the identifiable area are confused with other parts inside of it because of long-range ambiguities. As a consequence, TW and KS are negatively impacted. This shows the importance of limiting UEs to the identifiable zone. The chart obtained with the last variant is flattened as a consequence of the non-round identifiable neighborhoods, meaning that distances are perceived differently on the radial and angular directions. While there is little impact on TW and CT, KS is highly impacted because the global structure of the chart is not preserved. Finally, the chart obtained in the base scenario has the best overall performance and, although not perfect in terms of the global structure, represents a pretty faithful representation of the locations of the UEs.

6.5.2 ULA vs. UCA

Another experiment is considered, this time to further empirically assess the advantage of a UCA over a ULA, in order to validate the theoretical claims of section 6.3. A BS placed at the center of a map communicates with UEs uniformly distributed around it. The same system parameters as in the base scenario of the previous section are considered, with the difference that in one case a ULA of 16 antennas is considered, while in the other case a UCA of 64

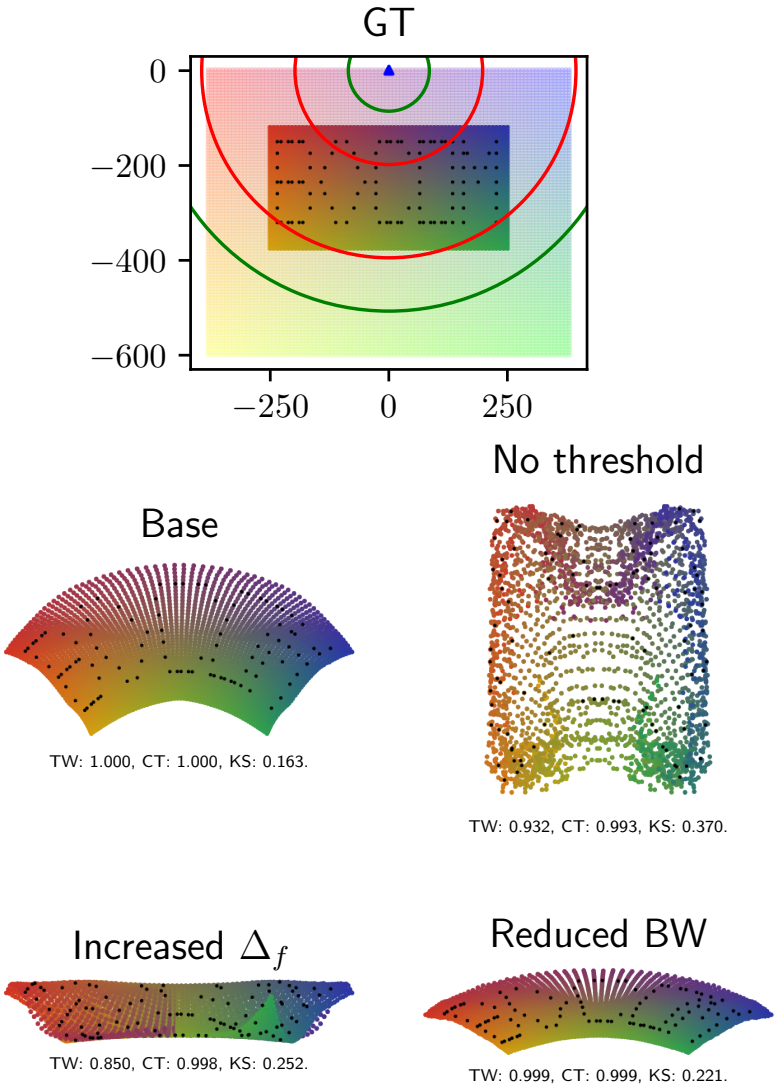


Figure 6.7 – The GT UEs' locations and the learned charts corresponding to the base scenario and its three variants.

antennas is considered. The difference in the number of antennas is to obtain lobes of comparable radial sizes in both cases. In addition, two areas that include the UEs' positions are considered: one that is identifiable according to the UCA and one that is identifiable according to the ULA also. The second one is included in the first one by definition. Note that the communication aspect of the system is ignored. Figure 6.8 shows the ground truth UEs' locations as well as the learned charts in each case. In the case of the red area, only points going from yellow to pink appear on the chart associated with the ULA, which represent locations on the right of the map. This is because they overlap with the locations on its left. The channel distance considers that points on the right coincide with points on the right due to its axial symmetry w.r.t. the vertical axis on the figure. Isomap then proceed to learn a chart that places the wrongly matched points at the same locations. On the other hand, the chart produced in the case of the UCA successfully reflects the circular structure of the locations. The TW score in particular being low in the case of the ULA indicates that a lot of false neighborhoods are introduced in the chart which is not the case in the case of the UCA, confirming the observations. In the case of the orange area, both charts appear to be of good quality. However, upon closer inspection, it appears that the chart associated with the UCA is of better quality as indicated by the lower (i.e., better) KS score. Indeed, the points appear to be distributed uniformly whereas in the case the ULA, their density seems to increase the more pink they get. This can be explained by the angular width of the identifiable neighborhoods being dependent on the azimuth in the case of the ULA.

6.5.3 Realistic scenario

In this experiment, the presented CC approach is tested on a more realistic scenario using the Sionna ray tracing engine (Hoydis, Ait Aoudia, et al. 2023) on the "etoile" scene, which is a 3D reconstruction of the area around the Arc De Triomphe in Paris. The BS is placed at the center of the scene and above the arc. It is equipped with a UCA of 64 elements and a radius of 20 cm. It communicates at a frequency of 3 GHz and over a bandwidth of 7 MHz. A total of $N = 3383$ UEs are distributed around it and chosen so that they meet the

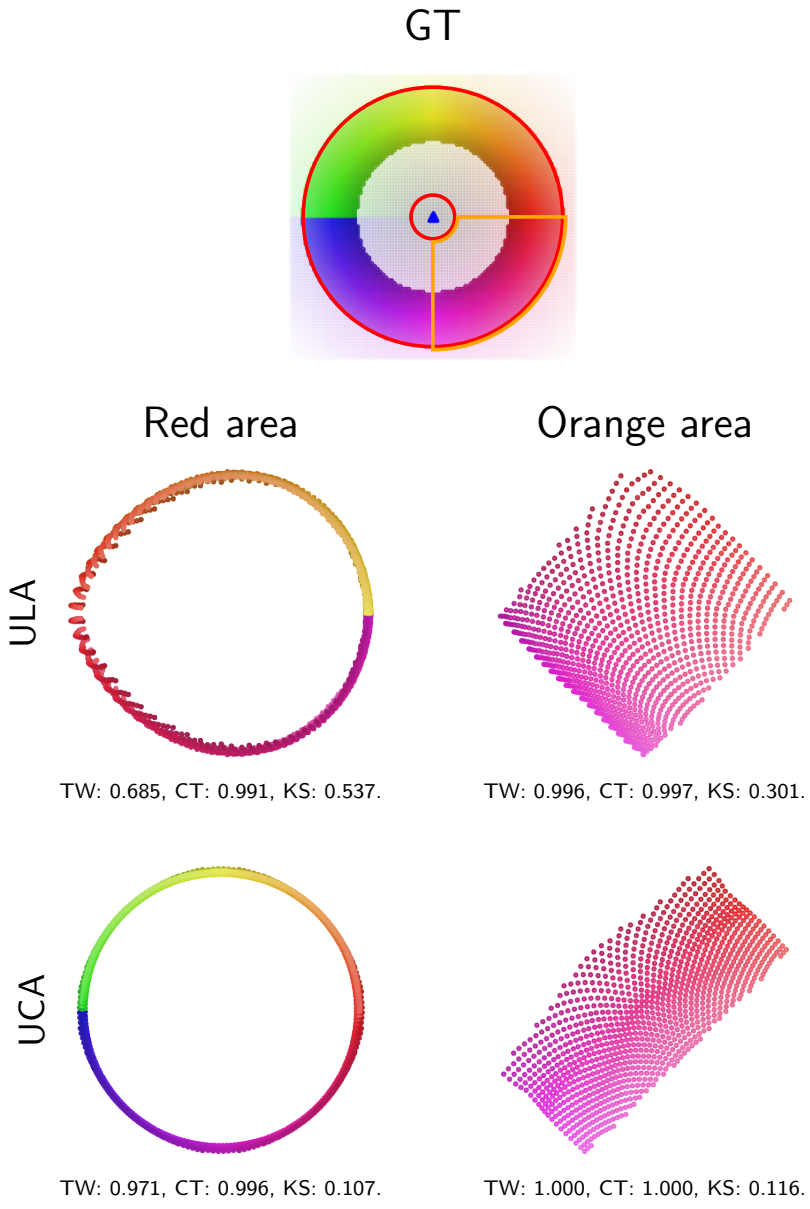


Figure 6.8 – The GT UEs' locations and the learned charts corresponding to the use of a ULA vs. a UCA for two different areas (red and orange).

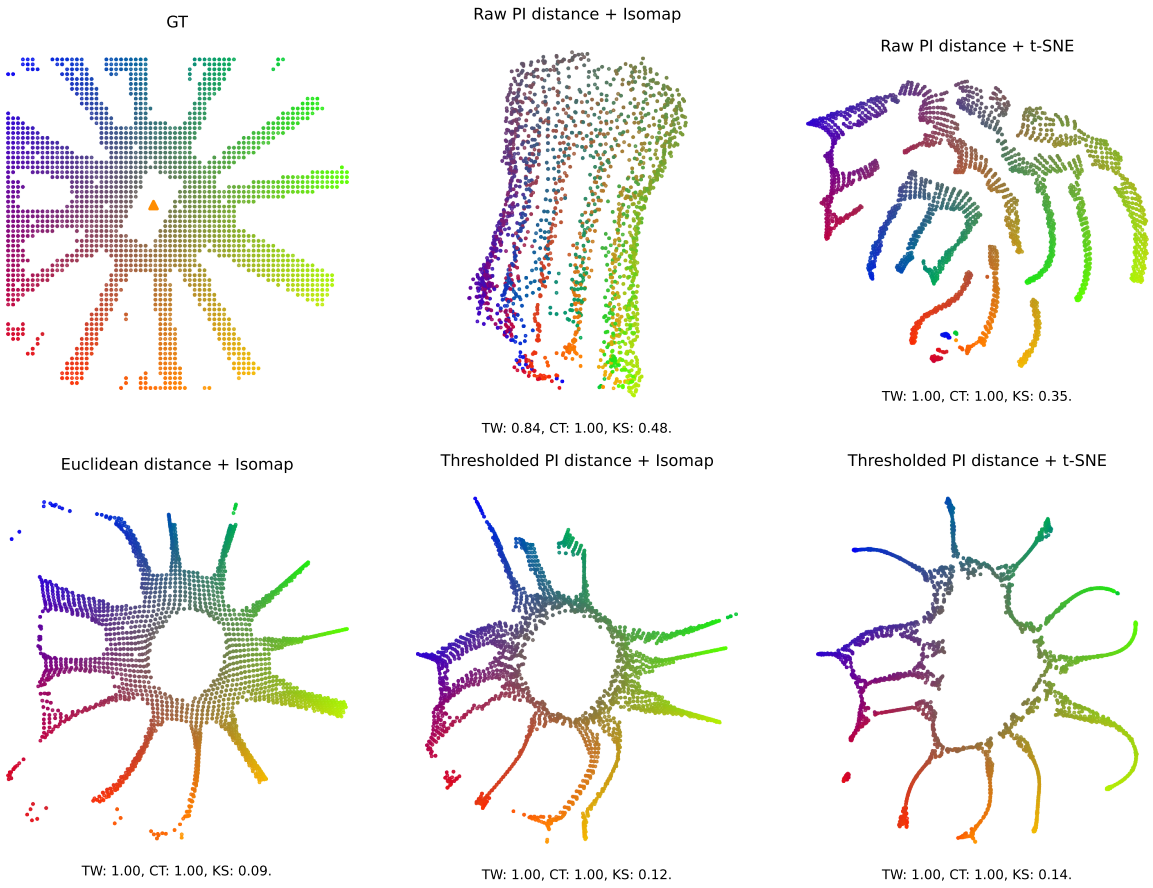


Figure 6.9 – The GT UEs' locations for the realistic scenario and the learned charts corresponding to the use of the Euclidean distance (gold standard), the raw PI distance and the proposed thresholded PI distance for both Isomap and t-SNE.

necessary identifiability condition of proposition 6.1′, effectively avoiding long-range ambiguities, and only the LoS path is considered. Note that although the theory developed in this chapter considers the BS and UEs to be on the same plane, this scenario however considers the more realistic case where the BS is at a height of 70 m while the UEs are 1.5 m above the ground. Figure 6.9 shows the ground truth (GT) locations and the charts generated by two dimensionality reduction methods, namely Isomap Tenenbaum, Silva, and Langford 2000 and t-SNE Maaten and G. Hinton 2008 using three different channel distances. First, the Euclidean distance is used to showcase the capabilities of Isomap with quasiperfect knowledge and to serve as a gold standard. In particular, a neighborhood graph is constructed where neighbors are limited to points present in a radius $\frac{L'_{\bar{f}}}{2}$ of each node. This amounts to restricting the knowledge to identifiable neighborhoods according to $\tilde{d}^{\star}$ as defined in proposition 6.2′. The resulting chart represents what can be achieved with a strongly identifiable distance (see definition 6.1), had one been available. The next chart is generated by constructing the neighborhood graph based on the raw PI channel distance, including its short-range ambiguities (i.e., not applying the threshold so that the sufficient identifiability condition of proposition 6.2′ is not met). The locations on the chart look warped and spatial neighborhoods are hardly distinguishable one from another when using Isomap, while t-SNE seems to behave better, although with neighborhoods that seem to be randomly layed out. The final charts are similarly generated by constructing the neighborhood graph based on the channel distance but this time applying the threshold (meeting the sufficient condition of proposition 6.2′) and effectively restricting neighborhoods to identifiable ones, which has the effect of eliminating the short-range ambiguities. This results in a much better chart that closely resemble the GT. As expected, thresholding allows to greatly improve TW and KS of the obtained chart, which is perfectly in line with the results of section 6.5.1. Note that for a fair comparison, t-SNE relies on the same geodesic distance as Isomap, obtained from a graph built from the thresholded PI distance. In this case however Isomap seems to slightly outperform t-SNE. Moreover,it is to be stressed that this paper proposes a new channel distance and not a new charting algorithm (the proposed distance can be used within any distance-based channel charting method), and we use here two charting

methods to show that the improvement brought by the proposed distance are not specific to a given charting algorithm.

6.6 Conclusion and perspectives

In this chapter, a theoretical framework for the identifiability of UEs from their channel measurements was introduced in the perspective of adequately setting the parameters of a MIMO OFDM system for channel charting purpose. To that end, the PI channel distance was analyzed, highlighting its short and long-range ambiguities. A practical method for mitigating these limitations and eliminating ambiguities was then proposed by translating the introduced identifiability rules into guidance in the design of the MIMO OFDM systems. The PI distance has been applied to CC on a MIMO OFDM system and the conducted experiments show a great improvement in the UEs' neighborhood consistency as long as the established designing rules are applied. In addition, the advantage of using UCAs instead of ULAs for CC was demonstrated and validated through simulations. All the achieved results show a very good concordance with the theory and make the proposed system design method very promising.

An interesting perspective for future work would be to test the proposed method on more realistic scenarios (e.g., multipath channels, edge UEs, etc.) and real-world measured data. In addition, it should be considered and extended in the context of ISAC systems with the goal of achieving a trade-off between communication and sensing.

As we have discussed in previous chapters, channel charting is not an end goal, but rather a preprocessing step that may benefit downstream tasks. Indeed, the hypothesis is that learning a low-dimensional representation of the channel that is closely related to the UEs locations should be easier than learning a mapping that goes directly from the space of channel vectors. This is tested in the context of beamforming in the next chapter.

Chapter 7

Channel charting for cell-free precoding

Location-based beamforming (LBB) (Kela et al. 2016) is a method for predicting precoders based on users' locations, thus suppressing the need for BSs to perform channel estimation or beam sweeping. LBB has recently been tackled successfully with supervised machine learning techniques (Le Magoarou, Yassine, et al. 2022b). However, the main drawback of LBB is that it requires having access to the precise locations, which are difficult to acquire within current systems. Since CC helps learn pseudo-locations that reflect the local geometrical neighborhoods, it could be used to relax the constraints of LBB, by replacing the needed locations which are difficult to estimate.

In this chapter, a method for channel charting-based beamforming is presented. First, the system setting is introduced. The objective is to take pseudo-locations from a learned chart as input for LBB. This allows a BS to choose appropriate precoders based on chart pseudo-locations instead of spatial locations. This alleviate the need for a precise estimation of user locations and opens the way to several applications such as channel mapping in space and frequency (Alrabeiah and Ahmed Alkhateeb 2019). The proposed method is described in terms of its training and how it can be used for inference. An interesting use case this method allows related to channel mapping is also discussed. Finally, it is empirically assessed on synthetic data and compared to the original LBB approach.

It is important to note that the LBB approach of (Le Magoarou, Yassine, et al. 2022b) can also be seen as an instance of model-based deep learning with the use of random Fourier features (RFFs) as we will see.

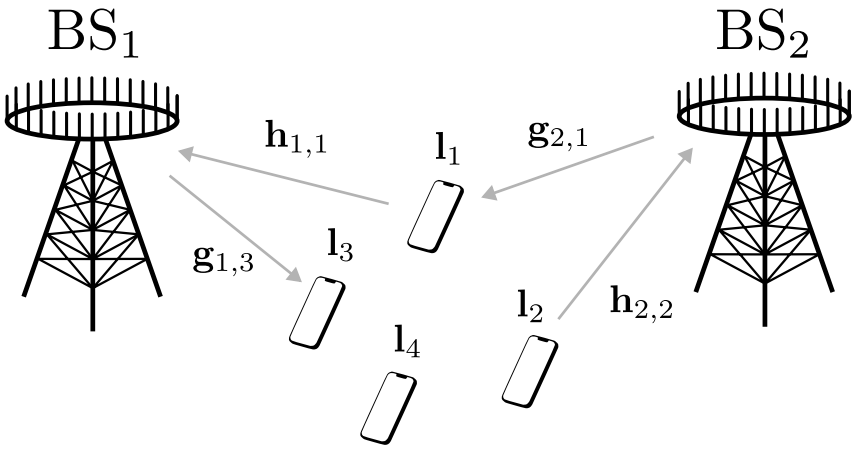


Figure 7.1 – Considered setting with $N_{\text{BS}} = 2$ and $N = 4$.

7.1 Problem formulation

Let us formulate the problem by describing the studied system and stating clearly the pursued objective.

7.1.1 Setting

A cell-free multi-user massive MIMO network is considered, in which N_{BS} BSs each comprising N_R antennas communicate over N_s subcarriers with N UEs each having a single antenna. The location of the jth UE is denoted $\mathbf{p}_j$, and the vectorized uplink (respectively downlink) channel between it and the ith base station is denoted $\mathbf{h}_{i,j} \in \mathbb{C}^{N_R N_s}$ (respectively $\mathbf{g}_{i,j} \in \mathbb{C}^{N_R N_s}$). Note that the uplink and downlink channels are not necessarily over the same band. The considered setting is shown on figure 7.1.

7.1.2 Objective

The goal of the proposed contribution is to map channels in space or frequency, which amounts to determine a downlink precoder $\mathbf{w}_{k,j}$ that should be highly correlated with the downlink (target) channel $\mathbf{g}_{k,j}$, based on the knowledge of the uplink (origin) channel $\mathbf{h}_{i,j}$. The origin and target channels are not necessarily on the same band nor even on the same base station (k and i may be different).

Performance measure In order to evaluate precoders, the normalized correlation between the precoder $\mathbf{w}_{k,j}$ and the target channel $\mathbf{g}_{k,j}$ is used. It is expressed as

$$\eta_{k,j} \triangleq \frac{|\mathbf{w}_{k,j}^{\mathsf{H}} \mathbf{g}_{k,j}|^2}{\|\mathbf{g}_{k,j}\|_2^2}. \tag{7.1}$$

It is between zero and one (for a perfect precoder), and is tightly linked to the downlink channel capacity (considering a single user), whose expression is

$$\log(1 + \eta_{k,j}.\mathrm{SNR}_{\mathrm{opt}}),$$

for received signal of the form $y = \sqrt{P_T}\mathbf{w}_{k,j}^{\mathsf{H}}\mathbf{g}_{k,j}s + n$ where $n \sim \mathcal{N}(0, \sigma^2)$ is additive noise, s is the sent symbol and P_T the transmit power. In that setting, $\mathrm{SNR}_{\mathrm{opt}} \triangleq \frac{P_T\|\mathbf{g}_{k,j}\|_2^2}{\sigma^2}$ is the highest achievable signal to noise ratio (SNR) (Björnson, Hoydis, Sanguinetti, et al. 2017). In summary, the correlation is a single number between zero and one allowing to determine the maximum achievable downlink spectral efficiency for any transmit power and noise variance considering a given precoder.

7.2 Proposed method

In this section, the inference and training phases of the proposed channel mapping method are described in details.

7.2.1 Inference

The proposed method comprises three steps:

1. At first, a pseudo-location corresponding to the vector $\mathbf{z}_{i,j} \in \mathbb{R}^D$ is computed from the origin channel $\mathbf{h}_{i,j}$ using channel charting methods (with $D \ll N_s N_R$) at the ith BS. In particular, the efficient channel charting method of (Le Magoarou 2021a) is used in this work, but any other CC method can be used.

2. Then, the pseudo-location $\mathbf{z}_{i,j}$, that can be seen as a compressed version of the channel is sent from the ith BS to the kth BS.

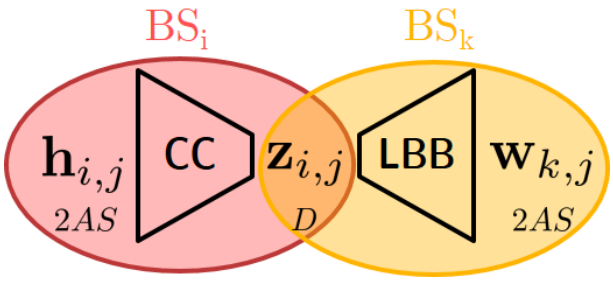


Figure 7.2 – Schematic view of the proposed method (the real dimension being shown below each variable).

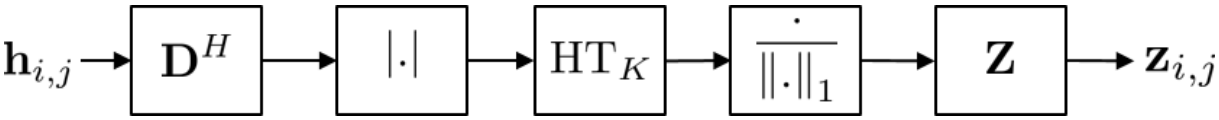


Figure 7.3 – Proposed similarity-based neural network for the CC step.

3. Finally, the kth BS performs location based beamforming using the pseudo-location as inputs, so as to output a precoder $\mathbf{w}_{k,j}$.

These three steps are depicted on figure 7.2. It is important to notice that the dimension is greatly reduced when performing channel charting, which limits the overhead induced by transmitting the latent variables from the ith to the kth BS.

7.2.2 Training

In order to calibrate the CC and LBB steps, a training phase is required. The hybrid encoder of (Yassine, Le Magoarou, et al. 2022) introduced in chapter 5 is used. First, the phase-insensitive distance d_{PI} between training channels $\{\mathbf{h}_{i,n}\}_{n=1}^{N}$ is first computed and then used within the Isomap to obtain pseudo-locations $\{\mathbf{z}_{i,n}\}_{n=1}^{N}$. The resulting matrices $\mathbf{D} \triangleq (\mathbf{h}_{i,1}, \ldots, \mathbf{h}_{i,N}) \in \mathbb{C}^{N_R N_s \times N}$ and $\mathbf{Z} \triangleq (\mathbf{z}_{i,1}, \ldots, \mathbf{z}_{i,N}) \in \mathbb{R}^{D \times N}$ are then used to form the feedforward neural network shown on figure 7.3 used at inference time to rapidly project channels vectors on the chart. Note that this time no neural training is performed and the encoder is used at initialization as is.

The LBB step is calibrated using the method proposed in (Le Magoarou, Yassine, et al. 2022b), except that the inputs are here pseudo-locations instead of true spatial locations. To do so, a database containing pseudo-

locations and associated target channel vectors has to be built (classical downlink channel estimation methods have to be used here): $\{\mathbf{z}_{i,n}, \mathbf{g}_{k,n}\}_{n=1}^{N}$. Note that the BS indices are different. This is because, as explained earlier, the BS producing the pseudo-locations can be different from the BS learning the precoder. For this, the pseudo-location is transmitter from the former to the latter. Then, a neural network comprising a random Fourier features (RFFs) layer is trained with a loss function expressed as

$$\mathcal{L}(\mathbf{w}_{k,n}, \mathbf{g}_{k,n}) \triangleq 1 - \frac{1}{N} \sum_{n=1}^{N} \frac{|\mathbf{w}_{k,n}^{\mathsf{H}} \mathbf{g}_{k,n}|^2}{\|\mathbf{g}_{k,n}\|_2^2}. \tag{7.2}$$

This loss measures the misalignment of the predicted precoders with respect to the training target channels and is between 0 (if the precoders are perfectly aligned with the target channels) and 1 (if the precoders are orthogonal to the target channels).

Highlight 7.1: Random Fourier Features

Classical neural networks are universal function approximators. In adequate settings, they are capable of learning complex functions provided data availability and quality. However, they are known for struggling when it comes to learning functions containing high frequencies. This phenomenon is known as the spectral bias (Rahaman et al. 2019; Basri et al. 2019). RFFs (Rahimi and Recht 2007; Tancik et al. 2020) have been shown to help eliminate it. Coordinate based neural networks whose objective is to learn a mapping starting from a very low-dimensional space particularly benefit from them. For example, RFF based neural networks have been applied successfully to 3D scene reconstruction (Mildenhall et al. 2021), where the network learns the mapping linking the 3D location and orientation to the radiance and density (which can be seen as the light channel). Figure 7.4 demonstrates how RFFs help with the spectral bias of NNs.

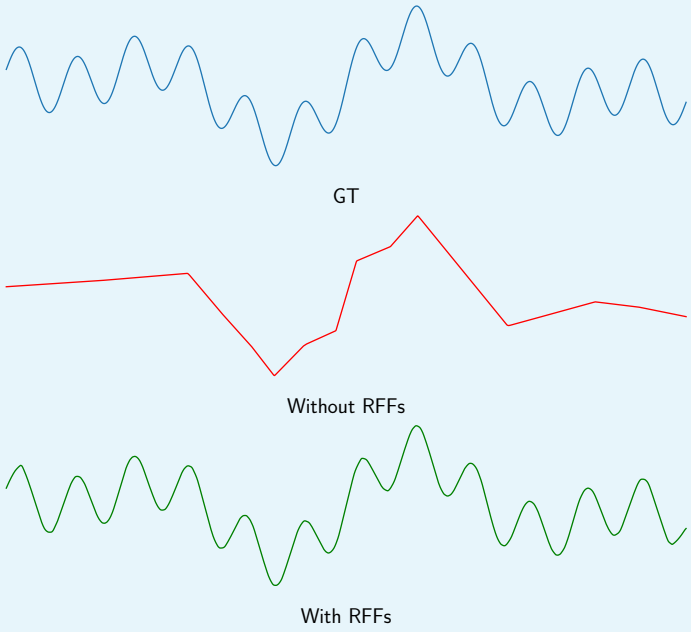


Figure 7.4 – Reconstruction of a signal containing high frequencies (top) by a normal NN (middle) and an equivalent NN with an RFFs layer (bottom).

In practice, they consist of an embedding layer f_{RFF} such that

$$f_{\text{RFF}}(\mathbf{z}) = (\cos(2\pi\mathbf{B}\mathbf{z}), \sin(2\pi\mathbf{B}\mathbf{z}))^\top, \tag{7.3}$$

where $\mathbf{z} \in \mathbb{R}^D$ is the input and $\mathbf{B} \in \mathbb{R}^{R\times D}$ is a matrix containing R D-dimensional (spatial) frequencies drawn randomly from $\mathcal{N}(0, s^2\mathbf{I})$ where s^2 controls the frequency range.

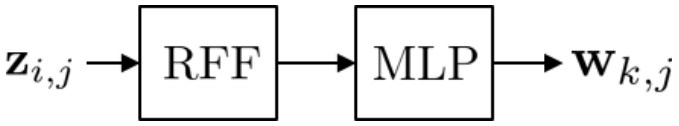


Figure 7.5 – Proposed RFF-based neural network for the LBB step.

7.2.3 Potential applications

The proposed method can be applied to several tasks, depending on the source and target channels. For example, if CC and LBB are done at the

same base station ($i = k$) but the frequency bands are different for the uplink (source) and downlink (target) channels, then channel mapping in frequency is carried out. In the more general case where CC and LBB are done in two distinct base stations ($i \neq k$), the proposed method allows to determine a precoder at one BS relying on a channel estimated at another BS (potentially in another frequency band), so that no channel estimation is required at the precoding BS. This is particularly interesting if the CC done at one BS is used to perform LBB at several other BSs (say $N_{\text{BS}} - 1$), since in that case channel estimation would be needed at only one BS to determine precoders at N_{BS} BSs. The overhead due to channel estimation would then be reduced by a factor of N_{BS}. Moreover, the backhaul requirements are kept reasonable, since only D real numbers (dimension of the channel chart which is much smaller than the dimension of the channel) have to be shared for each user.

7.3 Experiments

To evaluate the proposed approach, a set of experiments is conducted on synthetic data and the results are analyzed in this section.

7.3.1 Simulation settings

Multipath channel vectors are generated using the DeepMIMO dataset (A. Alkhateeb 2019). The urban outdoor scenario ('O1') (c.f. (A. Alkhateeb 2019, Figure 2)) is considered. Two base stations (i.e., $N_{\text{BS}} = 2$) on opposite sides of a street communicate with $N = 10\,000$ UEs spread across the map. The first BS corresponds to BS 16 in the original scenario and the second one corresponds to BS 15. Both BSs are equipped with $N_R = 8 \times 8$ UPAs. The first one operates at $3.5\,\text{GHz}$ while the second one operates at $28\,\text{GHz}$ and both transmit over $N_s = 16$ subcarriers spanning a bandwidth of $20\,\text{MHz}$. Note that for simplicity, the reported precoding results are for a single subcarrier (the central one). All in all, the obtained dataset allows to have access to a triplet $\{\mathbf{h}_{1,j}, \mathbf{g}_{2,j}, \mathbf{p}_j\}$ for each UE (uplink channel to the first BS, downlink channel from the second BS, user location).

7.3.2 Implementation details

The PyTorch library (Paszke et al. 2019) is used to implement and conduct the experiments, and implicitly handles the intricacies of neural network training (gradient computation, backpropagation, etc.). Complex vectors are converted to their real representation by stacking the real and imaginary parts resulting in vectors of twice the original size. The dataset is split into a training set and a test set. The training set size is set at $N_{\text{train}} = \%70 \times N = 7000$.

Channel charting The channel charting step can be performed in one of two manners: in a *one-shot* (to form a baseline) or *on the fly* (as proposed). The one-shot variant is obtained by applying Isomap to the distance matrix of all of the N channels. Only then is the chart split into the training and test sets. On the other hand, the on the fly variant is obtained by first computing matrices $\mathbf{D}$ and $\mathbf{Z}$, restricted to the training set of size N_{train}, that serve to initialize the NN. The test set is then obtained by computing the chart for the rest of the dataset using the latter. The neighborhood for Isomap is of size 5 for both variants.

The on-the-fly setting has the obvious advantage of being independent from training once performed, but is generally less robust than its one-shot counterpart.

Location based beamforming The input is of size $D = 5$ (i.e., dimension of the learned chart)[1], unless otherwise indicated. The number of RFFs is set to 600. The MLP consists of 4 layers. Each layer but the last one contains 300 neurons followed by a ReLU non-linearity. The model is trained for 30 epochs using the Adam optimizer (Kingma and Welling 2014) with a batch size of 100. Finally, the model outputs a vector $\mathbf{w} \in \mathbb{R}^{2N_R}$. For simplicity and to limit the complexity of the model in these initial experiments, the task is restricted to the prediction of the precoder corresponding to the central subcarrier only. Extension to the prediction of the full precoder is trivial but likely requires increasing the number of parameters of the NN.

1. Although the dimension of the chart was always the same as that of the UEs' locations, nothing prohibits it from being different.

7.3.3 Compared variants

Five variants of the model are considered for comparison.

— LBB is performed at the first BS using the real UEs' locations. Performance is evaluated using the uplink channels. This serves as a **baseline**.

— CC (one-shot) and LBB are both performed at the 1st BS and chained together as described in 7.2. The uplink channels are fed as input for training and used to evaluate the performance.

— CC (one-shot) is performed first at the 1st BS on the uplink channels. The learned chart is then transmitted to the 2nd BS for the LBB step on the downlink channels. This variant corresponds to the channel mapping setting.

— This variant is similar to the second one, except this time the learned chart is of dimension $D = 3$. The chart is computed in a one-shot.

— This variant is similar to the third one, except this time CC is computed on the fly. This last variant corresponds to the most realistic setting that is described in section 7.2.2.

7.3.4 Results

The five variants of the model are evaluated on the test set. Figure 7.6 shows the cumulative distribution function (CDF) of the correlations η—the closer to a Dirac at $x = 1$ the better. All methods perform relatively well and achieve a satisfactory level of performance. Interestingly, the CC based LBB variant performs better than the classical one, meaning that the 5 dimensions of the learned chart allow to convey more information than the 3-dimensional spatial locations. This hypothesis is consolidated when comparing classical LBB to the CC based one when keeping only 3 dimensions to the chart where we observe a small decline in performance. Furthermore, the figure shows that in the channel mapping setting, the model performs slightly worse but is on par with the classical one. Finally, the CC based LBB model where the chart is computed on the fly performs the worst, which is expected and highlights the influence of the quality of the learned chart on the subsequent tasks. It is worth noting that in this last case, an end-to-end training of the network

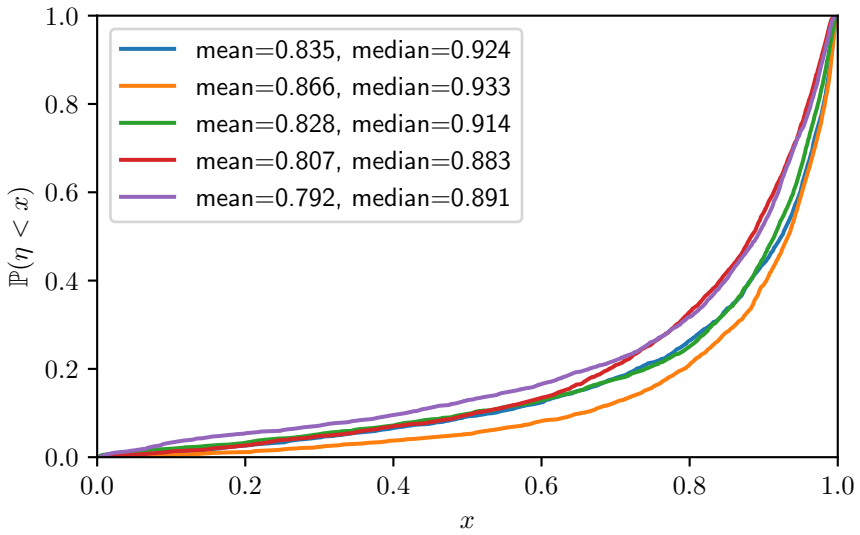


Figure 7.6 – CDF of the correlations. Blue curve corresponds to the original LBB at BS1. Orange curve corresponds to CC (one-shot) and LBB both at BS1 at the same frequency. Green curve corresponds to CC (one-shot) at BS1 and LBB at BS2 at different frequencies. Red curve corresponds to CC and LBB both at BS1 at the same frequency with a chart of dimensionality 3. Purple curve corresponds to CC (on the fly) at BS1 and LBB at BS2 at different frequencies.

(i.e., training both the CC and the LBB models) would certainly improve the performance.

Figure 7.8 shows the spatial distribution of the correlations for the last approach described in 7.3.3. The model succeeds in learning an adequate precoder for most of the spatial locations for both LoS and NLoS channels. Some areas show a high concentration in blue dots (top left and bottom left of the map) corresponding to precoders of low quality and could be explained by locations where propagation conditions with respect to the two BSs are very different (very weak channel between the user and one of the BSs).

7.4 Conclusion and outlook

In this chapter, a way to directly predict precoders from channel chart pseudo-locations was proposed. It amounts to combining wireless channel charting and location-based beamforming. It has the potential of greatly re-

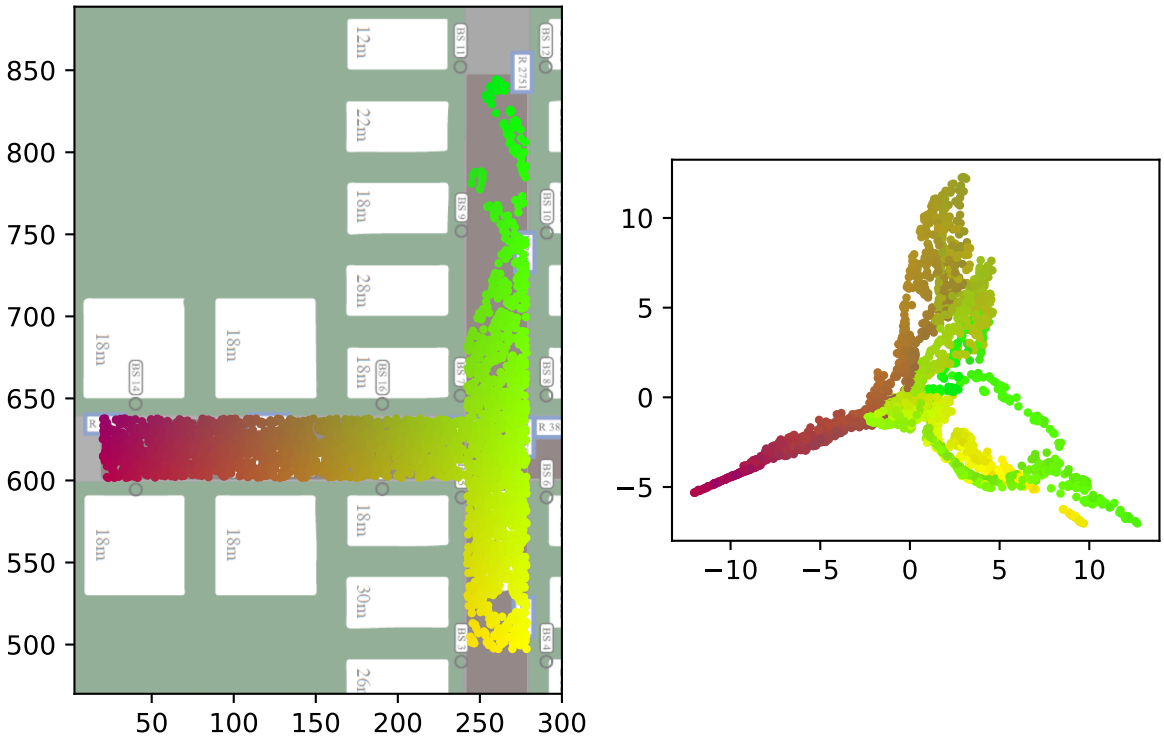


Figure 7.7 – Spatial distribution of the UEs in the test set (left) and their corresponding learned chart obtained on the fly (right). Color is added to distinguish local neighborhoods.

ducing the overhead due to channel estimation in cell-free systems, since CC can be carried out in a BS, and LBB in another one (or several others) that does not need to perform channel estimation nor beam sweeping. The amount of data to transfer between the two BS is reduced since the channel chart is of very low dimension. The proposed method was empirically assessed on realistic synthetic urban channels, showing the great potential of the approach, since it was found that the chart information (in dimension five) was as good as the real location information.

To continue on this line of work, it would be interesting to train the whole system end-to-end (CC and LBB), which raises the question of how to handle the transmission of gradients and parameters between base stations (decen-

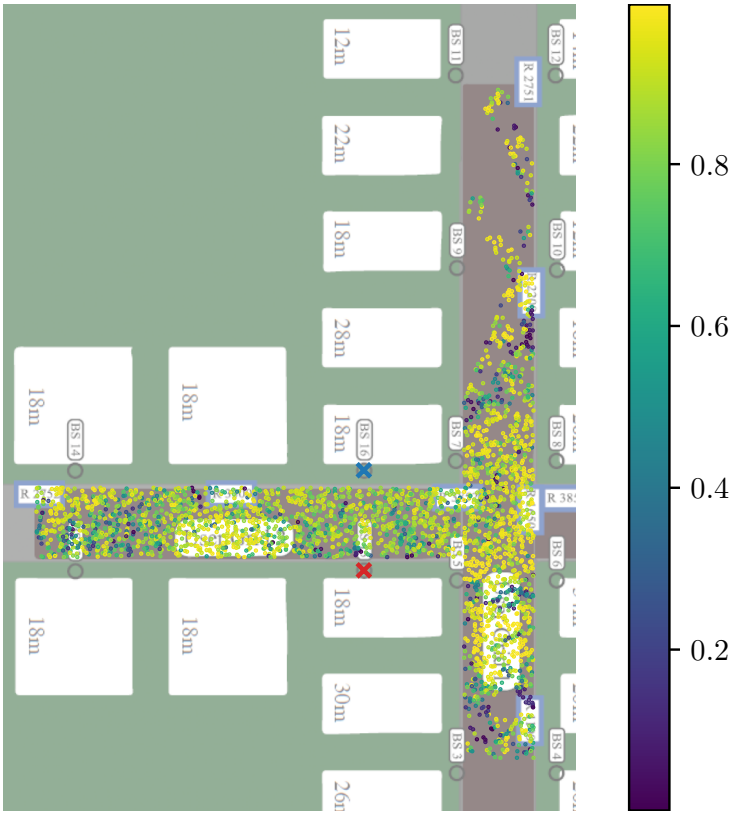


Figure 7.8 – Spatial distribution of the correlations. The blue and red crosses correspond to the 1st and 2nd BS respectively.

tralized federated learning). Moreover, a theoretical motivation based on the channel manifold in order to set hyperparameters in a principled way would be valuable. Finally, a similar approach could be envisioned for beam prediction within a finite set (as studied in (Ponnada et al. 2021)), since the problem can then be seen as a discrete version of the problem studied here.

In this final contributions' chapter, we explored an example of how channel charting can indeed be useful for beamforming. In our other work (Yassine, Chatelier, et al. 2023), we demonstrate the potential of a similar approach for beam prediction, a problem closely related to the one discussed here.

Conclusion and Perspectives

In this final chapter, let's take a step back and look at the big picture of the work accomplished and presented in this manuscript, and how the different chapters relate to each other. We will also go over some of the perspectives and how this work can be extended.

7.5 Conclusion

First, a brief overview of each chapter is given. Part I introduces the different notions and tools necessary to understand each contribution. In chapter 1, we saw how electromagnetic wave propagation, which is at the basis of modern wireless communication systems, can be modeled mathematically. Different physical phenomena at play (e.g., fading, multipath propagation, etc.) can be integrated into the model at will depending on the need for accuracy or simplicity. Models that try to be realistic pay the price in complexity and tend to require non-trivial considerations. The concept of propagation channels was also introduced. It allows to write the (MIMO) system model as an input/output relationship between the input signal $\mathbf{X}$ and the output $\mathbf{Y}$:

$$\mathbf{Y} = \mathbf{H}\mathbf{X} + \mathbf{N},$$

where $\mathbf{N}$ represents noise. All the mathematical modeling of the propagation environment is summarized in $\mathbf{H}$. To simulate a propagation scenario, two approaches prevail: deterministic models and statistical ones. We gave examples of both. Finally, in a communication context, estimating the channel is usually a critical step. We presented some of the classical estimation algorithms. These have historically proven to be robust, but as the dimension of the channel grows and the propagation environments evolve into more com-

plex configurations, the estimation task becomes challenging.

Since handcrafting channel models becomes tedious, and sometimes even unrealistic, relying on data-driven approaches gained popularity among the community. A growing interest in machine learning in communications was observed. Chapter 2 serves as an introduction to the matter. The general principle is to learn a function of interest by minimizing a given loss on the training set. These different components can be implemented in many ways as demonstrated in the given examples. One way in particular stands out: deep learning, whose seen a great amount of success in the recent years. At the moment of writing this manuscript, the world has its eyes on the not-so-old large language models like ChatGPT and Gemini for their impressive human mimicking capabilities. Like any deep learning application, the key to their success was the huge amount of available data and computing resources. Machine learning algorithms try to learn the underlying relationships that structure the data. We've seen that the manifold hypothesis, for example, suggests that natural data can usually be explained by a smaller number of variables. Dimensionality reduction methods aim to leverage this hypothesis to extract a compressed representation of the data, ultimately distilling their most salient features. The final part of the chapter explored the used of deep learning for channel estimation.

Chapter 3, where the concept of channel charting was introduced, is at the crossroads of the first two chapters. Although it is not immediately evident, learning a chart is ultimately a way of extracting the most important features from a propagation channel. In this context, the manifold hypothesis suggests that, in a (quasi-)static environment, the channel is governed by a few parameters pertaining to the TX location, i.e., $\mathbf{z} \approx \mathbf{p}$. Since these parameters are usually of very low dimension, it makes sense to try extracting them using the dimensionality reduction methods of chapter 2. Among a plethora of approaches explored in the literature, those based on distance measures in the space of channel vectors are of particular interest. Indeed, a number of manifold learning algorithms are based on the distance matrix of the inputs. Finding an adequate distance measure is thus of critical importance.

Part II goes into the details of each contribution. In chapter 4, mpNet is presented as a first contribution aiming at exploring the potential of a model-

based deep learning approach to improve the task of channel estimation under the sparsity assumption. As a model-based neural network, it is trained to learn a mapping from a received signal to the associated channel vector. In particular, the easy-to-compute LS estimate is used as input. This way, mpNet can be seen as a cheap denoising model. Indeed, compared to other generic networks, the model is designed to be computationally light. This is thanks to its structure that is obtained through the deep unfolding of the MP algorithm. mpNet is initialized from the initial imperfect knowledge of system parameters and is particularly efficient in learning to correct itself. For example, we showed that it can adapt overtime to anomalies like antenna damage or aging.

Another model-based network is presented in chapter 5, this time in the context of channel charting. It is based on the PI distance of (Le Magoarou 2021a) and takes the form of an encoder with a hybrid structure. We have seen that this model benefits from three main ideas: a smart initialization, the computation of the distance implicitly as part of the network's computations, and training in triplet mode. The latter exploits the fact that channel observations are usually accompanied by timestamps and that two samples close in time have a high probability of being close in space as well. We have shown how implementing each idea individually improves the quality of the learned chart.

We noticed that the PI distance has some limitations. In chapter 6, we identify and study these limitations in the context of MIMO OFDM systems. We have seen that they originate from the periodic and oscillatory nature of the distance. The notion of identifiability is proposed and describes a distance measure that can be reliably used to learn a chart free from ambiguities. Based on this definition, conditions are derived on system parameters to create a configuration where the PI distance can be used reliably. When put to practice, these conditions significantly improve the quality of the final chart.

In chapter 7, channel charting is considered from the application perspective. Once a chart is available, what can it be used for? Beamforming is taken as an example. In (Le Magoarou, Yassine, et al. 2022b), an NN is used to learn the optimal precoder corresponding to an input location. The same approach is tested when replacing real locations by pseudo-locations on a

learned chart. We show that the learned precoders are of good quality, even in NLoS conditions. Channel mapping, is also considered. Indeed, we also show that projecting a channel observation at a given BS first and then transmitting the pseudo-location to a different BS for precoder computation works well. This can be seen as a way of compressing information and only transmitting a distilled version of the channel to target BSs, removing the need for redundant channel estimation.

We have seen that mpNet can be seen as an autoencoder (highlight 4.2). This means that the encoder projects the channel vectors into a low-dimensional space. This space is however unstructured and the dynamic nature of the network doesn't help. Since the goal is to produce a channel estimate, the form of the latent space didn't matter. However, cell-leone can suppose that, as a general rule, having a well-behaved smooth space would help in the decoding process, regardless of the subsequent application. This is the premise of channel charting. Knowing that the channel is a direct product of the TX location, it is most probable that the low-dimensional space we're looking for is related to it, as per the manifold hypothesis. Channel charting aims at enforcing this a priori knowledge into the encoding. All in all, instead of trying to learn a mapping that goes directly from the space of channel vectors to a desired output, it is better to first go through a latent space in the form of a channel chart, especially when the quality of such a chart can be quantified and improved.

7.6 Perspectives

Since each chapter in part II is accompanied by a conclusion section going over the perspectives of each individual contribution, here we choose to give a broader view of how they can be extended in addition to some long-term perspectives.

Online learning Training an ML model to learn a task requires a training dataset. While the primary goal of unsupervised learning is to remove this requirement by finding ways of learning from the input data only, it also enables the model to continuously learn. Indeed, a system could reuse the data

it collects during its operation to readjust the model parameters to adapt to distribution shift. This was demonstrated in chapter 4. This principle can be extended to most applications of unsupervised learning. While the idea of training on new data is straightforward, it is not clear how to best implement it. Should the training be performed as soon as new samples are collected? Or maybe is it better to periodically re-train the model? Additionally, how should old data be handled? Should the retraining be limited to new data only? Or maybe is it better to keep the dataset at a fixed size and consider a first in first out (FIFO) model? All these questions are worth investigating. Beyond traditional gradient-based training, hybrid models like the one introduced in chapter 5 benefit from an initialization based on data available at the creation of the model. In particular, the samples that go into matrix D were shown to have a significant impact on the performance of the model. With time passing, the environment changes and newly collected samples better reflect that change. It would be interesting to investigate if gradually replacing elements of the matrix with new ones would result in a better adaptation of the system. A recent paper (Taner, Guillaud, et al. 2023) gives some initial insight on this matter.

End-to-end training As stated before, the task of learning a channel chart on its own doesn't serve much purpose. It is when paired with a downstream application that benefits from the chart become valuable. In the example we've seen where channel charting was applied to beamforming, the model is learned in two steps. First, the channel charting model is trained, then the beamforming decoder is trained on the obtained pseudo-locations. While this approach works well, it could be interesting to try end-to-end training. This could be done in one of two ways: either by doing it for the whole training phase, or by considering it as part of a subsequent fine-tuning step. The latter seems more adequate as it gives the charting model the opportunity to learn a latent space that fits the definition of a channel chart. This approach should also be tried on different downstream tasks to evaluate how their natures influence the performance of the encoder and its resulting chart.

CSI compression While the location dependent aspect of channel charting is important, it can be extended to a broader domain of application. As we have seen, learning a chart amounts to learning to reduce the dimensionality of the channel vectors. In other terms, it can be seen as a compression task. This can be useful for reducing the CSI feedback overhead in channel estimation (Yin et al. 2022) for example. Exploring the compression capabilities of the charting models and comparing them to pure compression approaches could bring forth some hidden insights that could improve both worlds. Maybe is there a tradeoff to be found between compressing the channel and keeping enough information for the subsequent task. Another possibility is that compressing through charting could allow for better levels of compression without degrading the performance on the final goal.

Inspiration from other fields Although machine learning and deep learning have been making headlines for some time now, one must admit that the wireless communications community is slow to embrace it. On the surface, this might seem like a disadvantageous situation, but in reality it is an incredible opportunity to draw inspiration from the work done in the other fields that have a lead. For example, the transformer architecture originally proposed for sequence to sequence translation (Vaswani et al. 2017) has been successfully applied to a number of different tasks, even outside of language modeling tasks. The attention mechanism is particularly interesting as it allows for different parts of the input, called tokens, to be compared to each others. This information is exploited by the neural network to efficiently decide what to output. Upon closer inspection, it appears that the hybrid encoder introduced in chapter 5 shares some similarities with the transformer. Indeed, the first step consisting of the computation of the correlation between an input channel vector $\mathbf{h}$ and the column of matrix $\mathbf{D}$ could be seen as the computation of the correlation between the keys and the queries of the transformer. Next, the hard thresholding is an alternative to the softmax operation and finally the weighted sum of the element of $\mathbf{Z}$ is similar to the weighted sum of the values. With this in mind, some other tricks of the transformer could be worth trying on the hybrid encoder. The first that comes to mind is to try having multiple heads with different matrices $\mathbf{D}$ and $\mathbf{Z}$ for each head. This is just one example

of how it can be beneficial to draw inspiration from what has been done in the other fields of machine learning. The literature is a gold mine of ideas to tackle challenging problems, and it is important to take advantage of it.

Part III

Appendices

APPENDIX A

CHAPTER 6'S APPENDIX

A.1 Proof of Lemma 6.1

The frequency term can be expressed as

$$\begin{aligned}
\bar{f}(r_i, r_j) &= \left|\left(\mathbf{f}(r_i)^{\mathsf{H}}\mathbf{f}(r_j)\right)\right| \\
&= \frac{1}{N_s}\left|\sum_{s=1}^{N_s} e^{-j2\pi\frac{r_i-r_j}{c}(f_s-f_c)}\right| \\
&\overset{(a)}{=} \frac{1}{N_s}\left|\sum_{s=1}^{N_s} e^{-j2\pi\frac{r_i-r_j}{c}\Delta_f(s-1)} e^{-j2\pi\frac{r_i-r_j}{c}(f_c-\Delta_f\frac{N_s-1}{2})}\right| \\
&= \frac{1}{N_s}\left|\sum_{s=0}^{N_s-1} e^{-j2\pi\frac{r_i-r_j}{c}\Delta_f s}\right| \\
&\overset{(b)}{=} \frac{1}{N_s}\left|\frac{\sin(\pi\frac{B(r_i-r_j)}{c})}{\sin(\pi\frac{B(r_i-r_j)}{cN_s})}\right| \\
&= \left|D_{N_s}\left(\frac{2\pi B(r_i-r_j)}{c}\right)\right|.
\end{aligned} \tag{A.1}$$

where (a) is obtained since $f_s = f_c - \Delta_f\frac{N_s-1}{2} + \Delta_f(s-1)$, and (b) comes from the computation of the geometric series sum.

A.2 Proof of Lemma 6.2

The angular term can be expressed as

$$\begin{aligned}\bar{a}(\theta_i,\theta_j) &= \left|\mathbf{a}(\theta_i)^{\mathsf{H}}\mathbf{a}(\theta_j)\right| \\ &= \frac{1}{N_R}\left|\sum_{n=0}^{N_R-1} e^{-j\frac{2\pi}{\lambda}\vec{a}_n(\vec{u}(\theta_i)-\vec{u}(\theta_j))}\right|.\end{aligned} \tag{A.2}$$

In particular, for a ULA and using the adequate steering vectors (6.4)

$$\begin{aligned}\bar{a}(\theta_i,\theta_j) &= \frac{1}{N_R}\left|\sum_{n=0}^{N_R-1} e^{-j2\pi\Delta_r n(\sin\theta_i-\sin\theta_j)}\right| \\ &= \frac{1}{N_R}\left|\frac{\sin(\pi\Delta_r N_R(\sin\theta_i-\sin\theta_j))}{\sin(\pi\Delta_r(\sin\theta_i-\sin\theta_j))}\right| \\ &= \left|D_{N_R}(2\pi\Delta_r N_R(\sin\theta_i-\sin\theta_j))\right|.\end{aligned} \tag{A.3}$$

A.3 Proof of Lemma 6.3

Considering a UCA and using the adequate steering vectors (6.4), $\bar{a}(\theta_i,\theta_j)$ becomes

$$\begin{aligned}\bar{a}(\theta_i,\theta_j) &= \left|\mathbf{a}(\theta_i)^{\mathsf{H}}\mathbf{a}(\theta_j)\right| \\ &= \frac{1}{N_R}\left|\sum_{i=1}^{N_R}\exp\Big(-j\frac{2\pi}{\lambda}R\right. \\ &\quad\Big(\cos(2\pi\frac{i}{N_R})(\cos\theta_i-\cos\theta_j) \\ &\quad\left.+\sin(2\pi\frac{i}{N_R})(\sin\theta_i-\sin\theta_j)\Big)\Big)\right|,\end{aligned} \tag{A.4}$$

for the discrete array case. Considering the continuous-aperture case by setting $N_R \to \infty$ leads to

$$\begin{aligned}\tilde{a}(\theta_i, \theta_j) = \frac{1}{2\pi} \Bigg| \int_0^{2\pi} \exp\Bigg(-j\frac{2\pi}{\lambda} R \Big(\cos\gamma(\cos\theta_i - \cos\theta_j) \\ + \sin\gamma(\sin\theta_i - \sin\theta_j) \Big) \Bigg) d\gamma \Bigg| \\ = \frac{1}{2\pi} \Bigg| \int_0^{2\pi} \exp\Bigg(-j\frac{2\pi}{\lambda} R \\ \Big(\cos(\gamma - \theta_i) - \cos(\gamma - \theta_j) \Big) \Bigg) d\gamma \Bigg|.\end{aligned}$$

Substituting ω for $\gamma - \theta_j$, and with $d\omega = d\gamma$ we have

$$\begin{aligned}= \frac{1}{2\pi} \Bigg| \int_{-\theta_j}^{2\pi-\theta_j} \exp\Bigg(-j\frac{2\pi}{\lambda} R \\ \Big(\cos(\omega + \theta_j - \theta_i) - \cos(\omega) \Big) \Bigg) d\omega \Bigg|,\end{aligned}$$

with $\Delta\theta = \theta_j - \theta_i$. Since $\cos$ is 2π periodic, we have

$$\begin{aligned}= \frac{1}{2\pi} \Bigg| \int_0^{2\pi} \exp\Bigg(-j\frac{2\pi}{\lambda} R \\ \Big(\cos(\omega + \theta_j - \theta_i) - \cos(\omega) \Big) \Bigg) d\omega \Bigg|.\end{aligned}$$

With the help of basic trigonometric identities, it can be shown that $\cos(\omega + \theta_j - \theta_i) - \cos(\omega) = 2\left|\sin\frac{\theta_j - \theta_i}{2}\right| \cos(\omega + \operatorname{atan2}(\sin(\theta_j - \theta_i), \cos(\theta_j - \theta_i) - 1))$, leading to

$$\begin{aligned}= \frac{1}{2\pi} \Bigg| \int_0^{2\pi} \exp\Bigg(-j\frac{4\pi}{\lambda} R \left|\sin\frac{\theta_j - \theta_i}{2}\right| \\ \cos\Big(\omega + \operatorname{atan2}(\sin(\theta_j - \theta_i), \cos(\theta_j - \theta_i) - 1)\Big) \Bigg) d\omega \Bigg|.\end{aligned}$$

Subsituting γ for $\omega + \text{atan2}(\sin(\theta_j - \theta_i), \cos(\theta_j - \theta_i) - 1) + \frac{\pi}{2}$, and with $d\gamma = d\omega$ we have

$$= \frac{1}{2\pi} \left| \int_{\text{atan2}(\sin(\theta_j-\theta_i),\cos(\theta_j-\theta_i)-1)+\frac{\pi}{2}}^{2\pi+\text{atan2}(\sin(\theta_j-\theta_i),\cos(\theta_j-\theta_i)-1)+\frac{\pi}{2}} e^{-j\frac{4\pi}{\lambda}R\left|\sin\frac{\theta_j-\theta_i}{2}\right|\sin\gamma} \, d\gamma \right|.$$

Finally, exploiting once again the 2π-periodic property of the sine, we have

$$= \frac{1}{2\pi} \left| \int_0^{2\pi} e^{-j\frac{4\pi}{\lambda}R\left|\sin\frac{\theta_j-\theta_i}{2}\right|\sin\gamma} \, d\gamma \right|.$$

This leads to a convenient way of expressing $\tilde{a}$ as it can be written as

$$\tilde{a}(\theta_i, \theta_j) = \left| J_0 \left(\frac{4\pi}{\lambda} R \left| \sin \frac{\theta_j - \theta_i}{2} \right| \right) \right|. \tag{A.5}$$

The validity of the integral form as an approximation was empirically verified and holds well when a great number of antennas N_R is considered.

BIBLIOGRAPHY

Aggarwal, Charu C., Alexander Hinneburg, and Daniel A. Keim (2001), « On the Surprising Behavior of Distance Metrics in High Dimensional Space », *in*: *Database Theory — ICDT 2001*, Springer Berlin Heidelberg, pp. 420–434.

Agostini, Patrick, Zoran Utkovski, and Sławomir Stańczak (2020), « Channel Charting: an Euclidean Distance Matrix Completion Perspective », *in*: Barcelona, Spain, Barcelona, Spain: IEEE, pp. 5010–5014.

Aharon, M., M. Elad, and A. Bruckstein (Nov. 2006), « K-SVD: An Algorithm for Designing Overcomplete Dictionaries for Sparse Representation », *in*: *Signal Processing, IEEE Transactions on* 54.*11*, pp. 4311–4322.

Akdeniz, Mustafa Riza et al. (2014), « Millimeter wave channel modeling and cellular capacity evaluation », *in*: *IEEE Journal on Selected Areas in Communications* 32.*6*, pp. 1164–1179.

Ali, Samad et al. (2020), « 6G white paper on machine learning in wireless communication networks », *in*: *arXiv preprint arXiv:2004.13875*.

Alkhateeb, A. (Feb. 2019), « DeepMIMO: A Generic Deep Learning Dataset for Millimeter Wave and Massive MIMO Applications », *in*: *Proc. of Information Theory and Applications Workshop (ITA)*, San Diego, CA, pp. 1–8.

Alrabeiah, Muhammad and Ahmed Alkhateeb (2019), « Deep learning for TDD and FDD massive MIMO: Mapping channels in space and frequency », *in*: *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, IEEE, pp. 1465–1470.

Bajwa, W. U. et al. (June 2010), « Compressed Channel Sensing: A New Approach to Estimating Sparse Multipath Channels », *in*: *Proceedings of the IEEE* 98.*6*, pp. 1058–1076.

Balatsoukas-Stimming, Alexios and Christoph Studer (Oct. 2019), « Deep Unfolding for Communications Systems: A Survey and Some New Directions »,

in: *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*, IEEE.

Balevi, Eren, Akash S. Doshi, and Jeffrey G. Andrews (Mar. 2020), « Massive MIMO Channel Estimation With an Untrained Deep Neural Network », *in*: *IEEE Transactions on Wireless Communications* 19.*3*, pp. 2079–2090.

Basri, Ronen et al. (2019), « The Convergence Rate of Neural Networks for Learned Functions of Different Frequencies », *in*: *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, Red Hook, NY, USA: Curran Associates Inc.

Bengio, Yoshua et al. (2003), « Out-of-Sample Extensions for LLE, Isomap, MDS, Eigenmaps, and Spectral Clustering », *in*: *Advances in Neural Information Processing Systems*, ed. by S. Thrun, L. Saul, and B. Schölkopf, vol. 16, MIT Press.

Biguesh, Mehrzad and Alex B Gershman (2006), « Training-based MIMO channel estimation: a study of estimator tradeoffs and optimal training signals », *in*: *IEEE transactions on signal processing* 54.*3*, pp. 884–893.

Björnson, Emil, Jakob Hoydis, Luca Sanguinetti, et al. (2017), « Massive MIMO networks: Spectral, energy, and hardware efficiency », *in*: *Foundations and Trends® in Signal Processing* 11.*3-4*, pp. 154–655.

Bottou, Léon (1998), « Online learning and stochastic approximations », *in*: *On-line learning in neural networks* 17.*9*, p. 142.

— (2010), « Large-scale machine learning with stochastic gradient descent », *in*: *Proceedings of COMPSTAT'2010*, Springer, pp. 177–186.

Box, George EP (1976), « Science and statistics », *in*: *Journal of the American Statistical Association* 71.*356*, pp. 791–799.

Bromley, Jane et al. (1993), « Signature Verification using a "Siamese" Time Delay Neural Network », *in*: *Advances in Neural Information Processing Systems*, ed. by J. Cowan, G. Tesauro, and J. Alspector, vol. 6, Morgan-Kaufmann.

Cai, T Tony and Lie Wang (2011), « Orthogonal matching pursuit for sparse signal recovery with noise », *in*: *IEEE Transactions on Information theory* 57.*7*, pp. 4680–4688.

Carlsson, Gunnar (2009), « Topology and data », *in*: *Bulletin of the American Mathematical Society* 46.*2*, pp. 255–308.

Chicco, Davide (Aug. 2020), « Siamese Neural Networks: An Overview », *in*: *Artificial Neural Networks*, Springer US, pp. 73–94.

Choi, Jun Won et al. (2017), « Compressed Sensing for Wireless Communications: Useful Tips and Tricks », *in*: *IEEE Communications Surveys & Tutorials* 19.*3*, pp. 1527–1550.

Cotter, S.F. and B.D. Rao (Mar. 2002), « Sparse channel estimation via matching pursuit with application to equalization », *in*: *IEEE Transactions on Communications* 50.*3*, pp. 374–377.

Cybenko, G. (Dec. 1989), « Approximation by superpositions of a sigmoidal function », *in*: *Mathematics of Control, Signals, and Systems* 2.*4*, pp. 303–314.

Das, Aniruddha and Bhaskar D. Rao (2012), « SNR and Noise Variance Estimation for MIMO Systems », *in*: *IEEE Transactions on Signal Processing* 60 (8), pp. 3929–3941.

Daubechies, Ingrid, Michel Defrise, and Christine De Mol (2004), « An iterative thresholding algorithm for linear inverse problems with a sparsity constraint », *in*: *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences* 57.*11*, pp. 1413–1457.

Decurninge, Alexis et al. (2018), « CSI-based Outdoor Localization for Massive MIMO: Experiments with a Learning Approach », *in*: *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, pp. 1–6.

Deng, Junquan et al. (2018), « Multipoint Channel Charting for Wireless Networks », *in*: *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, pp. 286–290.

Ding, Yacong and Bhaskar D Rao (2018), « Dictionary learning-based sparse channel representation and estimation for FDD massive MIMO systems », *in*: *IEEE Transactions on Wireless Communications* 17.*8*, pp. 5437–5451.

Donoho, D.L. (Apr. 2006), « Compressed sensing », *in*: *IEEE Transactions on Information Theory* 52.*4*, pp. 1289–1306.

Donoho, David L, Arian Maleki, and Andrea Montanari (2009), « Message-passing algorithms for compressed sensing », *in*: *Proceedings of the National Academy of Sciences* 106.*45*, pp. 18914–18919.

Dubey, Shiv Ram, Satish Kumar Singh, and Bidyut Baran Chaudhuri (Sept. 2022), « Activation functions in deep learning: A comprehensive survey and benchmark », *in*: *Neurocomputing* 503, pp. 92–108.

Duchi, John, Elad Hazan, and Yoram Singer (2011), « Adaptive Subgradient Methods for Online Learning and Stochastic Optimization », *in*: *Journal of Machine Learning Research* 12.*61*, pp. 2121–2159.

Dwivedi, Satyam et al. (2021), « Positioning in 5G Networks », *in*: *IEEE Communications Magazine* 59 (11), pp. 38–44.

Farhadi, Hamed et al. (Apr. 2023), *Hexa-X Deliverable D4.3 - AI-driven communication & computation co-design solutions*.

Ferrand, Paul, Alexis Decurninge, and Maxime Guillaud (2020), « DNN-based Localization from Channel Estimates: Feature Design and Experimental Results », *in*: *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, pp. 1–6.

Ferrand, Paul, Alexis Decurninge, Luis G. Ordoñez, et al. (2021), « Triplet-Based Wireless Channel Charting: Architecture and Experiments », *in*: *IEEE J. Sel. Areas Commun.* 39 (8), pp. 2361–2373.

Ferrand, Paul, Maxime Guillaud, et al. (Apr. 2023), « Wireless Channel Charting: Theory, Practice, and Applications », *in*.

Gao, Xuanxuan et al. (2018), « ComNet: Combination of deep learning and expert knowledge in OFDM receivers », *in*: *IEEE Communications Letters* 22.*12*, pp. 2627–2630.

Glorot, Xavier and Yoshua Bengio (2010), « Understanding the difficulty of training deep feedforward neural networks », *in*: *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, pp. 249–256.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016), *Deep Learning*, MIT Press.

Gregor, Karol and Yann LeCun (2010), « Learning fast approximations of sparse coding », *in*: *Proceedings of the 27th International Conference on International Conference on Machine Learning*, Omnipress, pp. 399–406.

He, Hengtao, Shi Jin, et al. (2019), « Model-driven deep learning for physical layer communications », *in*: *IEEE Wireless Communications* 26.*5*, pp. 77–83.

He, Hengtao, Chao-Kai Wen, et al. (2018a), « A model-driven deep learning network for MIMO detection », *in*: *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, IEEE, pp. 584–588.

— (2018b), « Deep learning-based channel estimation for beamspace mmWave massive MIMO systems », *in*: *IEEE Wireless Communications Letters* 7.5, pp. 852–855.

Hershey, John R, Jonathan Le Roux, and Felix Weninger (2014), « Deep unfolding: Model-based inspiration of novel deep architectures », *in*: *arXiv preprint arXiv:1409.2574*.

Hornik, Kurt, Maxwell Stinchcombe, and Halbert White (1989), « Multilayer feedforward networks are universal approximators », *in*: *Neural Networks* 2.5, pp. 359–366.

Hoydis, Jakob, Fayçal Ait Aoudia, et al. (Mar. 2023), « Sionna RT: Differentiable Ray Tracing for Radio Propagation Modeling », *in*: *arXiv preprint*.

Hoydis, Jakob, Sebastian Cammerer, et al. (Mar. 2022), « Sionna: An Open-Source Library for Next-Generation Physical Layer Research », *in*.

Huang, Pengzhi et al. (2019), « Improving Channel Charting with Representation-Constrained Autoencoders », *in*: *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, Cannes, France, Cannes, France: IEEE, pp. 1–5.

Jaeckel, Stephan et al. (2014), « QuaDRiGa: A 3-D multi-cell channel model with time evolution for enabling virtual field trials », *in*: *IEEE Transactions on Antennas and Propagation* 62.6, pp. 3242–3256.

Jaiswal, Ashish et al. (Dec. 2020), « A Survey on Contrastive Self-Supervised Learning », *in*: *Technologies* 9.1, p. 2.

Kamilov, Ulugbek S and Hassan Mansour (2016), « Learning optimal nonlinearities for iterative thresholding algorithms », *in*: *IEEE Signal Processing Letters* 23.5, pp. 747–751.

Kazemi, Parham, Tushara Ponnada, et al. (June 2021), « Channel Charting Based Beam SNR Prediction », *in*: *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, IEEE, pp. 72–77.

Kazemi, Parham, Hanan Al-Tous, et al. (2020), « SNR Prediction in Cellular Systems based on Channel Charting », *in*: *2020 IEEE Eighth International Conference on Communications and Networking (ComNet)*, pp. 1–8.

— (June 2022), « Channel Charting Assisted Beam Tracking », *in*: *2022 IEEE 95th Vehicular Technology Conference: (VTC2022-Spring)*, IEEE, pp. 1–5.

Kela, Petteri et al. (2016), « Location based beamforming in 5G ultra-dense networks », *in*: *2016 IEEE 84th Vehicular Technology Conference (VTC-Fall)*, IEEE, pp. 1–7.

Kingma, Diederik P. and Max Welling (Dec. 2014), « Auto-Encoding Variational Bayes », *in*: *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, arXiv.

Larsson, Erik G et al. (2014), « Massive MIMO for next generation wireless systems », *in*: *IEEE Communications Magazine* 52.2, pp. 186–195.

Le Magoarou, Luc (2021a), « Efficient Channel Charting via Phase-Insensitive Distance Computation », *in*: *IEEE Wireless Commun. Lett.* 10 (12), pp. 2634–2638.

— (2021b), « Similarity-Based Prediction for Channel Mapping and User Positioning », *in*: *IEEE Commun Lett* 25 (5), pp. 1578–1582.

Le Magoarou, Luc and Stéphane Paquelet (2018), « Parametric channel estimation for massive MIMO », *in*: *IEEE Statistical Signal Processing Workshop (SSP)*.

Le Magoarou, Luc, Taha Yassine, et al. (Oct. 2022a), « Channel charting based beamforming », *in*: *2022 56th Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, USA, Pacific Grove, CA, USA: IEEE, pp. 1185–1189.

— (May 2022b), « Deep Learning for Location Based Beamforming with Nlos Channels », *in*: *ICASSP 2022 - 2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, Singapore, Singapore, Singapore, Singapore: IEEE, pp. 8812–8816.

Leeuw, Jan de (1977), « Application of convex analysis to multidimensional scaling », *in*: *Recent developments in statistics*, pp. 133–145.

Lei, Eric et al. (2019), « Siamese Neural Networks for Wireless Positioning and Channel Charting », *in*: *Proc. 57th Ann. Allerton Conf. Commun., Control, and Comput.* Monticello, IL, USA, Monticello, IL, USA: IEEE, pp. 200–207.

Lu, Lu et al. (2014), « An overview of massive MIMO: Benefits and challenges », *in*: *IEEE journal of selected topics in signal processing* 8.*5*, pp. 742–758.

Maaten, Laurens van der and Geoffrey Hinton (2008), « Visualizing Data using t-SNE », *in*: *Journal of Machine Learning Research* 9.*86*, pp. 2579–2605.

Mairal, Julien et al. (Jan. 2010), « Online Learning for Matrix Factorization and Sparse Coding », Anglais, *in*: *Journal of Machine Learning Research* 11.*1*, pp. 19–60.

Makhzani, Alireza and Brendan Frey (2014), « k-Sparse Autoencoders. », *in*: *arXiv preprint arXiv:1312.5663*, ed. by Yoshua Bengio and Yann LeCun.

Mallat, S.G. and Z. Zhang (Dec. 1993), « Matching pursuits with time-frequency dictionaries », *in*: *Signal Processing, IEEE Transactions on* 41.*12*, pp. 3397–3415.

Mateos-Ramos, José Miguel et al. (Dec. 2022), « Model-Driven End-to-End Learning for Integrated Sensing and Communication », *in*.

Merluzzi, Mattia et al. (2023), « The Hexa-X Project Vision on Artificial Intelligence and Machine Learning-Driven Communication and Computation Co-Design for 6G », *in*: *IEEE Access* 11, pp. 65620–65648.

Metzler, Chris, Ali Mousavi, and Richard Baraniuk (2017), « Learned D-AMP: Principled neural network based compressive image recovery », *in*: *Advances in Neural Information Processing Systems*, pp. 1772–1783.

Metzler, Christopher A, Arian Maleki, and Richard G Baraniuk (2016), « From denoising to compressed sensing », *in*: *IEEE Transactions on Information Theory* 62.*9*, pp. 5117–5144.

Mildenhall, Ben et al. (Dec. 2021), « NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis », *in*: *Commun. ACM* 65.*1*, pp. 99–106.

Molisch, A.F. (2022), *Wireless Communications: From Fundamentals to Beyond 5G*, IEEE Press, Wiley.

Monga, Vishal, Yuelong Li, and Yonina C. Eldar (2021), « Algorithm Unrolling: Interpretable, Efficient Deep Learning for Signal and Image Processing », *in*: *IEEE Signal Processing Magazine* 38.*2*, pp. 18–44.

Nair, Vinod and Geoffrey E. Hinton (2010), « Rectified Linear Units Improve Restricted Boltzmann Machines », *in*: *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, Haifa, Israel: Omnipress, pp. 807–814.

Neculoiu, Paul, Maarten Versteegh, and Mihai Rotaru (Aug. 2016), « Learning Text Similarity with Siamese Recurrent Networks », *in*: *Proceedings of the 1st Workshop on Representation Learning for NLP*, ed. by Phil Blunsom et al., Berlin, Germany: Association for Computational Linguistics, pp. 148–157.

O'Shea, Timothy and Jakob Hoydis (2017), « An introduction to deep learning for the physical layer », *in*: *IEEE Transactions on Cognitive Communications and Networking* 3.4, pp. 563–575.

OpenStreetMap contributors (2017), *Planet dump retrieved from https://planet.osm.org*, https://www.openstreetmap.org.

Pai, Gautam et al. (2019), « DIMAL: Deep Isometric Manifold Learning Using Sparse Geodesic Sampling », *in*: Waikoloa, HI, USA, Waikoloa, HI, USA: IEEE, pp. 819–828.

Paszke, Adam et al. (2019), « Pytorch: An imperative style, high-performance deep learning library », *in*: *Advances in neural information processing systems* 32.

Peyré, Gabriel (2009), « Manifold models for signals and images », *in*: *Computer Vision and Image Understanding* 113.2, pp. 249–260.

Pihlajasalo, Jaakko et al. (Nov. 2020), « Absolute Positioning with Unsupervised Multipoint Channel Charting for 5G Networks », *in*: *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, IEEE, pp. 1–5.

Ponnada, Tushara et al. (June 2021), « Best Beam Prediction in Non-Standalone mm Wave Systems », *in*: *2021 Joint European Conference on Networks and Communications; 6G Summit (EuCNC/6G Summit)*, IEEE.

Qin, Zhijin et al. (2019), « Deep learning in physical layer communications », *in*: *IEEE Wireless Communications* 26.2, pp. 93–99.

Rahaman, Nasim et al. (Sept. 2019), « On the Spectral Bias of Neural Networks », *in*: *Proceedings of the 36th International Conference on Machine Learning*, ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov, vol. 97, Proceedings of Machine Learning Research, PMLR, pp. 5301–5310.

Rahimi, Ali and Benjamin Recht (2007), « Random Features for Large-Scale Kernel Machines », *in*: *Proceedings of the 20th International Conference on Neural Information Processing Systems*, NIPS'07, Vancouver, British Columbia, Canada: Curran Associates Inc., pp. 1177–1184.

Rappaport, Brian et al. (2021), « Improving Channel Charting using a Split Triplet Loss and an Inertial Regularizer », *in*: *Proc. IEEE 17th Int. Symp. Wireless Commun. Syst. (ISWCS)*, Berlin, Germany, Berlin, Germany: IEEE, pp. 1–6.

Remcom (n.d.), *Wireless InSite*, http://www.remcom.com/wireless-insite.

Ribeiro, Lucas, Markus Leinonealn, et al. (2022), « Channel Charting Aided Pilot Allocation in Multi-Cell Massive MIMO mMTC Networks », *in*: *2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, pp. 1–5.

Ribeiro, Lucas, Markus Leinonen, et al. (2020), « Channel Charting for Pilot Reuse in mMTC with Spatially Correlated MIMO Channels », *in*: *2020 IEEE Globecom Workshops (GC Wkshps*, pp. 1–6.

Rumelhart, D. E., G. E. Hinton, and R. J. Williams (1986), « Learning Internal Representations by Error Propagation », *in*: *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, Cambridge, MA, USA: MIT Press, pp. 318–362.

Rusek, Fredrik et al. (2013), « Scaling up MIMO: Opportunities and challenges with very large arrays », *in*: *IEEE Signal Processing Magazine* 30.*1*, pp. 40–60.

Samimi, Mathew K and Theodore S Rappaport (2016), « 3-D millimeter-wave statistical channel model for 5G wireless system design », *in*: *IEEE Transactions on Microwave Theory and Techniques* 64.*7*, pp. 2207–2225.

Samsung Electronics (2020), *Massive MIMO for New Radio*, tech. rep.

Samuel, Neev, Tzvi Diskin, and Ami Wiesel (2017), « Deep MIMO detection », *in*: *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, IEEE, pp. 1–5.

Sayeed, Akbar M (2002), « Deconstructing multiantenna fading channels », *in*: *IEEE Transactions on Signal Processing* 50.*10*, pp. 2563–2579.

Schroff, Florian, Dmitry Kalenichenko, and James Philbin (June 2015), « FaceNet: A Unified Embedding for Face Recognition and Clustering », *in*: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Shlezinger, Nir et al. (Dec. 2020), « Model-Based Deep Learning », *in*: *arXiv preprint arXiv:2012.08405*.

Simonyan, K and A Zisserman (2015), « Very deep convolutional networks for large-scale image recognition », *in*: Computational and Biological Learning Society, pp. 1–14.

Soltani, Mehran et al. (2019), « Deep learning-based channel estimation », *in*: *IEEE Communications Letters* 23.4, pp. 652–655.

Stahlke, Maximilian et al. (2023), « Indoor Localization With Robust Global Channel Charting: A Time-Distance-Based Approach », *in*: *IEEE Transactions on Machine Learning in Communications and Networking* 1, pp. 3–17.

Stephan, Phillip, Florian Euchner, and Stephan ten Brink (2023), « Angle-Delay Profile-Based and Timestamp-Aided Dissimilarity Metrics for Channel Charting », *in*: *arXiv preprint arXiv:2308.09539*.

Studer, Christoph et al. (2018), « Channel Charting: Locating Users Within the Radio Environment Using Channel State Information », *in*: *IEEE Access* 6, pp. 47682–47698.

Swati et al. (Oct. 2017), « Siamese Networks for Chromosome Classification », *in*: *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, IEEE, pp. 72–81.

Tancik, Matthew et al. (2020), « Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains », *in*: *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Vancouver, BC, Canada: Curran Associates Inc.

Taner, Sueda, Maxime Guillaud, et al. (2023), « Channel Charting for Streaming CSI Data », *in*: *arXiv preprint arXiv:2312.04514*.

Taner, Sueda, Victoria Palhares, and Christoph Studer (Aug. 2023), « Channel Charting in Real-World Coordinates », *in*.

Tenenbaum, Joshua B., Vin de Silva, and John C. Langford (Dec. 2000), « A Global Geometric Framework for Nonlinear Dimensionality Reduction », *in*: *Science* 290.5500, pp. 2319–2323.

Al-Tous, Hanan, Parham Kazemi, et al. (2022), « Channel Charting with Angle-Delay-Power-Profile Features and Earth-Mover Distance », *in*: Pacific Grove, CA, USA, Pacific Grove, CA, USA: IEEE, pp. 1195–1201.

Al-Tous, Hanan, Tushara Ponnada, et al. (June 2020), « Multipoint channel charting-based radio resource management for V2V communications », *in*: *EURASIP Journal on Wireless Communications and Networking* 2020.*1*.

van Lier, Michel et al. (2020), « OptComNet: Optimized Neural Networks for Low-Complexity Channel Estimation », *in*: *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, Dublin, Ireland, Dublin, Ireland: IEEE, pp. 1–6.

Varior, Rahul Rama et al. (2016), « A Siamese Long Short-Term Memory Architecture for Human Re-identification », *in*: *Lecture Notes in Computer Science*, Springer International Publishing, pp. 135–153.

Vaswani, Ashish et al. (2017), « Attention is All You Need », *in*: *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, Long Beach, California, USA: Curran Associates Inc., pp. 6000–6010.

Vieira, Joao et al. (2017), « Deep convolutional neural networks for massive MIMO fingerprint-based positioning », *in*: *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, IEEE, pp. 1–6.

Wang, Tianqi et al. (2017), « Deep learning for wireless physical layer: Opportunities and challenges », *in*: *China Communications* 14.*11*, pp. 92–111.

Wang, Xuyu et al. (2015), « DeepFi: Deep learning for indoor fingerprinting using channel state information », *in*: *2015 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1666–1671.

Wei, Xiuhong, Chen Hu, and Linglong Dai (2019), « Knowledge-Aided Deep Learning for Beamspace Channel Estimation in Millimeter-Wave Massive MIMO Systems », *in*: *arXiv preprint arXiv:1910.12455*.

Weisstein, Eric W (2020), « Bessel function zeros », *in*: *MathWorld–A Wolfram Web Resource*.

Wu, Rui, Wei Huang, and Di-Rong Chen (2012), « The exact support recovery of sparse signals with noise via orthogonal matching pursuit », *in*: *IEEE signal processing letters* 20.*4*, pp. 403–406.

Wymeersch, Henk et al. (2021), « Integration of Communication and Sensing in 6G: a Joint Industrial and Academic Perspective », *in*: Helsinki, Finland, Helsinki, Finland: IEEE, pp. 1–7.

Yassine, Taha, Baptiste Chatelier, et al. (Oct. 2023), « Model-Based Deep Learning for Beam Prediction Based on a Channel Chart », *in*: *2023 57th Asilomar Conference on Signals, Systems, and Computers*, IEEE, pp. 1636–1640.

Yassine, Taha, Luc Le Magoarou, et al. (July 2022), « Leveraging triplet loss and nonlinear dimensionality reduction for on-the-fly channel charting », *in*: *2022 IEEE 23rd International Workshop on Signal Processing Advances in Wireless Communication (SPAWC)*, Oulu, Finland, Oulu, Finland: IEEE, pp. 1–5.

Yin, Ziqing et al. (2022), « Deep CSI Compression for Massive MIMO: A Self-Information Model-Driven Neural Network », *in*: *IEEE Transactions on Wireless Communications* 21.*10*, pp. 8872–8886.

Zhang, Qianqian and Walid Saad (Dec. 2021), « Semi-Supervised Learning for Channel Charting-Aided IoT Localization in Millimeter Wave Networks », *in*: *2021 IEEE Global Communications Conference (GLOBECOM)*, IEEE, pp. 1–6.

COLLEGE DOCTORAL BRETAGNE | MATHS, TELECOMS INFORMATIQUE, SIGNAL SYSTEMES, ELECTRONIQUE

INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES RENNES

Titre : Approches hybrides pour le traitement de données : Concilier apprentissage automatique et modèles analytiques

Mot clés : Estimation du canal, cartographie du canal, apprentissage automatique, apprentissage profond basé sur un modèle

Résumé : Les nouvelles générations de systèmes de communication nous permettront d'entrer dans un monde de plus en plus connecté. Elles ouvriront la voie à des applications telles que la réalité augmentée en temps réel, des villes plus intelligentes et des soins médicaux à distance. Les systèmes à entrées multiples et sorties multiples (MIMO) sont un exemple de technologies habilitantes sur lesquelles ces systèmes seront construits. En combinant de multiples antennes pour transmettre et recevoir de l'information, ils permettent une utilisation des ressources plus efficace. Cependant, ils présentent de nouveaux défis en raison de leur complexité sous-jacente. Les algorithmes traditionnels qui font partie des blocs de construction fondamentaux des systèmes de communication atteignent leurs limites. D'un autre côté, l'apprentissage automatique en général, et l'apprentissage profond en particulier, ont émergé comme une alternative puissante aux méthodes traditionnelles, mais nécessitent souvent de grandes quantités de données. Les approches d'apprentissage profond basées sur des modèles semblent être un compromis entre la simplicité des méthodes classiques et la flexibilité de celles pilotées par les données. Mais comment ces approches s'appliquent-elles en pratique aux systèmes de communication ?

Dans ce manuscrit, l'estimation du canal et sa cartographie sont explorées. La première consiste à estimer le canal entre deux dispositifs de communication en s'appuyant sur des signaux transmis. L'utilisation d'un grand nombre d'antennes rend la tâche difficile. La cartographie de canal, quant à elle, est similaire à la localisation d'utilisateur, mais se concentre sur l'identification de leurs voisinages locaux et doit être menée de manière non supervisée.

Title: Hybrid approaches for data processing: Bridging machine learning and analytical models

Keywords: Channel Estimation, Channel Charting, Machine Learning, Model-based Deep Learning

Abstract: The new generations of communication systems will enable us to enter a world ever more connected. They will unlock applications such as real-time augmented reality, smarter cities and remote medical care. Multiple input multiple output (MIMO) systems are one example of enabling technologies on which these systems will be built. By combining multiple antennas to transmit and receive information, they allow for a more efficient use of resources. They however come with new challenges due to their underlying complexity. Traditional algorithms that are part of the fundamental building blocks of communication systems show their limits. On the other hand, machine learning in general, and deep learning in particular have emerged as a powerful alternative to handcrafted methods, but often require substantial amounts of data. Model-based deep learning appoaches arise as a tradeoff between the simplicity of classical methods and the flexibility of data-driven ones. But how do these approaches apply in practice to communication systems?

In this manuscript, both channel estimation and channel charting are explored. The former consists of estimating the channel between two communication devices by relying on transmitted signals. The use of a high number of antennas makes the task challenging. Channel charting on the other hand is akin to user localization, but focuses on identifying their local neighborhoods and must be conducted in an unsupervised manner.