UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA

YOSBEL RODRÍGUEZ ORTEGA

GRAPH SIGNAL PROCESSING AND MACHINE LEARNING FOR PREDICTION PROBLEMS IN MOBILE COMMUNICATIONS SYSTEMS

FORTALEZA
2021

YOSBEL RODRÍGUEZ ORTEGA

GRAPH SIGNAL PROCESSING AND MACHINE LEARNING FOR PREDICTION PROBLEMS IN MOBILE COMMUNICATIONS SYSTEMS

Tese apresentada ao Curso de Doutorado em Engenharia de Teleinformática da Universidade Federal do Ceará, como parte dos requisitos para obtenção do Título de Doutor em Engenharia de Teleinformática. Área de concentração: Sinais e Sistemas

Orientador: Prof. Dr. Francisco Rodrigo Porto Cavalcanti

Coorientador: Dr. Igor Moaco Guerreiro

FORTALEZA
2021

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária
Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

O88g Ortega, Yosbel Rodríguez.
Graph Signal Processing and Machine Learning for Prediction Problems in Mobile Communications Systems / Yosbel Rodríguez Ortega. – 2021.
122 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Fortaleza, 2021.
Orientação: Prof. Dr. Prof. Dr. Francisco Rodrigo Porto Cavalcanti.
Coorientação: Prof. Dr. Dr. Igor Moaco Guerreiro.

1. Graph signal processing. 2. Signal sampling and reconstruction. 3. Machine learning. 4. Beam management. 5. Feedback channel burden. I. Título.

CDD 621.3

YOSBEL RODRÍGUEZ ORTEGA

GRAPH SIGNAL PROCESSING AND MACHINE LEARNING FOR PREDICTION PROBLEMS IN MOBILE COMMUNICATIONS SYSTEMS

Tese apresentada ao Curso de Doutorado em Engenharia de Teleinformática da Universidade Federal do Ceará, como parte dos requisitos para obtenção do Título de Doutor em Engenharia de Teleinformática. Área de concentração: Sinais e Sistemas

Aprovado em: 10/05/2021.

BANCA EXAMINADORA

Prof. Dr. Francisco Rodrigo Porto Cavalcanti (Orientador)
Universidade Federal do Ceará

Dr. Igor Moaco Guerreiro (Coorientador)
Universidade Federal do Ceará

ACKNOWLEDGEMENTS

This new professional and personal achievement would not have been possible without the help and support of many people who are part of my life. I would like to offer my sincere thanks to all of them.

First and above all, I thank my wife Ivonne Montero for all her unconditional support, love, understanding and for providing me with my greatest treasure, my princess Isabella.

I also would like to thank and dedicate this achievement to my family in Cuba, for the spiritual support in all this time despite the distance. A lot of understanding and love is needed to achieve a goal away from the family. Especially my mother Victoria and my daughter Chenoa.

Special thanks go to my supervisor Prof. Dr. Fco. Rodrigo P. Cavalcanti. I greatly appreciate the trust placed in me and the guidance and support you offered when needed throughout this period. Your advice on both research as well as on my career have been invaluable.

I would also like to thank my co-supervisor Dr. Igor Moaco Guerreiro who guided me step by step into the graph research area and advised me throughout this period.

To my friends and colleagues of the Wireless Telecommunications Group-GTEL, for their help and for making me feel like another Brazilian.

To GTEL teachers and workers for their understanding, friendliness and support during this years.

Finally, I acknowledge the technical and financial support from FUNCAP, under grant BMD-0008-00514.01.19/17, CAPES under Finance Code 001 and Ericsson Research, Sweden, under UFC.46 and in part by CNPq (grants 315755/2018-0 and 309472/2017-2).

RESUMO

As comunicações móveis em cenários de quinta geração enfrentam desafios significativos devido à adoção de aplicativos em tempo real e serviços de missão crítica com requisitos rigorosos, como maior largura de banda e confiabilidade, menor latência e capacidade de suportar um grande número de usuários com uma intensa transferência de dados com as estações base. A fim de enfrentar alguns desses desafios, a presente tese analisa soluções para problemas de previsão baseados em processamento de sinais em grafo em conjunto com técnicas de aprendizagem supervisionada (ML, do inglês *machine learning*), para lidar de forma eficiente com dois casos de uso relevantes em sistemas de comunicações móveis, isto é: (i) gerenciamento de feixe e, (ii) aumento da carga do canal de *feedback*. Especificamente, ferramentas de amostragem e reconstrução baseadas em grafos são exploradas a fim de reduzir a carga do canal de *feedback*, que é usado para relatar as medições de usuário que alimentam os preditores baseados em ML. Antes de abordar esses dois casos de uso, é apresentada uma visão geral de algumas definições da teoria de grafos e são apresentadas algumas das ferramentas e conceitos que são empregados ao longo desta tese. Com relação ao problema de gerenciamento de feixes, é proposto um *framework* adaptativo de rastreamento de feixes para comunicações altamente direcionais, que explora as amostras em um conjunto de dados histórico de usuário, para estimar e prever eficientemente o estado do canal na estação base. Primeiramente, um algoritmo de aprendizado supervisionado, isto é K-vizinhos mais próximos (K-NN do inglês, *K-nearest neighbors*), é proposto e avaliado para rastreamento e predição de canal. Em seguida, como estágios preliminares do algoritmo K-NN, uma estratégia de amostragem e reconstrução baseada em grafo, denominada K-vizinhos mais próximos com reconstrução (K-NN-R), é proposta a fim de reduzir o espaço de busca do feixe durante a etapa de medição, o que permite um uso mais eficiente do canal de *feedback*. Depois disso, a estrutura bloco-diagonal sobre o grafo que representa um conjunto de dados de medições em uma rede celular veículo-para-tudo (C-V2X do inglês, *cellular vehicular-to-everything*) é explorada e, com base nela, uma estratégia de amostragem baseada em grafo, isto é amostragem Laplaciana inteligente (SLS, do inglês *smart Laplacian sampling*), é proposta para que (i) a estrutura/conectividade do grafo seja preservada e (ii) a quantidade de medições de usuário amostradas seja reduzida. Por fim, a proposta SLS é avaliada através de um problema de predição de latência, no qual é predito se um pacote pode ser entregue dentro de uma restrição de latência predeterminada.

Palavras-chave: processamento de sinais em grafo, amostragem e reconstrução de sinais, aprendizado de máquina, gerenciamento de feixe, carga do canal de *feedback*.

ABSTRACT

Mobile communications in fifth generation (5G) scenarios still face significant challenges due to the adoption of real-time applications and mission-critical services with stringent requirements, such as greater bandwidth and reliability, lower latency and the capacity to support a massive number of user equipments (UEs) with an intense data transfer between them and the base-stations (BSs). In order to tackle some of these challenges, the present thesis analyzes solutions for prediction problems based on graph signal processing (GSP) jointly with machine learning (ML) techniques, to efficiently deal with two relevant use cases of mobile communication systems, that are: (i) beam management and, (ii) increased feedback channel burden. Specifically, graph-based sampling and reconstruction tools are exploited in order to reduce the feedback channel burden, which is used to report the UE measurements that feed the ML-based predictors. Before addressing these two use cases, it is presented an overview of some graph-theoretic definitions and were introduced some of the tools and concepts that are employed throughout this thesis. Concerning the beam management problem, it is proposed an adaptive beam tracking framework for highly directional communications, that exploits the samples in a historical UE dataset (HUD), to efficiently estimate and predict the channel state at the BS. First, a supervised learning algorithm, namely K-nearest neighbors (K-NN), is proposed and evaluated for channel tracking and prediction. Then, as preliminary stages of the K-NN algorithm, a graph-based sampling and reconstruction strategy, called K-nearest neighbors with reconstruction (K-NN-R), is proposed in order to reduce the beam search space during the beam measurement stage, which allows a more efficient usage of the feedback channel. After that, the block-diagonal structure over graph that represents a cellular vehicular-to-everything (C-V2X) measurement dataset is exploited, and based on it, a graph-based sampling strategy, namely smart Laplacian sampling (SLS), is proposed so that (i) the graph structure/connectivity is preserved, and (ii) the amount of sampled UE measurements is reduced. Finally, the SLS proposal is evaluated through a latency prediction problem, in which is predicted whether a packet can be delivered within a predetermined latency constraint.

Keywords: graph signal processing, signal sampling and reconstruction, machine learning, beam management, feedback channel burden.

LIST OF FIGURES

Figure 1.1 – Typical fifth generation (5G) scenario with two technologies, long term evolution (LTE) and new radio (NR), coexisting. Depending on the assumed 5G architecture, the non-standalone architecture (NSA) relies on 4G EPC, while standalone architecture (SA) uses a 5G core. 22
Figure 1.2 – Beam management challenges in millimeter wave (mmWave) communications. 25
Figure 1.3 – Thesis structure. 31
Figure 2.1 – Different graph Laplacian eigenvectors for a random sensor graph with 20 vertexes. Graph vertex colors in (a) represent graph signal values, while in (b)-(e) they correspond to eigenvector values. 38
Figure 2.2 – Example of graph with multiple connected components. 39
Figure 2.3 – Result of sampling nodes using a random sampling strategy. Note that each of subgraph 2 is sampled. 44
Figure 2.4 – Example of graph signal reconstruction using the three batch reconstruction methods. 48
Figure 3.1 – Spatial mmWave channel model. 52
Figure 3.2 – K-NN Tracking Framework. 58
Figure 3.3 – Observation process over graph. 60
Figure 3.4 – Proposed K-NN-R tracking strategy process. 62
Figure 3.5 – Computational complexity of the tracking algorithms for $N_s = 1000$ and $G = 180$. The antenna configuration is uniform linear array (ULA) with 64 elements at base-station (BS) and 8 elements at user equipment (UE). 67
Figure 3.6 – Cross-validation K-NN for different values of β showing estimation error against K. There are 64 elements at BS and 8 elements at UE. 68
Figure 3.7 – Cross-validation of K-NN-R for different values of β and R_s, showing estimation error against K. There are 64 elements at BS and 8 elements at UE. 69
Figure 3.8 – Cross-validation of K-NN for $\beta = 10\%$ and different values of $\acute{N}_\tau$ showing prediction error against K. There are 64 elements at BS and 8 elements at UE. 69
Figure 3.9 – K-NN-R cross-validation for $\beta = 10\%$, $R_s = 25\%, 50\%$ and different values of $\acute{N}_\tau$, showing prediction error against K. There are 64 elements at BS and 8 elements at UE. 70
Figure 3.10–Analysis for different HUD size in tracking of the channel matrix $\mathbf{H}$, for $\beta = 10\%$, $K = 6$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE. 70
Figure 3.11–SNR analysis in tracking of the channel matrix $\mathbf{H}$ for $\sigma_u = 0.5°$, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE. 71

Figure 3.12–Standard deviation (σ_u) analysis in tracking of the channel matrix $\mathbf{H}$ for SNR = 20 dB, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE. 72
Figure 3.13–UE speed analysis in tracking of the channel matrix $\mathbf{H}$ for SNR = 20 dB, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE. 73
Figure 3.14–Tracking of the channel matrix $\mathbf{H}$ for $\beta = 10\%$ and $R_s = 25\%, 50\%$. The antenna configuration is ULA with 32 and 64 elements at BS, and 8 elements at UE. 74
Figure 3.15–Accuracy of the tracking methods finding the best beam for $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are either 32 or 64 elements at BS, and 8 elements at UE. 75
Figure 3.16–Channel prediction evaluation with K-nearest neighbors (K-NN) for $\beta = 10\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 76
Figure 3.17–Accuracy of finding the best beam with K-NN method for $\beta = 10\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 77
Figure 3.18–Channel prediction evaluation with K-nearest neighbors with reconstruction (K-NN-R) for $\beta = 10\%$, $R_s = 50\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 78
Figure 3.19–Accuracy of finding the best beam with K-NN-R for $\beta = 10\%$, $R_s = 50\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 78
Figure 3.20–Channel prediction evaluation with K-NN-R for $\beta = 10\%$, $R_s = 25\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 79
Figure 3.21–Accuracy of finding the best beam with K-NN-R for $\beta = 10\%$, $R_s = 25\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE. 79
Figure 4.1 – General considerations of the problem. 83
Figure 4.2 – Example of graph with multiple connected components showing the different subgraph types. 86
Figure 4.3 – Sampling and reconstruction procedures for the k-th subgraph signal at time instant τ. 87
Figure 4.4 – Example of smart Laplacian sampling (SLS) sampling strategy. 89
Figure 4.5 – Urban scenario with a wrapped-around seven-site hexagon layout, each site comprising three cells, on a Manhattan grid. 90
Figure 4.6 – Latency prediction problem. 91
Figure 4.7 – Reconstructed signals are used as the input of the machine learning experiment. 91

Figure 4.8 – Network performance. 94
Figure 4.9 – Reconstruction normalized mean square error (NMSE) of feature vectors using different reconstruction strategies. online smooth graph-time difference (O-SGTD) is compared with three batch reconstruction strategies using random sampling (RS) with different sampling rate. 96
Figure 4.10–NMSE of the three baseline algorithms with different sampling rate. 97
Figure 4.11–CDF curves of the number of unsampled UE measurements. SLS is compared with the three baseline algorithms. 98
Figure 4.12–Average of unsampled UE measurements per UE. 99
Figure 4.13–Number of unsampled UE measurements at each time instant. 99
Figure 4.14–Reconstruction NMSE of feature vectors using different sampling strategies: SLS and the three baseline algorithms. 100
Figure 4.15–Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS. 101
Figure 4.16–Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS. 102
Figure 4.17–f1-score of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS. 103
Figure 4.18–f1-score of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS. 104
Figure 4.19–Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) up to three UE measurements randomly chosen, iv) reconstructed feature vectors through O-SGTD with SLS. 105
Figure 4.20–Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) up to three UE measurements randomly chosen, iv) reconstructed feature vectors through O-SGTD with SLS. 106
Figure 4.21–Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) three sampling baseline algorithms, iv) reconstructed feature vectors through O-SGTD with SLS. 107
Figure 4.22–Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) three sampling baseline algorithms, iv) reconstructed feature vectors through O-SGTD with SLS. 107

LIST OF TABLES

Table 3.1 – Channel, system and simulation parameters. 64
Table 3.2 – Tracking algorithms setup. 64
Table 4.1 – Simulation parameters. 92
Table 4.2 – Learning, sampling and reconstruction parameters. 92
Table 4.3 – Runtime of the reconstruction algorithms. 95

LIST OF ABBREVIATIONS AND ACRONYMS

K-NN	K-nearest neighbors
K-NN-R	K-nearest neighbors with reconstruction
1G	first generation
2D	2-dimensional
2G	second generation
3D	3-dimensional
3G	third generation
3GPP	third generation partnership project
4G	fourth generation
5G	fifth generation
AMP	approximate message passing
AoA	angle of arrival
AoD	angle of departure
BLER	block error rate
BS	base-station
BV	beamforming vector
C-V2X	cellular vehicular-to-everything
CDF	cumulative distribution function
CGL	combinatorial graph Laplacian
CQI	channel quality indicator
CRS	cell-specific reference signal
CS	compressive sensing
CSI	channel state information
CV	combining vector
DDGL	diagonally dominant generalized graph Laplacian
DFT	discrete Fourier transform
EKF	extended Kalman filter
eMBB	Enhanced mobile broadband
EPC	evolved packet core
GFT	graph Fourier transform
GGL	generalized graph Laplacian
GSP	graph signal processing
GTTR	graph-time Tikhonov reconstruction
HARQ	hybrid automatic repeat request
HUD	historical UE dataset
IGFT	inverse graph Fourier transform

IoT	internet of things
IRC	interference rejection combination
IV	isolated vertex
IV-BS	subgraph with isolated-vertex from BS
IV-UE	subgraph with isolated-vertex from UE
KF	classical Kalman filters
LMS	least mean square
LR-MC	low-rank matrix completion
LS	least square
LSTM	long short term memory
LTE	long term evolution
Max-Det	maximization of determinant
Max-Mineig	maximization of the minimum nonzero eigenvalue
MIMO	multiple-input-multiple-output
Min-Msd	minimization of mean-square desviation
ML	machine learning
MLP	multilayer perceptron
mMTC	Massive machine type communication
mmWave	millimeter wave
MP	matching pursuit
MSD	mean-square desviation
MV	multiple vertexes
MV-BS	subgraph with multiple-vertexes from BS
MV-BU	subgraph with multiple-vertexes from BS and UE
MV-UE	subgraph with multiple-vertexes from UE
NMSE	normalized mean square error
NN	neural network
NR	new radio
NSA	non-standalone architecture
O-SGTD	online smooth graph-time difference
OFDM	orthogonal frequency-division multiplexing
OMP	orthogonal matching pursuit
PF	particle filter
QoS	quality of service
RAN	radio access network
RF	random forest
RLS	recursive least square
RNN	recurrent neural network
RS	random sampling

RSRP	reference signal received power
RSRQ	reference signal received quality
SA	standalone architecture
SCMA	sparse code multiple access
SGSR	smooth graph signal reconstruction
SGTD	smooth graph-time difference
SINR	signal-to-interference-plus-noise ratio
SLS	smart Laplacian sampling
SMS	short message service
SNR	signal-to-noise ratio
TTI	transmission time interval
TV	total variation
UDN	ultra dense network
UE	user equipment
UKF	unscented Kalman filters
ULA	uniform linear array
URLLC	Ultra-reliable and low-latency communications

LIST OF SYMBOLS

$\|\|\cdot\|\|_F$	Frobenius norm operator
$\odot$	Hadamard product operator
$(\cdot)^{\mathrm{T}}$	Vector/Matrix transposition operator
$(\cdot)^{\mathrm{H}}$	Hermitian conjugate operator
$\mathrm{Tr}(\cdot)$	Trace operator
$\mathbf{1}$	Column vector of ones
$\mathbf{0}$	Column vector of zeros
$\mathbf{A}$	Connectivity adjacency matrix of a graph $\mathcal{G}$
A	Angle of arrival
$\mathbf{a}_r(\cdot)$	Beam steering vectors for the UE
$\mathbf{a}_t(\cdot)$	Beam steering vectors for the BS
α	Complex gain
$\boldsymbol{\alpha}_\tau$	Complex gain vector
β	Correlation factor
φ, γ and δ	Regularization parameters
$\mathbf{D}$	Degree matrix of a graph $\mathcal{G}$
Δ	Maximum random delay
D	Angle of departure
d	Distance between the adjacent antennas
$\mathbf{f}_p$	Beamforming vector for transmission
$\mathbf{f}_q$	Combining vector for reception
$\mathcal{G}$	N-vertex, undirected, connected, and weighted graph
$\mathcal{G}_k$	Subgraph of the graph $\mathcal{G}$
GFT^{-1}	Inverse graph Fourier transform
$\mathbf{H}(\tau)$	Frequency response of a time-varying geometric 3-dimensional (3D) channel model
$\mathbf{H}(\mathcal{X}_\tau)$	Channel matrix parameterized by the state vector $\mathcal{X}_\tau$
$\mathbf{I}$	Identity matrix
K	Number of connected components of a graph $\mathcal{G}$
$\mathbf{k}$	Wave vector
$[\bar{\mathbf{M}}_\tau]_{u,u}$	Linear mask sampling operator
$[\mathbf{M}_\tau]_{u,\tau}$	Linear mask sampling operator over time

N	Number of graph vertexes
N_τ	Reporting temporal window
$\acute{N}_\tau$	Observed temporal window
N_t	Number of antenna elements at BS
N_r	Number of antenna elements at UE
$\mathcal{N}_j$	Zero-mean white Gaussian noise vector
∇_τ	Temporal difference operator
$[\nabla \mathbf{X}]_\tau$	Temporal difference signal operator
p	Single path
P_τ	Set of sampled vertexes at each time instant τ
ϕ	Azimuth component of the transmission direction
$\boldsymbol{\phi}_{D,\tau}$	AoD in the azimuth
$\boldsymbol{\phi}_{A,\tau}$	AoA in the azimuth
L	Number of clusters of a single scatterer
l	Cluster index
$\mathbf{L}$	Laplacian matrix of a graph $\mathcal{G}$
$\mathbf{L}_{CGL}$	Combinatorial graph Laplacian matrix
$\mathbf{L}_{GGL}$	Generalized graph Laplacian matrix
$\mathbf{L}_k$	Laplacian matrix of a subgraph $\mathcal{G}_k$
ν	Set of N vertexes
ν_k	Subset of the vertex set ν
λ	Carrier wavelength
λ_n	Set of real non-negative eigenvalues
$\mathbf{\Lambda}$	Matrix whose diagonal elements are the set of real eigenvalues λ_n
r	Receiver index
ρ	Channel time correlation coefficient
s	Pilot signal
$\mathbf{S}$	Data statistics matrix
$S(\mathbf{x})$	Smoothness of a graph signal $\mathbf{x}$
$S(\mathbf{x}_\tau)$	Smoothness of time-varying graph signal $\mathbf{x}_\tau$
σ_u	Angular standart deviation
t	Transmitter index
θ	Elevation component of the transmission direction

ε	Set of weighted edges
ε_k	Subset of the weighted edge set ε
ϱ	Sparsity level parameter.
$\mathbf{u}_n$	Set of real orthonormal eigenvectors
$\mathbf{U}$	Graph Fourier transform matrix
Υ	Set of index pairs of vertexes associated to weighted edge set ε
$\mathbf{V}$	Self-loop diagonal matrix of a graph $\mathcal{G}$
$\mathbf{v}_\tau$	i.i.d. Gaussian noise vector at time instant τ
$\mathbf{W}$	Weighted adjacency matrix of a graph $\mathcal{G}$
k	Subgraph index
τ	Time index
$\acute{\tau}$	Time gaps
i, j	Vertex indexes
$\mathbf{\Xi}$	Regularization matrix
$\mathbf{x}$	Graph signal
$\mathbf{x}_{\tau,k}$	Time-varying graph signal of a subgraph $\mathcal{G}_k$
$\mathbf{x}_\tau$	Time-varying graph signal
$\tilde{\mathbf{x}}$	Graph Fourier transform of a graph signal $\mathbf{x}$
$\mathbf{X}_\tau$	Collection of temporal samples of $\mathbf{x}_\tau$
$\hat{\mathbf{X}}_\tau$	Reconstructed graph signal of matrix $\mathbf{X}_\tau$
$\hat{\mathbf{x}}_\tau$	Reconstructed graph signal of vector $\mathbf{x}_\tau$ using the O-SGTD method
$\hat{\mathbf{X}}_\tau^{\text{SGSR}}$	Reconstructed graph signal of matrix $\mathbf{X}_\tau$ using the SGSR method
$\hat{\mathbf{X}}_\tau^{\text{GTTR}}$	Reconstructed graph signal of matrix $\mathbf{X}_\tau$ using the GTTR method
$\hat{\mathbf{X}}_\tau^{\text{SGTD}}$	Reconstructed graph signal of matrix $\mathbf{X}_\tau$ using the SGTD method
$\mathcal{X}_\tau$	Channel state vector
$\mathcal{X}'_{\tau,j}$	Set of historical UE dataset (HUD) samples
$\mathbf{Y}$	Received signal matrix
$\mathbf{y}_\tau$	Undersampled version of time-varying graph signal $\mathbf{x}_\tau$
$\mathbf{Y}_\tau$	Matrix that contains the unsampled entries of matrix $\mathbf{X}_\tau$

SUMMARY

1	**INTRODUCTION**	**20**
1.1	**Challenges in 5G scenarios**	**22**
1.1.1	***Establishing and maintaining a connection***	***22***
1.1.2	***Treatment of large amount of data***	***23***
1.2	**State-of-the-art**	**24**
1.2.1	***Related works on beam management in mmWave communications***	***24***
1.2.2	***Related works on sampling and reconstruction strategies based on GSP***	***28***
1.3	**Objectives and thesis structure**	**30**
1.4	**Scientific contributions**	**32**
2	**AN OVERVIEW OF GRAPH SIGNAL PROCESSING FUNDAMENTALS**	**33**
2.1	**Graphs**	**33**
2.2	**Signals on graphs**	**34**
2.2.1	***Time-varying graph signal***	***34***
2.3	**Laplacian of the graph**	**35**
2.3.1	***Spectral graph representation from Laplacian***	***36***
2.3.2	***Block-diagonal structure from Laplacian***	***37***
2.4	**Learning of the graph structure and connectivity**	**39**
2.4.1	***GGL learning method***	***41***
2.5	**Smoothness of time-varying graph signal**	**41**
2.5.1	***Temporal difference signal***	***42***
2.6	**Sampling via GSP**	**43**
2.6.1	***Random sampling (RS)***	***44***
2.7	**Reconstruction of time-varying graph signal**	**45**
2.7.1	***Batch reconstruction strategies***	***45***
2.7.1.1	***Smooth graph signal reconstruction***	***45***
2.7.1.2	***Graph-time Tikhonov reconstruction***	***46***
2.7.1.3	***Smooth graph-time difference reconstruction***	***47***
2.7.2	***Online reconstruction strategy***	***49***
3	**SUPERVISED LEARNING AND GRAPH SIGNAL PROCESSING STRATEGIES FOR BEAM TRACKING IN HIGHLY DIRECTIONAL MOBILE COMMUNICATIONS**	**50**
3.1	**Introduction**	**50**
3.2	**System model**	**51**
3.2.1	***Channel model***	***52***
3.2.2	***State evolution model***	***53***

3.2.3 *Beam training cycle* . . . 54
3.3 **Tracking with supervised learning** . . . 56
3.3.1 *K-NN tracking framework* . . . 57
3.4 **Graph signal processing strategies** . . . 59
3.4.1 *Graph signal model* . . . 59
3.4.2 *Learning of the graph from GGL method* . . . 60
3.4.3 *Time-varying graph signal sampling and reconstruction* . . . 61
3.4.4 *Time-varying graph signal reconstruction strategy* . . . 62
3.5 **Simulation results** . . . 63
3.5.1 *Baseline approaches* . . . 65
3.5.1.1 *Classical LS* . . . 65
3.5.1.2 *CS using OMP* . . . 65
3.5.1.3 *Kalman filter using EKF* . . . 65
3.5.2 *Complexity analysis* . . . 66
3.5.3 *K-NN-R tuning* . . . 67
3.5.4 *SNR influence analysis* . . . 71
3.5.5 *Standard deviation and UE speed influence analysis* . . . 72
3.5.6 *Channel tracking evaluation* . . . 74
3.5.7 *Channel prediction evaluation* . . . 76
3.6 **Chapter summary** . . . 80
4 **GRAPH SIGNAL PROCESSING FOR QOS PREDICTION IN CELLULAR V2X COMMUNICATIONS** . . . 81
4.1 **Introduction** . . . 81
4.1.1 *Contributions* . . . 82
4.2 **System model** . . . 83
4.3 **Graph representation of features** . . . 84
4.3.1 *Subgraph types* . . . 85
4.4 **Graph signal reconstruction** . . . 85
4.5 **Proposed graph signal sampling** . . . 87
4.6 **C-V2X dataset experiment** . . . 88
4.6.1 *Machine learning tool* . . . 90
4.7 **Simulation results** . . . 91
4.7.1 *Class imbalance treatment* . . . 93
4.7.2 *Reconstruction strategy evaluation* . . . 94
4.7.3 *SLS evaluation* . . . 96
4.7.3.1 *Baseline approaches* . . . 96
4.7.3.2 *Numerical results* . . . 98
4.7.4 *Latency prediction evaluation* . . . 100
4.8 **Chapter summary** . . . 106

5 CONCLUSIONS AND PERSPECTIVES 109
5.1 Conclusions 109
5.2 Future works 110
REFERENCES 113

1 INTRODUCTION

Mobile communications systems have revolutionized the way people communicate, combining communication and mobility. The recent years have witnessed a significant development in mobile technologies, such that the evolution of each technology has brought with it a new generation of mobile communications systems with remarkable advances related to performance and efficiency of communications.

Deployed in the eighties, the first generation (1G) used analog systems which served only to transmit voice with low security in communications. The second generation (2G) in the nineties evolved to digital systems capable of transporting voice and data at low speed (e.g., the short message service (SMS)), while improved the quality and security of communications. The third generation (3G) emerged in the year 2000, incorporating broadband speeds which allowed the use of mobile networks to access the Internet and its services (e.g., websites, emails, among others). Ten years later it starts the fourth generation (4G), which provides access to various telecommunications services combined with high mobility and/or high data rate applications, to support, for example, multimedia services.

The fifth generation (5G) mobile communications were introduced in response to the need for exponential growth in mobile data traffic and the provision of new generation real-time services and applications. In this context, three use cases are envisioned to work together to satisfy the requirements of new 5G networks and may be categorized as:

- **Enhanced mobile broadband (eMBB)**: This category aims to deliver high quality of service (QoS) of Internet access in environments under otherwise challenging or prohibitive conditions, providing a significant increment of data traffic and transmission rates. Specifically, eMBB may enhance broadband access in densely populated areas, may deliver wide-bandwidth connections to a massive number of users in environments with heavy data traffic and may provide continuous high-definition video streaming, mobile TV, among others.

- **Massive machine type communication (mMTC)**: This scenario involves a large number of low cost connected devices, in which the requirements mainly focus on long battery life and scalability. In particular, mMTC is a key enabler of the internet of things (IoT), which will reach a wide variety of areas and applications such as in transport and navigation, e-health, buildings management, smart cities and smart grids, smart agriculture, industrial automation and intelligent use of natural resources.

- **Ultra-reliable and low-latency communications (URLLC)**: This category targets real-time critical services and applications where latency and reliability of the communication are the highest priority requirements. For example, some URLLC use

cases involve applications in factory automation, remote medical surgery, smart grids, and cellular vehicular-to-everything (C-V2X).

The adoption of these services and applications of 5G systems will impose stricter technical requirements and different needs for each particular service, representing different QoS requirements. In general, 5G applications and services will have specific and diverse requirements, such as greater bandwidth, high reliability, lower latency, greater energy efficiency, critical synchronization, and the capacity to support a massive number of UEs with an intense data transfer from devices and guaranteeing a QoS at different levels [1].

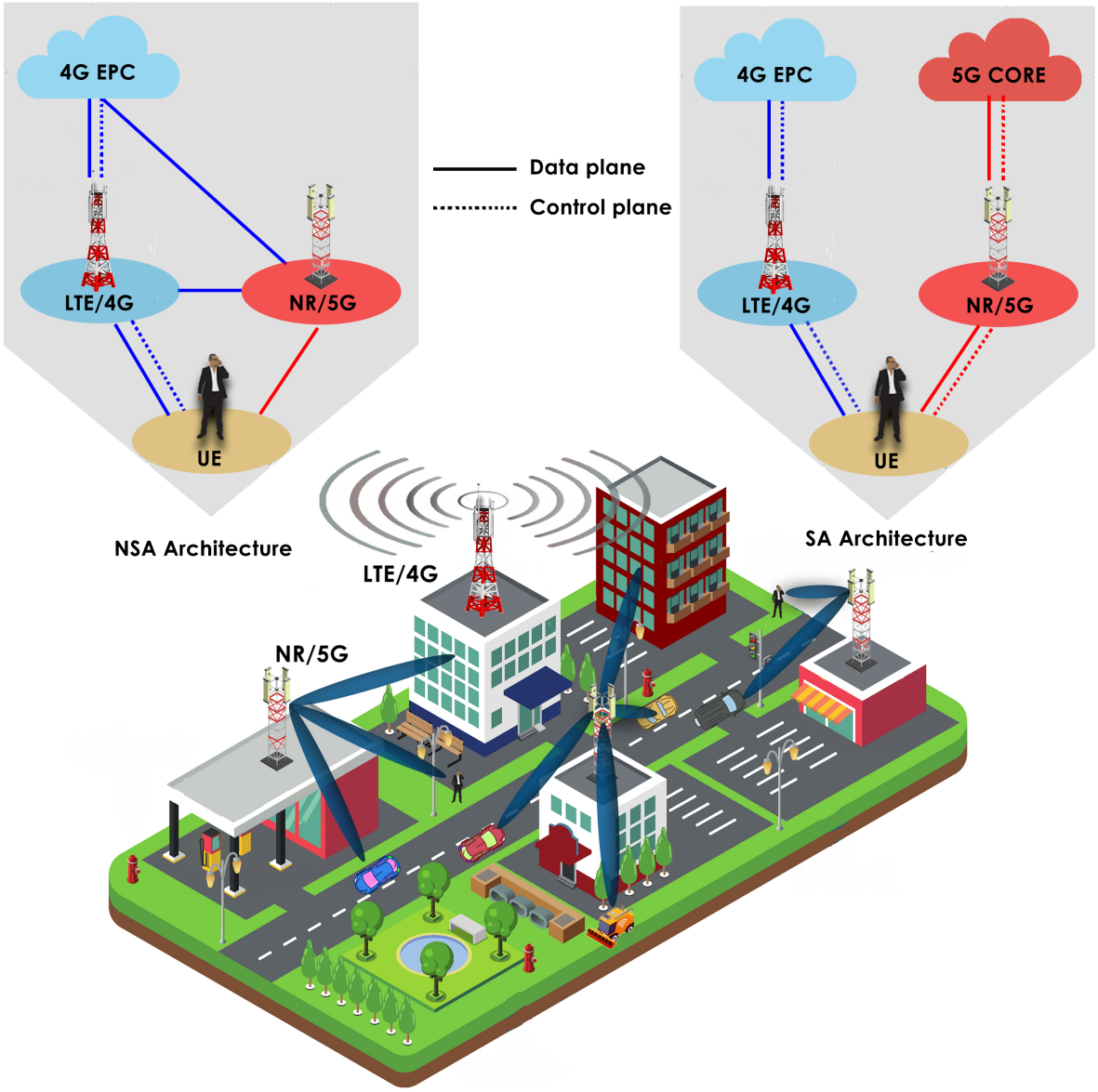
To deal with the above requirements, it has been considered the exploitation of millimeter wave (mmWave) spectrum and massive multiple-input-multiple-output (MIMO) systems [2] that are promising solutions for the future demands of 5G systems. Specifically, the use of the high frequency spectrum in 5G systems incorporates small mmWave cells (also called small-cells) with new radio (NR) technology, being standardized by the third generation partnership project (3GPP).

The mmWave BSs enable the use of a large number (in the order of hundreds) of antenna elements through massive MIMO, which allows highly directional communications through the use of narrow beams, also called pencil beams [3]. They provide coverage and services for mobile UEs (e.g. smart phones, vehicles, sensors, etc), which may also have several (in the order of dozens) antenna elements each one. The narrow beams offer a high array gain, which together with the large bandwidth available in the mmWave band, provide BSs with NR/5G technology with the ability to achieve higher data rates, support a large number of UEs and allow communications with greater energy efficiency. These make the mmWave spectrum in a vital technology to ensure the high throughput requirements in eMBB use case, and to develop high demanding application and services in URLLC.

Regarding the 5G networks architectures, two deployment options were defined in 3GPP specification 38 series Release 15 [4], viz. (i) non-standalone architecture (NSA) and, (ii) standalone architecture (SA). Delivered in December 2017, NSA anchors the control plane connections of NR/5G network to an enhanced 4G core (called evolved packet core (EPC)), then, it must be built over an existing 4G network using long term evolution (LTE) technology. On the other hand, SA was released six months later, allowing to connect the NR/5G directly to the 5G core, and does not need to rely on LTE/4G network for the control plane communications. In summary, 3GPP Release 15 [5] was the first full set of 5G standards, in which its initial specifications enabled NSA NR/5G systems integrated into previous-generation LTE/4G networks, and whose expansion managed to cover the completely independent operation of SA NR/5G systems.

Figure 1.1 illustrates an example of a typical 5G scenario, where BSs with NR/5G technology use multiple narrow beams to serve multiple UEs, while a BS with LTE/4G technology uses a wide beam. As illustrated, the NSA 5G architecture relies on 4G EPC, while SA architecture uses a 5G core.

Figure 1.1 – Typical 5G scenario with two technologies, LTE and NR, coexisting. Depending on the assumed 5G architecture, the NSA relies on 4G EPC, while SA uses a 5G core.



Source: Created by the author.

1.1 Challenges in 5G scenarios

1.1.1 Establishing and maintaining a connection

Despite high expectations, communications in 5G scenarios still face significant challenges due to, for example, higher path loss, large and small scale fading, lower diffraction and interference [6]. In particular, the use of the mmWave frequencies and highly directional communications through narrow beamforming make establishing and maintaining a connection between a BS and a UE a very difficult process [7]. Specifically, in these scenarios, any environmental change, such as device rotation, link blockage, or a slight beam misalignment, may cause a considerable drop in signal strength and, therefore, a potential link failure between a BS

and a UE. Moreover, highly directional communication limits the multipath diversity and makes communication susceptible to channel fluctuations.

In order to deal with these disadvantages, network resources and significant overheads are generally wasted on training (during initial access, e.g., in channel estimation) and beam tracking (while connected) procedures in order to determine the best directions of transmission/reception between a BS and a UE. Beam tracking and training mechanisms are necessary for good beamforming performance, which is essential for the high reliability of data reception. On the other hand, the low latency requirements of 5G systems make it difficult to optimize these mechanisms, because beamforming weights must be frequently calibrated to avoid channel aging as a result of high mobility [8, 9].

In this context, fast and efficient beam management mechanisms are of paramount importance for the efficient operation of 5G networks, maintaining high data rates, high reliability and/or low latency communication. Consequently, conventional beam tracking and training strategies which are mainly based on control theory (e.g., physical state transitions) are inadequate in satisfying strict reliability and/or latency constraints and low overheads of 5G systems [10]. Therefore, for the challenging requirements in 5G scenarios, it is not enough for beam management procedures to just track the beams. In fact, the mmWave BS must have the ability to predict the beams some time ahead, especially in high mobility scenarios.

In this thesis, it is proposed an adaptive beam tracking framework for highly directional communications, that exploits the samples in a historical UE dataset (HUD) to efficiently estimate and predict the channel state at the BS. Our proposal provides better performance in terms of NMSE and successful hit rate than three traditional/baseline prediction techniques.

1.1.2 Treatment of large amount of data

Regarding the 5G system's complexity, the amount of data generated, collected, stored and communicated represent an unprecedented challenge, which is commonly called mobile big data. Specifically, the amount of UE measurements and reports will drastically increase, and therefore, will burden feedback channels with a considerable amount of bandwidth-intensive data transfers [11].

These abundant sources of information provide countless opportunities to develop strategies capable of efficiently dealing with the problems described above and, thus, designing efficient and reliable 5G networks.

With the reduced costs of data storage devices and advancement of computational technologies, data-driven machine learning (ML) predictors seems to be the most promising approach in order to meet the stringent 5G requirements [12]. Generally, ML-based approaches represent an effective and versatile set of tools to explore and mine the enormous amounts of data generated and facilitate decision-making to improve performance on 5G networks. However, usually these strategies lead to expensive processing cost and intensive use of feedback channels for each BS-UE link. Thus, the challenge here is how to make an efficient use of feedback

resources and still provide accurate prediction features to efficiently deal with the problems described above, meeting the stringent requirements in the 5G communications, such as low latency and high reliability.

Graphical modeling of signals and their related signal processing tools, or simply graph signal processing (GSP), composes an emerging line of research for analyzing large volumes of data. It provides a flexible range of tools that usually outperform traditional signal processing techniques [13, 14]. The goal of GSP is to analyze and process large datasets and find any relation of similarity, dependency, physical proximity, or other properties among data samples which are modeled and inter-related over arbitrary graphs. Among many potential use cases of GSP tools, one is of great interest for UE measurement reporting: *sampling* and *reconstruction* of signals.

In this thesis, it is envisaged the use of graph-based sampling and reconstruction strategies in order to reduce the UE measurements feedback in the uplink and therefore, reduce the feedback channel burden. Specifically, the data or signals pre-processed by the GSP tools will be the input parameters of the ML-based prediction tasks for both a beam tracking problem in highly directional communications and a latency prediction problem in a C-V2X scenario.

Concerning the beam tracking problem, our proposal it is not much affected in terms of estimation/prediction error and exhibits a lower computational complexity when compared to the three baseline approaches. Regarding the latency prediction problem, our proposal outperforms some baseline greedy sampling strategies in terms of reconstruction error and feedback saving. Moreover, as for prediction accuracy, a small accuracy loss is observed when compared to full dataset availability case.

1.2 State-of-the-art

1.2.1 Related works on beam management in mmWave communications

The use of the mmWave band for communication in 5G systems presents several technical challenges, being one of the most important the management of narrow beams [15, 16, 17]. In this context, the use of efficient beam management mechanisms becomes a critical challenge in order to meet the stringent 5G requirements such as low latency in the URLLC use case. In general, it is possible to identify two beam management challenges in mmWave communications [18], as illustrated in Figure 1.2:

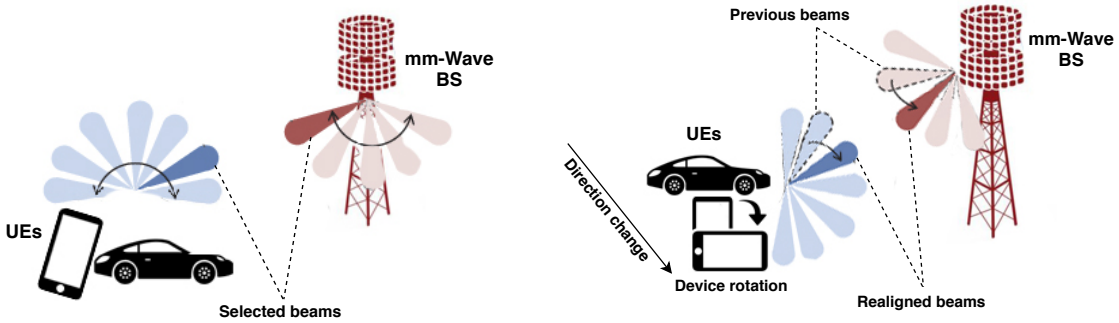
a) Initial beam alignment or beam training;

b) Beam tracking.

a) **Initial beam alignment or beam training**: The BS and the UEs (e.g., smart phones, vehicles, etc) initially align their beams to establish a mmWave link, as shown in Figure 1.2a. As described above, in mmWave communications beamforming with narrow beams

Figure 1.2 – Beam management challenges in mmWave communications.

(a) Initial beam alignment. (b) Beam tracking.



Source: Created by the author.

or directional antennas is generally used to provide high array gains. However, the use of narrow beamforming increases the overhead of the initial beam alignment, because the synchronization signals are transmitted by the BS repeatedly, one at a time, until "sweeping" all beamforming directions of interest. In general, as the directionality increases, the overhead of beam alignment increases due to the increase of beam candidates to be searched. The main challenge is that, because of the low-latency requirements in 5G scenarios, the initial beam alignment needs to be done quickly. This is even more challenging if high mobility terminals are considered.

b) **Beam tracking**: Even after the initial beam alignment, any environmental change due to UE mobility, direction change, device rotation, link blockages, among others, may cause a beam misalignment between the BS and UEs. To deal with misalignment of beams, it is not efficient to use the "sweeping" procedure, because the initial beam alignment does not use any temporal correlation of previously used beams. Therefore, a more efficient procedure is needed to reduce the beam tracking burden. A typical beam tracking procedure requires the transmitter (i.e., the BS) and receiver (i.e., the UEs) adjust their beams periodically (ideally at each beam coherence time[1]) to maintain the mmWave link, as shown in Figure 1.2b.

Several methods of beam alignment (including training and tracking in mmWave scenarios have been proposed in the literature, using well studied approaches such as, compressive sensing (CS), least square (LS) or matching pursuit (MP) techniques, see [20, 21, 22], and references therein. However, in scenarios with high overhead, these methods need a sufficiently large SNR or custom codebooks to efficiently handle the beam alignment with no prior channel state information (CSI).

Another widely used research line in the literature for beam alignment is the use of classical control theory, for example, mechanisms based on classical Kalman filters (KF). In this context, the pioneering work [23] showed that mmWave channel estimation can be performed efficiently by using the angular domain representation of channel, because most of the received energy is contained in only a few angular bins, which correspond to the angle of arrival (AoA) and the angle of departure (AoD). Both AoD and AoA can be estimated using a beam training

[1] Coherence time, refers to the time duration over which the beams remain aligned, depending on the beamwidth and the beam-pointing error [19].

procedure, and if one identifies the dominant angular bins one can make channel estimation without incurring in excessive computational complexity. In [24], a beam tracking method based on the AoD was proposed. This beam tracking technique exploits the continuous nature of time varying AoD for beam selection. The authors in [24] showed that the transmission of only two training beams is enough to track the time-varying AoD with good accuracy. However, this method relies on the AoD slow-varying assumption, being inadequate for non-stationary scenarios. In [25], a full scan of all possible beam combinations is used to create a measurement matrix to which the extended Kalman filter (EKF) is applied. A main drawback of this method is the requirement for a full scan, which makes it difficult to track fast-changing environments due to the long measurement time. In [15] it was proposed a beam tracking method which uses an estimation of only one pair of AoA and AoD measurement (i.e., not a full scan), tracking a particular propagation path defined by this pair. With a time-variant approach, an EKF was used as a tracking filter, which is applied at each step on the elements of the measurement matrix, in contrast with the technique proposed in [25]. The main drawback in the beam tracking technique proposed in [15] is the performance degradation when too narrow beams are assumed, due to beam misalignment during the measurement. In [26], the angular velocity was introduced in the EKF state estimation. Moreover, three received reference signals were used to jointly mitigate the influence of noise. However, the transmission of reference signals induces an extra burden on the communication channel. The unscented Kalman filters (UKF) and the particle filter (PF) are proposed in [27] and [28], respectively, to solve the beam tracking problem in highly directional non-linear systems. However, the disadvantage in this case is its high complexity. Moreover, these techniques use only a small number of pilots for beam tracking, leading to a limited combined filtering gain for angle estimation, and can therefore, suffer a serious performance loss, for example, in vehicle applications that require location tracking services with high precision.

However, all these methods described above based on KF assume simplified channel models, which are mainly applied to stationary channels. Scenarios with high mobility, such as C-V2X networks, pose severe challenges for an accurate training/tracking based on KF-like schemes, due to the difficulty in obtaining the state transition equations and prior information in non-stationary environments.

Using others strategies, the authors in [29] formulated a probabilistic optimization problem to model the temporal evolution of the mmWave channel under mobility and proposed a tracking strategy in which the information needed to update the steering directions (i.e., angular changes) was extracted from data packets. This allows that a spatial scan is not necessary during the transmission of data. However, this method needs assumptions on the prior distributions of the angular changes to operate correctly. Through adoption of the sparse beam space MIMO channel representation, the authors in [30] formulated the AoD and AoA tracking problem as a support recovery problem. To track the temporal evolution of AoD and AoA, it was proposed and modified an iterative approximate message passing (AMP) algorithm based on directional sounding beam design techniques. This modification, made it possible to counteract

the performance deterioration of the AMP algorithm, especially at the low signal-to-noise ratio (SNR) regime. The main drawback of this technique is the assumption that the previous estimate has small difference to the current AoA/AoD pair. Moreover, it does not exploit any dynamics of the channels. In [31] it was proposed *a priori* aided channel tracking scheme for mmWave massive MIMO systems. Specifically, physical direction between the BS and UE was used to obtain the prior information, i.e., the support, of sparse beamspace channel without channel estimation. Then, from this known support, the time-varying beamspace channels are tracked with low pilot overhead. However, this wireless transmission scheme lacks the ability to fully exploit time diversity, which limits its tracking performance in fast fading channels. The authors in [32] developed a low complexity beam tracking algorithm for mmWave systems that simultaneously optimizes the analog beamforming vector and estimates its directions. Based on stochastic approximation and recursive estimation with a control parameter, two variants of the beam tracking algorithm were implemented, i.e., for static and dynamic scenarios, respectively. However, this method assumes known channel coefficients, while both channel coefficient and beam direction might be unknown and time-varying in a real mobile system.

While the aforementioned works have a system modeling/signal processing perspective, the use of ML-based procedures has gained considerable attention lately to efficiently handle the beam management of mmWave systems. The main advantages of using ML-based solutions over the other beam management techniques described above are twofold: (i) they can capture the channel and environment dynamics well and, (ii) their relative low complexity of mathematical operations. The second advantage can be achieved if the complex mathematical operations needed by ML training step can be done in an offline phase at the BS side, not interfering in the dynamic estimation/tracking process. Once the training step has been completed, ML-based solutions typically require basic mathematical operations that can be performed easily in real-time applications.

The authors in [33] use neural networks (NNs) to predict the AoA at the UE side from observable features at the BS side. Specifically, UE pilot broadcasts are observed and processed by the BS to extract information about propagation channel features (e.g., received signal strength and relative path delay), which are used to predict the AoAs of the dominant propagation paths in the UE channels. However, this work focuses on single band and no beam-selection algorithm is developed. In [34], it was used an recurrent neural network (RNN) to track the AoA at the UE side when only the channel coefficient and previous estimates of AoAs are available. In particular, the authors proposes a RNN with a modified cost function and a long short term memory (LSTM) network architecture for the AoA tracking problem. However, this method needs to create features and label sets before proceeding with the training and testing steps, which represents a challenge for beam tracking problem in which the true label depends on the beam-steering decision of the UE in a previous time step. An ML-based beam prediction scheme was proposed in [10], where four ML tools and situational awareness were combined to learn the beam information, including power and optimal beam index from past observations.

The main drawback of this method is the need for the UE location information to train the ML algorithm, which incurs extra system overhead.

In contrast to prior ML-based works, in Chapter 3 the problem of beam tracking is tackled as a channel prediction tool, to obtain the channel matrix some time instants ahead from UE measurements (collected only throughout an observed temporal window) and taking advantage of the availability of a historical UE dataset (HUD). Another advantage of our ML-based framework is the feedback saving, in which just sampled version of the whole beam space measurement will be reported by the UEs in order to predict the channel at the BS. Moreover, the bulk of the processing is done in the BS, and the HUD generation is performed in an offline phase from the sampled version of UE measurements, reducing the computational complexity and not interfering in the dynamic tracking process. To the best of our knowledge, there has been no case in the literature that improves the robustness of beam tracking by such manner.

1.2.2 Related works on sampling and reconstruction strategies based on GSP

As mentioned before, graph signal processing (GSP) compose a potentially relevant line of research for analyzing large volumes of data. The goal of GSP is to analyze signals, defined over an irregular discrete domain, that are modeled over graphs. Graphs and its connections are built by finding any relation of similarity, dependency, physical proximity, or other properties among data samples.

However, in most real datasets the structural relations among data samples are usually unknown. Thus, it is necessary to implement graph learning strategies to obtain the graph topology and the edge weights. In [35], a framework for learning and estimating graphs from raw data was proposed. Specifically, three graph learning approaches are introduced, aiming at the estimation of graph Laplacian matrices from some observed data and under given structural constraints (e.g., graph connectivity and sparsity level). In [36], new contributions toward the development of an adaptive graph signal processing framework was provided, extending classical adaptive processing strategies for the analysis of signals defined over time-varying graphs. Two adaptive estimation methods for adaptive learning of graph signals were introduced, i.e., least mean square (LMS) and recursive least square (RLS). Both methods are based on a probabilistic sampling mechanism over the graph.

In the literature, a large number of applications is found, in which observed data can be modeled as signals defined over graphs, aimed at solving problems in diverse areas such as climate analysis [37, 38], biological networks [39, 40], statistical learning [41, 42] or sensor networks [43, 44].

Among many potential use cases of GSP tools, one that is of great interest is *sampling* and *reconstruction* of graph signals. Not surprisingly, many state-of-the-art works on sampling and reconstruction strategies based on GSP have been proposed in the literature. Some of these strategies are: natural neighbor interpolation [45], low-rank matrix completion [46], the Tikhonov regularization method [47], the smooth graph signal reconstruction method [48] and the smooth

graph-time difference [49].

In [45] it was proposed a grid based method to resample the data on a uniform grid, namely natural neighbor (Sibson) interpolation. However, this method is computationally expensive, especially when applied to higher-dimensional data. In [46], a matrix completion method was proposed to solve the nuclear norm minimization of a matrix by Bregman iterative algorithms. The main drawback of this method is the need of a sufficient number of observed entries, which leads to high computational cost when the matrix size is large. In [47], the Tikhonov regularization is used in the context of regularization by graph Laplacians to solve a problem of labeling a partially labeled graph. The authors in [48] formulated the reconstruction of smooth graph signals as a convex optimization problem. Specifically, it was proposed a cost function in which the signals on a subset of graph vertexes are predicted based on the smoothness of known signal values from other vertexes. In [49], the authors demonstrated that the temporal differences of real-world time-varying graph signals usually exhibit better smoothness on the graph representation than the signals themselves, in contrast with the technique proposed in [48]. According to this result, a batch reconstruction method of time-varying graph signals was proposed, by exploiting such smoothness. Driven by practical applications faced with real-time requirements, an online distributed strategy was proposed, based on the local properties of the temporal difference operator and the graph Laplacian matrix. Both strategies exhibited better performance than the other methods mentioned above.

Time-variance of real-world signals, which include those inherent to mobile communication systems, brings an interesting motivation to exploit signal processing techniques that deal with system dynamics. In this context, applying GSP tools to mobile communication systems has gained considerable attention lately, where applications span different use cases, such as interference coordination and resource allocation [50, 51, 52, 53], anomaly detection [54] and estimation [55].

In [50], it was proposed graph-based dynamic frequency reuse method on Femtocell Networks. A UE-oriented graph based frequency allocation algorithm is proposed in [51], aiming at coordinating the downlink interference in the densely deployed femtocell network. In [52], it was proposed a graph-based algorithm, in which mode selection jointly with resource allocation are used to maximize the system sum rate on a sparse code multiple access (SCMA) enabled uplink cellular network. The authors in [53] proposed a coloring-based cluster resource allocation algorithm based on the graph theory in order to improve spectral efficiency and coordinate the downlink interference for ultra dense network (UDN). The authors in [54] proposed a graph-based sensors anomaly detection algorithm on IoT sensor networks. The graph is built based on the nonlinear polynomial graph filter (NPGF), in which the irregular spatial information of the IoT sensors is utilized to learn the adjacency matrix from the distances among different sensors, and then, detect the existence of anomaly and localizing the faulty sensors. A distributed algorithm to estimate the singular values of the adjacency matrix for large-scale mobile communication networks is proposed in [55], aiming at network graph representation. Specifically, a UKF-based

model is established in order to track for real-time the estimation of graph eigenspectra.

In contrast to the aforementioned graph-based works for mobile communication systems, in this thesis it is envisaged the use of GSP tools to reduce the UE measurements reporting, using sampling and reconstruction techniques of time-varying signals. Specifically, in Chapters 3 and 4 are explored sampling and reconstruction tools in order to reduce the feedback channel burden and, therefore, counteracting the measurement process challenges.

The graph-based tools are used jointly with the supervised learning techniques, in order to solve efficiently and with less complexity the two prediction problems mentioned above, and, in which the data or signals pre-processed by the GSP tools will be the input parameters of the supervised learning techniques. To the best of our knowledge, no study has been performed to improve the feedback saving on mobile communication systems using sampling and reconstruction tools based on GSP.

1.3 Objectives and thesis structure

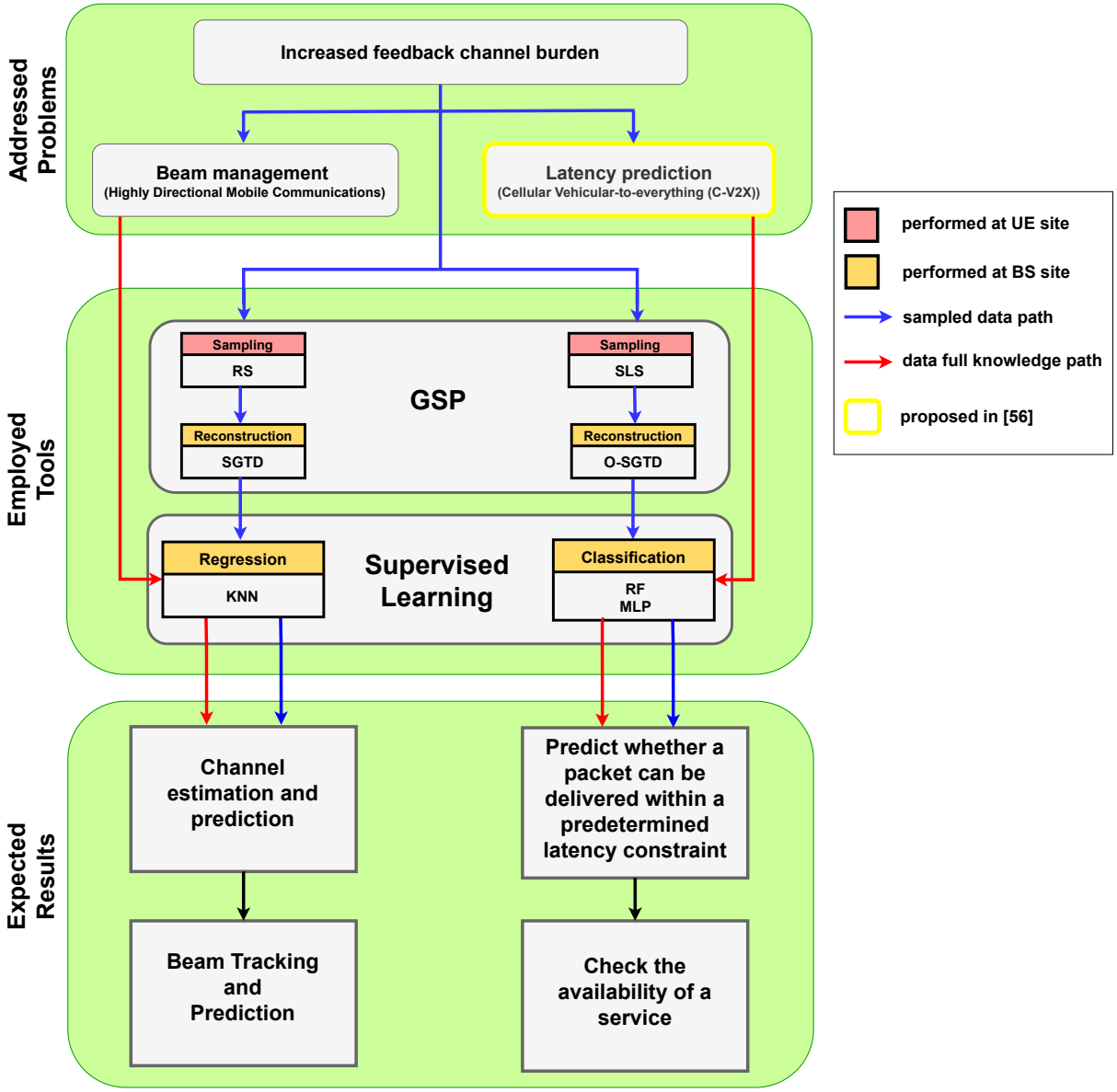
Considering the issues discussed in the previous sections, the main objective of this thesis is to study solutions for prediction problems in mobile communications. These solutions are based on GSP jointly with supervised learning techniques to address two relevant problems of mobile communication systems, which are: (i) beam management and, (ii) increased feedback channel burden. Specifically, sampling and reconstruction tools based on GSP are exploited in order to reduce the feedback channel burden, which is used to report the UE measurements that feed the ML-based predictors.

The thesis structure is presented in Figure 1.3 and is described in the following.

- **Chapter 2 - Graph Signal Processing Overview:** reviews basic graph-theoretic definitions, as well as, describes some of the tools and concepts that will be used throughout this thesis. First, it is introduced the concepts of graphs, signals on graphs and Laplacian of the graph. Next, it is described a graph learning method to obtain optimally the graph structure and connectivity from real measured datasets. Finally, some graph-based sampling and reconstruction techniques (namely, batch and online reconstruction strategies) that will be used to reduce the feedback channel burden in next chapters are explained.

- **Chapter 3 - Supervised Learning and Graph Signal Processing Strategies for Beam Tracking in Highly Directional Mobile Communications:** proposes an adaptive beam tracking framework for highly directional mobile communications, which exploits the samples in the HUD to efficiently estimate and predict the channel state at the BS. More specifically, a supervised learning method, namely K-nearest neighbors (K-NN), is evaluated as a means of finding the K most similar historical observation samples in the HUD to predict the corresponding channel state. Moreover, a sampling and batch reconstruction strategy based on GSP, called K-nearest

Figure 1.3 – Thesis structure.



Source: Created by the author.

neighbors with reconstruction (K-NN-R), is introduced as a pre-processing stage to the KNN algorithm, in order to reduce the beam search space during the beam measurement stage, which allows a more efficient usage of the feedback channel. Specifically, it is assumed a random sampling (RS) mechanism and the smooth graph-time difference (SGTD) batch reconstruction strategy as graph-based tools. Finally, some simulation results are exposed, which illustrate a better estimation and prediction performance in terms of complexity, NMSE and successful hit rate.

- **Chapter 4 - Graph Signal Processing for QoS Prediction in Cellular V2X Communications:** turns its attention to QoS predictability concept in a C-V2X scenario. Latency prediction is modeled as a binary classification problem, in which the goal is to predict whether a packet can be delivered within a predetermined latency constraint. However, it is assumed that the uplink feedback channel is limited. To

reduce the need for feeding UE measurements back to the network, sampled and reconstruction graph-based strategies were employed to i) model feature vectors as graph signals, ii) downsample feature vectors along the entries related to UE measurements, and iii) reconstruct the missing entries at the network side. A sampling strategy is proposed, namely smart Laplacian sampling (SLS), that i) exploits block-diagonal structure of the graph Laplacian and ii) reduces the UE measurement feedback. Moreover, an online reconstruction strategy is adopted, namely O-SGTD. Finally, a performance evaluation of the proposal is conducted, in terms of number of unsampled UE measurements, reconstruction error and prediction accuracy.

- **Chapter 5 - Conclusions:** the main conclusions of this thesis are summarized, and the possible avenues for further research are discussed.

1.4 Scientific contributions

Currently, the content and contributions of this thesis have been partially published and submitted with the following bibliographic information:

Published Journal Papers

- ORTEGA, Y. R.; GUERREIRO, I. M.; HUI, D.; CAVALCANTE, C. C.; CAVALCANTI, F. R. P. Supervised learning and graph signal processing strategies for beam tracking in highly directional mobile communications. **Transactions on Emerging Telecommunications Technologies**, v. 30, n. 9, p. 825–841, July 2019. ISSN 1932-4553. DOI: `10.1002/ett.3687`

Submitted Journal Papers

- ORTEGA, Y. R.; GUERREIRO, I. M.; MOREIRA, D. C.; CAVALCANTE, C. C.; CAVALCANTI, F. R. P. Graph Signal Processing for QoS Prediction in Cellular V2X Communications. **Transactions on Vehicular Technology**, *Under Review*, Aug. 2020

It is worth mentioning that this thesis was developed under the context of Ericsson/UFC technical cooperation project:

- UFC.46 - Network Assisted Intelligent Vehicle-to-Everything Communications (NAIVE), November/2018 - October/2020

in which a number of four technical reports have been delivered.

2 AN OVERVIEW OF GRAPH SIGNAL PROCESSING FUNDAMENTALS

GSP deals with signals which feature an underlying complex and irregular structure. GSP seeks to model that underlying structure by a graph and signals by graph signals, generalizing concepts and tools from classical discrete signal processing to graph signal processing. The intrinsic geometric structure and connectivity of the underlying graph is taken into account in order to efficiently represent and process graph signals. Some signal characteristics, such as smoothness, depend on the irregular topology of the graph on which the signal is defined.

In the last years, much effort has been dedicated to design new graph-based tools and algorithms that can handle efficiently the new challenges that arise from the irregular structure of communications networks and in other GSP applications. These graph tools are generally based on a combination of computational harmonic analysis with algebraic and spectral graph theoretical concepts [58].

For the sake of completeness, in this chapter some graph-theoretic definitions and some of the tools and concepts that will be employed throughout this thesis are provided. Although not all of them are going to be used in our evaluation scenarios we believe they are of interest to place the tool on the proper context. First, some basic definitions and notation for graphs and signals on graphs are given. This includes the concept of Laplacian matrix of a graph and the smoothness of graph signals, quantified through the Laplacian quadratic term. Next, the use of graph-based signal processing tools in different applications is reviewed. In particular, it is focused on learning of the graph structure and connectivity from observed signals, relying on a tool of learning/estimation of graph Laplacian matrices. Finally, the time-varying graph signal sampling and reconstruction techniques are briefly reviewed.

This overview does not intend to exhaust the graph theory topic and its applications. A greater in-depth analysis and much wider view of the topic can be found at some relevant classic references, such as [13, 59, 60, 61], among others.

2.1 Graphs

Graphs can be represented by different types of matrices, e.g., adjacency and Laplacian matrices. The choice of a matrix type depends on properties of the desired graph, modeling assumptions and the application of interest.

Structured data can be modeled as an N-vertex, undirected, connected, and weighted [60] graph $\mathcal{G} = (\nu, \varepsilon, \mathbf{W})$, where $\nu = \{v_1, v_2, \ldots, v_N\}$ is the set of N vertexes and ε is a set of weighted edges e. In addition, $\mathbf{W}$ is the weighted adjacency matrix, which denotes the collection of all the nonnegative weights w_{ij}. A nonnegative weight w_{ij} indicates the strength of the pairwise relation between two distinct vertexes i and j in ν. More specifically, if there is a link from vertex j to vertex i with $i, j \in \nu$, then $w_{ij} > 0$, otherwise $w_{ij} = 0$. Note that $\mathbf{W}$ is an $N \times N$ symmetric

matrix, i.e., $[\mathbf{W}]_{i,j} = [\mathbf{W}]_{j,i}$, where $[\cdot]_{i,j}$ stands for the (i, j)-th entry of a matrix. Moreover, the connectivity adjacency matrix $\mathbf{A}$ is also an $N \times N$ matrix, in which $[\mathbf{A}]_{j,i} = 1$ if $[\mathbf{W}]_{j,i} \neq 0$, and $[\mathbf{A}]_{j,i} = 0$ if $[\mathbf{W}]_{j,i} = 0$.

Furthermore, a subgraph $\mathcal{G}_k$ of a graph $\mathcal{G}$ is a graph whose vertex set ν_k is a subset of the vertex set ν, i.e., $\nu_k \subseteq \nu$, and whose edge set ε_k is a subset of the edge set ε, i.e., $\varepsilon_k \subseteq \varepsilon$. Besides, an isolated vertex is a vertex with zero degree; that is, a vertex that is not an endpoint of any edge. In this thesis, the isolated vertexes are called *trivial* subgraphs and give a notion of independence or not affinity of them with the others vertexes in ν set.

2.2 Signals on graphs

Examples of graph signals exist in numerous engineering and science fields, and are modeled depending on the problem to be analyzed. For example, time series, such as acoustic data and financial data, may be indexed by time stamps and are signals supported on a directed cyclic graph [62]. Each vertex corresponds to a time stamp, and each value is related to the value at the previous time stamp, reflecting the causality of a time series. This relation is asymmetric, hence all edges are directed. Images are indexed by pixels and are signals supported on a lattice graph [63]. That is, each vertex corresponds to a pixel, and each pixel value (intensity) is related to the values of the four adjacent pixels. This relation is symmetric, hence all edges are undirected, with possible exceptions of boundary vertexes that may have directed edges, depending on boundary conditions.

In communications networks, one may be interested in analyzing attributes describing the users or measurements collected by them over time. Those attributes or measurements are signals supported on relational graphs, where vertexes correspond to users and edges correspond to the relationship between users and the variation of the signal on the graph depends on the graph connectivity.

A graph signal $\mathbf{x}$ in the vertex domain is a real-valued function defined on the vertexes of the graph $\mathcal{G}$, such that $\mathbf{x}(i)$ is the value of the function at vertex $v_i \in \nu$. In other words, assuming a one-to-one mapping between the elements of ν and the entries of $\mathbf{x}$, a graph signal is defined as a function $\mathbf{x} : \nu \rightarrow \mathbb{R}$ that assigns a scalar value to each vertex of $\mathcal{G}$. Given a fixed ordering of vertexes, a sign coefficient is assigned to each vertex, and a graph signal can be written as a vector,

$$\mathbf{x} = [x_1 \quad x_2 \quad \dots \quad x_N]^T \in \mathbb{R}^N, \tag{2.1}$$

where x_i is the signal coefficient corresponding to the vertex i.

2.2.1 Time-varying graph signal

In time series data, one may be interested in the variation of the signal on the graph. Then, time-varying graph signals which can be represented as functions of time supported on the

nodes of a fixed graph are considered. In other words, the time-varying graph signals are scalar functions on $\mathcal{G}$, which represent vector-valued sequences of vector $\mathbf{x}$ indexed by time τ, such that $\{\mathbf{x}_\tau : \nu \rightarrow \mathbb{R}\}_\tau$. Thus, the structure of a time-varying graph signal $\mathbf{x}_\tau$ may be expressed by a matrix $\mathbf{X}_\tau$, that represents a collection of temporal samples of $\mathbf{x}_\tau$ from instant 1 to N_τ as

$$\mathbf{X}_\tau = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \dots \quad \mathbf{x}_{N_\tau}] \in \mathbb{R}^{N \times N_\tau}, \tag{2.2}$$

Note that, each row of $\mathbf{X}_\tau$ represents vertexes of the graph signal $\mathbf{x}$ and each column denotes the graph signal evaluated at different time instants τ during a temporal window N_τ.

2.3 Laplacian of the graph

The Laplacian matrix $\mathbf{L}$ of a graph will play a fundamental role in the tools addressed in this thesis. As it will be seen, this matrix has intimate connections to a variety of graph-theoretic notions. Much graph information is embedded on the Laplacian matrix, e.g., structure, connectivity, sparsity and affinity relations [64]. Thus, obtaining $\mathbf{L}$ ends up being an important step toward exploiting the graph structure for graph signal processing tools, such as, sampling/reconstruction strategies.

To define the Laplacian matrix, let us consider the N-vertex, undirected, connected, and weighted graph $\mathcal{G} = (\nu, \varepsilon, \mathbf{W})$, defined above. Now, the Laplacian matrix $\mathbf{L} \in \mathbb{R}^{N \times N}$ of the graph $\mathcal{G}$ is defined as

$$\mathbf{L} = \mathbf{D} - \mathbf{W}, \tag{2.3}$$

where $\mathbf{D} \in \mathbb{R}^{N \times N}$ denotes the diagonal degree matrix with $[\mathbf{D}]_{i,i} = \sum_{j, j \neq i} w_{ij}$. That is, its i-th diagonal element is the sum of the weights of all edges connected to vertex i, namely vertex or node degree.

The definition in Equation (2.3) is widely used in the literature and is known as combinatorial graph Laplacian (CGL) [61], [65], which represents graphs with zero vertex weights, i.e., graphs with no self-loops. Then, the definition in Equation (2.3) for a CGL matrix satisfies the following conditions

$$\mathbf{L}_{CGL} = \left\{ \mathbf{L} | \mathbf{L} \succeq 0, \mathbf{L}_{ij} \leq 0 \quad \text{for} \quad i \neq j, \mathbf{L1} = \mathbf{0} \right\}, \tag{2.4}$$

where $\succeq$ stands for the positive semidefinite matrix operator, $\mathbf{1}$ is a column vector of ones and $\mathbf{0}$ is a column vector of zeros. Note that, the $\mathbf{L1} = \mathbf{0}$ constraint is a mandatory condition to $\mathbf{L}$ be an $\mathbf{L}_{CGL}$ matrix. One can easily check that the entries of the $\mathbf{L}_{CGL}$ matrix are given by

$$[\mathbf{L}]_{i,j} = \begin{cases} \sum_{j, j \neq i} w_{ij}, & \text{if } i = j, \\ -w_{ij}, & \text{if there is a link from vertex } j \text{ to vertex } i, \\ 0, & \text{otherwise.} \end{cases} \tag{2.5}$$

where the off-diagonal entries of the $\mathbf{L}_{CGL}$ matrix are nonpositive and the main diagonal entries match to the values of the matrix $\mathbf{D}$.

Although several works and applications mainly focus on CGL and their normalized version, there are recent studies in which the use of generalized graph Laplacian (GGL) matrices (i.e., graphs with nonzero vertex weights) are shown to be useful [66], [67] and [68]. Specifically, GGL matrices are proposed to model image [69] and time series [70] signals. Moreover, in [71] it was discussed several potential machine learning applications using GGL.

In this context, a GGL matrix $\mathbf{L}_{GGL}$ can be defined as [35]

$$\mathbf{L}_{GGL} = \mathbf{D} - \mathbf{W} + \mathbf{V}, \tag{2.6}$$

where $\mathbf{V} \in \mathbb{R}^{N \times N}$ is the self-loop diagonal matrix, with $[\mathbf{V}]_{i,i} = f_v(v_i)$, in which $f_v(v_i)$ is the vertex (self-loop) weight function. That is, it is the matrix whose the i-th diagonal element is the real-valued weight of vertex v_i, namely self-loop weight. By the definition in Equation (2.6), the GGL matrix can also be written as

$$\mathbf{L}_{GGL} = \left\{ \mathbf{L} | \mathbf{L} \geq 0, \mathbf{L}_{ij} \leq 0 \quad \text{for} \quad i \neq j \right\}. \tag{2.7}$$

As observed in Equation (2.7) and by the definition of a generalized Laplacian matrix in Equation (2.6) with nonnegative edge weights, the $\mathbf{L}_{GGL}$ will be a real symmetric positive semidefinite matrix, in which the off-diagonal entries are nonpositive and the main diagonal entries match to the values of the matrix $\mathbf{D}$ plus the self-loop weights in that same position, adjusting all vertexes degrees. Due to this fact, the $\mathbf{L}_{GGL}$ matrix no longer has the $\mathbf{L1} = \mathbf{0}$ constraint.

2.3.1 Spectral graph representation from Laplacian

The fundamental analogy between traditional signal processing and graph signal processing is established through the spectral graph theory [61]. In particular, the generalization of the classical discrete Fourier transform (DFT) to graph settings (i.e., graph Fourier transform (GFT)) has been established through the eigenvectors and the eigenvalues of the graph Laplacian matrix, which carry a notion of frequency for graph signals. The spectral properties of matrix $\mathbf{L}$, that is, its eigenvalues and eigenvectors structure, are of particular importance to study the spectral graph theory.

The Laplacian matrix $\mathbf{L}$, regardless of its type (i.e., $\mathbf{L}_{CGL}$ or $\mathbf{L}_{GGL}$), is a real symmetric positive semidefinite matrix that has a complete set of real orthonormal eigenvectors $\mathbf{U} = \{\mathbf{u}_1 \quad \mathbf{u}_2 \quad \dots \quad \mathbf{u}_N\}$ with corresponding non-negative eigenvalues $\lambda_1 \leq \lambda_2 \leq \dots \lambda_N$. Then, its full eigen-decomposition can be written as [72]

$$\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T, \tag{2.8}$$

where $\mathbf{\Lambda}$ is the diagonal matrix whose diagonal elements are the set of real eigenvalues $\lambda_n \geq 0$ for $n = 1, \dots, N$. The matrix $\mathbf{U}$ is called GFT matrix and its eigenvectors are therefore considered to

represent a Fourier basis for graph signals. In $\mathbf{U}$, the number of eigenvectors (or graph frequency components) is the number of graph nodes N. The eigenvectors provide a natural way to order the GFT basis vectors in terms of frequency (i.e., the variations of their values on the graph). Further, the set of eigenvalues in $\mathbf{\Lambda}$ represents graph frequencies and is also named as graph sorted spectrum, satisfying $\mathbf{L}\mathbf{u}_n = \lambda_n \mathbf{u}_n$. Specifically, the graph Laplacian eigenvectors associated with small eigenvalues correspond to signals that vary slowly across the graph, hence they can be associated with the notion of low frequency. In other words, if two vertexes are connected by an edge with a large weight, the values of the low frequency eigenvectors at those locations are likely to be similar. The eigenvectors associated with larger eigenvalues take values that change more rapidly on the graph; they are more likely to have dissimilar values on vertexes connected by an edge with high weight. Note that, according to the $\mathbf{L1} = \mathbf{0}$ constraint, if $\mathbf{L}$ is a $\mathbf{L}_{CGL}$ matrix, the smallest eigenvalue is always zero, and the corresponding eigenvector is a constant vector. Moreover, the largest eigenvalue depends on the maximum degree of the graph.

Figure 2.1 illustrates an example of different graph Laplacian eigenvectors for a random sensor graph. Specifically, Figure 2.1a shows an arbitrary sensor graph with $N = 20$ vertexes, in which the signal on the graph is generated randomly from a Gaussian distribution, with zero mean and standard deviation of 5. Figures 2.1b, 2.1c, 2.1d and 2.1e show some Laplacian eigenvectors of the graph represented in Figure 2.1. For connected graphs, the Laplacian eigenvector $\mathbf{u}_1$ associated with the eigenvalue λ_1 is constant and equal to $\frac{1}{\sqrt{N}}$ at each vertex.

Since the expansion into the eigenbasis is given by the multiplication with the transpose of Fourier basis matrix, the GFT of a graph signal $\mathbf{x}$ can be defined as [58]

$$\tilde{\mathbf{x}} = [\tilde{x}_1 \quad \tilde{x}_2 \quad \dots \quad \tilde{x}_N] = \mathbf{U}^T\mathbf{x}, \tag{2.9}$$

Filtering a graph signal $\mathbf{x}$ with a graph filter $h(\mathbf{L})$ corresponds to element-wise multiplication (Hadamard product) in the frequency domain $[\lambda]$ by a scalar function ($h : [0 \quad \lambda_{max}]$), referred to as the graph frequency response. When a graph filter $h(\mathbf{L})$ is applied to $\mathbf{x}$, the output is [73]

$$h(\mathbf{L})\mathbf{x} \triangleq \mathrm{GFT}^{-1}\{h(\mathbf{\Lambda}) \odot \mathrm{GFT}\{\mathrm{x}\}\} = \mathbf{U}h(\mathbf{\Lambda})\mathbf{U}^T\mathbf{x}, \tag{2.10}$$

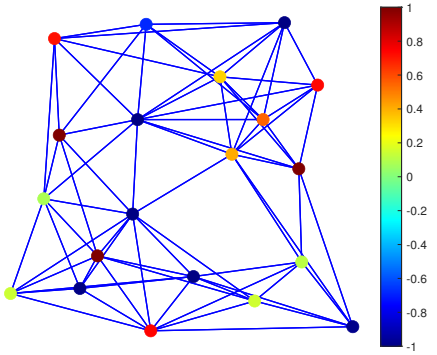
where $(\odot)$ is the Hadamard product and GFT^{-1} denotes the inverse graph Fourier transform (IGFT) defined by $\mathbf{x} = \tilde{x}_1\mathbf{u}_1 + \tilde{x}_2\mathbf{u}_2 + \quad \dots \quad + \tilde{x}_N\mathbf{u}_N = \mathbf{U}\tilde{\mathbf{x}}$. The function h in Equation (2.10) is applied to each diagonal entry of matrix $\mathbf{\Lambda}$.

2.3.2 Block-diagonal structure from Laplacian

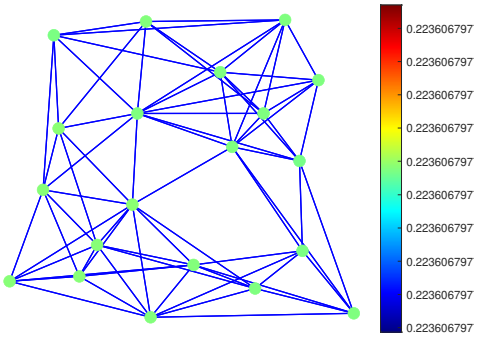
From the Laplacian matrix, the underlying graph may have multiple connected components and even isolated vertexes. That is, the graph itself may be a composition of K disjoint connected subgraphs and several isolated vertexes.

Figure 2.1 – Different graph Laplacian eigenvectors for a random sensor graph with 20 vertexes. Graph vertex colors in (a) represent graph signal values, while in (b)-(e) they correspond to eigenvector values.

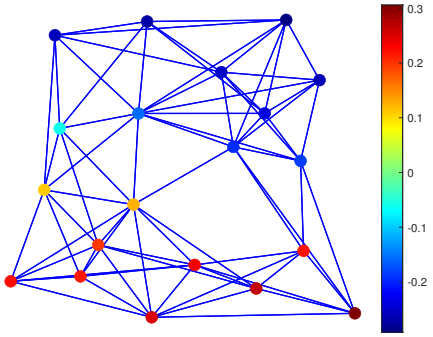
(a) Random signal defined on 20 vertexes.



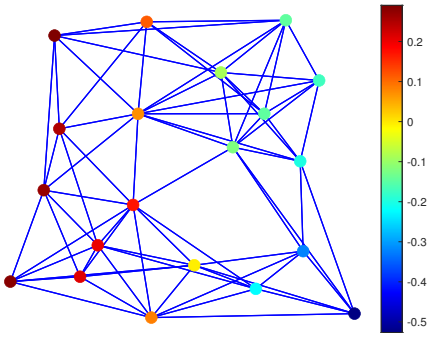
(b) Eigenvector $\mathbf{u}_1$.



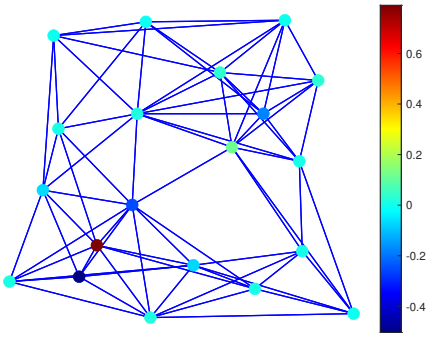
(c) Eigenvector $\mathbf{u}_2$.



(d) Eigenvector $\mathbf{u}_3$.



(e) Eigenvector $\mathbf{u}_{20}$.



Source: Created by the author.

In this context, the multiplicity of the zero eigenvalues of $\mathbf{L}$ is equal to the number of connected components K. When the graph $\mathcal{G}$ is fully-connected there is only one zero eigenvalue [59]. That is, an edge connects each pair of vertexes in the graph and $K = 1$. However, the graph $\mathcal{G}$ has K connected components if and only if [74]

$$\lambda_n \begin{cases} = 0, & \text{for} \quad n = 1, \ldots, K, , \\ > 0, & \text{for} \quad n = K+1, \ldots, N. \end{cases} \tag{2.11}$$

where λ_n, for $n = 1, \ldots, N$, are in ascending order.

This K-block structure leads to a block-diagonal graph Laplacian, i.e.,

$$\mathbf{L} = \begin{bmatrix} \mathbf{L}_1 & & \\ & \ddots & \\ & & \mathbf{L}_K \end{bmatrix}, \tag{2.12}$$

allowing matrix $\mathbf{L}$ to be split into K sub-matrices $\mathbf{L}_k$, in which each matrix $\mathbf{L}_k$ has one zero eigenvalue and the spectrum of $\mathbf{L}$ is given by the union of the spectrum of each $\mathbf{L}_k$.

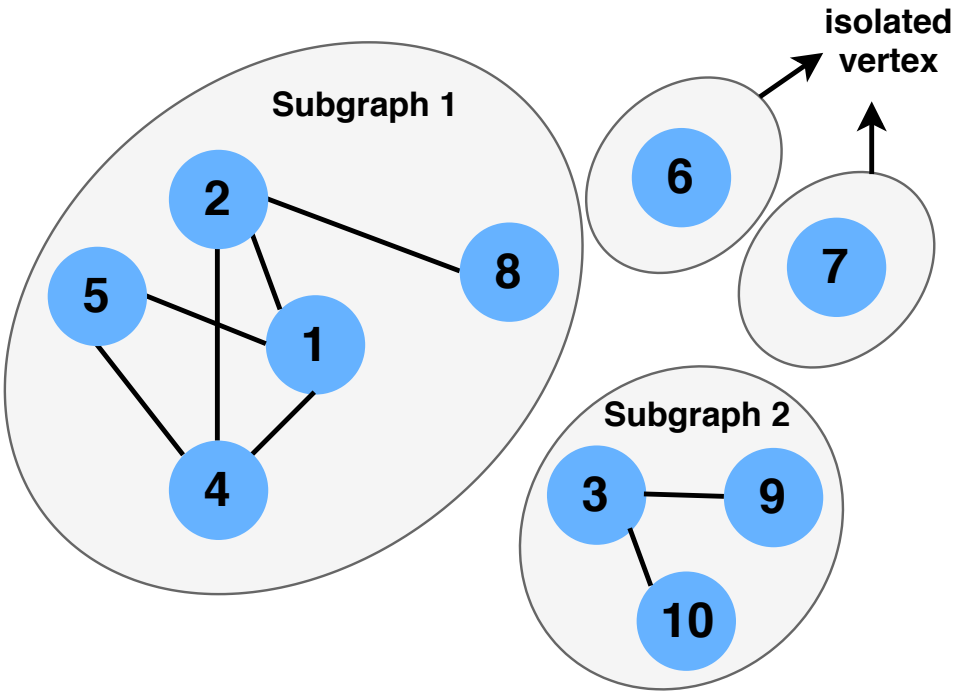
Given the block-diagonal structure of $\mathbf{L}$, it is straightforward to realize that the time-varying graph signal represented by vector $\mathbf{x}_\tau$ can also be split into K parts as

$$\mathbf{x}_\tau = \begin{bmatrix} \mathbf{x}_{\tau,1}^T & \cdots & \mathbf{x}_{\tau,K}^T \end{bmatrix}^T, \tag{2.13}$$

where $\mathbf{x}_{\tau,k}$ is the signal over subgraph $\mathcal{G}_k$.

Figure 2.2 illustrates an example of graph with ten vertexes. In this example, there are two connected components, i.e., two connected subgraphs, one comprising five vertexes and the other having three vertexes, and two isolated vertexes or trivial subgraphs.

Figure 2.2 – Example of graph with multiple connected components.



Source: Created by the author.

2.4 Learning of the graph structure and connectivity

The graph structure and connectivity can be estimated taking into account different criteria of a given vertex with respect to the others, such as the interdependence and the correlation.

Then, the existence or not of an edge between two vertexes, would depend on the analyzed criterion. For example, concerning the interdependence criterion:

- if the vertexes are dependent, there would be an edge;
- if they are independent, there would be no edge.

Regarding the correlation, two criteria can be used to build a graph, that is

- non-zero correlation, there would be an edge;
- null correlation there, would be no edge.

or

- correlation greater than a threshold[1], there would be an edge;
- correlation less than a threshold, there would be no edge.

Based on the aforementioned criteria, the values of the weighted adjacency matrix would be chosen, where higher values of weights would correspond to those edges with greater dependence for the interdependence criterion, and in the case of correlation criterion, in which edges with lower correlation, lead to choosing lower values of weights.

However, the construction of the graph structure and connectivity would require prior knowledge of these statistical properties, which is not always possible. In most real measured datasets the structural relations between data collected samples are usually unknown. That is, the graph information that underlies the structured data, such as the hidden structure (i.e., edges) and affinity relations (i.e., weights), is not explicitly available. Thus, it is crucial to use graph learning methods to estimate the graph topology and the edge weights. One of the most widely used graph learning approach found in the literature is the estimation of the Laplacian matrix $\mathbf{L}$ [35, 75]. Not surprisingly, several GSP techniques and applications, which have some equivalence with classical signal processing [76], rely on the knowledge of the Laplacian matrix.

In this regard, a general optimization framework for learning/estimation of three types of graph Laplacian matrices (i.e., CGL, GGL and diagonally dominant generalized graph Laplacian (DDGL)) from observed/collected signal is proposed in [35]. For this purpose, it was assumed that the signal has structural constraints, e.g., graph connectivity and sparsity level. Then, based on multiple observed signal vectors over time, the goal is to optimize a weighted graph in which the nonnegative edge weights characterize the correlation or affinity relationship between the entries of a signal vector.

In this thesis, it is paid special attention to the learning of the GGL, through the estimation of generalized Laplacian matrix $\mathbf{L}_{GGL}$, which has been shown to be useful in graph modeling of time series signals [70]. The following section describes the GGL learning method in more detail.

[1] The threshold value can be chosen to control the sparsity level of the graph.

2.4.1 GGL learning method

Let $\mathbf{X}_\tau$ be a matrix whose rows correspond to vertexes of the graph signal over time, and whose columns represents the graph signal evaluated at different time instants τ, as in Equation (2.2). A matrix $\mathbf{S}$ can be empirically obtained from the actual data as the covariance matrix of $\mathbf{X}$. Alternatively, it can be obtained by using some kernel function [77]. In practice, the use of kernel functions is more useful, as they capture nonlinear relations among random variables. The goal is to learn the GGL matrix $\mathbf{L}_{GGL} \in \mathbb{R}^{N\times N}$ from a data statistics matrix $\mathbf{S} \in \mathbb{R}^{N\times N}$.

In this thesis, as the input of the learning GGL algorithm, the data statistics matrix $\mathbf{S}$ is obtained by using a unit-variance Gaussian kernel function. To this end, let $\mathbf{s}_i$ be a vector defined as

$$\mathbf{s}_i = \begin{bmatrix} [\mathbf{X}]_{i,1} & [\mathbf{X}]_{i,2} & \cdots & [\mathbf{X}]_{i,N_\tau} \end{bmatrix}^T, \quad i = 1,\ldots,N, \tag{2.14}$$

Thus, the entries of matrix $\mathbf{S}$ are finally defined as

$$[\mathbf{S}]_{i,j} = e^{-\mathbf{s}_i^T \mathbf{s}_j}, \quad i,j = 1,\ldots,N. \tag{2.15}$$

The GGL method formulation in [35] involves the obtained matrix $\mathbf{S}$ and a symmetric regularization matrix $\mathbf{\Xi} \in \mathbb{R}^{N\times N}$ to minimize an objective function whose solution yields $\mathbf{L}$ from the following problem:

$$\begin{aligned} \underset{\mathbf{L}}{\text{minimize}} \quad & \text{Tr}(\mathbf{LK}) - \log\det(\mathbf{L}) \\ \text{s.t.} \quad & [\mathbf{L}]_{i,j} \leq 0, \quad i \neq j, \end{aligned} \tag{2.16}$$

where $\text{Tr}(\cdot)$ stands for the trace operator and $\mathbf{K} = \mathbf{S} + \mathbf{\Xi}$. Note that, the constraint $\{[\mathbf{L}]_{i,j} \leq 0, i \neq j\}$ ensures that the optimization variable $\mathbf{L}$ is an $\mathbf{L}_{GGL}$ matrix, based on the GGL definition in Equation (2.7), which forces the edge weights to be non-negative. Regarding the regularization matrix $\mathbf{\Xi}$, it can be written based on the standard l_1-regularization term with parameter ϱ as [78]

$$\mathbf{\Xi} = \varrho(2\mathbf{I} - \mathbf{1}\mathbf{1}^T), \tag{2.17}$$

where $\mathbf{I} \in \mathbb{R}^{N\times N}$ is the identity matrix and ϱ being a parameter that controls the sparsity level of the resulting $\mathbf{L}_{GGL}$ matrix. In practice, the value of ϱ is tuned until some desired level of sparsity is reached.

2.5 Smoothness of time-varying graph signal

From the point of view of graph theory, the term $\mathbf{x}^T\mathbf{L}\mathbf{x}$ is known as the graph Laplacian quadratic form [79]. Through algebraic manipulation $\mathbf{x}^T\mathbf{L}\mathbf{x} = \mathbf{x}^T(\mathbf{D} - \mathbf{W} + \mathbf{V})\mathbf{x}$ (in which the existence of vertex loops is assumed), the graph Laplacian quadratic form can be interpreted in terms of edge and vertex weights as [58]

$$2\mathbf{x}^T\mathbf{L}\mathbf{x} = \sum_{i=1}^{n} \mathbf{V}_{ii}x_i^2 + \sum_{(i,j)\in\Upsilon} \mathbf{W}_{ij}(x_i - x_j)^2, \tag{2.18}$$

where $\mathbf{V}_{ii} = \sum_{i=1}^{n} (\mathbf{L})_{ij}$, with $[\cdot]_{i,i}$ standing for the (i,i)-th entries of matrix $\mathbf{V}$ and $\mathbf{W}_{ij} = -(\mathbf{L})_{ij}$. Moreover, $\Upsilon = \left\{(i,j)|(v_i, v_j) \in \varepsilon\right\}$ is the set of index pairs of vertexes associated to weighted edge set ε. By direct inspection of (2.18), it can be noticed that assigning a greater edge weight in $\mathbf{W}_{ij}$ increases the probability of having a smaller squared difference between the corresponding x_i and x_j data points. Similarly, the smaller edge weight, the larger probability that the difference between the same points is greater.

The quadratic term $\mathbf{x}^T\mathbf{L}\mathbf{x}$ in (2.18) is commonly used to quantify and characterize the smoothness $S(\mathbf{x})$ of a graph signal $\mathbf{x}$ [35]. Thus,

$$S(\mathbf{x}) = \mathbf{x}^T\mathbf{L}\mathbf{x}\,. \tag{2.19}$$

In this sense, $S(\mathbf{x})$ can be interpreted as a quantitative characteristic that expresses how frequently a graph signal varies with respect to the underlying graph [80]. The smaller the $S(\mathbf{x})$ value, the smoother the graph signal $\mathbf{x}$ on the graph tends to be. Additionally, from the point of view of the vertex domain, $S(\mathbf{x})$ value will be low when neighboring vertexes connected by an edge with a greater weight has similar $\mathbf{x}$ values. On the other hand, from the point of view of the graph frequency domain, the smaller the frequency components associated with large eigenvalues, the lower the $S(\mathbf{x})$ value.

Regarding the structure of matrix $\mathbf{X}_\tau$, the smoothness of time-varying graph signal $\mathbf{x}_\tau$ can by defined as [49]

$$S(\mathbf{X}_\tau) = \sum_{\tau=1}^{N_\tau} S(\mathbf{X}_t) = \mathrm{Tr}(\mathbf{X}_\tau^T\mathbf{L}\mathbf{X}_\tau)\,. \tag{2.20}$$

2.5.1 Temporal difference signal

The smoothness of a time-varying graph signal is widely used in several tools based on GSP, e.g., sampling and reconstruction techniques. However, in many practical problems with real datasets, $\mathbf{x}_\tau$ is not smooth enough on the graph. For some graph-based techniques such as reconstruction, it may be interesting to maximize the smoothness level of the graph signal in order to optimize the algorithm's performance. For example, if a graph signal $\mathbf{x}_\tau$ is not smooth enough on the graph it could cause poor reconstructions with low quality [49]. One way to mitigate this problem is to exploit the correlations of the signal both on graph and along the time direction, that improves significantly the reconstruction quality [49].

Usually, the temporal difference signal (i,e,. $\mathbf{x}_\tau - \mathbf{x}_{\tau-1}$) exhibits smoothness on graph for most time-varying graph signals obtained from real datasets and exhibits better smoothness property compared with the original signals. The temporal difference operator ∇_τ can be denoted

as [49]

$$\nabla_\tau = \begin{bmatrix} -1 & & & \\ 1 & -1 & & \\ & 1 & \ddots & \\ & & \ddots & -1 \\ & & & 1 \end{bmatrix} \in \mathbb{R}^{N_\tau \times (N_\tau - 1)}, \tag{2.21}$$

where the maximum singular value of ∇_τ is larger than one. Similarly, the temporal difference signal operator is given by

$$[\mathbf{X}\nabla]_\tau = [\mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_3 - \mathbf{x}_2, \ldots, \mathbf{x}_{N_\tau} - \mathbf{x}_{N_\tau - 1}], \tag{2.22}$$

Note that, the reduction in the quantity $S([\mathbf{X}\nabla]_\tau)$ is associated to the inherent data structure and not due to that the operator ∇_τ reduces the magnitude of $\mathbf{X}_\tau$. Then, the ε-structured set of time-varying graph signals $\Phi_\varepsilon(\Gamma)$ is defined as

$$\Phi_\varepsilon(\Gamma) = \{\mathbf{X}_\tau : \mathrm{Tr}\left([\mathbf{X}\nabla]_\tau^T \mathbf{L} [\mathbf{X}\nabla]_\tau\right) \leq (N_\tau - 1)\varepsilon\}. \tag{2.23}$$

where $\Phi_\varepsilon(\Gamma)$ is composed of smoothly evolving graph signals and $\varepsilon \geq 0$ denotes the smoothness level, i.e., the smaller ε, the smoother the underlying graph signals.

2.6 Sampling via GSP

The sampling of a time-varying graph signal $\mathbf{x}_\tau$ can be represented by a linear model at every time instant τ as

$$\mathbf{y}_\tau = \bar{\mathbf{M}}_\tau \mathbf{x}_\tau, \tag{2.24}$$

where $\mathbf{y}_\tau$ is the undersampled version of $\mathbf{x}_\tau$ and $\bar{\mathbf{M}}_\tau$ is a diagonal matrix that acts as a linear mask sampling operator defined as [81]

$$[\bar{\mathbf{M}}_\tau]_{u,u} = \begin{cases} 1, & u \in P_\tau, \\ 0, & u \notin P_\tau, \end{cases} \tag{2.25}$$

with $P_\tau \subseteq \nu$ denoting the set of sampled vertexes at time instant τ.

Recalling the structure of the $N \times N_\tau$ matrix $\mathbf{X}_\tau$, Equation (2.24) can be conveniently expressed in matrix form as

$$\mathbf{Y}_\tau = \mathbf{M}_\tau \odot \mathbf{X}_\tau, \tag{2.26}$$

where $\mathbf{Y}_\tau$ is the matrix that contains the unsampled and/or missing entries of $\mathbf{X}_\tau$ and $\mathbf{M}_\tau$ being the linear mask sampling operator with dimension $N \times N_\tau$ defined as

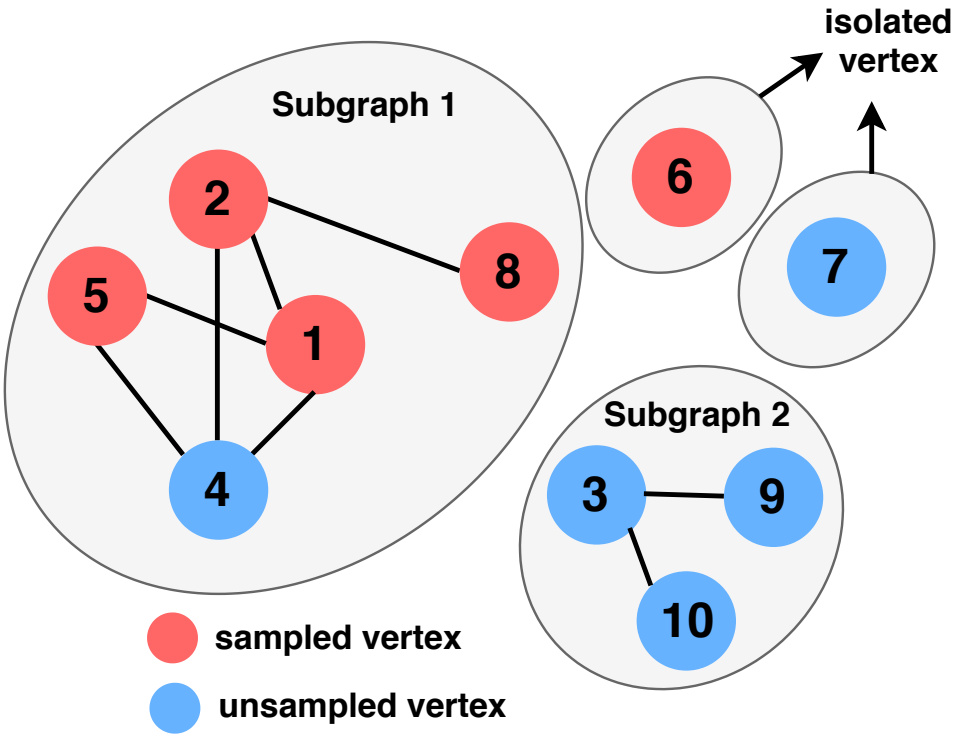
$$[\mathbf{M}_\tau]_{u,\tau} = \begin{cases} 1, & u \in P_\tau, \\ 0, & u \notin P_\tau, \end{cases} \tag{2.27}$$

with $P_\tau \subseteq \nu$ denoting the set of sampled vertexes at each time instant τ.

2.6.1 Random sampling (RS)

In [56] and [49], it was used an RS mechanism where vertexes of a graph are sampled following a uniform distribution. The number of measurements depends on a pre-established sampling rate R_s and there is no sampling constraint on the graph structure and connectivity. The non-exploitation of the graph structure may cause the loss of valuable pieces of information used by reconstruction strategies. In this sense, it may occur that all vertexes of a given subgraph are unsampled (including *trivial* subgraphs), hindering the reconstruction process of that subgraph.

Figure 2.3 – Result of sampling nodes using a random sampling strategy. Note that each of subgraph 2 is sampled.



Source: Created by the author.

Figure 2.3 illustrates an example of an RS mechanism with $R_s = 50\%$ over a graph with multiple connected components and ten vertexes. Denoting the sampled vertexes with the color red, five vertexes are randomly sampled regardless the subgraphs structure and connectivity. As can be seen, the vertex set of the subgraph 2 and the *trivial* subgraph represented by vertex 7 are unsampled, missing its connectivity and structure information to reconstruct them.

In order to mitigate the problem of RS and preserve the structure and connectivity of each subgraph, it will be proposed in later sections of this thesis a novel sampling strategy based on the graph Laplacian matrix.

2.7 Reconstruction of time-varying graph signal

The reconstruction of bandlimited graph signals requires that the signals must be exactly bandlimited on graph [49]. However, in real datasets, most graph signals tend to be smooth on graph, rather than strictly bandlimited. Theoretically, from an approach of reconstruction of smooth graph signals, if all signals $\mathbf{x}_\tau$, $\tau \in \{1, \ldots, N_\tau\}$, are smooth on graph, then it is possible to recover $\mathbf{X}_\tau$ column by column using graph reconstruction methods.

In this section, two reconstruction techniques via GSP are described, namely batch and online reconstruction strategies, which will be used throughout this thesis. In general, given the implemented sampling method and with the knowledge of the Laplacian matrix, a reconstruction strategy is carried out that collects the data over the time-varying graph and reconstructs the missing and/or sampled graph signal values. In the case of a batch reconstruction strategy, the aim is to reconstruct the matrix $\mathbf{Y}_\tau = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \ldots & \mathbf{y}_{N_\tau} \end{bmatrix}$, composed of all the undersampled vectors $\mathbf{y}_\tau$ along the temporal window N_τ. On the other hand, the online reconstruction strategy seeks to reconstruct the $\mathbf{y}_\tau$ vector at every time instant τ in a sequential manner.

As will be seen in later sections, the reconstructed measurement matrix $\hat{\mathbf{X}}_\tau$ or the reconstructed vector $\hat{\mathbf{x}}_\tau$ for the batch and online distributed strategies, respectively, may be used as input data of the machine learning (ML) approaches, in order to address the different problems treated later in this document.

2.7.1 Batch reconstruction strategies

The batch reconstruction of time-varying graph signal $\mathbf{x}_\tau$ requires to obtain the sampled values within a temporal window (e.g., N_τ) before reconstructing them all together into the matrix $\mathbf{X}_\tau$. The general batch reconstruction unconstrained optimization problem based on graph regularization is given by

$$\hat{\mathbf{X}}_\tau = \arg\min_{\mathbf{X}_\tau} ||\mathbf{M}_\tau \odot \mathbf{X}_\tau - \mathbf{Y}_\tau||_F^2 + \varphi \mathrm{Tr}\left(\mathbf{X}_\tau^T \mathbf{L} \mathbf{X}_\tau\right) + \gamma \mathrm{Tr}\left([\mathbf{X}\nabla]_\tau^T \mathbf{L} [\mathbf{X}\nabla]_\tau\right) + \delta ||[\breve{\nabla}\mathbf{X}]_\tau||_F^2, \quad (2.28)$$

where φ, γ and δ are regularization parameters, $||\cdot||_F$ denotes the Frobenius norm, $\mathbf{Y}_\tau$ is the undersampled version of the target matrix $\mathbf{X}_\tau$, $\mathbf{M}_\tau$ is the mask sampling operator defined in Equation (2.27), $[\mathbf{X}\nabla]_\tau$ is the temporal difference signal operator defined in Equation (2.22) and $\breve{\nabla}$ is the gradient of the signal $\mathbf{X}$.

The problem in (2.28) may be reduced to three sub-problems known in the literature, i.e., the smooth graph signal reconstruction (SGSR) [82], the graph-time Tikhonov reconstruction (GTTR) [47] and the smooth graph-time difference (SGTD) [56]. These reconstruction strategies will be discussed more extensively in the following.

2.7.1.1 Smooth graph signal reconstruction

By setting $\gamma = \delta = 0$ in Equation (2.28), the general reconstruction optimization problem may be reduced to the SGSR method proposed in [82]. Then, the objective function

in (2.28) can be rewritten as

$$\hat{\mathbf{X}}_{\tau}^{\mathrm{SGSR}} = \arg\min_{\mathbf{X}_{\tau}} ||\mathbf{M}_{\tau} \odot \mathbf{X}_{\tau} - \mathbf{Y}_{\tau}||_F^2 + \varphi \mathrm{Tr}\left(\mathbf{X}_{\tau}^T \mathbf{L} \mathbf{X}_{\tau}\right) . \tag{2.29}$$

The first term in Equation (2.29) is an energy term which constrains the denoised points to be close to their original positions. Theoretically, taking into account only this term, it would be possible to reconstruct the original matrix $\mathbf{X}_{\tau}$ from the partial noisy observations $\mathbf{Y}_{\tau}$, by applying low-rank matrix completion (LR-MC) techniques [83]. However, the reconstruction quality depends on many prior suitable conditions and assumptions for matrix $\mathbf{X}_{\tau}$, (e.g., degrees of freedom, matrix rank and number of sampled entries), which are hard to predict or control in real datasets.

Taking advantage that most graph signals are not usually strictly band-limited and tend to be smooth on graph, SGSR exploits the smoothness of time-varying graph signals in order to improve the reconstruction quality. Specifically, by controlling the value of φ in the second term of Equation (2.29), the smaller the $\mathrm{Tr}\left(\mathbf{X}_{\tau}^T \mathbf{L} \mathbf{X}_{\tau}\right)$ value, the smoother the signal on graph tends to be. As mentioned above, from the point of view of vertex domain, the $\mathrm{Tr}\left(\mathbf{X}_{\tau}^T \mathbf{L} \mathbf{X}_{\tau}\right)$ value will be low when neighboring vertexes connected by an edge with a greater weight has similar signal values. Thus, a smaller variation value would mean similar sample reconstructions between strongly connected vertexes.

Recalling the graph Laplacian eigendecomposition $\mathbf{L} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^T$ described in Section 2.3.1, an n-bandlimited signal lives in the span of the first n eigenvector of $\mathbf{U}$, i.e., where the eigenvalues of $\mathbf{L}$ are the smallest. The regularization term in Equation (2.29) satisfies $\mathbf{X}_{\tau}^T \mathbf{L} \mathbf{X}_{\tau} = \left(\mathbf{X}_{\tau}^T \mathbf{U}\right) \mathbf{\Lambda} \left(\mathbf{U}^T \mathbf{X}_{\tau}\right)$. Thus, this term penalizes signals with energy concentrated at high frequencies (i.e., greater eigenvalues) more than signals with energy concentrated at low frequencies (i.e., smallest eigenvalues). In other words, this regularization term encourages the reconstruction of low-frequency signals, i.e., signals approximately bandlimited.

The advantage of SGSR is that it can be solved efficiently by iterative methods even for large graphs, e.g., the conjugate gradient algorithm [84]. Indeed, each step of this algorithm can be implemented by using only matrix-vector multiplications with $\mathbf{M}_{\tau}$ and $\mathbf{L}$, which are both sparse matrices.

2.7.1.2 Graph-time Tikhonov reconstruction

First implemented in [85], Tikhonov's regularization became one of the most used methods to filter or denoise graph signals. Specifically, the general regularization term $\delta||\breve{\nabla}\mathbf{x}||_p^p$ is added to known convex optimization problems, in which $\breve{\nabla}$ is weighted by the selection of a scalar δ.

Tikhonov's regularization based on the l_1-norm (i.e., $p = 1$) is widely used in literature as image denoising and reconstruction [86, 87]. Also known as total variation (TV) regularization, it gained rapid popularity from its ability to produce denoised images that retain sharp intensity boundaries.

The quadratic Tikhonov-type regularization, based on the l_2-norm (i.e., $p = 2$), promotes smooth solutions over graph signal, reducing the high-frequency content such as noise. This represents an undesirable feature in image denoising problems, due to sharp intensity boundaries that are smoothed, thereby smearing sharp features embedded in the image. However, the smoothness enforced by the l_2-norm regularization, can be usefully exploited in order to improve the solution of reconstruction problems in which time-varying signals are assumed. Thus, by setting $\gamma = 0$ in Equation (2.28), the problem is reduced to the GTTR method proposed in [47]

$$\hat{\mathbf{X}}_\tau^{\text{GTTR}} = \arg\min_{\mathbf{X}_\tau} ||\mathbf{M}_\tau \odot \mathbf{X}_\tau - \mathbf{Y}_\tau||_F^2 + \varphi \text{Tr}\left(\mathbf{X}_\tau^T \mathbf{L} \mathbf{X}_\tau\right) + \delta ||[\breve{\nabla}\mathbf{X}]_\tau||_F^2 . \tag{2.30}$$

As noted, the GTTR method basically adds the term $\delta||[\breve{\nabla}\mathbf{X}]_\tau||_F^2$, corresponding to the Tikhonov's regularization based on the l_2-norm, in the SGSR optimization problem given by Equation (2.29). In this way, it is possible to reinforce the smoothness level of the time-varying signal $\mathbf{X}_\tau$ provided by the term $\varphi \text{Tr}\left(\mathbf{X}_\tau^T \mathbf{L} \mathbf{X}_\tau\right)$, and an improve in the reconstruction quality is expected.

2.7.1.3 Smooth graph-time difference reconstruction

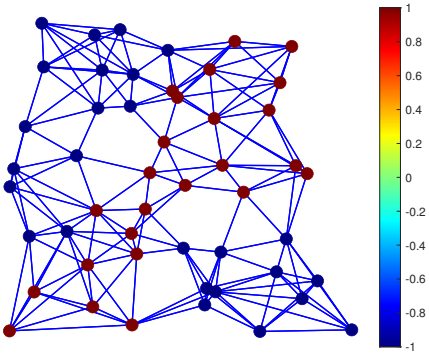
In many practical problems with real datasets, signal $\mathbf{x}_\tau$ is not smooth enough on graph, which can cause poor reconstructions with high reconstruction error levels. One way to mitigate this problem is by exploiting the correlations of the signal both on graph and along the time direction, which may significantly improve the reconstruction quality. In this context, the quadratic term $\left([\mathbf{X}\nabla]_\tau^T \mathbf{L} [\mathbf{X}\nabla]_\tau\right)$ in Equation (2.28), which represents the temporal differences of time-varying graph signals, usually exhibits better smoothness on the graph representation than the signals themselves [56]. Then, by setting $\varphi = \delta = 0$ in Equation (2.28), the SGTD problem can be rewritten as [49]

$$\hat{\mathbf{X}}_\tau^{\text{SGTD}} = \arg\min_{\mathbf{X}_\tau} ||\mathbf{M}_\tau \odot \mathbf{X}_\tau - \mathbf{Y}_\tau||_F^2 + \gamma \text{Tr}\left([\mathbf{X}\nabla]_\tau^T \mathbf{L} [\mathbf{X}\nabla]_\tau\right) . \tag{2.31}$$

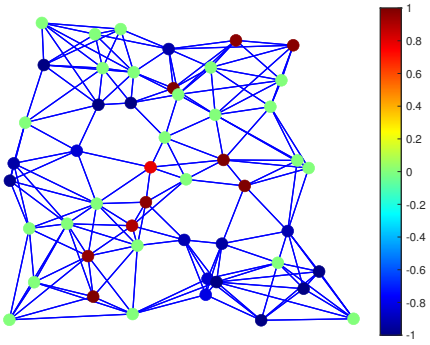
Figure 2.4 illustrates an example of graph signal reconstruction using the three batch reconstruction methods described above. For a better understanding of the reconstruction process, the graph signals are represented by a heatmap (i.e., the graph vertex colors represent graph signal values), in which its values are randomly distributed between the vertexes of the graph. Specifically, Figure 2.4a shows a graph signal over an arbitrary graph with $N = 50$ vertexes at a certain time instant τ. Figure 2.4b illustrates the graph signal $\mathbf{y}_\tau$ at the same time instant τ, after the application of the mask sampling operator $\mathbf{M}_\tau$ given in Equation (2.27). In this example, $\mathbf{M}_\tau$ is chosen with a sampling rate $R_S = 50\%$ (i.e., the graph signal of half of the vertexes has been set to 0). Figures 2.4c, 2.4d and 2.4e show the reconstructed graph signal $\hat{\mathbf{x}}_\tau$ by solving the SGTD, GTTR and SGSR reconstruction problems, respectively. The near-optimal values of the regularization parameters γ, δ and φ were found through a exhaustive grid search. By

Figure 2.4 – Example of graph signal reconstruction using the three batch reconstruction methods.

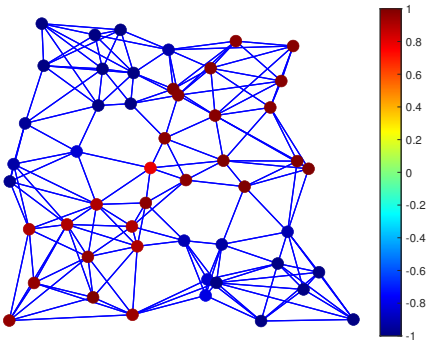
(a) Original graph signal $\mathbf{x}_\tau$ at a certain time instant τ.



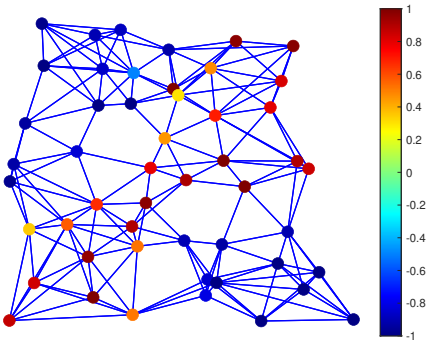
(b) Undersampled graph signal $\mathbf{y}_\tau$, with $R_S = 50\%$.



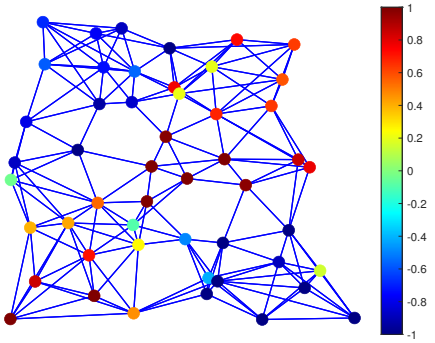
(c) Result of graph signal reconstruction using the SGTD method.



(d) Result of graph signal reconstruction using the GTTR method.



(e) Result of graph signal reconstruction using the SGSR method.



Source: Created by the author.

direct inspection of graph vertex colors, it can be noted which of the three batch reconstruction methods provides better reconstruction, when compared with the original graph signal values in Figure 2.4a. As can be seen, the SGTD solution in Figure 2.4c more closely resembles the original signal colors than the others two batch reconstruction techniques in Figures 2.4d and 2.4e. This result is expected because the graph signal used in the GTTR and SGSR methods is not smooth enough, which causes a significant reconstruction performance loss when compared to

the SGTD, that uses the temporal difference to improve the smoothness of the signal.

2.7.2 Online reconstruction strategy

The batch reconstruction strategy described above requires one to obtain the sampled values within a long time before reconstructing them all together. This can lead to a high reconstruction delay and a high computational complexity, not being suitable for practical applications that face real-time requirements and low latency. In this context, an online strategy proposed in [49], which is based on the local properties of the temporal difference operator and the graph Laplacian matrix, is adopted.

The online smooth graph-time difference (O-SGTD) reconstruction strategy aims to reconstruct the current missing graph signal values according to the sampled graph signal of current and previous time instants. Thus, the unconstrained optimization problem given by Equation (2.28) with $\beta = \delta = 0$ can be rewritten for the O-SGTD strategy as [49]

$$\hat{\mathbf{x}}_\tau = \arg\min_{\mathbf{x}_\tau} ||\bar{\mathbf{M}}_\tau \mathbf{x}_\tau - \mathbf{y}_\tau||_2^2 + \gamma \left(\mathbf{x}_\tau - \hat{\mathbf{x}}_{\tau-1}\right)^T \mathbf{L} \left(\mathbf{x}_\tau - \hat{\mathbf{x}}_{\tau-1}\right) , \tag{2.32}$$

where $\mathbf{x}_\tau$ is the current (i.e., at time instant τ) signal to be reconstructed and $\hat{\mathbf{x}}_{\tau-1}$ is the reconstructed signal at the previous $\tau - 1$ time instant. Note that, $\bar{\mathbf{M}}_\tau$ is the linear mask sampling operator defined in Equation (2.25).

It is worth recalling that the batch reconstruction problem SGTD, as described in Equation (2.28), needs to receive every vector $\mathbf{y}_\tau$ to compose matrix $\mathbf{Y}$ prior to the reconstruction of $\mathbf{X}_\tau$. On the other hand, the online strategy O-SGTD, as formulated in Equation (2.32), can obtain every $\mathbf{x}_\tau$ in a sequential fashion given only $\mathbf{y}_\tau$ and $\mathbf{x}_{\tau-1}$.

In next chapters, it will be used some of the graph-based tools and concepts described throughout this chapter, in order to reduce the feedback channel burden and therefore, improve the ML-based prediction tasks for two problems in mobile communications.

The main motivation of using the graph-based tools is identify and properly exploit data structures generated by the different sources of information in the 5G systems. Such data are usually very complex since they are intrinsically and possibly irregularly structured. For instance, mobile and wireless networks are irregularly deployed in space and their measurements depend on their geographical positions. Then, analysis and compression of such data is a challenging task, which can be tackled by using tools based on GSP.

3 SUPERVISED LEARNING AND GRAPH SIGNAL PROCESSING STRATEGIES FOR BEAM TRACKING IN HIGHLY DIRECTIONAL MOBILE COMMUNICATIONS

The use of efficient beam tracking mechanisms becomes necessary in highly directional communication of the 5G systems. In this context, this chapter analyzes the tracking problem and proposes a framework that exploits the samples in the HUD to efficiently estimate and predict the channel state at the BS. The framework is composed of two steps. First, a supervised learning algorithm, namely K-NN, is evaluated as a means of (i) finding the most similar historical samples to some beam measurements reported by the UE, and (ii) predict the corresponding channel state. As a second step, a sampling and reconstruction strategy based on GSP called K-NN-R is introduced in order to reduce the beam search space during the beam measurement stage, which allows a more efficient usage of the feedback channel. Simulation results illustrate the performance of the proposal in terms of NMSE in comparison with three traditional/baseline prediction techniques. The K-NN technique provides a better performance than the baseline approaches for any length of the observed temporal window and with full beam sweep. Meanwhile, the K-NN-R framework outperforms the baseline approaches with only half of the beam pairs and throughout a significantly low length of observed temporal window.

3.1 Introduction

Capacity demand in mobile communications has been continuously increasing and this trend is expected to continue. To improve system capacity, the use of mmWave bands are envisioned as a feasible solution in 5G systems, incorporating mmWave small cells into current 4G systems. Millimeter waves make possible the usage of a large number of antenna elements, which allows highly directional communications through the use of beams with very narrow beamwidth (called pencil beams). Pencil beams offer high array gain, providing mmWave small cell base-station (BS) with the capability of achieving the desired link budget targets, and enabling energy efficient communication. However, the use of mmWave frequencies also brings new challenges caused by severe path loss due to the communication channel, blockage and other environmental obstructions. Link establishment and maintenance between a BS and a user equipment (UE) in highly directional communications are often rather difficult processes. In order to cope with this disadvantage, significant overhead and network resources are usually wasted in beam training procedures with the aim to determine the best directions of transmission/reception pairs between a BS and a UE. In this context, fast and efficient beam training and beam tracking strategies are of paramount importance for efficient operation of mmWave in mobile environments.

Beam management problems can benefit from the availability of a HUD samples to provide solutions. The nature of the historical data may be quality measurements, e.g. SNR, CSI,

channel coefficients, AoD and AoA, performed by UEs that (once or periodically) report them to the network. As an example, in beam tracking, such set of data samples can help modeling the interdependency among BSs, beams and UEs data samples, which is suitable for ML-based solutions, such as supervised learning. Advantageously, beam tracking can then be performed more efficiently by predicting/estimating the direction changes of UEs, even some time instants ahead.

In this chapter the concept of an adaptive beam tracking framework is proposed, where the time evolution of the AoA and the AoD can be estimated to efficiently reduce the beam search space, relying on measurements of the UE output observations (once or periodically) and the AoA/AoD pairs associated with these measurements, which are stored in a HUD. A supervised learning method, namely K-NN, is evaluated as a means of finding the K most similar historical observation samples and learn to predict the corresponding channel state. Further, a sampling and reconstruction strategy based on GSP called K-NN-R is introduced, as a pre-processing stage to the K-NN algorithm. Such data pre-processing aims to reduce the feedback bandwidth and, therefore, counteracting the measurement process challenges. The evaluation of the proposed methods is carried out for the channel estimation perspective at each time instant during a reporting temporal window and as channel prediction tool, in which, from the measurements collected only throughout an observed temporal window, the channel matrix is predicted some time instants ahead.

3.2 System model

In this section we describe the system model for MIMO mmWave communications. First, we describe the spatial channel model and its state evolution model in the angular domain and then we introduce the procedure for beam training. Generally speaking, we formulate the state evolution model as

$$x_\tau \triangleq f(x_{\tau-1}) + w_{\tau-1}\,, \tag{3.1}$$

where x_τ stands for the state at time instant τ, $f(\cdot)$ is the transition function which evolves a state from a time instant to the next, and $w_{\tau-1}$ is the process noise that is assumed to be white and Gaussian distributed. The role of w_τ is to capture effects in the evolution process which are not modeled by $f(\cdot)$. Additionally, the measurements are modeled as

$$y_\tau \triangleq g(x_\tau) + v_\tau\,, \tag{3.2}$$

where y_τ denotes a measurement at time instant τ, $g(\cdot)$ is the observation function that maps a given state to a value that is measured (or observed), and v_τ is the measurement noise also assumed to be white and Gaussian distributed.

As it will be discussed throughout this chapter, the channel model is parameterized by state variables in terms of arrival/departure angles and path gains. Besides, the observation

function $g(\cdot)$ is described in terms of steering vectors that are mathematically modeled as complex exponential functions, i.e., it is nonlinear while the transition function $f(\cdot)$ is formulated as a linear one. At last, a measurement (or observation) is simply the received beamformed signal that is combined and sampled at time instant τ.

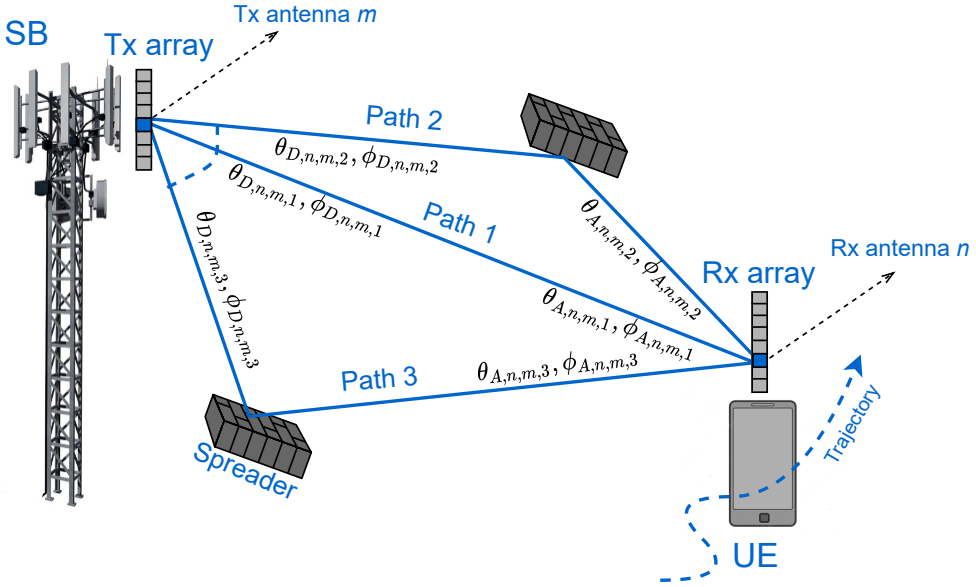
3.2.1 Channel model

In conventional mobile communications systems, with frequencies below 6 GHz, it is usual to consider rich-scattering channel models. However, MIMO mmWave channels are modeled with a few scatterers due to high directivity characteristics, narrow beams and strong attenuations. The number of propagation paths for these channel models is then limited, which results in a sparse channel matrix [88]. Due to this sparse nature, the channel matrix in the angular domain can be well defined in terms of AoA, AoD and path gains as a parametric channel model. Therefore, channel estimation can be reduced to the estimation of such parameters.

As described in [88], empirical measurements of MIMO mmWave channels show that delayed versions of the signal (paths) arrive at the receiver in clusters, such that each cluster has similar statistical characteristics of angles, power and delay. Thus, a cluster is a group of scatterers with similar statistics, and represents a macro region in which reflection, diffraction, attenuation and scattering are around well-defined statistical averages.

We assume a single user MIMO mmWave system in which the BS is equipped with an antenna array of N_t elements, indexed by $m = 1, 2, \ldots, N_t$, and a UE is equipped with N_r antenna elements, indexed by $n = 1, 2, \ldots, N_r$. We also consider L clusters of a single scatterer. For each combination between a receiving antenna element n, a transmitting antenna element m and a cluster l, there exists a single path p of index (n, m, l) as illustrated in Figure 3.1.

Figure 3.1 – Spatial mmWave channel model.



Source: Created by the author.

Still in Figure 3.1, a mmWave channel can be expressed in relation to the parameters defined for each path: AoA, AoD and complex gain α. Hence, the frequency response of the time-varying geometric 3D channel model with L paths at time τ is described by [15]

$$\mathbf{H}(\tau) = \sum_{l=1}^{L} \alpha_{l,\tau} \mathbf{a}_r(\theta_{A,l,\tau}, \phi_{A,l,\tau}) \mathbf{a}_t^H(\theta_{D,l,\tau}, \phi_{D,l,\tau}), \tag{3.3}$$

where $\mathbf{H}(\tau) \in \mathbb{C}^{N_r \times N_t}$ and D, A, t and r indicate AoD, AoA, transmitter and receiver, respectively. The superscript H indicates the Hermitian conjugate, $\alpha_{l,\tau}$ is the l-th path gain at time τ, and $\theta_{D,l,\tau}$,$\phi_{D,l,\tau}$ is the l-th path AoD at time τ, where θ and ϕ are the elevation and azimuth components of the transmission direction, respectively. Similarly, $\theta_{A,l,\tau}$,$\phi_{A,l,\tau}$ is the l-th path AoA. If we restrict the channel model for 2-dimensional (2D) coordinates, we only need to cope with angles in the azimuth.

Still in Equation (3.3), $\mathbf{a}_t(\cdot)$ and $\mathbf{a}_r(\cdot)$ are the beam steering vectors for the BS (transmitter) and the UE (receiver), respectively. The steering vectors, also known as array response, describe the phase delays of each plane wave generated (or received) by an antenna array element, and depend on the geometry of the array. Consider a 2D model ($\theta = 0$) and a uniform linear array (ULA) at both the transmitter and the receiver, the steering vectors in Equation (3.3) at each τ are given by

$$\mathbf{a}_t(\phi_{D,l,\tau}) = \frac{1}{\sqrt{N_t}} [1, \quad e^{jd\zeta(\phi_{D,l,\tau})}, \quad e^{j2d\zeta(\phi_{D,l,\tau})}, \ldots, e^{jd(N_t-1)\zeta(\phi_{D,l,\tau})}]^T, \tag{3.4}$$

and

$$\mathbf{a}_r(\phi_{A,l,\tau}) = \frac{1}{\sqrt{N_r}} [1, \quad e^{jd\zeta(\phi_{A,l,\tau})}, \quad e^{j2d\zeta(\phi_{A,l,\tau})}, \ldots, e^{jd(N_r-1)\zeta(\phi_{A,l,\tau})}]^T, \tag{3.5}$$

for the transmitter and the receiver, respectively [15], where d is a distance between the adjacent antennas. In addition, $\zeta(\phi_{D,l,\tau}) = \frac{2\pi}{\lambda}[\sin(\phi)]$ where $\phi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$ is a angle for AoD and λ is the carrier wavelength. Finally, $\zeta(\phi_{A,l,\tau})$ for AoA is defined similarly.

3.2.2 State evolution model

Let $\boldsymbol{\phi}_{D,\tau} = [\phi_{D,1,\tau} \ldots \phi_{D,L,\tau}]$, $\boldsymbol{\phi}_{A,\tau} = [\phi_{A,1,\tau} \ldots \phi_{A,L,\tau}]$, be vector variables that represent AoD in the azimuth, and AoA in the azimuth, respectively. In addition, $\boldsymbol{\alpha}_\tau = [\alpha_{1,\tau} \ldots \alpha_{L,\tau}]$ is the complex gain vector with real and imaginary part $\boldsymbol{\alpha}_\tau^{real} = \mathcal{R}(\boldsymbol{\alpha}_\tau)$ and $\boldsymbol{\alpha}_\tau^{img} = \mathcal{I}(\boldsymbol{\alpha}_\tau)$, respectively. Then, for a 2D channel model the state vector is defined as

$$\mathcal{X}_\tau = \begin{bmatrix} \boldsymbol{\phi}_{D,\tau} & \boldsymbol{\phi}_{A,\tau} & \boldsymbol{\alpha}_\tau^{real} & \boldsymbol{\alpha}_\tau^{img} \end{bmatrix}, \tag{3.6}$$

such that $\mathcal{X}_\tau$ has $D_L = 4L$ dimensions. Note that by using the real and imaginary parts of $\boldsymbol{\alpha}_\tau$, the state vector $\mathcal{X}_\tau$ is a real vector which helps to avoid implementation issues when real and complex numbers are mixed. It is assumed that these vector variables are independent of each other. For each variable in $\mathcal{X}_\tau$ a variance will be assigned. The variance indicates the extent

of the mobility for the UE. The higher rate mobility corresponds to higher values of variance. Hence, the appropriate variance value can be found by mapping the average speed of the UE. In this chapter, the variances of the angles will be written as a function of a single variance σ_u^2.

Let us also consider that $\boldsymbol{\alpha}_\tau$ follows the linear model given by [15]

$$\boldsymbol{\alpha}_\tau = \rho\boldsymbol{\alpha}_{\tau-1} + \psi_{l,\alpha,\tau-1}\,, \tag{3.7}$$

where ρ is the channel time correlation coefficient and $\psi_{l,\alpha,\tau-1} \sim \mathcal{N}\left(0,(1-\rho^2)\right)$. It is assumed a generic evolution model for the AoA/AoD angles that is driven by a linear Gaussian process. The state evolution model can be written as

$$\mathcal{X}_\tau = \mathbf{Z}\mathcal{X}_{\tau-1} + \mathbf{u}_{\tau-1}\,, \tag{3.8}$$

where $\mathbf{Z} = \text{diag}([\mathbf{1}_{1\times 2L}, \rho\mathbf{1}_{1\times 2L}])$, with $\text{diag}(\cdot)$ representing diagonal matrix. The vector $\mathbf{u}_{\tau-1} \sim \mathcal{N}(0,\boldsymbol{\Sigma}_u)$ with $\boldsymbol{\Sigma}_u = \text{diag}([\boldsymbol{\sigma}_\phi, \boldsymbol{\sigma}_\phi, \boldsymbol{\sigma}_\alpha, \boldsymbol{\sigma}_\alpha])$ for the 2D channel. Since the state vector $\mathcal{X}_\tau$ in (3.8) contains all the parameters that characterize the channel matrix in (3.3), one can express the channel matrix $\mathbf{H}$ in terms of $\mathcal{X}_\tau$, so that $\mathbf{H}(\mathcal{X}_\tau)$, which acts as the observation function. This way, once the state vector is estimated/tracked, the channel matrix can be straightforwardly obtained.

3.2.3 Beam training cycle

Consider the downlink communication in a ULA 2D channel model where a transmit codebook containing N_t transmit directions and a receive codebook containing N_r receive directions are used for beam training purposes. Specifically, it is assumed a DFT codebook proposed in [89]. Moreover, it is assumed that for each beam tracking instant a beam training cycle is performed during which the channel does not change. Each beam training cycle consists of a full scan within the beam space, which comprises all the $N_b = N_t N_r$ transmit-receive direction pairs. More specifically, the p-th beamforming vector (BV) $\mathbf{f}_p$ for transmission and the q-th combining vector (CV) $\mathbf{w}_q$ for reception are given by

$$\mathbf{f}_p = \mathbf{a}_t\left(\arcsin\left(-1+\frac{2(p-1)}{N_t}\right)\right), \quad p = 1,\ldots,N_t\,, \tag{3.9}$$

$$\mathbf{w}_q = \mathbf{a}_r\left(\arcsin\left(-1+\frac{2(q-1)}{N_r}\right)\right), \quad q = 1,\ldots,N_r\,, \tag{3.10}$$

where $\mathbf{f}_p \in \mathbb{C}^{N_t\times 1}$, $\mathbf{w}_q \in \mathbb{C}^{N_r\times 1}$, $\mathbf{a}_t(\cdot)$ and $\mathbf{a}_r(\cdot)$ are the steering vector functions described in Equations (3.4) and (3.5), respectively. Note that the azimuth angles that parameterize the steering vector functions above come from the N_t-point (or N_r-point) discretization of the $\sin(\cdot)$, defined in the function $\zeta(\cdot)$ in the interval $[-1,1)$. Such an interval is chosen to be in compliance with the azimuth angular spread which is within the interval $\left[-\frac{\pi}{2},\frac{\pi}{2}\right]$.

Moreover, a pilot signal s is transmitted for each transmit-receive direction pair. For simplicity, it is assumed $s = 1$. Now, let $g_{q,p}(\mathcal{X}_\tau) \in \mathbb{C}$ be the equivalent channel gain at time

instant τ when the p-th BV $\mathbf{f}_p$ and the the q-th CV $\mathbf{w}_q$ are selected and given by

$$g_{q,p}\left(\mathcal{X}_\tau\right) = \mathbf{w}_q^H \mathbf{H}\left(\mathcal{X}_\tau\right) \mathbf{f}_p\,, \tag{3.11}$$

where $\mathbf{H}\left(\mathcal{X}_\tau\right)$ is the channel matrix parameterized by the state vector $\mathcal{X}_\tau$. It is worth observing that $g_{q,p}(\cdot)$ acts as a nonlinear observation function in our general measurement model. Then, the received signal beamformed by the p-th BV and combined by the q-th CV at time instant τ is given as follows:

$$y_{q,p}\left(\tau\right) = g_{q,p}\left(\mathcal{X}_\tau\right) + \mathbf{w}_q^H \mathbf{v}_\tau\,, \tag{3.12}$$

where $\mathbf{v}_\tau \sim \mathcal{CN}(\mathbf{0}, \sigma_v^2 \mathbf{I}_{N_r})$ represents an i.i.d. Gaussian noise vector at time instant τ. Besides, all the $N_b = N_t N_r$ received signals can be conveniently expressed in matricial form. To this end, let $\mathbf{F} \in \mathbb{C}^{N_t \times N_t}$ be the transmit codebook matrix defined as

$$\mathbf{F} = \begin{bmatrix} \mathbf{f}_1 & \cdots & \mathbf{f}_{N_t} \end{bmatrix}. \tag{3.13}$$

Let $\mathbf{W} \in \mathbb{C}^{N_r \times N_r}$ be the receive codebook matrix defined as

$$\mathbf{W} = \begin{bmatrix} \mathbf{w}_1 & \cdots & \mathbf{w}_{N_r} \end{bmatrix}. \tag{3.14}$$

Thus, the received signal matrix $\mathbf{Y} \in \mathbb{C}^{N_r \times N_t}$ during the beam training cycle and parameterized by state vector $\mathcal{X}_\tau$ is described as

$$\mathbf{Y}\left(\mathcal{X}_\tau\right) = \underbrace{\mathbf{W}^H \mathbf{H}\left(\mathcal{X}_\tau\right) \mathbf{F}}_{\mathbf{G}(\mathcal{X}_\tau)} + \mathbf{W}^H \mathbf{V}\,, \tag{3.15}$$

where $\mathbf{V}$ represents the $\mathbf{v}_\tau$ vector at each time instant τ and $\mathbf{G} \in \mathbb{C}^{N_r \times N_t}$ is the observation matrix whose (q,p)-th entry denotes the observation function $g_{q,p}(\cdot)$ in (3.11) evaluated at $\mathcal{X}_\tau$.

Also for the sake of convenience, both the observation and received signal matrices can be expressed in vector form using the vectorization operator $\mathrm{vec}(\cdot)$ as follows:

$$\mathbf{g}_\tau = \mathrm{vec}\left(\mathbf{G}\left(\mathcal{X}_\tau\right)\right), \tag{3.16}$$

where $\mathbf{g}_\tau \in \mathbb{C}^{N_b \times 1}$, and

$$\mathbf{y}_\tau = \mathrm{vec}\left(\mathbf{Y}\left(\mathcal{X}_\tau\right)\right), \tag{3.17}$$

where $\mathbf{y}_\tau \in \mathbb{C}^{N_b \times 1}$. Besides, in order to avoid handling complex numbers, $\mathbf{g}_\tau$ and $\mathbf{y}_\tau$ can be split into their real and imaginary parts, which yield

$$\mathbf{g}_{\mathbb{R}|\mathcal{I}} = \begin{bmatrix} \mathbb{R}\left(\mathbf{g}_\tau\right) \\ \mathcal{I}\left(\mathbf{g}_\tau\right) \end{bmatrix}, \tag{3.18}$$

and

$$\mathbf{y}_{\mathbb{R}|\mathcal{I}} = \begin{bmatrix} \mathbb{R}\left(\mathbf{y}_\tau\right) \\ \mathcal{I}\left(\mathbf{y}_\tau\right) \end{bmatrix}. \tag{3.19}$$

where $\mathbb{R}(\cdot)$ and $\mathcal{I}(\cdot)$ stand for the real part and the imaginary part, respectively, of the corresponding input argument.

3.3 Tracking with supervised learning

Machine learning methods typically present two mainstreams, namely supervised learning and unsupervised learning. In supervised scenarios, each observed input data has a desired output/label so that, via an offline training phase, a function that maps input and output data can be learned. Then, for any new observed data its associated output can be obtained. Applications lying in this learning category are those of classification, prediction and regression.

An example of method to evaluate the learning aspect in the tracking problem is the so-called K-NN [90] for regression. A brief explanation of the general idea behind it goes as follows. Consider a generic *unlabeled* observation data x_u, i.e., its label y_u is unknown. One may then want to find (or predict) the label y_u of x_u. To this end, let us assume that one can rely on the availability of a (desirably rich) set Ψ of N_s training (or supervision) data pairs (x, y), where y corresponds to the label of x. Thus,

$$\Psi = \left\{(x_1, y_1), (x_2, y_2), \ldots, (x_{N_s}, y_{N_s})\right\}. \tag{3.20}$$

Then, by using some metric of similarity (e.g. Euclidean distance function $d(\cdot,\cdot)$) one can find the K nearest training input data in Ψ to the unlabeled data x_u by (i) calculating the distance between x_u and every training input data, and (ii) taking the K nearest ones. That is,

$$j_k = \arg\min_j{}^{(k)} d(x_u, x_j), \tag{3.21}$$

where x_j is the input data of $(x_j, y_j) \in \Psi$ and $\min^{(k)}$ stands for the k-th smallest value of the corresponding input argument. Once the K-NN input data are found, the estimate of the label of x_u is given by

$$\hat{y}_u = \frac{1}{K}\sum_{k=1}^{K} y_{j_k}, \tag{3.22}$$

where y_{j_k} is the output data of $(x_{j_k}, y_{j_k}) \in \Psi$. The right-hand part of (3.22) is simply the average of the K-NN output data found, but different functions can be used (e.g, weighted average, where weight may be the normalized distances). The prediction error in this case is the difference between the actual y_u and its estimate $\hat{y}_u$.

In general, the input data x can be multidimensional, each dimension standing for a feature of that data. In this case, each input data x is a vector whose entries are features. In a more general case, the evolution (in time, for instance) of each feature may also be stored as a single input, which makes input data x be a matrix. Then, distance function $\mathrm{d}(\cdot,\cdot)$ is conveniently generalized to be the Frobenius norm. The output y may also have multiple dimensions, each being a label. Besides, values for features and labels can generally belong to the set of real numbers.

Finding the best value for K is challenging due to the so-called bias-variance trade-off [90]. That is, extreme values of (either too small or too large) do not necessarily lead to a

small estimation/prediction error. In general, small values, say $K = 1$, corresponds to a predictor with low-bias and high-variance. In this case, it will always pick the nearest neighbor, which may vary as new training data is added into Ψ. On the other hand, large values, say $K = 100$, lead to a predictor with high-bias and low-variance. That is, the resulting estimate will be an average of a very large number of data samples with contribution of many labels different from the actual one. Besides, adding new training data does not affect much the prediction in this case. Consequently, in practice, none of them provides good predictions.

3.3.1 K-NN tracking framework

Bringing the idea behind the K-NN to the tracking problem, a label is defined as a given state vector $\mathcal{X}_\tau$, while an input data, either observation or training one, is a given observation vector $\mathbf{y}_\tau$. Recall that the observation vector $\mathbf{y}_\tau$ is the measured beamformed signal vector at time instant τ for every pair of BV and CV. Thus, the goal here is to apply K-NN to predict the corresponding state vector of the channel that generated the observation vector measured by the UE.

Now let us assume that the serving BS has a HUD available and stored in a central entity or in the cloud, whose entries are the observation vectors $\mathbf{y}_\tau$ and their corresponding state vectors $\mathcal{X}_\tau$, which represent historical measurements at some (relative) time instant τ. Moreover, $(\mathbf{y}_\tau, \mathcal{X}_\tau)$ pairs may span consecutive time instants to account for measurements along a trajectory of interest. It is worth mentioning that the measurement and estimation processes that feed the $(\mathbf{y}_\tau, \mathcal{X}_\tau)$ pairs into the HUD are carried out offline, not interfering or aggregating overload in the dynamic tracking process. Specifically, the received observation vectors and the estimated state vectors are fed back to the HUD, in order to update the historical samples and improve the richness of the database. However, the measurement and estimation processes are not error-free, being necessary the scheduled database refresh operations to periodically purge old data.

More specifically, since the channel model described in Section 3.2 is purely stochastic, the entries of the HUD are defined as being correlated versions of a given $(\mathbf{y}_\tau, \mathcal{X}_\tau)$ pair whose input data label needs to be predicted. The idea behind that is to emulate some randomness in the measurement and estimation processes of the channel within a trajectory of interest, as if they were carried out in multiple different days. To this purpose, for each unlabeled $\mathbf{y}_\tau$, a set of HUD samples are generated as follows:

$$\mathcal{X}'_{\tau,j} = \mathcal{X}_{\tau+\delta} + \sqrt{\beta \boldsymbol{\Sigma}_u)} \mathcal{N}_j \,, \tag{3.23}$$

where $\beta \in [0,1]$ is a correlation factor that scales the variance of the state variable noises, $\mathcal{N}_j$ is a zero-mean white Gaussian noise vector, and δ is a random delay uniformly distributed within $[-\Delta, \Delta]$. The delay δ introduces some delayed state vectors in the training data at time instant τ; otherwise, K-NN would act as an unbiased estimator, and the K value would be chosen as large as possible. Moreover, for each $\mathcal{X}'_{\tau,j}$, its corresponding observation vector $\mathbf{y}'_{\tau,j}$ is simply obtained as in Equation (3.17).

Therefore, for a given $\mathbf{y}_\tau$ whose label is unknown at time instant τ, there is set from the HUD given as

$$\Psi_{\text{HUD},\tau} = \left\{ (\mathbf{y}'_{\tau,1}, \mathcal{X}'_{\tau,1}), (\mathbf{y}'_{\tau,2}, \mathcal{X}'_{\tau,2}), \ldots, (\mathbf{y}'_{\tau,N_s}, \mathcal{X}'_{\tau,N_s}) \right\}, \tag{3.24}$$

which is used by the K-NN procedure, described in Section 3.3, to estimate the label (i.e., the state vector) of $\mathbf{y}_\tau$. That is, the k-th nearest neighbor to $\mathbf{y}_\tau$ is given by

$$j_k = \underset{\mathbf{y}\in\Psi_{\text{HUD},\tau}}{\arg\min}^{(k)} \mathrm{d}(\mathbf{y}_\tau, \mathbf{y}) \,. \tag{3.25}$$

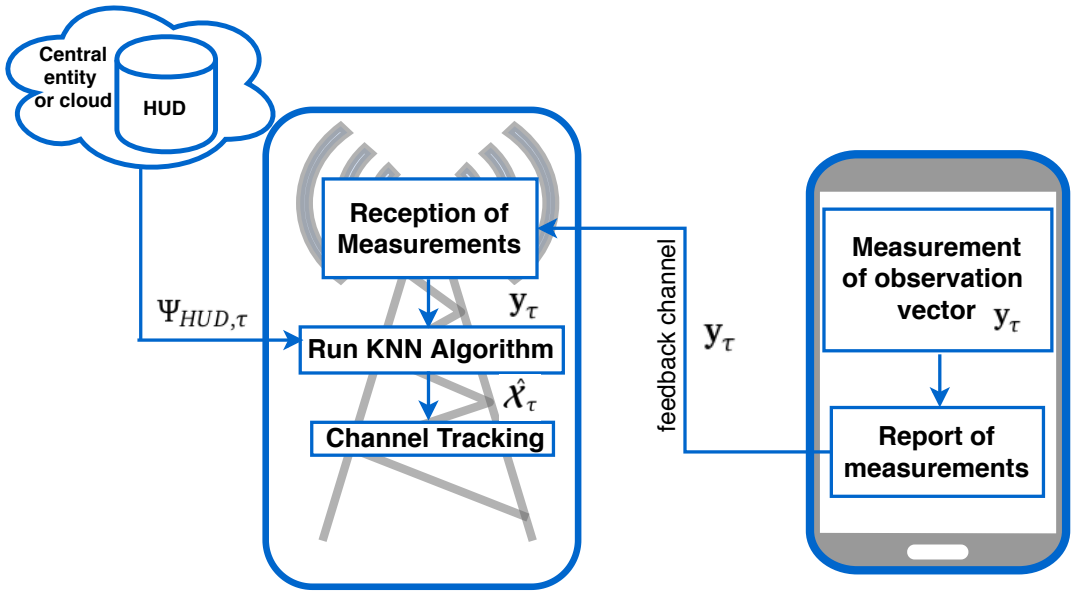
Once the K-NN input data are found in $\Psi_{\text{HUD},\tau}$, the estimate $\hat{\mathcal{X}}_\tau$ of the $\mathbf{y}_\tau$'s label is finally given by

$$\hat{\mathcal{X}}_\tau = \frac{1}{K} \sum_{k=1}^{K} \mathcal{X}'_{\tau,k}, \tag{3.26}$$

where $\mathcal{X}'_{\tau,k}$ is the output data of $(\mathbf{y}'_{\tau,k}, \mathcal{X}'_{\tau,k}) \in \Psi_{\text{HUD},\tau}$.

It is worth mentioning that our supervised learning tracking framework does not demand the track of state vectors on the UE side, as illustrated in Figure 3.2. Thus, the largest part of computational load occurs at the BS. In other words, only the measurement process of observation vectors $\mathbf{y}_\tau$ is carried out at the UE. Upon reception of such measurements from the UE, the serving BS applies the K-NN procedure, with the assumption of the HUD be available, to predict the state vector of the reported observation vector. Also note that, in practice, the HUD is built up from measured observation vectors and their (desirably precise and accurate) estimated state vectors. Thus, the state evolution model does not need to be known. The computational complexity of the K-NN method can then be defined as $O(N_s N_b)$ where $N_b = N_t N_r$ is the number of beam pairs.

Figure 3.2 – K-NN Tracking Framework.



Source: Created by the author.

3.4 Graph signal processing strategies

As discussed in the Section 3.3, an advantage of our K-NN tracking framework is that it does not need the AoD/AoA online estimation phase at the UE. The bulk of the processing is done in the mmWave BS, and the HUD generation is performed in an offline phase, reducing the computational complexity and not interfering in the dynamic tracking process. However, our K-NN method require the whole beam space measure at the UE and an exhaustive use of the feedback channel to report the full observed vector $\mathbf{y}_\tau$ or the full state vector $\mathcal{X}_\tau$ at each time instant τ, respectively. Although it is possible to work with the estimation on the BS and reporting of it to the UE, the limited feedback may be quite a problem and, in a practical environment, the feedback channel is likely to be limited. In addition, at mmWave frequencies, the measurement process can be a challenge due to severe path loss, blockage and other environmental obstructions. In this contex, sampling and reconstruction strategies based on GSP are envisaged.

3.4.1 Graph signal model

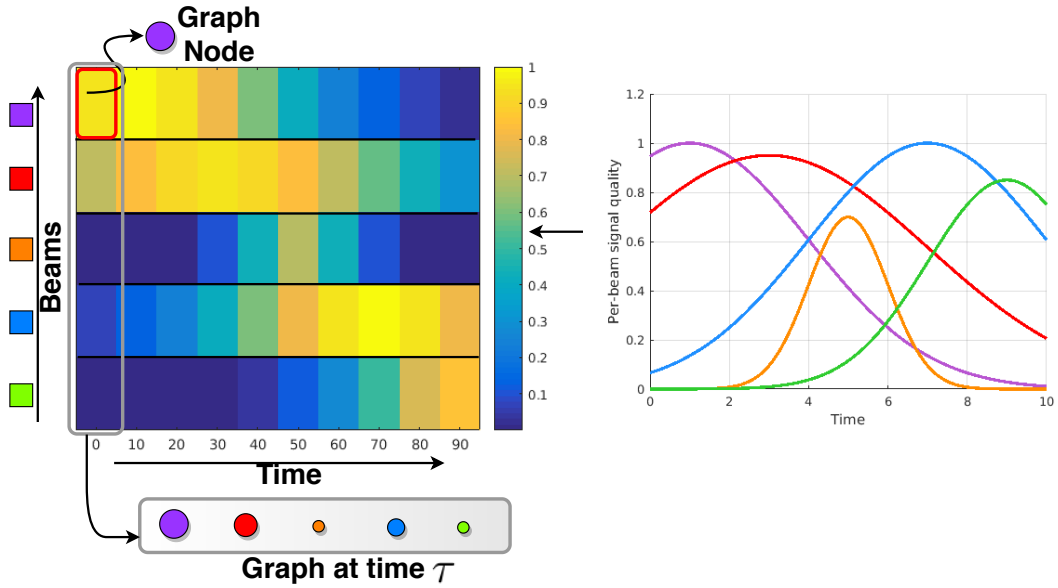
A scenario as described in Section 3.2 is assumed. Once the observation process is carried out, the UE sample the measured signals for the purpose of reporting a sampled observation vector $\mathbf{y}_\tau$ to the mmWave BS, assuming that the feedback channel is limited. Thus, from the BS perspective, the reported UE measurements are by definition a sampled version of the whole beam space measure $\mathbf{y}_\tau$. That is, the reported data are measurements of just a few monitored beams, while the beam space usually comprises a much larger set of beams.

For a better understanding of the observation and sampling process, the measurements of the five beams are represented by a heatmap, in which the temporal window is divided into ten equal time instants, which can be seen in Figure 3.3. The intensity of each beam varies at each time instant, following the sequence of measurements performed while the UE moves around.

In our example in Figure 3.3, the statistics within a temporal window of beam monitoring could be modeled as 10 graphs (one for each time instant τ) with 5 nodes each. Of course, in a more realistic analysis, the temporal window will be divided into a greater number of time intervals (e.g., 2 millisecond or less), better describing the signal quality behavior of the beams over time, and each of those small intervals will correspond to a graph as described.

Once the measurements have been collected, it is visualized that the GSP strategy needs two well-defined steps: 1) learning of the graph structure and connectivity from real data; and 2) a time-varying graph signal sampling and reconstruction step. Finally, our K-NN tracking framework will be used as the prediction step. Essentially, in the first step, the historical observation vectors $\mathbf{y}'_{\tau,j}$ defined in Equation (3.24) and stored in the same HUD will be modeled as a time-varying graph through the use of techniques of learning/estimation of graph structure and connectivity. The second step requires the development of an effective sampling method capable of learning and tracking dynamic graph signals of the time-varying sampling set. The

Figure 3.3 – Observation process over graph.



Source: Created by the author.

chosen sampling method is crucial to the strategy performance, reducing the sampling rate and creating the initial conditions for the reconstruction strategy. Subsequently, based on the implemented sampling method, a reconstruction strategy that collects the data over the time-varying graph and reconstruct the unsampled or nonexistent nodes is carried out.

The construction of the graph structure and connectivity can be estimated taking into account the interdependence of a given beam with respect to the others. Based on the observation vectors $\mathbf{y}'_{\tau,j}$ stored in the HUD, it would be possible to determine how much the variation of one beam affects another beam. The existence or not of an edge between two beams, would depend on the existence of a correlation between the measure values of both vertexes, or if they are independent of each other. If there is dependency, it would be possible to evaluate if the correlation between the measure values is positive or negative, as well as the magnitude of how dependent is a vertex of another. Based on this criterion, the values of the weighted adjacency matrix would be chosen, where higher values of weights would correspond to those edges with greater dependence, and in the case of low correlation, lower values of weights would be chosen.

3.4.2 Learning of the graph from GGL method

In this section, the graph structure and connectivity is learned from the GGL algorithm described in Section 2.4.1. Specifically, the structure and connectivity of the time-varying graph is estimated from real data that represent measurements of the received signal quality per beam and over time. The criterion used to construct the graph is the correlation between the measure values of the vertexes.

From the HUD defined in (3.24), it can be obtained the arithmetic mean of the

collected N_s observation vectors $\mathbf{y}'_\tau$ as

$$\bar{\mathbf{y}}_\tau = \frac{1}{N_s}\sum_{j=1}^{N_s}\mathbf{y}'_{\tau,j}\,. \tag{3.27}$$

Then, let $\bar{\mathbf{Y}} = \begin{bmatrix}\bar{\mathbf{y}}_1 & \bar{\mathbf{y}}_2 & \cdots & \bar{\mathbf{y}}_{N_\tau}\end{bmatrix}$ be a matrix whose rows correspond to vertexes of the graph signal stored in the HUD and over time, and whose columns represent the graph signal evaluated at different time instants τ. The graph Laplacian matrix $\mathbf{L}$ is learned from a data statistics matrix $\mathbf{S}$, which is used as the input of the learning GGL algorithm. Note that, in this problem, $\mathbf{L}$ and $\mathbf{S}$ matrices have the same dimension $N_b \times N_b$.

The data statistics matrix $\mathbf{S}$ is obtained by using a unit-variance Gaussian kernel function. To this end, the vector $\mathbf{s}_i$ defined in Equation (2.14) can be rewritten as

$$\mathbf{s}_i = \begin{bmatrix}\left[\bar{\mathbf{Y}}\right]_{i,1} & \left[\bar{\mathbf{Y}}\right]_{i,2} & \cdots & \left[\bar{\mathbf{Y}}\right]_{i,N_\tau}\end{bmatrix}^T, \quad i = 1,\ldots,N_b\,,$$

and the entries of matrix $\mathbf{S}$ are finally defined as in Equation (2.15), with $i, j = 1,\ldots,N_b$.

The GGL problem formulation is performed as described in Section 2.4.1, in which the target matrix $\mathbf{L}$ is obtained from the minimization optimization problem defined in Equation (2.16). To solve the optimization problem in (2.16), an iterative block-coordinate descent algorithm [91] was implemented. The use of this algorithm allows to decompose the original optimization problem into a series of lower-dimensional subproblems that facilitate their solution. Basically, the algorithm obtains each block-coordinate update (iteration) through fixing some of the elements in the set of target variables and updating the rest. For the specific GGL optimization problem that concerns us, the available structural constraints are also incorporated into the subproblem and the update variables are shaped by a row/column of the target $\mathbf{L}$ matrix at each block-coordinate. Specifically, the algorithm iteratively update rows/columns of $\mathbf{L}$ matrix and its inverse $\mathbf{L}^{-1}$, repeating the cycle of N_b row/column updates until a convergence to the global optimal point is achieved after at most N_b iterations for the worst-case. However, based on convergence criterion used in [35] with a certain error tolerance, a sufficiently accurate solution can be obtained after fewer iterations. Depending on the sparsity of graph solutions, each block-coordinate iteration has $O(\varsigma(s)+(N_b)^2)$ complexity [35] where the term $(N_b)^2$ is due to updating $\mathbf{L}^{-1}$ and $\varsigma(s)$ is the complexity of solving the subproblem with dimension s. For the worst-case, $\varsigma(s)$ has dimension N_b, but usually s is the maximum number of edges connected to a vertex, which is less than N_b.

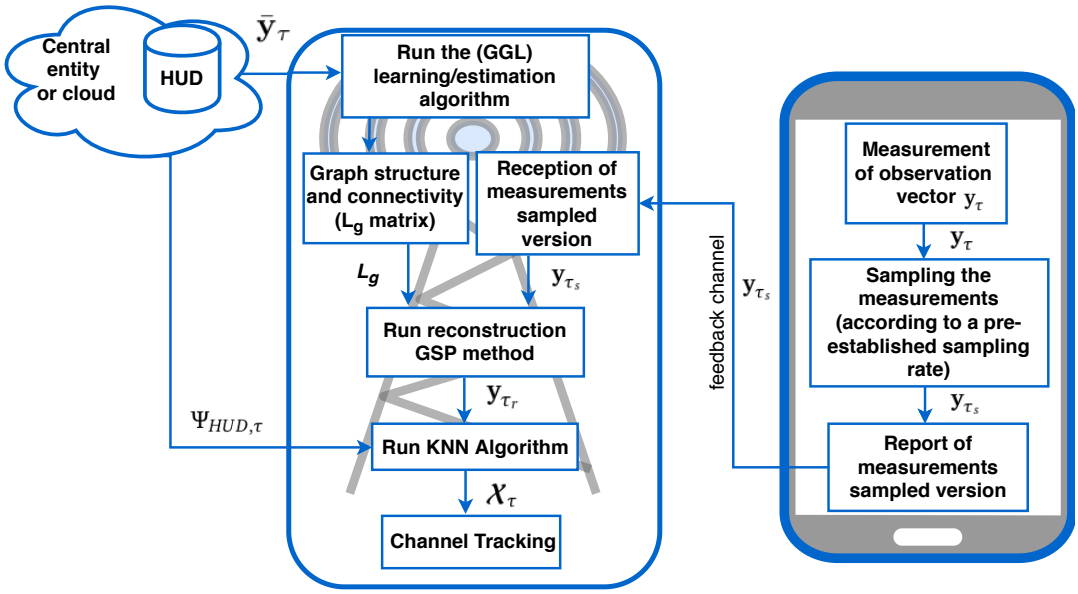
However, for the specific problem that concerns us, the GGL algorithm is carried out offline (similar to the HUD update process), not interfering or aggregating computational complexity in the dynamic tracking process.

3.4.3 Time-varying graph signal sampling and reconstruction

In the previous section, the structure and connectivity of the Laplacian graph from the HUD stored in a central entity or in the cloud were estimated. Then, it is assumed that the

obtained Laplacian matrix $\mathbf{L}$ is always available in the mmWave BS. Moreover, an observation process is carried out at the UE at every time instant τ. Assuming that the feedback channel is limited, the UE may need to sample the measured signals for the purpose of reporting to the mmWave BS, being necessary a sampling mechanism that ensures that the reported UE measurements are a sampled version of the received signal. That is, the reported vector $\mathbf{y}_{\tau_s}$ are measurements of just a few monitored beams, while the beam space usually comprises a much larger set of beams. Subsequently, based on the implemented sampling method in the UE and with the knowledge of the Laplacian matrix, a reconstruction strategy is carried out in the mmWave BS, which collects the data over the time-varying graph and reconstruct the unsampled or nonexistent vertexes. The reconstructed measurement vector $\mathbf{y}_{\tau_r}$ can then be used as the input of the K-NN tracking strategy. A summary of our proposed K-nearest neighbors with reconstruction (K-NN-R) tracking framework based on GSP can be observed in Figure 3.4.

Figure 3.4 – Proposed K-NN-R tracking strategy process.



Source: Created by the author.

In this chapter, it is assumed a random sampling mechanism on the UE side where there are measurements of just a few monitored beams according to a sampling rate R_s. For example, for a beam space of 8×8 beam pairs, a sampling rate R_s of 25% means that the UE receives (or observes) only 2 beam pairs at each time instant. On the other hand, the reconstruction technique is assumed to be implemented on the BS side. The used reconstruction strategy will be discussed more extensively in the following.

3.4.4 Time-varying graph signal reconstruction strategy

Let $\mathbf{Y}_\tau = \begin{bmatrix} \mathbf{y}_1 & \mathbf{y}_2 & \cdots & \mathbf{y}_{N_\tau} \end{bmatrix}$ be the measured time-varying graph signal matrix, where each row represents a vertex with measurements $\mathbf{y}_\tau$ as a graph signal, and each column denotes the temporal window N_τ on the corresponding vertex, and in which the graph signal

$\mathbf{y}_\tau$ is measured by the UE at different time instant τ. Then, assuming the existence of the HUD defined in (3.24), and the knowledge of the Laplacian matrix ($\mathbf{L}$) in the mmWave BS, a batch reconstruction method for time-varying graph signals is adopted in this chapter. Specifically, the smooth graph-time difference (SGTD) batch reconstruction strategy described in Section 2.7.1.3 is employed. Recalling Section 2.7.1.3, the SGTD strategy is based on the smoothness of the temporal difference and requires that the UE obtains the sampled beam measurements within a predefined temporal window (N_τ). Thus, with the help of the Laplacian matrix, the batch reconstruction is carried out in the mmWave BS, reconstructing them all together.

Let $\mathbf{Y}_{\tau_s} = \begin{bmatrix} \mathbf{y}_{s_1} & \mathbf{y}_{s_2} & \cdots & \mathbf{y}_{s_{N_\tau}} \end{bmatrix}$ be the reported matrix whose columns represent the sampled graph signal vectors $\mathbf{y}_{\tau_s}$ within the temporal window N_τ. Then, using the definition in Equation (2.23), and the general SGTD reconstruction problem, the Equation (2.31) can be rewritten as

$$\hat{\mathbf{Y}}_{\tau_r}^{\text{SGTD}} = \arg\min_{\mathbf{Y}_\tau} ||\mathbf{M}_\tau \odot \mathbf{Y}_\tau - \mathbf{Y}_{\tau_s}||_F^2 + \gamma \text{Tr}\left([\mathbf{Y}\nabla]_\tau^T \mathbf{L} [\mathbf{Y}\nabla]_\tau\right) . \tag{3.28}$$

where γ is the regularization parameter and the resulting reconstructed matrix $\hat{\mathbf{Y}}_{\tau_r}^{\text{SGTD}}$ is used as the input of the K-NN tracking strategy. To solve the problem in (3.28), a conjugate gradient algorithm [84] is used. Specifically, at each iteration, the algorithm works in two well-defined steps: 1) the determination of stepsize, and 2) the update of the next search direction. For the specific problem that concerns us, the global optimal point is usually found after at most N_b iterations. Although when N_b is large, a sufficiently accurate solution can be obtained after fewer iterations based on the stopping criteria used in [49] with a certain error tolerance. Thus, the algorithm has complexity $O(N_b^2)$ for the worst-case [84]. Additionally, the solution of (3.28) is the signal with the smoothest temporal difference, and not strictly the original signal. This means that there may be a signal which is the same as the original one at sampled points, but with smoother temporal difference than the original signal.

3.5 Simulation results

In this section, it is numerically investigated the performance of the proposed tracking framework. The system model as described in Section 3.2 is considered, where a single moving UE is served in the downlink direction by one BS. Additionally, analog beamforming is assumed. The system, channel and simulation parameters are summarized in Table 3.1, while the tracking algorithms parameters are listed in Table 3.2.

Regarding the general simulation parameters, the number of paths (i.e., scatterer clusters) equals 4 and the center frequency is 28 GHz. There is a single BS serving one UE and the SNR is set to 20 dB. A ULA with 8 antenna elements equips the UE, while the BS is equipped with a ULA with 32 or 64 antenna elements. Each ULA has element spacing equal to half-wavelength and its elements radiate energy omnidirectionally. As for channel parameters, the initial AoD and AoA angles were generated following a $\mathcal{U}\left(-\frac{\pi}{2}, \frac{\pi}{2}\right)$ and the initial path gains following a $\mathcal{CN}\left(0, \frac{N_b}{L}\right)$. At last, the total number of simulation runs is for averaging purposes.

Table 3.1 – Channel, system and simulation parameters.

Center frequency [GHz]	28
Number of BSs	1
Number of UEs	1
signal-to-noise ratio (SNR) [dB]	20
BS antenna array	ULA 32 or 64
UE antenna array	ULA 8
Antenna radiation pattern	Omnidirectional
Antenna element spacing d	Half-wavelength
Number of scatterer clusters	4
Number of simulation runs	2000
Codebook	DFT

Source: Created by the author.

Table 3.2 – Tracking algorithms setup.

Time correlation coefficient ρ	0.995
Angular std. deviation σ_u [deg.]	0.5
Reporting temporal window N_τ (multiple of τ)	50
Observed temporal window $\acute{N_\tau}$ ($\acute{N_\tau} < N_\tau$)	5, 10, 20 or 30
Baseline algorithms	LS, CS and EKF
CS solving algorithm	OMP
Grid size G (OMP)	180
K values (K-NN)	$[1, 20]$
Similarity metric	Frobenius norm
Average metric	Arithmetic mean
HUD size N_s	100
Correlation factor β (%)	10, 15 or 25
Max. random delay Δ (multiple of τ)	5
Sampling rate R_s (%)	25 or 50

Source: Created by the author.

With respect to the state evolution parameters, the standard deviation σ_u equals 0.5°, which in practice dictates how fast the angles vary [15]. Although this angular variation seems very small, the period between two consecutive time instants τ takes a few milliseconds, e.g., 2 ms or less. This actually corresponds to rather fast angular changes. Besides, the time correlation ρ is set to 0.995, which according to [92] is an adequate value for a moving UE at speed of 5 km/h.

From the channel estimation perspective, the proposed tracking framework is evaluated at each time instant τ during a reporting temporal window N_τ of 50 consecutive τ. Along this period, the UE has a random movement profile following the state evolution model. Regarding the K-NN algorithm, the number of neighbors K varies from 1 to 20, the HUD has $N_s = 100$ samples with correlation factor β set to 10, 15 or 25% and random delay $\delta \sim \mathcal{U}(-5, 5)$. The K-NN algorithm uses the Frobenius norm and the arithmetic mean as similarity and average metrics, respectively. For the K-NN-R framework, the random sampling has sampling rate R_s set

to 25 or 50%.

Additionally, the proposed framework is evaluated as a channel prediction tool to obtain the channel matrix $\mathbf{H}$ during a reporting temporal window N_τ, from the measurements collected only throughout an observed temporal window $\acute{N}_\tau$ in which $\acute{N}_\tau < N_\tau$. The temporal window $\acute{N}_\tau$ is set to 5, 10, 20 or 30 consecutive time instants τ, while $N_\tau = 50$ is fixed.

3.5.1 Baseline approaches

As classical/baseline approaches, three well-known prediction algorithms found in the literature are adopted, i.e., LS [93], CS [93] and EKF [15], which are compared with our proposed framework in terms of complexity and performance.

3.5.1.1 Classical LS

Consider the algebraic identity $\text{vec}(\mathbf{ABC}) = (\mathbf{C}^T \otimes \mathbf{A}) \cdot \text{vec}(\mathbf{B})$, where $\otimes$ represents the product of Kronecker. Equation (3.15) can be rewritten as $\mathbf{y}_\tau = \mathbf{Q}\mathbf{h}_\tau + \text{vec}(\mathbf{W}^H\mathbf{V})$, where $\mathbf{h}_\tau = \text{vec}(\mathbf{H}_\tau)$ and $\mathbf{Q} = \sqrt{P}(\mathbf{F}^T \otimes \mathbf{W}^H) \in \mathbb{C}^{N_b \times N_b}$ with transmit power P. The channel estimation can be obtained directly by LS as $\mathbf{h}_\tau = (\mathbf{Q}^H\mathbf{Q})^{-1}\mathbf{Q}^H\mathbf{y}_\tau$ [93].

3.5.1.2 CS using OMP

Assuming the CS method based on orthogonal matching pursuit (OMP) implemented in [93], first a set of discrete angles is selected, which can be defined as $\mathbf{\Phi}_G = \{\phi_g \in [0, \pi) : g = 1, 2, \ldots, G\}$, which are uniformly distributed and are an approximation of the channel angular domain. The size of the grid G must be $G >> L$ to achieve the desired resolution. Then, it can be defined the array response matrices $\hat{\mathbf{A}}_t \in \mathbb{C}^{N_t \times G}$ and $\hat{\mathbf{A}}_r \in \mathbb{C}^{N_r \times G}$ whose columns entries are the array response vectors corresponding to the candidate angles in $\mathbf{\Phi}_G$. Thus, the channel model in (3.3) can be approximated in matrix form as $\mathbf{H}_\tau \cong \hat{\mathbf{A}}_r\hat{\mathbf{H}}_a\hat{\mathbf{A}}_t^H$, where $\hat{\mathbf{H}}_a \in \mathbb{C}^{G \times G}$ is an L-sparse matrix. Equation (3.15) can be rewritten as $\mathbf{y}_\tau = \hat{\mathbf{Q}}\hat{\mathbf{h}}_a + \text{vec}(\mathbf{W}^H\mathbf{V})$, where $\hat{\mathbf{Q}} = (\hat{\mathbf{A}}_t^H\mathbf{F})^T \otimes (\mathbf{W}^H\hat{\mathbf{A}}_r) \in \mathbb{C}^{N_b \times G^2}$. Finally, the vector $\hat{\mathbf{h}}_a = \text{vec}(\hat{\mathbf{H}}_a)$ is obtained with the OMP sparse signal recovery technique. In this work, $G = 180$ is adopted as in [93].

3.5.1.3 Kalman filter using EKF

The EKF prediction technique [25] is the suboptimal KF variation that recursively searches for the estimated state $\hat{\mathcal{X}}_\tau$ of $\mathcal{X}_\tau$, based on the previous estimate $\hat{\mathcal{X}}_{\tau-1}$ and the current observations $\mathbf{y}_\tau$. Specifically, the EKF approximates nonlinearities by making linear approximations around the current state. Thus, the Jacobian matrix $\mathbf{C}_\tau$ is substituted into the normal KF equations for the linear transformations. Then, it is necessary to calculate the partial derivative matrix $\mathbf{C}_\tau \in \mathbb{C}^{N_b \times D_L}$, in which the rows and columns represent the derivatives of $\mathbf{g}_\tau(\mathcal{X}_\tau)$ with respect to the variables $\boldsymbol{\phi}$ and $\boldsymbol{\alpha}$. In this work, the EKF algorithm is implemented based on [15], in which as input parameters the initial AoD and AoA angles as well as the initial

path gains were generated following the same channel parameters. The state evolution model $\mathcal{X}_\tau = \delta(\mathcal{X}_{\tau-1}) + \mathbf{u}_{\tau-1}$ is assumed, where $\mathbf{u}_{\tau-1} \sim \mathbb{C}\mathcal{N}(0, \mathbf{Q}_\tau)$, and $\mathbf{Q}_\tau = \sigma_u^2 \mathbf{I}_{D_L}$. As for the measurement model, $\mathbf{y}_\tau = \gamma(\mathcal{X}_\tau) + \mathbf{v}_\tau$, where $\mathbf{v}_\tau \sim \mathbb{C}\mathcal{N}(0, \mathbf{R}_\tau)$, and $\mathbf{R}_\tau = \sigma_v^2 \mathbf{I}_{2N_b}$, for $\sigma_v = \sqrt{\frac{P}{2}} 10^{\frac{\text{SNR}}{20}}$.

The EKF works in two steps: i) a prediction or forecast step, and ii) an update or data assimilation step. Specifically, in the first step the predicted state and the predicted covariance $\mathbf{P}_\tau$ are estimated:

$$\begin{aligned} \mathcal{X}_{\tau|\tau-1} &= \delta\left(\hat{\mathcal{X}}_{\tau-1|\tau-1}\right), \\ \mathbf{P}_{\tau|\tau-1} &= \mathbf{J}_{\tau-1}\mathbf{P}_{\tau-1|\tau-1}\mathbf{J}_{\tau-1}^T + \mathbf{Q}_{\tau-1}, \end{aligned} \tag{3.29}$$

where $\mathbf{J}_{\tau-1}$ is the Jacobian matrix of the evolution function δ. In the second step, the variables are updated based on the observation vector $\mathbf{y}_\tau$:

$$\begin{aligned} \mathbf{K}_\tau &= \mathbf{P}_{\tau|\tau-1}\mathbf{C}_\tau^T \left(\mathbf{C}_\tau \mathbf{P}_{\tau|\tau-1}\mathbf{C}_\tau^T + \mathbf{R}_\tau\right)^{-1}, \\ \hat{\mathcal{X}}_{\tau|\tau} &= \mathcal{X}_{\tau|\tau-1} + \mathbf{K}_\tau \left(\mathbf{y}_\tau - \gamma\left(\mathcal{X}_{\tau|\tau-1}\right)\right), \\ \hat{\mathbf{P}}_{\tau|\tau} &= \left(\mathbf{I}_{D_L} - \mathbf{K}_\tau \mathbf{C}_\tau\right)\mathbf{P}_{\tau|\tau-1}. \end{aligned} \tag{3.30}$$

3.5.2 Complexity analysis

Usually, the computational complexity of the beam tracking algorithms is strongly linked to the number of beam pairs N_b. This issue is especially critical in mmWave communication in which N_b is usually large.

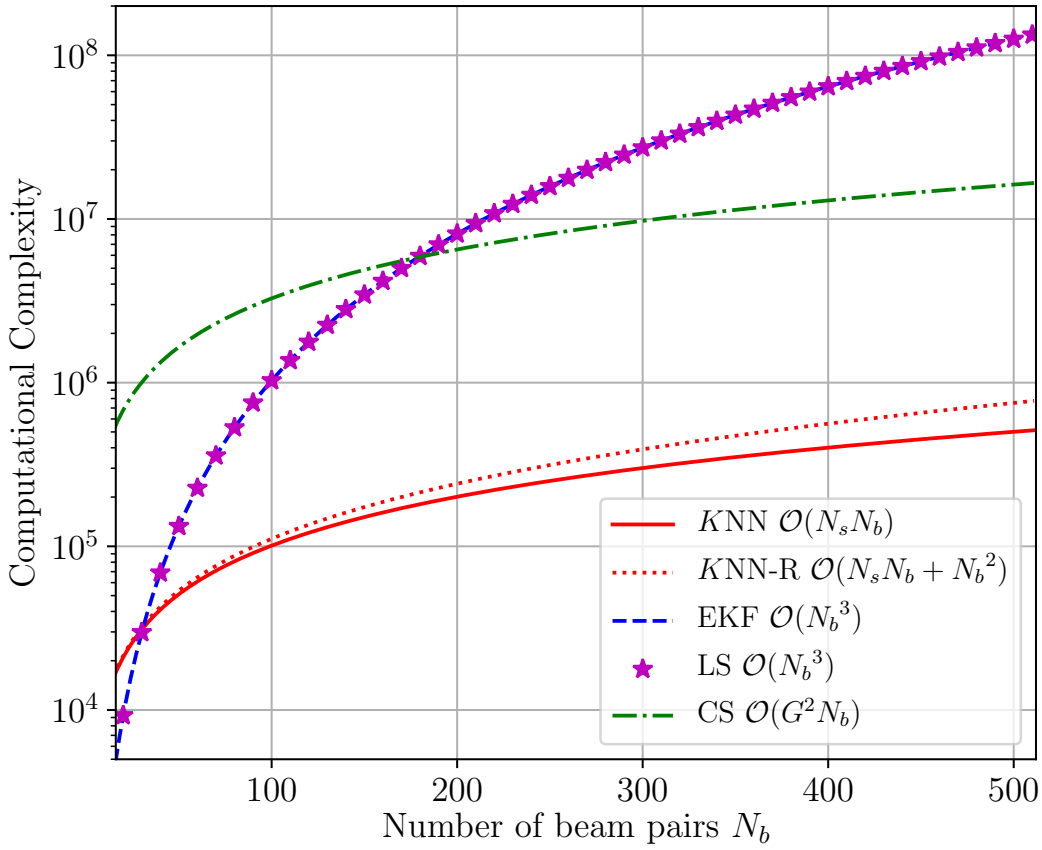
Recalling Section 3.3, the complexity of K-NN algorithm has order of magnitude $O(N_s N_b)$. Note that, complexity of K-NN directly depends on N_s, i.e., the amount of training data available. On the other hand, while the GGL learn process is carried out offline, the complexity of reconstruction algorithm based on GSP, which is described in Section 3.4.4, has complexity $O(N_b^2)$. Thus, the complexity of K-NN-R is $O(N_s N_b + N_b^2)$.

Moreover, the computational complexity of the LS and CS methods are $O(N_b^3)$ and $O(G^2 N_b)$, respectively, according to [93]. In the case of LS, this is due to inverse operation of $\mathbf{Q}^H\mathbf{Q}$ and the large number of antenna elements. Concerning CS complexity, since the AoDs and AoAs are generated from a continuous uniform distribution, increasing the number of grid points G in the CS method based on OMP can improve the estimation performance. However, using large G in OMP algorithm leads to heavy computational load.

At last, following the complexity analysis in [94], it is calculated the complexity of the EKF algorithm for our mmWave scenario. Specifically, the complexity of EKF has order of magnitude $O(N_b^3)$, mainly due to the multiplication and inverse operations of the Jacobian $\mathbf{C}_\tau$ in the update step.

Now, in order to evaluate the different tracking algorithms in terms of complexity, the HUD with $N_s = 1,000$ is set, the number of grid points $G = 180$ and ULA with 64 elements at BS and 8 elements at UE. Figure 3.5 illustrates the computational complexity of all tracking algorithms against the number of beam pairs N_b. As can be seen, K-NN and K-NN-R exhibits the lowest complexity for $N_b > 10$. Besides, as will be seen in Section 3.5.6, $N_s = 100$ is enough

Figure 3.5 – Computational complexity of the tracking algorithms for $N_s = 1000$ and $G = 180$. The antenna configuration is ULA with 64 elements at BS and 8 elements at UE.



Source: Created by the author.

for K-NN to outperform the baseline algorithms in terms of estimation error. Therefore, either increasing N_s or decreasing G can only enlarge the performance gap between, for instance, K-NN and CS. In order words, for large N_b the K-NN algorithm and our proposed K-NN-R framework are less complex than the baseline algorithms herein evaluated.

3.5.3 K-NN-R tuning

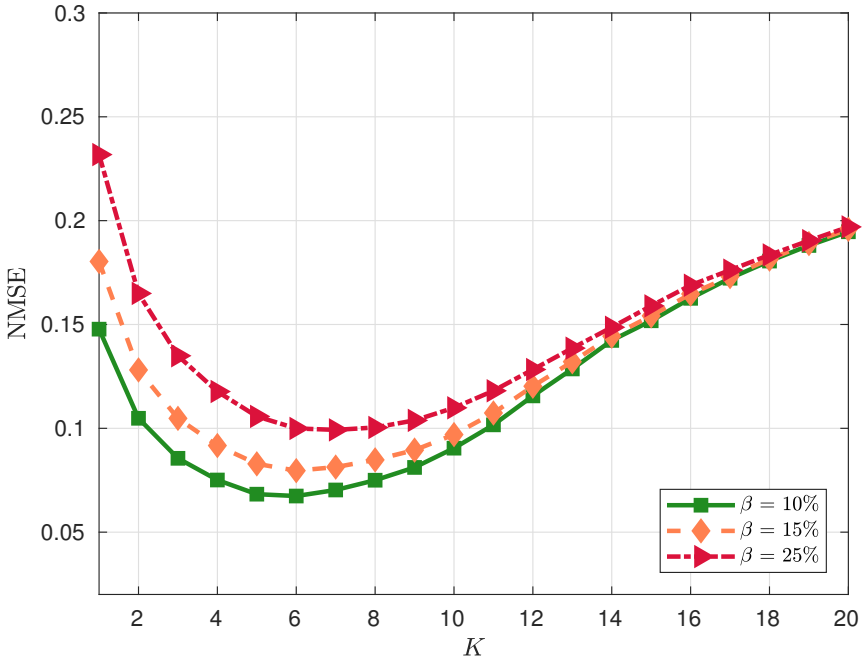
A cross-validation procedure was carried out to find the K value that minimizes the estimation and prediction errors in terms of NMSE for the simulation scenario described above. To this end, the NMSE is averaged over the temporal window N_τ, as follows below:

$$\text{NMSE} = \frac{1}{N_\tau} \sum_{\tau=1}^{N_\tau} \frac{||\mathbf{H}(\mathcal{X}_\tau) - \mathbf{H}(\hat{\mathcal{X}}_\tau)||_F^2}{||\mathbf{H}(\mathcal{X}_\tau)||_F^2}.$$

Regarding K-NN, Figure 3.6 shows that for any value of β the best value of K is 6. To capture only the effect of β in the estimation error, no sampling or reconstruction is applied. Additionally, one may observe that some increase in the algorithm performance is observed as

the value of β decreases. This makes sense because β controls the correlation between HUD samples, according to Equation (3.23).

Figure 3.6 – Cross-validation K-NN for different values of β showing estimation error against K. There are 64 elements at BS and 8 elements at UE.



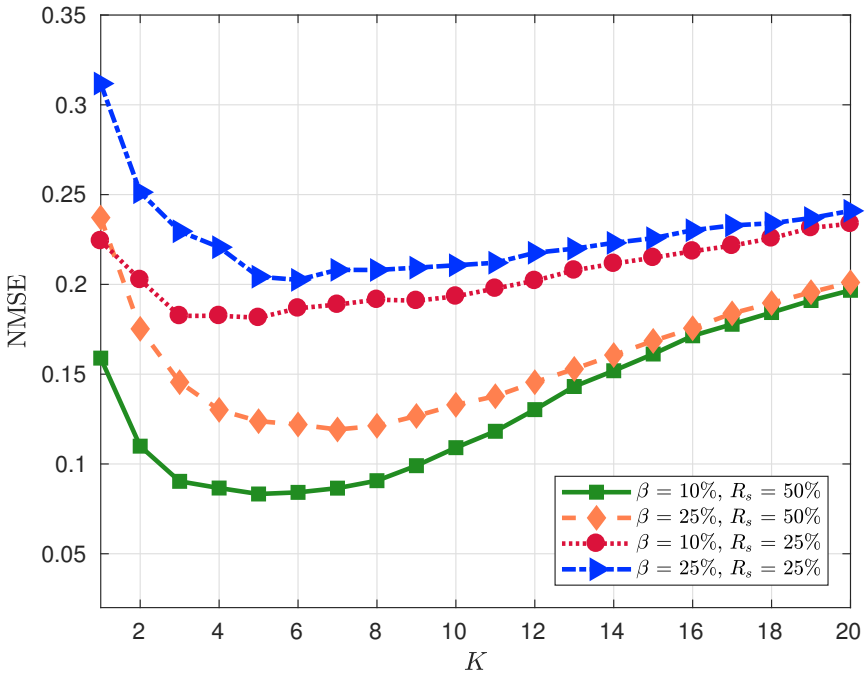
Source: Created by the author.

A further cross-validation was performed by varying the sampling rate R_s along with β regarding K-NN-R. The stopping criteria for the GGL algorithm discussed in Section 3.4.2 are such that the maximum number of iterations is 10^2 and the error tolerance is 10^{-6}. Besides, for the batch reconstruction discussed in Section 3.4.3, the stopping criteria are such that the maximum number of iterations is 10^4 and the error tolerance is 10^{-6}. As expected, the higher R_s the lower NMSE, for any value of K, which is shown in Figure 3.7. Regardless of the β value, curves referring to $R_s = 50\%$ show lower estimation error than the ones referring to $R_s = 25\%$. Besides, as previously seen, the lower β the lower the estimation error. Moreover, the best K value found in the previous cross-validation, i.e., $K = 6$, maintains its status for $R_s = 50\%$ and $\beta = 10\%$, while the best value of K varies for the other combinations of β and R_s but stays around $K = 6$, reaching values between $K = 5$ and $K = 7$.

On the other hand, a cross-validation procedure was carried out for the prediction case by varying the observed temporal window $\acute{N}_\tau$. Analyzing only the K-NN algorithm, Figure 3.8 illustrates that for any value of $\acute{N}_\tau$ the best value of K is also 6. Additionally, one may observe that the higher $\acute{N}_\tau$ the lower prediction NMSE. This makes sense because the K-NN algorithm can have an improved learning experience with more training data.

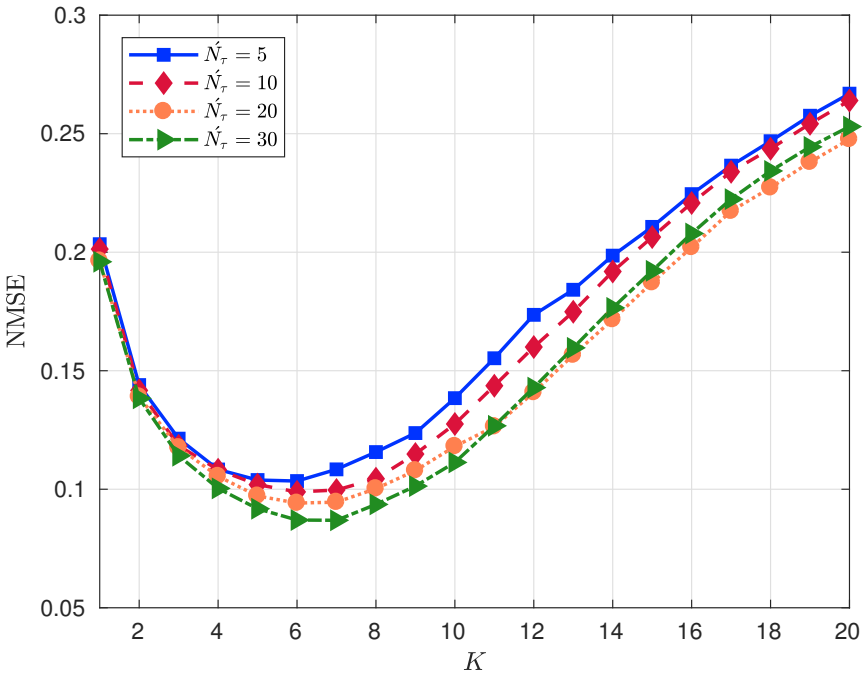
Setting the same parameter values for the GGL and batch reconstruction algorithms used above, Figure 3.9 shows the effect of the $\acute{N}_\tau$ in K-NN-R framework for different sampling

Figure 3.7 – Cross-validation of K-NN-R for different values of β and R_s, showing estimation error against K. There are 64 elements at BS and 8 elements at UE.



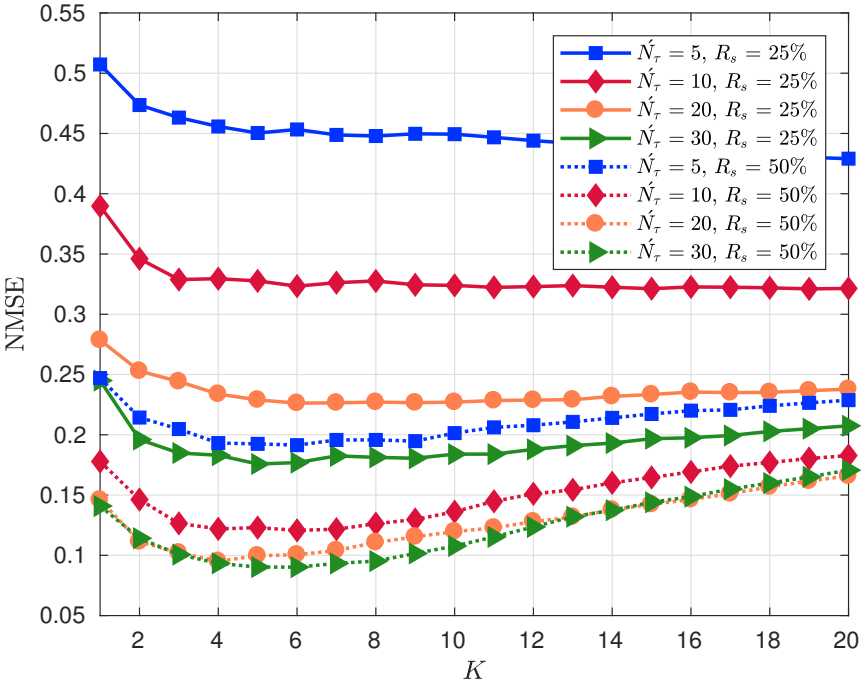
Source: Created by the author.

Figure 3.8 – Cross-validation of K-NN for $\beta = 10\%$ and different values of $\acute{N}_\tau$ showing prediction error against K. There are 64 elements at BS and 8 elements at UE.



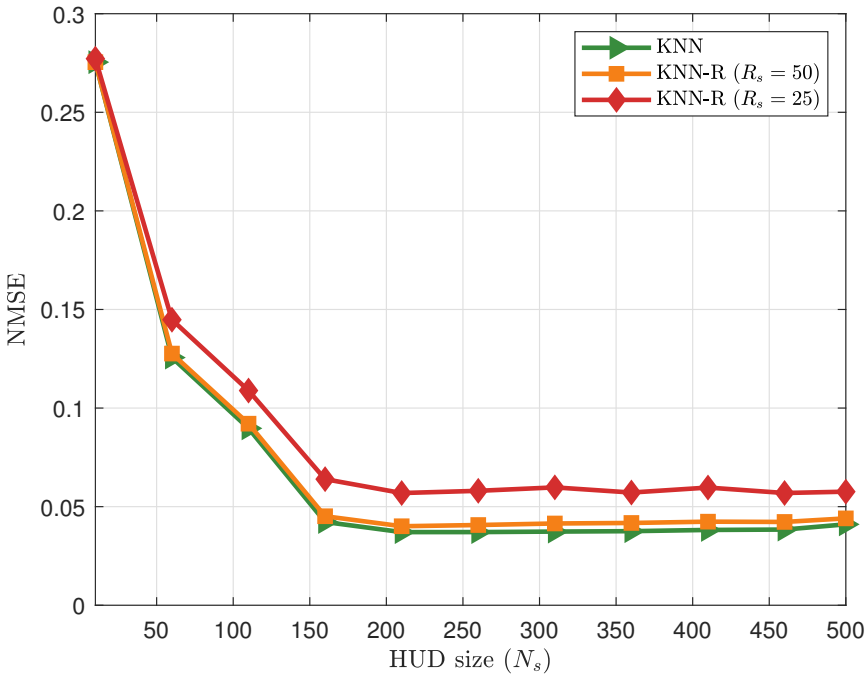
Source: Created by the author.

Figure 3.9 – K-NN-R cross-validation for $\beta = 10\%$, $R_s = 25\%, 50\%$ and different values of $\acute{N}_\tau$, showing prediction error against K. There are 64 elements at BS and 8 elements at UE.



Source: Created by the author.

Figure 3.10 – Analysis for different HUD size in tracking of the channel matrix $\mathbf{H}$, for $\beta = 10\%$, $K = 6$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE.



Source: Created by the author.

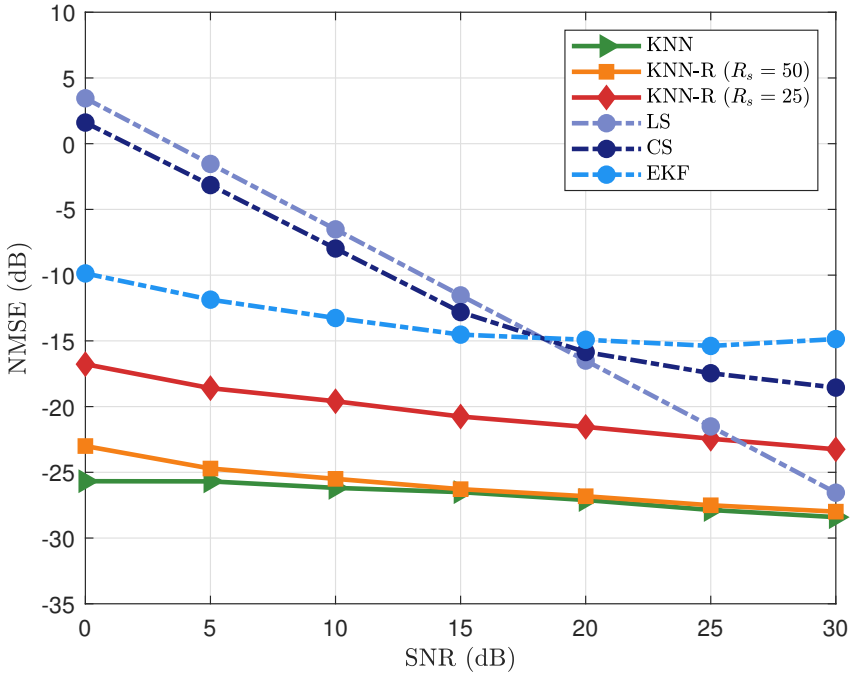
rates, i.e., $R_s = 25\%$ and $R_s = 50\%$. As expected, considering similar $\acute{N}_\tau$ values, the curves that represent $R_s = 50\%$ show lower prediction error than the ones refering to $R_s = 25\%$. Besides, the best value of K stays around $K = 6$ for all combinations.

Finally, the performance of the KNN and KNN-R algorithms is evaluated by varying the the amount of training data available. Figure 3.10 illustrates the NMSE per HUD size N_s for $\beta = 10\%$, $K = 6$, based on the cross-validation performed above. As expected, the higher the HUD size the lower NMSE for both KNN and KNN-R algorithms. Moreover, the K-NN algorithm is just slightly better than the proposed K-NN-R framework with $R_s = 50\%$ for all N_s value,s and K-NN-R with $R_s = 25\%$ performs worse than all. However, it is interesting to note that the three curves provides the same converging characteristic, that is, converge for $N_s = 150$.

3.5.4 SNR influence analysis

To assess the SNR influence in the different tracking algorithms, the values of K and β based on the cross-validation performed above are set, i.e., $K = 6$ and $\beta = 10\%$. In terms of the estimation error, K-NN and K-NN-R are compared with the three baseline tracking algorithms defined in Section 3.5.1.

Figure 3.11 – SNR analysis in tracking of the channel matrix $\mathbf{H}$ for $\sigma_u = 0.5°$, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE.



Source: Created by the author.

Figure 3.11 illustrates the NMSE per SNR of all tracking algorithms for $\sigma_u = 0.5°$. As can be seen, the higher the SNR value the lower NMSE for all the tracking algorithms. Note that CS performed slight better than LS for an SNR regime below 15 dB. This is because

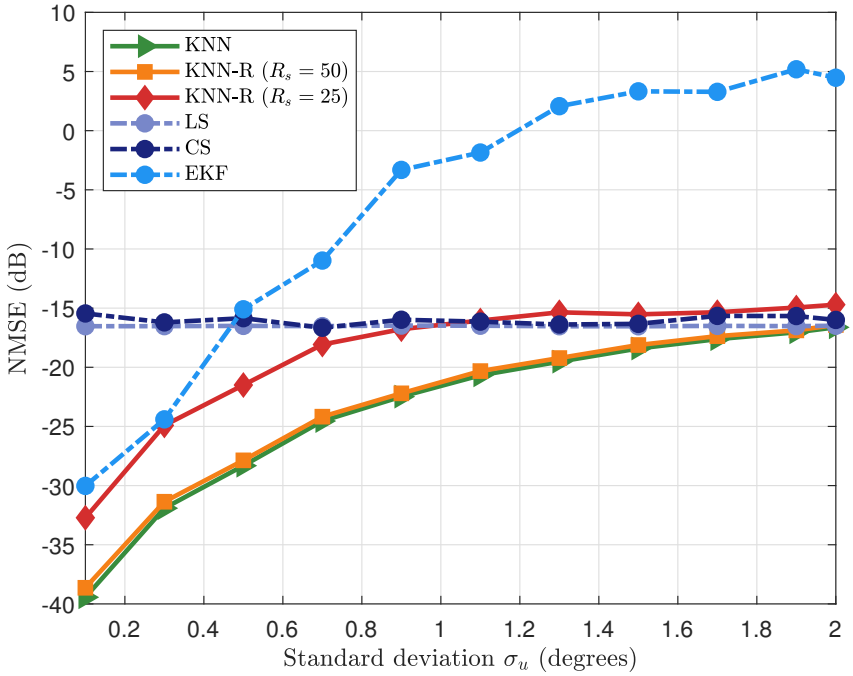
CS looks for maximum correlation to array responses within a dictionary, which makes it less susceptible to noise errors than LS for lower SNR values. Moreover, the EKF algorithm exhibits a significant performance gain than CS and LS in low SNR regime. However, it is interesting to note that for SNR values greater or equal than 20 dB, both CS and LS perform better than EKF.

Additionally, still in Figure 3.11, both K-NN and K-NN-R framework provide a lower NMSE than the three baseline tracking algorithms, independently of the SNR value. Only the LS method provided a lower NMSE than K-NN-R for $R_S = 25\%$ for the highest SNR value (i.e., 30 dB), but with greater complexity. Moreover, the K-NN algorithm is just slightly better than the proposed K-NN-R framework with $R_s = 50\%$ for all SNR values.

3.5.5 Standard deviation and UE speed influence analysis

This section analyzes the influence of the standard deviation σ_u and the mean speed of the UE in the different tracking algorithms. The values of K and β are also based on the cross-validation performed above, i.e., $K = 6$ and $\beta = 10\%$. Specifically, the K-NN and K-NN-R frameworks are compared with the three baseline tracking algorithms defined in Section 3.5.1, in terms of the estimation error for SNR = 20 dB.

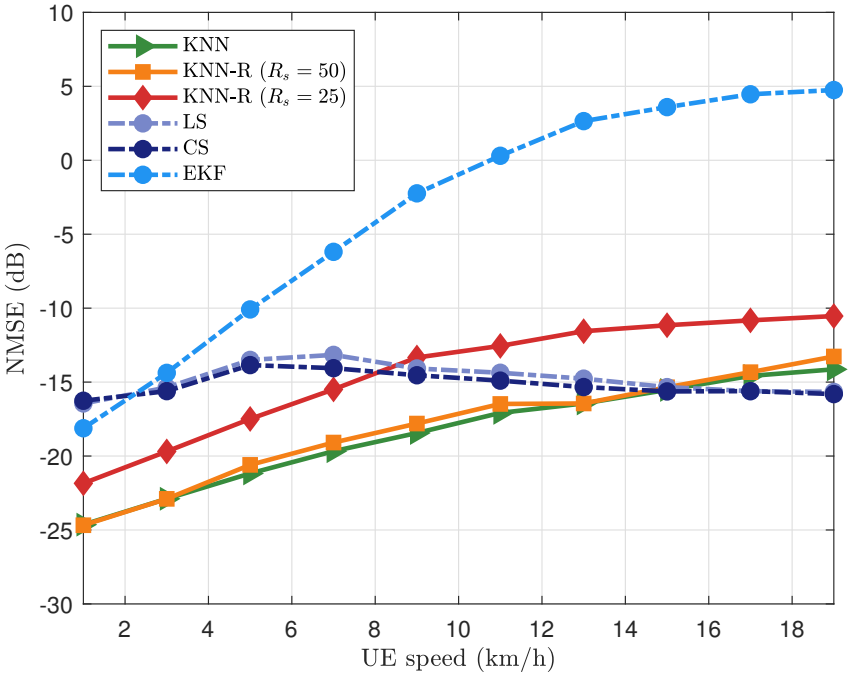
Figure 3.12 – Standard deviation (σ_u) analysis in tracking of the channel matrix $\mathbf{H}$ for SNR = 20 dB, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE.



Source: Created by the author.

Figure 3.12 illustrates the NMSE per σ_u of all tracking algorithms. It is assumed an UE speed of 5 km/h and standard deviation values between $\sigma_u = 0.1°$ and $\sigma_u = 2°$. As can be seen, the angular variation has little influence on the performance of both LS and CS strategies, since

Figure 3.13 – UE speed analysis in tracking of the channel matrix $\mathbf{H}$ for SNR = 20 dB, $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are 32 elements at BS and 8 elements at UE.



Source: Created by the author.

these only depend on the scan measurements of the current block. Moreover, the performance of EKF becomes much worse by increasing the σ_u value. This is because rather fast angular changes may cause uncertainties in the state model of the EKF.

Additionally, still in Figure 3.12, the performance of both K-NN and K-NN-R framework also worsens by increasing the σ_u value, due to increased decorrelation between samples at adjacent time points. However, both K-NN and K-NN-R with $R_S = 50\%$ provide a lower NMSE than the three baseline tracking algorithms for all σ_u values. Moreover, even for $R_S = 25\%$, the performance loss of K-NN-R framework is still smaller than the baseline solutions for σ_u values less or equal than 0.9°, and exhibits a similar behavior than both CS and LS for σ_u values greater or equal than 1°.

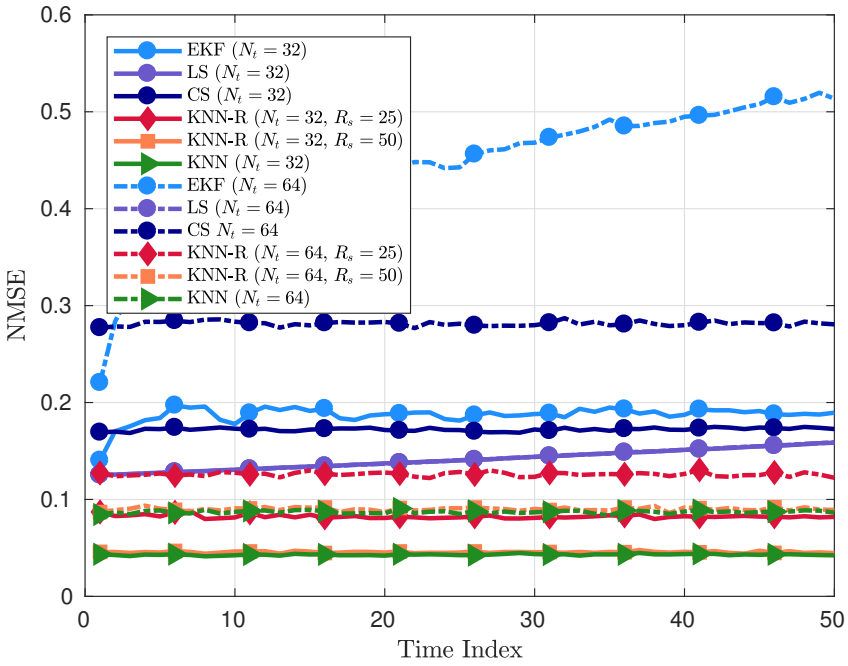
On the other hand, Figure 3.13 illustrates the NMSE per UE speed of all tracking algorithms. UE speed values between $v = 1$ km/h and $v = 20$ km/h are assumed, in which for each speed value was uniformly assigned an σ_u of 0.5° to 1°. Moreover, according to [92], the time correlation ρ was from 0.9993 (for $v = 1$ km/h) to 0.7902 (for $v = 20$ km/h). As can be seen, both LS and CS may suffer some performance loss due to the variation of the channel coefficients. Moreover, the performance of EKF becomes much worse by increasing the UE speed. Further, the performance of both K-NN and K-NN-R framework also worsens by increasing the UE speed, due to increased decorrelation between samples at adjacent time points. However, both K-NN and K-NN-R with $R_S = 50\%$ provide a lower NMSE than the three baseline tracking algorithms for UE speed less or equal than 13 km/h. Moreover, even for $R_S = 25\%$, the K-NN-R framework

have lower NMSE for values of UE speed less or equal than 9 km/h. Note that, these speed values are consistent with the pedestrian speed range.

3.5.6 Channel tracking evaluation

To assess the different tracking algorithms, the values of K and β based on the cross-validation performed above are set, i.e., $K = 6$ and $\beta = 10\%$. In terms of the estimation error per time instant, K-NN and our proposed K-NN-R framework are compared with the three baseline tracking algorithms defined in Section 3.5.1 for different sampling rates R_s.

Figure 3.14 – Tracking of the channel matrix $\mathbf{H}$ for $\beta = 10\%$ and $R_s = 25\%, 50\%$. The antenna configuration is ULA with 32 and 64 elements at BS, and 8 elements at UE.



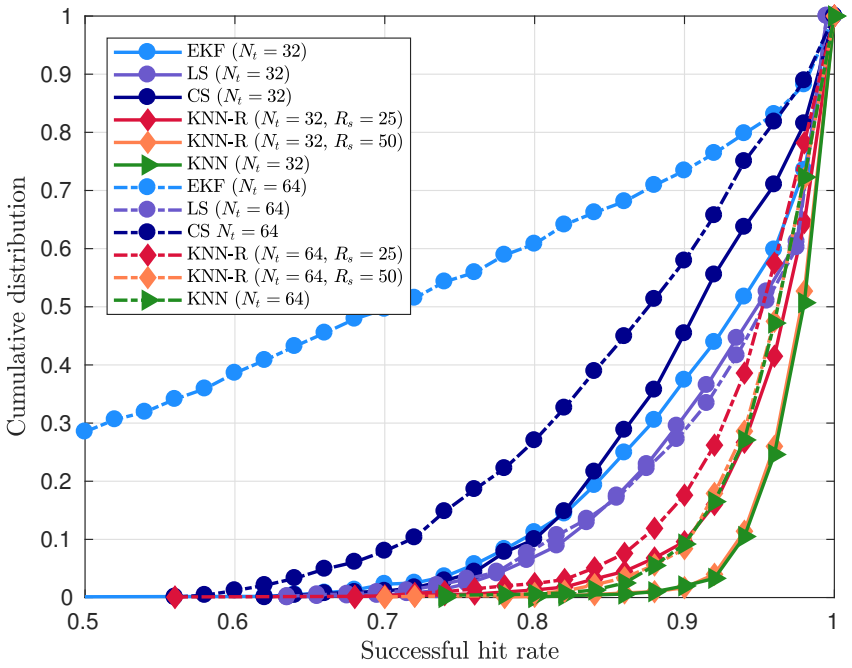
Source: Created by the author.

The tracking of the channel matrix $\mathbf{H}$ for the estimation is evaluated for a ULA array with 32 or 64 elements at the BS and 8 antenna elements at the UE.

Figure 3.14 illustrates the NMSE per time instant of all tracking algorithms along the reporting temporal window N_τ. As can be seen, the lower the number of antennas at the ULA the lower NMSE for all the tracking algorithms. Note that the performance of EKF becomes much worse by increasing the number of antennas at the BS. This is because when the array size becomes larger, the analog beams become narrower. Then, the process noise can lead the AoD/AoA to certain values that cause beam misalignment, which results in poor received signal power at the UE. Consequently, this significantly affects the update step of EKF.

Additionally, still in Figure 3.14, both K-NN and K-NN-R framework provide a lower NMSE than the three baseline tracking algorithms, independently of the number of antennas

Figure 3.15 – Accuracy of the tracking methods finding the best beam for $\beta = 10\%$ and $R_s = 25\%, 50\%$. There are either 32 or 64 elements at BS, and 8 elements at UE.



Source: Created by the author.

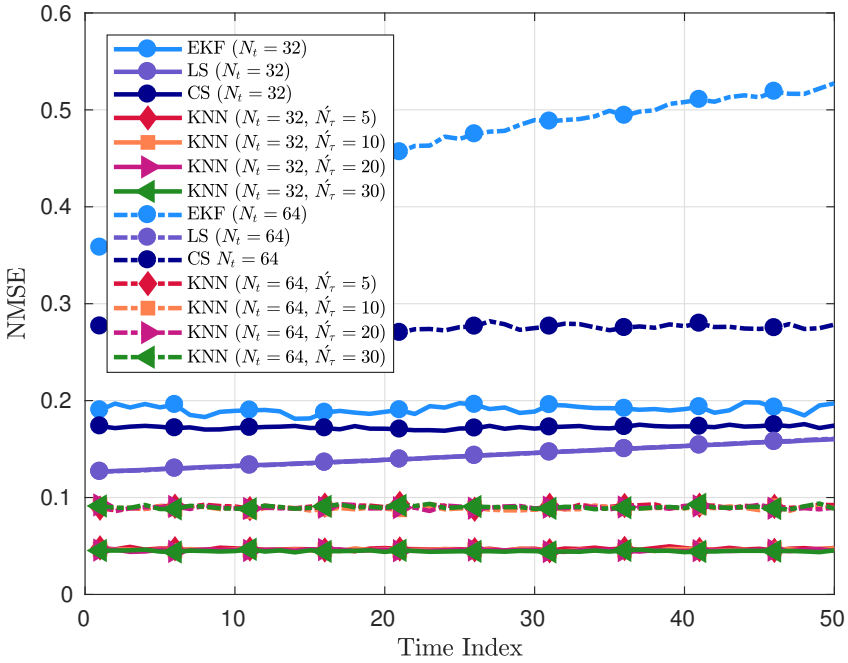
at the BS. The K-NN algorithm is just slightly better than the proposed K-NN-R framework with $R_s = 50\%$. This shows that measurements with only half of the beam pairs are enough to reliably estimate the channel state at the BS. It advantageously uses less the uplink feedback channel, then decreasing the signaling load. Moreover, even for $R_s = 25\%$, the performance loss of K-NN-R framework is still smaller than the baseline solutions along the entire temporal window. Interestingly, it allows a certain level of flexibility when choosing the percentage of beams to be randomly selected.

The accuracy of hitting the best transmit beam is also evaluated. That is, the best beam obtained at time instant τ based on the estimated matrix $\mathbf{H}(\hat{\mathcal{X}}_\tau)$ is compared to the actual best one, where a success occurs if they are the same. The cumulative distribution function (CDF) curves of the successful hit rate for the different tracking algorithms, for a ULA with either 32 or 64 elements at BS, and 8 elements at UE, are shown in Figure 3.15. As expected, regardless of the number of antennas at the ULA, the proposed K-NN hits more often the best beam than the baseline approaches for any value of R_s. The proposed K-NN-R with $R_s = 50\%$ exhibits a very close performance than K-NN, and a slight performance loss with $R_s = 25\%$. If the number of antenna elements increases to 64 at the BS, the performance loss of all tracking algorithms decreases, except for the LS algorithm, which exhibited a slight performance gain. Note that, as in Figure 3.14, the EKF algorithm exhibits a significant performance loss and a poor accuracy of hitting the best beam.

3.5.7 Channel prediction evaluation

In order to evaluate performance of the proposed K-NN and K-NN-R tracking algorithms as a channel prediction tool, it is assumed a fixed reporting temporal window $N_\tau = 50$ within which the channel matrix $\mathbf{H}(\mathcal{X}_\tau)$ is predicted from measurements collected by the UE only throughout a previous observed temporal window $\acute{N_\tau} < N_\tau$. An $\acute{N_\tau}$ to 5, 10, 20 or 30 consecutive time instants τ that precede the fixed N_τ is set. In all the simulations of this section, a ULA array with 32 or 64 elements at the BS and 8 antenna elements at the UE are assumed, and the values of K and β are set based on the cross-validation performed above, i.e., $\beta = 10\%$ and K the value that obtained the lowest NMSE for each $\acute{N_\tau}$ value. The three baseline approaches are also evaluated at each time instant τ of $N_\tau = 50$. Thus, the curves will exhibits a very close behavior to that observed in Section 3.5.6.

Figure 3.16 – Channel prediction evaluation with K-NN for $\beta = 10\%$ and different values of $\acute{N_\tau}$. There are either 32 or 64 elements at BS, and 8 elements at UE.

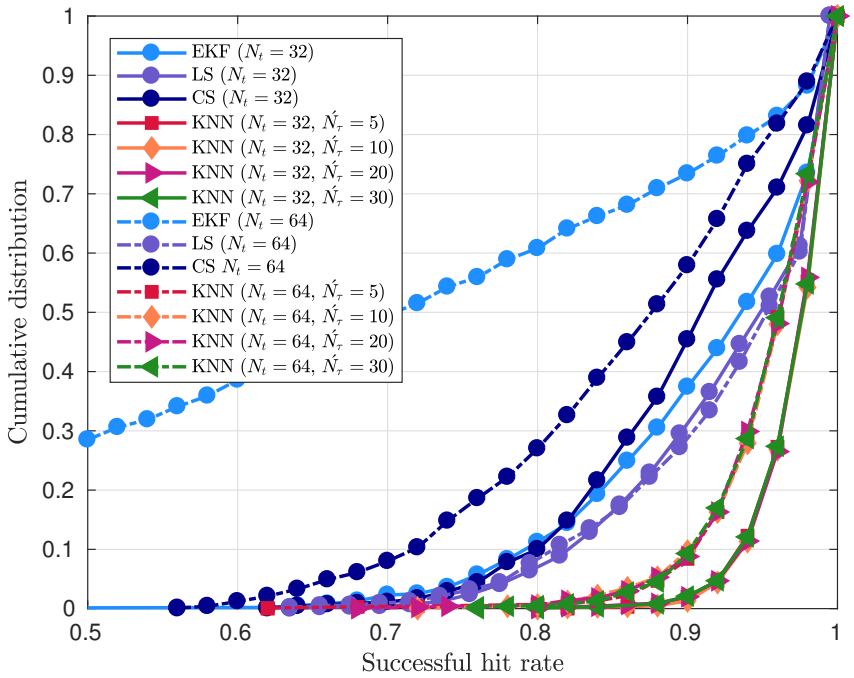


Source: Created by the author.

Figure 3.16 illustrates the NMSE per time instant of the K-NN tracking algorithm along the reporting temporal window N_τ for different values of observed temporal window $\acute{N_\tau}$. As can be seen, K-NN provides a lower NMSE for all $\acute{N_\tau}$ values, regardless of the number of antennas at the BS. Further, the $\acute{N_\tau}$ value did not impact significantly the performance of K-NN. It seems that the richness of the HUD is good enough, along with the cross-validation, thus making K-NN robust to different choices of the flexible parameter $\acute{N_\tau}$.

The accuracy of hitting the best beam of the K-NN algorithm is also evaluated. The CDF of the successful hit rate is shown in Figure 3.17. As expected due to its lower prediction

Figure 3.17 – Accuracy of finding the best beam with K-NN method for $\beta = 10\%$ and different values of $\acute{N}_\tau$. There are either 32 or 64 elements at BS, and 8 elements at UE.



Source: Created by the author.

error, the K-NN algorithm hits more often the best beam than the baseline approaches for any value of $\acute{N}_\tau$ and antenna configuration.

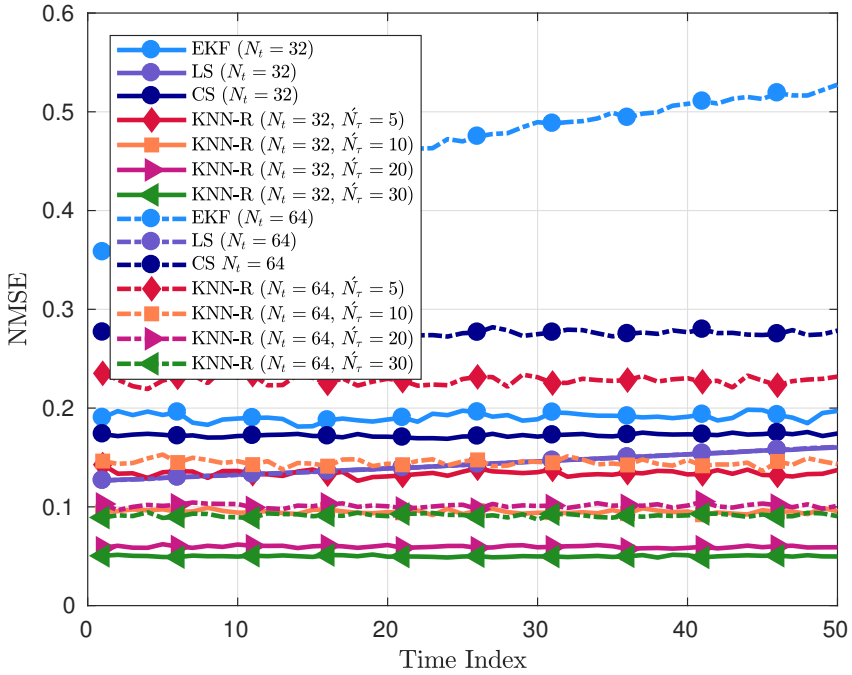
The same sort of analysis is also carried out for the proposed K-NN-R framework with R_s equal to 50% and 25%. Figure 3.18 illustrates the NMSE per time instant of the K-NN-R framework with $R_s = 50\%$ and for different values of observed temporal window $\acute{N}_\tau$.

As can be seen, the larger $\acute{N}_\tau$ the lower NMSE for the K-NN-R. Moreover, the BS with 32 antenna elements provides a lower NMSE than with 64 elements for all curves with similar $\acute{N}_\tau$ value. Additionally, the proposed K-NN-R framework outperforms the baseline algorithms for an $\acute{N}_\tau$ greater than 10 when 32 antenna elements at the BS are considered, and for an $\acute{N}_\tau$ greater than 20 for the case with 64 antenna elements. This result show that measurements with only 50% of the beam pairs and throughout an significantly lower observed temporal window, e.g., $\acute{N}_\tau = 10$, are enough to reliably predict the channel state on the BS side, thus yielding bandwidth savings related to the use of the uplink feedback channel.

The accuracy of hitting the best beam of the K-NN-R algorithm with $R_s = 50\%$ is also evaluated. The CDF of the successful hit rate is shown in Figure 3.19. As expected, for the ULA with 32 elements, the proposed K-NN-R hits more often the best beam than the baseline approaches for any observed temporal window length. Moreover, a general slight performance loss is observed at the K-NN-R framework when a ULA with 64 elements is assumed, and the $\acute{N}_\tau = 10$ value is required to outperform the LS method.

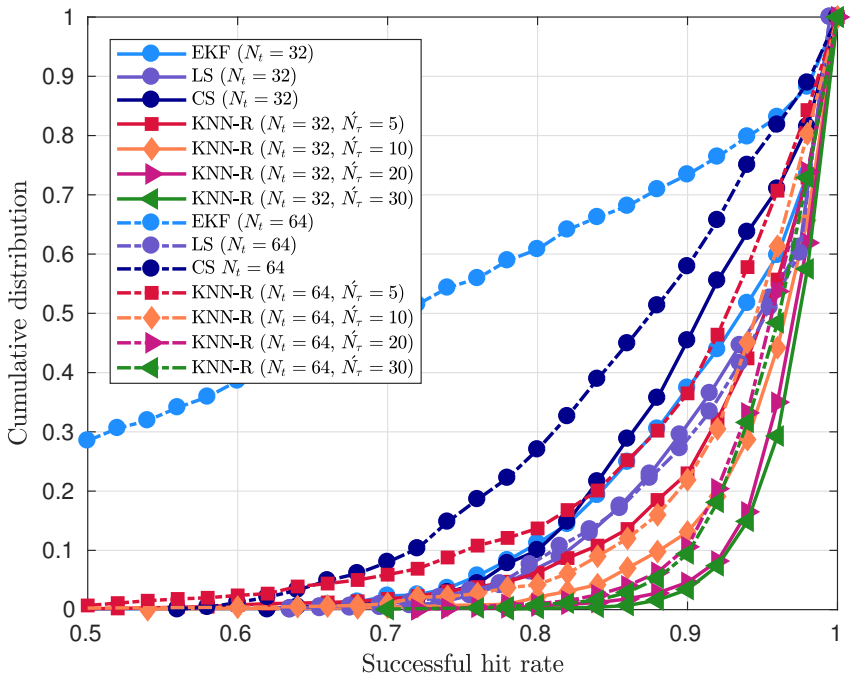
Finally, the NMSE per time instant and the accuracy of hitting the best beam of the

Figure 3.18 – Channel prediction evaluation with K-NN-R for $\beta = 10\%$, $R_s = 50\%$ and different values of $\acute{N}_\tau$. There are either 32 or 64 elements at BS, and 8 elements at UE.



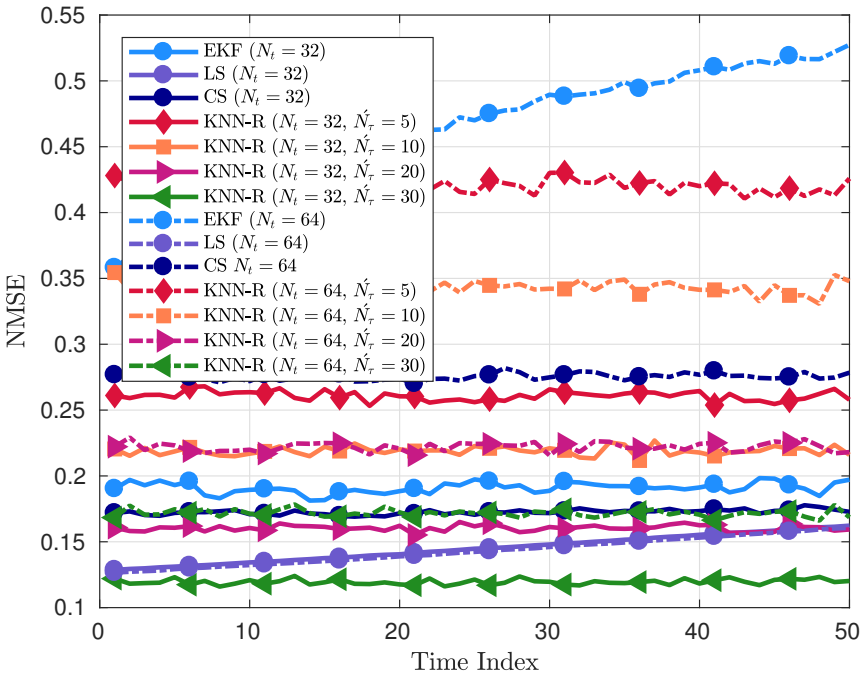
Source: Created by the author.

Figure 3.19 – Accuracy of finding the best beam with K-NN-R for $\beta = 10\%$, $R_s = 50\%$ and different values of $\acute{N}_\tau$. There are either 32 or 64 elements at BS, and 8 elements at UE.



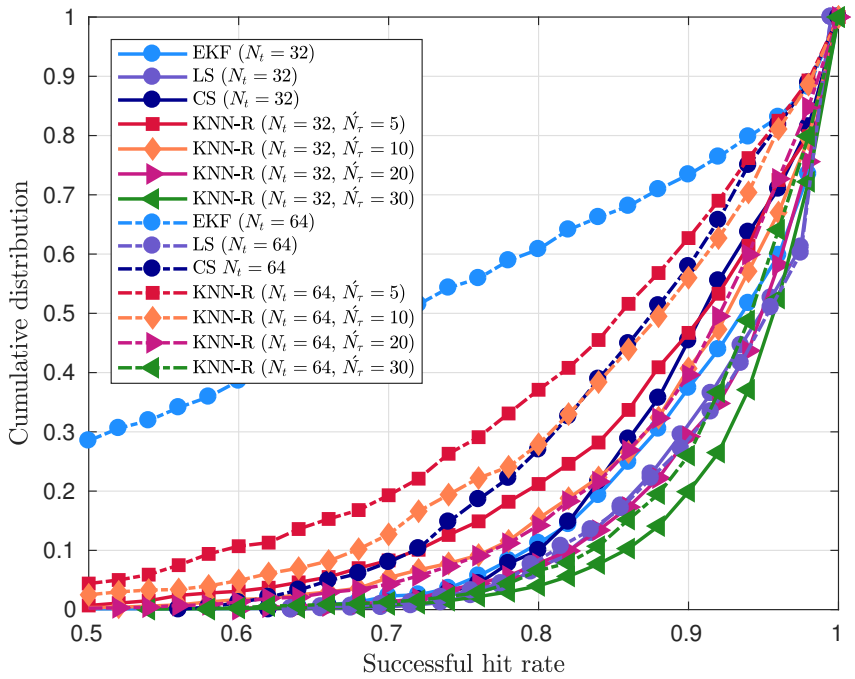
Source: Created by the author.

Figure 3.20 – Channel prediction evaluation with K-NN-R for $\beta = 10\%$, $R_s = 25\%$ and different values of $\acute{N}_\tau$. There are either 32 or 64 elements at BS, and 8 elements at UE.



Source: Created by the author.

Figure 3.21 – Accuracy of finding the best beam with K-NN-R for $\beta = 10\%$, $R_s = 25\%$ and different values of $\acute{N}_\tau$. There are either 32 or 64 elements at BS, and 8 elements at UE.



Source: Created by the author.

proposed K-NN-R framework with $R_s = 25\%$ are illustrated in Figures 3.20 and 3.21, respectively. As expected, a longer observed temporal window $\acute{N_\tau}$ is necessary to keep a low prediction error when a lower sampling rate R_s value is assumed. As can be seen in Figure 3.20, for $R_s = 25\%$ and a BS with 32 elements, only the curve with $\acute{N_\tau} = 30$ reached lower NMSE than all the baseline algorithms. Meanwhile, the curves with $\acute{N_\tau} = 20$ for a BS with 32 elements and $\acute{N_\tau} = 30$ for a BS with 64 elements exhibit a very close performance compared to the best baseline method, i.e., LS. Moreover, the configuration with $\acute{N_\tau} = 30$ and 32 antenna elements at the BS is the only one that hits more often the best beam than the LS algorithm, as shown in Figure 3.21.

3.6 Chapter summary

In this chapter it was proposed and evaluated a tracking strategy based on supervised learning and GSP. Specifically, the use of the so-called K-NN algorithm for channel tracking and prediction was proposed. As a supervised learning technique, K-NN relies on the availability of a HUD set for the purpose of offline training. In addition, as preliminary stages of the K-NN algorithm, sampling and reconstruction strategies based on GSP were introduced, which was referred to it as K-NN-R, in order to efficiently use the uplink feedback channel. As baseline solutions, three classical tracking/prediction algorithms were adopted, namely LS, CS and EKF.

Simulation results showed that the use of K-NN provided better estimation and prediction performance in terms of NMSE and successful hit rate. Moreover, while the baseline approaches and K-NN with no sampling demand full beam sweep and reporting, the proposed K-NN-R framework selects only a few beam pairs at random to be monitored and sampled over time. According to simulation results, the performance of K-NN-R is not affect much in terms of estimation/prediction error. Additionally, only half of the beam pairs ($R_s = 50\%$) throughout a significantly lower observed temporal window (i.e., $\acute{N_\tau} = 10$ for 32 antenna elements and $\acute{N_\tau} = 20$ for 64 antenna elements) were shown to be enough to reliably predict the channel state on the BS side, which consequently uses more efficiently the uplink feedback channel. Furthermore, our proposed K-NN-R exhibits the lower computational complexity when compared to the three baseline approaches, specially in mmWave scenarios that usually lead to the use of a large antenna arrays.

4 GRAPH SIGNAL PROCESSING FOR QOS PREDICTION IN CELLULAR V2X COMMUNICATIONS

In this chapter, the concept of QoS predictability in a C-V2X scenario is addressed. Latency prediction is modeled as a binary classification problem, in which measurements taken on both BS and UE sides are considered as features. The uplink feedback channel for UE measurement reporting is assumed to be limited. In this context, it is important to reduce the feedback overhead for the operation of the ML-based latency predictor. For this goal, a graph-based sampling strategy, namely SLS, is proposed and evaluated so that i) the graph structure/connectivity is preserved, and ii) the amount of sampled UE measurements is reduced. An online reconstruction strategy is adopted, namely O-SGTD, which shows a lower computational complexity compared to a batch reconstruction strategies. In turn, reconstructed data are used as input of the ML-based predictor for the underlying latency prediction problem. Simulation results show a better performance of the proposed approach in terms of number of unsampled UE measurements and reconstruction error, when compared to three baseline sampling algorithms. Moreover, it exhibits prediction performance close to the case when full data are available on the network side.

4.1 Introduction

In the context of 5G wireless communications and beyond, rapidly-growing data volume in C-V2X communications poses a significant challenge in data analysis and processing tasks for the purpose of network optimization and decision-making processes [95]. For instance, connected vehicles are constantly sharing data with either the network or neighboring vehicles, or both, so that they can be aware of pedestrians, other vehicles — and any other object that may potentially cause an accident — in their vicinity.

Most use cases in vehicular communications demand the delivery of critical data within tight latency constraints [96]. It is anticipated that C-V2X will enable real-time control applications such as high-level autonomous driving, dynamic route, accident assistance system and collision warning system, traffic efficiency control, among others [97]. The use of these real-time vehicular applications poses a meaningful challenge because its end-to-end latency has to be very low and its data delivery ratio must be very reliable. However, poor channel conditions and/or data traffic congestion can be an obstacle to a reliable delivery of such critical data within the latency requirements accepted by the applications. In this context, the management of radio resources somehow needs to adaptively change network parameters to overcome such impairments.

The QoS prediction of radio links among vehicles and the network is a means of proactively reacting to the aforementioned impairments. Vehicular applications may require from the network accurate QoS indicators predictions, in order to adapt themselves to the current channel

conditions and then react appropriately. Specifically, the main idea behind QoS prediction is to inform to the vehicular application ahead of time about, for instance, the connectivity quality (i.e., that the network can support the QoS required by that application), or availability (e.g., coverage hole), or capability (e.g., the network can or can not support this service). Such procedure may increase the reliability and comply the latency requirements in real-time applications.

Regardless of the methodology devised for QoS prediction, the network must acquire a potentially large amount of network/device/context data related to the connected vehicles. In practice, vehicles' devices, or simply UEs, carry out measurements of signals at different layers and report them to the network. UE measurements can in general act as input to train ML models [98]. In principle, every measurement should be reported to the network for model training purposes. Once the model is trained, UEs keep reporting measurements for the network to predict future QoS levels. These predictions are used by the network to make its own decisions or notify to a vehicle in advance, so that the vehicle is able to act on this information. However, the uplink channel for UE measurement reporting has limited feedback bandwidth. Thus, the network should make an efficient use of it.

The challenge here is to determine which signals UEs should measure and how often they should report them. Implicitly, some spatio-temporal interdependency can be observed among data acquired by the network. Besides, such data can inherently have irregular domains, e.g., signal values that depend on cell identity, device identity, among others, upon which some basic operations on signals [59] are not directly well defined. These two aspects lead to the interest in graphically modeling signals to capture the structure that underlies signals that convey network data.

In this context, it is envisaged the use of GSP to incorporate additional structures and further improve the ML-based prediction tasks reducing the search space and thus computation efforts.

4.1.1 Contributions

The works mentioned in Section 1.2.2 model the data as graph signals in a connected graph by assuming the existence of a dependency between each pair of vertexes in the graph. This leads to significant challenges in data processing and understanding in scenarios in which the amount of the generated and collected data increases quickly, as in C-V2X communications. Moreover, many of these works use RS without any sampling constraint on the graph structure and connectivity, hindering the reconstruction process of graph signals.

In this chapter, the graph structure that represents C-V2X measurement dataset for QoS prediction is exploited. Two different groups of data are defined regarding their availability, namely BS data and UE data. The former denotes C-V2X measurements already available at the BS, while the latter stands for those available at UEs and must eventually be reported. The main contribution then emerges by analyzing the graph Laplacian learned from C-V2X dataset. First, under certain conditions, it can present a block-diagonal structure indicating that GSP tools can

be applied into each sub-graph independently. If that is the case, sampling of UE data is avoided in sub-graphs whose vertex set comprises vertexes representing both data types. In this case, the recovery of missing UE data is based only on sampled BS ones, which drastically reduces the need of UE measurement reporting.

By using the C-V2X measurement dataset described in [99], the proposed sampling method is assessed in terms of reconstruction error, while the overall reconstructed dataset is evaluated as input of an ML-based QoS prediction solution.

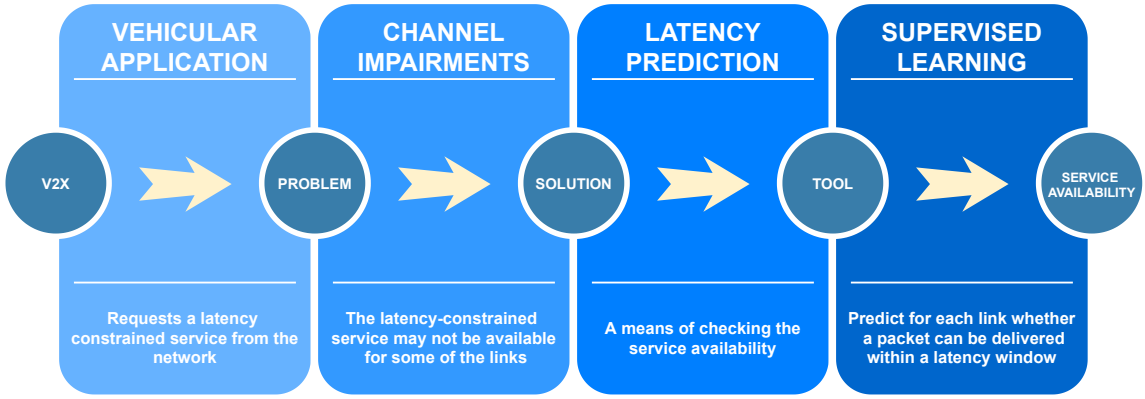
4.2 System model

Let V be the number of vehicle devices, or simply UEs, moving in a C-V2X scenario. Each UE is served by one BS and a BS can serve multiple UEs. Also, such BSs form a radio access network (RAN).

Now consider that a vehicular application requests a latency-constrained service from the RAN. It demands a periodic data traffic for each BS-UE link. However, due to channel impairments and network overload, the latency-constrained service may not be available for some of the V links.

Latency prediction then emerges as a means of checking the service availability. To this end, any supervised learning technique can be adopted to predict for each link whether a packet can be delivered within a latency window L in the downlink, as illustrated in Figure 4.1.

Figure 4.1 – General considerations of the problem.



Source: Created by the author.

The problem is modeled as a binary classification for each link with $y_{\tau+\acute{\tau}} \in \{0, 1\}$ denoting binary labels. Let $\acute{\tau}$ be a future time gap. Then, $y_{\tau+\acute{\tau}}$ indicates if at instant τ a packet transmitted at $\tau+\acute{\tau}$ can be delivered *in time* or will be *delayed* given the latency L. The prediction is based on a feature vector $\mathbf{x}_\tau \in \mathbb{R}^{N_f}$ that comprises N_f measurements taken on both BS and UE sides at instant τ. For simplicity, link-specific indexing is omitted in labels and feature vectors.

For a given link, the influence of the other $V-1$ links on packet delays is captured

by its feature vector. Moreover, its system model at time instant τ is as follows:

$$y_{\tau+\acute{\tau}} = h(\mathbf{x}_\tau), \tag{4.1}$$

where $h(\cdot)$ is a function that maps measurements taken at instant τ into the binary label at $\tau+\acute{\tau}$. In practice, $h(\cdot)$ is unknown. Thus, the goal of the supervised learning algorithm is to learn $h(\cdot)$ in order to predict $y_{\tau+\acute{\tau}}$ for every link. Besides, training and prediction are assumed to be carried out at each BS for each of its associated links.

As mentioned before, $\mathbf{x}_\tau$ is composed of two parts, i.e., some of its entries represent BS measurements, while others refer to measurements available only at the UE. Particularly, UE measurements must be fed back to its serving BS via the uplink feedback channel as the prediction functionality is assumed to be carried out at the BS.

The uplink channel use is assumed to be limited for each link, which makes UE measurement reporting costly. To overcome this issue, feature vectors are modeled as graph signals. Then, sampling and reconstruction of graph signals are exploited to decrease the need for UE measurement reporting.

4.3 Graph representation of features

In this section, the feature vector $\mathbf{x}_\tau$ is represented as a time-varying graph signal, i.e., a signal defined on a graph $\mathcal{G}$. However, the graph topology that represents the feature vectors is not readily available. In this chapter, such a topology, or more specifically the graph Laplacian of $\mathcal{G}$, is learned from multiple observations of $\mathbf{x}_\tau$ by means of a statistical approach.

Let $\mathbf{X}_\tau \in \mathbb{R}^{N_f \times N_\tau}$ be the observation matrix comprising N_τ observations of feature vectors and whose τ-th column denotes the observation $\mathbf{x}_\tau$. That is, for a given link, N_f measurements are taken along the first N_τ consecutive time instants.

Then, the matrix $\mathbf{L}$ in Equation (2.6) is learned during the training phase from matrix $\mathbf{X}_\tau$, which spans the first N_τ time instants of the measurements, and following the GGL graph learning technique described in Section 2.4.1. To this end, the data statistics matrix $\mathbf{S}$ is obtained by using the unit-variance Gaussian kernel function defined in Equation (2.15), with $i, j = 1, \ldots, N_f$. Moreover, each input vector $\mathbf{s}_i$ of the kernel function is defined as in Equation (2.14) with $i = 1, \ldots, N_f$ as

$$\mathbf{s}_i = \begin{bmatrix} [\mathbf{X}_\tau]_{i,1} & [\mathbf{X}_\tau]_{i,2} & \cdots & [\mathbf{X}_\tau]_{i,N_\tau} \end{bmatrix}^T, \quad i = 1, \ldots, N_f,$$

Specifically, the target matrix $\mathbf{L}$ is obtained from the minimization optimization problem defined in Equation (2.16). The problem in (2.16) is solved by using the iterative block-coordinate descent algorithm described in [91], in which each block-coordinate iteration has $O(\varsigma(s) + (N_f)^2)$ complexity, with $\varsigma(s)$ be the complexity of solving the subproblem with dimension s.

Once the Laplacian matrix is learned, the graph $\mathcal{G}$ may have K connected components, such that the number of connected components is related to the spectral property of the Laplacian

matrix as described in Section 2.3.2. That is, it can be a composition of $\mathcal{G}_k$ for $k = 1, \ldots, K$ disjoint connected subgraphs–including *trivial* subgraphs. This k-block structure leads to a block-diagonal graph Laplacian as formulated in Equation (2.12). In this chapter, it is paid special attention to the use of a dataset comprising typical C-V2X measurements whose underlying structure presents this block-diagonal graph Laplacian structure.

Thus, the feature vector $\mathbf{x}_\tau$ that represents the time-varying graph signal can also be split into the K parts or subgraphs as formulated in Equation (2.13).

4.3.1 Subgraph types

As mentioned above, each connected component K in $\mathcal{G}$ denotes a subgraph that comprises one isolated vertex (IV) or multiple vertexes (MV). Besides, it can represent measurements available at the BS or at the UE, or a combination of both. Thus, five types of subgraphs are defined:

1. **MV-UE**: Vertexes of the subgraph are only related to UE measurements.
2. **IV-UE**: An isolated vertex of a trivial subgraph represents a particular UE measurement.
3. **MV-BS**: Vertexes of the subgraph are only related to BS measurements.
4. **IV-BS**: An isolated vertex of a trivial subgraph represents a particular BS measurement.
5. **MV-BU**: Some vertexes are related to BS measurements, while others are related to UE measurements.

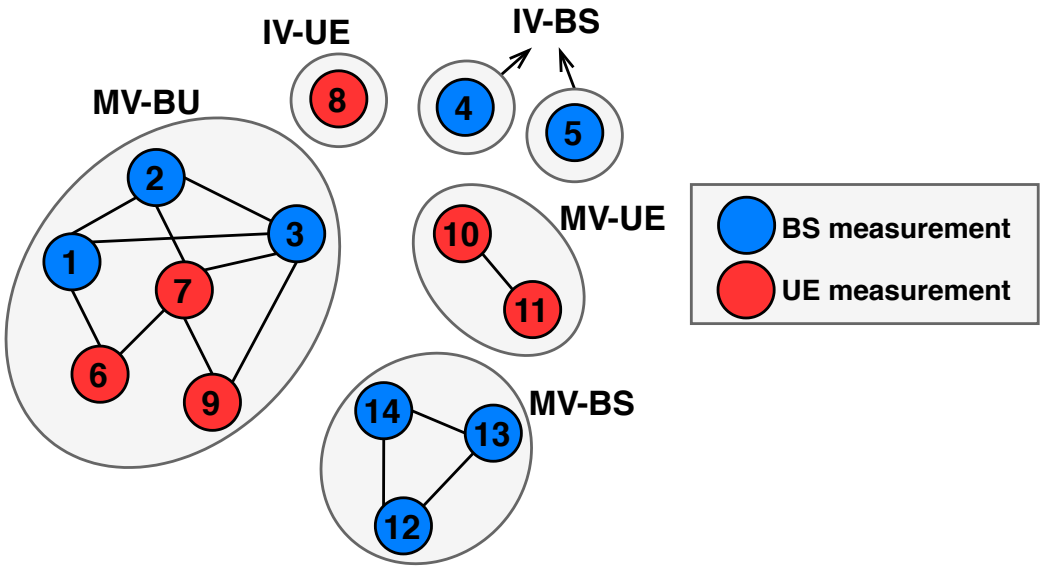
In practice, a graph with multiple connected components does not necessarily have all subgraph types.

Figure 4.2 illustrates an example of a graph with multiple connected components composed of fourteen vertexes in total. In this example, all subgraph types mentioned above are present. Regarding MV subgraphs, there is one MV-BU subgraph comprising six vertexes, one MV-BS subgraph having three vertexes, one MV-UE subgraph composed of two vertexes. Furthermore, there are two IV-BS trivial subgraphs and one of the IV-UE type. This means that the graph Laplacian of this illustrative graph can be split into $K = 6$ blocks.

4.4 Graph signal reconstruction

Let us suppose that at each time instant a different subset of the entries of $\mathbf{x}_t$ is missing at the BS. This can be represented at a time instant τ by the linear model defined in Equation (2.24). Note that in principle the missing entries may be from the set of BS and/or UE

Figure 4.2 – Example of graph with multiple connected components showing the different subgraph types.



Source: Created by the author.

measurements. Besides, this sampling mechanism is carried out only during the prediction phase, which begins at time instant $N_\tau + 1$.

Since the time-varying graph signal $\mathbf{x}_\tau$ is divided into K parts according to Equation (2.13), as a consequence $\bar{\mathbf{M}}_\tau$ has also a block-diagonal structure, which gives

$$\bar{\mathbf{M}}_\tau = \begin{bmatrix} \bar{\mathbf{M}}_{\tau,1} & & \\ & \ddots & \\ & & \bar{\mathbf{M}}_{\tau,K} \end{bmatrix}, \tag{4.2}$$

and therefore,

$$\mathbf{y}_\tau = \begin{bmatrix} \mathbf{y}_{\tau,1} & \cdots & \mathbf{y}_{\tau,K} \end{bmatrix}, \tag{4.3}$$

where the k-th independent term of $\mathbf{y}_\tau$ in (2.24) is given by

$$\mathbf{y}_{\tau,k} = \bar{\mathbf{M}}_{\tau,k}\mathbf{x}_{\tau,k}. \tag{4.4}$$

Given a sampling strategy and the estimate of the Laplacian matrix, the signal reconstruction of $\mathbf{x}_\tau$ is carried out at the BS. In this context, it is assumed the use of the online smooth graph-time difference (O-SGTD) strategy formulated in Equation (2.32), for full-band graph signals (not bandlimited). Recalling Section 2.7.2, the goal of O-SGTD strategy is then to recover the vectors $\mathbf{x}_\tau$ from $\mathbf{y}_\tau$ given matrices $\mathbf{L}$ and $\bar{\mathbf{M}}_\tau$.

For a better understanding of the reconstruction procedure for the k-th subgraph signal, the objective function in Equation (2.32) can be rewritten as

$$\hat{\mathbf{x}}_\tau = \arg\min_{\mathbf{x}} f(\mathbf{x}), \tag{4.5}$$

with

$$f(\mathbf{x}) = ||\bar{\mathbf{M}}_\tau \mathbf{x} - \mathbf{y}_\tau||_2^2 + \gamma(\mathbf{x} - \hat{\mathbf{x}}_{\tau-1})^T \mathbf{L}(\mathbf{x} - \hat{\mathbf{x}}_{\tau-1}), \tag{4.6}$$

where $\hat{\mathbf{x}}_{\tau-1}$ is the reconstructed signal at the previous $\tau - 1$ time instant and γ is the regularization parameter. The second term in the objective function acts as a smoothness constraint in the time domain of the graph signal $\mathbf{x}$.

Again, given the block-diagonal structure of the graph Laplacian, the objective function in (4.5) can be factorized into K parts, each denoting an independent reconstruction problem for each subgraph signal, i.e.,

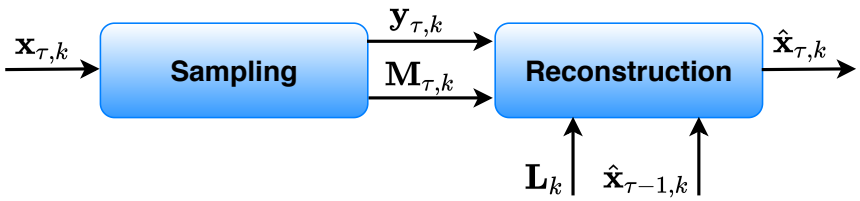
$$f(\mathbf{x}) = \sum_{k=1}^{K} f_i(\mathbf{x}_k)\,. \tag{4.7}$$

where the $\bar{\mathbf{M}}_\tau$, $\mathbf{L}$ and $\mathbf{y}_\tau$ terms in the objective function (4.6) can be replaced by its correspondent $\bar{\mathbf{M}}_{\tau,k}$, $\mathbf{L}_{\tau,k}$ and $\mathbf{y}_{\tau,k}$ terms given by Equations (4.2), (2.12) and (4.4), respectively. Thus, the l-2 norm in Equation (4.6) is factorized into K norms, as well as the regularization term. Thus, the k-th term of $\hat{\mathbf{x}}_\tau$ in (4.5) can be obtained as

$$\hat{\mathbf{x}}_{\tau,k} = \arg\min_{\mathbf{x}} f_k(\mathbf{x}), \tag{4.8}$$

given $\mathbf{L}_k$, $\bar{\mathbf{M}}_{\tau,k}$ and $\mathbf{y}_{\tau,k}$, as illustrated in Figure 4.3. As visualized, each reconstruction subproblem with lower dimensionality may be solved in parallel, which leads to a lower runtime than O-SGTD method [49]. Finally, each reconstructed signal $\hat{\mathbf{x}}_{\tau,k}$ is in turn concatenated to obtain the resulting reconstructed vector $\hat{\mathbf{x}}_\tau$.

Figure 4.3 – Sampling and reconstruction procedures for the k-th subgraph signal at time instant τ.



Source: Created by the author.

4.5 Proposed graph signal sampling

In this section, it is proposed a sampling strategy that takes advantage of the block-diagonal structure of the graph Laplacian. Sampling rules are defined in order to reduce the UE measurement feedback signaling in the uplink. Since K independent reconstruction problems are obtained according to (4.8), the same sampling rules can be applied to each one. The sampling strategy is called smart Laplacian sampling (SLS).

Similarly to (2.25), $\bar{\mathbf{M}}_{\tau,k}$ is defined as

$$[\bar{\mathbf{M}}_{\tau,k}]_{u,u} = \begin{cases} 1, & u \in P_{\tau,k}, \\ 0, & u \notin P_{\tau,k}, \end{cases} \tag{4.9}$$

where $P_{\tau,k} \subseteq \nu_k$ for $k = 1, \ldots, K$ is the set of sampled vertexes of subgraph $\mathcal{G}_k$. The definition of $P_{\tau,k}$ is guided by two sampling rules as follows:

- **Sampling rule 1**: *At least one vertex must be sampled in each subgraph independently of its type*. It implies that the graph signal of trivial subgraphs, i.e., IV-BS and IV-UE subgraphs, is not part of the reconstruction problem as no missing entry will exist in such cases.
- **Sampling rule 2**: *Every vertex related to BS measurements is sampled*. The consequence of this rule is threefold. First, vertexes of MV-BS subgraphs will always be sampled implying that signal reconstruction is not needed in such cases. Second, in MV-BU subgraphs all the vertexes related to BS measurements are sampled while the remaining vertexes–that are related to UE measurements–are reconstructed. At last, in MV-UE subgraphs a random sampling with sampling rate R_S is performed and the missing values are reconstructed.

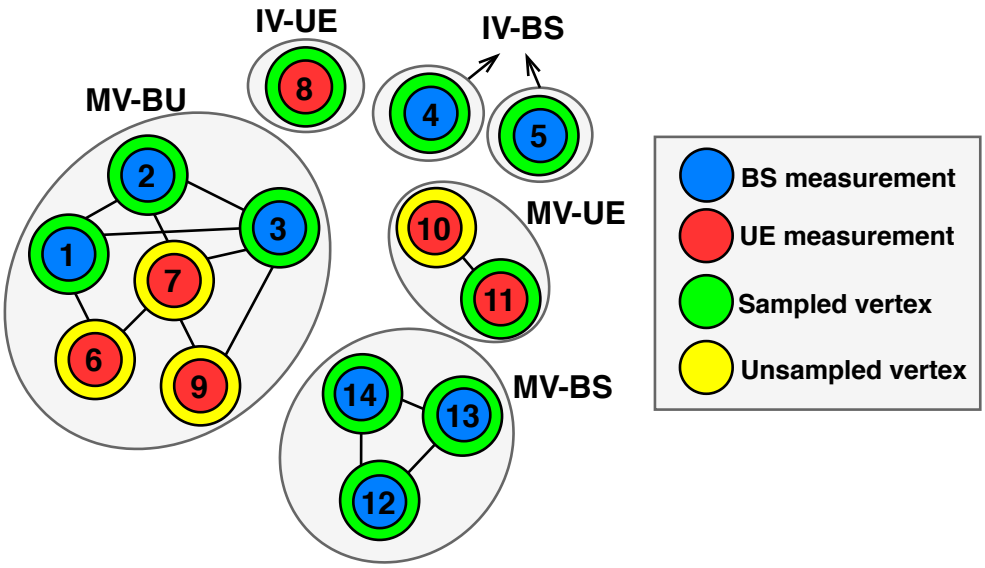
It is worth mentioning that these sampling rules imply that the feedback of some UE measurements is needed. For example, samples related to IV-UE subgraph vertexes need to be fed back. A fraction (ruled by R_S) of the graph signal related to MV-UE subgraphs also need to be fed back. On the other hand, a significant signaling saving is achieved thanks to sampling rule 2, specifically when the graph has the MV-BS and MV-BU subgraph types. Furthermore, the number of required reconstruction subproblems in Equation (4.8) is less than K when the graph comprises *trivial* subgraphs and/or MV-BS subgraphs.

Figure 4.4 illustrates an example of an SLS strategy over a graph with $K = 6$ connected components and 14 vertexes. As can be seen, at least one vertex is sampled for each subgraph, in accordance with the Sampling Rule 1. Moreover, both the IV-BS and IV-UE trivial subgraphs are forced to be sampled regardless of the measurements source that they represent. Further, the vertexes related to BS measurements in the MV-BS and MV-BU subgraph types are sampled in accordance with the Sampling Rule 2. At last, one vertex is sampled in the MV-UE subgraph, according to a random sampling with $R_S = 50\%$. Statistically, 33.3% (i.e., 2 of 6) vertexes related to UE measurements are sampled. This implies that SLS saves the feedback of 4 UE measurements.

4.6 C-V2X dataset experiment

In this section, it is described a dataset comprising typical C-V2X measurements whose underlying structure presents a block-diagonal graph Laplacian. Such a dataset will be

Figure 4.4 – Example of SLS sampling strategy.



Source: Created by the author.

used to assess the performance of the SLS. That is, it is intended to show that the performance of a selected supervised learning technique is not significantly impacted by the sampling and reconstruction procedures, while the uplink channel use is decreased due to the proposed sampling strategy.

The dataset is comprised of N_f typical BS and UE measurements as a collection of VT feature vectors $\mathbf{x}_\tau$, where $T > N_\tau$. In accordance with (4.1), each feature vector $\mathbf{x}_\tau$ is associated with a target label $y_{\tau+\acute{\tau}}$.

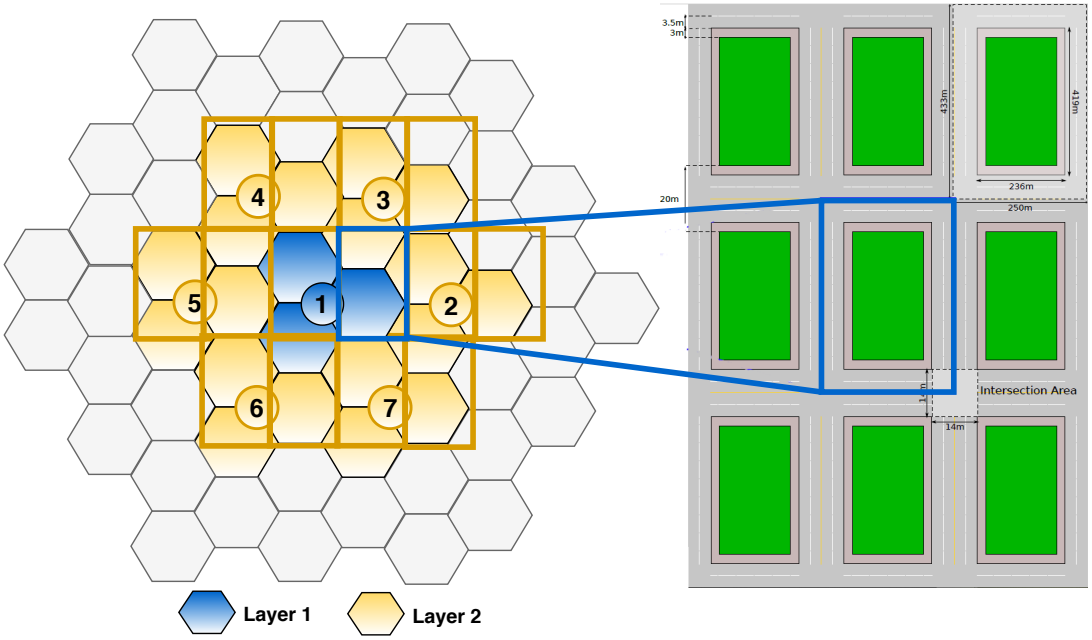
More specifically, data were generated in a Manhattan grid [100] scenario, where V vehicles moving at a constant speed are placed in a road grid topology with restricted moves into regular rectangular blocks along T time instants. Each road block has dimensions 250 m $\times$ 433 m. Both horizontal and vertical directions have four lanes, each with a width of 3.5 m. Vehicles, or simply UEs, have a mobility pattern so they can only move in the streets and change their directions at the intersection as follows:

- Go straight with probability 0.5,
- Turn left with probability 0.25,
- Turn right with probability 0.25.

The cellular network has a wrapped-around seven-site hexagon layout, each site comprising three sectors as in [101]. That is, a urban scenario with 7 sites and 21 cells/sectors. Specifically, each BS in a cell site is located in the center of the cell site, but with its antenna array normal pointing in the direction of its corresponding sector. BSs are equipped with a 2-element antenna with a single cell-specific reference signal (CRS) port, such that each BS serves several

vehicles. Each vehicle is equipped with two antenna elements using an interference rejection combination (IRC) receiver. Moreover, UEs are considered to be receiving data from the network in downlink unicast. Figure 4.5 illustrates the considered urban scenario and the layout of the Manhattan grid model.

Figure 4.5 – Urban scenario with a wrapped-around seven-site hexagon layout, each site comprising three cells, on a Manhattan grid.



Source: Created by the author.

A channel model that follows the 3GPP specification for low frequency ranges is assumed [102], i.e., a 2 GHz carrier with a 5 MHz bandwidth. Moreover, a traffic model in which fixed size packages are periodically sent with a fixed transmit time interval is adopted [101]. In this context, every sent packet is expected to be delivered with a lower delay than a predefined latency requirement. Herein, latency is defined from the point of view of the transport layer. More detailed information about the dataset are available in [99].

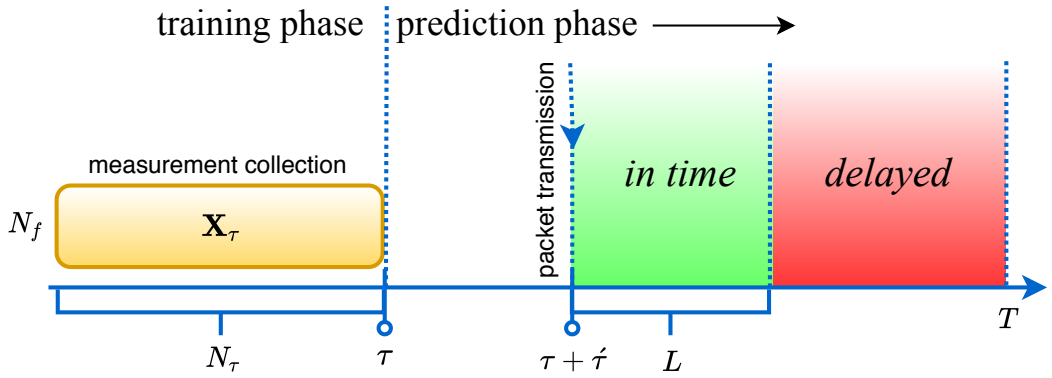
4.6.1 Machine learning tool

Two machine learning (ML) techniques as the supervised learning tool for the binary classification problem described in Section 4.2 are employed. Specifically, the well-known multilayer perceptron (MLP) and random forest (RF) techniques are employed as ML predictors. The ML techniques are trained to learn the function $h(\cdot)$ given the observation matrix $\mathbf{X}_\tau$ described in Section 4.3. That is, the training phase spans the first N_τ time instants of the measurements, while the remaining $T - N_\tau$ time instants are used in prediction mode using the trained ML predictors.

Therefore, for $N_\tau + 1 \leq \tau \leq T$ the trained predictors at time instant τ classifies whether a packet generated at $\tau + \acute{\tau}$ can be delivered or not within latency window L based on a feature

vector as input. Figure 4.6 illustrates the prediction problem.

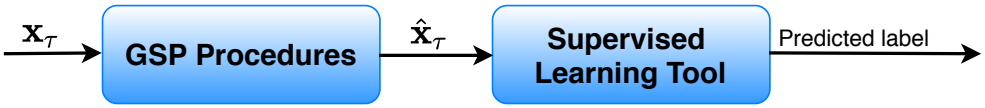
Figure 4.6 – Latency prediction problem.



Source: Created by the author.

Recall that the input of the ML can be a reconstructed feature vector $\hat{\mathbf{x}}_\tau$ because some UE measurements are not fed back, being reconstructed instead based on sampled measurements as described in Section 4.4. Figure 4.7 illustrates the block diagrams of GSP and the ML tools in the prediction problem.

Figure 4.7 – Reconstructed signals are used as the input of the machine learning experiment.



Source: Created by the author.

4.7 Simulation results

In this section, it is investigated the performance of the proposed sampling strategy, namely SLS, described in Section 4.5 and in the context of the data set described in Section 4.6. First, it is compared the chosen online reconstruction strategy with the three batch reconstruction strategies defined in Section 2.7.1 in terms of reconstruction error and runtime. Then, it is evaluated SLS in terms of feedback saving, as well as the resulting reconstruction error. Finally, the QoS predictability is evaluated assuming that the predictor accounts for reconstructed feature vectors as input.

The general simulation parameters are summarized in Table 4.1. The Manhattan grid scenario, as described in Sections 4.2 and 4.6, is considered in the simulation setup. Specifically, seven BSs, each with three sectors, serve $V = 700$ moving UEs at a speed of 60 km/h. Further, it is assumed a packet size S of 8,000 bits and a latency window L of 100 ms. An underlying orthogonal frequency-division multiplexing (OFDM)-based physical layer is assumed for data transmission.

Measurements are drawn from the dataset described in Section 4.6. The feature vector is comprised of 8 BS measurements and 6 UE measurements, i.e., $N_f = 14$, which are taken in a transmission time interval (TTI). Specifically:

- **BS measurements**: i) past 5 packet delays[1], ii) average hybrid automatic repeat request (HARQ) block error rate (BLER), iii) average throughput of the UE serving cell, and iv) load of the UE serving cell[2].
- **UE measurements**: i) UE x-coordinate[3], ii) UE y-coordinate, iii) average reference signal received power (RSRP), iv) average reference signal received quality (RSRQ), v) average downlink signal-to-interference-plus-noise ratio (SINR), and vi) average channel quality indicator (CQI).

Table 4.1 – Simulation parameters.

Number of sites	7
Number of sectors per site	3
Bandwidth [MHz]	5
Central frequency [GHz]	2
Subcarrier bandwidth [kHz]	15
Number of UEs V	700
UE speed [km/h]	60
TTI [ms]	1
Latency window L [ms]	100
Packet size S [bits]	8000
Number of features N_f	14
Number of BS measurements	8
Number of UE measurements	6

Source: Created by the author.

Table 4.2 – Learning, sampling and reconstruction parameters.

Training/learning phase duration N [ms]	180
Prediction phase duration [ms]	60
Sparsity level parameter ϱ GGL [35]	0.01
Regularization parameter γ [Eq. (4.5)]	near-optimal
Sampling rate R_S [%] SLS	50
Number of estimators for RF	100
Number of neurons in MLP hidden layer	100

Source: Created by the author.

[1] A delay is defined as the delivery time of a previous packet, e.g., analyzing the i-th packet, the first past delay means the delivery time of the $i-1$ packet.

[2] The load means the number of active UEs in the serving BS

[3] In practice, each vehicle has a connected GPS informing its coordinates

The learning, sampling and reconstruction parameters setup is shown in Table 4.2. A single realization of $T = 240$ ms is considered in the simulation setup, where both the ML training phase and the graph learning are done in the first $N_\tau = 180$ time instants, and the ML prediction phase with reconstructed feature vectors is carried out in the following 60 ms.

From the GSP perspective, the Laplacian matrix $\mathbf{L}$ is learned from the training set through the GGL optimization problem as discussed in Section 2.4.1. It is solved the GGL problem in Equation (2.16) with an iterative block-coordinate descent algorithm [103], in which $\varrho = 0.01$ and its stopping criteria are such that the maximum number of iterations is 10^2 and the error tolerance is 10^{-6} as in [35]. Further, a sampling rate of $R_S = 50\%$ is assumed in the SLS sampling rule over MV-UE subgraph type. Moreover, the online reconstruction problem, given by Equation (4.8), is solved through a distributed gradient descent algorithm [84], with a stopping criteria in which the maximum number of iterations is 10^4 and the error tolerance is 10^{-6} as in [49]. Further, it is found the near-optimal value for the γ regularization parameter in Equation (4.6) through a comprehensive grid search.

With regard to the employment ML algorithms, it is set 100 as the number of estimators in the RF algorithm and 100 neurons as size of the hidden layer of the MLP algorithm. Moreover, it is assumed the default values from the scikit-learn library [104] in Python for the other algorithm parameters. As a preprocessing step, missing value issues in the dataset are overcome by replacing any missing value of a feature with the mean of the available values of that feature. Then, for each feature, values are normalized to be in the interval $[-1, 1]$, following the "max abs scaler".

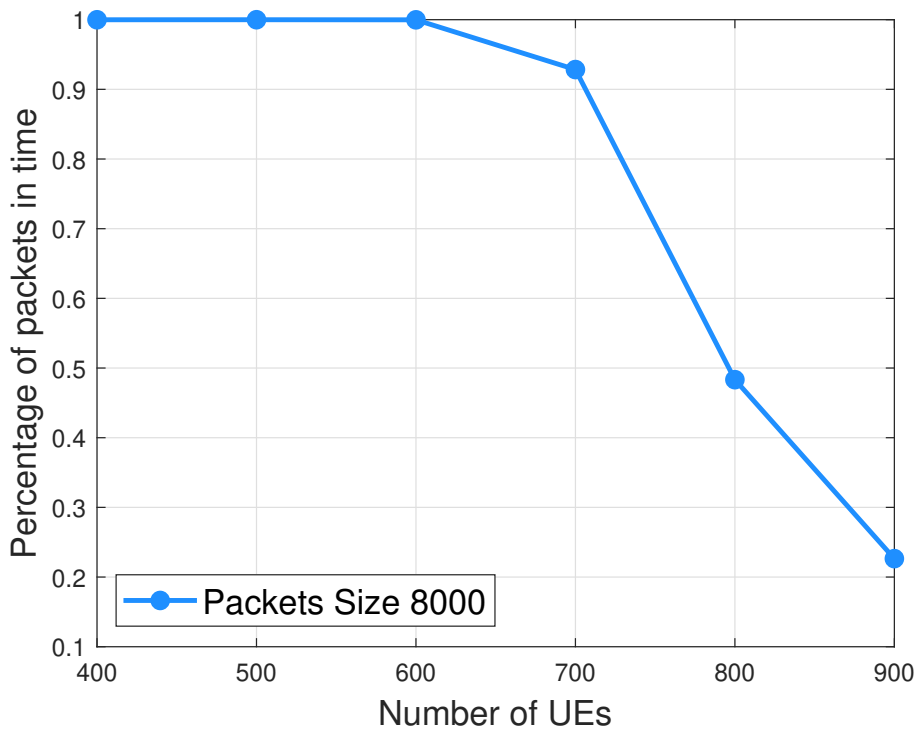
4.7.1 Class imbalance treatment

Usually, estimation of the accuracy metric is good enough to evaluate binary classification problems with ML strategies, in which the accuracy is defined as the rate of correct predictions of the algorithm. However, in binary classification models with class imbalance, the accuracy values can be misleading due to the imbalance of both labels.

In the context of the data set described in Section 4.6, previous assessments in [99] showed that the network performance is really imbalanced when a packet size of 8,000 bits and a 700 UE load are assumed as simulation setup. Figure 4.8 illustrates the network performance for the different UE loads and packet size of 8,000. Note that, the values in Figure 4.8 correspond to the proportion of each label in our binary classification problem. For lower loads, almost all packets arrive at the destination within the desired latency window of 100 ms. That is, most packets are delivered *in time* and therefore, almost all samples that will feed the ML model will be labeled as one. As the network load increases, the number of packets delivered outside the desired latency window increases and thus, more samples are labeled zero or *delayed*. Specifically, for a 700 UE load, more than 92% of the packets arrive are delivered *in time* and the model has class imbalance.

In this context, a more suitable metric is the f1-score, that are defined separately for

Figure 4.8 – Network performance.



Source: [99].

each label as

$$\text{f1-score} = 2 \times \frac{precision \times recall}{precision + recall}.$$

where *precision* is the metric that measures the ability of the classifier not to label as positive a sample that is actually negative. Moreover, *recall* metric measures the ability of the classifier to find all the positive samples. They are defined as

$$precision = \frac{\text{TP}}{\text{TP} + \text{FP}},$$

and

$$recall = \frac{\text{TP}}{\text{TP} + \text{FN}},$$

respectively, where TP is a "true positive" meaning an outcome where the algorithm correctly predicts the positive class, FP is a "false positive" outcome in which the algorithm incorrectly predicts the positive class and FN is a "false negative" outcome meaning where the algorithm incorrectly predicts the negative class. Thus, *precision* and *recall* are a good metrics to compare algorithms when the cost of a FP and a FN are high, respectively.

4.7.2 Reconstruction strategy evaluation

In this section, the performance of the chosen O-SGTD reconstruction strategy is evaluated in terms of NMSE and runtime. To this end, let ϵ be the normalized reconstruction

Table 4.3 – Runtime of the reconstruction algorithms.

Rec. Algorithm	**Runtime [s]**
O-SGTD	6.06×10^{-3}
GTTR	38.59×10^{-3}
SGTD	40.56×10^{-3}

Source: Created by the author.

error for a particular BS-UE link, herein defined as

$$\epsilon = \frac{1}{(T-N_\tau)K'}\sum_{k=1}^{K'}\sum_{\tau=N_\tau+1}^{T-N_\tau}\frac{\left|\mathbf{x}_{\tau,k}-\hat{\mathbf{x}}_{\tau,k}\right|^2}{\left|\mathbf{x}_{\tau,k}\right|^2}, \tag{4.10}$$

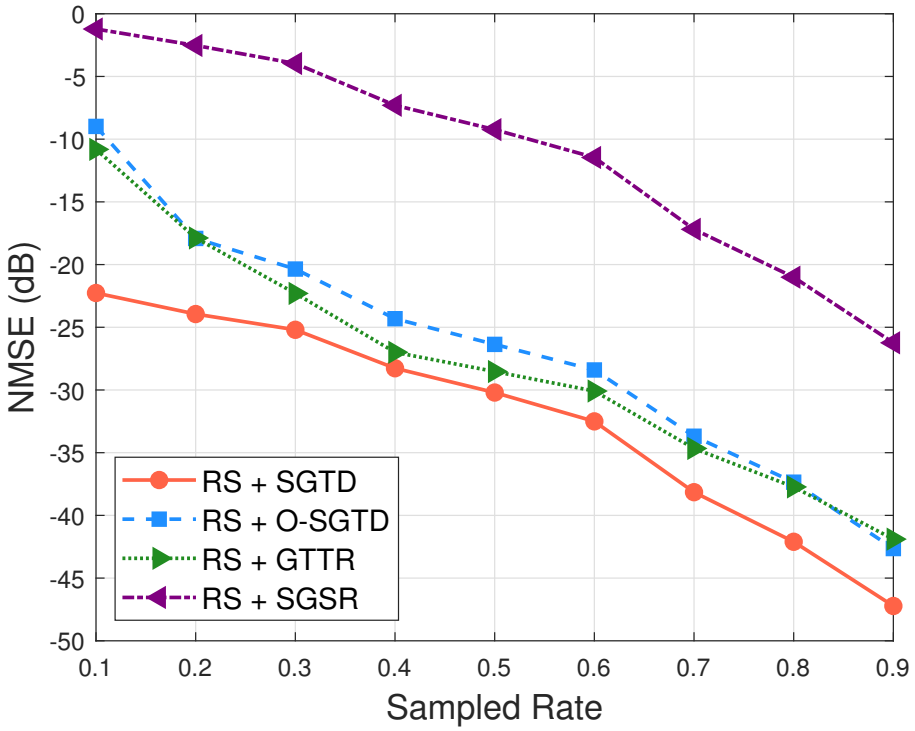
where $K' \leq K$ stands for the number of MV-BU and MV-UE subgraphs in the graph of that particular link. Then, the NMSE is obtained by averaging ϵ over the V links.

Specifically, O-SGTD is compared with the three batch reconstruction strategies described in Section 2.7.1. It is worth recalling that the batch reconstruction strategies needs to receive every sampled feature vector $\mathbf{x}_\tau$ within the prediction phase window $T-N_\tau$ before reconstructing them all together. To solve the optimization problems in Equation (2.28) it is used a conjugate gradient algorithm with a stopping criteria in which the maximum number of iterations is 10^4 and the error tolerance is 10^{-6}. Further, the near-optimal values for the β, γ and δ regularization parameters are found through a comprehensive grid search. Then, the performance of all the reconstruction methods is tested using random sampling (RS) under different sampling rate.

Figure 4.9 illustrates the NMSE per sampling rate (R_S) for the four reconstruction strategies. The NMSE is further averaged over all $V = 700$ reconstruction instances. As can be seen, the higher sampling rate, the lower NMSE for all the reconstruction algorithms. Moreover, the reconstruction of the SGTD strategy provides a lower NMSE for all R_S values. This result is expected because the temporal difference of signal usually exhibits better smoothness than the signal themselves.However, it is interesting to note that for values greater or equal than $R_S = 40\%$, the GTTR and the O-SGTD algorithms exhibits a very close NMSE than the SGTD strategy.

It is analyzed the runtime of the three reconstruction algorithms with very close NMSE performance and for $R_S = 50\%$. The hardware used to conduct analysis has the following specifications: Desktop PC with Ubuntu 18.04 system, Intel Core i5 Processor (4x 3.1 GHz) and 16 GB DDR4 RAM. Table 4.3 shows the average runtime for 700 vehicles for reconstructing the entire matrix $\hat{\mathbf{X}}_\tau$. Note that, the O-SGTD algorithm needs 5 times less runtime to reconstruct iteratively the matrix $\hat{\mathbf{X}}_\tau$ for one UE, than the batch reconstruction strategies. This result is expected as both the GTTR and the SGTD algorithms require obtaining the sampled values within the $T-N_\tau$ window and reconstruct them all together, which increases the computational burden.

Figure 4.9 – Reconstruction NMSE of feature vectors using different reconstruction strategies. O-SGTD is compared with three batch reconstruction strategies using RS with different sampling rate.



Source: Created by the author.

4.7.3 SLS evaluation

The performance of SLS is assessed in terms of feedback saving by counting the number of unsampled UE measurements during the ML testing phase considering all BS-UE links. Note that when SLS is the sampling strategy, only some UE measurements in MV-UE subgraphs are fed back, as well as all UE measurements in IV-UE subgraphs. Moreover, SLS with O-SGTD is evaluated in terms of NMSE as described in Equation (4.10).

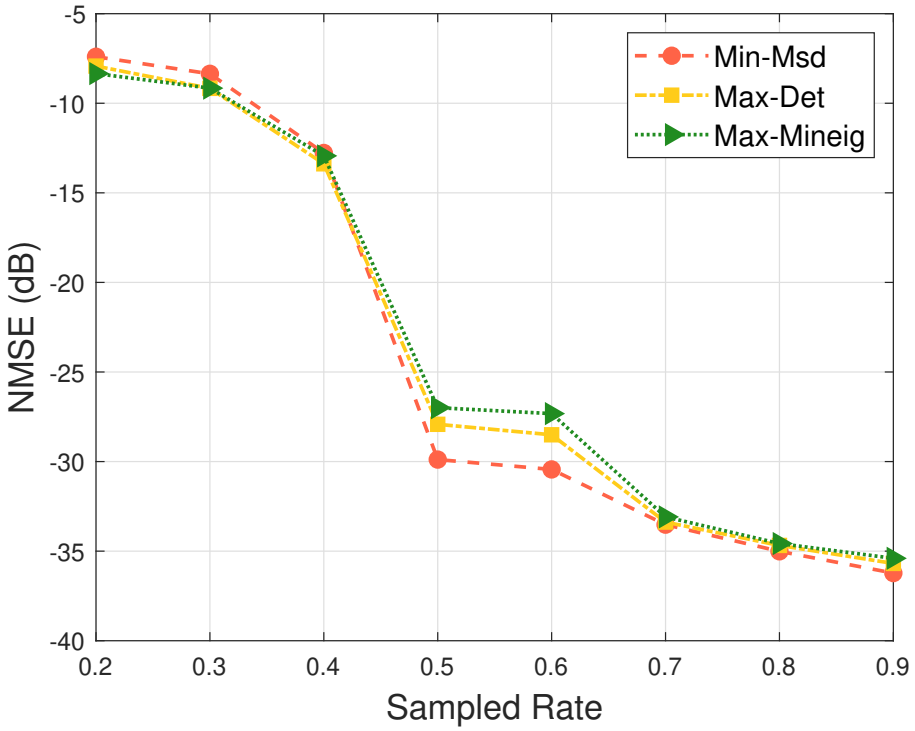
4.7.3.1 Baseline approaches

As baseline approaches, three graph-based greedy sampling algorithms found in the literature are adopted, i.e., minimization of mean-square desviation (Min-Msd) [105], maximization of determinant (Max-Det) [105] and maximization of the minimum nonzero eigenvalue (Max-Mineig) [106], which are compared with our proposed SLS strategy in terms of feedback saving and NMSE.

The baseline algorithms are adopted to find the set of sampled vertexes $P_{\tau,k}$ regarding every MV-BU and MV-UE subgraph types, without any restriction on the measurement source. For the others subgraph types, the algorithms follow a similar vertex sampling mechanism as SLS.

Specifically, the three algorithms employ at each iteration a greedy search to select

Figure 4.10 – NMSE of the three baseline algorithms with different sampling rate.



Source: Created by the author.

the vertex i to be added to the current $P_{\tau,k}$, based on a certain criterion and according to a pre-established sampling rate R_s.

Concerning the vertex selection criterion, the Min-Msd strategy selects the vertexes from the graph that lead to the largest reduction in terms of mean-square desviation (MSD). Moreover, the Max-Det algorithm aims at maximizing the pseudo-determinant of the matrix $\mathbf{U}^T\bar{\mathbf{M}}_\tau\mathbf{U}$ (i.e., the product of all nonzero eigenvalues). The idea behind that is to design a well suited basis for the graph signal that is to be reconstructed [105]. Finally, using similar arguments as the Max-Det strategy, the Max-Mineig algorithm selects the vertexes in order to maximize the minimum nonzero eigenvalue of the matrix $\mathbf{U}^T\bar{\mathbf{M}}_\tau\mathbf{U}$.

Figure 4.10 illustrates the NMSE per sampling rate (R_s) for the three baseline sampling algorithms. After applying the sampling, the NMSE is further obtained using the O-SGTD reconstruction strategy and by averaging ϵ over all the $V = 700$ links. As can be seen, the higher sampling rate, the lower NMSE for all the algorithms. Moreover, although it is highly costly to compute, the Min-Msd strategy outperforms all other methods when the number of sampled vertexes increases, i.e., for values greater or equal than $R_s = 50\%$. This is because Min-Msd takes into account information from both spatial distribution of the observation noise and graph topology [105]. Besides, the Max-Det algorithm outperforms the Max-Mineig strategy as it considers all the eigenvalues and not just one. However, it is interesting to note that the initial decrease in NMSE is greater until the value of $R_s = 50\%$ for the three baseline algorithms. Because of this, a sampling rate of $R_s = 50\%$ will be used for the three baseline sampling algorithms to be compared with our proposed SLS strategy in the next section.

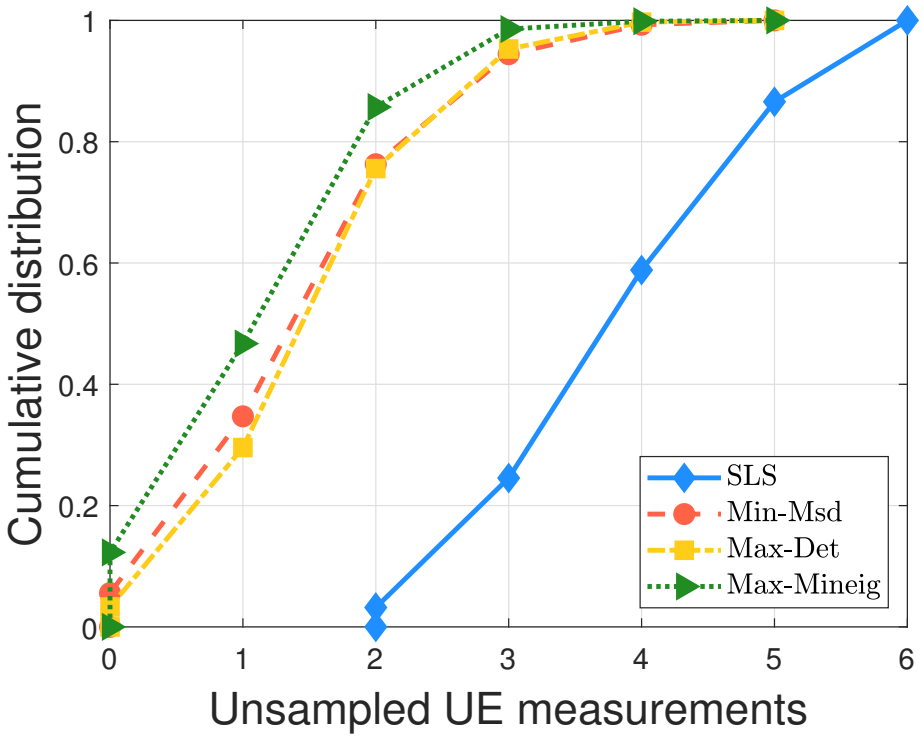
4.7.3.2 Numerical results

In this section, the performance of SLS strategy is evaluated in terms of feedback saving and NMSE.

Figure 4.11 shows the CDF of the number of unsampled UE measurements. SLS is compared with the three baseline algorithms mentioned above. As can be seen, the 60th percentile can be observed when 4 out of 6, i.e., 66.6%, of the UE measurements are unsampled for SLS. The three baseline algorithms presented a worse performance than SLS, as only 2 out of 6, i.e., 33.3%, of the UE measurements are unsampled around the 80th percentile. This is expected because the baseline algorithms sample the vertexes following a certain selection criterion and not aimed at the measurement source, increasing the chance of picking vertexes related to UE measurements. Moreover, Min-Msd and Max-Det strategies provides a similar feedback saving performance and Max-Mineig performs worse than all.

In other words, SLS manages to save an average of a little more than 4 out of 6 UE measurements per user, i.e., 4.27, as shown in Figure 4.12. Meanwhile, none of the three baseline algorithms achieve a feedback saving of at least 2 out of 6 UE measurements on average.

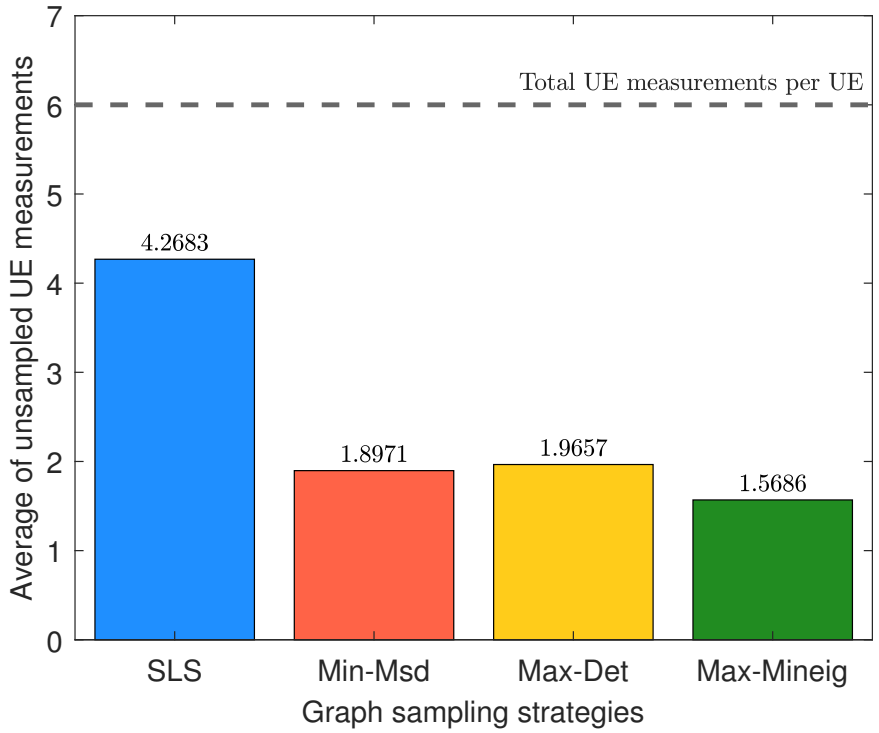
Figure 4.11 – CDF curves of the number of unsampled UE measurements. SLS is compared with the three baseline algorithms.



Source: Created by the author.

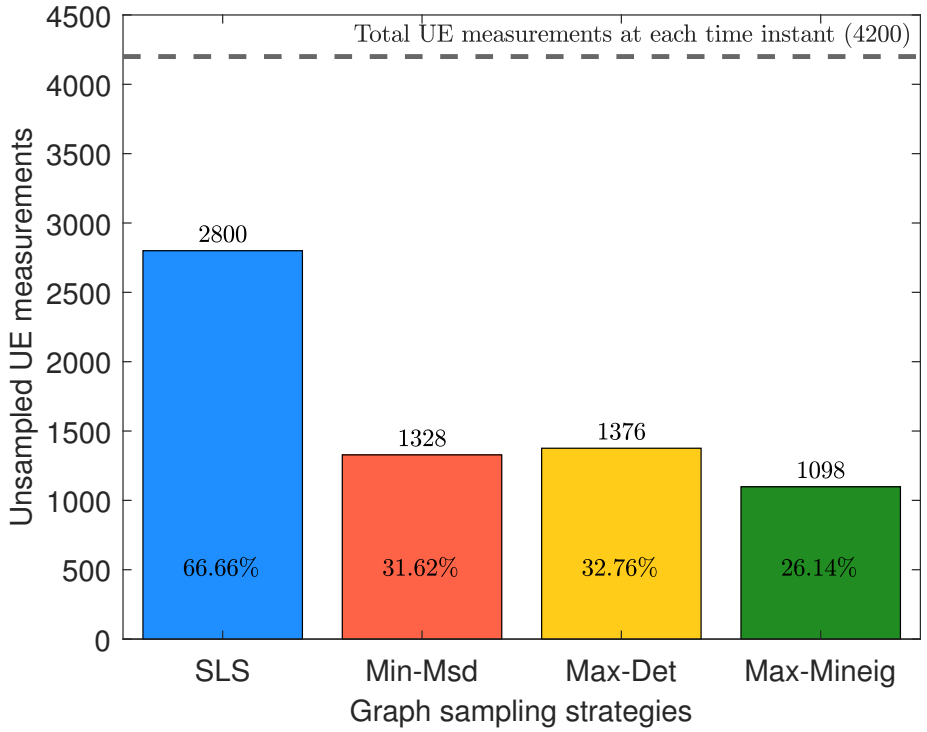
Quantitatively, at each time instant there are $VN_f = 4,200$ UE measurements. SLS can save the feedback of $2,800$ measurements, which represents 66.6%, as shown in Figure 4.13. Meanwhile, the percentage of feedback saving for the Min-Msd, the Max-Det and the Max-Mineig strategies is 31.6%, $32,8$% and 26.1%, respectively. Consequently, SLS is 34% better in

Figure 4.12 – Average of unsampled UE measurements per UE.



Source: Created by the author.

Figure 4.13 – Number of unsampled UE measurements at each time instant.

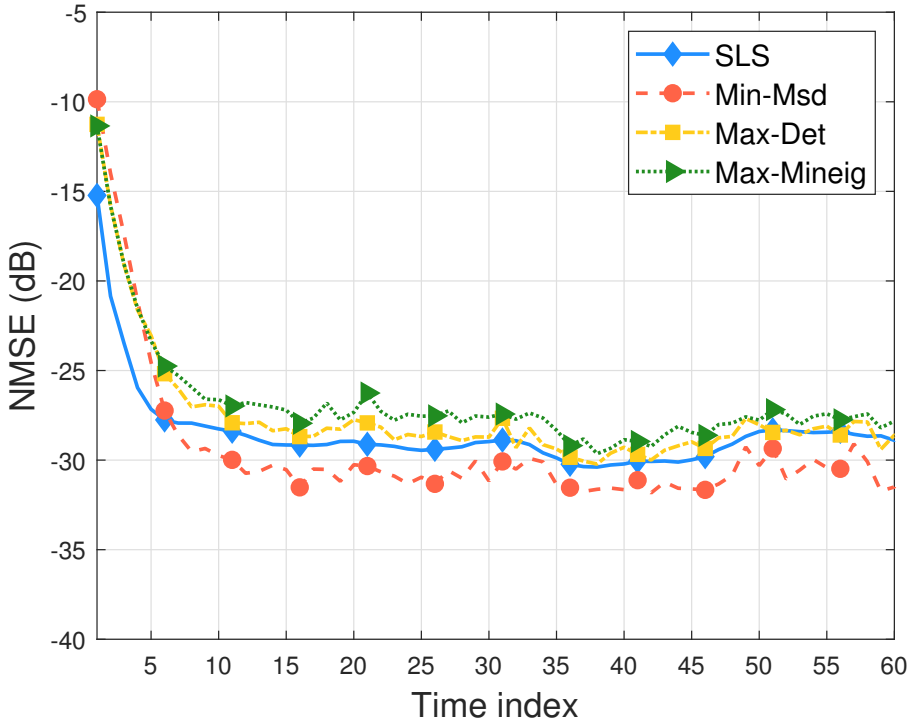


Source: Created by the author.

terms of feedback saving than the best baseline algorithm, i.e., the Max-Det strategy.

Figure 4.14 shows the NMSE against time for the reconstruction error of the O-

Figure 4.14 – Reconstruction NMSE of feature vectors using different sampling strategies: SLS and the three baseline algorithms.



Source: Created by the author.

SGTD algorithm. Again, SLS is compared with the three baseline algorithms. As can be noticed, O-SGTD exhibits very close performance in terms of NMSE both for all the baseline algorithms and for SLS. This is expected since all of them manage to maintain subgraph connectivity by taking advantage of the block-diagonal structure, which boosts the reconstruction performance of O-SGTD. However, the Max-Mineig algorithm performs worse than all. Moreover, SLS exhibits a similar behavior as Max-Det and a slight performance loss when compared to the Min-Msd strategy, i.e., approximately 1 dB. However, it is interesting to notice that SLS increases uplink feedback savings over baseline algorithms, while maintaining a good reconstruction performance.

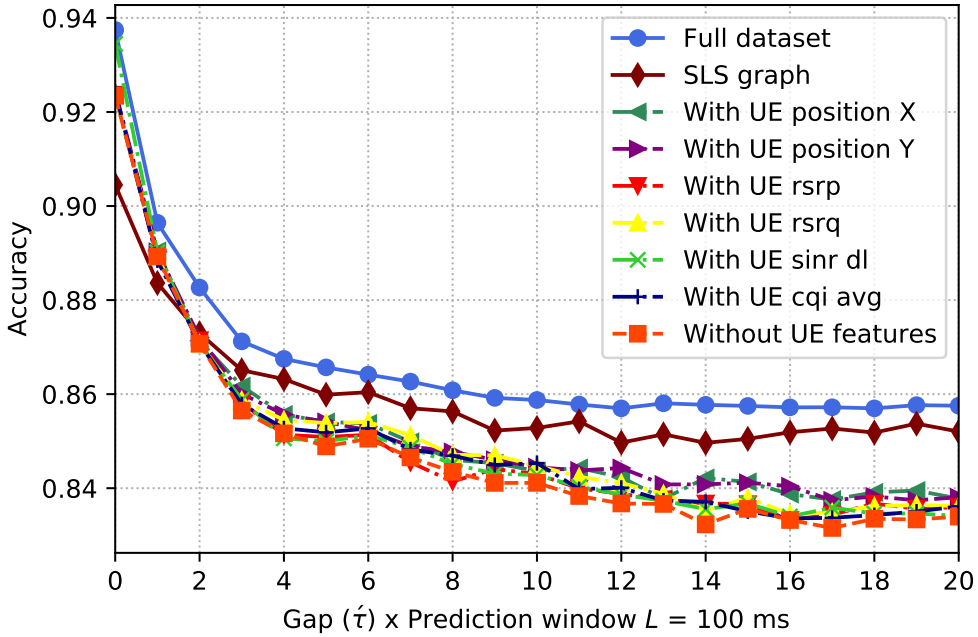
4.7.4 Latency prediction evaluation

The prediction performance of the ML algorithms (i.e., RF and MLP) is evaluated in terms of accuracy and f1-score against different time gaps $\acute{\tau}$. In this context, accuracy stands for the rate of correct predictions. Also, $\acute{\tau} = 0$ means that prediction is done for a packet generation in the present instant, while e.g. $\acute{\tau} = 20$ means that prediction is for a packet generation at a time instant $20L$ ms ahead, i.e., $2,000$ ms as $L = 100$ ms.

The goal here is to evaluate how the predictability of the RF and the MLP algorithms is affected by the assumption of reconstructed feature vectors. The importance of UE measurements in the prediction problem is also assessed by assuming different subsets of UE measurements as features, including the case of an empty subset. A *full dataset* case, i.e., when

all UE measurements are perfectly fed back, is also evaluated. When sampling/reconstruction applies, O-SGTD is evaluated with SLS and the three baseline algorithms.

Figure 4.15 – Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS.

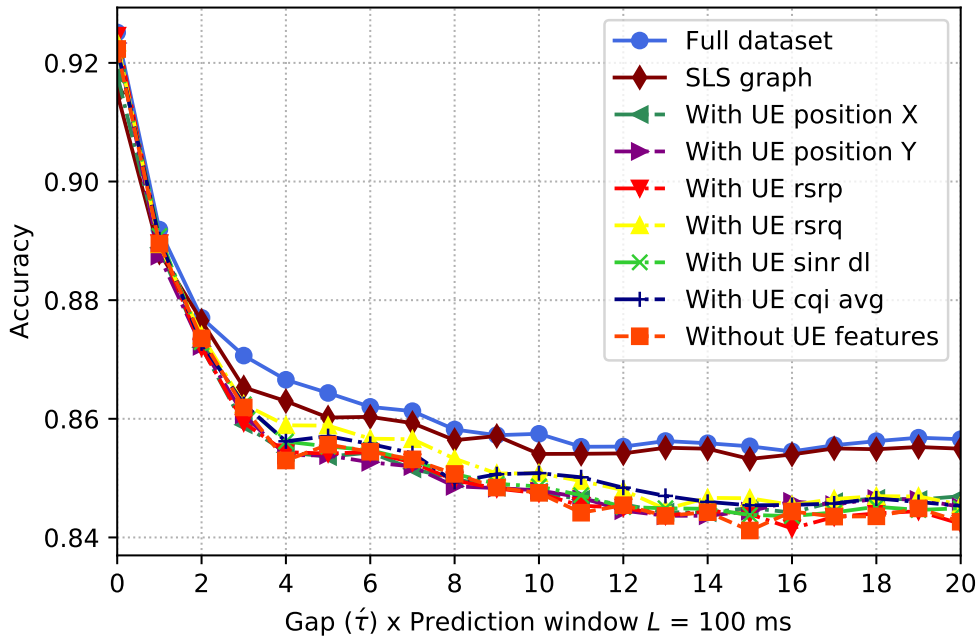


Source: Created by the author.

Figure 4.15 shows the accuracy of RF against different time gaps $\acute{\tau}$. RF is evaluated with full dataset, without UE measurement and for each UE measurement individually. It is also assessed with reconstructed features through O-SGTD with SLS. All the curves have a larger initial decrease for the first five gaps values. However, once the gap length is increased, the decrease in performance slows down. As expected, the accuracy of RF with full dataset is better than all the other cases, which indicates that in general UE measurements are relevant for the prediction problem. RF with reconstructed feature vectors provides a good accuracy with a small performance loss compared to the full dataset case, and a good performance gain compared to the other cases for $\acute{\tau} > 2$. Further, accuracy curves in which is assumed one or no UE measurement lead to a worse performance in general. For instance, for $\acute{\tau} = 20$, RF with full dataset has an accuracy of around 0.86, while RF with O-SGTD and SLS has approximately a 0.855 accuracy. When up to one UE measurement is considered, RF has accuracy no better than 0.84 at $\acute{\tau} = 20$.

The same sort of analysis is also carried out for the MLP algorithm. To this end, Figure 4.16 shows the accuracy of MLP against different time gaps $\acute{\tau}$. As can be seen, the accuracy of MLP is slightly worse than the RF when full dataset is assumed. However, note that MLP with reconstructed feature vectors provides a lower performance loss compared to the full

Figure 4.16 – Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS.



Source: Created by the author.

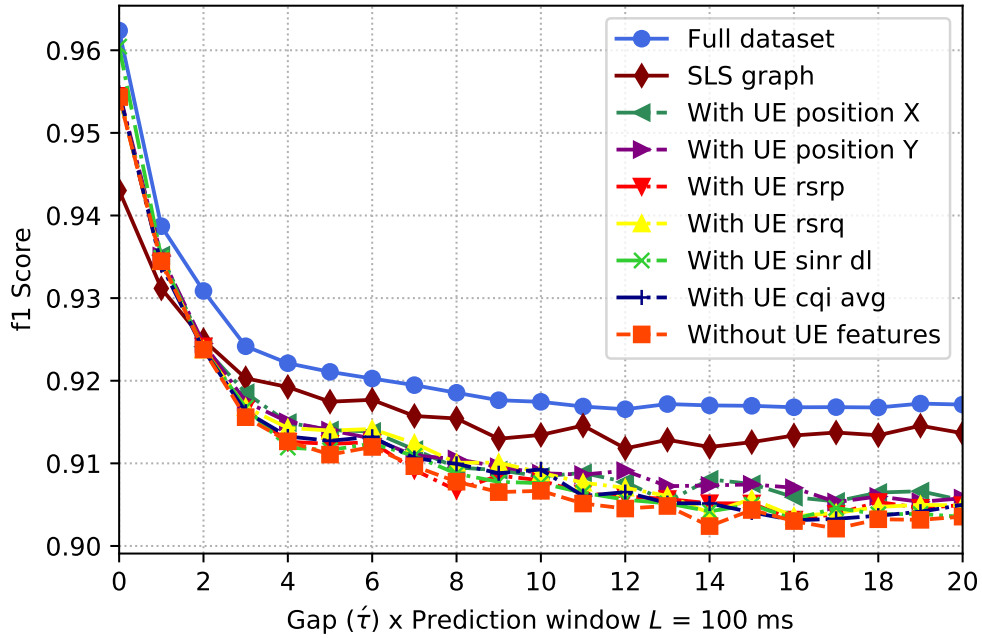
dataset case than the RF algorithm. Moreover, the accuracy curves in which is assumed one or no UE measurement performs better when the MLP algorithm is tested, decreasing its performance for the RF. This indicates that in general UE measurements are more relevant for the prediction problem when the RF algorithm is used.

Figure 4.17 illustrates the f1-score of the RF against different time gaps $\acute{\tau}$, and assuming the same sort of analysis as in Figure 4.15. Specifically, Figures 4.17a and 4.17b show the f1-score for the classes *in time* and *delayed*, respectively. As can be seen, the f1-score of the class *in time* exhibits better performance than the *delayed* for all the curves. This result is expected because the *in time* class is better represented for UE load due to a high network performance. In general terms, the f1-score curves exhibits a similar behavior than the accuracy illustrated in Figure 4.15. That is, the curves that represent the use of reconstructed feature vectors at input exhibit a very close performance to RF with full dataset, and a good performance gain compared to the other cases for $\acute{\tau} > 2$.

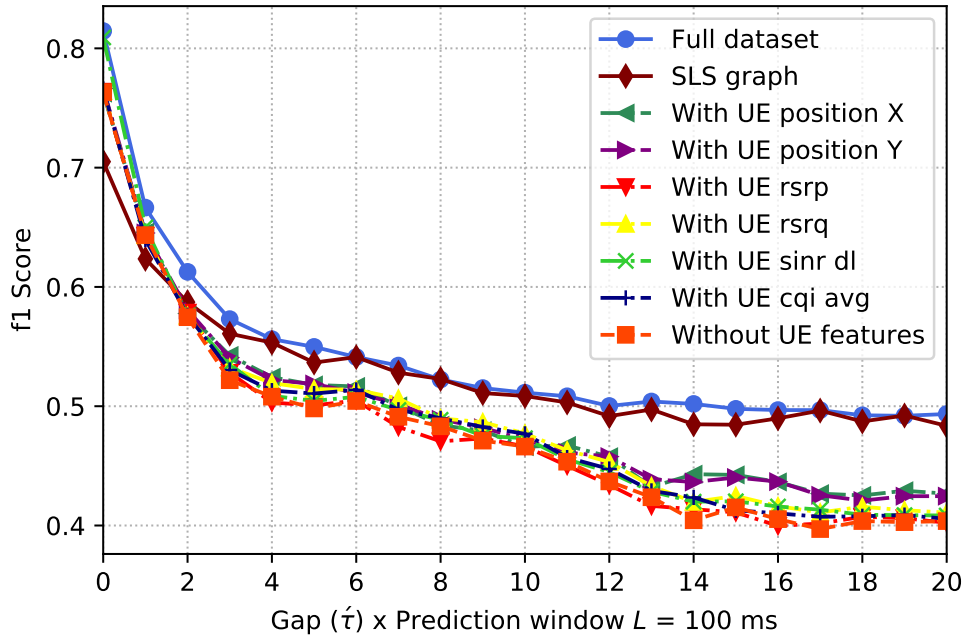
The same sort of analysis is also carried out for the MLP algorithm. To this end, Figures 4.18a and 4.18b show the f1-score of the MLP against different time gaps $\acute{\tau}$ for the classes *in time* and *delayed*, respectively. In general terms, the f1-score curves exhibit a similar behavior the accuracy illustrated in Figure 4.16, i.e., the curves that represent the full dataset and the reconstructed feature vectors, respectively, exhibit a very close performance. Moreover, the MLP algorithm curves achieve a just slightly better performance than RF algorithm in terms of the f1-score metric for both classes.

Figure 4.17 – f1-score of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS.

(a) Class *in time*.



(b) Class *delayed*.

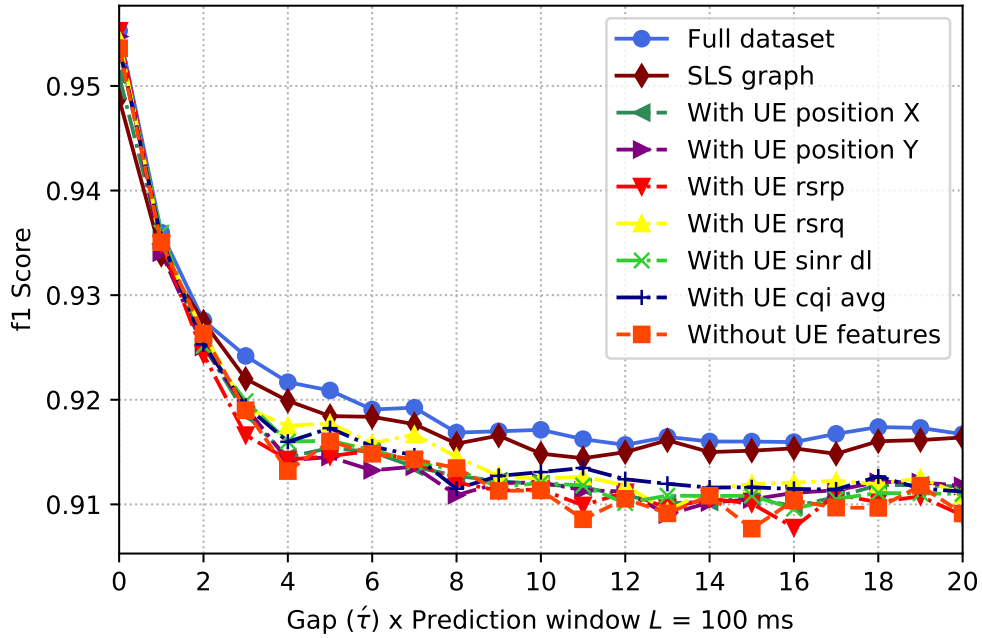


Source: Created by the author.

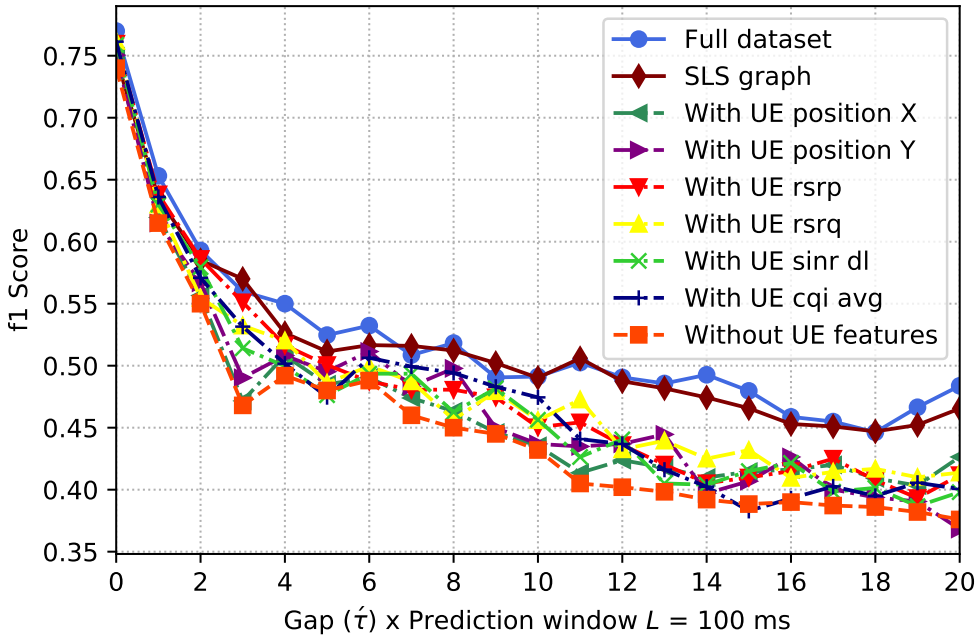
Figure 4.19 also shows the accuracy of RF against different time gaps $\acute{\tau}$. In this case, RF is evaluated with full dataset, without UE measurement, with reconstructed features through O-SGTD with SLS, and finally with up to three UE measurement randomly chosen. As can be

Figure 4.18 – f1-score of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) each of the UE measurements individually, iv) reconstructed feature vectors through O-SGTD with SLS.

(a) Class *in time*.



(b) Class *delayed*.

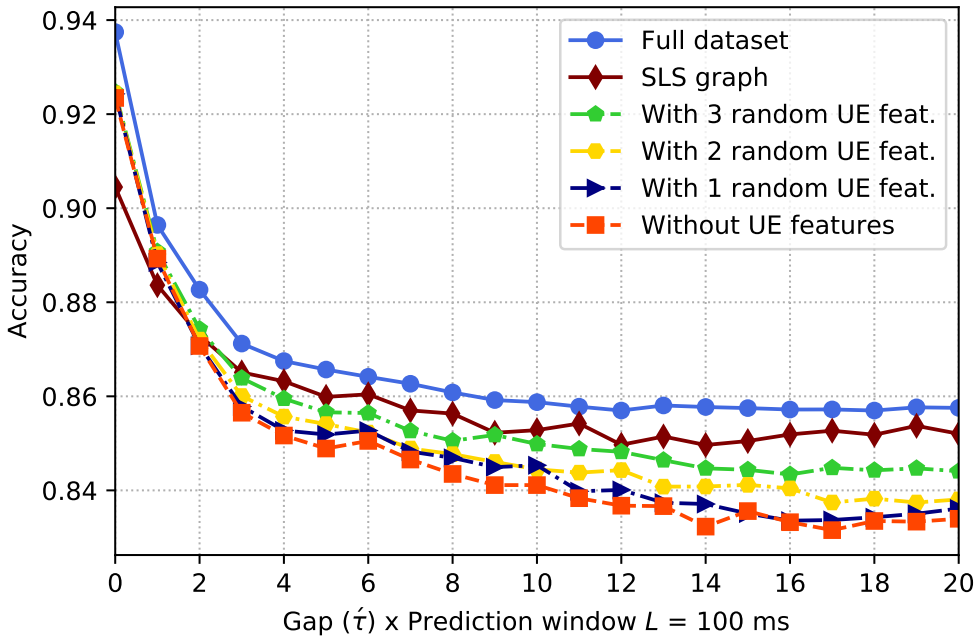


Source: Created by the author.

seen, the curves representing the full dataset case and O-SGTD with SLS are the same as in the previous Figure 4.15. Both cases outperform the three curves that represent the cases with up to three randomly chosen UE measurements. That is, RF with reconstructed feature vectors can still

be better than randomly selecting three UE measurements to be fed back and used as feature.

Figure 4.19 – Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) up to three UE measurements randomly chosen, iv) reconstructed feature vectors through O-SGTD with SLS.



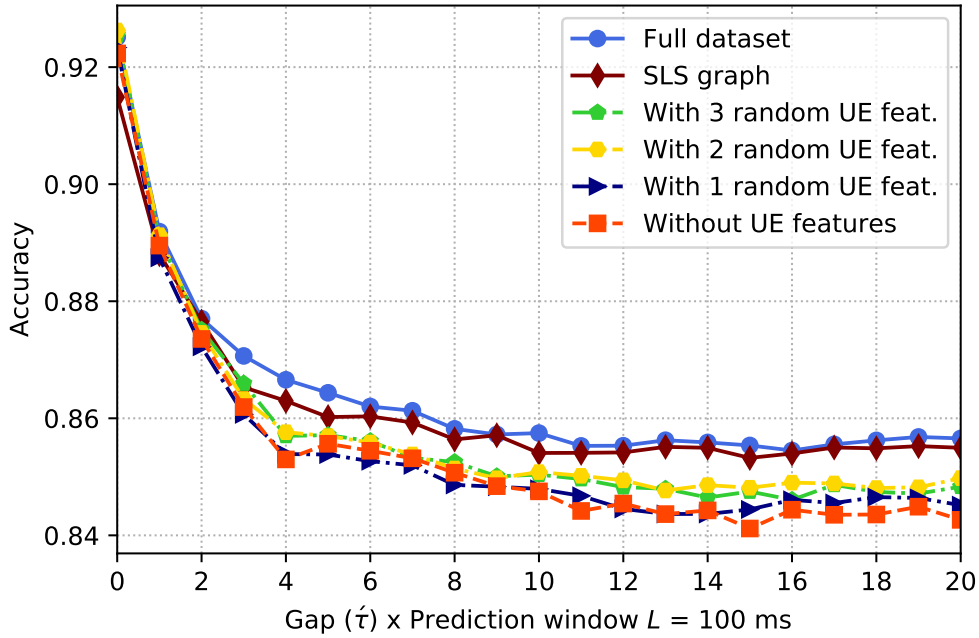
Source: Created by the author.

The same sort of analysis is also carried out for the MLP algorithm in Figure 4.20, in which the curves that represent one, two and three UE measurements randomly chosen, respectively, are compared with the reconstructed features through O-SGTD with SLS. In general, the same behaviors described for the RF case are observed for all the MLP curves. However, note that the accuracy curves that represent up to three UE measurement randomly chosen, perform better when the RF algorithm is tested, decreasing its performance for MLP.

Moreover, an analysis of the f1-score was performed for both RF and MLP, assuming the same input configurations of UE measurements as in Figures 4.19 and 4.20, respectively. These figures were not included since, in general terms, the f1-score curves exhibit a similar accuracy behavior

Figure 4.21 shows the accuracy of RF against different time gaps, but now evaluating different sampling strategies in case of feature vector reconstruction. More specifically, O-SGTD is evaluated with SLS and also with the three baseline algorithms mentioned above. The curves for full dataset case, O-SGTD with SLS, and RF without UE measurements are the same as in the previous Figures 4.15 and 4.19. As can be seen, RF with O-SGTD exhibits very close performance in terms of accuracy for all gaps values, both for the three baseline algorithms and for SLS. This is expected since NMSE curves with very close performance cause a small impact on the RF accuracy. However, RF with Max-Mineig performs worse than all. Moreover, the

Figure 4.20 – Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) up to three UE measurements randomly chosen, iv) reconstructed feature vectors through O-SGTD with SLS.



Source: Created by the author.

proposed SLS exhibits a similar behavior than Max-Det and a slight performance loss when compared to the Min-Msd sampling strategy, fundamentally for gaps values greater or equal than 12. However, it indicates that the proposed SLS strategy can lead to a considerable feedback saving without significantly impacting the prediction accuracy of the RF.

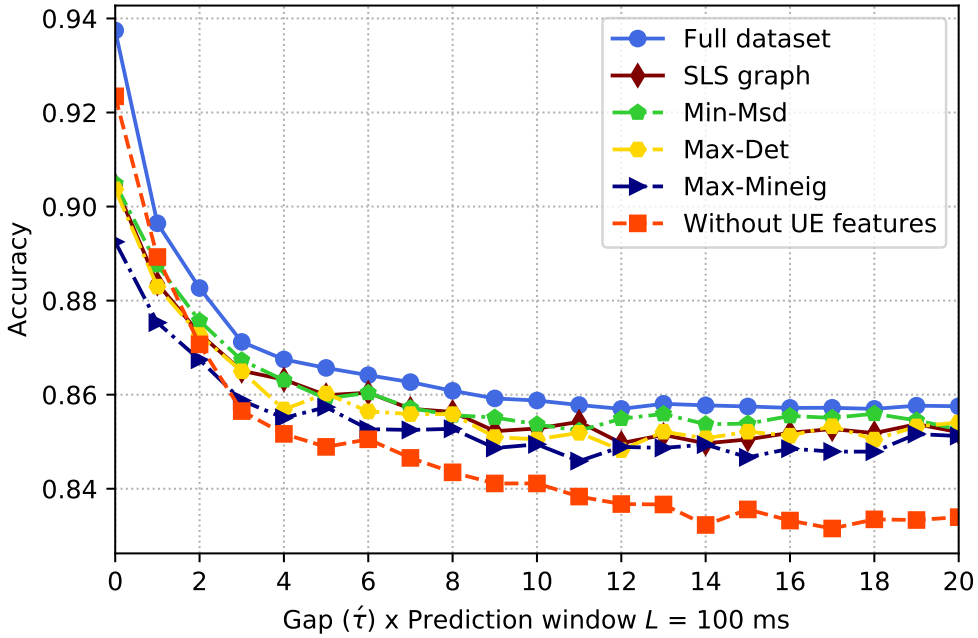
The same sort of analysis is also carried out for the MLP algorithm. To this end, Figure 4.22 shows the accuracy of MLP against different time gaps $\acute{\tau}$. As can be seen, the MLP algorithm with reconstructed feature vectors provides a lower performance loss compared to the full dataset case than the RF algorithm. Moreover, the accuracy curves in which is assumed the three baseline sampling algoritms perform better when the MLP algorithm is tested, decreasing performance for the RF.

Moreover, it was performed the analysis of the f1-score performance. Again, the f1-score curves exhibit a similar behavior the accuracy illustrated in Figures 4.21 and 4.22, for both RF and MLP, respectively.

4.8 Chapter summary

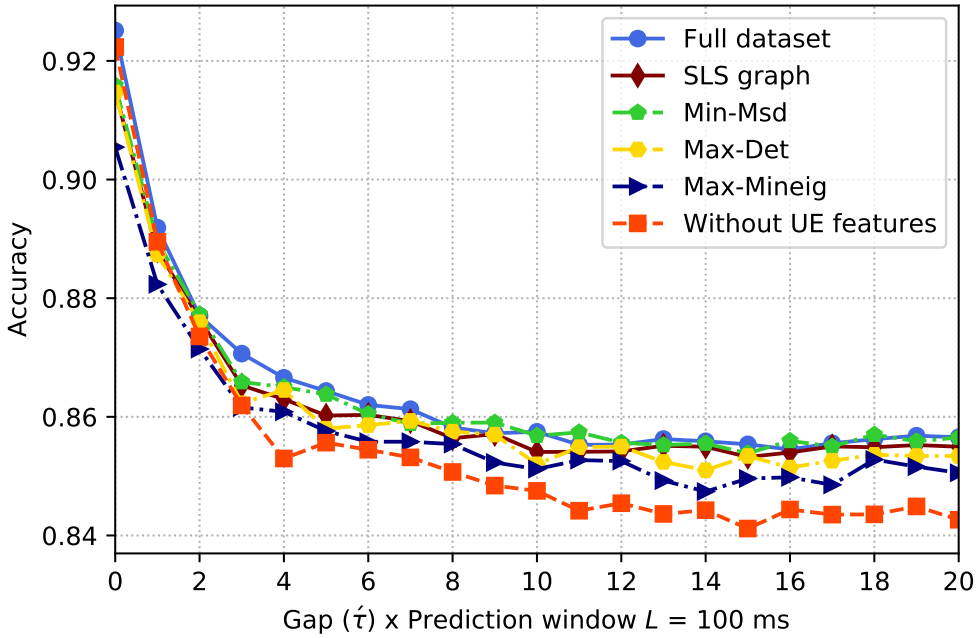
In this chapter, it was addressed the concept of QoS predictability in a C-V2X scenario. The goal was to predict whether a packet can be delivered within a predetermined latency constraint. Such a latency prediction was modeled as a binary classification problem to be handled by a machine learning (ML) algorithms, i.e., RF and MLP. Measurements taken

Figure 4.21 – Accuracy of the RF algorithm evaluated with i) full dataset, ii) no UE measurement, iii) three sampling baseline algorithms, iv) reconstructed feature vectors through O-SGTD with SLS.



Source: Created by the author.

Figure 4.22 – Accuracy of the MLP algorithm evaluated with i) full dataset, ii) no UE measurement, iii) three sampling baseline algorithms, iv) reconstructed feature vectors through O-SGTD with SLS.



Source: Created by the author.

on both BS and UE sides were considered as features to the ML-based predictor carried out on the network side. However, the uplink feedback channel was assumed to be limited. To reduce the need for feeding UE measurements back to the network, GSP tools were employed to i) model feature vectors as graph signals, ii) downsample feature vectors along the entries related to UE measurements, and iii) reconstruct the missing entries on the network side. An online signal reconstruction strategy that preserves the signal smoothness over the graph was adopted. A sampling strategy, namely SLS, that i) exploits block-diagonal structure of the graph Laplacian and ii) reduces the UE measurement feedback was proposed and evaluated. A dataset with BS and UE measurements taken in a typical C-V2X scenario was used for evaluation purposes. In terms of reconstruction error and feedback saving, SLS outperforms some baseline sampling algorithms. As for prediction accuracy, simulation results showed that, as expected, both the RF and MLP with reconstructed feature vectors as input have a small accuracy loss when compared to full dataset availability case. Quantitatively, SLS can save the feedback of 2800 UE measurements at each time instant, which represents the 66.6% of the UE measurements using the ML-based algorithms with full dataset. On the other hand, it has good accuracy when compared to cases when smaller subsets of UE measurements are considered, and when the three baseline sampling algorithms are used. The results then indicate that SLS can lead to a feedback saving without significantly impacting the prediction accuracy of the ML-based algorithms in the prediction problem.

5 CONCLUSIONS AND PERSPECTIVES

5.1 Conclusions

Mobile communications in 5G scenarios still face significant challenges due to the adoption of real-time applications and mission-critical services with stringent requirements, such as low latency and high reliability.

In order to tackle some of these challenges, the main purpose of this thesis was to study solutions for prediction problems in mobile communications. These solutions are based on graph signal processing (GSP) jointly with supervised learning techniques to address two relevant use cases of mobile communication systems, namely: (i) beam management and, (ii) increased feedback channel burden.

The literature review presented in Chapter 1 helped us to identify alternative solutions to beam management problem in highly directional communications and different approaches from a QoS prediction perspective in URLLC. Moreover, it was showed the advantages of using ML-based solutions over classical methods based on system modeling and/or signal processing, mainly in non-stationary scenarios. However, ML-based prior works assumed full knowledge of actual data in the network, including all UE measurements, and did not address the problem of UE measurement feedback signaling reduction.

To the best of our knowledge, no study has been performed to improve the feedback saving on mobile communication systems using sampling and reconstruction tools based on GSP. Then, for a better understanding of the graph-based modeling of signals and their related graph signal processing tools, in Chapter 2 some graph-theoretic definitions were introduced along with some of the tools and concepts that were employed throughout this thesis.

The key contributions of this thesis, in Chapters 3 and 4, addressed the use of GSP-based tools to reduce the UE measurements reporting and improve the robustness of the ML-based prediction solutions.

In Chapter 3, it was proposed and evaluated a tracking strategy based on supervised learning and GSP. First, it was proposed a K-NN-based algorithm to track and predict the channel state, relying on the availability of a HUD set for the purpose of offline training. Then, as preliminary stages of the K-NN algorithm, it was introduced sampling and reconstruction strategies based on GSP, which was referred to as K-NN-R, in order to efficiently use the uplink feedback channel. Simulation results showed that the use of K-NN provided better estimation and prediction performance in terms of NMSE and successful hit rate, than three classical tracking algorithms that were used as baseline. It was showed that while the baseline approaches and K-NN demand full beam sweep and reporting, the proposed K-NN-R framework selects only a few beam pairs at random to be monitored and sampled over time. With respect to the simulation results, it was concluded that the performance of K-NN-R is not affected much in terms of

estimation/prediction error. It was also concluded that, only half of the beam pairs throughout a significantly lower observed temporal window were enough to reliably predict the channel state some time ahead on the BS side. Finally, it was showed that our proposed K-NN-R exhibited lower computational complexity when compared to the three baseline approaches, especially in mm-Wave scenarios that usually lead to the use of a large antenna arrays.

In Chapter 4, it was exploited the graph structure that represents a C-V2X measurement dataset in order reduce the amount of reported UE measurements. In this context, it was demonstrated that, under certain conditions, the C-V2X dataset can present a block-diagonal structure indicating that GSP tools can be applied into each sub-graph independently. Taking advantage of this, it was proposed and employed GSP-based sampling and reconstruction tools, in order to reduce the need for feeding UE measurements back to the network. Specifically, it was proposed and evaluated a sampling strategy, namely SLS, that i) exploits block-diagonal structure of the graph Laplacian and ii) reduces the UE measurement feedback. Moreover, it was adopted an online signal reconstruction strategy that preserves the signal smoothness over the graph. As a means of evaluating our graph-based proposal, it was adopted an ML-based latency prediction problem that sought to predict whether a packet could be delivered within a predetermined latency constraint. Formulated as a binary classification problem, measurements taken on both the BS and the UE sides worked as features to feed ML-based predictors at the network side. However, it was considered that the uplink feedback channel for UE measurement reporting was limited, and the GSP-based sampling/reconstruction tools were used to reduce the feedback channel burden. With respect to the simulation results, it was showed that the proposed SLS outperforms some baseline sampling algorithms in terms of reconstruction error and feedback saving. It was also showed that both the RF and MLP with reconstructed feature vectors as input have a small prediction accuracy loss when compared to full dataset availability case. Quantitatively, SLS managed to save the feedback of 2800 UE measurements at each time instant, which represented the 66.6% of the UE measurements using the ML algorithms with full dataset. On the other hand, it was reached good accuracy when compared to cases when smaller subsets of UE measurements were considered, and when the baseline sampling algorithms were used. The simulation concluded that the proposed SLS can lead to a feedback saving without significantly impacting the prediction accuracy of the ML-based algorithms in the latency prediction problem.

5.2 Future works

It follows a list of ideas for extensions of this thesis in future works. Regarding both beam tracking and latency prediction problems:

- **One may evaluate the problems addressed in this thesis leveraging different machine learning tools**: Although the supervised learning methods provided promising results, they need to create the features and labels sets before proceeding with the

training and testing steps. This represent a challenge for efficient and real-time prediction solutions as in latency prediction problem, in which the package labeling process can be computationally expensive, especially when applied to higher-dimensional data. Moreover, the scheduled HUD refresh operations in the beam tracking problem can be time-consuming and make it difficult to optimize this mechanism, since observation vectors and their corresponding state vectors must be frequently updated to avoid channel aging. In this context, it is envisaged the use of algorithms that learn to react to the environment on their own, such as reinforcement learning and deep learning;

- **One may evaluate the time-varying graph signal as approximately bandlimited graph signal**: In this thesis, reconstruction techniques based on GSP was exploited, in which the graph signal it was assumed as smooth in the vertex domain. However, it is natural in many graph-based tools to also assume that the time-varying graph signal is somehow bandlimited, or approximately bandlimited, in the frequency domain. In this context, it is envisaged the study and evaluation of graph-based reconstruction algorithms that also exploit the graph signal properties in the frequency domain, in order to improve the online reconstruction and tracking time-varying graph signals in noisy environments.

  Concerning beam tracking problem addressed in Chapter 3:

- **A study and evaluation of prediction techniques can also be made on time-varying graphs as a complement of the proposed K-NN-R framework**: In this work, time series data was modeled and pre-processed as signals defined over graphs, and then, it was made the forecasting using a ML-based strategy. However, tackle the spatial prediction and temporal prediction tasks separately can incur failures to capture the inherent coupling between time and space domains of time series data. Especially, when lower sampling rates are assumed. In this context, it is envisaged the study and evaluation of graph-based strategies, which jointly interpolating and forecasting the time-varying graph signal over the underlying graph.

- **One may evaluate K-NN-R with nonrandom graph sampling schemes**: In this thesis, it was assumed a random sampling mechanism in the beam tracking problem. However, the chosen sampling method is crucial to the graph-based step performance, reducing the sampling rate and creating the initial conditions for the reconstruction strategy, in the sense of minimizing reconstruction error. In this context, it is envisaged the study and evaluation of nonrandom graph-based sampling strategies, in order to seek the optimal sampling sets for graph signal estimation. Specifically, it is of great interest to explore sampling strategies in which the graph signals will be assumed to be bandlimited or approximately bandlimited;

Regarding the latency prediction problem addressed in Chapter 4:

- **One can evaluate the impact of feature vector reconstruction in different prediction tools and problems**: In this thesis, it was evaluated how the predictability of the ML-based algorithms was affected by the assumption of reconstructed feature vector in a latency prediction problem. However, the abundant sources of information and the open challenges in 5G networks provide countless opportunities to develop and evaluate graph-based sampling and reconstruction strategies;

- **One can considering clustering aspects and/or optimal sampling strategies in the different subgraph types, in order to optimize the SLS strategy and further improve the signaling feedback saving**: In this thesis, it was proposed a deterministic sampling strategy, namely SLS, in which sampling rules are defined based on the block-diagonal structure of the Laplacian and the measurement source. From these rules, SLS creates the basis for the use of optimized sampling schemes at each subgraph types, in datasets that have these diagonal block characteristics in the graph domain. Regardless of the sampling strategy assumed, the first rule will have to be fulfilled to preserve the structure and connectivity of the graph. As a consequence, trivial subgraphs and MV-BS subgraphs will always be sampled. However, it is important to optimize the vertexes sampling in the MV-UE and MV-BU subgraph types. In the MV-UE case, it is envisaged the use of a nonrandom sampling strategy, in order to seek the optimal sampling sets. Moreover, for the case of MV-BU subgraphs, it would be possible to adaptively analyze the need to sample some vertexes related to UE measurements, to achieve a desired accuracy in the prediction step.

REFERENCES

1 AL-FUQAHA, A. et al. Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications. **IEEE Communications Surveys Tutorials**, v. 17, n. 4, p. 2347–2376, 2015. DOI: 10.1109/COMST.2015.2444095.

2 LARSSON, E. et al. Massive MIMO for Next Generation Wireless Systems. **Submitted to IEEE Commun. Mag.**, v. 52, n. 2, p. 186–195, Feb. 2014. DOI: 10.1109/MCOM.2014.6736761.

3 SUN, S. et al. MIMO for Millimeter-Wave Wireless communications: Beamforming, Spatial Multiplexing, or Both? v. 52, n. 12, p. 110–121, 2014. DOI: 10.1109/MCOM.2014.6979962.

4 3GPP. **3GPP Specification Serie 38**. 2018. Available from: <http://www.3gpp.org/DynaReport/38-series.htm>. Visited on: 19 Feb. 2021.

5 _____. **3GPP Release 15**. July 2018. Available from: <http://www.3gpp.org/release-15>. Visited on: 19 Feb. 2021.

6 TAHERIBAKHSH, M. et al. **5G Implementation: Major Issues and Challenges**. In: 2020 25th International Computer Conference, Computer Society of Iran (CSICC). [S.l.: s.n.], 2020. P. 1–5. DOI: 10.1109/CSICC49403.2020.9050110.

7 BOGALE, T. E.; LE, L. B. Massive MIMO and mmWave for 5G Wireless HetNet: Potential Benefits and Challenges. v. 11, n. 1, p. 64–75, Mar. 2016. DOI: 10.1109/MVT.2015.2496240.

8 DENG, R. et al. Intermittent CSI Update for Massive MIMO Systems With Heterogeneous User Mobility. **IEEE Transactions on Communications**, v. 67, n. 7, p. 4811–4824, 2019. DOI: 10.1109/TCOMM.2019.2911575.

9 JIANG, Z. et al. AI-Assisted Low Information Latency Wireless Networking. **IEEE Wireless Communications**, v. 27, n. 1, p. 108–115, 2020. DOI: 10.1109/MWC.001.1900279.

10 WANG, Y.; NARASIMHA, M.; HEATH, R. W. **MmWave Beam Prediction with Situational Awareness: A Machine Learning Approach**. In: 2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC). [S.l.: s.n.], 2018. P. 1–5. DOI: 10.1109/SPAWC.2018.8445969.

11 3GPP. **Beam Measurement Quantity Reporting**. [S.l.], Oct. 2017. Available from: <http://www.3gpp.org/ftp/tsg_ran/wg2_rl2/TSGR2_99bis/Docs/>. Visited on: 19 Oct. 2018.

12 LIU, Y. et al. **Deep Reinforcement Learning-Based Beam Tracking for Low-Latency Services in Vehicular Networks**. In: p. 1–7. DOI: 10.1109/ICC40277.2020.9148759.

13 ORTEGA, A. et al. Graph Signal Processing: Overview, Challenges, and Applications. **Proceedings of the IEEE**, v. 106, n. 5, p. 808–828, 2018.

14 SANDRYHAILA, A.; MOURA, J. M. F. Discrete Signal Processing on Graphs. **IEEE Transactions on Signal Processing**, v. 61, n. 7, p. 1644–1656, 2013.

15 VA, V.; VIKALO, H.; HEATH, R. W. **Beam tracking for mobile millimeter wave communication systems**. In: 2016 IEEE Global Conference on Signal and Information Processing (GlobalSIP). [S.l.: s.n.], Dec. 2016. P. 746–747. DOI: 10.1109/GlobalSIP.2016.7905941.

16 CHOI, J. et al. Millimeter-Wave Vehicular Communication to Support Massive Automotive Sensing. **IEEE Communications Magazine**, v. 54, n. 12, p. 160–167, 2016. DOI: 10.1109/MCOM.2016.1600071CM.

17 GIORDANI, M.; ZANELLA, A.; ZORZI, M. **Millimeter wave communication in vehicular networks: Challenges and opportunities**. In: 2017 6th International Conference on Modern Circuits and Systems Technologies (MOCAST). [S.l.: s.n.], 2017. P. 1–6. DOI: 10.1109/MOCAST.2017.7937682.

18 SHIMIZU, T. et al. **Millimeter Wave V2X Communications: Use Cases and Design Considerations of Beam Management**. In: 2018 Asia-Pacific Microwave Conference (APMC). [S.l.: s.n.], 2018. P. 183–185. DOI: 10.23919/APMC.2018.8617303.

19 VA, V.; CHOI, J.; HEATH, R. W. The Impact of Beamwidth on Temporal Channel Variation in Vehicular Channels and Its Implications. **IEEE Transactions on Vehicular Technology**, v. 66, n. 6, p. 5014–5029, 2017. DOI: 10.1109/TVT.2016.2622164.

20 ABARI, O.; HASSANIEH, H.; KATABI, D. **Millimeter Wave Communications: From Point-to-Point Links to Agile Network Connections**. In: p. 169–175. DOI: 10.1145/3005745.3005766.

21 SONG, X.; HAGHIGHATSHOAR, S.; CAIRE, G. A Scalable and Statistically Robust Beam Alignment Technique for Millimeter-Wave Systems. **IEEE Transactions on Wireless Communications**, v. 17, n. 7, p. 4792–4805, 2018. DOI: 10.1109/TWC.2018.2831697.

22 HEATH, R. W. et al. An Overview of Signal Processing Techniques for Millimeter Wave MIMO Systems. **IEEE Journal of Selected Topics in Signal Processing**, v. 10, n. 3, p. 436–453, 2016. DOI: 10.1109/JSTSP.2016.2523924.

23 SAYEED, A. M. Deconstructing multiantenna fading channels. **IEEE Transactions on Signal Processing**, v. 50, n. 10, p. 2563–2579, Oct. 2002.

24 BAE, J.; LIM, J. H.; CHOI, J. W. New beam tracking technique for millimeter wave-band communications. **arXiv preprint arXiv:1702.00276**, Feb. 2017.

25 ZHANG, C.; GUO, D.; FAN, P. **Tracking angles of departure and arrival in a mobile millimeter wave channel**. In: 2016 IEEE International Conference on Communications (ICC). [S.l.: s.n.], May 2016. P. 1–6.

26 LIU, F.; ZHAO, P.; WANG, Z. EKF-Based Beam Tracking for mmWave MIMO Systems. **IEEE Communications Letters**, v. 23, n. 12, p. 2390–2393, 2019. DOI: 10.1109/LCOMM.2019.2940660.

27 LAREW, S. G.; LOVE, D. J. Adaptive Beam Tracking With the Unscented Kalman Filter for Millimeter Wave Communication. **IEEE Signal Processing Letters**, v. 26, n. 11, p. 1658–1662, 2019. DOI: 10.1109/LSP.2019.2944255.

28 LIM, J.; PARK, H.; HONG, D. Beam Tracking Under Highly Nonlinear Mobile Millimeter-Wave Channel. **IEEE Communications Letters**, v. 23, n. 3, p. 450–453, 2019. DOI: 10.1109/LCOMM.2019.2894340.

29 PALACIOS, J.; DE DONNO, D.; WIDMER, J. **Tracking mm-Wave Channel Dynamics: Fast Beam Training Strategies under Mobility**. In: IEEE Conference on Computer Communications (INFOCOM). [S.l.: s.n.], Apr. 2017. P. 1–9. DOI: 10.1109/INFOCOM.2017.8056991.

30 DUAN, Q. et al. **AoD and AoA tracking with directional sounding beam design for millimeter wave MIMO systems**. In: 2015 IEEE 26th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC). [S.l.: s.n.], Aug. 2015. P. 2271–2276. DOI: 10.1109/PIMRC.2015.7343676.

31 GAO, X. et al. Fast Channel Tracking for Terahertz Beamspace Massive MIMO Systems. **IEEE Transactions on Vehicular Technology**, v. 66, n. 7, p. 5689–5696, July 2017. ISSN 0018-9545. DOI: 10.1109/TVT.2016.2614994.

32 LI, J. et al. **Analog beam tracking in linear antenna arrays: Convergence, optimality, and performance**. In: 2017 51st Asilomar Conference on Signals, Systems, and Computers. [S.l.: s.n.], Oct. 2017. P. 1193–1198. DOI: 10.1109/ACSSC.2017.8335540.

33 NAVABI, S. et al. **Predicting Wireless Channel Features Using Neural Networks**. In: p. 1–6. DOI: 10.1109/ICC.2018.8422221.

34 BURGHAL, D.; ABBASI, N. A.; MOLISCH, A. F. **A Machine Learning Solution for Beam Tracking in mmWave Systems**. In: 2019 53rd Asilomar Conference on Signals, Systems, and Computers. [S.l.: s.n.], 2019. P. 173–177. DOI: 10.1109/IEEECONF44664.2019.9048770.

35 EGILMEZ, H. E.; PAVEZ, E.; ORTEGA, A. Graph Learning From Data Under Laplacian and Structural Constraints. **IEEE Journal of Selected Topics in Signal Processing**, v. 11, n. 6, p. 825–841, Sept. 2017. ISSN 1932-4553. DOI: 10.1109/JSTSP.2017.2726975.

36 DI LORENZO, P. et al. Adaptive Graph Signal Processing: Algorithms and Optimal Sampling Strategies. **IEEE Transactions on Signal Processing**, v. 66, n. 13, p. 3584–3598, July 2018.

37 DONG, D. et al. **Laplacian matrix learning for smooth graph signal representation**. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], Apr. 2015. P. 3736–3740. DOI: 10.1109/ICASSP.2015.7178669.

38 MEI, J.; MOURA, J. M. F. **Signal processing on graphs: Estimating the structure of a graph**. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], Apr. 2015. P. 5495–5499. DOI: 10.1109/ICASSP.2015.7179022.

39 MUTLU, A. Y.; BERNAT, E.; AVIYENTE, S. A Signal-Processing-Based Approach to Time-Varying Graph Analysis for Dynamic Brain Network Identification. **Computational and Mathematical Methods in Medicine**, v. 2012, p. 10, 2012. DOI: 10.1155/2012/451516.

40 LEONARDI, N.; VAN DE VILLE, D. Tight wavelet frames on multislice graphs. **IEEE Transactions Signal Processing**, v. 61, n. 13, p. 3357–3367, July 2013.

41 GADDE, A.; ANIS, A.; ORTEGA, A. **Active semi-supervised learning using sampling theory for graph signals**. In: 20TH ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. [S.l.: s.n.], 2014. P. 492–501.

42 GADDE, A.; ORTEGA, A. **A probabilistic interpretation of sampling theory of graph signals**. In: 2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], Apr. 2015. P. 3257–3261. DOI: 10.1109/ICASSP.2015.7178573.

43 ZHU, X.; RABBAT, M. **Graph spectral compressed sensing for sensor networks**. In: 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], Mar. 2012. P. 2865–2868.

44 SHI, X. et al. Infinite impulse response graph filters in wireless sensor networks. **IEEE Signal Processing Letters**, v. 22, n. 8, p. 1113–1117, Aug. 2015.

45 PARK, S. W. et al. Discrete Sibson interpolation. **IEEE Transactions on Visualization and Computer Graphics**, v. 12, n. 2, p. 243–253, Mar. 2006. ISSN 1077-2626. DOI: 10.1109/TVCG.2006.27.

46 MA, S.; GOLDFARB D. AND CHEN, L. Fixed point and Bregman iterative methods for matrix rank minimization. **Mathematical Programming**, v. 128, n. 1, p. 321–353, June 2011. ISSN 1436-4646. DOI: 10.1007/s10107-009-0306-5.

47 BELKIN, M.; MATVEEVA, I.; NIYOGI, P. **Tikhonov regularization and semi-supervised learning on large graphs**. In: 2004 IEEE International Conference on Acoustics, Speech, and Signal Processing. [S.l.: s.n.], May 2004. v. 3, p. iii–1000. DOI: 10.1109/ICASSP.2004.1326716.

48 NARANG, S. K. et al. **Localized iterative methods for interpolation in graph structured data**. In: 2013 IEEE Global Conference on Signal and Information Processing. [S.l.: s.n.], Dec. 2013. P. 491–494. DOI: 10.1109/GlobalSIP.2013.6736922.

49 QIU, K. et al. Time-Varying Graph Signal Reconstruction. **IEEE Journal of Selected Topics in Signal Processing**, v. 11, n. 6, p. 870–883, Sept. 2017. ISSN 1932-4553. DOI: 10.1109/JSTSP.2017.2726969.

50 UYGUNGELEN, S.; AUER, G.; BHARUCHA, Z. **Graph-Based Dynamic Frequency Reuse in Femtocell Networks**. In: 2011 IEEE 73rd Vehicular Technology Conference (VTC Spring). [S.l.: s.n.], 2011. P. 1–6. DOI: 10.1109/VETECS.2011.5956438.

51 ZHIRONG, L. et al. User-oriented graph based frequency allocation algorithm for densely deployed femtocell network. **China Communications**, v. 10, n. 12, p. 57–65, 2013. DOI: 10.1109/CC.2013.6723879.

52 ZHAO, K. et al. **Graph-based joint mode selection and resource allocation scheme for D2D and cellular hybrid network using SCMA**. In: 2016 8th International Conference on Wireless Communications Signal Processing (WCSP). [S.l.: s.n.], 2016. P. 1–5. DOI: 10.1109/WCSP.2016.7752515.

53 CAIDAN ZHAO et al. **A coloring-based cluster resource allocation for ultra dense network**. In: 2016 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC). [S.l.: s.n.], 2016. P. 1–5. DOI: 10.1109/ICSPCC.2016.7753602.

54 XIAO, Z.; FANG, H.; WANG, X. Nonlinear Polynomial Graph Filter for Anomalous IoT Sensor Detection and Localization. **IEEE Internet of Things Journal**, v. 7, n. 6, p. 4839–4848, 2020. DOI: 10.1109/JIOT.2020.2971237.

55 CHEN, M. et al. **Distributed Estimation of Graph Eigenspectra in Mobile Communication Networks**. In: 2020 12th International Conference on Communication Software and Networks (ICCSN). [S.l.: s.n.], 2020. P. 43–50. DOI: 10.1109/ICCSN49894.2020.9139080.

56 ORTEGA, Y. R. et al. Supervised learning and graph signal processing strategies for beam tracking in highly directional mobile communications. **Transactions on Emerging Telecommunications Technologies**, v. 30, n. 9, p. 825–841, July 2019. ISSN 1932-4553. DOI: 10.1002/ett.3687.

57 ORTEGA, Y. R. et al. Graph Signal Processing for QoS Prediction in Cellular V2X Communications. **Transactions on Vehicular Technology**, *Under Review*, Aug. 2020.

58 I., S. D. et al. The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains. **IEEE Signal Processing Magazine**, v. 30, n. 3, p. 83–98, May 2013. ISSN 1053-5888. DOI: 10.1109/MSP.2012.2235192.

59 SHUMAN, I. et al. The Emerging Field of Signal Processing on Graphs: Extending High-Dimensional Data Analysis to Networks and Other Irregular Domains. **IEEE Signal Processing Magazine**, v. 30, Oct. 2012. DOI: 10.1109/MSP.2012.2235192.

60 WILSON, R. **Introduction to Graph Theory**. 4. ed. Harlow, England: Longman Group Limited, 1996. ISBN 9780273728894.

61 CHUNG, F. R. K. **Spectral Graph Theory**. [S.l.]: American Mathematical Society, 1997.

62 QUEEN, C.; WRIGHT, B.; ALBERS, C. Eliciting a directed acyclic graph for a multivariate time series of vehicle counts in a traffc network. **Australian and New Zealand Journal of Statistics**, v. 49, Sept. 2007. DOI: 10.1111/j.1467-842X.2007.00477.x.

63 HE, Y. et al. **Object recognition in images via a factor graph model**. In: _____. **Ninth International Conference on Graphic and Image Processing (ICGIP 2017)**. [S.l.]: SPIE, 2018. v. 10615. International Society for Optics and Photonics, p. 326–335. DOI: 10.1117/12.2303409. Available from: <https://doi.org/10.1117/12.2303409>.

64 GERA, R. et al. Identifying network structure similarity using spectral graph theory. **Applied Network Science**, v. 3, Jan. 2018. DOI: 10.1007/s41109-017-0042-3.

65 SPIELMAN, D. **Algorithms, Graph Theory, and Linear Equations in Laplacian Matrices**. In: p. 2698–2722. DOI: 10.1142/9789814324359_0164.

66 SLAWSKI, M.; HEIN, M. **Estimation of positive definite M-matrices and structure learning for attractive Gaussian Markov Random fields**. [S.l.: s.n.], 2014. arXiv: 1404.6640 [math.ST].

67 PAVEZ, E.; ORTEGA, A. **Generalized Laplacian precision matrix estimation for graph signal processing**. In: 2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], 2016. P. 6350–6354.

68 BIYIKOĞU, T.; LEYDOLD, J.; STADLER, P. F. Laplacian Eigenvectors of Graphs. In: LAPLACIAN Eigenvectors of Graphs: Perron-Frobenius and Faber-Krahn Type Theorems. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. P. 15–27. ISBN 978-3-540-73510-6. DOI: 10.1007/978-3-540-73510-6_2. Available from: <https://doi.org/10.1007/978-3-540-73510-6_2>.

69 PAVEZ, E. et al. **GTT: Graph template transforms with applications to image coding**. In: 2015 Picture Coding Symposium (PCS). [S.l.: s.n.], 2015. P. 199–203.

70 EGILMEZ, H. E. et al. **GBST: Separable transforms based on line graphs for predictive video coding**. In: 2016 IEEE International Conference on Image Processing (ICIP). [S.l.: s.n.], 2016. P. 2375–2379.

71 KURRAS, S.; LUXBURG, U.; BLANCHARD, G. The f-adjusted graph laplacian: A Diagonal modification with a geometric interpretation. **31st International Conference on Machine Learning, ICML 2014**, v. 4, p. 3446–3466, Jan. 2014.

72 PERRAUDIN, N.; VANDERGHEYNST, P. Stationary Signal Processing on Graphs. **IEEE Transactions on Signal Processing**, v. 65, n. 13, p. 3462–3477, July 2017. ISSN 1053-587X. DOI: 10.1109/TSP.2017.2690388.

73 PERRAUDIN, N. et al. **Towards stationary time-vertex signal processing**. In: 2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). [S.l.: s.n.], Mar. 2017. P. 3914–3918. DOI: 10.1109/ICASSP.2017.7952890.

74 VON LUXBURG, U. A Tutorial on Spectral Clustering. **Statistics and Computing**, v. 17, n. 4, p. 395–416, Aug. 2007. DOI: https://doi.org/10.1007/s11222-007-9033-z. eprint: 0711.0189.

75 PAVEZ, E.; EGILMEZ, H. E.; ORTEGA, A. Learning Graphs With Monotone Topology Properties and Multiple Connected Components. **IEEE Transactions on Signal Processing**, v. 66, n. 9, p. 2399–2413, May 2018. ISSN 1053-587X. DOI: 10.1109/TSP.2018.2813337.

76 DONG, X. et al. Learning Graphs From Data: A Signal Representation Perspective. **IEEE Signal Processing Magazine**, v. 36, n. 3, p. 44–63, May 2019. ISSN 1053-5888. DOI: 10.1109/MSP.2018.2887284.

77 THEODORIDIS, S. **Machine Learning: A Bayesian and Optimization Perspective**. 1. ed. Orlando, FL, USA: Academic Press, Inc., 2015. ISBN 9780128015223.

78 CANDES, E. J.; WAKIN, M. B.; BOYD, S. P. Enhancing Sparsity by Reweighted l1 Minimization. **Journal of Fourier Analysis and Applications**, v. 14, n. 5, p. 877–905, Dec. 2008. ISSN 531-5851. DOI: 10.1007/s00041-008-9045-x.

79 BOUGLEUX, S.; ELMOATAZ, A.; MELKEMI, M. **Discrete Regularization on Weighted Graphs for Image and Mesh Filtering**. In: _____. **Scale Space and Variational Methods in Computer Vision**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007. P. 128–139. ISBN 978-3-540-72823-8.

80 CHEN, S. et al. Signal Recovery on Graphs: Variation Minimization. **IEEE Transactions on Signal Processing**, v. 63, n. 17, p. 4609–4624, Sept. 2015. ISSN 1053-587X. DOI: 10.1109/TSP.2015.2441042.

81 CHEN, S. et al. Discrete Signal Processing on Graphs: Sampling Theory. **IEEE Transactions on Signal Processing**, v. 63, n. 24, p. 6510–6523, 2015.

82 PUY, G. et al. Random sampling of bandlimited signals on graphs. **Applied and Computational Harmonic Analysis**, v. 44, n. 2, p. 446–475, 2018. ISSN 1063-5203. DOI: https://doi.org/10.1016/j.acha.2016.05.005. Available from: <http://www.sciencedirect.com/science/article/pii/S1063520316300215>.

83 CANDÈS, E. J.; RECHT, B. Exact Matrix Completion via Convex Optimization. **Found. Comput. Math.**, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 9, n. 6, p. 717–772, Dec. 2009. ISSN 1615-3375. DOI: 10.1007/s10208-009-9045-5. Available from: <https://doi.org/10.1007/s10208-009-9045-5>.

84 BERTSEKAS, D. **Nonlinear Programming**. 2. ed. [S.l.]: Athena Scientific, 1999.

85 TIKHONOV, A.; ARSENIN, V. **Solutions of ill-posed problems**. [S.l.]: Winston, 1977. (Scripta series in mathematics). ISBN 9780470991244. Available from: <https://books.google.com.br/books?id=ECrvAAAAMAAJ>.

86 SCHOENENBERGER, Y.; PARATTE, J.; VANDERGHEYNST, P. **Graph-based denoising for time-varying point clouds**. In: 2015 3DTV-Conference: The True Vision - Capture, Transmission and Display of 3D Video (3DTV-CON). [S.l.: s.n.], July 2015. P. 1–4. DOI: 10.1109/3DTV.2015.7169366.

87 LU, W. et al. Graph- and finite element-based total variation models for the inverse problem in diffuse optical tomography. **Biomed. Opt. Express**, OSA, v. 10, n. 6, p. 2684–2707, June 2019. DOI: 10.1364/BOE.10.002684. Available from: <http://www.osapublishing.org/boe/abstract.cfm?URI=boe-10-6-2684>.

88 RAPPAPORT, T. S. et al. Millimeter Wave Mobile Communications for 5G Cellular: It Will Work! **IEEE Access**, v. 1, p. 335–349, May 2013. ISSN 2169-3536. DOI: 10.1109/ACCESS.2013.2260813.

89 YANG, D.; YANG, L.-L.; HANZO, L. **DFT-Based Beamforming Weight-Vector Codebook Design for Spatially Correlated Channels in the Unitary Precoding Aided Multiuser Downlink**. In: 2010 IEEE International Conference on Communications. [S.l.: s.n.], 2010. P. 1–5. DOI: 10.1109/ICC.2010.5502350.

90 JAMES, G. et al. **An Introduction to Statistical Learning: With Applications in R**. [S.l.]: Springer Publishing Company, Incorporated, 2014.

91 WU, X.; LU, J. **Deterministic coordinate descent algorithms for smooth convex optimization**. In: 2017 IEEE 56th Annual Conference on Decision and Control (CDC). [S.l.: s.n.], Dec. 2017. P. 709–714. DOI: 10.1109/CDC.2017.8263744.

92 LIMA, A. R. **Channel tracking evaluation method based on Kalman filters for Beamtracking**. Feb. 2018. MA thesis – Federal University of Ceará, Fortaleza, Brazil.

93 LEE, J.; GIL, G.; Y. LEE, H. **Exploiting spatial sparsity for estimating channels of hybrid MIMO systems in millimeter wave communications**. In: 2014 IEEE Global Communications Conference (GLOBECOM). [S.l.: s.n.], Dec. 2014. P. 3326–3331. DOI: 10.1109/GLOCOM.2014.7037320.

94 MONTELLA, C. The Kalman Filter and Related Algorithms: A Literature Review, May 2011.

95 LOHMAR, T.; ZAIDI, A.; OLOFSSON, H. e. a. Driving transformation in the automotive and road transport ecosystem with 5G. **Ericsson Technology Review**, Sept. 2019.

96 FERNANDEZ, A. E.; SERVEL, A.; TIPHENE, J. e. a. 5GCAR scenarios, use cases, requirements and KPIs. Version 1.0: **5GCAR/D2.1, Deliverable**, Aug. 2017.

97 LEMA, M. A. et al. Business Case and Technology Analysis for 5G Low Latency Applications. **IEEE Access**, v. 5, p. 5917–5935, 2017. DOI: 10.1109/ACCESS.2017.2685687.

98 SAMBA, A. et al. **Instantaneous throughput prediction in cellular networks: Which information is needed?** In: 2017 IFIP/IEEE Symposium on Integrated Network and Service Management (IM). [S.l.: s.n.], May 2017. P. 624–627.

99 MOREIRA, D. C. et al. **QoS Predictability in V2X Communication with Machine Learning**. In: 2020 IEEE 91st Vehicular Technology Conference (VTC). [S.l.: s.n.], May 2020.

100 TECHNICAL SPECIFICATION GROUP RADIO ACCESS NETWORK. **Study on evaluation methodology of new Vehicle-to-Everything V2X use cases for LTE and NR**. [S.l.: s.n.], Sept. 2018.

101 NETWORK, T. S. G. R. A. **Study on LTE-based V2X Services**. [S.l.: s.n.], June 2016.

102 TECHNICAL SPECIFICATION GROUP RADIO ACCESS NETWORK. **Study on channel model for frequencies from 0.5 to 100 GHz**. [S.l.: s.n.], June 2018.

103 WU, X.; LU, J. **Deterministic coordinate descent algorithms for smooth convex optimization**. In: 2017 IEEE 56th Annual Conference on Decision and Control (CDC). [S.l.: s.n.], Dec. 2017. P. 709–714. DOI: 10.1109/CDC.2017.8263744.

104 PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, v. 12, p. 2825–2830, Oct. 2011.

105 DI LORENZO, P. et al. Adaptive Least Mean Squares Estimation of Graph Signals. **IEEE Transactions on Signal and Information Processing over Networks**, v. 2, n. 4, p. 555–568, 2016. DOI: 10.1109/TSIPN.2016.2613687.

106 CHEN, S. et al. Discrete Signal Processing on Graphs: Sampling Theory. **IEEE Transactions on Signal Processing**, v. 63, p. 6510–6523, 2015.