COLLEGES BRETAGNE LOIRE | SCIENCES POUR L'INGENIEUR ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

THÈSE DE DOCTORAT DE

L'ÉCOLE NATIONALE SUPÉRIEURE
MINES-TÉLÉCOM ATLANTIQUE BRETAGNE
PAYS DE LA LOIRE – IMT ATLANTIQUE

ÉCOLE DOCTORALE N° 648
Sciences pour l'Ingénieur et le Numérique
Spécialité : *Télécommunication*

Par
Cheikh Malainine CHEIKH MELAININE

Self-Configuration of Telecommunication Receivers in Adverse Environments with Bayesian Optimization

Thèse présentée et soutenue à IMT Atlantique, Brest, le 28/05/2026
Unité de recherche : Laboratoire des Sciences et Technologies de l'information de la Communication et de la Connaissance (Lab-STICC), CNRS UMR 6285
Thèse N° : 2026IMT0A540

Rapporteurs avant soutenance :

Philippe CIBLAT	Professeur, Telecom Paris
Laurent ROS	Professeur, Grenoble-INP (Université Grenoble Alpes)

Composition du Jury :

Président :	Thomas BEGIN	Professeur, Université Claude Bernard Lyon 1
Examinateur·trice·s :	Philippe CIBLAT	Professeur, Telecom Paris
	Laurent ROS	Professeur, Grenoble-INP (Université Grenoble Alpes)
	Béatrice TOMASI	Maître de conférences, IMT Atlantique
Dir. de thèse :	François-Xavier SOCHELEAU	Professeur, IMT Atlantique
Co-encadrant :	Arnaud JARROT	Expert senior en mathématiques appliquées et télémétrie, SLB

Invité·e·(s) :

Lionel BENETEAU	Référent transformation numérique, SLB

This PhD was hosted by IMT Atlantique within the Mathematical, Electrical and Engineering Department (Lab-STICC, UMR 6285), and by SLB (formerly Schlumberger) within the Schlumberger Riboud Product Center, Well Construction Division, Telecommunications and Surveying Team, from April 2023 to June 2026.

This PhD was co-financed by SLB and ANRT under a CIFRE convention.





Acknowledgement

First and foremost, I would like to express my warmest and most sincere gratitude to my supervisors, Arnaud Jarrot and François-Xavier Socheleau, for their patience, genuine kindness, and close guidance throughout these past years. Our many insightful discussions, often leading to unexpected ideas and outcomes, made this journey consistently stimulating and intellectually rewarding. Without their mentorship, the successful completion of this PhD would undoubtedly not have been possible. I could not have wished for better supervisors. Thank you both once again.

I would also like to extend my sincere thanks to the SLB and the Telecommunications and Surveying team for providing such a friendly, supportive, and intellectually challenging environment. Being surrounded by this team greatly contributed to both my professional growth and my enjoyment of the PhD journey. I am particularly grateful for the in-depth discussions over coffee breaks and around whiteboards, as well as for the stimulating and enjoyable work we carried out together. I would especially like to thank Arnaud Croux and Lionel Beneteau for making this PhD project possible and for their supportive management.

I am also grateful to IMT Atlantique and the MEE Department for the stimulating and collaborative environment they provided. My visits to the department offered valuable opportunities for scientific growth and allowed me to meet and exchange ideas with inspiring researchers.

I would like to dedicate this achievement to my family. To my mother, my immeasurable source of support and no words could adequately express my gratitude. To my father, an inexhaustible source of inspiration. Thank you for being a role model throughout my life and for your unwavering support and guidance.

Last but certainly not least, I would like to thank my smaller family. To my wife, thank you for your unfailing help and support, for patiently listening to my sometimes lengthy stories about simulations, technical difficulties, and presentations, and, above all, for sharing both the struggles and the joys of this journey with me.

Finally, to my son, Mohamed, thank you for the happiness you bring and for your many enthusiastic attempts to contribute through the keyboard. There is no doubt that science would have advanced had those contributions been preserved.

Table of Contents

Acronyms **17**

Notation **19**

Introduction **23**

1 Receiver fine-tuning problem **29**
1.1 Introduction . . . 29
1.2 Underwater acoustic communications . . . 30
1.2.1 General overview . . . 30
1.2.2 UAC channels . . . 31
1.3 Signal Processing Hyperparameters . . . 38
1.4 Receiver Black-box . . . 44
1.5 Conclusion . . . 49

2 Black-boxes and Bayesian optimization **50**
2.1 Introduction . . . 50
2.2 Black-box Optimization techniques . . . 51
2.3 Bayesian Optimization . . . 53
2.3.1 General approach . . . 53
2.3.2 Surrogates models . . . 55
2.3.3 Common Acquisition Policies . . . 59
2.4 Conclusion . . . 64

3 Search Method **66**
3.1 Introduction . . . 66
3.1.1 Challenge overview . . . 66
3.1.2 Benchmarks . . . 67
3.2 Search method setup . . . 68
3.3 Numerical results . . . 70
3.3.1 Performance on simulated channels . . . 71
3.3.2 Performance on real channel . . . 76
3.4 Conclusion . . . 79

4 Tracking Method **81**
4.1 Introduction . . . 81
4.1.1 Challenge overview . . . 81
4.1.2 Search or Tracking method . . . 82
4.2 Tracking method setup . . . 83
4.3 Test scenario . . . 85
4.4 Single-Agent adaptation method . . . 88
4.4.1 Objective function and implementation . . . 88
4.4.2 Single-agent implementation . . . 89
4.4.3 Results . . . 91
4.4.4 Single-agent limitations . . . 92
4.5 WandS: A multi-agent Tracker . . . 93
4.5.1 WandS features and objective function . . . 93
4.5.2 Time-varying surrogate model . . . 95
4.5.3 Multi-agent approach . . . 96
4.5.4 Simulation results . . . 101
4.5.5 Channel replay . . . 107
4.6 Conclusion . . . 109

5 Decentralized multi-agent optimization **111**
5.1 Introduction . . . 111
5.1.1 Bandit problems . . . 112
5.1.2 Regret analysis . . . 113
5.2 Problem and Notation . . . 114
5.3 X-KB-UCB . . . 116
5.3.1 Arm selection . . . 116
5.3.2 Cross-Agent Kriging–Believer . . . 116
5.4 Theoretical results . . . 119
5.4.1 Augmented agent regrets . . . 119
5.4.2 Maximum information gain . . . 121
5.4.3 KB overconfidence . . . 121
5.4.4 Main theorems . . . 122
5.5 Experimental Results . . . 124
5.5.1 Synthetic static reward function . . . 125
5.5.2 Synthetic Time-Varying reward function . . . 126
5.5.3 Real-data reward function . . . 127
5.6 Conclusion . . . 128

Conclusion **130**
5.7 General conclusion and contributions . . . 130
5.8 Future perspectives . . . 132

6 Appendix **135**

6.1 Search method for Mud-Pulse Telemetry . . . 135
6.1.1 Measurement-While-Drilling . . . 135
6.1.2 MPT objective function . . . 137
6.1.3 Numerical results . . . 138
6.1.4 Conclusion . . . 141
6.2 Chapter 5 . . . 143
6.2.1 Proof of Lemma 1 . . . 143
6.2.2 Theorem 1 (Stationary reward) . . . 144
6.2.3 Theorem 2 (Time-Varying reward) . . . 146
6.2.4 Experimental Details . . . 148

7 Résumé en français **153**
7.1 Chapitre 1 : Problème de réglage fin du récepteur . . . 154
7.2 Chapitre 2 : Fonctions boîtes noires et optimisation bayésienne . . . 156
7.3 Chapitre 3 : Méthode de recherche hors ligne . . . 158
7.4 Chapitre 4 : Méthode de suivi en ligne . . . 160
7.5 Chapitre 5 : Optimisation multi-agent décentralisée . . . 163
7.6 Conclusion générale et perspectives . . . 166

Bibliography **169**

List of Figures

1 Manuscript structure and contributions. 27

1.1 Sound Speed Profile example in warm deep waters. 32
1.2 Characterization of the Loch Ness channel, Scotland, 2023. Top row: instantaneous channel power in the time-delay domain (left) and spreading function (right). Bottom row: power-delay profile (left) and Doppler spectrum (right). Doppler is compensated in these figures. 35
1.3 Packet structure, with a preamble for detection and synchronization, followed by the training sequence used by the equalizer, the payload, and the CRC sequence. 39
1.4 Schematic of the receiver. A Doppler-bank matched filter produces coarse Doppler and timing estimates, which are used by the synchronization stage for timing alignment and Doppler compensation. The Doppler-corrected signal is then equalized by a decision-feedback equalizer, and subsequently processed by a PLL to correct residual frequency offsets prior to symbol detection. 40
1.5 Channel power (dB) in the time-delay domain. Left: channel before Doppler compensation. Right: channe after Doppler compensation using the Doppler-bank matched estimate. 41
1.6 DFE hallucination example. In (a), the top plot shows the filter output norms: the curve marked with –o– corresponds to the feedforward ℓ_2 output norm, and the curve marked with –*– corresponds to the feedback ℓ_2 output norm. The bottom plot shows the SNR evolution during hallucination (red curve —). In (b), the instantaneous channel power in the time–delay domain of the corresponding simulated channel is shown. Doppler is compensated in the right figure. 45
1.7 Five channel realizations (top row) with their corresponding power-delay profiles (bottom row). 47
1.9 Objective function f for (L_A, L_B, K_1, K_2) in the same simulated channel. K_1, and K_2 are shown in log-scale. 48
1.8 Objective function f for four different noise realizations over (L_A, L_B) in the same simulated channel. 48

2.1 Closed-loop process of **one** Bayesian Optimization iteration 55

2.2 (Left) Prior Gaussian process (GP) over a range of hyperparameter values. In the absence of training data, the mean remains flat and uncertainty is high, as reflected by the wide 95% confidence interval (blue region). (Right) Posterior GP after incorporating training observations (blue dots). The mean function (green) exhibits learned structure, while the uncertainty band contracts around observed points, indicating reduced variance in data-rich regions. 56
2.3 GP posterior with an SE kernel for two different lengthscale values. 58
2.4 Qualitative comparison of common acquisition functions. The top row shows the same Gaussian process posterior, including the predictive mean, uncertainty bands, and observed data, while the bottom row displays the corresponding acquisition functions. From left to right, EI concentrates mass near regions that jointly promise improvement and uncertainty, UCB favors uncertain regions according to the exploration schedule, PI focuses on regions with high likelihood of outperforming the incumbent, LCB emphasizes conservative exploitation of low-mean and low-uncertainty regions, and the TPE highlights regions that are statistically similar to previously successful configurations. In all cases, the dashed vertical line indicates the selected candidate hyperparameter $\boldsymbol{\theta}^{\text{can}}$ for the next evaluation. 64

3.1 Search method. The receiver processes data originating either from recordings of previous operations or from simulation. At each iteration, the receiver attempts to recover the data using hyperparameter values suggested by the agent. 69
3.2 Simulated channel 1 with delay spread 110.5ms and Doppler spread 0.78Hz. The input SNR is $\rho_{\text{IN}} = 15$ dB (1.27) 72
3.3 Simulated channel 2 with delay spread 32ms and Doppler spread 1.84Hz. The input SNR is $\rho_{\text{IN}} = 15$ dB (1.27) 73
3.4 Best objective value (higher is better) versus iteration for two simulated channels: (a) Channel 1 (Figure 3.2) and (b) Channel 2 (Figure 3.3). Curves show agents using EI (in blue), UCB (in green), and TPE (in orange). The random-sampling benchmark is shown in red and the channel-estimation benchmark is shown as a black dashed line. Each curve is averaged over 30 packets. The input SNR $\rho_{\text{IN}} = 15$ dB. 75
3.5 Estimated CDF of BER for the channel-estimation benchmark (black), random sampling (gray), and the Search method using a TPE agent (orange). 76
3.6 Loch Ness measurements and objective maps with delay spread 12.7 ms and Doppler spread 0.8 Hz. The input SNR $\rho_{\text{IN}} = 15$ dB. © 2024 IEEE. . . 77
3.7 Loch Ness: performance and optimization behavior. 78

4.1 Flowchart of the online tuning process. 84

4.2 Simulated scenario with a moving receiver (yellow vehicle) and a source at 50 m depth (gray). The direct path is shown in red, surface-reflected paths in green, and bottom-reflected paths in light blue. 85
4.3 Simulated scenario. (a) Radial velocity versus range. (b) Sound-speed profile. (c) Ray tracing at receiver range 1000 m (direct path in red, surface reflections in green, bottom reflections in blue). 86
4.4 Channel generated at receiver range $RX_r = 1000$ m. 87
4.5 Channel dynamics in the simulated scenario. Panel (a) shows the Delay-Spread evolution and Panel (b) shows the corresponding in-band SNR evolution. 88
4.6 Objective-function structure for a representative packet. Left: objective map over L_A and L_B with color-coded output SNR and gray crosses indicating hallucinations. Right: power profile showing the distribution of received Power over delay. 89
4.7 Objective function over L_A and L_B for two packets observed at different ranges. The input SNR is 15 dB, and bandwidth is 10 kHz. 90
4.8 Single-agent Tracking method results on the simulated scenario. Panel (a) reports PER and panel (b) reports average output SNR. © 2025 IEEE. . . . 91
4.9 Objective function over L_A and L_B filter lengths at two ranges within the same simulated subsea environment. The input SNR is 15 dB, and bandwidth is 15 kHz. 94
4.10 Objective function over L_A, L_B, K_1 , and K_2 for two communication ranges in the same simulated subsea environment. The input SNR is 15 dB, and bandwidth is 15 kHz. 95
4.11 Effect of excessive hallucination penalization on GP surrogate fitting. Left: GP fit with reasonable hallucination value (-50 dB). Right: GP fit with one hallucination assigned an extreme penalty (-100 dB), which can induce spurious high predictions near the hallucination. 99
4.12 Overview of the *WandS* routine. **Warm-up mode** ($t \leq t_w$): random hyperparameters are evaluated on the first packet to build the initial dataset D_w. **Flywheel mode** ($t > t_w$): for each arriving packet p_t, a TV-GP with forgetting ϵ_{TV} is updated; the Wise (exploit) and Scout (explore) policies propose $\boldsymbol{\theta}_t^{\mathrm{W}}$ and $\boldsymbol{\theta}_t^{\mathrm{S}}$; both are evaluated by the receiver to obtain their scores and CRC flags. A CRC-based selection then chooses the packet's hyperparameter $\boldsymbol{\theta}_t$ from $\{\boldsymbol{\theta}_t^{\mathrm{W}}, \boldsymbol{\theta}_t^{\mathrm{S}}\}$ using their measured performance and CRC status, as specified in Algorithm 3. 101
4.13 Example PER evolution for the Wise agent, the Scout agent, and the *WandS* routine versus received packet index. 102
4.14 Simulation results for *WandS* compared to benchmarks. (a) PER comparison with benchmarks at 15 kHz; (b) PER versus bandwidth comparing *WandS* to the fixed-preamble Channel-Estimation benchmark. 103

4.15 PER comparison with single-agent schemes (UCB with $\lambda = 10$, EI, and PI). The diamond and star markers denote the mean and median PER, respectively. . . . 104
4.16 PER comparison for *WandS* using different TV-GP forgetting factors ϵ_{TV}. 104
4.17 Sensitivity of *WandS* to the exploration parameters λ_{W} and λ_{S} of the Wise and Scout agents, respectively. Brighter squares indicate lower mean PER. 105
4.18 Mean PER versus Doppler–bank filter resolution δ_{DB}. Bar heights are means across simulation runs; numeric labels report the mean PER in %. . 106
4.19 PER histograms across Doppler–bank resolutions for two policy variants. (a) *WandS* with the hallucination-aware penalty $\kappa(\boldsymbol{\theta})$ (Equation 4.24). (b) *WandS* without this penalty (hallucination-unaware). . . . 106
4.20 Power in time-delay domain. Panel (a) shows SAUWAN channel, and Panel (b) shows BCH1 channel. . . . 108
4.21 PER evolution during communication with real channel replay. Panel (a) shows PER evolution for SAUWAN channel, and Panel (b) shows PER evolution for BCH1 channel. . . . 108

5.1 Illustration of the gossip update used by X-KB-UCB at a fixed time t (here $\varepsilon = 0$, no temporal carryover). **Left:** pre-gossip posterior with mean μ_t (gray) and uncertainty band $\pm 2\sigma_t$ (light blue). At the **middle (KB):** Kriging–Believer update using the received evaluations $\{x^{(1)}, \ldots, x^{(4)}\}$: the mean μ_t is frozen and only the standard deviation is updated (blue band). **Right (X-KB-UCB):** action selection with UCB computed from the frozen μ_t and the post-gossip standard deviation, yielding the next query $\boldsymbol{\theta}_t$. 118
5.2 Impact of system parameters on the average regret over the Styblinski-Tang function. Each panel isolates a factor: (a) number of agents, (b) gossip period, and (c) average number of received tuples N_a. . . . 126
5.3 Time-varying setting: impact of system parameters on the average cumulative regret. Each panel isolates a factor: (a) number of agents, (b) gossip period, and (c) average number of received tuples per round. . . . 127
5.4 Final average augmented regret for different numbers of agents, namely $N = 1$, $N = 4$, and $N = 8$, under mismatch in the model drift parameter and the observation noise. . . . 128
5.5 X-KB-UCB performance on the sensor temperature dataset for three networks of agents $N = 1$, $N = 4$, and $N = 8$. Each panel corresponds to a different gossip period t_g, with $N_a = 0.8$. . . . 128

6.1 Graph of Mud-Pulse Telemetry where a Transmitter downhole relay measurements to a surface receiver. . . . 136
6.2 Receiver performance as a function of two PLL hyperparameters (bandwidth and damping factor). Left: drilling operation (a) at 1350 ft. Right: drilling operation (b) at a different location at 1400 ft. . . . 138

6.3 Fraction of hallucination selections per iteration, aggregated across slices, for the two objectives shown in Figure 6.2. Orange: optimization using $f^{(h)}$ in (6.1). Blue: optimization using f. 140
6.4 Average best receiver performance versus iteration for the two operations shown in Figure 6.2. Left: operation (a). Right: operation (b). 140
6.5 Post-operation recovery objective as a function of a DFE forgetting factor. Left: operation (a). Right: operation (b). 141
6.6 Time-varying reward snapshots for drift levels $\varepsilon \in \{0.001, 0.01, 0.03\}$ at times $t \in \{10, 50, 90\}$. Each row fixes ε (top to bottom: 0.001, 0.01, 0.03) and each column fixes t (left to right: 10, 50, 90). The white star marks the instantaneous maximizer $\boldsymbol{\theta}_t^\star$ on the rendering grid in each panel. 151
6.7 Time-varying experiments with $N_a = N$: final average regret as a function of the number of agents and the gossip period t_g for different drift levels ϵ. Lower is better. 152

List of Tables

1.1 Transmission settings for the packet-based link. 40

3.1 Simulated channel configurations. Doppler and delay spreads are computed using a -20 dB threshold on the Doppler power spectrum and the power-delay profile, respectively. 71

4.1 Channel Characteristics for SAUWAN and BCH1 107

6.1 Optimized Mud-Pulse Telemetry receiver versus expert values. Green blocks indicate the best performance for each recording. N_{sync} denotes is the number of frames synchronized, N_{CS} is the number of Corrected Surveys, and N_{GS} is the number of Good Surveys. 142

6.2 Average instantaneous regret $rt_$ (lower is better) for $N = 4$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**. 149

6.3 Average instantaneous regret $rt_$ (lower is better) for $N = 8$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**. 150

6.4 Average instantaneous regret $rt_$ (lower is better) for $N = 12$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**. 150

Acronyms

For readability, acronyms used throughout the manuscript are reminded below.

AUV	Autonomous Underwater Vehicle
BER	Bit Error Rate
BO	Bayesian Optimization
CRC	Cyclic Redundancy Check
DFE	Decision Feedback Equalizer
EI	Expected Improvement
GP	Gaussian Process
IPNLMS	Improved Proportionate Normalized Least Mean Squares
ISI	InterSymbol Interference
LCB	Lower Confidence Bound
PDP	Power-Delay Profile
PDS	Doppler Spectrum
PER	Packet Error Rate
PLL	Phase-Locked Loop
PI	Probability of Improvement
SNR	Signal-to-Noise Ratio
SOFAR	SOund Fixing And Ranging
SSP	Sound-Speed Profile
TPE	Tree-structured Parzen Estimator
TV-GP	Time-Varying Gaussian Process
UAC	Underwater Acoustic Communications
UCB	Upper Confidence Bound

NOTATION

The main notations used throughout the manuscript are gathered below.

General notation

x	vector
A	matrix
$(\cdot)^T$	transpose
$(\cdot)^*$	complex conjugate
$(\cdot)^\dagger$	Hermitian transpose
$\odot$	Hadamard product
$[\cdot]_{k,\ell}$	element of row k and column ℓ of a matrix
I_N	identity matrix of size $N \times N$
$\mathbf{1}_N$	all-ones vector of size $N \times 1$
$D(x)$	diagonal matrix containing the entries of vector x
$\|\cdot\|_F$	Frobenius norm
$\|\cdot\|$	Euclidean norm
$\mathrm{tr}(\cdot)$	trace
$\mathbb{E}\{\cdot\}$	expectation
$\mathcal{GP}(\mu,k)$	Gaussian distribution with mean μ and kernel k
$\Re$	real part
$\Im$	imaginary part
$\angle$	argument of a complex number
$\|\cdot\|$	magnitude
$\delta(\cdot)$	Dirac distribution
$s(t)$	transmitted waveform
$r(t)$	received signal
$h(\tau,t)$	time-varying channel impulse response
τ	propagation delay
t	time index / continuous time, depending on context
$P(\tau,t)$	instantaneous channel power in the time-delay domain
$P_{\mathrm{PDP}}(\tau)$	power-delay profile
$\widetilde{P}_{\mathrm{PDP}}(\tau)$	thresholded power-delay profile
τ_g	center of gravity of the power-delay profile
τ_{rms}	RMS delay spread

$S(\tau, \nu)$	spreading function in the delay-Doppler domain
ν	Doppler frequency
$P_{\mathrm{DS}}(\nu)$	Doppler spectrum
$\widetilde{P}_{\mathrm{DS}}(\nu)$	thresholded Doppler spectrum
ν_g	center of gravity of the Doppler spectrum
ν_{rms}	RMS Doppler spread
τ_{th}	power threshold used when computing thresholded delay/Doppler spreads
B_ℓ	bandwidth configuration indexed by ℓ
f_c	carrier frequency
c_{off}	root-raised-cosine roll-off factor
R_s	symbol rate
T_{info}	information payload duration
N_{info}	number of payload symbols
r_c	channel-coding rate
$r[n]$	baseband sampled received signal
$s_v[n]$	Doppler-scaled preamble associated with Doppler hypothesis v
$z(v, \tau)$	Doppler-bank matched-filter output
$\hat{v}$	coarse Doppler estimate
t_0	estimated packet start time
ΔV_{DB}	total Doppler search range of the Doppler bank
N_{DB}	number of Doppler hypotheses in the Doppler bank
δ_{DB}	Doppler-bank resolution
$q[k]$	signal after coarse synchronization / Doppler compensation
A	feedforward filter of the DFE
B	feedback filter of the DFE
L_A	feedforward filter length
L_B	feedback filter length
$\hat{\phi}[k]$	PLL phase estimate
K_1	proportional PLL gain
K_2	integral PLL gain
$e_\phi[k]$	PLL error signal
$z_1[k]$	feedforward-filter output
$z_2[k]$	feedback-filter output
$\hat{d}[k]$	detected symbol / feedback decision term used by the DFE
$e[k]$	DFE error signal
μ_A	feedforward-filter step size
μ_B	feedback-filter step size
β	IPNLMS channel-sparseness parameter
G_A	diagonal weighting matrix of the feedforward IPNLMS update
G_B	diagonal weighting matrix of the feedback IPNLMS update
ϵ	regularization constant in the IPNLMS weighting expression

D	dataset of evaluated hyperparameter vectors and objective values
F	surrogate model of the black-box objective
$a(\theta)$	acquisition function
k_{SE}	squared-exponential kernel
k_ν	Matérn kernel with smoothness parameter ν
ℓ_p	kernel lengthscale along input dimension p
σ_f^2	GP kernel output scale
N	evaluation budget / number of BO iterations
n	number of initial random evaluations
$f_t(\theta)$	time-indexed objective function at packet t
F_t	time-varying GP surrogate at packet t
ϵ_{TV}	forgetting factor of the time-varying surrogate
D_w	warm-up dataset for realtime adaptation
N_w	number of warm-up evaluations
t_w	warm-up duration
x_t	stacked hyperparameter vectors used by the TV-GP at time t
y_t	stacked objective values used by the TV-GP at time t
I_t	time-index vector associated with the TV-GP dataset
$k_t(\theta)$	time-weighted kernel vector
K_t	time-weighted covariance matrix
ψ	dual-agent decision routine in WandS
π_W	Wise-agent policy
π_S	Scout-agent policy
θ_t^W	Wise-agent proposal at packet t
θ_t^S	Scout-agent proposal at packet t
a_W	Wise-agent acquisition function
a_S	Scout-agent acquisition function
λ_W	exploration–exploitation parameter of the Wise agent
λ_S	exploration–exploitation parameter of the Scout agent
$\kappa(\theta)$	penalty term used to avoid hallucination-prone regions
p_t	received packet at time t

Chapter 5

$\mathcal{A}$	set of agents
a	agent index
T_{iter}	iteration horizon
$\theta_{a,t}$	arm selected by agent a at iteration t
$y_{a,t}$	noisy reward observed by agent a at iteration t
$\varsigma_{a,t}$	observation noise in the decentralized reward model
t_g	gossip period

N_g	number of gossip rounds
$P(a, v)$	probability that agent a receives a tuple from agent v at a gossip round
$\eta_{a,t}$	number of tuples received by agent a at gossip round t
N_a	expected number of tuples received by agent a per gossip round
T_η	effective number of reward evaluations available to agent a
$\theta_t^\star$	instantaneous optimal arm at time t

Introduction

Context and Motivation

Telecommunications convey information from a transmitter to a receiver through a physical propagation channel. During propagation, the transmitted waveform is altered by its interaction with the channel, sometimes to an extent that prevents direct recovery of the transmitted data. On the receiver side, signal processing algorithms are therefore required to mitigate these distortions and reconstruct the original data. In many industrial domains, the quality of the decoded data is critical because it impacts operational decisions and safety.

A common industrial use case is telemetry, where measurements must be transmitted reliably to support decision making during operations. The work presented in this manuscript was carried out at SLB (ex. Schlumberger), a global technology provider for the oil and gas and new energies sector, where telemetry systems are used to relay measurements in offshore and drilling. Within this scope, we focus on the telemetry receiver used for subsea equipment inspection and monitoring, where offshore drilling and production operations deploy high value assets in challenging subsea environments. Failures in such settings can have severe consequences for safety and the environment, as illustrated by major accidents and blowouts such as Piper Alpha, Deepwater Horizon, and Montara (Cullen, 1990 [1]; BP Deepwater Horizon, 2011 [2]; Borthwick, 2010 [3]). These accidents underline the severity of offshore failures. Although subsea inspection alone is not sufficient to prevent such disasters, it remains a relevant risk-mitigation measure.

Subsea inspection and monitoring are traditionally carried out using wired links. These links are robust and can support very high data rates, reaching up to gigabits per second in fiber-based systems (Phillips et al., 2021 [4]). However, they are expensive to deploy and maintain (Nguyen et al., 2025 [5]), and they limit the mobility of inspection platforms because of the tether and the associated surface support (Nauert et al., 2023 [6]). This motivates the use of wireless underwater communications, in particular underwater acoustic communications (Walree, 2011 [7]).

Underwater acoustic communication is the dominant solution in the subsea environment, especially for long-range links (Walree, 2011 [7]). However, sound propagation underwater is strongly constrained by the environment. As the communication range in-

creases, the optimal carrier frequency decreases, and the usable bandwidth decreases as well. For example, the optimal carrier frequency is around 20 kHz for ranges up to a few hundred meters, with a usable bandwidth of about 12 kHz. The bandwidth drops to about 7 kHz at 10 km, and to about 2.5 kHz at 50 km (Pottier, 2018 [8]). In addition, propagating sound undergoes several other forms of distortion, as discussed in Chapter 1, which makes signal decoding a sophisticated task.

Despite these limitations, underwater acoustic communications can reduce reliance on cables and enable data exchange with subsea assets, with data rates on the order of hundreds of kbit/s, in favorable conditions (Jarrot et al., 2021 [9]; Shimura et al., 2021 [10]). Beyond the energy industry, underwater acoustic communications are used in a wide range of domains, from environmental studies such as oceanography and marine biology to industrial applications such as exploration missions, military operations, and fish farming (Li et al., 2025 [11]). In this context, this manuscript focuses on improving receiver performance through signal processing, with underwater acoustic communication receivers as the main application.

Improving receiver performance requires more than selecting signal processing algorithms. The effectiveness of these algorithms depends critically on a large number of strongly coupled configuration parameters. These parameters define the operating regime of the receiver by encoding assumptions about the distortions the signal is expected to experience during propagation. For instance, when the signal is likely to interact with reflectors along its propagation path, the receiver can be configured to mitigate their impact by selecting appropriate algorithms or by tuning internal components such as filters and adaptation mechanisms. Throughout this work, we refer to these configuration parameters as ***hyperparameters***.

Definition 1 *Hyperparameters are design parameters of an algorithm that control its behavior or structure and are set externally.*

Choosing appropriate receiver hyperparameters is difficult and requires substantial expertise in both signal processing techniques and the characteristics of the channel. Channels exhibit significant variability driven by factors such as geographical location, transmitter–receiver geometry, and operating conditions. As a result, receiver algorithms are most often tuned manually by experienced practitioners. In some cases, heuristic rules based on partial measurements of the channel or operating conditions are used to assist this tuning process.

Both approaches suffer from fundamental limitations. Expert defined configurations may perform well under specific conditions but provide no guarantee of robustness when

conditions change, nor of transferability across different channels. Measurement based heuristics are constrained by the quality of the available measurements. When channel information is noisy, incomplete, or inaccurate, the resulting parameter choices may degrade performance or even render the receiver ineffective. Receiver performance is therefore highly sensitive to these hyperparameters, and the lack of reliable automation for this tuning process remains a central challenge in telecommunications.

From an operational standpoint, receiver tuning is widely regarded as time consuming and unpleasant. It often relies on trial-and-error procedures that require expert intervention and provide limited guarantees of optimality. Moreover, this tuning is usually performed offline, while changing configurations under operational conditions is costly in practice. To the best of our knowledge, no general and reliable automation of this task has been available prior to this work. This work addresses this gap by developing automated methods that replace manual tuning and ad hoc heuristics with principled optimization procedures.

Our approaches rely on Bayesian optimization, a family of derivative free and sample efficient optimization methods that has demonstrated strong empirical performance in a variety of domains, including experimental design and model selection in machine learning. In this work, Bayesian optimization is used to automate the fine tuning of telecommunication receivers.

The developed approach goes beyond a direct application of Bayesian optimization to receiver hyperparameter tuning. The main reason is that acoustic telemetry is special because the communication channel is the subsea environment, which is highly dynamic and heterogeneous, and receiver configurations must adapt in real time. Addressing this requirement necessitates extensions of standard Bayesian optimization techniques. To this end, we develop multi-agent optimization frameworks in which several agents coordinate, either through a centralized decision mechanism or in a decentralized manner, to adapt receiver configurations online during communications.

Thesis structure and contributions

The dissertation is organized into five chapters.

Chapter 1 provides an overview of acoustic telemetry, with a focus on the underlying technology of Underwater Acoustic Communication, its specificities compared to other telecommunication techniques, and the receiver architecture considered in this work. It also introduces a main contribution of the dissertation, namely a first formalization, to the best of our knowledge, of receiver fine-tuning as an optimization problem defined through an unknown objective function. Even though this function outputs a known and

measurable performance metric, its relation to the input hyperparameters is not known a priori.

Chapter 2 reviews existing tools for optimizing complex functions of this type, with a particular emphasis on Bayesian optimization as the main methodological framework used throughout this dissertation.

The subsequent chapters present original automated frameworks for receiver fine-tuning that address the optimization problem introduced in Chapter 1. In particular, Chapter 3 considers receiver fine-tuning using recorded data or simulated scenarios, corresponding to an offline optimization setting. The developed method is referred to as *Search Method*. The content of this chapter is closely related to the following publication:

C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Self-Configurable Receiver for Underwater Acoustic Communications Using Bayesian Optimization," Seventh Underwater Communications and Networking Conference (UComms), 2024.

Chapter 4 addresses real-time receiver fine-tuning during operation. The aim is to develop the *Tracking method*, a centralized multi-agent technique. We first introduce a simplified single-agent version, and then extend it to the centralized multi-agent setting. The developments presented in this chapter are strongly inspired by the following works:

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Sequential Optimization of Decision Feedback Equalizer Hyperparameters in Mobile Acoustic Channels," OCEANS 2025, Brest.*

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Self-Configurable Receiver for Single-Carrier Underwater Acoustic Communications via Bayesian Optimization," Journal of Oceanic Engineering, 2026, accepted for publication.*

Finally, Chapter 5 extends the centralized online multi-agent framework developed in the previous chapter to a decentralized setting. This chapter goes beyond UAC and opens a broader perspective. It places particular emphasis on theoretical guarantees, with the analysis conducted in a bandit framework rather than in the specific UAC setting. This chapter rely on : *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "X-KB-UCB: Decentralized Cross Kriging-Believer UCB for Static and Time-Varying GP Bandits", TMLR 2026, under review.*

An overview of the manuscript structure as well as the main contributions can be viewed in Figure 1.

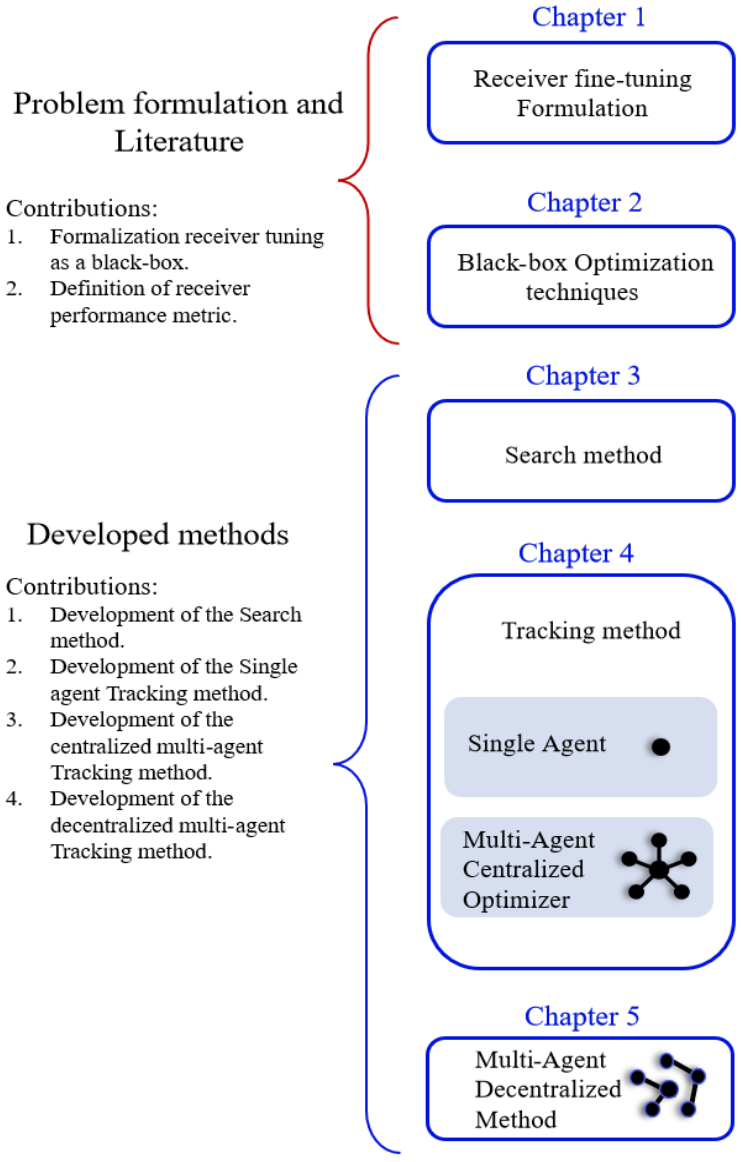


Figure 1 – Manuscript structure and contributions.

The list of all peer-reviewed publications and patent applications during this PhD project are listed bellow :

International conferences :

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Self-Configurable Receiver for Underwater Acoustic Communications Using Bayesian Optimization," Seventh Underwater Communications and Networking Conference (UComms), 2024.*

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Sequential Optimization of Decision Feedback Equalizer Hyperparameters in Mobile Acoustic Channels," IEEE OCEANS 2025, Brest.*

Journal papers:

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "Self-Configurable Receiver for Single-Carrier Underwater Acoustic Communications via Bayesian Optimization," IEEE Journal of Oceanic Engineering, 2026, accepted for publication.*

— *C. M. Cheikh Melainine, F.-X. Socheleau, and A. Jarrot, "X-KB-UCB: Decentralized Cross Kriging-Believer UCB for Static and Time-Varying GP Bandits," Transactions on Machine Learning Research 2026, under review.*

Patents:

— *C. M. Cheikh Melainine, A. Jarrot, F.-X. Socheleau, and A. Croux, "Bayesian Techniques of Hyperparameters for Telecommunication Systems," PCT Patent App. No. PCT/US2025/038681, filed July 22, 2025.*

— *C. M. Cheikh Melainine, A. Jarrot, F.-X. Socheleau, and A. Croux, "Bayesian Techniques of Hyperparameter Selection for Telecommunication Systems," U.S. Patent App. No. 19/452,481, filed January 19, 2026*

Chapter 1

Receiver fine-tuning problem

1.1 Introduction

Underwater Acoustic Communications (UAC) involve strong and time-varying channel effects, and receiver performance depends on the combined effect of propagation conditions, hardware constraints, and signal processing hyperparameters. These hyperparameters, introduced in Section 1.3, have a direct impact on the quality of the decoded data and, more generally, on receiver performance.

Quantifying receiver performance is itself challenging, as discussed in Section 1.4. In particular, it is non-trivial to predict performance from the chosen hyperparameters, and the impact of these parameters is only partially characterized in the literature. This motivates viewing receiver fine-tuning as the optimization of a ***black-box*** function (Garnett, 2023a [12]).

Definition 2 *A black-box is a function*
- *for which no analytic form is available*
- *whose gradient cannot be computed or estimated reliably*
- *and whose evaluation is expensive*

The inputs of the receiver black-box considered in this work are referred to as hyperparameters, since they correspond to configuration parameters of the signal processing algorithms implemented at the receiver.

Recent work adopts the black-box perspective by treating the receiver chain as an opaque system and proposing data-driven approaches to improve decoding quality, including adaptive receivers and learning-based components, to achieve reliable subsea communications (Zuberi et al., 2023 [13]; Zhang et al., 2022a [14]; Zhang et al., 2023 [15]). However, to the best of our knowledge, the specific problem of automating receiver fine-tuning through a principled optimization formulation remains insufficiently characterized

in the literature. Existing works are usually tailored to only a subset of hyperparameters and perform adaptation at the symbol rate. This is the case, for instance, for variable-length equalizers, where the filter lengths are increased or decreased based on the output mean squared error (MSE) (Riera-Palou et al., 2000a [16]; Kar et al., 2011 [17]), or in more recent UAC applications (Zhou et al., 2024 [18]), where the adaptation relies on estimates of the noise variance and remaining distortion after equalization. Other techniques aim to adapt update parameters, such as step sizes and gain parameters, within a transmitted packet as symbols are decoded (Liu et al., 2024 [19]; Mathews et al., 2002 [20]), using the instantaneous error.

Although these approaches can improve over fixed-configurations, they remain heuristic and are restricted to a limited set of parameters. This is a limitation because receiver configuration are often coupled, as shown in this dissertation, and optimizing only one subset may lead to suboptimal overall configurations. More importantly, such methods are often tied to specific receiver structures and adaptation rules, which limits their portability across architectures. They also rely on local instantaneous indicators, such as error, MSE, or residual distortion, which do not necessarily reflect the final end-to-end performance of the receiver. The focus of this manuscript is to automate receiver fine-tuning by optimizing the configuration parameters, or hyperparameters, of receiver-side signal processing algorithms.

This chapter is organized as follows. Section 1.2 provides an overview of Underwater Acoustic Communications, covering representative use cases and the main challenges of this domain. Section 1.3 then introduces the signal processing techniques within the receiver architecture considered throughout this manuscript and identifies the corresponding hyperparameters to be tuned. Finally, Section 1.4 defines the receiver black-box that is optimized in order to automate the hyperparameter tuning of the considered receiver algorithms.

1.2 Underwater acoustic communications

1.2.1 General overview

In underwater environments, the propagation medium is seawater, which constrains the choice of physical carriers. Radio and optical communications suffer from severe attenuation and scattering in water, restricting their effective range to specialized or short-range conditions. In contrast, acoustic waves can propagate over distances reaching up to 700 km (Jiang et al., 2022 [21]; Song et al., 2011 [22]), with detections reported up to 3000 km (Roux et al., 2011 [23]). Acoustic signaling therefore remains the dominant technology for long-range underwater links, despite limited bandwidth, high latency, and sensitivity to

reflections and time variations (Theocharidis et al., 2025 [24]).

Underwater Acoustic Communications has stimulated human curiosity for a long time. Roughly 2400 years ago Aristotle, in one of the oldest written records, noted that sound could be heard in water and not only in air. One millennium later, Leonardo da Vinci wrote in 1490: "If you cause your ship to stop and place the head of a long tube in the water and place the other extremity to your ear, you will hear ships at great distances". This intuition was later tested in the 18th and 19th century with increasingly controlled experiments. The goal was not yet communications, but metrology: measuring the speed of sound in water using bells, gunpowder flashes, hunting horns, or human voices, with the human ear as the receiver. A landmark result was obtained in September 1826 when the Swiss physicist J. D. Colladon and the French mathematician J. K. F. Sturm conducted one of the first widely known measurements of sound speed in water on Lake Geneva at a temperature of 8°C. A bell suspended from a boat was struck as an acoustic source, and a flash of light was generated by igniting gunpowder to provide a timing reference. The receiver was a trumpet-like device with one end submerged and the other at the observer's ear. Their reported speed, about 1435 m/s, is close to modern values for those conditions. Subsequent decades combined improved transducers, better receivers, and stronger physical understanding, enabling the transition from qualitative observations to engineered underwater acoustic systems (Bjørnø et al., 2017 [25]).

1.2.2 UAC channels

In UAC, the signal consisting in acoustic waves propagates through the subsea environment, which is our channel. UAC faces significant challenges due to rapid channel variability and diverse environmental conditions, and sometimes described as operating in nature's most unforgiving communication medium for wireless communications (Walree, 2011 [7]). UAC supports oceanographic data collection, environmental monitoring, subsea infrastructure inspection, defense and surveillance operations, and networking of fixed or mobile underwater platforms such as autonomous underwater vehicles and sensor networks (Theocharidis et al., 2025 [24]). From a system perspective, the subsea environment induces three dominant classes of distortions on an acoustic waveform propagating through it: sound-speed variability and refraction, multipath propagation, Doppler effects, noise and interference.

The first mechanism is sound-speed variability. In the mid-1920s, the U.S. Coast and Geodetic Survey and the U.S. Naval Research Laboratory reported an operational observation: sonar performance could degrade from morning to afternoon, later known as the "afternoon effect". A convincing explanation was established in 1937 by acousticians and oceanographers associated with the U.S. Submarine Signal Company and Columbus Iselin: daytime surface heating modifies the vertical sound-speed structure, which changes refraction and can redirect energy away from the intended region of reception (Bjørnø et

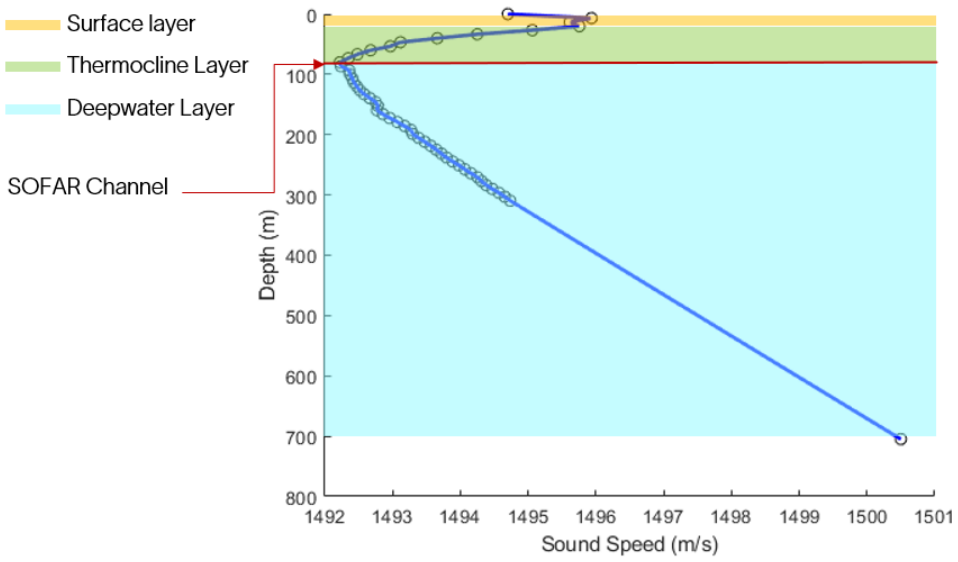


Figure 1.1 – Sound Speed Profile example in warm deep waters.

al., 2017 [25]). The governing physical rule is that acoustic energy refracts toward lower sound-speed layers. The engineering consequence is that propagation geometry, and therefore the set of dominant paths, changes over time.

Sound speed is heterogeneous because it depends primarily on salinity, temperature, and pressure (depth). This differs from classical RF communication, where electromagnetic waves propagate in air at essentially constant speed within a given band. A Sound-Speed Profile (SSP) illustrating this variability is shown in Figure 1.1. This SSP was measured in the North Atlantic in spring. A typical warm-water SSP exhibits: a near-surface layer influenced by temperature and salinity, a thermocline where sound speed and temperature decreases with depth as temperature drops, and a deep-water region where sound speed increases again under rising pressure. Within such profiles, a minimum may appear. This minimum defines the SOFAR (SOund Fixing And Ranging) channel: upper layers refract sound downward, while lower layers refract sound upward toward the minimum. This can trap acoustic energy and enable propagation over hundreds or even thousands of kilometers (Worzel et al., 1948 [26]). The SSP therefore shapes the feasible sound paths.

Aside from spatial heterogeneity in the sound speed within the channel, the propagating acoustic wave is also affected by multipath propagation, Doppler effects, and noise. These sources of distortion originate from multiple physical mechanisms. The subsea environment contains numerous reflectors and scatterers, including the sea surface, the seabed and its topography, subsea equipment or vehicles, and marine fauna (Walree, 2011 [7]). As a consequence, the receiver observes multiple replicas of the transmitted waveform with

different amplitudes and arrival times, giving rise to multipath propagation. Multipath propagation impacts the receiver's ability to reliably distinguish the transmitted data. Binary transmitted information is mapped to symbols through a modulation scheme, and the channel creates multiple delayed replicas of these symbols that arrive through different paths. Since the path delays can be comparable to the symbol duration, delayed replicates from previously transmitted symbols overlaps with the current symbol at the receiver. This phenomenon is known as InterSymbol Interference (ISI). ISI is typically handled through equalization, which is a core signal processing component in this dissertation, with major hyperparameters listed in Section 1.3. Multipath also creates deep fades in the received signal strength due to destructive interference between paths, leading to strong frequency selectivity. In practice, this often motivates multicarrier modulations (Bourré et al., 2013 [27]; Socheleau et al., 2012 [28]; Bocus et al., 2020 [29]). For simplicity, deep fades are not precisely handled in this text and are left for future work.

Doppler effects can be understood from (1.1), which shows that two types of time variation may occur in the channel: the variation of the path delays $\tau_l(t)$ and the variation of the path amplitudes $\alpha_l(t)$.

The variation of the path delays $\tau_l(t)$ produces compressions and dilations of the received waveform. This corresponds to the Doppler scale, that is, a time-scaling effect. It arises when the propagation path lengths vary in time, for instance because of transmitter or receiver motion, or because moving sea-surface reflection points modify the geometry of the signal propagation paths.

The variation of the path amplitudes $\alpha_l(t)$ produces fluctuations of the received energy over Doppler frequencies. This corresponds to the Doppler spread. It reflects the time variability of the propagation conditions, for example due to sea-surface scattering or other moving boundaries.

Both effects are significant in underwater acoustics because the speed of sound in water is low, about 1500 m/s. They distort time and frequency synchronization: symbol boundaries drift over time, the effective sampling instants no longer align with the transmitted symbol rhythm, and carrier offsets together with time scaling induce phase rotation. This necessitates receiver-side synchronization and compensation techniques that estimate Doppler and correct it through resampling and tracking loops. These techniques are further detailed in Section 1.3.

All these phenomena are captured by the channel impulse response expressed as

$$h(\tau, t) = \sum_{l=1}^{L} \alpha_l(t)\delta(\tau - \tau_l(t))\, e^{-i2\pi f_c \tau_l(t)}, \tag{1.1}$$

where L is the number of propagation paths, $\alpha_l(t)$ and $\tau_l(t)$ denote respectively the time-varying amplitude and delay of the l-th path, and f_c is the carrier frequency (Walree

et al., 2013 [30]).

In practice, path amplitudes also undergo frequency-dependent attenuation, which we model separately through Thorp attenuation (Thorp, 1967 [31]). Under the usual short-packet assumption, the delays are assumed to vary linearly in time, so that $\tau_l(t) = \tau_{0,l} + \epsilon_l t$. The channel impulse response then becomes

$$h(\tau, t) = \sum_{l=1}^{L} \alpha_l(t)\, \delta(\tau - \tau_{0,l} - \epsilon_l t)\, e^{-i2\pi f_c(\tau_{0,l}+\epsilon_l t)}. \tag{1.2}$$

The factor ϵ_l is often expressed as $\epsilon_l = v_l/c_0$, where v_l is the relative velocity between the transmitter and the receiver (as seen from the l-th path) and c_0 is the average speed of the acoustic wave along that path. In addition to these effects, Ambient noise also affects the received signal and originates from various sources, including marine life, ship traffic, rainfall, and breaking surface waves (Walree, 2011 [7]).

Characterizing the underwater acoustic channel relies on a set of proxy metrics, which are introduced below. A standard assumption in underwater acoustic communications is to model the channel as a linear time-varying system. Denoting by $s(t)$ the transmitted waveform and by $r(t)$ the received signal, the relationship between the input and output of the channel can be expressed as

$$r(t) = \int_{-\infty}^{\infty} h(\tau, t) s(t - \tau) d\tau, \tag{1.3}$$

where $h(\tau, t)$ is the time-varying impulse response and τ denotes the propagation delay.

To represent multipath propagation, one can examine the instantaneous channel power in the time-delay domain:

$$P(\tau, t) = \left| h(\tau, t) \right|^2. \tag{1.4}$$

An example of this representation is shown in Figure 1.2, which corresponds to a channel sounded in freshwater at Loch Ness, Scotland, by SLB in 2023 (Cheikh Melainine et al., 2024 [32]), with a central frequency of 35 kHz and a bandwidth of 5 kHz. The top-left plot displays the power of each propagation path (in dB) as a function of delay. This representation highlights both the relative delays and the power distribution of the individual paths. A measure of how much the channel is spread in delay is obtained from the power-delay profile (PDP), shown in the lower-left plot of Figure 1.2. The PDP represents the received power as a function of delay and can be used to estimate the effective delay spread between the most energetic paths, which are typically responsible for the strongest ISI. Using the instantaneous channel power $P(\tau, t) = |h(\tau, t)|^2$, the PDP is defined as the time-integrated power at each delay (or, equivalently, time-averaged up to the scaling factor $1/T_{\text{obs}}$), i.e.

$$P_{\text{PDP}}(\tau) = \int_{t_0}^{t_0+T_{\text{obs}}} P(\tau, t) dt \tag{1.5}$$

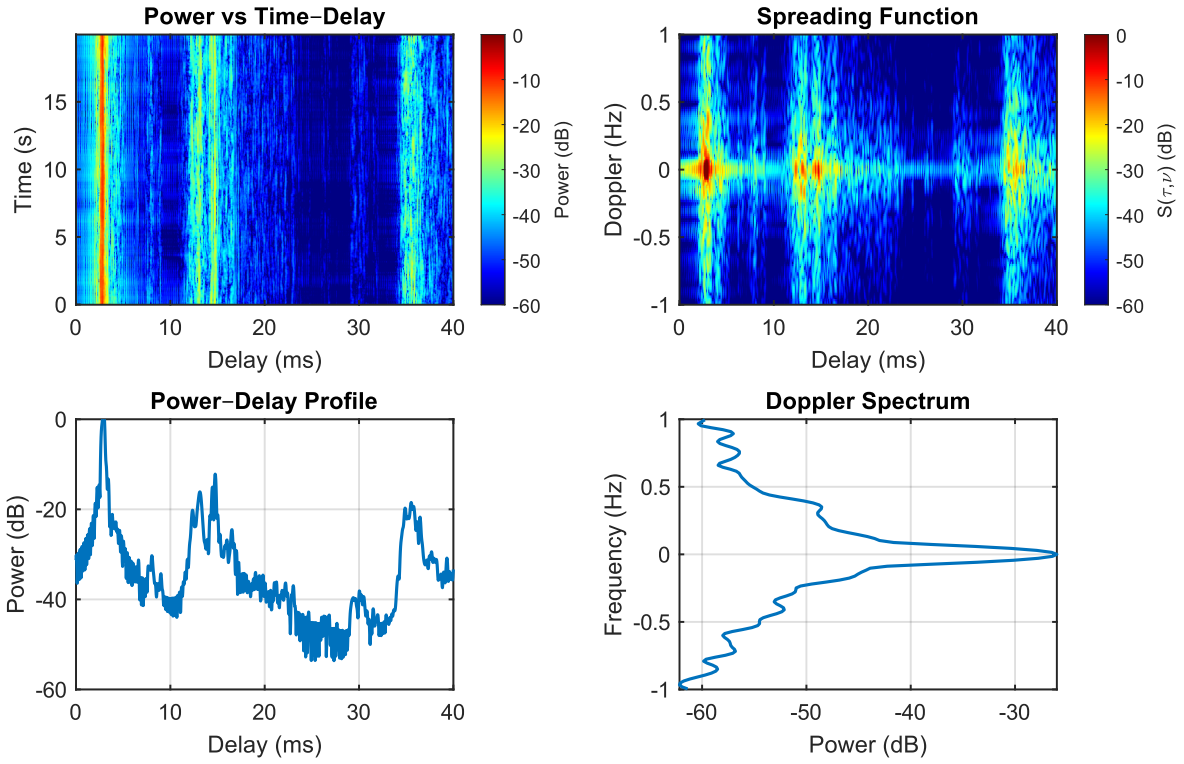


Figure 1.2 – Characterization of the Loch Ness channel, Scotland, 2023. Top row: instantaneous channel power in the time-delay domain (left) and spreading function (right). Bottom row: power-delay profile (left) and Doppler spectrum (right). Doppler is compensated in these figures.

for an observation window of duration T_{obs}. In this example, multiple paths are concentrated around a delay of approximately 3 ms, with another cluster around 13 ms.

In practice, both delay- and Doppler-spread metrics are sensitive to noise and low-power tails. Therefore, in this manuscript, RMS spreads are computed after applying a power threshold τ_{th} (in dB, relative to the peak of the corresponding profile/spectrum). Specifically, we define the thresholded PDP

$$\widetilde{P}_{\text{PDP}}(\tau) = P_{\text{PDP}}(\tau)\mathbf{1}\{P_{\text{PDP}}(\tau) \geq P_{\text{PDP,max}} 10^{-\tau_{\text{th}}/10}\}, \tag{1.6}$$

where $P_{\text{PDP,max}} = \max_\tau P_{\text{PDP}}(\tau)$ and $\mathbf{1}\{\cdot\}$ is the indicator function. The delay spread is then quantified using the RMS definition:

$$\tau_{\text{rms}} = \left(\frac{\int_{-\infty}^{\infty} \widetilde{P}_{\text{PDP}}(\tau)\big(\tau - \tau_g\big)^2 d\tau}{\int_{-\infty}^{\infty} \widetilde{P}_{\text{PDP}}(\tau) d\tau} \right)^{\frac{1}{2}}, \tag{1.7}$$

where τ_g denotes the center of gravity of the PDP,

$$\tau_g = \frac{\int_{-\infty}^{\infty} \tau \widetilde{P}_{\mathrm{PDP}}(\tau) d\tau}{\int_{-\infty}^{\infty} \widetilde{P}_{\mathrm{PDP}}(\tau) d\tau}. \tag{1.8}$$

In the Loch Ness example, $\tau_{\mathrm{rms}} = 12.7$ ms for $\tau_{\mathrm{th}} = 20$ dB.

To further characterize channel distortions in the delay-Doppler domain with continuous variables, consider the time-varying impulse response $h(\tau, t)$, where $\tau \in \mathbb{R}$ denotes delay and $t \in \mathbb{R}$ denotes time. The spreading function is defined as the Fourier transform of $h(\tau, t)$ with respect to t,

$$S(\tau, \nu) = \left| \int_{-\infty}^{\infty} h(\tau, t) \exp\left(-j2\pi\nu t\right) \, dt \right|^2, \tag{1.9}$$

where $\nu \in \mathbb{R}$ is the Doppler frequency. The spreading function describes how signal energy is distributed jointly in delay and Doppler. Figure 1.2's top right plot presents an estimate of $S(\tau, \nu)$ for the Loch Ness channel and highlights the spreading around the multipath clusters. Increased Doppler spreading indicates stronger temporal variability of the channel and results in more challenging signal recovery.

The Doppler spectrum is defined as the delay-marginal energy density

$$P_{\mathrm{DS}}(\nu) = \int_{-\infty}^{\infty} S(\tau, \nu) \, d\tau. \tag{1.10}$$

Similarly to the delay-domain treatment, we apply a threshold τ_{th} and define

$$\widetilde{P}_{\mathrm{DS}}(\nu) = P_{\mathrm{DS}}(\nu) \, \mathbf{1}\left\{ P_{\mathrm{DS}}(\nu) \geq P_{\mathrm{DS,max}} 10^{-\tau_{\mathrm{th}}/10} \right\}, \tag{1.11}$$

where $P_{\mathrm{DS,max}} = \max_\nu P_{\mathrm{DS}}(\nu)$. The RMS Doppler spread is then defined as

$$\nu_{\mathrm{rms}} = \left(\frac{\int_{-\infty}^{\infty} \widetilde{P}_{\mathrm{DS}}(\nu) \left(\nu - \nu_g\right)^2 d\nu}{\int_{-\infty}^{\infty} \widetilde{P}_{\mathrm{DS}}(\nu) \, d\nu} \right)^{\frac{1}{2}}, \tag{1.12}$$

where ν_g denotes the center of gravity of the Doppler spectrum,

$$\nu_g = \frac{\int_{-\infty}^{\infty} \nu \, \widetilde{P}_{\mathrm{DS}}(\nu) \, d\nu}{\int_{-\infty}^{\infty} \widetilde{P}_{\mathrm{DS}}(\nu) \, d\nu}. \tag{1.13}$$

In the considered example, this measure yields $\nu_{\mathrm{rms}} = 0.8\,\mathrm{Hz}$ for $\tau_{\mathrm{th}} = 20\,\mathrm{dB}$.

In practice, the impulse response $h(\tau, t)$ exhibits strong variability and diversity due to numerous factors related to geography, weather conditions, operational activity, and

local ecosystems. These factors are only partially predictable, and data acquisition in such environments is costly.

These Doppler and Multipath distortions determine the processing requirements of a UAC receiver. A practical receiver therefore combines synchronization and Doppler tracking with equalization to mitigate multipath effects. The next section introduces these receiver mechanisms and the associated parameter tuning problem.

Another challenge is that the subsea propagation channel is time-varying across multiple time scales. To separate the main mechanisms, we assume in the following that both transmitter and receiver are fixed and we group the variability into three categories.

First, *slow variations* evolve over days to seasons and primarily change the sound-speed profile and large-scale boundary conditions. Typical drivers include seasonal changes in temperature and stratification, and other slowly varying oceanographic conditions (Socheleau, 2019 [33]; Walree, 2011 [7]). These changes are usually slow compared to the duration of a communication packet and, throughout this manuscript, they are therefore neglected when analyzing the physical-layer receiver, so the channel is treated as quasi-static at that scale.

Second, *short-term variations* occur on time scales comparable to a link operation and can modify the impulse response during reception. Public datasets often provide channel recordings over tens of seconds to about one minute (Walree et al., 2017 [34]; Underwater Acoustic Channels Group, 2026 [35]), which implicitly assumes that this window is a useful time scale to represent a channel for a given location and operating condition. To quantify short-term variability, a temporal coherence can be defined from the normalized correlation between a reference channel snapshot and later snapshots, with a threshold such as 0.5 to mark significant change (Badiey et al., 2000 [36]). Short-term fluctuations are driven by, among others,

- tidal evolution changes over minutes to hours (Walree, 2013 [37]; Tomasi et al., 2011 [38]),
- internal waves that perturb the water-column sound speed on time scales of minutes (Walree, 2013 [37]),
- ocean swell, that is long-period surface gravity waves generated by distant winds, which modulate surface interactions on time scales on the order of 10 seconds (Walree, 2013 [37]),
- wind-driven surface roughness, which can reduce coherence on sub-second time scales, with decorrelation times around 0.8–1 s for calm conditions (about 2 m/s) and around 0.3–0.5 s for rougher sea states (about 12 m/s) (Badiey et al., 2000 [36]).

These short-term changes are the ones that most directly challenge receiver synchronization, equalization, and tracking during communication and they motivate adaptive receiver configurations and hyperparameter tuning.

Finally, *fast fluctuations* occur at time scales that the considered signal processing loops cannot track reliably and are treated as effective noise. Examples include rapid scattering from small-scale surface roughness, intermittent bubble activity under breaking waves, and other high-rate stochastic disturbances of the received phase and amplitude (Walree, 2011 [7]).

If transmitter or receiver motion is present, an additional source of variability appears through kinematic Doppler effects, which induce time scaling and drift of multipath delays. In that case, the dominant rate of change is jointly determined by platform motion and surface-induced path fluctuations, and the channel coherence typically decreases compared to the fixed-geometry case (Worcester et al., 1999 [39]).

This variability matters because receiver hyperparameters reflect receiver-side assumptions about the channel, for example the channel delay spread through the equalizer filter length, and the Doppler spread through the tracking rates. As a result, receiver performance improves when the chosen hyperparameters are well matched to the channel conditions. When the channel changes over the short-term scale, the hyperparameter region that best matches it changes as well, so the optimal hyperparameters drift with the same underlying physical variability.

1.3 Signal Processing Hyperparameters

UAC use a range of modulation and waveform designs (Haziq et al., 2025 [40]), including single-carrier constellations such as Binary Phase-shift Keying (BPSK), Quadrature Phase-Shift Keying (QPSK), and M-ary Quadrature Amplitude Modulation (M-QAM), as well as multicarrier or more recent designs such as Orthogonal Frequency Division Multiplexing (OFDM) and Orthogonal Time Frequency Space Modulation (OTFS), among others. The choice depends on the target trade-off between robustness, receiver complexity, and spectral efficiency. In channels with strong multipath and Doppler, low-order constellations are often preferred because they reduce sensitivity to channel estimation errors and Doppler-induced time variation.

In this study, we consider a Single-Input Single-Output (SISO) packet-based link, meaning one transmitter and one receiver operating on finite-length packets. Each packet is modulated using QPSK, that is a four-point phase constellation. We use QPSK because it is a standard baseline in UAC (Stojanovic et al., 2002 [41]; Zia et al., 2021 [42]; Pranitha et al., 2020 [43]; Hsieh et al., 2023 [44]) and it offers a practical compromise between robustness and complexity: compared to higher-order constellations it is less sensitive to channel estimation errors and residual Doppler.

The packet format used in this work is as follows : Packets begin with a 511-symbol

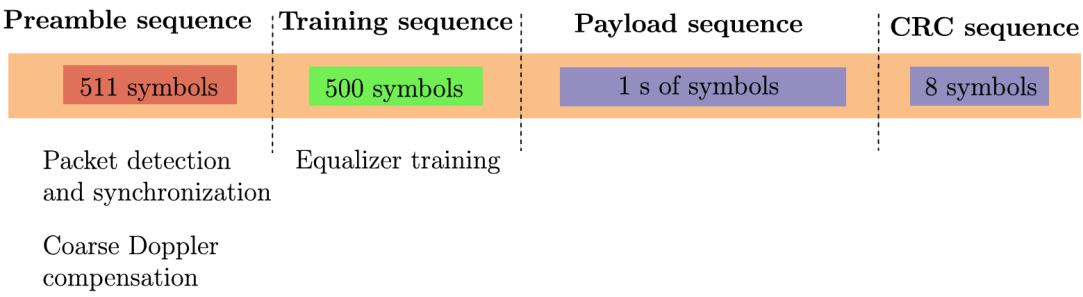


Figure 1.3 – Packet structure, with a preamble for detection and synchronization, followed by the training sequence used by the equalizer, the payload, and the CRC sequence.

preamble, as commonly used in the literature (Maia et al., 2018 [45]; Unal et al., 2024 [46]), followed by a training sequence and payload symbols carrying information bits. The carrier frequency is fixed to $f_c = 20$ kHz across all configurations. To enhance reliability, convolutional error-correcting coding at rate $r_c = \frac{1}{2}$ and a 16 bit Cyclic Redundancy Check (CRC) are applied. A root-raised-cosine matched filter is used with roll-off factor $c_{\text{off}} = \frac{1}{3}$. The packet structure is summarized in Figure 1.3.

Throughout this study, four bandwidth configurations are considered:

$$B_1 = 4.5\,\text{kHz}, \quad B_2 = 5\,\text{kHz}, \quad B_3 = 10\,\text{kHz}, \quad B_4 = 15\,\text{kHz}. \tag{1.14}$$

Several bandwidth configurations are considered, with values representative of UAC practice (Walree et al., 2017 [34]). Increasing the bandwidth increases the symbol rate and, for a fixed physical delay spread, increases the apparent channel length in samples. This yields longer effective channels for the receiver and makes equalization and synchronization more challenging. In our experiments, we increase the bandwidth to stress the receiver and evaluate the fine-tuning methods under progressively harder conditions.

With roll-off c_{off}, the symbol rate is $R_s = \frac{B_\ell}{1+c_{\text{off}}}$. Fixing the information payload duration to $T_{\text{info}} = 1$s, the number of payload symbols is

$$N_{\text{info}} = R_s\, T_{\text{info}} = \frac{B_\ell}{1 + c_{\text{off}}}\, T_{\text{info}}. \tag{1.15}$$

This yields packet durations of 1.3 s, 1.27 s, 1.13 s, and 1.09 s for B_1, B_2, B_3, and B_4, respectively. The main transmission settings are summarized in Table 1.1.

Now that the transmission settings have been specified, we describe the receiver architecture to identify the hyperparameters to be tuned. Classical UAC receivers involve many such parameters, but the proofs of concept in this manuscript focus on a restricted yet representative subset associated with the Decision Feedback Equalizer (DFE) and the Phase-Locked Loop (PLL) (see (1.23); (1.24), and (1.19).

Table 1.1 – Transmission settings for the packet-based link.

Parameter	**Value**
Link type	SISO, packet-based
Packet duration	1.3 s, 1.27 s, 1.13 s, 1.09 s
Modulation	QPSK
Preamble length	511 symbols
Training length	500 symbols
Bandwidth set	$B_1 = 4.5$ kHz, $B_2 = 5$ kHz, $B_3 = 10$ kHz, $B_4 = 15$ kHz
Carrier frequency	$f_c = 20$ kHz
Channel coding	Convolutional code, $r_c = \frac{1}{2}$
Error detection	CRC
Pulse Shaping	RRC, Roll-off $c_{\text{off}} = \frac{1}{3}$ height

Figure 1.4 illustrates the receiver architecture considered in this work and highlights the processing blocks in which these hyperparameters intervene.

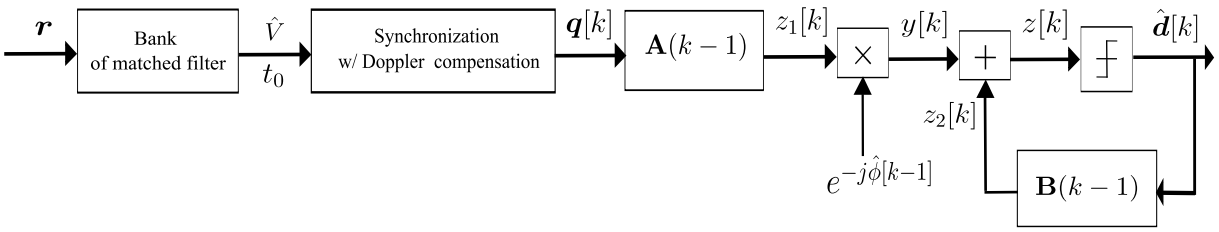
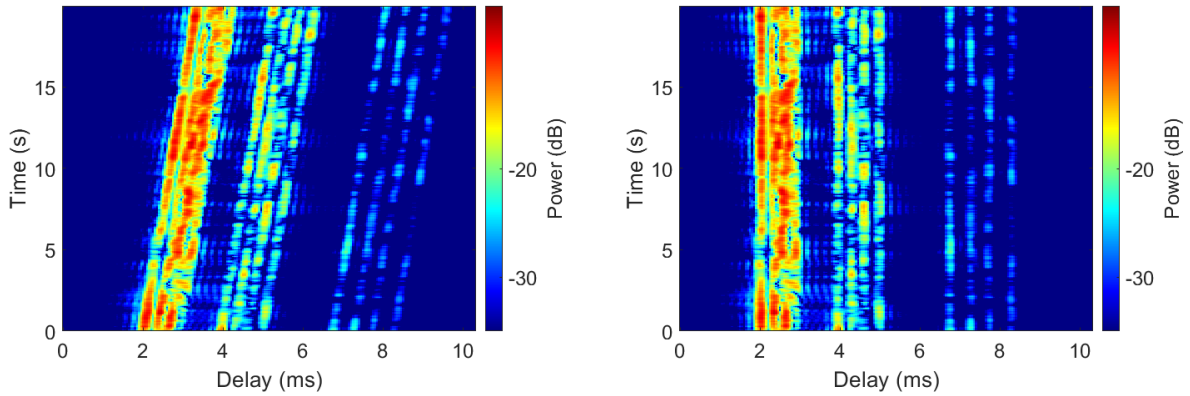


Figure 1.4 – Schematic of the receiver. A Doppler-bank matched filter produces coarse Doppler and timing estimates, which are used by the synchronization stage for timing alignment and Doppler compensation. The Doppler-corrected signal is then equalized by a decision-feedback equalizer, and subsequently processed by a PLL to correct residual frequency offsets prior to symbol detection.

The baseband sampled received signal is denoted by $\mathbf{r} = [r[1], r[2], \ldots, r[L]]$, where L is the packet length. The signal is first processed by a Doppler-bank matched filter (Johnson et al., 1997a [47]) built from a known preamble $s_p(t)$. The bank evaluates a grid of Doppler hypotheses $\mathcal{V}$ and for each $v \in \mathcal{V}$ computes the matched-filter output

$$z(v, \tau) = \sum_{n \in [1,L]} r[n]\, e^{j2\pi \frac{v}{c} f_c \frac{n}{F_s}}\, s_p^*\!\left((n/F_s - \tau)\left(1 - \frac{v}{c}\right)\right), \tag{1.16}$$

where τ is the packet start-time hypothesis expressed in samples, f_c is the carrier frequency, F_s is the sampling frequency, and c is the propagation speed. The factor $\left(1 - \frac{v}{c}\right)$ explicitly accounts for the Doppler-induced time compression or dilation of the preamble, while the exponential term accounts for the associated carrier-frequency shift. The coarse



(a) Uncompensated Doppler. (b) Compensated Doppler.

Figure 1.5 – Channel power (dB) in the time-delay domain. Left: channel before Doppler compensation. Right: channe after Doppler compensation using the Doppler-bank matched estimate.

packet start time and Doppler estimate are then obtained as

$$(\hat{v}, t_0) = \arg\max_{v \in \mathcal{V}, \tau} |z(v, \tau)|. \tag{1.17}$$

These estimates are used to perform coarse synchronization: the received packet is aligned using t_0 and resampled to undo the Doppler scale implied by $\hat{v}$, which compensates the dominant Doppler-induced time dilation or compression before subsequent processing.

The resolution of the Doppler-bank matched filter, denoted δ_{DB}, is defined as the spacing between adjacent Doppler hypotheses in the filter bank. If the Doppler search spans a total range ΔV_{DB} using N_{DB} uniformly spaced hypotheses, the resolution is given by

$$\delta_{DB} = \frac{\Delta V_{DB}}{N_{DB} - 1}. \tag{1.18}$$

A smaller value of δ_{DB} enables more accurate Doppler estimation and compensation but increases computational complexity due to the larger number of filter branches. The choice of δ_{DB} therefore reflects a trade-off between estimation accuracy and receiver complexity. An illustration is provided in Figure 1.5. For this example, we use a Doppler-bank matched $\mathcal{V} = [-5, 5]$ m/s with resolution $\delta_{DB} = 0.1$ m/s and a simulated Doppler corresponding to 0.1 m/s, which lies on the grid and yields an essentially exact coarse compensation. The plots show the channel power (dB) in the time-delay domain, as perceived by the receiver, before compensation (left) and after Doppler compensation by resampling using the estimated Doppler scale (right).

After Doppler compensation and synchronization, the resulting signal vector $\mathbf{q}[k]$ is passed to a decision feedback equalizer (DFE). The DFE consists of a feedforward filter

A of length L_A and a feedback filter **B** of length L_B. The feedforward filter mitigates precursor inter-symbol interference, while the feedback filter compensates for postcursor interference using past symbol decisions (Stojanovic et al., 1993 [48]).

Often the output of the Doppler-bank still exhibits a residual frequency offset whose magnitude depends on the chosen Doppler resolution. This residual offset is corrected using a PLL coupled to the DFE (Cannelli et al., 2013 [49]; Stojanovic et al., 1993 [48]; Eynard et al., 2008 [50]; Oh et al., 1996 [51]; Cuji et al., 2024a [52]; Johnson et al., 1997b [53]). The PLL phase estimate $\hat{\phi}$ evolves according to

$$\hat{\phi}[k] = \hat{\phi}[k-1] + K_1 e_\phi[k] + K_2 \sum_{n=0}^{k} e_\phi[n], \tag{1.19}$$

where K_1 and K_2 are the proportional and integral PLL hyperparameters, respectively. The PLL error signal is defined as

$$e_\phi[k] = \Im\Big[z_1[k] e^{-j\hat{\phi}[k-1]} \cdot (z_2[k] + \hat{d}[k])^*\Big], \tag{1.20}$$

where $\Im$ is the imaginary part, $z_1[k]$ and $z_2[k]$ denote the outputs of the feedforward and feedback filters, respectively. The vector of decided symbols fed back to the equalizer is given by

$$\hat{\mathbf{d}}[k] = \Big[\hat{d}[k-1], \hat{d}[k-2], \ldots, \hat{d}[k-L_B]\Big]^T. \tag{1.21}$$

The DFE error signal is defined as

$$e[k] = \hat{d}[k] - z[k], \tag{1.22}$$

where $z[k]$ is the soft symbol estimate at time index k. It is worth noting that the PLL error signal is derived from the same instantaneous squared-error criterion as that used for the DFE adaptation. Indeed, using the equalizer output relation $z[k] = z_1[k]e^{-j\hat{\phi}[k-1]} - z_2[k]$, the DFE error can be written as $e[k] = z_2[k] + \hat{d}[k] - z_1[k]e^{-j\hat{\phi}[k-1]}$. The negative gradient of the instantaneous cost $\mathbb{E}\{|e[k]|^2\}$ with respect to the phase estimate therefore yields $e_\phi[k]$. Consequently, the DFE and the PLL are coupled through the same mean-square-error objective, $\mathbb{E}\{|e[k]|^2\}$.

To track channel variations, the filter coefficients are updated at each symbol using an adaptive algorithm. In this work, the Improved Proportionate Normalized Least Mean Squares (IPNLMS) algorithm is employed (Pelekanakis et al., 2010 [54]). This algorithm combines the fast convergence of proportionate methods in sparse channels (Duttweiler, 2000 [55]) with the robustness of normalized LMS in more dispersive environments.

The filter update equations are given by

$$\mathbf{A}(k) = \mathbf{A}(k-1) + \mu_A \frac{\mathbf{G}_A(k-1)\mathbf{x}(k)e(k)^*}{\mathbf{x}(k)^\dagger \mathbf{G}_A(k-1)\mathbf{x}(k) + \delta_A} e^{j\phi(k-1)}, \tag{1.23}$$

$$\mathbf{B}(k) = \mathbf{B}(k-1) - \mu_B \frac{\mathbf{G}_B(k-1)\hat{\mathbf{d}}(k)e(k)^*}{\hat{\mathbf{d}}(k)^\dagger \mathbf{G}_B(k-1)\hat{\mathbf{d}}(k) + \delta_B}, \tag{1.24}$$

where μ_A and μ_B are the step sizes of the feedforward and feedback filters, respectively, and k is the symbol index. δ_A and δ_B are regularization constants. The input vector is defined as $\mathbf{x}(k) = [x(k), x(k-1), \ldots, x(k-L_A+1)]^T$.

The diagonal weighting matrix $\mathbf{G}_A$ has entries

$$g_A^{(i)}(k) \triangleq \frac{1-\beta}{2L_A} + (1+\beta)\frac{|\mathbf{A}_i(k)|}{2\|\mathbf{A}(k)\|_1 + \epsilon}, \tag{1.25}$$

where $\mathbf{A}_i(k)$ is the ith tap of the feedforward filter, ϵ is a regularization constant, and β controls the degree of proportionality in the update. The matrix $\mathbf{G}_B$ is defined analogously for the feedback filter. The parameter β governs the sensitivity of the update to tap magnitudes, with $\beta = -1$ corresponding to the NLMS update and $\beta = 1$ yielding a fully proportionate update (Pelekanakis et al., 2010 [54]).

The receiver architecture described above involves several hyperparameters whose values strongly influence performance. In this work, the following hyperparameters are considered for optimization, together with their corresponding ranges:

Main Hyperparameters

- $\beta \in [-1, 1]$: channel sparseness parameter in IPNLMS.
- $\mu_A \in [10^{-5}, 1]$: step size of the feedforward filter.
- $\mu_B \in [10^{-5}, 1]$: step size of the feedback filter.
- $L_A \in [10, 3000]$: length of the feedforward filter.
- $L_B \in [0, 1000]$: length of the feedback filter.
- $K_1 \in [0, 2]$: proportional gain of the PLL.
- $K_2 \in [0, 2]$: integral gain of the PLL.

The hyperparameters of the receiver architecture shown in Figure 1.4 encode implicit assumptions about the channel state. For optimal performance, these assumptions must be consistent with the actual channel conditions. For example, the adaptive equalizer step sizes μ_A and μ_B, as well as the PLL gain parameters K_1 and K_2, must be tuned to match the channel variation rates, while the equalizer filter lengths should be chosen to cover the observed delay spread, with larger delay spreads requiring more filter coefficients (Riera-Palou et al., 2000b [56]; Kusuma et al., 2018 [57]; Bae et al., 1997 [58]; Ye et al., 2021 [59]).

1.4 Receiver Black-box

Hyperparameter tuning of underwater acoustic receivers is typically performed through manual adjustment (Mahmutoglu et al., 2016 [60]; Yang et al., 2013 [61]; Stojanovic et al., 1993 [48]; Deguchi et al., 2023 [62]; Cuji et al., 2024b [63]; Pelekanakis et al., 2010 [54]; Selmi et al., 2010 [64]). While this approach can yield acceptable performance, it implicitly assumes channel stationarity, does not account for sensitivity to varying operational conditions, and overlooks the complexity induced by hyperparameter interdependence. Dynamic adaptation methods have been proposed, but they typically require frequent channel estimation, which limits their practical applicability (Chen et al., 2024 [65]; Riera-Palou et al., 2000b [56]).

DFE performance is highly sensitive to the feedforward and feedback filter lengths, a well-known fact in the UAC community (Kusuma et al., 2018 [57]; Cheikh Melainine et al., 2024 [32]). The PLL, which is typically used to mitigate Doppler effects through carrier-frequency synchronization (Eynard et al., 2008 [50]; Oh et al., 1996 [51]; Cuji et al., 2024a [52]; Johnson et al., 1997b [53]), primarily depends on its gain parameters and is strongly coupled to the DFE hyperparameters.

The aim of this study is to automate receiver tuning, i.e., to automate the search for hyperparameter values that maximize receiver performance. A prerequisite is therefore to define a receiver performance metric, which is a non-trivial task. Ideally, one would evaluate the Bit-Error-Rate (BER) or Packet-Error-Rate (PER). However, such metrics cannot be computed without access to the ground truth, i.e., the transmitted bits, which is generally unavailable in practice. Using a large known portion of the received signal is also undesirable, as it reduces the useful data rate. A common alternative is to use a proxy metric for BER, such as the average output Signal-to-Noise Ratio (SNR) computed over a packet of N_s symbols:

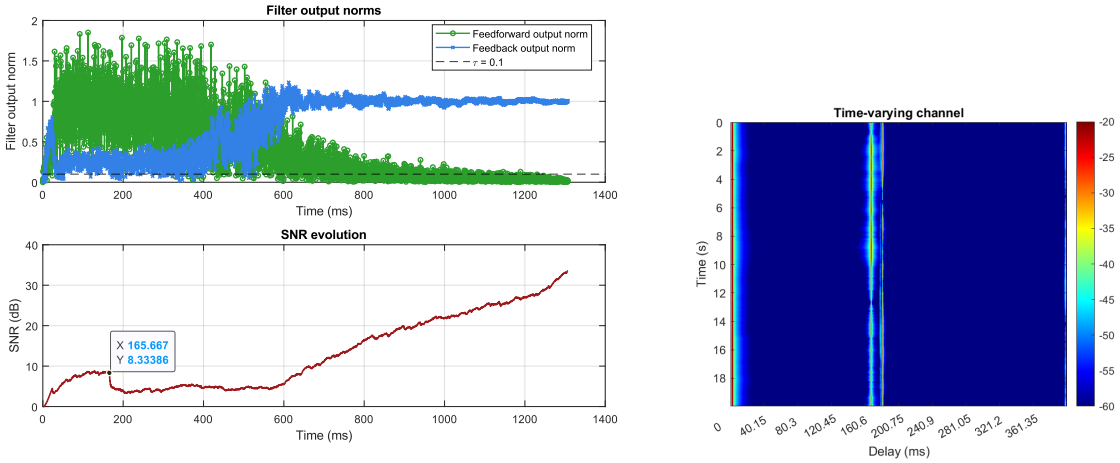
$$\rho_{\text{SNR}} = -10\log_{10}\left(\frac{1}{N_s}\sum_{k=1}^{N_s}\left|e[k]\right|^2\right), \tag{1.26}$$

where $e[k]$ is the error signal between the soft and hard symbol estimates defined in (1.22).

This quantity is to be contrasted with **input SNR** (in dB) as

$$\rho_{\text{IN}} = 10\log_{10}\left(\frac{E_b}{N_0}\right) + 10\log_{10}\left(\frac{2R_s}{B_\ell}\right), \tag{1.27}$$

where N_0 is the noise power spectral density, E_b/N_0 is the energy-per-bit to noise spectral density ratio, R_s is the symbol rate, and B_ℓ is the occupied bandwidth. This quantity characterizes the SNR available at the receiver input given a target spectral efficiency $2R_s/B_\ell$ and is also known as the In-band SNR.



(a) (b)

Figure 1.6 – DFE hallucination example. In (a), the top plot shows the filter output norms: the curve marked with –o– corresponds to the feedforward ℓ_2 output norm, and the curve marked with –*– corresponds to the feedback ℓ_2 output norm. The bottom plot shows the SNR evolution during hallucination (red curve —). In (b), the instantaneous channel power in the time–delay domain of the corresponding simulated channel is shown. Doppler is compensated in the right figure.

A major challenge when using the output SNR as an objective is the degenerate DFE state coined in this study as ***DFE hallucination***, illustrated in Figure 1.6.

In this regime, the norm of the feedforward filter output decreases toward zero, while the feedback filter output increases toward the constellation magnitude (QPSK). DFE hallucinations typically stem from a strong disturbance that induces a large decision error in decision-directed mode. Such a disturbance may, for instance, be the abrupt appearance of an additional echo. This is the case in Figure 1.6, where a second replica of the transmitted signal appears around the delay of the second echo in (b) (approximately 165 ms), inducing a sharp SNR drop after which the DFE does not recover.

The hyperparameters $\boldsymbol{\theta}$ that produce hallucinations often yield the largest apparent values of ρ_{SNR} in decision-directed mode, for the following reason: the SNR proxy in (1.26) is maximized when the decided symbols match the soft estimates (i.e., $e[k] = 0$). Such behavior occurs in a hallucination state since the feedforward branch progressively collapses toward zero; once its contribution becomes negligible relative to the feedback branch, the feedback filter effectively predicts its own output, which can lead to repeating patterns of decided symbols and spuriously high output SNR. In this regime, the decision device converges to a state where decisions are driven almost exclusively by the feedback filter and no longer by the feedforward filter.

DFE hallucinations must therefore be detected and penalized by the objective function we are constructing, in order to have an accurate description of receiver performance. Detection is performed by thresholding the energy of the last m feedforward outputs:

$$\mu_m \triangleq \sum_{k=N_s-m}^{N_s} \left|z_1[k]\right|^2, \tag{1.28}$$

where $z_1[k]$ is the kth feedforward output. Subsequently, if $\mu_m \leq \tau_{\text{hall}}$, the objective strongly penalized the associated hyperparameter values. Detecting hallucinations in long packets can be achieved with relatively small values of τ_{hall}, as suggested by the trends in Figure 1.6. Selecting τ_{hall} involves balancing false positives (more likely for larger thresholds) against missed detections (more likely for smaller thresholds, especially in short packets). In our study, we set $\tau_{\text{hall}} = 0.1$ based on empirical tuning, and $m = 200$. It should be noted that hallucinations could also be detected using CRC checks. For instance, if the estimated average output SNR is high, above a chosen threshold, while the CRC still fails, this may indicate a hallucination. However, this would require defining a minimum SNR threshold above which CRC success is expected. This question is outside the scope of the present work.

The objective function used in this work computes the average output SNR (in dB) when no hallucination is detected. If a hallucination is detected, a penalty is assigned. This objective function is defined as:

Receiver black-box function

$$f(\boldsymbol{\theta}) \triangleq \begin{cases} \rho_{\text{SNR}} & \text{if } \mu_m > \tau_{\text{hall}}, \\ -50 \text{ dB} & \text{if } \mu_m \leq \tau_{\text{hall}}. \end{cases} \tag{1.29}$$

where μ_m is defined in (1.28), $z_1[k]$ is the kth feedforward output, N_s is the number of symbols in a packet, $\tau_{\text{hall}} = 0.1$ is the hallucination detection threshold, and $m = 200$.

Now that f has been defined, we inspect its properties. To proceed, we generate five channels using the Maximum Entropy simulator (Socheleau et al., 2010 [66]). Packets are transmitted at $B_1 = 4.5$ kHz bandwidth , and additive white Gaussian noise is added such that the receiver input SNR is 12 dB.

First, to simplify the analysis, we focus on the sensitivity of f with respect to the DFE filter lengths L_A and L_B, while fixing the remaining hyperparameters to

$$K_1 = 0.034, \quad K_2 = 7.3 \times 10^{-5}, \quad \mu_A = 0.05, \quad \mu_B = 0.1, \quad \beta = 0.25. \tag{1.30}$$

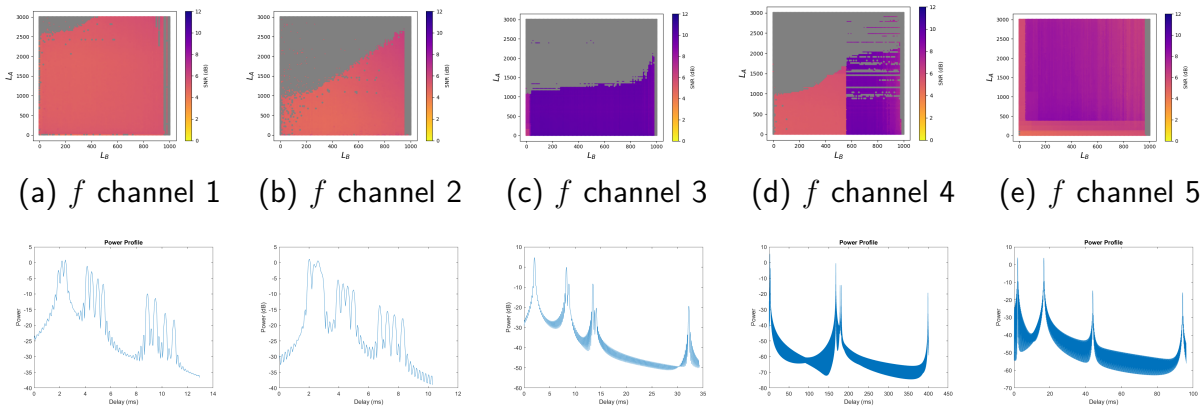
Figure 1.7 shows five realizations of f over (L_A, L_B), each computed on a single packet corresponding to one of the five channels. Three regions can be distinguished:

— **Low-output SNR region**: lighter-colored points (orange/yellow) indicating poor performance.

— **High-output SNR region**: darker-colored points (blue/purple) indicating strong performance.

— **Hallucination-prone region**: gray points indicating parameter settings prone to hallucination.

We observe that f varies substantially across channels (i.e., operational conditions). In particular, larger delay spreads generally require larger feedback filter lengths L_B.



(a) f channel 1 (b) f channel 2 (c) f channel 3 (d) f channel 4 (e) f channel 5

(f) Power profile 1 (g) Power profile 2 (h) Power profile 3 (i) Power profile 4 (j) Power profile 5

Figure 1.7 – Five channel realizations (top row) with their corresponding power-delay profiles (bottom row).

In Figure 1.8, f is computed for the same channel (channel (i) in Figure 1.7) but for different noise realizations. We observe that the performance associated with a given parameter setting can change significantly across noise realizations. Hallucinations are also sensitive to noise because noise can induce decision errors in decision-directed operation.

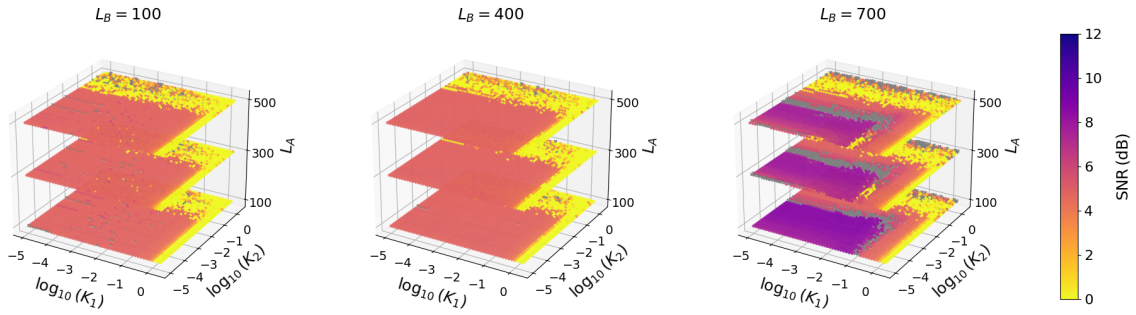
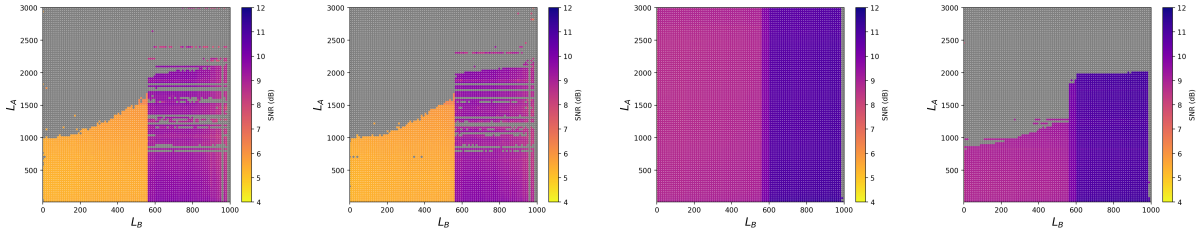


Figure 1.9 – Objective function f for (L_A, L_B, K_1, K_2) in the same simulated channel. K_1, and K_2 are shown in log-scale.



(a) (b) (c) (d)

Figure 1.8 – Objective function f for four different noise realizations over (L_A, L_B) in the same simulated channel.

In higher dimension, for instance if we consider $\boldsymbol{\theta} = [L_A, L_B, K_1, K_2]^T$, these hyperparameters are strongly correlated. Consequently, to obtain an optimal receiver, they must be tuned jointly, which is not straightforward as featured in Figure.1.9. From the previous figures, we can make the following assumption:

Assumption 1 *The performance metric function f is highly sensitive to:*
- *the operational conditions, i.e., the communication channel;*
- *the noise realization;*
- *non-smooth behavior, as it often contains distinct regimes;*
- *increasing complexity with dimensionality;*
- *limited* a priori *knowledge of its structure;*
- *the prohibitive cost of exhaustive evaluation.*

For these reasons, f is treated as a black-box function. Identifying high-performing regions of this objective is non-trivial without prior knowledge, and exhaustive search quickly becomes infeasible. For example, a grid search with only 50 candidate values per hyperparameter over five hyperparameters already requires $50^5 = 3.125 \times 10^8$ evaluations.

Exploring other dimensions of f would be redundant, and we have observed experimentally that the hyperparameters are often highly correlated.

1.5 Conclusion

In this chapter, we provided an overview of UAC, with an emphasis on the challenges specific to this wireless communication technology. We described the main sources of distortion, namely multipath, Doppler effects, and noise, and discussed the time-varying nature of the channel and typical rates of change. We then specified the transmitter settings used throughout the remainder of the manuscript and presented the receiver architecture considered in this study. In particular, we introduced the associated signal-processing blocks and the hyperparameters that govern them, focusing on those of the DFE and the PLL tracking loop.

Finally, we defined the receiver performance metric f as a function of the receiver hyperparameters based on the output SNR and introduced the hallucination pathology that can affect this metric. We proposed a hallucination detection method based on monitoring the norms of the DFE feedforward and feedback filter outputs.

The performance metric f and the hallucination detection procedure constitute original contributions of this study.

Chapter 2

Black-boxes and Bayesian optimization

2.1 Introduction

In Chapter 1, we defined a receiver performance metric as a function of a set of receiver hyperparameters. Optimizing this objective provides hyperparameter values that improve receiver performance, but the mapping from hyperparameters to performance is not available in closed form, its gradients are not reliable, and each evaluation requires running the full receiver chain. We therefore treat this objective as a black-box function and leverage black-box optimization methods to automate receiver fine-tuning.

Black-box optimization appears across many scientific and engineering domains where evaluations are expensive and the system is too complex for exact analysis. In chemistry and materials science, design variables encode compositions or reaction conditions and the objective is a scalar laboratory or simulation outcome (Wu et al., 2024 [67]; Frazier, 2015 [68]; Trabucco et al., 2022 [69]). In experimental biology, designs correspond to protocols or formulations and evaluations require wet-lab measurements (Kawabata et al., 2025 [70]; Rosa et al., 2022 [71]). In data-driven design, datasets are used to learn surrogates that can be optimized to propose improved designs (Kim et al., 2025 [72]). In modern machine learning, hyperparameter tuning and architecture search are expensive because each evaluation involves training and validation (Franceschi et al., 2024 [73]; Shahriari et al., 2016 [74]; Young et al., 2020 [75]; Eimer et al., 2023 [76]).

To the best of our knowledge, this black-box formulation has not been used as a principled framework for tuning telecommunication receivers, and in particular UAC receivers. This chapter is organized as follows. Section 2.2 reviews black-box optimization techniques other than Bayesian optimization. Section 2.3 then introduces Bayesian optimization and discusses it in detail.

2.2 Black-box Optimization techniques

Formally, Black-box optimization is modeled as an optimization problem

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}\subset\mathbb{R}^d} f(\boldsymbol{\theta}), \tag{2.1}$$

where $\boldsymbol{\theta}$ collects the tunable design or tuning variables, $f(\boldsymbol{\theta})$ is a scalar objective that can only be observed through evaluations at chosen points (possibly with noise), and analytic gradients and Hessians are unavailable or unreliable. The only information available to the optimizer is the set of past evaluations $\mathcal{D} \triangleq \{(\boldsymbol{\theta}, f(\boldsymbol{\theta}))\}$. Derivative-free optimization (DFO) methods are explicitly designed for this regime. Within DFO, methods are commonly organized into three families.

Direct and pattern-search methods (Larson et al., 2019 [77]). Algorithms such as coordinate search, generalized pattern search, and mesh adaptive direct search explore a neighborhood of the current iterate, that is the latest tested point $\boldsymbol{\theta}$, using a finite set of search directions. At each iteration, they evaluate candidate steps along these directions and accept those that yield sufficient improvement.

At iteration k, given a mesh size Δ_k and a set of polling directions $D_k = \{d_1, \ldots, d_m\}$, the method evaluates $f(\boldsymbol{\theta} + \Delta_k d_i)$ for one or more $d_i, i \in [1, \ldots, m]$. If any poll point yields a better function value, the best such point becomes $\boldsymbol{\theta}_{k+1}$ and the mesh size may be increased. Otherwise, $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}$ and Δ_k is reduced. Under mild conditions on the direction sets, such as positive spanning of $\mathbb{R}^d$, one can prove convergence of limit points to stationary points even though no explicit gradient is computed. Simpler heuristics such as Nelder–Mead (Nelder et al., 1965 [78]) maintain a simplex of $d+1$ points and use reflection, expansion, and contraction moves based on the relative objective values at the simplex vertices. These methods are conceptually simple and robust to modest non-smoothness and noise, but they can be evaluation-intensive, especially in higher dimensions.

Model-based DFO and trust-region methods (Larson et al., 2019 [77]). These methods build a local surrogate model $\hat{f}_k$ of f around the current iterate using interpolation or regression on a small set of previously evaluated points, and then optimize $\hat{f}_k$ inside a trust region. Typical choices include fully quadratic polynomials or radial basis function interpolants. At iteration k, a trust-region subproblem

$$s_k = \arg\min_{\|s\|\le\Delta_k} \hat{f}_k(\boldsymbol{\theta}_k + s) \tag{2.2}$$

is approximately solved, $f(\boldsymbol{\theta} + s_k)$ is evaluated, and the ratio of actual to predicted improvement,

$$\rho_k = \frac{f(\boldsymbol{\theta}) - f(\boldsymbol{\theta} + s_k)}{\hat{f}_k(\boldsymbol{\theta}) - \hat{f}_k(\boldsymbol{\theta} + s_k)}, \tag{2.3}$$

is used to decide whether to accept the step and how to adjust Δ_k. Under suitable model-quality conditions, these methods converge to first-order stationary points. Compared with direct search, they exploit local curvature information encoded in $\hat{f}_k$ and can be significantly more sample-efficient near optima, but they require careful maintenance of interpolation sets and can suffer when noise or strong time variation undermines the surrogate's local accuracy, which is precisely the regime encountered when the channel is evolving.

Randomized gradient-approximation schemes (Larson et al., 2019 [77]). A third family approximates gradients using randomized finite differences and then applies a gradient-type update. Prototypical examples include Kiefer–Wolfowitz (Kiefer et al., 1952 [79]) schemes and Simultaneous Perturbation Stochastic Approximation (SPSA) (Cullen, 1990 [1]). In a two-point random-direction scheme, one samples a random direction u_k (often Gaussian), evaluates

$$f(\boldsymbol{\theta} + c_k u_k), \qquad f(\boldsymbol{\theta} - c_k u_k), \tag{2.4}$$

and forms a stochastic gradient estimate

$$g_k = \frac{f(\boldsymbol{\theta} + c_k u_k) - f(\boldsymbol{\theta} - c_k u_k)}{2c_k} u_k. \tag{2.5}$$

Then $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta} - \alpha_k g_k$, with step-size schedules α_k and c_k. These schemes scale more gently with dimension than coordinate-wise finite differences, but their convergence can be sensitive to noise and schedules.

In parallel to these numerical-analysis style DFO methods, the evolutionary computation community has developed population-based meta-heuristics such as genetic algorithms, evolution strategies, particle swarm optimization, differential evolution, covariance matrix adaptation evolution strategies, and many variants that treat the objective as a fitness function evaluated on a population of candidate solutions (Michalewicz, 2013 [80]; Kennedy et al., 1995 [81]; Dorigo et al., 2006 [82]). A generic evolutionary loop maintains a population $\{\boldsymbol{\theta}^{(j)}\}_{j=1}^{N}$, applies variation operators such as mutation and crossover, and then selects individuals based on fitness values $f(\boldsymbol{\theta}^{(j)})$. Surrogate-assisted evolutionary algorithms insert machine-learning models inside the evolutionary loop to predict fitness and selectively call the expensive true objective only on informative points. Surveys document their successful use in expensive engineering design (He et al., 2023 [83]).

2.3 Bayesian Optimization

2.3.1 General approach

Bayesian Optimization (BO) sits at the intersection of global optimization, surrogate modeling, and uncertainty quantification. In the broader landscape of black-box optimization, it complements classical derivative-free optimization (DFO) methods, such as direct search and trust-region DFO, as well as population-based evolutionary heuristics, such as genetic algorithms and evolution strategies, as discussed in Section 2.2. These families target the same regime: objectives that are expensive to evaluate and do not provide reliable derivatives. However, they differ in how they spend a scarce evaluation budget.

The main practical advantage of BO is sample efficiency under tight budgets. Many DFO schemes are designed primarily for local improvement. Under standard assumptions, they provide convergence guarantees to stationary points, so their guarantees are local rather than global. In addition, they typically require multiple evaluations per iteration (polling directions, interpolation sets, or finite-difference-style probes), which can become costly when each evaluation is expensive or when noise forces repeated sampling to stabilize decisions. For expensive systems, this makes it difficult to allocate enough evaluations for both global exploration and local refinement within a small budget. (Larson et al., 2019 [77])

Evolutionary methods, by construction, trade evaluations for robustness. They maintain a population of candidate solutions and update it over many generations. In continuous domains, typical population sizes are on the order of tens to hundreds, and practical runs often require tens to hundreds of generations to stabilize. This corresponds to on the order of 10^3 to 10^5 objective evaluations in routine use, and sometimes more when the search space is noisy or poorly scaled. Such budgets are reasonable when evaluations are cheap, but they become prohibitive when each trial is a high-fidelity simulation, a long-running experiment, or an operational field test. (Katoch et al., 2021 [84])

BO is designed for precisely the opposite regime. It maintains an explicit surrogate with quantified uncertainty and uses that uncertainty to guide evaluation selection. The acquisition policy concentrates evaluations where they are expected to be most useful, either because the surrogate predicts strong performance or because uncertainty is high enough that a single evaluation could change the decision landscape. This uncertainty-aware mechanism is a direct lever for controlling the exploration–exploitation trade-off in a way that is difficult to achieve systematically with local DFO steps or large-population evolutionary search.

As a result, BO is often the preferred choice in the "few dozen to few hundred evaluations" regime that arises in high-fidelity simulation tuning and in operational tuning

of real communication receivers, where each configuration change incurs non-trivial cost and delay. In such receiver-tuning settings, BO has been shown to provide systematic, data-efficient configuration improvement without relying on expert hand-tuning, aligning directly with the black-box constraints and operational requirements described earlier.

We write the black-box tuning task as

$$\boldsymbol{\theta}^{\star} = \arg\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f(\boldsymbol{\theta}), \tag{2.6}$$

where $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subset \mathbb{R}^d$ collects the d tunable hyperparameters and $f(\boldsymbol{\theta})$ is an expensive-to-observe objective (possibly noisy).

For performance metrics naturally stated as maximization, one simply minimizes $-f(\boldsymbol{\theta})$ without changing the BO machinery.

Remark 1 *Note that hyperparameter vectors are normalized within the unit hypercube.*

This normalization is crucial to avoid and consequences from hyperparameter spaces with different scales. BO algorithms operate following a sequential decision-making loop where, at each iteration, we decide on the next hyperparameters to test on the black-box function. Concretely, given an evaluation budget of N trials, BO proceeds by building and refining a surrogate from observed data and using an acquisition policy to select the next evaluation. A typical BO iteration procedure is:

1. Initialize a warm-up dataset by evaluating n initial configurations (often space-filling or random):
$$\mathcal{D}_0 \triangleq \{(\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i))\}_{i=1}^{n}. \tag{2.7}$$
2. From $\mathcal{D}_{j-1}$, fit a surrogate F that returns a predictive distribution for $f(\boldsymbol{\theta}), \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}$.
3. Select the next candidate by computing an acquisition policy π by choosing
$$\boldsymbol{\theta}^{can} = \pi(F) \triangleq \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} a(\boldsymbol{\theta}), \tag{2.8}$$
where $a(\cdot)$ trades off *exploitation* (preferring configurations predicted to be good) and *exploration* (probing uncertain regions).
4. Evaluate the black-box at $\boldsymbol{\theta}^{can}$ and update the dataset with both $\boldsymbol{\theta}^{can}$ and $f(\boldsymbol{\theta}^{can})$
$$\mathcal{D}_j = \mathcal{D}_{j-1} \cup \{(\boldsymbol{\theta}^{can}, f(\boldsymbol{\theta}^{can})\}. \tag{2.9}$$

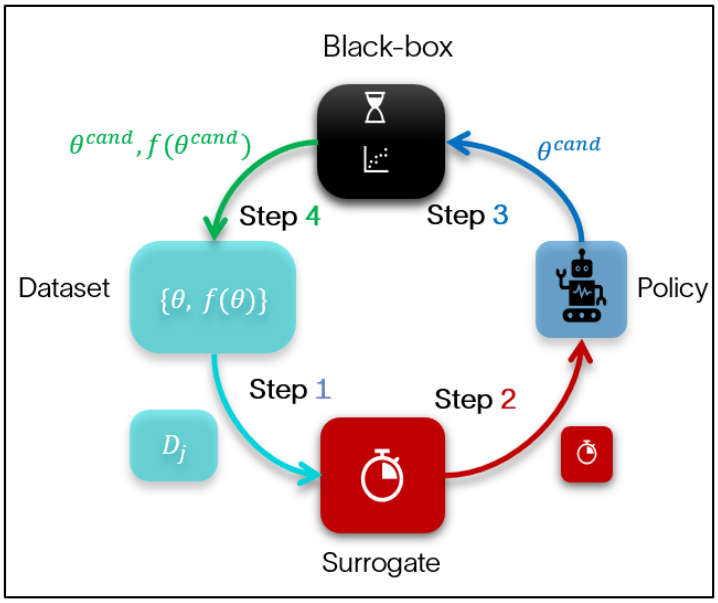


Figure 2.1 – Closed-loop process of **one** Bayesian Optimization iteration

Figure 2.1 illustrates this closed-loop process. Once the budget N of iterations is exhausted, the recommended configuration is typically the best observed point,

$$\hat{\boldsymbol{\theta}}_N = \arg \min_{(\boldsymbol{\theta}, y) \in \mathcal{D}_N} y, \tag{2.10}$$

or, in some implementations, the minimizer of the surrogate expectation (Hernández-Lobato et al., 2016 [85]). Termination conditions in the BO literature are most commonly based on a fixed budget, expressed either in wall-clock time or in a maximum number of iterations. Throughout this work we fix a number of iterations as a budget. Other termination conditions have also been considered. For greedy policies, the algorithm may be stopped if it stagnates, for example if the proposed points remain within a region of fixed radius over several iterations. BO procedures can also be terminated when the best observed objective value has not improved over a prescribed number of iterations. In high-dimensional spaces, selecting an adequate budget a priori is challenging. In that case, termination may also be triggered when the acquisition function values become uniformly low, indicating limited expected benefit from further evaluations (see Section 2.3.3) (Bischl et al., 2023 [86]).

2.3.2 Surrogates models

In this section, we focus on the classical setting where the objective function is static over the optimization budget. The time-varying case is presented in Chapter 4. Bayesian

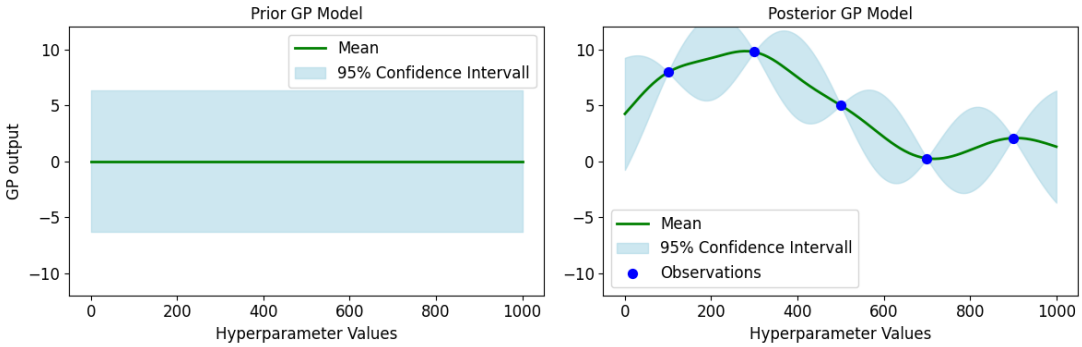


Figure 2.2 – (Left) Prior Gaussian process (GP) over a range of hyperparameter values. In the absence of training data, the mean remains flat and uncertainty is high, as reflected by the wide 95% confidence interval (blue region). (Right) Posterior GP after incorporating training observations (blue dots). The mean function (green) exhibits learned structure, while the uncertainty band contracts around observed points, indicating reduced variance in data-rich regions.

optimization relies on a surrogate model because $f(\boldsymbol{\theta})$ is expensive to evaluate: the surrogate provides a data-efficient approximation of the objective together with uncertainty estimates that guide exploration (Garnett, 2023a [12]). A common choice of surrogate F is Gaussian process (GP) regression (Rasmussen et al., 2006 [87]; Garnett, 2023a [12]):

$$F \sim \mathcal{GP}\Big(m(\cdot), k(\cdot,\cdot)\Big), \tag{2.11}$$

which is fully characterized by its mean function $m : \boldsymbol{\Theta} \to \mathbb{R}$ and covariance function $v : \boldsymbol{\Theta} \times \boldsymbol{\Theta} \to \mathbb{R}$:

$$m(\boldsymbol{\theta}) \triangleq \mathbb{E}[F(\boldsymbol{\theta})] \tag{2.12}$$

$$v(\boldsymbol{\theta}, \boldsymbol{\theta}') \triangleq k(\boldsymbol{\theta}, \boldsymbol{\theta}') = \mathbb{E}\Big[(F(\boldsymbol{\theta}) - m(\boldsymbol{\theta})) \times (F(\boldsymbol{\theta}') - m(\boldsymbol{\theta}'))\Big] \tag{2.13}$$

where $F(\boldsymbol{\theta}) \sim \mathcal{N}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta}, \boldsymbol{\theta})\Big)$.

Before any data are acquired, the GP model is referred to as the *prior*. Let $X = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n]^\top$ denote the set of evaluated hyperparameter configurations, and let $\mathbf{y} = [f(\boldsymbol{\theta}_1), \ldots, f(\boldsymbol{\theta}_n)]^\top$ be the corresponding black-box evaluations. Conditioning the GP on these observations yields updated mean and variance functions given by

$$m(\boldsymbol{\theta}) = \boldsymbol{k}(\boldsymbol{\theta})^\top \boldsymbol{K}^{-1} \boldsymbol{y}, \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{2.14}$$

$$v(\boldsymbol{\theta}, \boldsymbol{\theta}') = k(\boldsymbol{\theta}, \boldsymbol{\theta}') - \boldsymbol{k}(\boldsymbol{\theta})^\top \boldsymbol{K}^{-1} \boldsymbol{k}(\boldsymbol{\theta}'), \quad \forall (\boldsymbol{\theta}, \boldsymbol{\theta}') \in \boldsymbol{\Theta}^2. \tag{2.15}$$

where $\boldsymbol{k}(\boldsymbol{\theta}) = \Big[k(\boldsymbol{\theta}, \boldsymbol{\theta}_1), \ldots, k(\boldsymbol{\theta}, \boldsymbol{\theta}_n)\Big]^\top$. The kernel matrix $\boldsymbol{K}$ is defined by $\Big[\boldsymbol{K}\Big]_{ij} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$. The GP endowed with these updated mean and variance functions is referred to as the *posterior* model.

Figure 2.2 illustrates how $m(\boldsymbol{\theta})$ and $v(\boldsymbol{\theta})$ evolve as additional observations—hyperparameter configurations and their corresponding black-box evaluations—are incorporated.

The function k is the kernel and plays a central role in the construction of the surrogate model. It encodes prior assumptions about properties such as smoothness, stationarity, anisotropy, and compositional structure. In practice, choosing a kernel amounts to specifying which variations of f are deemed plausible *a priori*. The most common assumptions include:

— **Stationarity:** correlations depend only on $|\boldsymbol{\theta} - \boldsymbol{\theta}'|$. Many commonly used kernels satisfy this property.

— **Smoothness class:** the number of derivatives that typical sample paths admit (e.g., Matérn kernels explicitly control differentiability).

While many kernel families exist, we restrict ourselves in this manuscript to two widely used choices: the Squared Exponential (SE) kernel and the Matérn family. These kernels are flexible, well understood, and capable of modeling a broad range of smoothness behaviors.

To define these kernels, let $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_d] \in \boldsymbol{\Theta}$ denote a d-dimensional input vector.

Definition 3 *The Squared Exponential (SE) kernel k_{SE} is defined, for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \boldsymbol{\Theta}$, as*

$$k_{\mathrm{SE}}(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sigma_f^2 \exp\left(-\sum_{p=1}^{d} \frac{(\theta_p - \theta_p')^2}{2l_p^2}\right), \tag{2.16}$$

where σ_f^2 is the output scale and $\{l_p\}_{p=1}^d$ are the lengthscales, which respectively control the overall range of the objective function f and its smoothness along each input dimension.

Remark 2 *Larger values of l_p correspond to smoother surrogate behavior with lower uncertainty along dimension p, whereas smaller values of l_p induce higher roughness, as illustrated in Figure 2.3.*

Definition 4 *The Matérn kernel family k_ν is defined, for all $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \boldsymbol{\Theta}$, as*

$$k_\nu(\boldsymbol{\theta}, \boldsymbol{\theta}') = \sigma_f^2 \frac{2^{1-\nu}}{\Gamma(\nu)} \left(\sqrt{2\nu} \sum_{p=1}^{d} \frac{(\theta_p - \theta_p')^2}{2l_p^2}\right)^{\nu} K_\nu \left(\sqrt{2\nu} \sum_{p=1}^{d} \frac{(\theta_p - \theta_p')^2}{2l_p^2}\right), \tag{2.17}$$

where $\Gamma(\nu)$ denotes the Gamma function and K_ν the modified Bessel function with smoothness parameter ν.

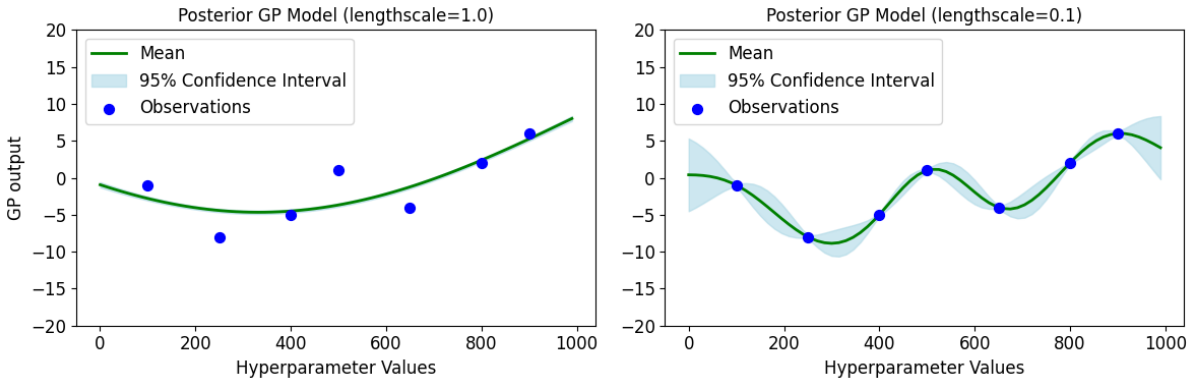


Figure 2.3 – GP posterior with an SE kernel for two different lengthscale values.

Remark 3 *The Matérn kernel is parameterized by the output scale σ_f^2, the lengthscales $\{l_p\}$, and the smoothness parameter ν. Sample paths (realizations) of the corresponding GP are $\lceil \nu - 1 \rceil$ times differentiable. In particular, $\nu = \frac{1}{2}$ yields the exponential kernel, which produces rough sample paths, while $\nu \to \infty$ recovers the SE kernel with infinitely differentiable realizations.*

Learning kernel hyperparameters. As shown in Definitions 3 and 4, these kernels—as is the case for all kernels used in Bayesian optimization—depend on multiple hyperparameters. Moreover, the structure of the surrogate is highly sensitive to these hyperparameters, as highlighted in Remarks 2 and 3. Fortunately, Gaussian process surrogates provide an automatic and principled mechanism for learning these hyperparameters from data.

Let $X = [\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_n]^\top$ denote the set of evaluated black-box hyperparameters, and let $\mathbf{y} = [f(\boldsymbol{\theta}_1), \ldots, f(\boldsymbol{\theta}_n)]^\top$ be the corresponding black-box evaluations. The kernel matrix $\boldsymbol{K}$ is defined by $\left[\boldsymbol{K}\right]_{ij} = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)$, where k denotes the chosen kernel function. The kernel hyperparameters are commonly estimated by maximizing the log marginal likelihood (Garnett, 2023a [12]),

$$\log p(\mathbf{y} \mid X, h_k) = \underbrace{-\frac{1}{2}\mathbf{y}^\top(\mathbf{K} + \sigma_n^2\mathbf{I})^{-1}\mathbf{y}}_{\text{data fit}} \underbrace{-\frac{1}{2}\log\left|\mathbf{K} + \sigma_n^2\mathbf{I}\right|}_{\text{model complexity}} \underbrace{-\frac{n}{2}\log(2\pi)}_{\text{regularization term}} . \tag{2.18}$$

Here, h_k denotes the vector of kernel hyperparameters, for example $h_k = [\{l_p\}_{p=1}^d, \sigma_f]^\top$ in the case of the SE kernel. Maximizing (2.18) yields an automatic Occam-style trade-off between data fit and model complexity (Rasmussen et al., 2006 [87]). The resulting objective is differentiable and can be efficiently optimized using standard gradient-based optimizers. However, reliable estimation of kernel hyperparameters typically requires a sufficient number of observations, which is often not available in practical BO settings.

Beyond GP surrogates. In addition to Gaussian processes, several alternative surrogate models have been proposed in the literature. Tree-based surrogates, such as random forests, are effective in search spaces that mix discrete and continuous hyperparameters and can scale to higher-dimensional settings (Hutter et al., 2011 [88]; Eggensperger et al., 2013 [89]). Density-based surrogates, such as the Tree-structured Parzen Estimator (TPE), scale well to high-dimensional hyperparameter optimization problems but decouple density estimation from acquisition interpretability (Bergstra et al., 2011 [90]; Watanabe, 2023 [91]). Neural surrogates—including Bayesian neural networks and neural network ensembles, where multiple networks are trained on bootstrap resamples of the original dataset—offer increased expressivity and scalability. However, these methods are typically data-hungry, and in low-data regimes they may lead to over-exploration or over-exploitation depending on architectural choices. They also incur higher training costs and rely on approximate uncertainty estimates, although they are naturally parallelizable, which is particularly attractive for large-scale hyperparameter tuning of machine learning models (Snoek et al., 2015 [92]; Li et al., 2023a [93]).

For the problem of hyperparameter tuning of telecommunication receivers operating in adverse environments, the limited amount of data available for each operational setting makes Gaussian process surrogates the default and most appropriate choice.

2.3.3 Common Acquisition Policies

As illustrated in Figure 2.1, at each iteration j a surrogate model is constructed from the dataset $\mathcal{D} = \{(\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i))\}_{i=1}^{j-1}$. An acquisition function is then used to interpret this surrogate and select the candidate hyperparameter configuration $\boldsymbol{\theta}^{\text{can}}$ to be evaluated on the black-box objective. The choice of acquisition policy is therefore critical to the overall performance of Bayesian optimization.

In this section, we present common acquisition policies that rely on Gaussian process surrogates—which are the surrogate model of choice throughout this study—as well as one policy that is surrogate-free. Let the surrogate posterior predictive distribution at a candidate point $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ be

$$F(\boldsymbol{\theta}) \sim \mathcal{N}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta})\Big),$$

where $m(\boldsymbol{\theta})$ denotes the posterior mean and $v(\boldsymbol{\theta})$ the posterior variance, as introduced in the previous section.

Definition 5 *An acquisition policy π is defined as*

$$\pi(m, v) = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} a\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta})\Big), \tag{2.19}$$

where a is an acquisition function *encoding the exploration–exploitation trade-off at each Bayesian Optimization iteration.*

The acquisition function a is typically inexpensive to evaluate and can be maximized using multi-start gradient-based optimization techniques or, alternatively, through extensive random sampling. Most GP-based acquisition policies are distinguished by their choice of acquisition function a. Below, we list the main ones considered in this study.

Expected Improvement and Probability of Improvement. Two widely used acquisition policies in Bayesian optimization are Expected Improvement (EI), denoted π_{EI}, and Probability of Improvement (PI), denoted π_{PI}. To define these criteria, we first introduce the incumbent best observed value in the minimization setting,

$$y^\star \triangleq \min_{i\in\{1,\dots,j-1\}} y_i,$$

where j denotes the current BO iteration.

Definition 6 *The Expected Improvement (EI) acquisition function is defined as*

$$a_{\text{EI}}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta})\Big) = \underbrace{\Big((y^\star - \xi) - m(\boldsymbol{\theta})\Big)\Phi\Big(\frac{(y^\star - \xi) - m(\boldsymbol{\theta})}{\sqrt{v(\boldsymbol{\theta})}}\Big) + \sqrt{v(\boldsymbol{\theta})}\,\phi\Big(\frac{(y^\star - \xi) - m(\boldsymbol{\theta})}{\sqrt{v(\boldsymbol{\theta})}}\Big)}_{\textit{closed form}} \tag{2.20}$$

$$= \underbrace{\mathbb{E}\Big[\max\Big(0, (y^\star - \xi) - f(\boldsymbol{\theta})\Big)\Big]}_{\textit{integral form}}, \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{2.21}$$

where Φ and ϕ denote the cumulative distribution function and probability density function of the standard normal distribution, respectively. The parameter $\xi \geq 0$ is an exploration margin used to encourage exploration when desired.

The closed-form expression in (2.20) quantifies the expected amount by which a candidate evaluation $f(\boldsymbol{\theta})$, with $f(\boldsymbol{\theta}) \sim \mathcal{N}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta})\Big)$, improves upon the incumbent best

value $y^\star$, up to a margin ξ. This interpretation is more directly conveyed by the integral form in (2.21), which explicitly shows that when no improvement is expected, the expected improvement collapses to zero.

Remark 4 *EI takes large values either when the posterior mean is significantly better than the incumbent (exploitation) or when the posterior uncertainty is large (exploration). An important advantage of EI is that, in its basic form ($\xi = 0$), it is parameter-free while still automatically balancing exploration and exploitation. However, EI can become overly exploitative in the presence of high observation noise or when the surrogate becomes overconfident, and it may focus excessively near the current best solution if uncertainty shrinks too rapidly.*

Definition 7 *The Probability of Improvement (PI) acquisition function is defined, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, as*

$$a_{\mathrm{PI}}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta})\Big) = \underbrace{\Phi\Big(\frac{(y^\star - \xi) - m(\boldsymbol{\theta})}{\sqrt{v(\boldsymbol{\theta})}}\Big)}_{\textit{closed form}} \tag{2.22}$$

$$= \underbrace{\mathbb{P}(f(\boldsymbol{\theta}) \leq y^\star - \xi)}_{\textit{integral form}}. \tag{2.23}$$

Remark 5 *PI favors points that are likely to outperform the incumbent best value $y^\star$, regardless of the magnitude of the improvement. It is simple and inexpensive to compute compared to other acquisition functions in Bayesian Optimization. However, in the absence of a margin parameter ξ, PI is typically overly greedy and may prematurely focus on exploitation.*

Confidence-bound policies. Confidence-bound policies are also widely used, as they enable explicit control over the exploration–exploitation trade-off. These policies combine the surrogate mean and uncertainty in a linear manner.

Definition 8 *The Upper Confidence Bound (UCB) and the Lower Confidence Bound (LCB) acquisition functions are defined, analogously, as*

$$a_{\mathrm{UCB}}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta}), \lambda_j\Big) = \underbrace{-m(\boldsymbol{\theta})}_{\textit{exploitation term}} + \underbrace{\sqrt{\lambda_j}\sqrt{v(\boldsymbol{\theta})}}_{\textit{exploration term}}, \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{2.24}$$

$$a_{\mathrm{LCB}}\Big(m(\boldsymbol{\theta}), v(\boldsymbol{\theta}), \lambda_j\Big) = \underbrace{-m(\boldsymbol{\theta})}_{\textit{exploitation term}} - \underbrace{\sqrt{\lambda_j}\sqrt{v(\boldsymbol{\theta})}}_{\textit{exploration term}}, \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}. \tag{2.25}$$

Here $\lambda_j > 0$ is an exploration schedule that may evolve across iterations, hence the index j. For example, one may favor exploration during the early stages of Bayesian Optimization and progressively shift toward exploitation in later iterations when the total budget N is large. Conversely, when N is small, a lower value of λ_j may be preferred in order to emphasize exploitation.

Remark 6 *UCB-based acquisition policies favor points that jointly exhibit low predicted objective values (exploitation, due to minimization) and high predictive uncertainty (exploration). The relative importance of uncertainty is governed by the schedule λ_j: larger values promote exploration, while smaller values emphasize exploitation. Theoretical guarantees for UCB-type acquisition functions are well established in GP bandit settings (Srinivas et al., 2012 [94]), and optimal schedules admit closed-form expressions. However, computing these schedules in practice is demanding, and the choice of λ_j is often made heuristically.*

Remark 7 *In contrast to UCB, the LCB acquisition function assigns lower scores to candidates with high posterior uncertainty and therefore prioritizes points that are predicted to be good and sufficiently certain. As a result, LCB behaves more exploitatively than UCB, which may be beneficial in low-noise settings or when the optimization budget is limited. However, if λ_j is chosen small, LCB may become overly greedy and risk premature convergence to suboptimal regions. Conversely, larger values of λ_j increase the influence of uncertainty targeting only certain regions.*

Tree-structured Parzen Estimator (TPE). The Tree-structured Parzen Estimator (TPE) is a widely used alternative for hyperparameter optimization, particularly well suited to conditional search spaces, where some hyperparameters must be selected before others, and to mixed-variable domains involving both discrete and continuous hyperparameters. Unlike the acquisition policies introduced earlier, which rely on an explicit surrogate model of the black-box objective, TPE instead models the distribution of hyperparameters directly (Bergstra et al., 2011 [90]).

The TPE approach partitions the dataset $\mathcal{D} = \{\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i)\}_{i=1}^{j-1}$ into a *good* population $\mathcal{D}^g$ and a *bad* population $\mathcal{D}^b$ using a quantile threshold $\gamma \in (0, 1)$. The good population contains the γL configurations with the lowest objective values, while the bad population contains the remaining $(1-\gamma)L$ configurations, where $L = |\mathcal{D}|$.

Definition 9 *The TPE acquisition function is defined, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, as*

$$a_{\text{TPE}}(\boldsymbol{\theta}) = \frac{p\Big(S = \boldsymbol{\theta} | S \in \mathcal{D}^g\Big)\Big\}\ \textit{good population}}{p\Big(S = \boldsymbol{\theta} | S \in \mathcal{D}^b\Big)\Big\}\ \textit{bad population}} \tag{2.26}$$

Here, the random variable S represents the next candidate hyperparameter configuration $\boldsymbol{\theta}^{\text{can}}$. The probability density associated with the good population is estimated as

$$p(S = \boldsymbol{\theta} | S \in \mathcal{D}^g) = \frac{1}{N_g} \sum_{\boldsymbol{\theta}_i \in \mathcal{D}^g} k_g(S, \boldsymbol{\theta}_i), \tag{2.27}$$

where $k_g(S, \boldsymbol{\theta}_i)$ is typically chosen as a Gaussian kernel and $N_g = |\mathcal{D}^g|$. The density $p(S \in \mathcal{D}^b)$ is defined analogously using the bad population.

It has been shown that maximizing this ratio is equivalent to a Probability of Improvement criterion under the TPE modeling assumptions (Garnett, 2023b [95]; Song et al., 2022 [96]).

Remark 8 *The TPE acquisition function is computationally cheaper than the classical PI defined in Definition 2.23, as it avoids repeated evaluation of a global predictive distribution (Watanabe, 2023 [91]). TPE proposes configurations that are statistically similar to previously successful configurations while being dissimilar to poor ones. It scales well in many practical hyperparameter optimization problems, naturally accommodates conditional parameters, and avoids reliance on Gaussian predictive assumptions. Uncertainty quantification is indirect and arises from density estimation and the population split. The exploration–exploitation trade-off is controlled through design choices such as the quantile γ, kernel bandwidths, and the parametrization of the density models (Bergstra et al., 2011 [90]; Watanabe, 2023 [91]).*

To summarize this section, at each Bayesian optimization iteration a surrogate model is constructed from the current dataset. This surrogate is then coupled with an acquisition policy that encodes the desired exploration–exploitation trade-off in order to select the next hyperparameter configuration to be evaluated on the black-box objective. The resulting observation enriches the dataset $\mathcal{D}$ with a new hyperparameter–value pair, thereby completing one iteration of the Bayesian optimization loop.

This section presented the main acquisition functions considered in this study, namely:

1. Expected Improvement (EI) and Probability of Improvement (PI), which are based on the notion of improvement and rely on a Gaussian process surrogate,
2. Upper and Lower Confidence Bounds (UCB and LCB), which also rely on a Gaussian process surrogate and explicitly trade off predictive mean and uncertainty,
3. The Tree-structured Parzen Estimator (TPE), which adopts a density-based and surrogate-free formulation.

Figure 2.4 provides a qualitative illustration of the behavior of the acquisition functions considered in this study. By coupling a common surrogate model with different acquisition strategies, the figure highlights how distinct formulations induce markedly different exploration-exploitation behaviors, leading to different candidate selections despite identical posterior information.

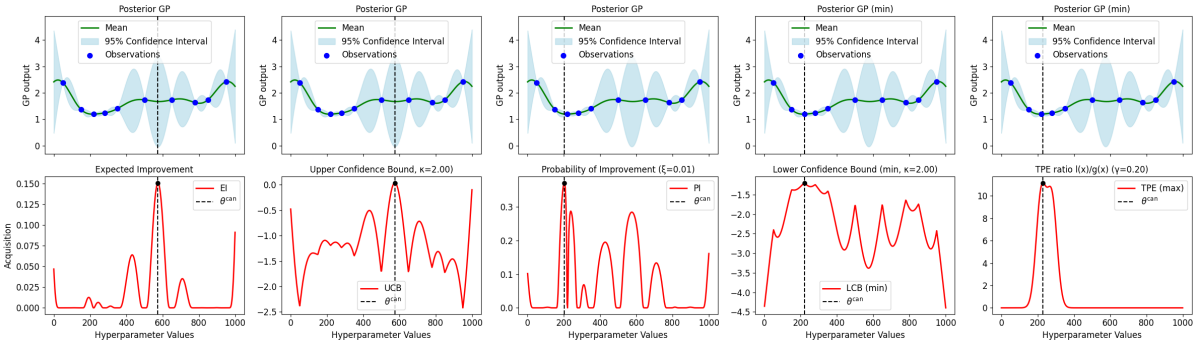


Figure 2.4 – Qualitative comparison of common acquisition functions. The top row shows the same Gaussian process posterior, including the predictive mean, uncertainty bands, and observed data, while the bottom row displays the corresponding acquisition functions. From left to right, EI concentrates mass near regions that jointly promise improvement and uncertainty, UCB favors uncertain regions according to the exploration schedule, PI focuses on regions with high likelihood of outperforming the incumbent, LCB emphasizes conservative exploitation of low-mean and low-uncertainty regions, and the TPE highlights regions that are statistically similar to previously successful configurations. In all cases, the dashed vertical line indicates the selected candidate hyperparameter $\boldsymbol{\theta}^{\text{can}}$ for the next evaluation.

2.4 Conclusion

In this chapter, we presented an overview of black-box optimization techniques, including derivative-free optimization and evolutionary methods. We compared these approaches to BO and argued that BO is well suited to the receiver tuning objective defined in Chapter 1, since it is designed for expensive evaluations and aims to reach good objective values

with few function calls. We also detailed the main components of BO, including the GP surrogate model used throughout this manuscript and the acquisition policies that drive the exploration–exploitation trade-off.

The remainder of this manuscript explores how BO can be used to automate the tuning of receiver hyperparameters.

Chapter 3

Search Method

3.1 Introduction

3.1.1 Challenge overview

Without precise and extensive prior knowledge of the operational conditions, whether real or simulated, under which data were transmitted, identifying receiver hyperparameters that decode the data reliably is akin to finding a needle in a haystack.

This chapter introduces an automation of this process through the *Search method*, which targets settings where the operational conditions can be treated as fixed over a tuning session, so that several BO iterations can be run to identify high-performing hyperparameters.

Definition 10 *The **Search method** is the process of identifying hyperparameter values under specific operational conditions by running BO for multiple iterations until an iteration budget is exhausted.*

This method corresponds to several practical scenarios. One may have recordings of received data that were not successfully decoded, and the goal is to recover the transmitted data by finding hyperparameters that better match the operational conditions during which the recordings were made. Another scenario is to use recordings collected at a given location and time, for example sounded channels (Walree et al., 2017 [34]; Socheleau et al., 2011 [97]), for replay, in order to identify hyperparameters expected to perform well in future operations at the same site under limited drift. The search method can also be used under controlled operational conditions when prior knowledge is available, for example by relying on high-fidelity simulators (Socheleau et al., 2010 [66]; Porter, 2011 [98]; Walree, 2011 [7]).

Throughout this manuscript, the developed algorithms are built upon BO entities referred to as agents.

Definition 11 *An agent consists of a BO algorithm, with its own surrogate and policy. An agent only interacts with the black-box f to infer performance for specific hyperparameter values.*

3.1.2 Benchmarks

To the best of our knowledge, there is no fully agnostic approach in the literature, in the sense of requiring no prior knowledge, for automating receiver hyperparameter tuning against which the developed algorithms can be systematically benchmarked.

In practice, tuning is often performed manually using experience and any available knowledge about operational conditions. Other approaches include brute-force grid sampling and random sampling, and more sophisticated heuristics such as channel-estimation-based rules (Brossier et al., 2002a [99]; Brossier et al., 2002b [100]; Geller et al., 1996 [101]; Gerzaguet et al., 2018 [102]; Shu et al., 2015 [103]; El Husseini et al., 2019a [104]; Shu et al., 2014 [105]). Consequently, the methods developed in this chapter are benchmarked against reference baselines.

The first benchmark is a filter size-estimation-based method:

Channel-estimation-based benchmark

This benchmark uses the dynamic channel-estimation-based approach presented in (Amis et al., 2007 [106]). In this approach, filter lengths (L_A for feedforward and L_B for feedback) are computed from preamble-based channel estimates. Specifically, the maximum detected delay spread τ_m, measured between the strongest and weakest detected paths, is used to define

$$\hat{L}_A = \max\Big(80, 2\tau_m \cdot n_s \cdot s_r + 1\Big), \tag{3.1}$$

$$\hat{L}_B = \max\Big(20, \tau_m \cdot s_r + 1\Big), \tag{3.2}$$

where $s_r = \frac{B_L}{1+c_{\text{off}}}$ denotes the symbol rate, with B_L the bandwidth and c_{off} the roll-off factor. Here, n_s represents the number of samples per symbol at the input of the DFE.

The criteria (3.1) and (3.2) ensure at least 80 feedforward taps and 20 feedback taps. The feedforward filter length is set to twice the detected delay spread to capture multi-path arrivals, while the feedback length is chosen to cover the observed delay spread. The

performance of channel-estimation-based methods depends heavily on the accuracy of channel estimates, which in turn depends strongly on the preamble length. If the preamble is not sufficiently long, the delay spread τ_m will be underestimated, leading to poor estimates of $\hat{L}_A$ and $\hat{L}_B$. Since UAC channels can exhibit extensive delay spreads, the preamble length itself is a crucial hyperparameter.

The second benchmark is random sampling:

Random sampling benchmark

Within this benchmark, at each iteration, a hyperparameter vector is drawn uniformly from the search space and tested on the receiver.

Grid sampling is discarded as a benchmark, since its complexity and performance depend on the grid resolution. It is instead used for domain exploration, as featured in previous and subsequent objective function representations.

In this chapter, BO is applied to UAC receiver hyperparameter tuning through the Search method. For this UAC proof of concept, evaluation is conducted over simulated channels generated with the parametric Maximum Entropy model (Socheleau et al., 2010 [66]), as well as over a real channel recorded in freshwater at Loch Ness, Scotland, in November 2023 (Cheikh Melainine et al., 2024 [32]). An additional telecommunication example, namely Mud-Pulse Telemetry, is presented in Appendix 6, Section 6.1, where the search method is also applied.

3.2 Search method setup

The Search method is formalized as follows

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f(\boldsymbol{\theta}), \tag{3.3}$$

where f denotes the receiver performance when decoding replay data using hyperparameters $\boldsymbol{\theta}$ (see Section 1.4). In an offline scenario where the transmitted data are known, one could optimize directly over the BER rather than a proxy such as f. However, this assumption is not made in this chapter and we treat the transmitted data as unknown.

The optimization problem (3.3) is approached under the following assumption:

Assumption 2 *The objective function f is static, i.e., it does not change from one BO iteration to another.*

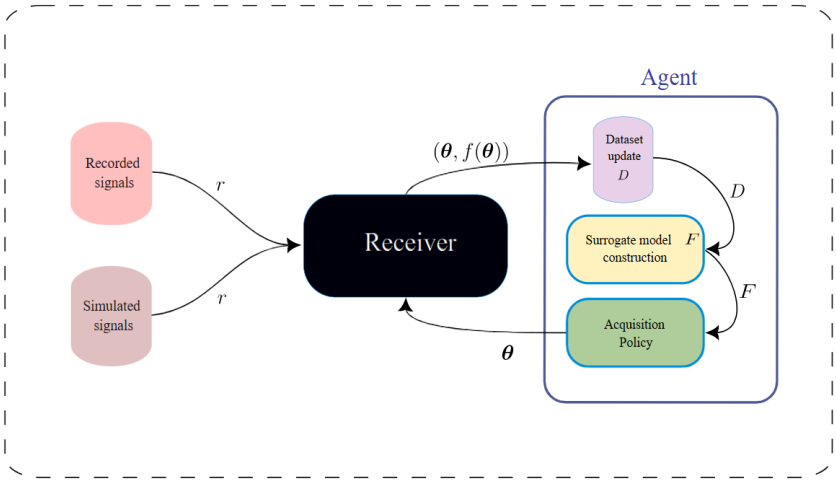


Figure 3.1 – Search method. The receiver processes data originating either from recordings of previous operations or from simulation. At each iteration, the receiver attempts to recover the data using hyperparameter values suggested by the agent.

This assumption holds when the objective is to identify the best hyperparameters for a given set of packets. In that case, the same set of packets is replayed at each iteration until the evaluation budget is exhausted, which is the setting considered in this chapter. For instance, when the goal is to recover important transmitted data that was not successfully decoded. If the packets change from one iteration to another, the objective function changes accordingly and the assumption no longer holds.

The hyperparameter vector considered in this chapter is

$$\boldsymbol{\theta} \triangleq [L_A, L_B, \beta, \mu_A, \mu_B]^T. \tag{3.4}$$

Here, μ_A, μ_B, and β are real-valued parameters, while the filter lengths L_A and L_B are integer-valued parameter (see Chapter 1, Section 1.3).

Throughout the proof-of-concept of the Search method, tuning is restricted to the hyperparameters in (3.4). The Doppler-bank matched-filter resolution is set to $\delta_{\text{DB}} = 0.1$ m/s and spans values between -5 et 5 m/s, in order to correctly detect the simulated Doppler scale corresponding to $V = 3$ m/s, which makes the PLL dispensable in this controlled setting. The importance of PLL hyperparameters and the receiver sensitivity to them are more appropriately highlighted in the dynamic scenarios considered in Chapter 4. No general consensus exists on techniques for systematically selecting optimal values, apart from exhaustive search. This chapter therefore presents a method to automate the tuning of UAC receiver hyperparameters without prior knowledge of the operating conditions.

This method is analogous to the classical use of BO in other domains, such as machine

Algorithm 1 Search method pseudo-algorithm

1: **Warm-up session**: Construct an initial dataset $\mathcal{D} = \{(\boldsymbol{\theta}_i, f(\boldsymbol{\theta}_i))\}_{i=1}^{n}$ by randomly selecting and evaluating n hyperparameter vectors $\boldsymbol{\theta}$ on the replay data r.
2: **Agent iterations**:
3: **while** iteration j is within the budget N **do**
4: 1) **Maximize the acquisition policy**: suggest $\boldsymbol{\theta} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} a(\boldsymbol{\theta})$, where a is defined by the chosen policy (e.g., TPE, EI, PI, or UCB) in Section 2.3.3.
5: 2) **Run receiver with $\boldsymbol{\theta}$**: run the receiver on the replay data and obtain the performance $f(\boldsymbol{\theta})$.
6: 3) **Augment the dataset**: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{\theta}, f(\boldsymbol{\theta}))\}$.
7: 4) **Fit the surrogate model**: fit F on $\mathcal{D}$ using (5.16) and (5.17), then update a accordingly.
8: **end while**

learning, where a black-box function is optimized through multiple BO iterations, as presented in Chapter 2. In the present use case, namely offline receiver hyperparameter tuning, the method is illustrated in Figure 3.1.

In Figure 3.1, an agent, as defined in Definition 11, interacts with the receiver by suggesting new hyperparameters and collecting the corresponding receiver performance over the received data r. Over a budget of N iterations, the agent maintains a dataset $\mathcal{D}$, which may be initialized using a small number of random evaluations, and fits a surrogate model F on $\mathcal{D}$. Once the surrogate is available, the acquisition policy proposes a new hyperparameter vector $\boldsymbol{\theta}$. The receiver is then run with $\boldsymbol{\theta}$ and returns the associated performance value, yielding an observation $(\boldsymbol{\theta}, f(\boldsymbol{\theta}))$ that is appended to $\mathcal{D}$.

The Search method algorithm is summarized in Algorithm 1. Note that the last step is not specific to PI, EI, and UCB policies.

3.3 Numerical results

Throughout the Search method, the goal is to identify hyperparameter values that are best suited to the available replay data. To assess its effectiveness, the method is benchmarked against random sampling (see Benchmark 3.1.2) and a channel-estimation-based benchmark (see Benchmark 3.1.2), which selects L_A and L_B. Throughout this section, the default hyperparameter values are denoted by $\boldsymbol{\theta}_{\text{default}}$ and are set to

$$L_A = 800, \quad L_B = 800, \quad \beta = 0.25, \quad \mu_A = 0.05, \quad \mu_B = 0.1. \tag{3.5}$$

These values are chosen as follows. The filter lengths are selected to cover all simulated delay spreads. The step-size parameters and the IPNLMS hyperparameter are chosen by

domain experts and are known to perform well on average across operational conditions. The default values are used for hyperparameters that are not estimated by the channel-estimation benchmark, namely μ_A, μ_B, and β when using this benchmark. They are also used for objective-function visualizations when fixing non-displayed dimensions.

Validation is performed on simulated channels generated using the Maximum Entropy model and on the Loch Ness sounded channel recorded in Scotland in November 2023(Cheikh Melainine et al., 2024 [32]).

The replay data consist of packets of 3375 symbols, corresponding to 1.2 s transmissions, sent with bandwidth $B_1 = 4.5$ kHz and with the remaining transmission settings reported in Table 1.1. For each packet, additive white Gaussian noise is included, with noise variance N_0 chosen to yield an in-band input SNR equal to the value specified for each experiment. Unless otherwise specified, the input SNR is fixed to 15 dB.

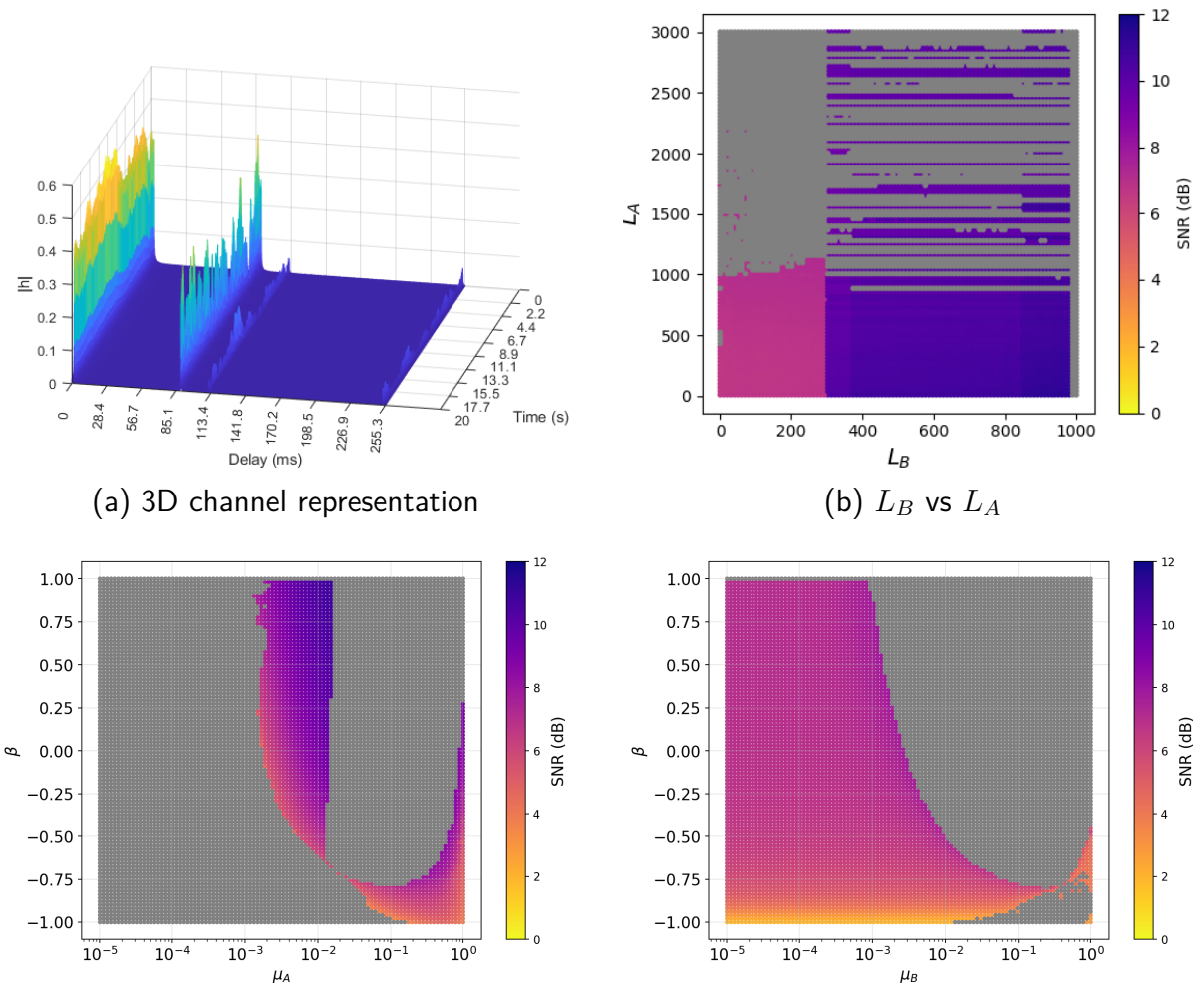
The objective is to discover the best hyperparameters for each transmitted packet. The same packet is therefore replayed over multiple iterations until the evaluation budget is exhausted. Each agent uses 5 warm-up evaluations followed by 45 BO iterations.

3.3.1 Performance on simulated channels

Two experiments are conducted in this section. The first experiment considers two simulated channels and aims to provide a convergence analysis of the Search method. The second experiment provides an extensive evaluation of the Search method, benchmarked against the proposed baselines, over 12 different simulated channels and 12,901 packets. This experiment highlights the empirical performance of the method across a wide range of operational conditions. Channels are simulated using the parameter ranges in Table 3.1. This simulation setup spans various operational conditions, ranging from short- to long-range communications and from relatively static to highly dynamic scenarios.

Configuration	***Minimum***	***Maximum***
Delay spread	6.6 ms	180 ms
Doppler spread	0.46 Hz	1.87 Hz
Water depth	100 m	200 m
Transmission range	500 m	1500 m
Rice factor	5	75

Table 3.1 – Simulated channel configurations. Doppler and delay spreads are computed using a -20 dB threshold on the Doppler power spectrum and the power-delay profile, respectively.



(a) 3D channel representation

(b) L_B vs L_A

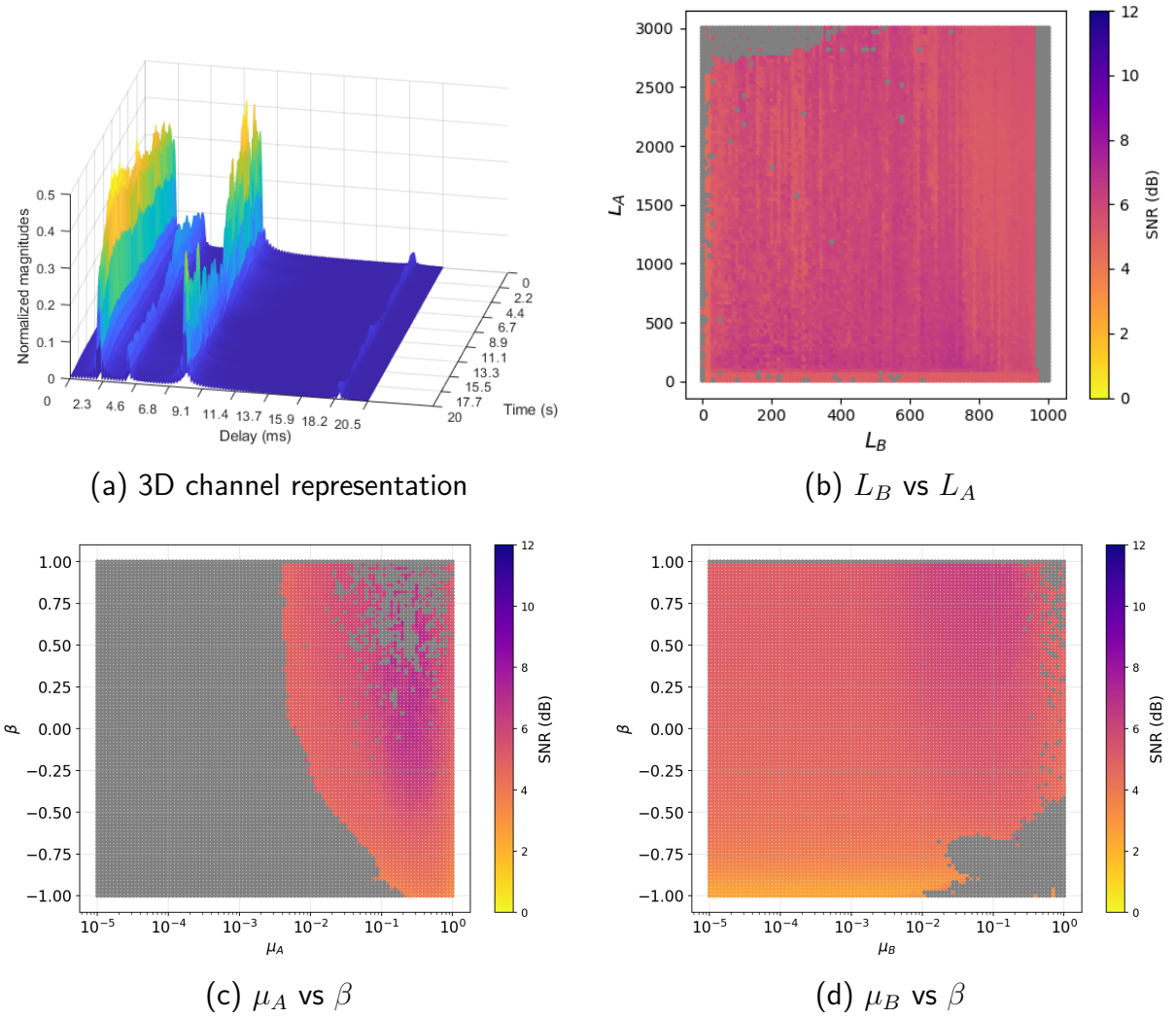
(c) μ_A vs β

(d) μ_B vs β

Figure 3.2 – Simulated channel 1 with delay spread 110.5ms and Doppler spread 0.78Hz. The input SNR is $\rho_{\text{IN}} = 15$ dB (1.27)

It should be noted that this is a five-dimensional problem and visualizing the objective function is challenging. The approach taken here is to visualize two-dimensional sections of the objective, while fixing the remaining hyperparameters to their default values. This only provides a partial representation of the objective function, and the coupling between hyperparameters can be significantly more complex than what these sections suggest.

Regarding the first experiment, Figures 3.2 and 3.3, two simulated channels with different delay spreads are displayed, with compensated Doppler scale. The high-performance regions are narrow and depend strongly on the delay and Doppler spreads. For example, for channel 1, Figure 3.2b shows substantial roughness for high values of L_A due to frequent hallucinations. In addition, good performance is observed only for values of L_B



(a) 3D channel representation

(b) L_B vs L_A

(c) μ_A vs β

(d) μ_B vs β

Figure 3.3 – Simulated channel 2 with delay spread 32ms and Doppler spread 1.84Hz. The input SNR is $\rho_{\text{IN}} = 15$ dB (1.27)

greater than 320 taps (≈ 95 ms), which is a direct consequence of the large delay spread. In contrast, channel 2 differs markedly in Figure 3.3b, where the objective is relatively flat and sensitivity to filter lengths is limited. Small performance jumps are observed around $L_A \geq 80$ and $L_B \geq 20$, which is consistent with the smaller delay spread compared to channel 1. The Doppler spread for channel 1 is lower than that of channel 2. This is reflected in the sensitivity with respect to the adaptation hyperparameters μ_A, μ_B, and β. As Doppler spread increases, channel variability increases, and larger step sizes are required. This is observed in Figure 3.3c, where the Doppler spread is 1.89 Hz and the preferred values of μ_A increase toward approximately 0.5, while the associated β values in that region are smaller and can become negative. This behavior contrasts with cases where μ_A is not chosen optimally for this channel, such as the default value $\mu_A = 0.05$. In

that case, sensitivity with respect to β is reduced and negative values become more prone to hallucinations.

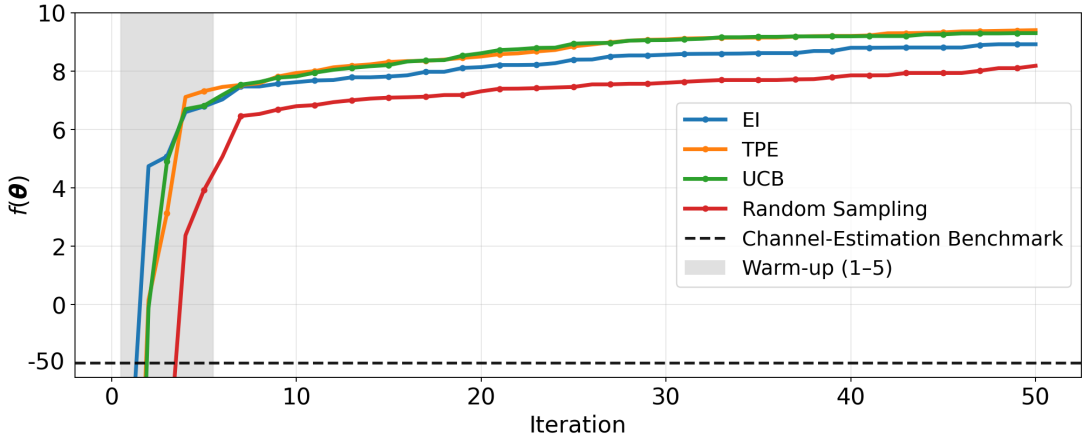
For Channel 1, where the Doppler spread is 0.78 Hz (significantly lower than channel 2), higher values of μ_A and μ_B are more prone to hallucinations, as shown in Figs. 3.2c and 3.2d. The sparsity of Channel 1 is also visible in Figure 3.2a, where two main paths are clearly observed, with a secondary arrival around 90 ms and few intermediate echoes. Such sparse channels benefit from sparse treatments, and higher values of β allocate proportionally larger updates to high-energy taps in the DFE filters, as discussed in Section 1.3 of Chapter 1. Accordingly, higher values of β are favored in Figs. 3.2c and 3.2d.

In Figure 3.4, the average over 30 packets of the best performance achieved up to each iteration is shown for three BO agents, each using a different policy, namely EI, UCB, or TPE (see Section 2.3.3 in Chapter 2). The random-sampling benchmark is shown in red, while the channel-estimation benchmark is indicated by a black dashed horizontal line. In both panels, BO agents outperform both benchmarks, regardless of the policy used. The objective values correspond to SNR values in dB and are penalized when hallucinations occur.

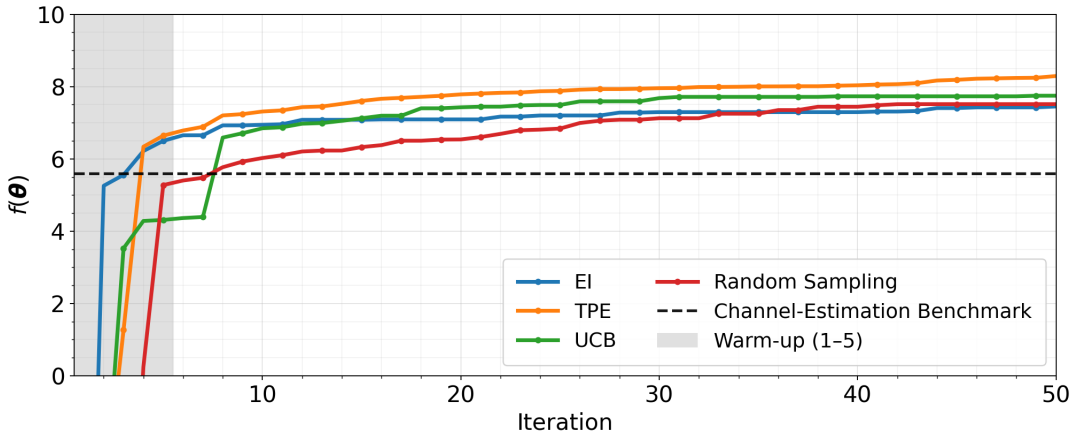
In Figure 3.4a, the Channel-estimation benchmark yields only hallucination-penalized values. This mainly comes from an overestimation of the feedforward length. The benchmark sets the feedforward filter to cover twice the measured delay spread, a reasonable rule of thumb, but here it leads to an excessively large L_A. The preamble length is 511 symbols (Table 1.1), which is about 151.4 ms and should be sufficient to estimate the delay spread. For this channel, the benchmark selects $L_A = 1320$ and $L_B = 330$. These values frequently lie in hallucination-prone regions (Figure 3.2), which explains the low performance.

For Channel 2 (Figure 3.4b), the channel-estimation benchmark yields $L_A = 680$ and $L_B = 170$, which avoids hallucination-prone regions for the remaining hyperparameters. However, its performance remains below that achieved by the Search method.

For both channels, random sampling remains relatively close to the Search method, with a maximum performance gap of about 2 dB. This gap can be significant when translated to BER.



(a) Channel 1.



(b) Channel 2.

Figure 3.4 – Best objective value (higher is better) versus iteration for two simulated channels: (a) Channel 1 (Figure 3.2) and (b) Channel 2 (Figure 3.3). Curves show agents using EI (in blue), UCB (in green), and TPE (in orange). The random-sampling benchmark is shown in red and the channel-estimation benchmark is shown as a black dashed line. Each curve is averaged over 30 packets. The input SNR $\rho_{\text{IN}} = 15$ dB.

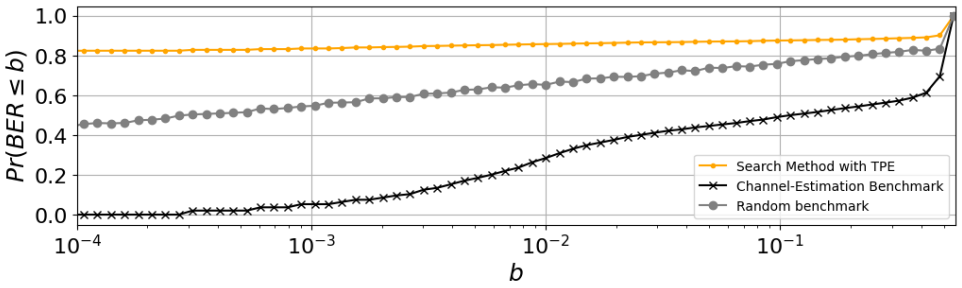


Figure 3.5 – Estimated CDF of BER for the channel-estimation benchmark (black), random sampling (gray), and the Search method using a TPE agent (orange).

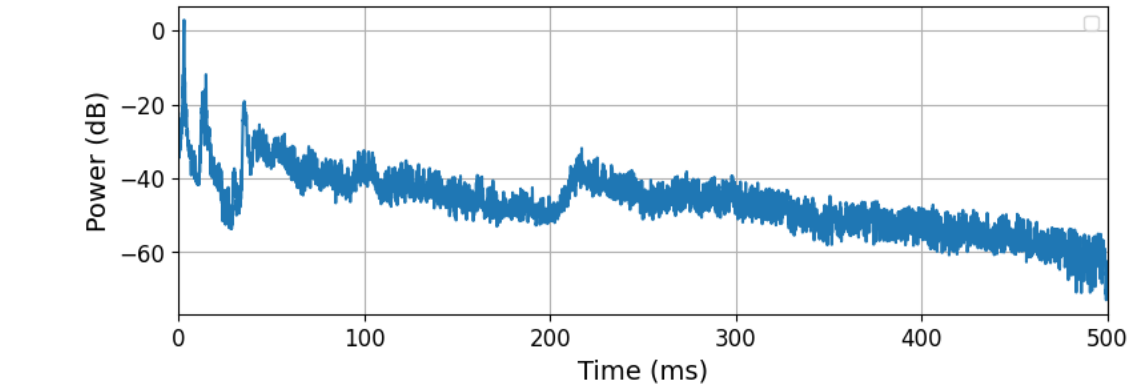
The second experiment presents performance in BER. Indeed, if transmitted bits are available, BER provides a direct way to evaluate whether hyperparameters are truly adapted to a given channel, for example when reporting hyperparameter values for specific operational conditions. However, in the present setting, the objective is to recover data from replay, and the only available metric is the proxy objective f defined in Section 1.4 and (1.29). The same optimization setup described above is therefore used. For this experiment, we restrict attention to TPE agents for simplicity, since the different agents yield similar results and TPE performs best in our tests.

Figure 3.5 shows an estimate of the cumulative distribution function (CDF) of the BER for the Channel-estimation benchmark (black), Random sampling (red), and an agent using TPE (orange).

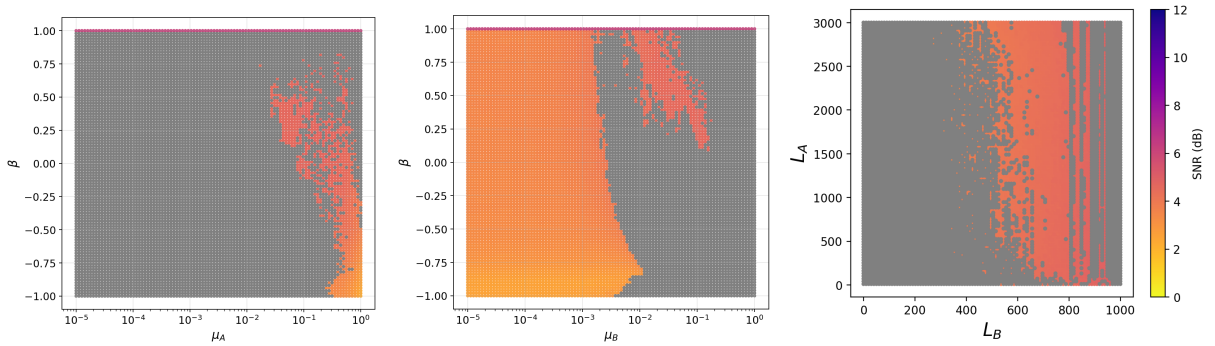
To generate this figure, whenever the channel-estimation benchmark fails to fully recover a packet, an optimization is triggered to identify the best $\boldsymbol{\theta}$ for that packet. The input SNR lies in the range $[10, 15]$ dB. Overall, 70% of the packets are successfully recovered, and the average MSE improvement relative to the channel-estimation benchmark is 3 dB. Figure 3.5 shows that the Search method yields improved BER compared with both the Channel-estimation and Random sampling benchmarks. Although the gains in the performance metric f (the output SNR-based black-box defined in Section 1.4) are modest, differences of a few dB can translate into substantial BER reductions, since BER typically decreases exponentially as the input SNR increases (1.27).

3.3.2 Performance on real channel

The power-delay profile depicted in Figure 3.6a was measured in freshwater at Loch Ness, Scotland, in November 2023. However, the temporal variability of the channel was not fully characterized. To address this limitation, the power-delay profile and the estimated Rice factors for each tap in Figure 3.6a were used as inputs to the MAXENT



(a) Power-delay profile

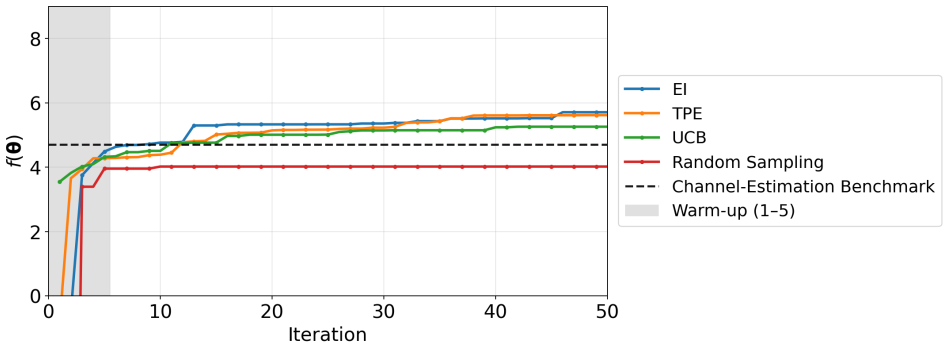


(b) μ_A vs β (c) μ_B vs β (d) L_B vs L_A

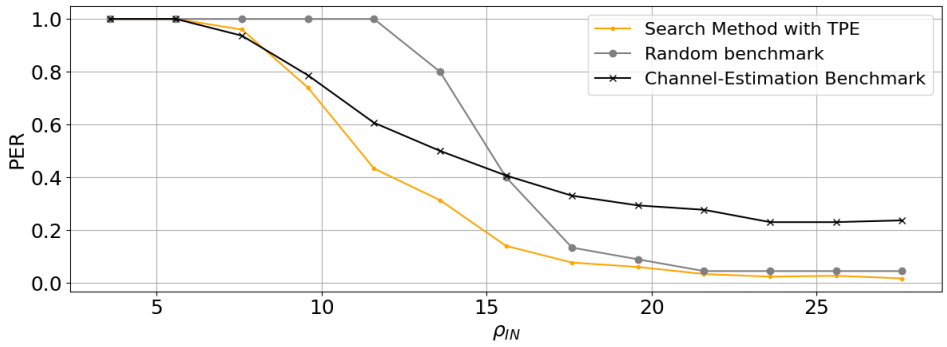
Figure 3.6 – Loch Ness measurements and objective maps with delay spread 12.7 ms and Doppler spread 0.8 Hz. The input SNR $\rho_{\text{IN}} = 15$ dB. © 2024 IEEE.

simulator. A constant Doppler spread of 0.8 Hz was then added, yielding a hybrid channel. Two experiments are conducted. The first experiment aims to inspect the Loch Ness channel and the associated objective function, as well as to provide a convergence analysis against the benchmarks. The second experiment aims to highlight the performance of the Search method through extensive testing over 300 packets at different input SNR values. The transmission settings in Table 1.1 where adopted for this section.

In Figure 3.6, the channel exhibits a long reverberation tail, which requires long filters, as illustrated in Figure 3.6d. In addition, as highlighted by the IPNLMS update equations (1.23) and (1.24), the updates are scaled by the step sizes but also divided by the filter lengths. This explains the need for larger step sizes in Figures 3.6b and 3.6c. The channel is highly dispersive beyond a delay of 40 ms, while three dominant paths can be clearly distinguished before that delay. This constitutes an edge scenario. If the update algorithm attempts to isolate the dominant paths, namely the three paths before 40 ms, through a sparse channel representation corresponding to large β, good performance can



(a) Best objective values obtained during optimization, with input SNR $\rho_{\mathrm{IN}} = 15$ dB.



(b) PER versus input SNR ρ_{IN} (1.27).

Figure 3.7 – Loch Ness: performance and optimization behavior.

be achieved at the expense of an increased risk of hallucinations due to the neglect of the reverberation tail, as shown in Figures 3.6b and 3.6c. As highlighted in Section 1.4 of Chapter 1, an abrupt arrival of new paths may induce unrecoverable decision errors and therefore hallucinations. Alternatively, increased stability can be achieved by adopting a more dispersive assumption for the channel, corresponding to smaller β, at the expense of reduced performance. Overall, this channel is challenging for the considered receiver architecture, in particular due to the high prevalence of hallucinations and the limited extent of non-hallucinating regions.

Similarly to the previous section, Figure 3.7a reports the best performance achieved up to each iteration, averaged over 30 packets. The Channel-Estimation benchmark underestimates the delay spread and selects $L_A = 80$ and $L_B = 20$. This choice yields mostly hallucinations and, depending on the noise realization, can also land in low-performing combinations with $f(\boldsymbol{\theta}) \approx 4$ dB. Random Sampling reaches a performance plateau around 4 dB, mainly because the high-performance region in Figure 3.6 is narrow. The BO agents outperform both benchmarks and achieve values comparable to those obtained in the high-performance region.

The second experiment, shown in Figure 3.7b, presents the evolution of packet error rate for different input SNR values, with the Search method using a TPE policy benchmarked against the baselines. The results show a clear outperformance of the proposed method, with gains of up to 20% at high input SNR ρ_{IN} compared to the Channel-Estimation benchmark. Random Sampling performs worst at low input SNR, with $PER = 1$ for $\rho_{\text{IN}} \leq 12$ dB. A main reason is that the Loch Ness channel exhibits a high fraction of hallucinations (see Figure 3.6), and the hallucination-prone region of the search space expands as the noise level increases, that is as the input SNR decreases. As ρ_{IN} increases, the objective becomes smoother and Random Sampling improves. Its PER decreases and matches the Channel-Estimation benchmark for $\rho_{\text{IN}} = 15.5$ dB, then for $\rho_{\text{IN}} \geq 19.5$ dB it reaches the performance of the Search method and outperforms the Channel-Estimation benchmark. Search method performs best throughout all tested input SNR values.

3.4 Conclusion

This chapter introduced the *Search method* for identifying receiver hyperparameters best suited to recorded channels, replay data, or simulations. The method is implemented through a Bayesian optimization entity, referred to as an agent, equipped with its own surrogate model and policy. This constitutes a first step toward assessing the potential of Bayesian optimization for tuning telecommunication receivers, with a particular focus on UAC.

Although the proof of concept developed in this chapter targets a UAC receiver, the method can in principle be extended to other hyperparameters within the same receiver architecture, to other receiver architectures, and even beyond UAC, as illustrated by the Mud-Pulse Telemetry results in Appendix 6, Section 6.1. To the best of our knowledge, this is the first systematic of hyperparameter search for telecommunication signal-processing algorithms beyond brute-force grid sampling and Random sampling, both of which are markedly less efficient than the proposed approach. The method can therefore help practitioners automate receiver tuning, a task that is often time-consuming and tedious. It can also reduce the need for costly prior knowledge of the operating conditions, which is otherwise often gathered in advance to prepare field operations.

The *Search method* was benchmarked against Random sampling and a Channel-estimation-based benchmarks on both simulated and real channels. It outperformed both in terms of PER and BER. Even when the gain in the objective function value was modest, it still translated into meaningful reductions in PER and BER.

In its present form, the *Search method* is better suited to offline use cases, such as the recovery of high-importance packets when recordings are available, or the preparation of

receiver configurations before operations using past recordings or high-fidelity simulators. In the recovery scenario, the objective function derived in Chapter 1, Section 1.4, is the most appropriate and, to the best of our knowledge, the only one currently available in the literature for this purpose. In the preparation scenario, BER should instead be optimized whenever it is accessible.

The method nevertheless has limitations that should be addressed in future studies. First, the Bayesian optimization surrogates and policies used here are generally considered capable of tuning up to about 20 hyperparameters jointly, with some studies, for instance using TPE, pushing this limit to around 50 (Watanabe, 2023 [91]). Whether such limits remain realistic for telecommunication receivers should be assessed. We also stress that choosing which hyperparameters to tune is not a trivial task. The reader should therefore carefully assess the sensitivity of the performance metric to the selected hyperparameters, as some were found to have little impact in many scenarios.

A second limitation is that the developed method requires several iterations before convergence. As each iteration corresponds to a full receiver run on a packet, it is not directly suitable for adaptation at packet-rate. This motivates a different formulation of the problem, which is addressed in Chapter 4.

Overall, the *Search method* constitutes an original contribution of this work in the context of telecommunication receiver tuning. To the best of our knowledge, prior approaches relied on Grid search, Random sampling, manual tuning, Channel-estimation-based tuning and other heuristic strategies.

Chapter 4

Tracking Method

4.1 Introduction

4.1.1 Challenge overview

Chapter 3 showed that Bayesian optimization, through the Search method, can identify receiver hyperparameters that improve decoding performance under fixed operational conditions, using replay data or simulations. However, the Search method is not suited to operational settings where the channel evolves during transmission and receiver hyperparameters must be updated online. In UAC, short-term environmental variations are pervasive (Chapter 1, Section 1.2.2). Internal waves, ocean swell, and wind-driven surface roughness can modify propagation conditions on time scales ranging from fractions of a decisecond to minutes, and these changes are reflected in the performance metric f (Chapter 1, Section 1.4). In this regime, receiver tuning becomes harder because the optimal hyperparameters drift over time.

This motivates the *Tracking method*, whose objective is to maintain reliable decoding over an entire transmission under non-stationary channel conditions by updating hyperparameters during operation with acceptable complexity. The goal is to enable receiver fine-tuning that is robust to changing conditions and compatible with real-time constraints.

Definition 12 *The **Tracking method** is the process through which hyperparameter values are updated during operations, potentially at packet rate, using BO.*

The need for tracking is tied to the short-term variability of the subsea channel. These variations manifest in the received signals through delay spread, driven by multiple reflectors such as the sea surface and seabed, and Doppler spread, driven mainly by relative

motion and time-varying surface interactions (Jarrot et al., 2021 [9]; Unal et al., 2023 [107]; Cuji et al., 2024c [108]; Imbert et al., 2019 [109]). Both spreads depend on operational conditions and can be non-stationary during communication. For example, changes in relative velocity can alter Doppler spreading, while changes in range and boundary geometry can modify the delay profile. These time-varying distortions are a primary source of performance degradation (Walree, 2011 [7]).

4.1.2 Search or Tracking method

A direct extension of the Search method to packet-based online adaptation is computationally problematic. Search typically requires multiple objective-function evaluations to converge, which translates into multiple receiver runs. Running such an optimization for each packet is prohibitive, and triggering it only intermittently can yield hyperparameters that are tuned to a small subset of packets while conditions continue to evolve.

In addition to this complexity issue, one may be tempted to use the Search method to preset a single hyperparameter vector prior to deployment. This is operationally appealing, but it implicitly assumes that the optimizer is approximately invariant across environments, as stated in Assumption 2. The next theoretical comparison highlights why this choice can be suboptimal when operational conditions vary, by contrasting an idealized Search procedure that learns a single hyperparameter vector from replay data aggregated across contexts with a perfect Tracking method that selects hyperparameters separately for each context.

Let c denote an operational context and let $f^{(c)}$ be the corresponding receiver performance function, viewed as a stochastic process, that is the collection of random variables $\{f^{(c)}(\boldsymbol{\theta})\}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}$. Define the cross-context mean performance as

$$\mu(\boldsymbol{\theta}) \triangleq \mathbb{E}_c\Big[f^{(c)}(\boldsymbol{\theta})\Big]. \tag{4.1}$$

Assuming replay data from all possible contexts, the idealized infinite-context Search-method choice is

$$\boldsymbol{\theta}_{\mathrm{SM}} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mu(\boldsymbol{\theta}). \tag{4.2}$$

In contrast, within a given context c, a perfect Tracking method would instead choose

$$\boldsymbol{\theta}^{(c)}_{\mathrm{TM}} = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f^{(c)}(\boldsymbol{\theta}), \tag{4.3}$$

which is context dependent.

First, by Jensen's inequality (Jensen, 1906 [110]) and the convexity of $\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}}(\cdot) : \mathbb{R}^M \to \mathbb{R}$, it follows that

$$\mu(\boldsymbol{\theta}_{\mathrm{SM}}) = \max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_c\Big[f^{(c)}(\boldsymbol{\theta})\Big] \le \mathbb{E}_c\Big[\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f^{(c)}(\boldsymbol{\theta})\Big] = \mathbb{E}_c\Big[f^{(c)}\Big(\boldsymbol{\theta}^{(c)}_{\mathrm{TM}}\Big)\Big]. \tag{4.4}$$

Equation (4.4) states that, on average over contexts, a perfect Tracking method yields performance at least as high as the idealized Search method's choice. The inequality is typically strict when the identity of the maximizer varies with c, meaning that different operational conditions favor different hyperparameter settings.

The goal of this chapter is to develop a Tracking method able to fine-tune receiver hyperparameters at packet rate. To this end, the Tracking problem is described and formalized in Section 4.2. Section 4.4 then introduces a simplified single-agent approach operating in flywheel mode with a static surrogate model. Section 4.5 presents a multi-agent Tracking method that uses a time-varying surrogate model to handle rapidly changing dynamics and two custom policies designed to be more robust to hallucinations. Since the objective functions in the previous chapter showed limited performance gains for $L_A > 1000$, we restrict the search range to $L_A \in [80, 1000]$ throughout this chapter.

4.2 Tracking method setup

In packet-based communication, for each received packet p_t in a communication link, the aim is to determine the hyperparameters that maximize receiver performance under the channel conditions at time t:

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} f_t(\boldsymbol{\theta}). \tag{4.5}$$

Here, $\boldsymbol{\theta}_t^*$ denotes the optimal hyperparameter vector at time t, where t is an integer indexing packet p_t, and $\boldsymbol{\Theta}$ denotes the search space of hyperparameter vectors. The objective function f_t evaluates receiver performance as a function of $\boldsymbol{\theta}$. In contrast with Assumption 2 used for the Search method, the following assumption is made:

Assumption 3 *The objective function f is time-varying, meaning that it can change from one received packet to the next. Changes occurring at a rate faster than the packet rate are assumed to be untrackable.*

For each received packet p_t, problem (4.5) is addressed using Bayesian optimization, and Assumption 3 reflects the impact of dynamic subsea conditions, which introduce time-varying distortions in the received signal.

To track these variations, this chapter introduces a Tracking method technique for Bayesian optimization of receiver performance based on a flywheel mode. The proposed online tuning algorithm performs only one objective-function evaluation per received packet, which is consistent with fine-tuning at packet rate. The Tracking method minimizes overhead while continuously improving performance as packets arrive. The overall procedure is illustrated in Figure 4.1.

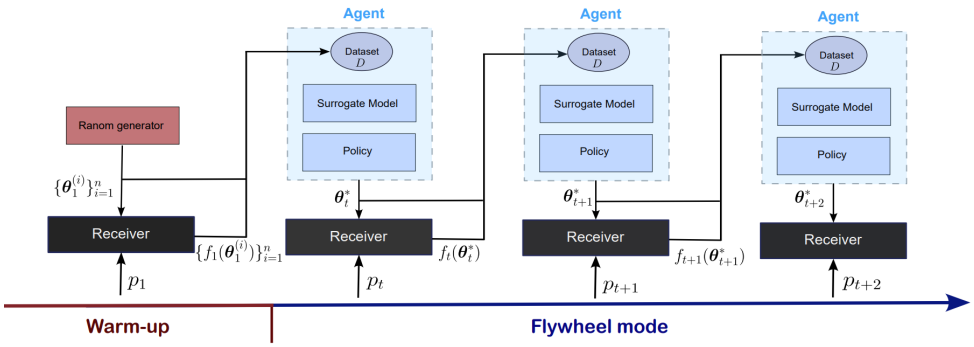


Figure 4.1 – Flowchart of the online tuning process.

The implementation begins with a warm-up session, during which n hyperparameter vectors $\{\boldsymbol{\theta}_1^{(i)}\}_{i=1}^n$ are randomly sampled and evaluated on the first received packet p_1 to construct an initial dataset $\mathcal{D} = \{(\boldsymbol{\theta}_1^{(i)}, f_1(\boldsymbol{\theta}_1^{(i)}))\}_{i=1}^n$. This phase provides baseline information for subsequent optimization. After warm-up, each new packet p_t received at time t enters flywheel mode, where the following steps are executed:

1. Update the surrogate model using the current dataset $\mathcal{D}$.

2. Select the next hyperparameter vector using the acquisition policy, yielding a single vector $\boldsymbol{\theta}_t$ for the current packet p_t.

This one-shot strategy accumulates knowledge over time while avoiding the computational expense of multiple evaluations per packet.

Throughout this chapter, the Tracking method is benchmarked against the same baselines as Chapter 3, namely Random Sampling and the Channel-Estimation benchmark. Since the Channel-Estimation benchmark depends on the accuracy of the estimated delay spread, and this accuracy is strongly affected by the preamble length, we also consider a third baseline corresponding to the case where the delay spread is known.

Known delay spread benchmark

Within this benchmark, the Channel-Estimation baseline is used with a preamble length chosen to match the true delay spread τ_m.

This benchmark represents an idealized best-case scenario for channel-estimation-based methods, since the true delay spread cannot be known a priori.

4.3 Test scenario

Testing the realtime adapation method requires a time-varying channel that is sufficiently long to assess whether the method can converge toward hyperparameter values that remain adapted as conditions evolve. Conducting a sea trial is expensive, and available open-access sounded datasets that include mobile transmitters or receivers typically cover only tens of seconds (Underwater Acoustic Channels Group, 2026 [35]; Walree et al., 2017 [34]). Simulation frameworks for UAC nevertheless enable realistic scenario generation. Bellhop (Porter, 2011 [98]), for instance, is a physics-based ray-tracing model that simulates propagation paths and amplitudes from the SSP, sea-surface and seabed properties, source and receiver positions, and other environmental inputs. Statistical models can then be used to add time variations. This is the case of the Maximum Entropy model (Socheleau et al., 2010 [66]), which simulates channel fluctuations from input statistics such as Doppler spread and average echo power. The objective here is to create channel conditions in which the near-optimal hyperparameter regions vary over time.

Channel variation is achieved by simulating a moving receiver whose range varies from 1900 m to 25 m (or conversely) at 3 m/s, as illustrated in Figure 4.2. Packets are transmitted with a packet gap of 1 s.

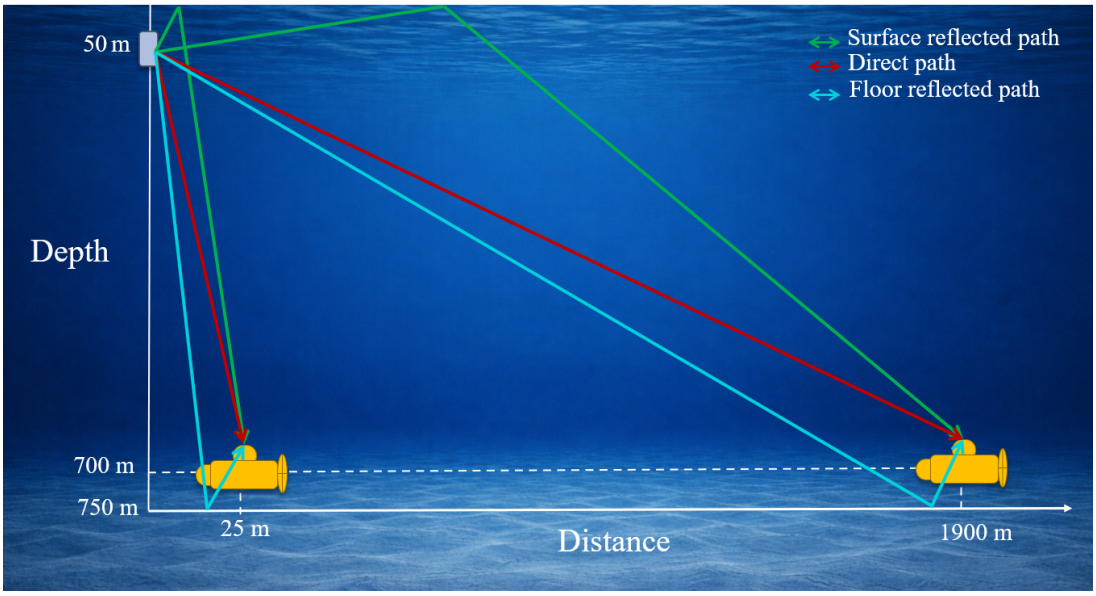
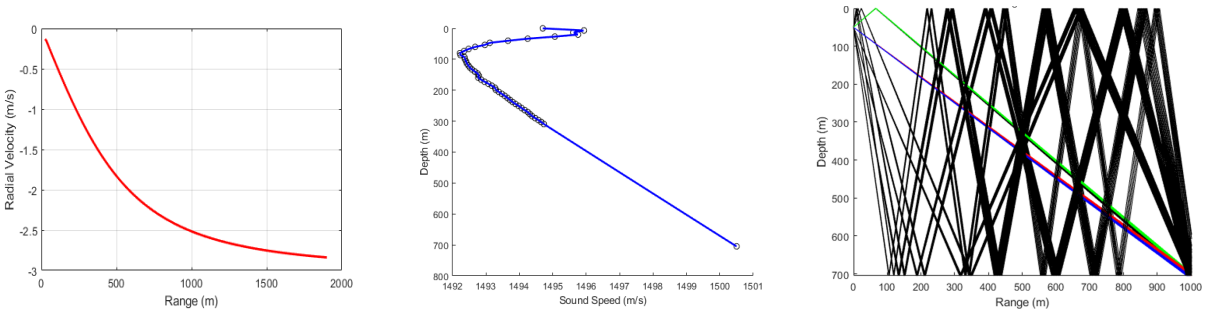


Figure 4.2 – Simulated scenario with a moving receiver (yellow vehicle) and a source at 50 m depth (gray). The direct path is shown in red, surface-reflected paths in green, and bottom-reflected paths in light blue.



(a) (b) (c)

Figure 4.3 – Simulated scenario. (a) Radial velocity versus range. (b) Sound-speed profile. (c) Ray tracing at receiver range 1000 m (direct path in red, surface reflections in green, bottom reflections in blue).

For each transmitted packet, Bellhop generates a channel impulse response using the following parameters:

— Flat seabed and flat sea surface,
— Central frequency: $F_c = 20$ kHz,
— Source depth: 50 m,
— Receiver depth: 700 m.

To increase realism, raw Bellhop outputs are attenuated using Thorp attenuation to account for frequency-dependent propagation loss (Thorp, 1967 [31]). The attenuated impulse responses are then processed using the Maximum Entropy model to introduce time-varying fluctuations with:

— Doppler spread randomly drawn in $[0.2, 0.5]$ Hz,
— Rice factor $K = 40 \times (\frac{1}{2})^{n_r}$ per received path, where n_r is the number of surface and bottom reflections.

Figure 4.3 summarizes the simulation scenario. Panel (a) plots the line-of-sight (radial) velocity versus horizontal range. The source moves toward the receiver, so the range d_{range} decreases at a constant 3 m/s. With fixed receiver depth $d_{\text{depth}} = 700$ m, the slant distance is $d_{\text{SR}} = \sqrt{d_{\text{range}}^2 + d_{\text{depth}}^2}$. As d_{range} decreases, the radial velocity (the time derivative of d_{SR}) decreases in magnitude and tends to zero, while the radial acceleration increases. Panel (b) shows the sound-speed profile, same as Figure 1.1, measured in the North Atlantic in spring. Panel (c) displays ray tracing at a communication range of 1000 m. The line-of-sight path (red) and the bottom-reflected path (blue) arrive close in time because the receiver depth is 700 m and the water depth is 705 m. The surface-reflected path (green) arrives later, which produces a significant Delay spread, as illustrated in Figure 4.4.

Throughout the scenario, as the receiver moves from 1900 m to 25 m or vice versa, the magnitudes of the line-of-sight and surface-reflected paths remain comparable. The main variation occurs in the Delay spread, defined in Chapter 1, equation (1.7). The Delay

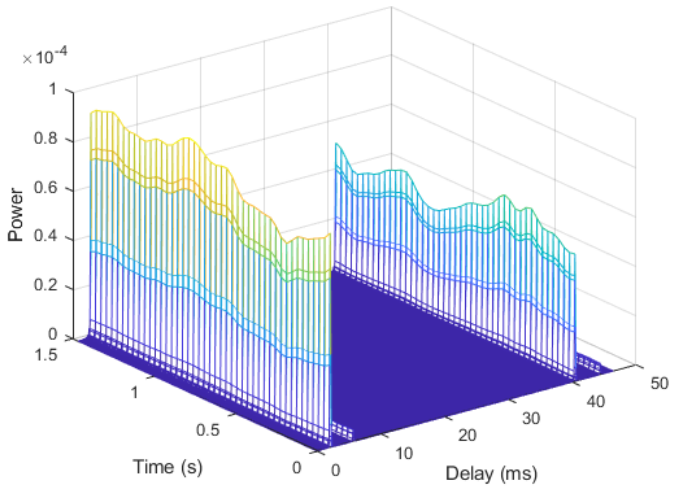


Figure 4.4 – Channel generated at receiver range $RX_r = 1000$ m.

spread increases from approximately 37 ms to 74 ms, as shown in Figure 4.5 panel (a). Instances where the Delay spread drops below 8 ms correspond to shadow zones in which the surface-reflected path does not reach the receiver. These dynamics directly impact the objective function (see Sections 4.4 and 4.5) and drive the need for longer feedback filters (L_B) to maintain good receiver performance.

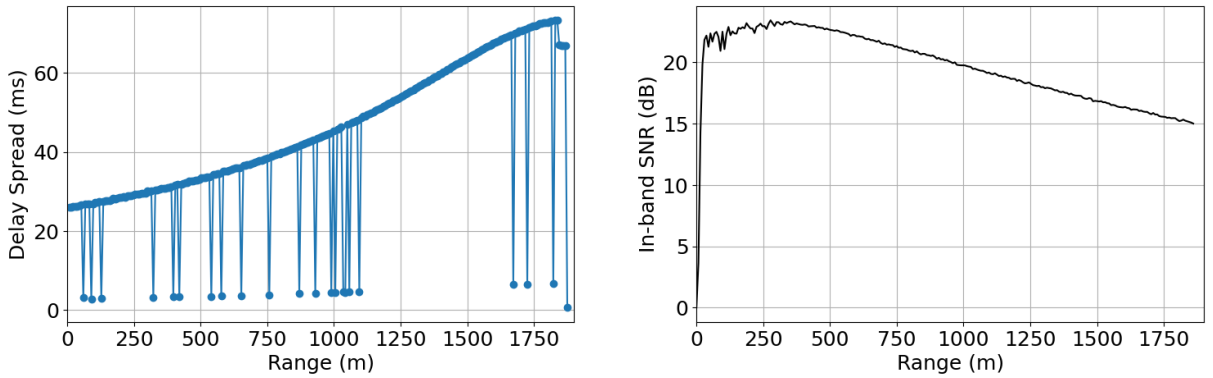
The noise is white Gaussian and its variance is set so that the input in-band SNR at 1000 m is 20 dB. The input SNR at other ranges is determined by the propagation loss relative to the fixed noise variance. Figure 4.5 panel (b) shows the resulting SNR evolution over range. The SNR increases as the range decreases, but it drops sharply in shadow zones at short range.

Throughout the study, real open access sounded channels will be used for assessing the Tracking method, however, due to the short coverage times, these channels will be replayed several times to mimic long coverage times.

Remark 9 *The geometry of the simulated test scenario is chosen to generate a sufficiently long and challenging channel, with Delay spreads that stress both the receiver and the Channel-Estimation benchmark. At the same time, the channel must remain optimizable with the considered receiver architecture. Many geometries lead to unoptimizable channels, meaning that no hyperparameter combination yields successful decoding. The selected geometry is a compromise that remains realistic while still producing packets that can be decoded for some region of the hyperparameter space, even if that region is small.*

The range is capped at 1900 *m because, beyond that point, the objective function becomes trivial, as the Delay spread becomes too small to require tracking. Ranges below* 25

m are also discarded because they correspond mainly to shadow zones, where no rays reach the receiver. Finally the simulated scenario correspond to a coverage time of 10.4 *minutes.*



(a) Delay-spread vs range.

(b) Simulated in-band SNR.

Figure 4.5 – Channel dynamics in the simulated scenario. Panel (a) shows the Delay-Spread evolution and Panel (b) shows the corresponding in-band SNR evolution.

4.4 Single-Agent adaptation method

4.4.1 Objective function and implementation

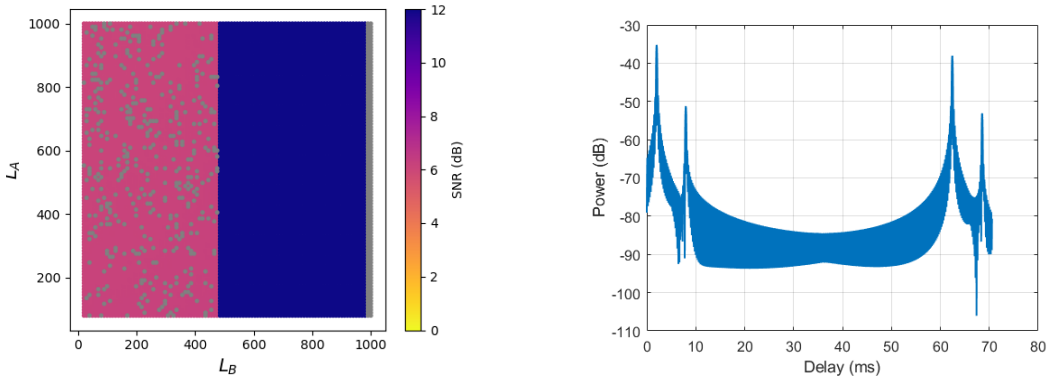
The flywheel mode relies on the hypothesis that the near-optimal region in the hyper-parameter space drifts over time and can be tracked by observing the objective function once per received packet. The key idea is to build an agent that reuses information gathered from previous packets while still exploring, so that it can follow channel variations.

We start by building a simplified Tracking method based on flywheel mode. The method uses a single agent with a UCB policy, since its exploration level is explicit and controlled through a user-defined parameter. In contrast, for other policies, exploration is either absent or depends implicitly on the size of the dataset $\mathcal{D}$, see Chapter 2, Section 2.3.3. For simplicity, the surrogate model is static in this first version.

Although several hyperparameters are involved in the considered DFE structure (Chapter 1, Section 1.3), we focus here on the filter lengths L_A and L_B.

$$\boldsymbol{\theta} \triangleq [L_A, L_B]^T. \tag{4.6}$$

The remaining hyperparameters are considered in the more complete Tracking method. Experiments are conducted on the simulated scenario of Section 4.3 for ranges from 25 m to 1900 m at speed 3 m/s.



(a) Objective map over L_A and L_B.

(b) Power profile.

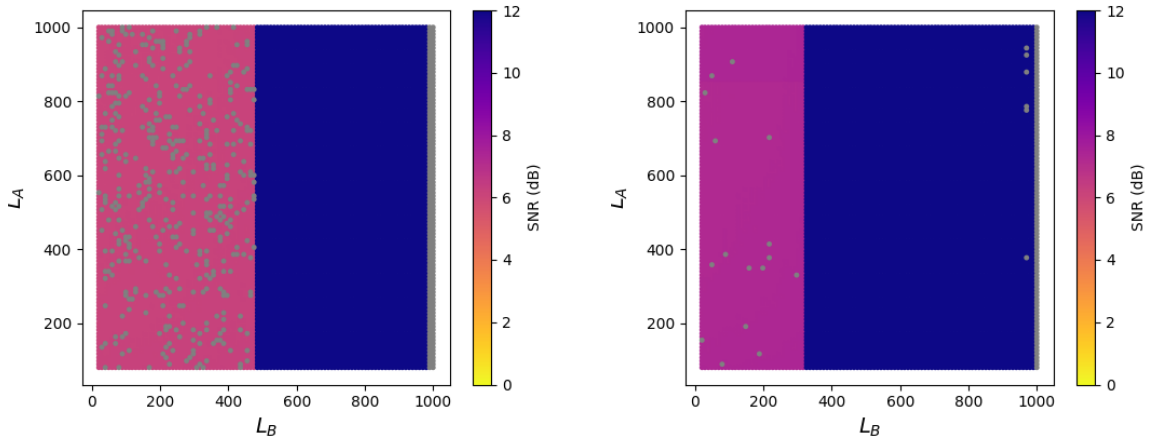
Figure 4.6 – Objective-function structure for a representative packet. Left: objective map over L_A and L_B with color-coded output SNR and gray crosses indicating hallucinations. Right: power profile showing the distribution of received Power over delay.

Figure 4.6(a) shows the objective function over (L_A, L_B), where gray crosses indicate hallucinations. Dark-blue regions correspond to higher output SNR, while orange regions correspond to lower output SNR. A marked performance transition occurs when L_B exceeds approximately 480 taps, which coincides with the appearance of a second echo around 67 ms. Figure 4.6(b) confirms this interpretation, showing a dominant arrival at about 2 ms and a secondary arrival around 67 ms. This example illustrates how Delay spread shapes the objective landscape and, in particular, how it imposes a minimum feedback length requirement.

Figure 4.7 shows two objective functions corresponding to two different ranges separated by 900 m. The minimum feedback length required for good performance decreases with increasing range, which is consistent with the reduction in Delay spread observed in Figure 4.5a as the relative separation between the line-of-sight and surface-reflected arrivals shrinks.

4.4.2 Single-agent implementation

In this section, the surrogate model is a GP with squared-exponential kernel k_{SE}, as defined in Chapter 2, Definition 3. In data-rich settings, the output scale and lengthscales can be optimized by marginal likelihood maximization, as discussed in Chapter 2 (see (2.18)). However, in the packet-based receiver-tuning setting considered here, each evaluation is expensive and the amount of data is limited. For this reason, these parameters are fixed to $l = 0.1$ and $\sigma^2 = 20$, based on preliminary experimentation that yielded stable behavior over typical DFE filter-length ranges.



(a) Range = 1000 m. (b) Range = 1900 m.

Figure 4.7 – Objective function over L_A and L_B for two packets observed at different ranges. The input SNR is 15 dB, and bandwidth is 10 kHz.

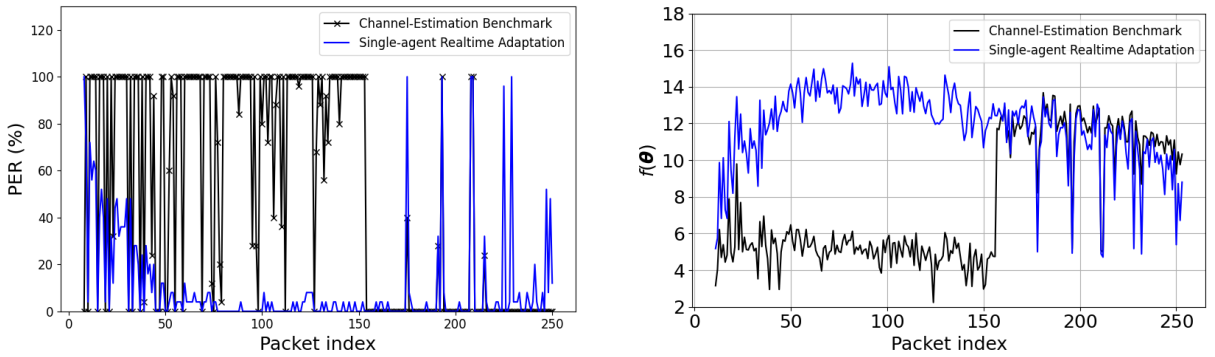
Algorithm 2 summarizes the single-agent Tracking method.

Algorithm 2 Tracking method, single agent

1: **Warm-up session**: construct an initial dataset $\mathcal{D}$ by randomly selecting and evaluating hyperparameters $\boldsymbol{\theta}$ on the earliest received packet(s).
2: **Flywheel mode**:
3: **for** each newly received packet p_t after warm-up **do**
4: 1) Update the surrogate model F using $\mathcal{D}$.
5: 2) Select a hyperparameter vector $\boldsymbol{\theta}_t$ using an acquisition policy based on F.
6: 3) Run the receiver once with $\boldsymbol{\theta}_t$ and compute $f_t(\boldsymbol{\theta}_t)$.
7: 4) Update the dataset: $D \leftarrow D \cup \{(\boldsymbol{\theta}_t, f_t(\boldsymbol{\theta}_t))\}$.
8: **end for**

During operation, the GP posterior is updated using the dataset $\mathcal{D}$. The policy π (UCB in this section, see Section 2.3.3) then selects a hyperparameter vector $\boldsymbol{\theta}_t$ intended to be better matched to the current channel conditions. The receiver is executed once with this candidate, and the resulting observation $(\boldsymbol{\theta}_t, f_t(\boldsymbol{\theta}_t))$ is appended to $\mathcal{D}$.

A single objective evaluation has computational cost $O\Big(N \max(L_A, L_B)\Big)$, where N is the number of symbols in the packet. Updating the GP posterior with n observations scales as $O(n^2)$. Since n grows by one per packet, the update cost remains manageable for moderate packet streams. If required, older observations can be discarded to cap n and preserve realtime feasibility.



(a) PER versus packet index.

(b) Average output SNR versus packet index.

Figure 4.8 – Single-agent Tracking method results on the simulated scenario. Panel (a) reports PER and panel (b) reports average output SNR. © 2025 IEEE.

4.4.3 Results

The scenario described in Figure 4.2 is replayed 25 times, each time with a different random warm-up consisting of $n = 10$ hyperparameter vectors (Section 4.4.2) and independent noise realizations. Packets are 1.5 s long and transmitted with bandwidth $B_3 = 10$ kHz (Chapter 1, Table 1.1). This bandwidth is selected because the apparent channel length increases with bandwidth, which increases Delay spread and yields a narrower region of interest, making the tuning problem more challenging. Figure 4.8 compares the Channel-Estimation benchmark (black) and the Tracking method (blue). Panel (a) shows the packet-error rate (PER) versus packet index. The Channel-Estimation benchmark relies on echo-delay estimates obtained from correlating the received signal with the preamble. It fails to decode packets with indices $t \leq 155$ (range ≤ 1162.5 m), for which PER reaches 100%. The primary cause is that the second most energetic echo arrives at a delay larger than the preamble duration. For packets with indices $t \geq 156$, the Channel-Estimation benchmark succeeds, since the second echo falls within the preamble span.

In contrast, the Tracking method exhibits a convergence phase and stabilizes around packet $t \approx 80$. For packets in $t \in [80, 220]$, the PER remains below approximately 10%, except for six packets where channel conditions are sufficiently severe that neither method yields successful decoding. A performance degradation appears on the last ten packets relative to the Channel-Estimation benchmark. This behavior is attributed to the exploration level of the UCB policy (here $\lambda_t = 1, \forall t$), which remains active even when the objective becomes closer to stationary at short ranges. This highlights the need for a more judicious selection, or adaptation, of the exploration parameter λ_t.

PER is computed for each packet and then averaged over all 25 replays. Panel (b)

shows the corresponding average output SNR versus packet index. The convergence phase is visible over the first ~ 80 packets, with output SNR increasing from about 6 dB to 14 dB. Over packets $t \in [81, 220]$, a slight downward trend is observed, which is consistent with the input-SNR decrease shown in Figure 4.5. The Tracking method maintains output SNR above 10 dB, while the Channel-Estimation benchmark yields output SNR between 2 dB and 8 dB for the first 155 packets, except for one packet. For $t \geq 156$, the Channel-Estimation benchmark approaches the performance of the Tracking method and slightly exceeds it over the final ten packets, where the Delay spread becomes small relative to the preamble. For presentation purposes, hallucination output-SNR values are set to 0 rather than -50 dB.

4.4.4 Single-agent limitations

While the single-agent Tracking method demonstrates that a drifting region of near-optimal hyperparameters can be tracked with only one objective-function evaluation per received packet, its performance is constrained by two structural limitations.

First, the method is sensitive to the exploration–exploitation trade-off imposed by the UCB policy through its exploration factor λ_t. If λ_t is too small, the Tracking method may under-explore and become trapped in a suboptimal region learned from early channel states. If λ_t is too large, excessive exploration can degrade stability and lead to unnecessary performance losses when the objective becomes closer to stationary. In rapidly changing scenarios, such as the present simulation with a receiver speed of 3 m/s, larger λ_t values can be beneficial to preserve tracking capability. Conversely, under more stationary conditions, smaller λ_t values promote faster convergence and better stability. Empirically, stable behavior is observed for $\lambda_t \in [1, 10]$ in the simulated scenario. This limitation becomes apparent at short range, where the objective is closer to stationary and the Channel-Estimation benchmark becomes competitive, making persistent exploration detrimental. This motivates Tracking method strategies that control, or coordinate, exploration more explicitly.

Second, the current implementation treats past observations and recent observations equally when updating the surrogate. In a time-varying channel, this can cause stale information to dominate decisions. For example, hallucinations observed for $L_B \leq 470$ taps in Figure 4.7 panel (a) should not be treated as universally bad across the trajectory. At other ranges, such as in panel (b), the same region can become non-hallucinating and may even yield good performance. Treating earlier hallucinations as persistent can therefore bias the surrogate away from regions that later become optimal. This motivates incorporating temporal effects into the surrogate and the decision rule, for instance through forgetting mechanisms or explicitly time-varying models.

In summary, a more complete Tracking method requires
— the ability to, simultaneously, explore extensively the search space, and exploit extensively to choose only high performance high confidence regions. balance exploration and exploitation in a manner that is robust to changing operational regimes
— a surrogate model that progressively discounts old observations
Such a Tracking method is introduced in the next section.

4.5 WandS: A multi-agent Tracker

4.5.1 WandS features and objective function

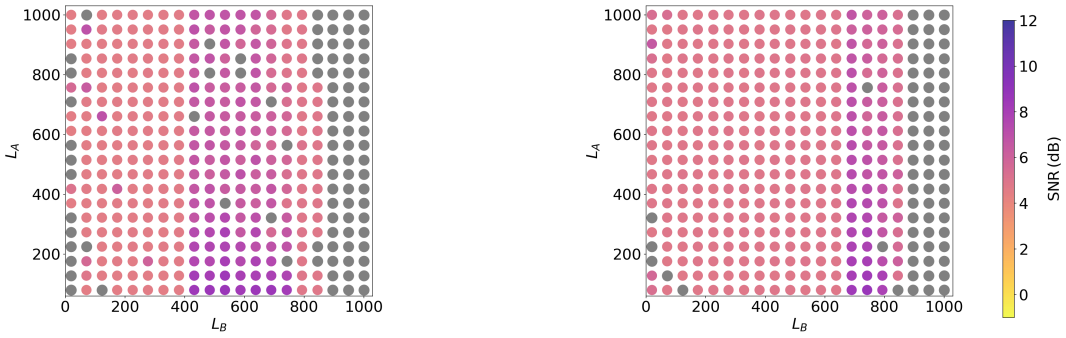
Following the single-agent proof of concept in Section 4.4, this section introduces ***WandS***, a more complete Tracking method. WandS is designed to remain effective along a full communication link, where two regimes typically coexist: highly dynamic phases, in which the near-optimal region drifts and must be tracked through exploration, and quasi-stationary phases, in which excessive exploration becomes detrimental. It also explicitly addresses the fact that past observations can become stale as the channel state evolves.

WandS is built on two ingredients.

First, Section 4.5.2 introduces a time-varying surrogate model that progressively discounts older observations. This keeps the surrogate responsive to recent packets and mitigates overfitting to past channel states, which is critical when the objective varies at packet rate, as in Assumption 3.

Second, Section 4.5.3 presents a multi-agent coordination strategy based on two BO agents with complementary roles, which motivates the name *WandS*: a ***W****ise* agent that performs conservative exploitation ***and*** a ***S****cout* agent that prioritizes exploration. These two agents are run in parallel. A central node coordinates both agents so that exploration and exploitation occur simultaneously at packet rate and, for each received packet, selects the most adapted hyperparameter suggestion among those proposed by *Wise* and *Scout*. To preserve stable performance, both agents rely on hallucination-aware policies that steer the search away from hallucination-prone regions.

Throughout this section, the hyperparameters of interest are the feedforward and feedback filter lengths (L_A, L_B) and the PLL gains (K_1, K_2) (see Chapter 1, Section 1.3). Other IPNLMS hyperparameters (Pelekanakis et al., 2010 [54]) are not explored, since preliminary experiments indicated limited sensitivity under the test scenario described in Section 4.3.



(a) Range = 1900 m. (b) Range = 1000 m.

Figure 4.9 – Objective function over L_A and L_B filter lengths at two ranges within the same simulated subsea environment. The input SNR is 15 dB, and bandwidth is 15 kHz.

Two hyperparameter vectors are considered:

$$\boldsymbol{\theta}_{2D} = [L_A, L_B]^T, \tag{4.7}$$

and

$$\boldsymbol{\theta}_{4D} = [L_A, L_B, K_1, K_2]^T. \tag{4.8}$$

Figures 4.9 and 4.10 provide snapshots of the corresponding objective functions for $\boldsymbol{\theta} = \boldsymbol{\theta}_{2D}$ (Equation 4.7) and $\boldsymbol{\theta} = \boldsymbol{\theta}_{4D}$ (Equation 4.8). In both figures, color indicates the average output SNR in dB: lighter colors correspond to low-SNR (poor) performance, darker colors to high-SNR (good) performance, and gray points indicate hallucination-prone settings. Throughout this section, the bandwidth is set to $B_4 = 15$ kHz. Compared to Section 4.4, this larger bandwidth increases the apparent channel length and yields narrower high-performance regions, making the tuning problem more challenging.

Figure 4.9 shows the 2D landscape over (L_A, L_B) at two ranges within the same environment (Section 4.3). The left panel corresponds to a range of 1900 m and the right panel to 1000 m. The near-optimal region shifts with range: for this environment, the shorter range requires larger L_B values, consistent with the trajectory-dependent increase in delay spread. The figure also illustrates why tuning is harder at $B_4 = 15$ kHz: the high-SNR region is comparatively narrow and hallucinations become more prevalent, particularly for $L_B \geq 900$.

Figure 4.10 extends the picture by including the PLL gains (K_1, K_2). Besides confirming the range-dependent shift of the best-performing regions, it highlights the coupling between equalizer and PLL hyperparameters: good performance requires coordinated tuning, and the landscape is both rough and time-varying. These observations motivate the core design of WandS: a surrogate that adapts to regime changes and a multi-agent strategy that maintains exploration while keeping packet-rate decisions conservative.

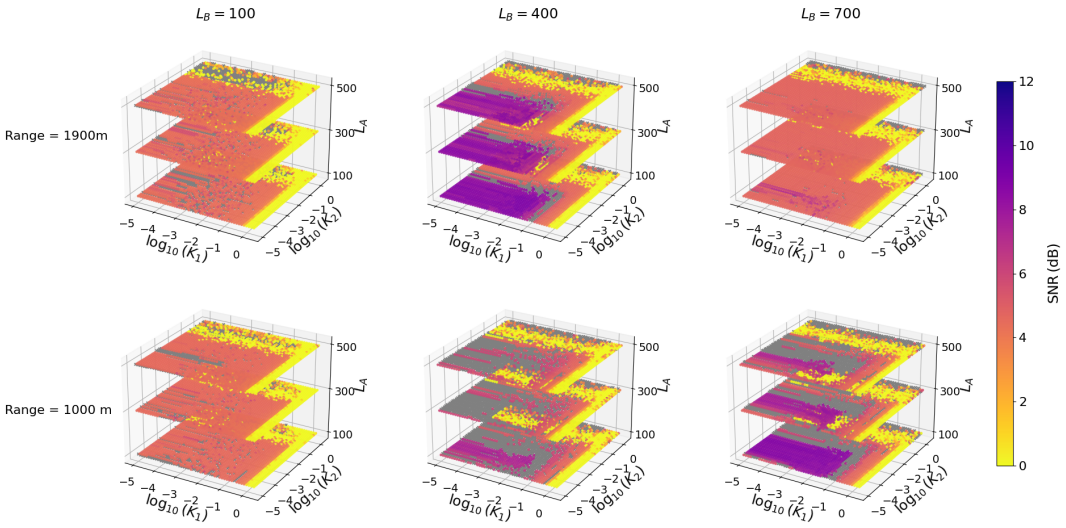


Figure 4.10 – Objective function over L_A, L_B, K_1 , and K_2 for two communication ranges in the same simulated subsea environment. The input SNR is 15 dB, and bandwidth is 15 kHz.

4.5.2 Time-varying surrogate model

Assumption 3 requires a time-varying model of f_t to track its changes over time. A standard modeling approach is to index the objective by time and assume smooth temporal drift:

$$f_t(\boldsymbol{\theta}) \triangleq f(\boldsymbol{\theta}, t), \qquad t = 1, 2, \ldots \tag{4.9}$$

To model such a function, this work employs a Time-Varying Gaussian Process (TV-GP) model (Bogunovic et al., 2016a [111]), adapted for receiver hyperparameter tuning via Bayesian optimization. To the best of our knowledge, this work constitutes the first application of such a model to communication receiver hyperparameter tuning. Initially, the surrogate model is:

$$F_1 \sim \mathcal{GP}(0, v), \quad v \equiv \sigma^2, \tag{4.10}$$

and for $t > t_w$ (t_w is the warm-up duration), $\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}$,

$$F_{t+1}(\boldsymbol{\theta}) = \sqrt{1 - \epsilon_{TV}} \times F_t(\boldsymbol{\theta}) + \sqrt{\epsilon_{TV}} \times \varsigma(\boldsymbol{\theta}), \quad \varsigma \sim \mathcal{GP}(0, v) \tag{4.11}$$

where ϵ_{TV} is the forgetting factor. This parameter determines how quickly older information is forgotten and uncertainty is introduced via ς, with the intuition that as time goes by, uncertainty increases for hyperparameter vectors whose influence on the model becomes weaker. Consequently, larger values (closer to 1) are appropriate for highly dynamic conditions, while smaller values better suit stationary environments. The parameter σ^2 is the outputscale of the corresponding kernel function.

$D = \{(\boldsymbol{\theta}_1^{(i)}, f_1(\boldsymbol{\theta}_1^{(i)}), 1)\}_{i=1}^{N_w} \cup \{(\boldsymbol{\theta}_l, f_l(\boldsymbol{\theta}_l), l)\}_{l=t_w+1}^{t}$, the stacked hyperparameter vectors, objective function values, and time index vectors are:

$$\boldsymbol{X}_t = \left[\boldsymbol{\theta}_1^{(1)}, \ldots, \boldsymbol{\theta}_1^{(t_w)}, \boldsymbol{\theta}_{t_w+1}, , \ldots, \boldsymbol{\theta}_t\right]^T \tag{4.12}$$

$$\boldsymbol{y}_t = \left[f(\boldsymbol{\theta}_1^{(1)}), \ldots, f(\boldsymbol{\theta}_1^{(t_w)}), f(\boldsymbol{\theta}_{t_w+1}), \ldots, f(\boldsymbol{\theta}_t)\right]^T \tag{4.13}$$

$$I_t = [\underbrace{1, \ldots, 1}_{t_w}, t_w + 1, \ldots, t] \tag{4.14}$$

The first t_w entries in I_t correspond to the warm-up phase.

The mean and covariance update as in (2.14) and (2.15), but with exponential time weighting:

$$m_{t+1}(\boldsymbol{\theta}) = \boldsymbol{k}_t(\boldsymbol{\theta})^T \boldsymbol{K}_t^{-1} \boldsymbol{y}_t, \forall \boldsymbol{\theta} \in \boldsymbol{\Theta} \tag{4.15}$$

$$v_{t+1}(\boldsymbol{\theta}, \boldsymbol{\theta}') = k_{SE}(\boldsymbol{\theta}, \boldsymbol{\theta}') - \boldsymbol{k}_t(\boldsymbol{\theta})^T \boldsymbol{K}_t^{-1} \boldsymbol{k}_t(\boldsymbol{\theta}'), \forall (\boldsymbol{\theta}, \boldsymbol{\theta}') \in \boldsymbol{\Theta}, \tag{4.16}$$

where the weighted kernel vector is

$$\boldsymbol{k}_t(\boldsymbol{\theta}) = \boldsymbol{k}_s(\boldsymbol{\theta}) \circ \boldsymbol{g}_t, \qquad \boldsymbol{g}_t = \left[(1 - \epsilon_{TV})^{(t-i)/2}\right]_{i \in I_t}^T, \tag{4.17}$$

with $\circ$ denoting the Hadamard product, and

$$\boldsymbol{k}_s(\boldsymbol{\theta}) = [k_{SE}(\boldsymbol{\theta}, \boldsymbol{X}_t[1]), \ \ldots, \ k_{SE}(\boldsymbol{\theta}, \boldsymbol{X}_t[t])]^T. \tag{4.18}$$

The full covariance is

$$\boldsymbol{K}_t = \boldsymbol{K}_s \circ G_t, \qquad G_t = \left[(1 - \epsilon_{TV})^{|l-m|/2}\right]_{l,m \in I_t}, \tag{4.19}$$

where $[\boldsymbol{K}_s]_{i,j} = k_{SE}(\boldsymbol{X}_t[i], \boldsymbol{X}_t[j])$ for $i, j \in \{1, \#\boldsymbol{X}_t\}$ ($\#\boldsymbol{X}_t$ is the number of elements in $\boldsymbol{X}_t$).

This TV-GP surrogate model applies a smooth forgetting mechanism to aging observations in $\mathcal{D}$, at a rate controlled by ϵ_{TV}, thereby reducing the influence of past channel states and mitigating overfitting.

4.5.3 Multi-agent approach

The second ingredient of WandS is its multi-agent structure, which enables simultaneous conservative hyperparameter tuning and adaptability to channel variations. Together, The *Wise* and *Scout* agents are designed to maintain stable performance from one packet to the next while still allowing exploration when needed.

Algorithm 3 ψ: Dual-agent function

Require:
— $D = \{(\boldsymbol{\theta}_j, f_j(\boldsymbol{\theta}_j), j)\}_{j=1}^{t-1}$.
— F_t: current surrogate model of f_t, with mean m_t and variance v_t.
$\boldsymbol{\theta}_t^{\mathrm{S}} \leftarrow \pi_{\mathrm{S}}(m_t, v_t)$ // Scout proposal
$\boldsymbol{\theta}_t^{\mathrm{W}} \leftarrow \pi_{\mathrm{W}}(m_t, v_t)$ // Wise proposal
$y_{\mathrm{S}} \leftarrow f_t(\boldsymbol{\theta}_t^{\mathrm{S}})$
$y_{\mathrm{W}} \leftarrow f_t(\boldsymbol{\theta}_t^{\mathrm{W}})$
$CRC_{\mathrm{S}} \leftarrow \mathrm{CRC}(p_t; \boldsymbol{\theta}_t^{\mathrm{S}})$
$CRC_{\mathrm{W}} \leftarrow \mathrm{CRC}(p_t; \boldsymbol{\theta}_t^{\mathrm{W}})$
if $(CRC_{\mathrm{S}} = \mathrm{OK}\ \&\ CRC_{\mathrm{W}} = \mathrm{KO})$ **OR** $\left(CRC_{\mathrm{W}} = \mathrm{KO}\ \&\ y_{\mathrm{S}} \geq y_{\mathrm{W}}\right)$ **then**
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_t^{\mathrm{S}}, \quad y_t \leftarrow y_{\mathrm{S}}$
else
  $\boldsymbol{\theta}_t \leftarrow \boldsymbol{\theta}_t^{\mathrm{W}}, \quad y_t \leftarrow y_{\mathrm{W}}$
end if
$D \leftarrow D \cup \{(\boldsymbol{\theta}_t^{\mathrm{S}}, y_{\mathrm{S}}, t), (\boldsymbol{\theta}_t^{\mathrm{W}}, y_{\mathrm{W}}, t)\}$
Train F_{t+1} using $\mathcal{D}$ // update surrogate for next packet

Return: $(\boldsymbol{\theta}_t, y_t, t)$ and $\mathcal{D}$

In order to build the multi-agent method ***WandS***, the following sequential flywheel procedure is considered, analogous to the one introduced in Section 4.2. It is divided into two phases:

— **Warm-up phase:** An initial dataset $D_w \triangleq \{(\boldsymbol{\theta}_1^{(i)}, f_1(\boldsymbol{\theta}_1^{(i)}), 1)\}_{i=1}^{N_w}$ is constructed by evaluating N_w randomly sampled hyperparameter vectors $\boldsymbol{\theta}_1^{(i)} \in \boldsymbol{\Theta}$ on the first received packet p_1. Any prior knowledge about plausible hyperparameter values may also be included. The warm-up duration, i.e., the number of packets processed while building D_w, is denoted by t_w.

— **Flywheel mode:** For each packet p_t with $t \geq t_w + 1$, a dual-agent routine $\psi(D, F_t)$ (Algorithm 3) selects a single hyperparameter vector $\boldsymbol{\theta}_t$. Here, $\mathcal{D}$ denotes the cumulative dataset (updated in place by ψ) and F_t denotes the surrogate model at time t. The objective $f_t(\boldsymbol{\theta}_t)$ is then evaluated and the record $(\boldsymbol{\theta}_t, f_t(\boldsymbol{\theta}_t), t)$ is appended to $\mathcal{D}$.

The core component of WandS is the dual-agent function ψ, detailed in Algorithm 3. At each time t, it takes as input the current dataset $\mathcal{D}$ and the corresponding fitted TV-GP surrogate F_t, and returns the hyperparameter vector selected for the incoming packet p_t.

Steps 2 and 3 of Algorithm 3 implement the distinct decision rules of the Wise and Scout agents. Both agents select a candidate hyperparameter vector by maximizing their

respective acquisition functions over $\boldsymbol{\Theta}$:

$$\pi_{\mathrm{W}}(m_t, v_t) \triangleq \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \Big(a_{\mathrm{W}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{W}}\Big)\Big), \tag{4.20}$$

$$\pi_{\mathrm{S}}(m_t, v_t) \triangleq \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \Big(a_{\mathrm{S}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{S}}\Big)\Big). \tag{4.21}$$

The functions a_{W} and a_{S} are acquisition functions (Chapter 2, Section 2.3.3) that reflect the Wise agent's conservative behavior and the Scout agent's exploratory role. In this work, they are defined as:

$$a_{\mathrm{W}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{W}}\Big) \triangleq a_{\mathrm{LCB}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{W}}\Big) + \kappa(\boldsymbol{\theta}), \tag{4.22}$$

$$a_{\mathrm{S}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{S}}\Big) \triangleq a_{\mathrm{UCB}}\Big(m_t(\boldsymbol{\theta}), v_t(\boldsymbol{\theta}), \lambda_{\mathrm{S}}\Big) + \kappa(\boldsymbol{\theta}), \tag{4.23}$$

where a_{LCB} and a_{UCB} denote the Lower Confidence Bound and Upper Confidence Bound acquisition functions, respectively, as defined in Chapter 2, Section 2.3.3.

Here, λ_{S} and λ_{W} control the exploration–exploitation trade-offs of the Scout and Wise agents, respectively. In this work, these parameters are kept fixed across packets. This simplifies interpretation and, under the simulated scenarios considered in this chapter, was empirically sufficient to obtain stable behavior.

The Scout agent uses a UCB-type criterion, which promotes exploration by assigning larger scores to hyperparameters that combine a high predicted performance with high predictive uncertainty. This encourages the agent to probe uncertain but potentially rewarding regions, thereby reducing model uncertainty where it matters. In contrast, the Wise agent uses an LCB-type criterion, which promotes exploitation by favoring hyperparameters with high predicted performance and low uncertainty, consistent with a conservative selection strategy.

A second key aspect of the proposed policies is the avoidance of hallucination-prone regions. Although the objective function defined in Section 1.4 of Chapter 1 penalizes hallucinations by assigning a fixed value of -50 dB, this penalty alone is not sufficient to reliably mitigate hallucinations through fine-tuning at packet-rate, for three reasons.

First, hallucinations tend to cluster in specific areas of the search space, as illustrated in Figure 4.10. If a hallucination is not detected (depending on the detection threshold τ), it may be assigned an artificially high output SNR. This can bias the surrogate toward that region and, in turn, make neighboring hyperparameter vectors appear attractive even though the region is contaminated.

Second, even when hallucinations are detected and assigned the fixed penalty, their spatial influence on the GP posterior depends strongly on the kernel length-scale l (Chapter 2, Definition 3). Large values of l spread the penalty over a broad neighborhood,

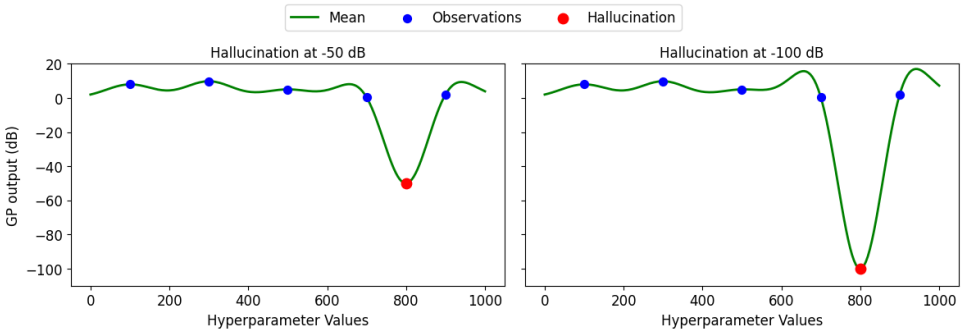


Figure 4.11 – Effect of excessive hallucination penalization on GP surrogate fitting. Left: GP fit with reasonable hallucination value (-50 dB). Right: GP fit with one hallucination assigned an extreme penalty (-100 dB), which can induce spurious high predictions near the hallucination.

whereas small values of l produce a narrow local dip around the hallucination (see Chapter 2, Remark 2). As a result, the same fixed penalty can be either too diffuse or too local depending on l.

Third, increasing the penalty magnitude is not a safe remedy. Overly severe penalties can induce oscillations in the GP predictive mean through interpolation, as illustrated in Figure 4.11. In that example, assigning an extreme value (e.g., -100 dB) to a hallucination can create spurious high predictions in its neighborhood, paradoxically attracting the optimizer toward hallucination-adjacent regions.

For these reasons, the acquisition functions in (4.22) and (4.23) include an additional term $\kappa(\boldsymbol{\theta})$ that smoothly discourages sampling near hallucination-prone regions. Moreover, as the set of detected hallucinations grows, this mechanism increasingly prevents repeated exploration of neighborhoods that have already been identified as problematic.

Both agents use the penalty term $\kappa(\boldsymbol{\theta})$ to avoid hallucinations:

$$\kappa(\boldsymbol{\theta}) = (1 - \epsilon_{TV})^{t_h/2} \log\left(\|\boldsymbol{\theta}_h - \boldsymbol{\theta}\|^2\right), \tag{4.24}$$

where $\boldsymbol{\theta}_h = \arg\min_{\boldsymbol{\theta}' \in \boldsymbol{\Theta}_h} \|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2$ denotes the nearest previously observed hallucination in the set of detected hallucinations $\boldsymbol{\Theta}_h$, and t_h is the elapsed time since $\boldsymbol{\theta}_h$ was added to the dataset $\mathcal{D}$. Hyperparameter vectors are normalized to the unit hypercube, so most distances satisfy $\|\boldsymbol{\theta}_h - \boldsymbol{\theta}\| \in [0, 1]$, which makes the logarithm non-positive and strongly penalizes points close to hallucinations. The multiplicative factor $(1 - \epsilon_{TV})^{t_h/2}$ progressively reduces the influence of older hallucinations, allowing the search to recover flexibility when channel conditions change. Note that hallucinations are still assigned the fixed score of -50 dB in $f_t(\boldsymbol{\theta})$, $\kappa(\boldsymbol{\theta})$ is an additional mechanism used inside the acquisition functions to steer the search away from hallucination-prone regions.

After the candidate hyperparameter vectors are selected (Algorithm 3, steps 2–3), the receiver is run with both candidates to obtain $y_{\mathrm{S}} = f_t(\boldsymbol{\theta}_t^{\mathrm{S}})$ and $y_{\mathrm{W}} = f_t(\boldsymbol{\theta}_t^{\mathrm{W}})$. By default, the Wise suggestion $\boldsymbol{\theta}_t^{\mathrm{W}}$ is applied to the current packet p_t. However, if the Scout suggestion $\boldsymbol{\theta}_t^{\mathrm{S}}$ yields higher measured performance and passes the Cyclic Redundancy Check (CRC), then $\boldsymbol{\theta}_t^{\mathrm{S}}$ is selected. In Algorithm 3 (steps 13–14), both evaluations are appended to $\mathcal{D}$ and the surrogate is updated accordingly.

Acquisition functions can be maximized using multi-start gradient descent, grid search, or random sampling (Gardner et al., 2018 [112]). In this chapter, we adopt random sampling, since acquisition values are orders of magnitude cheaper to evaluate than a receiver run. Concretely, we draw N_{cand} candidates uniformly from $\boldsymbol{\Theta}$, evaluate the acquisition on these candidates, and select the maximizer. In our experiments, $N_{\mathrm{cand}} = 1000$ provides a practical balance between computational load and optimization quality. Importantly, the receiver is not run on every candidate: the acquisition values in (4.22)–(4.23) are computed via GP inference, and only the top-scoring candidate is evaluated by the receiver.

From a complexity perspective, policy maximization requires sampling N_{cand} hyperparameter vectors and evaluating each through GP inference, which gives an overall cost of $O(N_{\mathrm{cand}} n^2)$, where n is the number of training points in $\mathcal{D}$. This remains modest compared to the DFE complexity, which scales as $O\left(N_s n_s \max(L_A, L_B)\right)$, where N_s is the number of input symbols and n_s the number of samples per symbol.

In summary, the complete routine is illustrated in Figure 4.12. The process begins with a warm-up phase, where random hyperparameters are evaluated on the first packets to build the initial dataset D_w. The algorithm then enters flywheel mode: for each arriving packet p_t, the TV-GP surrogate is updated, the Wise (exploit) and Scout (explore) agents propose $\boldsymbol{\theta}_t^{\mathrm{W}}$ and $\boldsymbol{\theta}_t^{\mathrm{S}}$, both candidates are evaluated by the receiver, and the CRC-based selection rule in Algorithm 3 selects the hyperparameter vector applied to the packet. This coordinated routine—built around the **W**ise **and** **S**cout agents—is referred to as ***WandS***.

WandS introduces several hyperparameters: the TV-GP forgetting factor ϵ_{TV}, the number of warm-up samples, and the exploration parameters governing the Wise and Scout acquisition functions. The number of warm-up samples controls the standard trade-off between (i) building a stronger initial surrogate and (ii) entering flywheel mode earlier. In stationary channels, a longer warm-up is beneficial; in highly dynamic channels, a shorter warm-up is preferable because warm-up information becomes obsolete more quickly. Warm-up can also be used to seed the search with hyperparameter values expected to perform well. The remaining WandS hyperparameters are straightforward to tune, and, as shown in Section 4.4.3, the method exhibits limited sensitivity to their precise values.

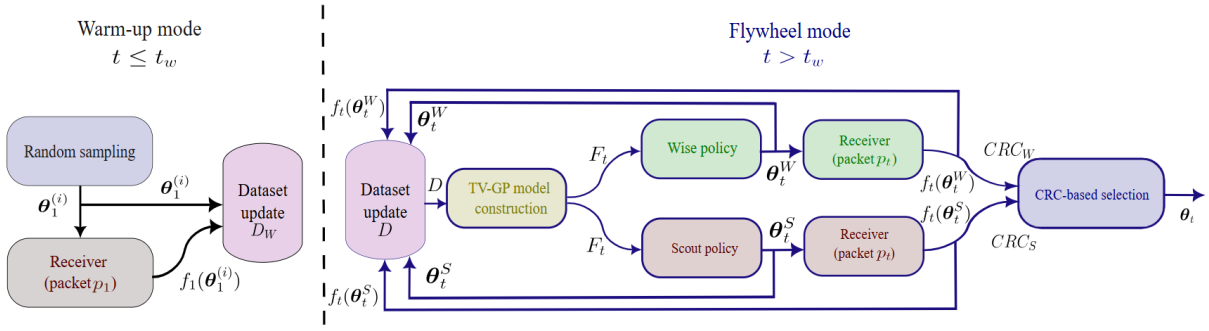


Figure 4.12 – Overview of the *WandS* routine. **Warm-up mode** ($t \leq t_w$): random hyperparameters are evaluated on the first packet to build the initial dataset D_w. **Flywheel mode** ($t > t_w$): for each arriving packet p_t, a TV-GP with forgetting ϵ_{TV} is updated; the Wise (exploit) and Scout (explore) policies propose $\boldsymbol{\theta}_t^{\mathrm{W}}$ and $\boldsymbol{\theta}_t^{\mathrm{S}}$; both are evaluated by the receiver to obtain their scores and CRC flags. A CRC-based selection then chooses the packet's hyperparameter $\boldsymbol{\theta}_t$ from $\{\boldsymbol{\theta}_t^{\mathrm{W}}, \boldsymbol{\theta}_t^{\mathrm{S}}\}$ using their measured performance and CRC status, as specified in Algorithm 3.

4.5.4 Simulation results

This section evaluates *WandS* on the simulated time-varying scenario introduced in Section 4.3. In contrast with the single-agent proof of concept (Section 4.4), the simulated trajectory goes from 1900 m down to 25 m.

We first consider two-dimensional tuning with $\boldsymbol{\theta}_{2D}$ in (4.7) to isolate the effect of the proposed dual-agent architecture. In this setting, *WandS* is benchmarked against a single-agent Tracking method to assess the benefit of coordinating a *Wise* (conservative exploitation) and a *Scout* (exploration) agent. Robustness to channel lengthening is then assessed by repeating the experiment for three bandwidths, $B_2 = 5$ kHz, $B_3 = 10$ kHz, and $B_4 = 15$ kHz (the case $B_1 = 4.5$ kHz is omitted since it is close to B_2).

Next, *WandS* is benchmarked against the Channel-Estimation benchmark (Benchmark 3.1.2), the Known-Delay benchmark (Benchmark 4.2), and an idealized Exhaustive Random search baseline based on random sampling (Benchmark 3.1.2). In this Exhaustive Random search baseline, for each packet, 500 hyperparameter vectors are sampled uniformly at random and evaluated using the bit-error rate (BER); the best-performing configuration is retained. While this procedure is impractical for realtime operation—because it requires hundreds of receiver runs per packet and uses BER, which is not available in operation—it provides a useful proxy for near-optimal per-packet performance.

Finally, we evaluate the sensitivity of *WandS* to its own hyperparameters: the TV-GP forgetting factor ϵ_{TV} in (4.11), also used in the hallucination penalty κ in (4.24), and the exploration factors λ_{W} and λ_{S} used in the Wise and Scout policies in (4.22)–(4.23).

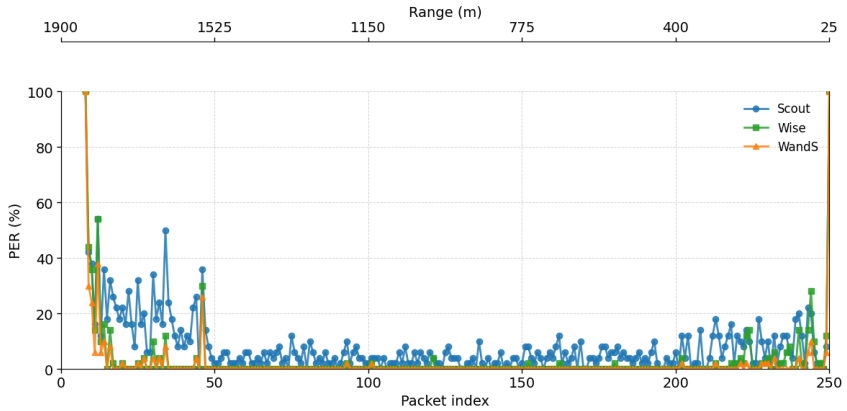


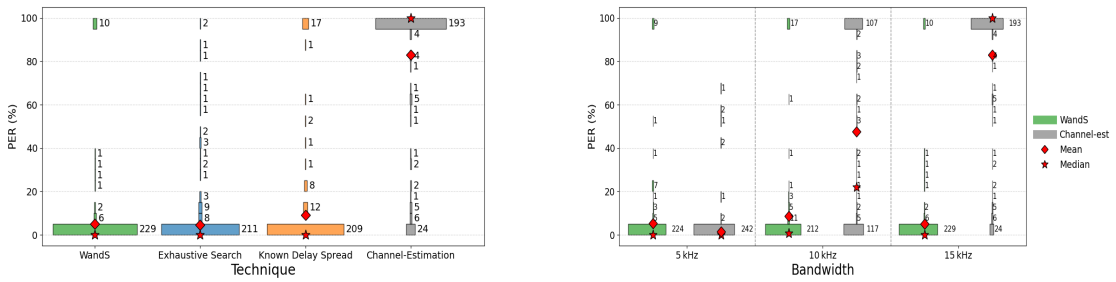
Figure 4.13 – Example PER evolution for the Wise agent, the Scout agent, and the *WandS* routine versus received packet index.

We then extend tuning to the higher-dimensional setting $\boldsymbol{\theta}_{4D}$ in (4.8). In this case, *WandS* is benchmarked against the Exhaustive Random search baseline, the Channel-Estimation benchmark, and the Known-Delay benchmark, and we also report results with and without the hallucination penalty term κ in (4.24).

For all subsequent results in this chapter, PER is computed at each packet index (i.e., for each channel state along the trajectory) and results are averaged over 25 independent Monte Carlo runs. Unless otherwise stated, the default parameters are $\lambda_{\mathrm{S}} = \lambda_{\mathrm{W}} = 10$, $\epsilon_{TV} = 0.01$, and a communication bandwidth of 15 kHz. The warm-up is performed using 10 random samples tested over the first packet p_1.

Two-dimensional tuning

We begin with optimization over $\boldsymbol{\theta}_{2D}$. In this setup, the Doppler-matched filter bank (see Chapter 1, Section 1.3, (1.16)) provides effective compensation of the simulated Doppler, using a velocity resolution of $\delta_{\mathrm{DB}} = 0.1$ m/s, which is sufficient to track the radial-velocity evolution. Consequently, the PLL is omitted in this experiment. Figure 4.13 shows an example PER trajectory comparing the Wise agent, the Scout agent, and the full *WandS* routine. As expected, the Scout agent—which prioritizes exploration—exhibits a higher PER than the Wise agent and *WandS*. The Wise agent leverages accumulated knowledge to stabilize performance, while *WandS* combines both behaviors and achieves the lowest PER overall in this example. Curves are shown starting from packet $t = 7$, since the warm-up phase spans approximately 17 s. The convergence phase lasts roughly 7–10 packets, after which performance stabilizes. Around packet $t = 46$, the channel enters a shadow zone (only the line-of-sight arrival is present), which causes a sharp PER increase; *WandS* recovers by the next packet, consistent with the Scout agent maintaining exploration capability.



(a) Benchmarks comparison

(b) PER vs. bandwidth

Figure 4.14 – Simulation results for *WandS* compared to benchmarks. (a) PER comparison with benchmarks at 15 kHz; (b) PER versus bandwidth comparing *WandS* to the fixed-preamble Channel-Estimation benchmark.

For the remainder of this section, histogram-based plots are used to summarize PER distributions over the full trajectory. Each histogram reports the frequency of PER values within bins, and mean (diamond) and median (star) markers are overlaid. Narrower distributions concentrated at low PER indicate more consistent performance, while lower mean and median values indicate better overall decoding reliability.

Figure 4.14 summarizes performance for $\boldsymbol{\theta}_{2D}$. Panel (a) compares *WandS* to the Exhaustive Random search baseline (500 random samples per packet, using BER), the Known-Delay benchmark, and the Channel-Estimation benchmark with a fixed 511-symbol preamble. For this configuration, *WandS* yields the most favorable PER distribution, with the largest number of near-zero PER packets (229 for *WandS* versus 211 for Exhaustive Random search, 209 for Known Delay, and 29 for Channel-Estimation), and the lowest mean and median PER. The fixed-preamble Channel-Estimation benchmark exhibits the highest failure rate, with many packets near 100% PER. While Exhaustive Random search and Known Delay improve substantially over Channel-Estimation, they still fall short of *WandS* under these conditions.

Panel (b) compares *WandS* against the fixed-preamble Channel-Estimation benchmark across bandwidths. At 5 kHz, the preamble can cover the relevant delay spread (in symbols), and the Channel-Estimation benchmark becomes competitive, occasionally surpassing *WandS*. At higher bandwidths, however, the Channel-Estimation benchmark degrades sharply because the effective delay spread (in symbols) increases and the fixed preamble becomes insufficient. In contrast, *WandS* remains robust across bandwidths without requiring prior channel knowledge or manual adjustment of the preamble length.

To isolate the contribution of the dual-agent architecture, Figure 4.15 compares *WandS* to single-agent Tracking method schemes based on UCB (with $\lambda = 10$), Expected Improvement (EI), and Probability of Improvement (PI) (see Section 2.3.3). *WandS* achieves lower mean and median PER and a higher frequency of packets with PER below 20%, indicating

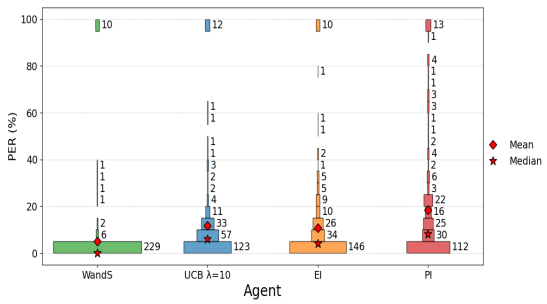


Figure 4.15 – PER comparison with single-agent schemes (UCB with $\lambda = 10$, EI, and PI). The diamond and star markers denote the mean and median PER, respectively.

improved stability relative to these single-agent strategies.

Finally, Figures 4.16 and 4.17 investigate sensitivity to *WandS* hyperparameters. Figure 4.16 shows that performance is robust provided ϵ_{TV} is neither 0 (no forgetting) nor 1 (aggressive forgetting). Figure 4.17 shows that exploration by the Scout agent is essential in this dynamic scenario: increasing λ_{S} improves average PER by maintaining tracking capability, whereas λ_{W} has a more limited effect for the default $\epsilon_{TV} = 0.01$. This is consistent with the TV-GP progressively discounting older information: as ϵ_{TV} increases, the separation between high-mean and low-variance regions becomes more pronounced, and the Scout's exploratory behavior becomes more influential. Overall, these results indicate that *WandS* is not overly sensitive to its hyperparameter settings within reasonable ranges.

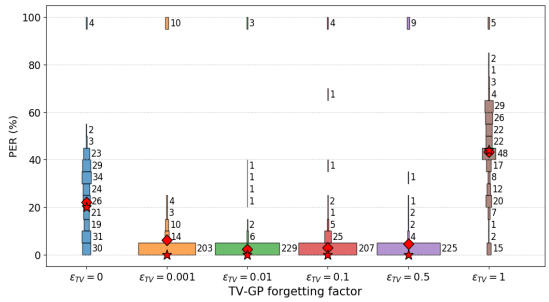


Figure 4.16 – PER comparison for *WandS* using different TV-GP forgetting factors ϵ_{TV}.

Four-dimensional tuning

Next, we consider tuning of the four-dimensional vector $\boldsymbol{\theta}_{4D}$ (Equation 4.8), where both the DFE filter lengths and the PLL gains are optimized. This setting becomes relevant when the Doppler-bank matched-filter resolution δ_{DB} is not sufficiently fine, which is

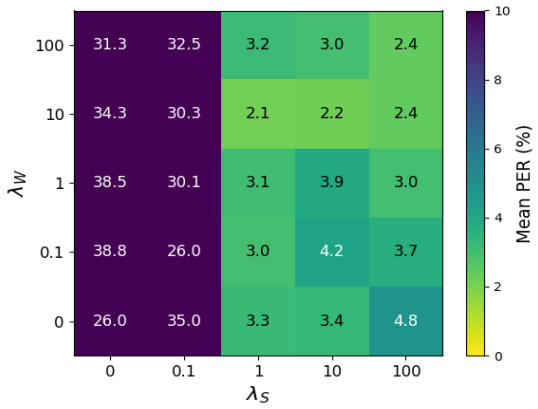


Figure 4.17 – Sensitivity of *WandS* to the exploration parameters λ_{W} and λ_{S} of the Wise and Scout agents, respectively. Brighter squares indicate lower mean PER.

common in low-complexity implementations where computational cost must be controlled.

This four-dimensional study serves two purposes: it evaluates *WandS* against the benchmarks in a higher-dimensional search space, and it quantifies the benefit of the hallucination penalty κ.

In this experiment, the PLL hyperparameters K_1 and K_2 are parameterized as (Eynard et al., 2008 [50])

$$K_1 = 2\varsigma\omega_n \left(1 - \frac{\omega_n}{2\varsigma}\right), \qquad K_2 = \omega_n^2, \tag{4.25}$$

with default values

$$\varsigma = \frac{1}{\sqrt{2}}, \qquad \omega_n = 2\pi \cdot \frac{F_c}{c} \cdot \frac{T_s}{1.5}, \tag{4.26}$$

where T_s is the symbol duration, F_c is the carrier frequency, and $c = 1500$ m/s is the nominal sound speed.

Figure 4.18 reports the mean PER as a function of the Doppler-bank resolution δ_{DB}. For fine resolutions ($\delta_{DB} < 0.2$ m/s), PLL compensation is unnecessary: tuning only the DFE filter lengths ("*WandS* optimizes $\boldsymbol{\theta}_{2D}$ + PLL removed") achieves the lowest mean PER, corresponding to near-complete packet recovery. In this regime, joint tuning ("*WandS* optimizes $\boldsymbol{\theta}_{4D}$") is the next best performer and it outperforms the "Exhaustive Random Search" baseline. Note that this baseline requires 500 receiver runs per packet, which is unrealistic, and it selects its best hyperparameters using the BER, which is not accessible in practice. Methods that tune only PLL gains while using fixed or poorly chosen filter lengths (for instance, Channel-Estimation with a fixed preamble) perform worst, with near-total packet loss across all resolutions. Even the idealized "Known Delay spread" benchmark does not achieve the best results when PLL gains are tuned alone, indicating that filter lengths and PLL gains are coupled and should be optimized jointly.

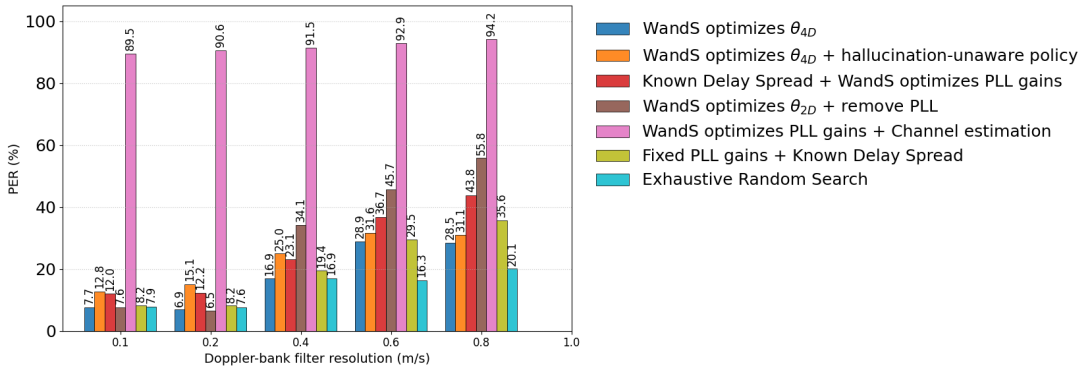
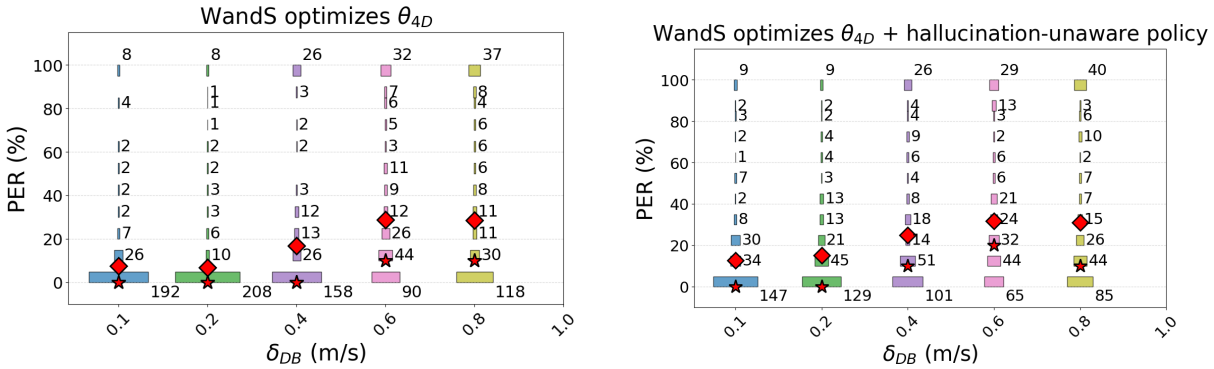


Figure 4.18 – Mean PER versus Doppler–bank filter resolution δ_{DB}. Bar heights are means across simulation runs; numeric labels report the mean PER in %.

For coarser resolutions ($\delta_{DB} \geq 0.4$ m/s), tuning PLL gains becomes increasingly important, and *WandS* yields the best mean PER among practical methods. Finally, comparing the two policy variants, the hallucination-aware version (with the penalty $\kappa(\boldsymbol{\theta})$, Equation 4.24) consistently achieves lower mean PER than the hallucination-unaware version, with the benefit most pronounced for $\delta_{DB} \leq 0.4$ m/s.

The overall performance degradation observed across all methods as δ_{DB} becomes coarser is explained by Doppler-bank mismatch. As δ_{DB} increases, the selected Doppler-bank matched filter is more likely to deviate from the true Doppler scale of the simulated channel, which induces an increasing residual time scaling (and hence timing drift) in the compensated signal. Since the PLL primarily corrects phase (and small frequency offsets) rather than time scaling, this residual mismatch is not fully tracked, leading to a progressive increase in decoding failures and thus higher PER for larger δ_{DB}.



(a) (b)

Figure 4.19 – PER histograms across Doppler–bank resolutions for two policy variants. (a) *WandS* with the hallucination-aware penalty $\kappa(\boldsymbol{\theta})$ (Equation 4.24). (b) *WandS* without this penalty (hallucination-unaware).

Figure 4.19 complements these averages with PER histograms for multiple δ_{DB} values. While the mean PER values can be close, incorporating the hallucination-aware policy shifts probability mass toward PER ≈ 0. Averaged across δ_{DB}, the hallucination-aware policy yields about 30% of packets with PER ≈ 0, highlighting the practical value of hallucination-aware policies.

4.5.5 Channel replay

To further validate the *WandS* method, we conducted tests using real-channel replay. Extensive testing across additional real channels was not feasible due to the limited availability of publicly accessible datasets sufficiently long to demonstrate *WandS*'s tracking capabilities. Consequently, evaluations were conducted using recordings from the SAUWAN/La Spezia channel (Pottier et al., 2021 [113]) and the BCH1 channel (Walree et al., 2017 [34]). These measurements correspond to two shallow-water harbor environments: La Spezia harbor, Italy, with a communication range of about 600 m, and Brest Commercial Harbor, France, with an 800 m link at 20 m water depth. Their main characteristics are summarized in Table 4.1 and illustrated in Figure 4.20. Other channels presented in (Walree et al., 2017 [34]) were also examined. However, they either did not produce a solution through Exhaustive Random search or provided nearly identical performance in most hyperparameter in the search space.

Table 4.1 – Channel Characteristics for SAUWAN and BCH1

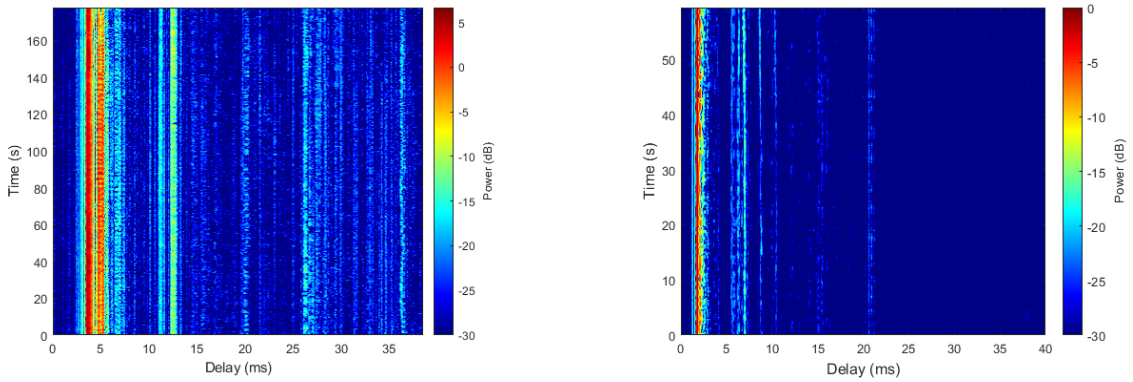
Parameter	SAUWAN Channel	BCH1 Channel
Bandwidth	4800Hz	5000Hz
Central frequency	11000Hz	35000Hz
Relative motion	0.0015m/s	0.0769m/s

No prior knowledge or assumptions regarding channel conditions were utilized. For SAUWAN, a 170-second recording enabled the transmission of 70 packets (each packet is 1.5 s in duration), whereas for BCH1, the 60-second recording enabled transmission of 24 packets. The SNR was consistently fixed at 12 dB for both channel replays.

Figure 4.21 illustrates the average PER, computed over 25 Monte Carlo runs, for *WandS*, the Channel-Estimation benchmark (utilizing a 511-symbol preamble), and the Exhaustive Random search benchmark. Panel (a) corresponds to the SAUWAN channel, and panel (b) corresponds to the BCH1 channel. The initial warm-up phase of *WandS* (consisting of 10 random samples) spans 8 packets, equivalent to 20 seconds. Subsequently, during the flywheel phase, *WandS* achieves immediate stabilization from the first packet for BCH1, while stabilization occurs after approximately 6 packets for SAUWAN. On

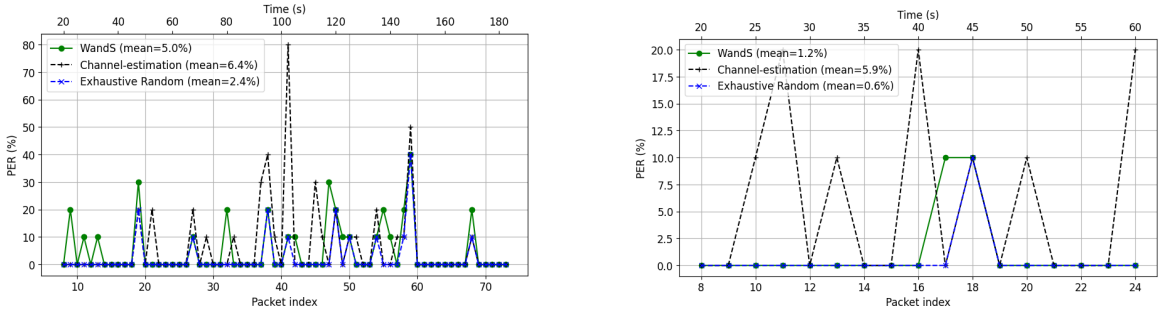
average, *WandS* outperforms the Channel-Estimation benchmark on the SAUWAN channel but remains inferior to the idealized Exhaustive Random Search benchmark. The benchmark itself is computationally infeasible in practice and only used as an idealized baseline.

These results substantiate that *WandS* effectively converges and performs reliably under realistic channel conditions without relying on any prior environmental information.



(a) (b)

Figure 4.20 – Power in time-delay domain. Panel (a) shows SAUWAN channel, and Panel (b) shows BCH1 channel.



(a) (b)

Figure 4.21 – PER evolution during communication with real channel replay. Panel (a) shows PER evolution for SAUWAN channel, and Panel (b) shows PER evolution for BCH1 channel.

4.6 Conclusion

This chapter introduced a Tracking method framework for tracking near-optimal receiver hyperparameters as the underwater acoustic channel evolves from packet to packet.

Section 4.4 first presented a single-agent flywheel routine, in which one agent performs one objective-function evaluation per received packet. Using a standard GP surrogate and a UCB policy, this proof-of-concept demonstrated clear gains over the Channel-Estimation benchmark on the simulated time-varying scenario of Section 4.3. However, three limitations remained. First, the surrogate was stationary despite the time-varying nature of the objective (Assumption 3). Second, using a single agent forces a fragile choice between exploration (to remain robust to channel changes) and exploitation (to stabilize performance), with no mechanism to handle regime shifts along the link. Third, hallucinations were not handled explicitly beyond the fixed hallucination penalty at -50 dB.

To address these limitations, Section 4.5 introduced a dual-agent Tracking method, based on a ***W**ise* agent and a ***S**cout* agent, hence the name ***WandS***. The Scout prioritizes exploration while the Wise prioritizes conservative exploitation, and both operate at packet rate under a shared time-varying surrogate model (TV-GP). Hallucination avoidance is reinforced through a dedicated penalty term that steers the search away from hallucination-prone regions, while the influence of past hallucinations fades over time consistently with Assumption 3 and with the TV-GP forgetting mechanism.

Overall, *WandS* improves packet reliability and robustness across a range of channels. It consistently outperforms the benchmarks and the single-agent counterpart without requiring prior environmental knowledge, and it consistently approaches the "Exhaustive Random Search" upper performance bound. The results also highlight the practical value of separating exploration and exploitation across two coordinated agents. When only one receiver instance is available, alternating between Scout and Wise decisions can still provide benefits, although the design of optimal alternation rules remains an open direction.

This chapter introduces original contributions of this PhD work. First, the *flywheel mode* is adapted to receiver hyperparameter tuning at packet rate, both in its single-agent form and in its multi-agent extension. Second, the Tracking method framework is coupled with a *time-varying surrogate model* (TV-GP). Third, hallucination handling is strengthened through a *time-varying hallucination penalization* whose influence decays in time.

Several extensions are possible. First, throughout this work we fixed the number of agents to two, the Wise and the Scout. Increasing the number of agents and quantifying how agent count affects performance and stability is a natural direction. Second, the agents rely on UCB and LCB policies. Although these policies reflect complementary design choices, they introduce their own tuning parameters. Exploring hyperparameter-

free alternatives, or policies with fewer tuning knobs, would simplify deployment.

Another important factor is the drift parameter of the TV-GP surrogate, which controls both the expected rate of objective variation and the effective forgetting speed. It is unclear whether the Wise and the Scout should share the same forgetting rate. Allowing distinct drift parameters per agent could improve tracking, for example if one agent benefits from slower forgetting while the other reacts more aggressively to change.

Third, we restricted the tuning scope to DFE and PLL hyperparameters as a proof-of-concept. In principle, *WandS* can be used to tune other receiver blocks, and extending the method to a broader set of hyperparameters would clarify which parts transfer directly and which require additional treatment.

We also relied on a TV-GP surrogate to handle time variation by discounting older observations. However, even when their influence becomes small, observations remain stored in $\mathcal{D}$. Alternative time-varying surrogate strategies could be considered, including periodic GP restarts (Zhao et al., 2020 [114]), and sliding-window updates that discard older observations (Zhou et al., 2021 [115]). More recent work proposes explicit deletion criteria based on observation relevance. In particular, (Bardou et al., 2024a [116]) propose a Wasserstein-distance-based criterion to quantify how much an observation matters for future predictions, enabling bounded dataset sizes while preserving predictive quality and improving sampling frequency in long-horizon dynamic optimization.

Finally, the current implementation uses a central coordinator to synchronize $\mathcal{D}$ and the shared surrogate, and to select among the hyperparameter proposals produced by the agents. This can be restrictive in settings where agents are distributed across mobile devices with limited communication. Extending *WandS*, or more generally the multi-agent BO framework, to fully decentralized coordination over multiple agents optimizing a shared objective remains an open direction that is addressed in Chapter 5.

Chapter 5

Decentralized multi-agent optimization

5.1 Introduction

The WandS multi-agent optimizer (see Section 4.5) is a centralized technique. The agents (Scout and Wise) report their tested hyperparameters and the corresponding receiver performances to a central node. This node aggregates the data, updates the shared surrogates used by WandS, and selects the next hyperparameters from the candidates proposed by the agents at each packet.

This chapter extends the multi-agent structure to a decentralized setting beyond UAC. Decentralized learning and control arise in multi-robot teams, mobile sensor networks, and embedded platforms, where full centralization is limited by bandwidth, energy constraints, and latency induced by hardware or the environment (Boyd et al., 2006 [117]; Gielis et al., 2022 [118]; Halsted et al., 2021 [119]; Li et al., 2016 [120]; Imbert et al., 2022 [121]; Anderson et al., 2021 [122]). In such systems, agents act online in a potentially noisy environment, selecting actions over continuous search spaces, without a central coordinator. Sample efficiency is important because each new environmental measurement can be costly.

The same constraints motivate decentralized coordination in underwater multi-agent systems. In swarms of AUVs, acoustic links are typically low-bandwidth and energy demanding, which favors sparse exchanges of compact messages rather than continuous communication (Fattah et al., 2020 [123]; Gussen et al., 2021 [124]). Centralized coordination is also difficult at scale because the channel exhibits strong range and frequency dependent attenuation (Thorp, 1967 [31]), together with Doppler and multipath effect (see Chapter 1), which are velocity and range dependent. As a result, maintaining robust links from a single coordinator to all vehicles over varying distances is challenging, and would require additional equipment that is itself subject to the same channel limitations.

In UAC receiver hyperparameter tuning, communication is costly and energy auton-

omy is a primary operational constraint, which motivates modeling coordination through a decentralized architecture with periodic, low-payload information exchange.

The goal in this chapter is to design a coordination mechanism for decentralized agents in such settings. This chapter broadens the scope beyond UAC and opens a more general perspective. We cast the problem in the bandit framework (Section 5.1.1) and derive theoretical convergence guarantees through regret analysis (Section 5.1.2).

5.1.1 Bandit problems

The proposed algorithm is developed and analyzed within the multi-armed bandit framework. Consider an action set of arms $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Pulling an arm $\boldsymbol{\theta}$ yields a stochastic reward with an unknown distribution. Let $\mathcal{R}_{\boldsymbol{\theta}}(t)$ denote the reward observed when selecting arm $\boldsymbol{\theta}$ at time t. An agent selects arms sequentially up to a horizon T using a policy that specifies which arm to choose at each time step, with the goal of maximizing the cumulative reward over all selections. In the UAC receiver hyperparameter tuning problem, an arm corresponds to a hyperparameter configuration, and the reward corresponds to the objective function defined in Chapter 1.

Bandit problems are a model for sequential decision making with many applications, including clinical trials for efficient treatment discovery, online pricing under demand uncertainty, and recommendation systems (Bouneffouf et al., 2020 [125]). They are closely related to Bayesian optimization in the sense that Bayesian optimization aims to identify high-performing arms with few evaluations by trading off exploration of the search space and exploitation of current knowledge (Garnett, 2023a [12]).

Multi-armed bandits can be divided into discrete and continuous settings. In the discrete case, arms are chosen from a finite set $\Theta = \{\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(M)}\}$, where M is the number of arms. Let $\mu(\boldsymbol{\theta}) = \mathbb{E}[\mathcal{R}_{\boldsymbol{\theta}}]$ and let $\mu^\star = \max_{\boldsymbol{\theta}\in\Theta} \mu(\boldsymbol{\theta})$. A standard performance metric is the pseudo-regret

$$\bar{R}_T = \sum_{t=1}^{T} \Big(\mu^\star - \mu(\boldsymbol{\theta}_t)\Big), \tag{5.1}$$

where $\boldsymbol{\theta}_t$ is the arm selected at time t (Bubeck et al., 2012 [126]). A naive strategy estimates each mean by

$$\hat{\mu}_{\boldsymbol{\theta}} = \frac{1}{n_{\boldsymbol{\theta}}} \sum_{i=1}^{n_a} \mathcal{R}_{\boldsymbol{\theta}}(i), \tag{5.2}$$

where $n_{\boldsymbol{\theta}}$ is the number of times arm $\boldsymbol{\theta}$ has been selected and $\mathcal{R}_{\boldsymbol{\theta}}(i)$ denotes the i-th observed reward obtained from arm $\boldsymbol{\theta}$. The agent then chooses the arm with the largest estimate. This can fail when some arms are under-explored. More robust strategies explicitly trade exploration-exploitation, for example Thompson sampling, which samples

actions from a posterior distribution over rewards (Agrawal et al., 2013 [127]), as well as UCB policies.

In its classic discrete form, rewards are treated as independent across arms, so learning each arm separately becomes inefficient when M is large. In this work, and throughout the previous chapters, the search space of hyperparameters is continuous. To make learning feasible, we assume the reward function has structure, which is often the case in practice (Garnett, 2023a [12]). A common way to encode structure with sufficient flexibility is to use a GP prior.

GP bandits (Srinivas et al., 2012 [94]) provide a principled exploration–exploitation tradeoff with finite-time guarantees under structured reward functions and are used across a range of applications (Garnett, 2023a [12]; Parker-Holder et al., 2020 [128]). In the static setting, the sequential observation model is

$$y_t = f(\boldsymbol{\theta}_t) + \varsigma_t, \qquad t \geq 1, \tag{5.3}$$

where ς_t is Gaussian noise and f is modeled as a GP specified by a mean and a kernel. The GP prior induces correlation across nearby arms, so observing one arm provides information about neighboring arms. The cumulative regret is

$$R_T = \sum_{t=1}^{T} r_t, \tag{5.4}$$

where $r_t = f(\boldsymbol{\theta}^\star) - f(\boldsymbol{\theta}_t)$ is the instantaneous regret, $\boldsymbol{\theta}^\star \in \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f(\boldsymbol{\theta})$, and $\boldsymbol{\theta}_t$ is the arm selected at time t. This quantity penalizes missed opportunities to select the optimal arm throughout the search.

5.1.2 Regret analysis

A main advantage of the bandit framework is that it supports regret analysis, namely the study of how the cumulative regret R_T scales with time under a given policy. In (Srinivas et al., 2012 [94]), the GP-UCB policy is introduced, using the UCB acquisition rule presented in Chapter 2, Section 2.3.3, and it satisfies a no-regret property.

Definition 13 *A policy is said to have the no-regret property if the average cumulative regret vanishes with time*

$$\frac{R_T}{T} = \frac{1}{T}\sum_{t=1}^{T}\Big(f(\boldsymbol{\theta}^\star) - f(\boldsymbol{\theta}_t)\Big) \xrightarrow[T\to\infty]{} 0, \tag{5.5}$$

where $\boldsymbol{\theta}^\star$ is an optimal arm.

Moreover, (Bogunovic et al., 2016b [129]) introduced a time-varying variant of GP-UCB, termed TV-GP-UCB, in which the reward function depends on time, written as f_t. The model uses a spatio-temporal GP kernel, as presented in Chapter 4, Section 4.5. Under controlled drift assumptions, the relevant notion is dynamic regret

$$\frac{R_T}{T} = \frac{1}{T}\sum_{t=1}^{T}\Big(f_t(\boldsymbol{\theta}_t^\star) - f_t(\boldsymbol{\theta}_t)\Big) \xrightarrow[T\to\infty]{} 0, \tag{5.6}$$

where $\boldsymbol{\theta}_t^\star \in \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f_t(\boldsymbol{\theta})$ denotes the optimal arm at time t. This highlights that, in the time-varying case, the objective is to track the evolving maximizer rather than to identify a single fixed optimum.

5.2 Problem and Notation

We propose X-KB-UCB, a decentralized UCB policy designed to coordinate exploration without centralized planning. The method relies on gossip, a cooperation protocol in which agents exchange small messages that convey partial information about the reward function. We consider a setting where gossip occurs periodically at fixed rounds. At each gossip round, the method applies a cross-agent Kriging–Believer (KB) update (see Section 5.3.2), which motivates its name.

The proposed policy is closely related to existing GP bandit methods. In our framework, outside gossip rounds each agent reduces to GP-UCB in the static case and to TV-GP-UCB in the time-varying case.

We consider multiple agents $a \in \mathcal{A}$ that cooperate via gossip to optimize a common time-indexed reward $f_t : \boldsymbol{\Theta} \to \mathbb{R}$ over a continuous action space $\boldsymbol{\Theta} \subset \mathbb{R}^d$.

Time proceeds in discrete iterations $t = 1, \dots, T_{\text{iter}}$. At iteration t, agent a selects an arm $\boldsymbol{\theta}_{a,t} \in D$ and observes an immediate noisy reward

$$y_{a,t} = f_t(\boldsymbol{\theta}_{a,t}) + \varsigma_{a,t}, \qquad \varsigma_{a,t} \overset{\text{i.i.d.}}{\sim} \mathcal{N}(0, \sigma_f^2). \tag{5.7}$$

Communication is available only at iterations that are multiples of a fixed period $t_g \in \{1, \dots, T_{\text{iter}}\}$. These are the gossip rounds, during which each agent sends a constant-size message (its most recent $(\boldsymbol{\theta}_{a,t-1}, y_{a,t-1}, t-1)$) to its neighbors. The parameter t_g controls the communication budget: smaller t_g yields more frequent gossip, larger t_g less frequent gossip.

At iterations $t \in [1, T_{\text{iter}}]$ with $t \bmod t_g = 0$, each agent broadcasts only the previous observation $(\boldsymbol{\theta}_{a,t-1}, y_{a,t-1}, t-1)$. $N_g = \lfloor T_{\text{iter}}/t_g \rfloor$ is the number of gossip rounds. Let

$P \in [0,1]^{|\mathcal{A}|\times|\mathcal{A}|}$ denote a (possibly directed) gossip matrix governing communication: at each gossip round t, agent a receives a tuple from agent v with probability $P(a,v)$. The number of tuples received by agent a in round t is modeled, within this work, as

$$\eta_{a,t} = \sum_{v\in\mathcal{A}, v\neq a} B_{a,v}, \quad B_{a,v} \sim \text{Bernoulli}(P(a,v)) \tag{5.8}$$

Denote the expected number received tuples per gossip round t by

$$N_a := \mathbb{E}[\eta_{a,t}] = \sum_{v\neq a, v\in\mathcal{A}} P(a,v). \tag{5.9}$$

The effective number of reward evaluations available to agent a is

$$T_\eta = T_{\text{iter}} + \sum_{k=1}^{N_g} \eta_{a,k}. \tag{5.10}$$

Each agent searches for the best arm in a continuous domain $D \subset \mathbb{R}^d$. We formalize the two settings regarding the reward function as assumptions.

Assumption 4 *The environment is static: $f_t \equiv f$ on D,*

$$f \sim \mathcal{GP}(0,k) \tag{5.11}$$

k being the kernel function.

Assumption 5 *The environment is time-varying and follows*

$$f_0(\boldsymbol{\theta}) = g_0(\boldsymbol{\theta}), \tag{5.12}$$

$\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \forall t \in \{1,\dots,T_{iter}\}$,

$$f_t(\boldsymbol{\theta}) = \sqrt{1-\epsilon} f_{t-1}(\boldsymbol{\theta}) + \sqrt{\epsilon} g_t(\boldsymbol{\theta}), \tag{5.13}$$

with $g_t \sim \mathcal{GP}(0,k)$ independent across t, forgetting factor $\epsilon \in [0,1]$, and k the kernel function (Bogunovic et al., 2016b [129]).

Under Assumption 4, each agent aims to identify

$$\boldsymbol{\theta}^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f(\boldsymbol{\theta}). \tag{5.14}$$

Under Assumption 5, the goal is to track the instantaneous maximizer, $\forall t \in \{1,\dots,T_{\text{iter}}\}$:

$$\boldsymbol{\theta}_t^* = \arg\max_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} f_t(\boldsymbol{\theta}), \tag{5.15}$$

5.3 X-KB-UCB

Our bandit method leverages GP-UCB (Srinivas et al., 2012 [94]) under Assumption 4 and TV-GP-UCB (Bogunovic et al., 2016b [129]) under Assumption 5. We first recall the GP surrogate used by each agent and the candidate arm selection technique (Section 5.3.1), then we describe our Kriging–Believer (KB) UCB decision rule (Section 5.3.2).

5.3.1 Arm selection

At each time step t, every agent chooses an arm $\boldsymbol{\theta}_{a,t}$. Each agent builds its own surrogate of the reward function from its history: $\boldsymbol{X}_{a,t-1} = [\boldsymbol{\theta}_{a,1}, \dots, \boldsymbol{\theta}_{a,t-1}]$ and $\boldsymbol{Y}_{a,t-1} = [y_{a,1}, \dots, y_{a,t-1}]$.

Under Assumption 4, agent a's surrogate is a Gaussian process, fully specified by its posterior mean $\mu_{a,t}$ and variance $\sigma^2_{a,t}$, for any $x \in D$:

$$\mu_{a,t}(\boldsymbol{\theta}) = \boldsymbol{k}_{t-1}(\boldsymbol{\theta})^\top (\boldsymbol{K}_{t-1} + \sigma_f^2 \boldsymbol{I}_{t-1})^{-1} \boldsymbol{Y}_{a,t-1}, \tag{5.16}$$

$$\sigma^2_{a,t}(\boldsymbol{\theta}) = k(\boldsymbol{\theta}) - \boldsymbol{k}_{t-1}(\boldsymbol{\theta})^\top (\boldsymbol{K}_{t-1} + \sigma_f^2 \boldsymbol{I}_{t-1})^{-1} \boldsymbol{k}_{t-1}(\boldsymbol{\theta}). \tag{5.17}$$

Here σ_f^2 is the noise variance, $\boldsymbol{k}_{t-1}(\boldsymbol{\theta}) = [k(\boldsymbol{\theta}_i, \boldsymbol{\theta})]_{i=1}^{t-1}$, $\boldsymbol{K}_{t-1} = [k(\boldsymbol{\theta}, \boldsymbol{\theta}')]_{\boldsymbol{\theta},\boldsymbol{\theta}' \in \boldsymbol{X}_{a,t-1}}$, and $\boldsymbol{I}_{t-1}$ is the $(t-1) \times (t-1)$ identity. We assume $k(\boldsymbol{\theta}, \boldsymbol{\theta}) \leq 1$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. We use two standard kernels for analysis, for all $(\boldsymbol{\theta}, \boldsymbol{\theta}') \in \boldsymbol{\Theta}^2$:

(Bogunovic et al., 2016b [129]) extend (5.16)–(5.17) to the time-varying setting of Assumption 5 using a separable space–time kernel:

$$k^{dyn}(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) = k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j) \cdot k_s(i, j), \tag{5.18}$$

where $k_s(i, j) = (1 - \epsilon)^{|i-j|/2}$,, and $i, j \geq 1$

When $t \bmod t_g \neq 0$, the next arm is selected by maximizing the UCB:

$$\boldsymbol{\theta}_{a,t} = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mu_{a,t}(\boldsymbol{\theta}) + \alpha_t^{1/2} \sigma_{a,t}(\boldsymbol{\theta}). \tag{5.19}$$

The schedule α_t balances exploration and exploitation and is chosen to guarantee high-probability regret bounds (see Section 5.4).

5.3.2 Cross-Agent Kriging–Believer

In a multi-agent setting, some mechanism is needed to coordinate exploration across agents. Indeed, if all agents share the same GP prior, observe the same history through

gossip, and use the same UCB schedule, then perfect maximization of the acquisition function would make them select the same query points at each iteration. In that case, the agents would collect redundant observations and gossip would provide little additional benefit. For this reason, we incorporate an explicit cross-agent coordination mechanism via a cross-agent KB construction, which promotes diverse queries.

At a gossip round, agent a receives

$$S_{a,t} = \{(\boldsymbol{\theta}_{v,t-1}, y_{v,t-1}, t-1)\}_{v \in \mathcal{A}_a \subset \mathcal{A}},$$

where $\mathcal{A}_a$ is the subset of agents whose messages reached a (as determined by P).

KB is widely used to select multiple nonredundant candidates in batch BO (Desautels et al., 2014 [130]; Parker-Holder et al., 2020 [128]). We adapt it to the gossip setting: freeze the mean $\mu_{a,t}$, insert the received arms into the design, recompute the variance, and then pick the next arm. More specifically, at a gossip round

$$\boldsymbol{X}_{a,t} = [\boldsymbol{\theta}_{a,1}, \ldots, \boldsymbol{\theta}_{a,t-1}] \cup [\boldsymbol{\theta}_{v,t-1}]_{v \in \mathcal{A}_a}, \tag{5.20}$$

compute $\widehat{\sigma}^2_{a,t}$ from (5.17) using $\boldsymbol{X}_{a,t}$ (with the mean frozen), and maximize

$$\boldsymbol{\theta}_{a,t} = \arg\max_{\boldsymbol{\theta} \in \boldsymbol{\Theta}} \mu_{a,t}(\boldsymbol{\theta}) + \beta_t^{1/2} \widehat{\sigma}_{a,t}(\boldsymbol{\theta}). \tag{5.21}$$

This encourages $\boldsymbol{\theta}_{a,t}$ to differ from the received arms in $S_{a,t}$ while preserving UCB-driven exploration. The schedule β_t is adapted to gossip rounds to compensate for KB overconfidence (Section 5.4).

In summary, as shown in Algorithm 4, at a gossip round t, agent a (i) freezes its posterior mean $\mu_{a,t}$, (ii) recomputes the posterior variance $\widehat{\sigma}_{a,t}$ by appending the arms of the received tuples $S_{a,t} = \{(\boldsymbol{\theta}_{v,t-1}, y_{v,t-1}, t-1)\}_{v \in \mathcal{A}_a}$ to its design, and (iii) selects its candidate $\boldsymbol{\theta}_{a,t}$ via (5.21). Crucially, after selecting $\boldsymbol{\theta}_{a,t}$, agent a updates its mean function using (5.16) after assimilating both the received tuples and its new sample. After acquisition, the received tuples $S_{a,t} = \{(\boldsymbol{\theta}_{v,t-1}, y_{v,t-1}, t-1)\}$ are inserted with their original timestamps, and the new tuple $(\boldsymbol{\theta}_{a,t}, y_{a,t}, t)$ is added, thus $\mathcal{D}_a \leftarrow \mathcal{D}_a \cup S_{a,t} \cup \{(\boldsymbol{\theta}_{a,t}, y_{a,t}, t)\}$.

Discussion. Cross-Agent UCB (X-KB-UCB) is a novel algorithm introduced for the first time in this work, to the best of the authors' knowledge. It fundamentally differs from centralized batch selection methods such as GP-BUCB (Desautels et al., 2014 [130]) and PB2 (Parker-Holder et al., 2020 [128]). In batch BO, one plans a set of B arms and uses KB to collapse posterior variance around those same future candidates within the batch, explicitly pushing the next picks away from them. In X-KB-UCB, at time t we pick a single arm while incorporating a set of received arms from time $t-1$. We then freeze the mean and update only the variance using those received arms.

Algorithm 4 X-KB-UCB (stationary/time-varying reward)

Input: Domain D, gossip period t_g, schedules α_t, β_t, gossip matrix P
Initialize $\mathcal{D}_a \leftarrow \emptyset$ for each agent $a \in \mathcal{A}$
for $t = 1, 2, \ldots, T_{\text{iter}}$ **do**
  if $t \bmod t_g = 0$ **then**
    / gossip round */*
    All agents broadcast $(\boldsymbol{\theta}_{a,t-1}, y_{a,t-1}, t-1)$
    Agent a receives $\{(\boldsymbol{\theta}_{v,t-1}, y_{v,t-1}, t-1)\}_{v \in \mathcal{A}_a}$ and forms $S_{a,t}$
    Compute the candidate $\boldsymbol{\theta}_{a,t}$ using (5.21)
    Sample $y_{a,t} = f(\boldsymbol{\theta}_{a,t}) + \varsigma_{a,t}$
    $\mathcal{D}_{a,t} \leftarrow \mathcal{D}_{a,t} \cup S_{a,t} \cup \{(\boldsymbol{\theta}_{a,t}, y_{a,t}, t)\}$
  else
    Compute the candidate $\boldsymbol{\theta}_{a,t}$ using (5.19)
    Sample $y_{a,t} = f(\boldsymbol{\theta}_{a,t}) + \varsigma_{a,t}$
    $\mathcal{D}_{a,t} \leftarrow \mathcal{D}_{a,t} \cup \{(\boldsymbol{\theta}_{a,t}, y_{a,t}, t)\}$
  end if
  Update the mean with $D_{a,t}$ using (5.16)
end for

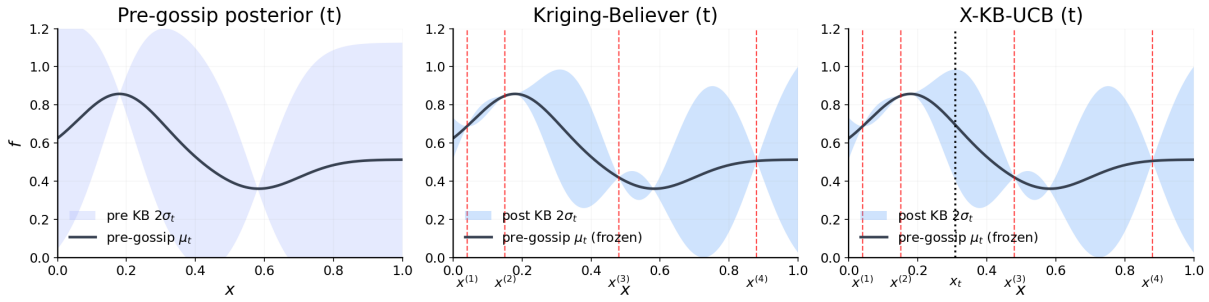


Figure 5.1 – Illustration of the gossip update used by X-KB-UCB at a fixed time t (here $\varepsilon = 0$, no temporal carryover). **Left:** pre-gossip posterior with mean μ_t (gray) and uncertainty band $\pm 2\sigma_t$ (light blue). At the **middle (KB):** Kriging–Believer update using the received evaluations $\{x^{(1)}, \ldots, x^{(4)}\}$: the mean μ_t is frozen and only the standard deviation is updated (blue band). **Right (X-KB-UCB):** action selection with UCB computed from the frozen μ_t and the post-gossip standard deviation, yielding the next query $\boldsymbol{\theta}_t$.

Under Assumption 5, the time indices matter: received arms correspond to $t-1$, and the Markov-drift prior (5.13) reintroduces uncertainty at time t. The temporal innovation yields a nonzero variance floor at time t even if an arm x was just observed at $t-1$. Hence X-KB-UCB updates (but does not collapse) the uncertainty around received points; larger drift (larger ϵ) naturally increases this residual uncertainty. This contrasts with PB2/GP-BUCB-style intra-batch variance collapse.

An illustration of the X-KB-UCB is shown in Figure 5.1, which summarizes the technique described in this section using a toy example.

5.4 Theoretical results

We proceed as follows. First, in Section 5.4.1 we introduce a purely analytical device used to quantify the theoretical benefit of gossip between agents referred to as the ***augmented-agent***. Next, in Section 5.4.2 we review the maximum information gain, a central quantity in GP regret analyses. We then discuss the overconfidence issue induced by Kriging–Believer in Section 5.4.3. Finally, we state the main theorems in Section 5.4.4.

5.4.1 Augmented agent regrets

We depart from the classic single-agent sequential (GP-UCB/TV-GP-UCB) and batch (GP-BUCB/PB2) settings and adopt an agent-centric notion of feedback that counts both locally played tuples and tuples received via gossip. In this abstraction, we define the following object.

Definition 14 *An **augmented agent** is an agent whose information set is augmented by the data it receives from its neighbors.*

The augmented agent is responsible for both what it actively queries and what it passively receives. If an agent receives from all neighbors at all iterations, its augmented version operates close to full information and, in a static scenario, is close in spirit to a batch GP method such as GP-BUCB, while its time-varying counterpart is closer to a time-varying batch variant such as PB2 (Parker-Holder et al., 2020 [128]). Conversely, if the agent receives no tuples from other agents, the augmented agent reduces to the purely local setting. This construction is meaningful in our framework because the agent is assumed to accept any tuples delivered by gossip whenever communication allows it. The augmented-agent view is a mathematical device, introduced in this work, that enables a regret analysis under gossip sharing. Throughout the remainder of this work, all regret statements are with respect to augmented agents.

$$\mathcal{T}_g := \Big\{ t \in \{1, \dots, T_{\text{iter}}\} : t \bmod t_g = 0 \Big\}. \tag{5.22}$$

At a gossip round $t \in \mathcal{T}_g$, agent a receives

$$S_{a,t} := \Big\{ (v, \boldsymbol{\theta}_{v,t-1}, t-1) : v \in \mathcal{A}_a(t) \Big\} \tag{5.23}$$

with $\eta_{a,t}$, as defined in (5.8) is $|S_{a,t}|$. Define the agent's tuple collection

$$\mathcal{Z}_a := \Big\{(a, \boldsymbol{\theta}_{a,\tau}, \tau) : \tau = 1, \dots, T_{\text{iter}}\Big\} \cup \bigcup_{t \in \mathcal{T}_g} S_{a,t}. \tag{5.24}$$

Its size is denoted T, and satisfies :

$$\mathbb{E}\Big[\,|\mathcal{Z}_a|\,\Big] = T_{\text{iter}} + \sum_{t \in \mathcal{T}_g} \mathbb{E}[\eta_{a,t}] = T_{\text{iter}} + N_g N_a := T. \tag{5.25}$$

Under Assumption 4,

$$R^{st}_{a,T} := \mathbb{E}\left[\sum_{(i,x,\tau) \in \mathcal{Z}_a} \Big(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta})\Big)\right]. \tag{5.26}$$

Under Assumption 5,

$$R^{dyn}_{a,T} := \mathbb{E}\left[\sum_{(i,\boldsymbol{\theta},\tau) \in \mathcal{Z}_a} \Big(f_\tau(\boldsymbol{\theta}^*_\tau) - f_\tau(\boldsymbol{\theta})\Big)\right]. \tag{5.27}$$

These are augmented agents' cumulative regrets: they include tuples obtained via gossip as well as locally decided ones, showcasing the benefit of gossip with an expected N_a received tuples per gossip round.

We focus on the upper bounds on this cumulative regrets in (5.26) and (5.27) that imply no-regret for our X-KB-UCB method.

To establish these upper bounds, we have the following assumption:

Assumption 6 *There exist $q_1, q_2 \geq 0$ such that for all $L \geq 0$,*

$$\Pr\left(\sup_{x \in D} \left|\frac{\partial f}{\partial x^{(j)}}\right| \geq L\right) \leq q_1 \exp\left(-\left(\tfrac{L}{q_2}\right)^2\right), \quad j = 1, \dots, d, \tag{5.28}$$

with $f \sim \mathcal{GP}(0, k)$. Here $\frac{\partial f}{\partial x^{(j)}}$ refers to the partial derivative of f with respect to its jth elements.

Assumption 6 is a standard smoothness tail bound (used in (Srinivas et al., 2012 [94]; Bogunovic et al., 2016b [129])). The SE and Matérn kernels (with $\nu > 2$) in (3)–(4) satisfy this assumption.

Before the main theorems, we recall two quantities that drive the regret bounds: (i) the *maximum information gain* and (ii) the *KB overconfidence* adjustment at gossip rounds.

5.4.2 Maximum information gain

The first key quantity is the maximum information gain after B evaluations:

$$\gamma_B = \max_{X \subset D, |X|=B} I(\boldsymbol{f}_X; \boldsymbol{y}_X), \tag{5.29}$$

where $\boldsymbol{f}_X = [f(\boldsymbol{\theta})]_{x \in X}$ and $\boldsymbol{y}_X = [f(\boldsymbol{\theta}) + \varsigma_x]_{x \in X}$ with independent noise $\varsigma_x \sim \mathcal{N}(0, \sigma_f^2)$. For Gaussian processes,

$$I(\boldsymbol{f}_X; \boldsymbol{y}_X) = \tfrac{1}{2} \log \det \left(\boldsymbol{I} + \sigma_f^{-2} \boldsymbol{K}_X \right) \tag{5.30}$$

with $\boldsymbol{K}_X = [k(x, x')]_{x,x' \in X}$ and k the kernel (k^{dyn} under Assumption 5). Under Assumption 5, the same identity holds with the space–time Gram matrix $\boldsymbol{K}_X^{dyn} [k^{dyn}(x, x')]_{x,x' \in X}$.

5.4.3 KB overconfidence

The sequential GP confidence band is

$$CI_t(\boldsymbol{\theta}) = \left[\mu_t(\boldsymbol{\theta}) \pm \alpha_t^{1/2} \sigma_t(\boldsymbol{\theta}) \right]. \tag{5.31}$$

KB reduces posterior variance using received arms at gossip rounds. The resulting band,

$$CI_{t_g}(\boldsymbol{\theta}) = \left[\mu_t(\boldsymbol{\theta}) \pm \beta_t^{1/2} \widehat{\sigma}_t(\boldsymbol{\theta}) \right], \tag{5.32}$$

can be narrower, potentially breaking the inclusion

$$f(\boldsymbol{\theta}) \in CI_t(\boldsymbol{\theta}) \Rightarrow f(\boldsymbol{\theta}) \in CI_{t_g}(\boldsymbol{\theta}) \tag{5.33}$$

needed to inherit both the sequential GP-UCB and TV-GP-UCB proofs. Following (Desautels et al., 2014 [130]), we prevent this by inflating β_t so that $CI_t(\boldsymbol{\theta}) \subseteq CI_{t_g}(\boldsymbol{\theta})$ at gossip rounds. For all $x \in D$,

$$I\Big(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1} \Big) = \tfrac{1}{2} \log \Big(\frac{\sigma_t^2(\boldsymbol{\theta})}{\widehat{\sigma}_t^2(\boldsymbol{\theta})} \Big). \tag{5.34}$$

Thus, if $I(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1}) \leq C_a$ then $\sigma_t(\boldsymbol{\theta}) \leq e^{C_a} \widehat{\sigma}_t(\boldsymbol{\theta})$. Choosing

$$\beta_t = e^{2C_a} \alpha_t$$

at gossip rounds yields $\alpha_t^{1/2} \sigma_t(\boldsymbol{\theta}) \leq \beta_t^{1/2} \widehat{\sigma}_t(\boldsymbol{\theta})$ and restores the inclusion. An upper bound on C_a is presented in Lemma. 1, and the proof is provided in the Appendix 4.

Lemma 1 *Let $\mathcal{A}$ denote the set of agents. At gossip round t, let $S_{a,t}$ be the set of tuples received by agent a, let $\boldsymbol{y}_{S_{a,t}}$ denote the corresponding noisy rewards, with noise variance σ_f^2, and let $\boldsymbol{y}_{1:t-1}$ denote the rewards observed by agent a up to time $t-1$ (including previously gossiped data). Then, for all $x \in D$,*

$$I\Big(f_t(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1}\Big) \leq \gamma_{n_{a,t}} \tag{5.35}$$

$$\leq \frac{|\mathcal{A}|-1}{2\sigma_f^2}. \tag{5.36}$$

where $n_{a,t} = |S_{a,t}|$. In particular, choosing $C_a = \frac{|\mathcal{A}|-1}{2\sigma_f^2}$ guarantees the band inclusion at gossip rounds.

We use schedules that certify the uniform confidence event

$$\Pr\left(\max_{1\leq t\leq T} \sup_{x\in D} |f_t(\boldsymbol{\theta}) - \mu_{a,t}(\boldsymbol{\theta})| \leq \sqrt{\alpha_t}\,\sigma_{a,t}(\boldsymbol{\theta})\right) \geq 1-\delta, \tag{5.37}$$

obtained via the standard discretization and time–grid union bound (Srinivas et al., 2012 [94]; Bogunovic et al., 2016b [129]). The specific information-gain bounds are schedule-independent. At gossip rounds we inflate widths to preserve band inclusion.

One can keep the uniform cap C_a from Lemma 1 and use the inflation factor $\exp(2C_a)$. Alternatively, let $S_{a,t}$ be the received set at round t. It can be shown that the exact mutual information is (see Appendix 6.2.1)

$$C_a \leq I(f_{S_{a,t}}; \boldsymbol{y}_{S_{a,t}}) = \frac{1}{2}\log\det(I + \sigma_f^{-2} K_{S_{a,t}}) = C_{a,t}. \tag{5.38}$$

Using $C_{a,t}$ in place of a uniform C_a leads to a tight, per-round inflation factor $\exp(2C_{a,t})$. Since $|S_{a,t}| \approx N_a$, this can be computed efficiently via incremental Cholesky updates. If one wants a single worst-case budgeted cap for sets of size at most m, one can approximate $\max_{|S|\leq m} I(f_S; \boldsymbol{y}_S)$ via greedy maximization, which provides a $(1-1/e)$-approximation due to submodularity, as in the information-gain analysis of GP-UCB (Srinivas et al., 2012 [94]).

Remark. C_a inflates α_t at gossip rounds to ensure the theoretical band inclusion. Our implementation uses a common schedule across all iterations and sets $\beta_t = \alpha_t$. This simplifies tuning, while the theoretical guarantees correspond to the inflated version.

5.4.4 Main theorems

We now present regret bounds for decentralized GP bandits running X-KB-UCB, for both stationary rewards (Thm. 1) and time-varying rewards (Thm. 2). The bounds apply

to a fully decentralized setting and account for rewards that drift over time. They are stated for augmented agents, meaning each agent is analyzed together with the tuples it receives through gossip, which makes explicit how performance shifts from the no-gossip regime toward the near-centralized regime as gossip becomes more informative.

Theorem 1 *Let $D \subset [0,r]^d$ be compact and convex, $d \in \mathbb{N}$, $r > 0$. Let k be a kernel such that $f \sim \mathcal{GP}(0,k)$ satisfies Assumptions 4 (static reward) and 6 (smoothness tail bound). Fix $\delta \in (0,1)$ and $C_a \geq 0$ with $I(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1}) \leq C_a$ at gossip rounds (all agents are* peers*: they all run X-KB-UCB with the same schedules and exchange their previous UCB-selected* arms*). Let α_t be the exploration schedule defined as*

$$\alpha_t^{\mathrm{UCB}} = 2\log\Big(\frac{4\pi t^2}{3\delta}\Big) + 2d\log\Big(t^2\, d\, q_2\, r\sqrt{\log\Big(\frac{8d\, q_1}{\delta}\Big)}\Big),$$

, and set

$$\beta_t = \begin{cases} \alpha_t, & t \bmod t_g \neq 0, \\ e^{2C_a}\alpha_t, & t \bmod t_g = 0. \end{cases}$$

Then, for each agent $a \in \mathcal{A}$, with probability at least $1-\delta$,

$$R_{T,a}^{st} \leq \sqrt{2C_1\Big([T+\phi_g]\alpha_{T+\phi_g}\gamma_{T+\phi_g}\Big(1+e^{2C_a}\Big)\Big)} + 4, \tag{5.39}$$

where $T = T_{iter} + N_a N_g$, $C_1 = \frac{8}{\log(1+\sigma_f^{-2})}$, and $\phi_g = N_a(T_{iter} - N_g)$ is the gossip gap. This regret bound yields the no-regret property of X-KB-UCB in static environments.

Here, N_a and N_g are defined around (5.10) and (5.9). The proof is provided in the Appendix 6.2. The key point is that, by the Gaussian Mutual Information identity (5.30) the classic and batch confidence intervals are contained in our inflated band at gossip rounds. Also, at non-gossip rounds $\beta_t = \alpha_t$ and the KB variance does not enlarge the band. Therefore we can upper bound the non-gossip rounds part of the regret by the GP-UCB bound (with horizon T) and the gossiped part by the GP-BUCB bound (with horizon $T_B = T_{\mathrm{iter}} + N_a T_{\mathrm{iter}} = T + \phi_g$).

Baselines: if $N_a = 0$ (no information sharing, $P = [0]^{|\mathcal{A}|\times|\mathcal{A}|}$) then $C_a = 0$ (since $S_{a,t} = \emptyset$) and $T = T_{\mathrm{iter}}$, recovering the classic GP-UCB upper bound. If $N_g = 1$ (one gossip), then $T + \phi_g = (T_{\mathrm{iter}}+1)N_a$, i.e., the largest gap between a single agent GP-UCB on T_{iter} arms and a one-shot GP-BUCB with batch size $1 + N_a$. If $N_g = T_{\mathrm{iter}}$ (gossip every round), then $\phi_g = 0$ and the upper bound behaves like that of GP-BUCB with batch $1 + N_a$. The term $\gamma_{T+\phi_g}$ is the maximum information gain; the bound inherits its standard rates (Srinivas et al., 2012 [94]).

Theorem 2 *Let $D \subset [0,r]^d$ be compact and convex, $d \in \mathbb{N}$, $r > 0$. Let k be a kernel such that $f \sim \mathcal{GP}(0,k)$ satisfies Assumptions 5 (time-varying reward) and 6. Fix $\delta \in (0,1)$ and $C_a \geq 0$ with $I(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_t) \leq C_a$. Define*

$$\alpha_t := \max\left\{ \alpha_t^{\mathrm{UCB}},\ \alpha_t^{\mathrm{TV}} \right\}. \tag{5.40}$$

where

$$\alpha_t^{\mathrm{UCB}} = 2\log\!\left(\frac{4\pi t^2}{3\delta}\right) + 2d\log\!\left(t^2\, d\, q_2\, r\, \sqrt{\log\!\left(\frac{8d\, q_1}{\delta}\right)}\right), \tag{5.41}$$

$$\alpha_t^{\mathrm{TV}} = 2\log\!\left(\frac{\pi^2 t^2}{2\delta}\right) + 2d\log\!\left(r\, d\, q_2\, t^2\, \sqrt{\log\!\left(\frac{d\, q_1\, \pi^2 t^2}{2\delta}\right)}\right), \tag{5.42}$$

and use the same schedule $\beta_t = \alpha_t$ when $t \bmod t_g \neq 0$ and $\beta_t = e^{2C_a}\alpha_t$ when $t \bmod t_g = 0$. Then, for each agent $a \in \mathcal{A}$, with probability at least $1-\delta$, and for any $\tilde{N} \in \{1,\dots,T\}$,

$$R_{a,T}^{dyn} \leq \sqrt{2C_1\left(\gamma_{\tilde{N}} + \tilde{N}^3\epsilon\right)\left[M_T + M_{T+\phi_g}\right]} + 4 \tag{5.43}$$

with

$$M_T = T\alpha_T\left(\frac{T}{\tilde{N}} + 1\right) \tag{5.44}$$

$$M_{T+\phi_g} = (T+\phi_g)e^{2C_a}\alpha_{T+\phi_g}\left(\frac{T+\phi_g}{\tilde{N}} + 1\right) \tag{5.45}$$

where $T = T_{iter} + N_a N_g$, $C_1 = \frac{8}{\log(1+\sigma_f^{-2})}$, $\phi_g = N_a(T_{iter} - N_g)$ is the gossip gap, and ϵ is the forgetting factor in (5.13).

The proof is similar to that of Theorem 1, and leverages the regret bounds derived in (Bogunovic et al., 2016b [129]) and (Desautels et al., 2014 [130]). In the time-varying GP model of (Bogunovic et al., 2016b [129], Cor. 4.1), the average regret vanishes whenever the drift decreases polynomially with the horizon: if $\epsilon \leq T^{-p}$ for some $p > 0$, then $R_{a,T}^{\mathrm{dyn}}/T \to 0$. Conversely, for any fixed $\epsilon > 0$, the average regret does not vanish.

For the SE kernel, $\gamma_T = \mathcal{O}\big((\log T)^{d+1}\big)$; for Matérn-3/2, $\gamma_T = \mathcal{O}\big(T^{\frac{d(d+1)}{3+d(d+1)}}\log T\big)$ (Srinivas et al., 2012 [94]).

5.5 Experimental Results

This section studies the effect of three system parameters, the number of agents N, the gossip period t_g, and the expected number of received tuples per round N_a, in both

static and time-varying settings. All results are averaged over 30 Monte Carlo runs, and the reported regrets are augmented-agent regrets, averaged over N. The gossip matrix P is sampled uniformly and then rescaled to match a target N_a, where larger N_a corresponds to higher connectivity. With $N = 1$, the method reduces to GP-UCB in the static case and to TV-GP-UCB in the time-varying case. We use $\beta_t = \log(t)$ and inflate the kernel output scale to widen posterior bands. Additional details and experiments are provided in Appendix 6.2.4.

Section 5.5.1 reports a static synthetic study on the four-dimensional Styblinski–Tang function (6.36) with $T_{\text{iter}} = 200$. Section 5.5.2 reports a controlled time-varying synthetic study with $T_{\text{iter}} = 100$, and includes the effect of mismatches between the true dynamics and the TV-GP surrogate. Section 5.5.3 reports real-data experiments on the sensor temperature dataset (Madden, 2004 [131]) (time-varying), following (Bogunovic et al., 2016b [129]).

5.5.1 Synthetic static reward function

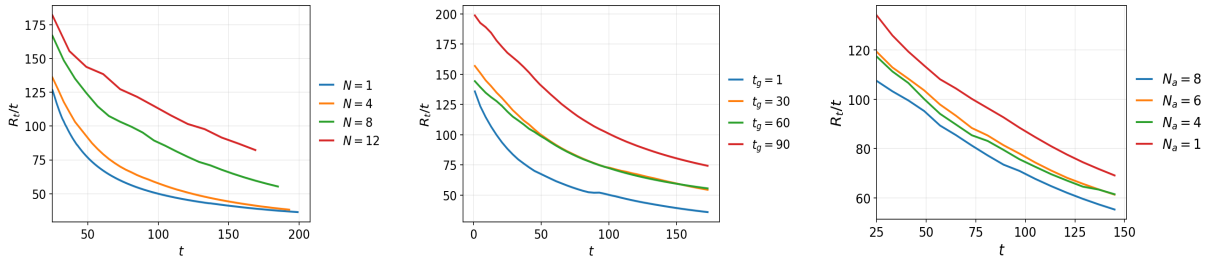
Throughout this section, The hyperparameters of the GPs (the lengthscale and the outputscale) of each agent are learned with every iteration $t \in [1, \dots, T_{iter}]$ using maximum log-likelihood (see Rasmussen et al., 2006 [87]). The iteration budget is fixed to T_{iter}. With N agents in parallel, each agent runs $\lfloor T_{\text{iter}}/N \rfloor$ iterations, so wall-clock time effectively scales by N.

Figure 5.2 reports performance using the per-agent average regret $R^{\text{st}}_{a,T}/T$ (lower is better), averaged across agents. The number of points in the displayed curves are lower that T_{iter} due to the fact that the gossip matrix P is random and that the there is no gossip at $t = 1$.

Figure 5.2a considers $t_g = 1$ (gossip at every iteration) with a fully connected gossip matrix $P = \mathbf{1}^{|\mathcal{A}|\times|\mathcal{A}|}$. Because work is spread across agents, the speedup is close to linear: with N agents we nearly match an N-times faster single-agent run. A small gap appears as N increases, which we attribute to the Kriging–Believer step shrinking variance around received arms and slightly tempering exploration.

Figure 5.2b fixes $N = 4$ and a randomly generated P with $N_a = N$, and varies the gossip period t_g. More frequent communication (smaller t_g) consistently reduces the average regret.

Figure 5.2c varies the effective connectivity of the gossip network through N_a, the expected number of received tuples per round. Larger N_a improves coordination among agents and lowers regret.



(a) Impact of the number of agents N.

(b) Impact of the gossip period t_g.

(c) Impact of N_a.

Figure 5.2 – Impact of system parameters on the average regret over the Styblinski-Tang function. Each panel isolates a factor: (a) number of agents, (b) gossip period, and (c) average number of received tuples N_a.

In summary, for static environments, we recommend keeping the gossip period t_g small whenever budget permits. Connectivity N_a is dependent on the environment and is independent of user choices. Further results on other synthetic functions are presented in Appendix 6.2.4.

5.5.2 Synthetic Time-Varying reward function

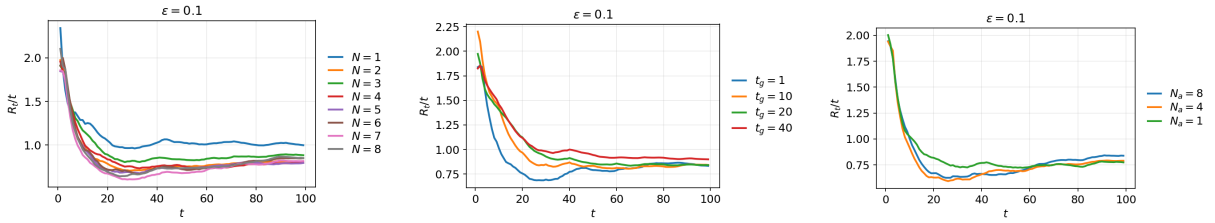
We consider a time-varying reward that follows the Markov model in Assumption 5. We generate a two-dimensional function with lengthscale 0.2 and unit output scale; the drift parameter is fixed to $\epsilon = 0.1$. For simplicity, the agents are assumed to know ϵ, the lengthscale, and the output scale.

Similarly to Section 5.5.1, we first study the effect of the number of agents N. In Figure 5.4a, agents communicate every iteration ($t_g = 1$) over a fully connected network $P = \mathbf{1}^{|\mathcal{A}|\times|\mathcal{A}|}$. Increasing N substantially improves tracking compared to the single-agent case, reflecting the benefit of parallel exploration under frequent gossip.

Next, Figure 5.4b shows the sensitivity to the gossip period t_g with $N = 8$ and a fully connected network $P = \mathbf{1}^{|\mathcal{A}|\times|\mathcal{A}|}$. More frequent gossip (smaller t_g) consistently improves tracking, with a pronounced gain relative to the single-agent baseline.

Finally, Figure 5.3c examines the effect of network connectivity for $N = 8$ by varying the expected number of received tuples per round N_a. The curves are close, indicating a limited sensitivity to N_a in this setup; in other words, performance is relatively robust to the particular connectivity pattern encoded by P.

Another important property of the proposed technique is its resilience to mismatches



(a) Impact of the number of agents N.

(b) Impact of the gossip period t_g.

(c) Impact of N_a.

Figure 5.3 – Time-varying setting: impact of system parameters on the average cumulative regret. Each panel isolates a factor: (a) number of agents, (b) gossip period, and (c) average number of received tuples per round.

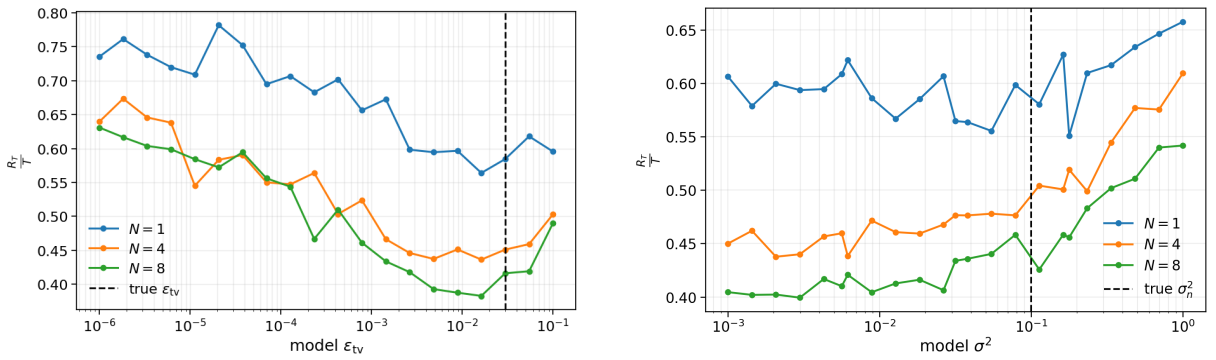
between the drifting reward and the surrogate model used by each agent or the observation noise variance of the reward σ_f^2. In practice, each agent can tune its own hyperparameters by maximizing the marginal likelihood of its GP surrogate, as discussed in the supplementary material of (Bogunovic et al., 2016b [129]), provided that enough observations are available. Throughout this section, we assumed a match between the reward function and the agents' surrogate models.

To test robustness to mismatch, Figure 5.4 reports an example involving two hyperparameters, the observation noise σ_f^2 in (5.7) and the drift parameter ϵ_{TV}. Panel (a) shows the final average augmented regret obtained when the true reward drifts at rate $\epsilon_{\mathrm{TV}} = 0.03$ while the agent surrogate models use different values of ϵ_{TV}. We repeat the experiment for different numbers of agents N over 30 Monte Carlo runs. Across all cases, the final average augmented regret degrades smoothly and remains robust under moderate mismatch, in line with (Bogunovic et al., 2016b [129]). For simplicity, the impact of mismatch in other modeling choices, such as the kernel type and GP kernel hyperparameters, is left for future work.

5.5.3 Real-data reward function

In this section, the proposed algorithm is tested in both a static scenario and a time-varying scenario. We highlight the impact of gossip when the expected number of received tuples per agent is $N_a = 0.8$, for three values of the gossip period $t_g \in \{1, 10, 100\}$.

We consider a time-varying reward function based on real sensor measurements. The goal is to track, over time, the sensor index with the highest temperature among 46 sensors. This defines a one-dimensional reward function whose maximizer changes with time. Each agent uses a time-varying GP surrogate with drift parameter $\epsilon_{\mathrm{TV}} = 0.006$, which we found to work well empirically. The remaining surrogate hyperparameters, using



(a) Drift parameter mismatch.

(b) Observation noise mismatch.

Figure 5.4 – Final average augmented regret for different numbers of agents, namely $N = 1$, $N = 4$, and $N = 8$, under mismatch in the model drift parameter and the observation noise.

the squared exponential kernel k_{SE} in (3), are learned by maximum likelihood on the first 500 datapoints. The algorithm is then run on the next 1500 datapoints (iterations) with these learned hyperparameters. Figure 5.5 reports the average cumulative regret over time, averaged across agents.As in the static case, the cumulative regret decreases as cooperation increases, with a clear gain relative to the single-agent baseline.

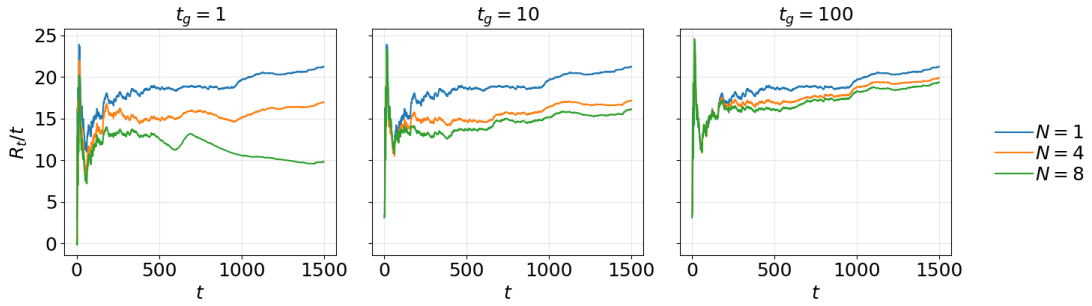


Figure 5.5 – X-KB-UCB performance on the sensor temperature dataset for three networks of agents $N = 1$, $N = 4$, and $N = 8$. Each panel corresponds to a different gossip period t_g, with $N_a = 0.8$.

5.6 Conclusion

Within this chapter, we introduced *X-KB-UCB*, a decentralized UCB policy that assimilates gossiped observations through a cross-agent Kriging–Believer step to coordinate exploration while keeping decision making local. To analyze the effect of gossip, we rely

on an augmented-agent abstraction that makes explicit the information available to an agent after receiving tuples from its neighbors, and it enables a comparison between regimes ranging from isolated learning (no gossip) to near-centralized information sharing (frequent gossip). We prove high-probability no-regret guarantees in the static case and extend them to time-varying rewards under a Markov-drift model. Experiments support the analysis, showing that gossip improves sample efficiency on static objectives and improves tracking performance when the maximizer drifts.

Several avenues merit further work. First, it would be useful to study mismatches between the true reward kernel and the kernel assumed by the agents' surrogates. Second, throughout this work we assumed homogeneous agents using GP surrogates with the same kernel. Allowing heterogeneity, for example agents using different kernels or operating under different drift rates, is a natural extension. Third, the cross-agent KB coordination step could be tested in settings with non-identical reward functions across agents and under communication constraints induced by privacy. Finally, we considered synchronous communication, and extending the analysis to asynchronous gossip with delays or dropped messages is an important direction.

The main contribution of this chapter is the introduction of X-KB-UCB, a decentralized policy for coordinating exploration under strict communication budgets. Two contributions support this development. First, we propose a cross-agent KrB coordination step at gossip rounds, which lets each agent incorporate neighbors' recent actions into its uncertainty update while keeping decision making decentralized. Second, we introduce an agent-centric augmented-agent abstraction that makes explicit what information an agent has after gossip and enables a direct regret analysis of the effect of communication. Existing GP bandit results provide principled guarantees for single-agent methods in static settings (Srinivas et al., 2012 [94]) and under time variation (Bogunovic et al., 2016b [129]), as well as centralized batch-selection methods (Desautels et al., 2014 [130]; Parker-Holder et al., 2020 [128]) that can be interpreted as a single decision maker selecting multiple points per round. In contrast, the setting studied here involves multiple agents that must coordinate in a decentralized way to optimize a shared noisy objective under limited, periodic communication, which we treat as a distinct gap in the literature.

CONCLUSION

5.7 General conclusion and contributions

This section presents the general conclusion and the main contributions of this work.

In this dissertation, we introduced receiver hyperparameter tuning as a research problem that has been overlooked in the telecommunications literature. In practice, this task is often approached through brute-force trials, experience-based guessing, manual tuning, or heuristics. However, systematic automation of this tuning process has remained limited in the literature.

— In Chapter 1, we provided an overview of the telecommunication system of interest in this study, namely Underwater Acoustic Communications, and highlighted its main distinctive attributes compared to other telecommunication systems, in particular severe Doppler effects, strong multipath propagation, and their rates of change. We also presented the transmission setup considered in the dissertation and the receiver architecture based on a Decision Feedback Equalizer.

  Building on this setting, we defined and formalized receiver hyperparameter tuning as a black-box optimization problem, which constitutes a contribution of this work. We defined a black-box function that takes hyperparameter values as input and outputs receiver performance. This performance metric was built on the output SNR and included a hallucination penalty. Hallucinations were detected by analyzing the behavior of the DFE filters, which constitutes an original contribution of this work. Maximizing this receiver performance function provided access to hyperparameters that were well matched to specific channel conditions.

— In Chapter 2, we provided an overview of techniques for tuning the black-box objective functions developed in the previous chapter. Among existing approaches, the Bayesian optimization framework, as an iterative procedure designed for expensive evaluations, was identified as a good fit for this setting. We detailed its main components, including acquisition policies that proposed relevant hyperparameters and the Gaussian-process surrogate model used to guide decisions. We also presented several policies with different exploration–exploitation trade-offs that were subsequently used by our solutions.

— Chapter 3 introduced a first solution for receiver hyperparameter tuning based on a direct application of Bayesian optimization. The receiver was run multiple times with hyperparameter values proposed by BO, and the hyperparameter vector achieving the highest black-box value was selected at the end of the budget. This solution, named the *Search method*, used a single BO agent, that is an entity with its own policy and surrogate. The Search method was best adapted to static conditions, for instance when the goal was to find the best hyperparameter match to recorded channels, replay data, or simulations. It highlighted the potential of automating a task that is known by practitioners to be unpleasant and time-consuming, since manual tuning can require repeated trial-and-error and may take days. The Search method constitutes an original contribution of this work.

— While the Search method was adapted to static conditions, UAC channels were time-varying, and hyperparameters that fit a given received packet could become unsuitable for subsequent packets. This motivated a different problem, formalized in Chapter 4, where the objective was to track good hyperparameters online as packets arrived. This solution was called the *Tracking method*.

The Tracking method relied on a new BO operating mode, referred to as the flywheel mode, which performed one BO iteration per received packet, thereby optimizing receiver performance along the communication link under a packet-rate constraint. We developed the Tracking method in two phases. First, we studied a simplified version that tested the flywheel mode with a single agent using a UCB policy and a static GP surrogate. This version showed promise but also revealed limitations due to the use of a single agent and a static surrogate. In particular, a single agent struggled to balance exploration and reliability under strong variability, and a static surrogate assigned the same weight to old and recent observations even though the channel, and thus the objective, changed over time. In addition, hallucination-prone regions could evolve, which further motivated time-aware treatment.

The second version leveraged two agents running in parallel, a Wise agent that made conservative choices based on gathered observations up to the current packet, and a Scout agent that explored the hyperparameter space. This Tracking method was called *WandS*. Each agent used a policy aligned with its role. These policies introduced a discounted hallucination penalty, meaning that hallucination regions were penalized strongly when they were recent and the penalty faded gradually as communication evolved. *WandS* also relied on a common time-varying surrogate that discounted old observations, enabling it to cope with channel variations. Both the Tracking method formulation and *WandS* constitute original contributions of this work.

— Finally, Chapter 5 addressed a main limitation of *WandS*, namely the need for a central node to synchronize the Wise and the Scout, select among their proposals, and refit the common surrogate. This could be costly and technically challenging. We therefore extended the multi-agent structure to a decentralized setting and moved to a more general framework than UAC, namely GP bandits. We introduced an original UCB-based policy that leveraged a coordination mechanism via Kriging–Believer, called *X-KB-UCB*. This policy allowed agents to ingest data received from neighboring agents through a gossip mechanism in a way that improved performance. In this chapter, the method was studied in the GP-bandit setup and we provided theoretical guarantees showing that receiving data from other agents could be beneficial compared to isolated learning, and could approach centralized performance when gossip was frequent. The decentralized GP-bandit setting considered here, as well as the *X-KB-UCB* policy and its analysis, constitute original contributions of this work.

5.8 Future perspectives

Several extensions of the proposed methods can be considered. We group them into two categories: future improvements of the proposed algorithms and extensions to applications beyond underwater acoustic receivers.

Regarding algorithmic improvements:

— The black-box objective used in this dissertation is a proxy for BER based on the output SNR, together with a hallucination penalty. Some hyperparameters, especially the DFE filter lengths, also directly affect the computational complexity. Introducing a complexity penalty, for instance penalizing long filters, would formalize the trade-off between performance and runtime.

— The proposed methods tuned up to about five hyperparameters simultaneously and could potentially scale to around twenty, which is a common rule of thumb for the surrogates and policies used in this work (Garnett, 2023a [12]). Extending these methods to higher-dimensional settings is therefore a natural direction, for example through dimensionality-reduction techniques (Wang et al., 2016 [132]), or function decomposition (Bardou et al., 2024b [133]).

— Increasing the number of agents in *WandS* could further improve performance and stability. In addition, the current agents rely on UCB and LCB policies, which themselves introduce tuning parameters. Exploring hyperparameter-free alternatives, such as expected improvement, could reduce the number of design parameters in *WandS*, even though in our experiments the method was not strongly sensitive

to them.

WandS relies on a TV-GP surrogate to handle time variation by discounting older observations. However, even when their influence becomes negligible, these observations remain stored in $\mathcal{D}$, thereby increasing the computational complexity of the procedure. Alternative strategies could therefore be considered, including periodic GP restarts (Zhao et al., 2020 [114]) and sliding-window updates that discard older observations (Zhou et al., 2021 [115]). More recent work has also proposed explicit deletion criteria based on observation relevance. In particular, (Bardou et al., 2024a [116]) introduced a Wasserstein-distance-based criterion to quantify how much an observation contributes to future predictions, enabling bounded dataset sizes while preserving predictive quality and increasing the sampling frequency in long-horizon dynamic optimization.

Finally, although *WandS* was tested on both synthetic scenarios and real recordings, a dedicated sea trial would provide a stronger validation of the method.

— In Chapter 5, we assumed homogeneous agents using the same surrogate and the same policy. Allowing heterogeneity across agents is a natural extension. We also assumed that all agents optimize the same objective function, and relaxing this assumption would broaden the applicability of the framework. Finally, we assumed perfect synchrony at gossip rounds. Studying asynchronous gossip, including delayed or dropped messages, and its impact on the regret guarantees constitutes a complementary direction.

Regarding algorithmic extensions:

— Adaptive Modulation and Coding (AMC) techniques (Goldsmith et al., 1998 [134]) have been investigated in recent works, including in UAC (Cooper-Baldock et al., 2025 [135]). These techniques aim to close the loop between the receiver and the transmitter, by feeding back information that helps the transmitter select an appropriate Modulation and Coding Scheme (MCS). This is typically done using BER, PER, or effective SNR, the latter being computed from the preamble and the training sequence (Zhang et al., 2022b [136]), since plain SNR can become unreliable in corrupted regimes that give rise to hallucinations. The performance metric introduced in Chapter 1 could help reduce the overhead compared with these alternatives, since it overcomes hallucinations.

In addition, AMC feedback often relies on measured channel information, and therefore depends on the quality of these measurements. Leveraging the *Tracking* method could alleviate this dependence on channel estimation. Another limitation of AMC is the outdatedness of the feedback, since the suggested MCS may become stale as the channel evolves during the feedback delay, which can be significant

in the subsea environment because of the low sound speed. This issue could also be addressed by *WandS*, by using a TV-GP surrogate on the transmitter side to predict the performance of the candidate MCSs.

More broadly, the *Tracking* method could be extended to jointly tune the transmitter and the receiver, by selecting both an appropriate MCS and a matching receiver configuration for the current channel conditions. In that case, other objectives such as throughput could be introduced and jointly optimized with other performance metrics, such as BER, or the objective function in Chapter 1. This naturally connects with multi-objective Bayesian optimization (Daulton et al., 2022 [137]; Paria et al., 2020 [138]; Suzuki et al., 2020 [139]), whose acquisition functions could be integrated into the algorithms proposed in this work.

— The *Tracking* method enables the tracking of a drifting optimal region in a search space. This problem also arises when tuning receivers in other communication domains, especially when the channel is time-varying. For instance, in vehicular communications, Kalman filters are used for channel estimation and involve sensitive hyperparameters that are often tuned through grid search or manual adjustment (El Husseini et al., 2019b [140]). Likewise, the *Search* method can extend beyond underwater acoustic communications. For example, in the Internet of Things, connected devices must perform time synchronization, often through a maximum-likelihood detector whose hyperparameters, such as step sizes, require tuning for accurate synchronization (Nasr et al., 2016 [141]).

More generally, tracking problems arise far beyond telecommunications. In environmental monitoring, the most informative observation locations can drift over time with weather and environmental conditions (Imamura et al., 2020 [142]). In recommendation systems, user preferences evolve with time and trends (Hariri et al., 2015 [143]). Therefore, these domains constitute promising application areas for the *Tracking* method.

Chapter 6

Appendix

6.1 Search method for Mud-Pulse Telemetry

6.1.1 Measurement-While-Drilling

In this section, the Search method for tuning receiver hyperparameters, introduced in Chapter 3, is applied to a different domain, namely measurement-while-drilling (MWD). In the oil and gas industry, the exploration stage, during which drilling is performed to assess underground resources, is critical and is often considered the riskiest phase of operations (Jarrot et al., 2018 [144]).

Figure 6.1 shows a schematic representation of the drilling system. A drill bit penetrates the formation and is connected to the surface through a drill string composed of drill pipes. Downhole tools located near the drill bit measure quantities such as temperature, inclination angle, radioactivity, and other indicators that help guide drilling toward productive zones. These measurements must be conveyed to the surface to support operational decision making and are transmitted through telemetry systems that rely on different physical carriers. Telemetry systems can be broadly classified as wired or wireless. Wired telemetry while drilling relies on wired drill pipe technology to transmit downhole measurements to the surface. Signal transmission is achieved through electromagnetic induction, where electrical currents in coils generate time-varying magnetic fields that carry information along the drill string. A main advantage of this approach is its high data rate, commonly around 100 kb/s, with some experimental studies reporting rates up to 2 Mb/s over 2 km wells (Han et al., 2021 [145]). Despite this performance advantage, the technology has practical limitations. It typically requires homogeneous pipe sections, which may be difficult to guarantee in field operations. In addition, specialized connectors between pipe joints increase operational complexity and cost. These constraints remain major barriers to widespread adoption.

Wireless telemetry systems are generally less costly to deploy and maintain and can be operationally robust when wiring is impractical. Wireless approaches include acoustic

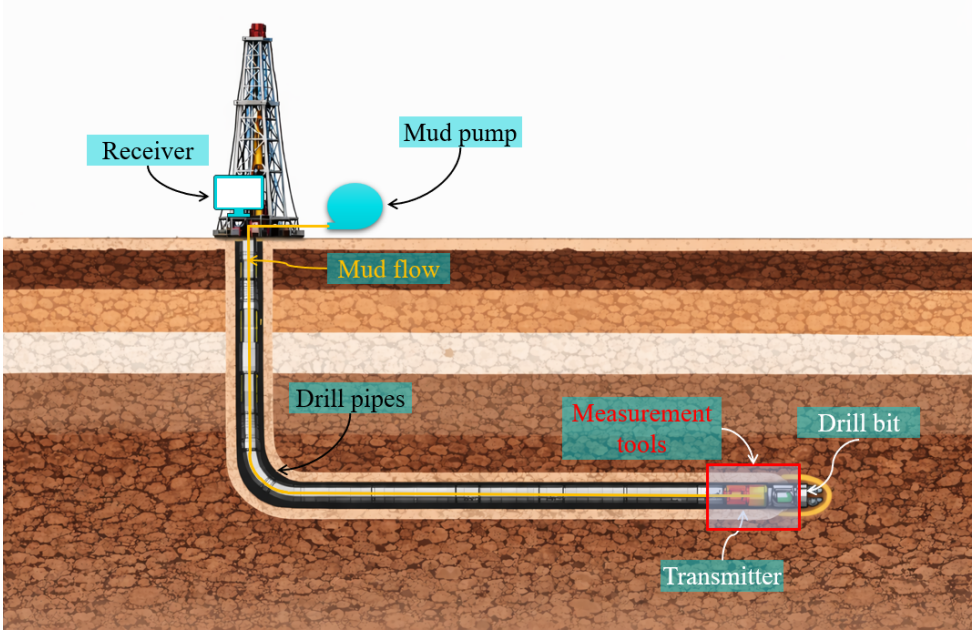


Figure 6.1 – Graph of Mud-Pulse Telemetry where a Transmitter downhole relay measurements to a surface receiver.

telemetry (AT) (Li et al., 2023b [146]), electromagnetic telemetry (EM) (Popovici et al., 2017 [147]), and mud-pulse telemetry (MPT) (Li et al., 2023b [146]). These technologies differ in their physical carriers. AT conveys information through acoustic waves propagating in the formation and drill string. EM relies on electromagnetic fields transmitted through the surrounding formation. MPT transmits information through pressure waves propagating in the drilling mud inside the drill pipes.

AT can, in principle, achieve data rates on the order of 400 bps in favorable conditions, but it is often limited in practice by strong interference and severe attenuation (Pelekanakis et al., 2014 [148]). EM is primarily used for onshore and relatively shallow operations (Jarrot et al., 2018 [144]), and its range and reliability are strongly constrained by formation resistivity, which can induce significant attenuation. MPT, while typically limited to data rates on the order of 20 bps at best, is widely adopted for deep and high-value drilling operations due to its reliability and comparatively low operational cost (Jarrot et al., 2018 [144]). MPT is also regarded as the most mature MWD telemetry technology, with early deployments dating back to the early 1970s (Spinnler et al., 1978 [149]). For these reasons, the scope of this study is restricted to MPT.

The objective of this appendix section is to illustrate how the proposed approach can automate receiver-tuning processes for MWD systems. For this reason, and for clarity, transmitter details are omitted, except that pressure modulation is produced through a rotor-stator mechanism.

6.1.2 MPT objective function

Within this work, SLB's receiver for Mud-Pulse Telemetry (MPT) is used to highlight the benefits of the Search technique. For confidentiality reasons, the full receiver architecture is not disclosed. Only the elements required to define the optimization problem and interpret the results are described in this section.

The propagation of pressure waves through circulating drilling mud is affected by several impairments, including variations in wave velocity, multipath propagation, and severe noise.The pressure-wave velocity depends mainly on mud density and compressibility, and it is also influenced by gas contamination, temperature, and pressure. Since these quantities vary with mud formulation and downhole conditions, the wave velocity can change across wells and with depth. As a result, the propagation speed can vary over a broad range, approximately 1000 to 1500 m/s (Chin et al., 2024 [150]). This variability, together with clock and phase offsets induced by the oscillators, motivates the use of phase-locked-loop (PLL) tracking. The channel itself, namely the mud column inside the drill string, is dispersive due to reflections at pipe joints, surface equipment, and flow variations. It is also time-varying because mud properties and the number of pipe joints evolve with depth (Spinnler et al., 1978 [149]). In addition, strong noise is present, in particular from surface equipment. For example, pump pistons can create narrowband harmonics that appear as cyclic pressure imbalances (Jarrot et al., 2018 [144]).

These impairments motivate sophisticated receiver processing and introduce a large number of signal-processing hyperparameters. In this study, the scope is restricted to decision-feedback equalizer (DFE) hyperparameters (Jarrot et al., 2018 [144]) together with PLL tracking-loop hyperparameters. The PLL hyperparameters considered here are bandwidth parameters and damping factors.

This study is conducted on recordings from previous operations. The ground-truth transmitted bits are not available, which requires using the objective function introduced in Chapter 1, Section 1.4. Figure 6.2 shows two example objective landscapes obtained from two operations, both reaching depths around 1400 ft but recorded at different locations. For each operation, the same surface recording is replayed multiple times using hyperparameters proposed either by optimizer-agent runs or by random sampling. Since a single replay can take up to one hour, exhaustive grid sampling, as used earlier in the manuscript for cheaper experiments, is computationally prohibitive in this setting. The landscapes shown in Figure 6.2 are therefore assembled from Monte Carlo runs, where each run corresponds to one optimization trajectory. These Monte Carlo simulations are the same ones used to produce the convergence plots in Figures 6.3 and 6.4.

Figure 6.2 highlights two important observations. First, operation (a) does not exhibit a clear high-performance region. In contrast, operation (b) shows a more identifiable region of high SNR values (blue points), which is the region targeted by Search method. Second,

hallucination-prone configurations (gray points) are prevalent, and their prevalence is significantly higher than in the UAC use cases. In operation (a), hallucinations are spread across the search space, which makes tuning substantially more difficult than in operation (b). A plausible explanation is that operation (a) contains many drilling stops. Each stop can alter the effective channel, and as discussed in Section 1.4, abrupt changes in the propagation medium can trigger hallucination. operation (a) also exhibits stronger harmonic noise, which further increases the likelihood of decision errors and hallucination. operation (b), on the other hand, displays a clearer region of interest at higher PLL bandwidth and higher damping-factor values, with fewer hallucinations in that region. The concentration of evaluated points around this region reflects where Search method spends most of its evaluations. Overall, both operations yield high SNR for non-hallucinating configurations, but the associated objective functions are highly non-trivial.

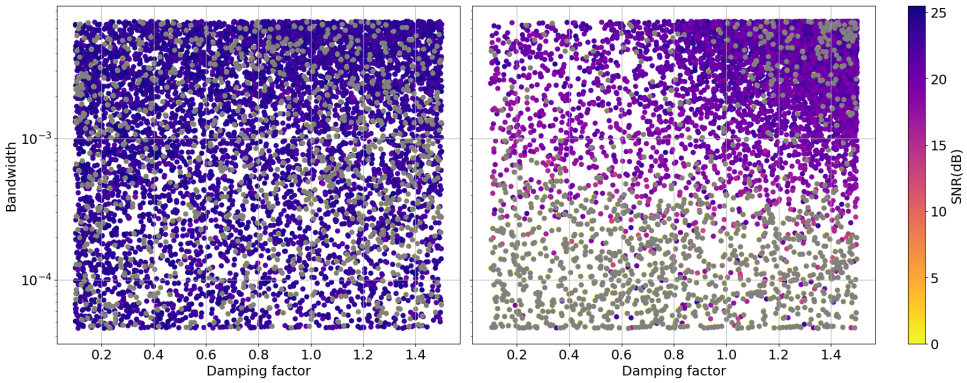


Figure 6.2 – Receiver performance as a function of two PLL hyperparameters (bandwidth and damping factor). Left: drilling operation (a) at 1350 ft. Right: drilling operation (b) at a different location at 1400 ft.

6.1.3 Numerical results

Two scenarios are considered for MPT using the Search method. The first scenario is within-operation fine-tuning. In this scenario, the Search method is run on recordings from recent hours or days, and the selected hyperparameters are then used for subsequent received measurements. To emulate this setting offline, each operation recording is split into consecutive slices.

The second scenario is post-operation recovery. In this scenario, full replays are performed and the goal is to identify hyperparameters that best match a given location and target depth under fixed operational conditions. In that case, one full replay of the operation recording is performed per Search method iteration.

For within-operation fine-tuning, the Search method is applied sequentially across slices using a TPE-based agent. For each new slice, a warm-up phase is performed by evaluating the expert default configuration, then re-evaluating on the current slice the 10 best hyperparameter configurations obtained on the previous slice, followed by 50 purely random evaluations. Then, 100 TPE-driven BO iterations are performed on that slice. This emulates periodic retuning as conditions evolve while reusing configurations discovered on earlier slices.

The objective functions in Figure 6.2 contain a higher fraction of hallucinations than in the UAC experiments, which requires additional care. A natural idea would be to increase the hallucination penalty beyond -50 dB. However, as highlighted in Chapter 4, Section 4.5, Figure 4.11, a GP surrogate can create spurious high predictions near hallucination regions due to interpolation, which can mislead the optimizer.

To better steer the search away from hallucinations, a hallucination-aware objective is introduced

$$f^{(h)}(\boldsymbol{\theta}) = f(\boldsymbol{\theta}) \cdot \log\left(\|\boldsymbol{\theta}_h - \boldsymbol{\theta}\|^2\right), \tag{6.1}$$

where $\boldsymbol{\Theta}_h$ denotes the set of hallucination configurations observed so far, and

$$\boldsymbol{\theta}_h = \arg\min_{\boldsymbol{\theta}' \in \boldsymbol{\Theta}_h} \|\boldsymbol{\theta}' - \boldsymbol{\theta}\|^2 \tag{6.2}$$

is the nearest previously observed hallucination to $\boldsymbol{\theta}$. This objective penalizes candidates that lie close to known hallucination.

Figure 6.3 reports the fraction of hallucination selections per iteration, aggregated across slices, for the two objectives shown in Figure 6.2. The percentage of hallucinations is higher in early iterations (gray region) because these correspond to the warm-up phase, during which hallucinations are not explicitly avoided. For the operation (b) objective, the hallucination-aware objective (orange curve) reduces the hallucination fraction by roughly a factor of two. For operation (a), the improvement is more modest, which is consistent with the fact that hallucinations are widely spread across the search space.

Figure 6.4 shows the evolution of the best performance achieved, averaged across slices, as a function of the iteration index. For each slice, the first evaluated configuration corresponds to expert values selected by domain experts. The next 10 configurations correspond to the best configurations obtained on the previous slice and re-evaluated on the current slice. The remaining warm-up evaluations are random samples. After warm-up, the optimizer agent runs for 100 TPE-driven iterations. Due to the high computational cost of each iteration, only expert values are used as a benchmark for within-operation fine-tuning.

For operation (a), Search method does not lead to a significant improvement beyond expert values, especially after the warm-up phase, which is consistent with the high

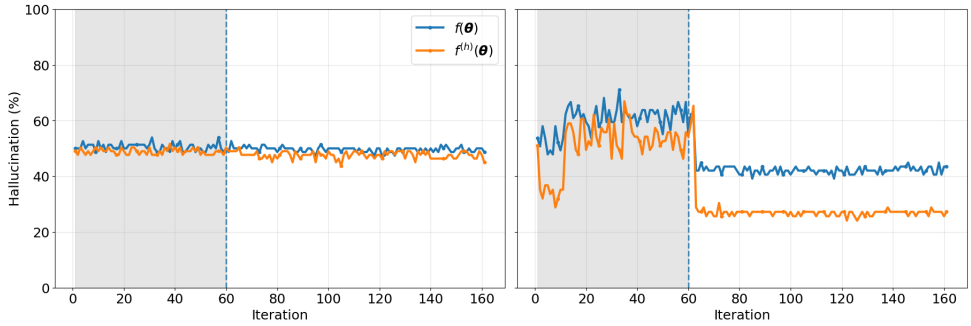


Figure 6.3 – Fraction of hallucination selections per iteration, aggregated across slices, for the two objectives shown in Figure 6.2. Orange: optimization using $f^{(h)}$ in (6.1). Blue: optimization using f.

prevalence of hallucinations. For operation (b), Search method yields a clearer gain. A performance improvement of approximately 4 dB over expert values is observed, and convergence occurs rapidly.

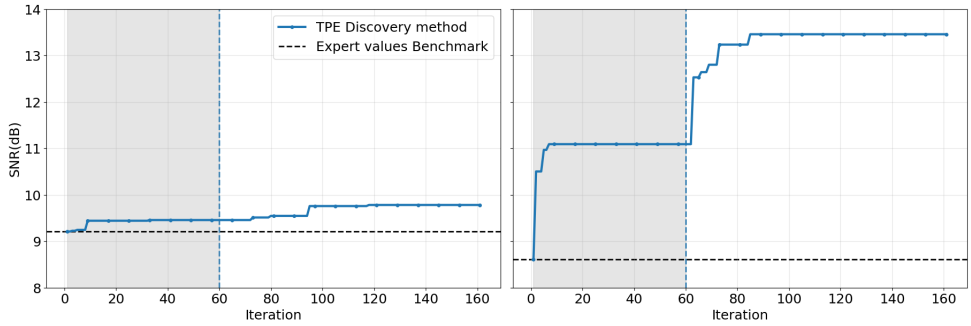


Figure 6.4 – Average best receiver performance versus iteration for the two operations shown in Figure 6.2. Left: operation (a). Right: operation (b).

In post-operation recovery, additional performance proxy indicators are available even without knowledge of the transmitted bits. In particular, it is possible to count how often the receiver remains in synchronization, how many surveys (i.e., individual telemetry packets carrying measurement data) pass CRC checks (Good Surveys), and how many surveys are recovered through error-correction decoding (Corrected Surveys). These indicators are not reliable on short slices because too few synchronization words are observed given the extremely low bitrate.

For the post-operation recovery scenario, optimization is therefore performed using the following objective function

$$\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \quad f^{(\text{replay})}(\boldsymbol{\theta}) = N_{\text{sync}} + 5N_{\text{CS}} + 10N_{\text{GS}}, \tag{6.3}$$

where N_{sync} is the number of frames synchronized (sync), N_{CS} is the number of Corrected Surveys (CS), and N_{GS} is the number of Good Surveys (GS). The weights in this objective

reflect the relative importance of each component, as assessed by domain experts during manual tuning.

Examples of this objective are shown in Figure 6.5 for the same replay data used in Figure 6.2. This objective is discrete and can be volatile. Due to the high cost of post-operation recovery optimization, the scope is restricted to a single one DFE forgetting factor. For the remaining hyperparameters, we observed with extensive random search that expert values were close to optimal.

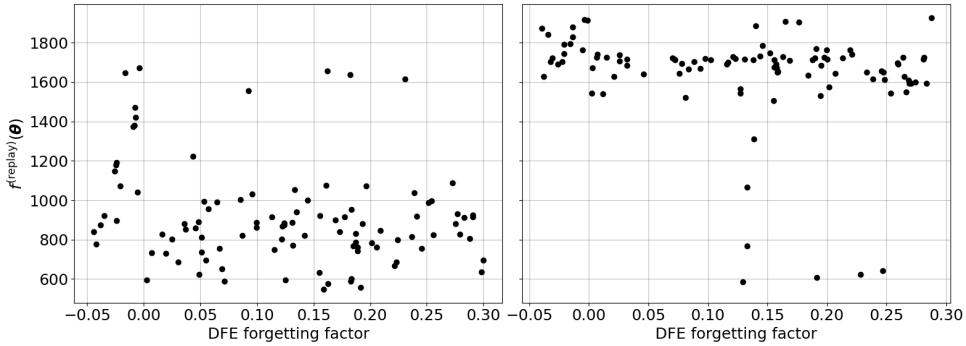


Figure 6.5 – Post-operation recovery objective as a function of a DFE forgetting factor. Left: operation (a). Right: operation (b).

To assess Search method performance in post-operation recovery, a UCB-based agent is used because initial experimentation indicated better results than TPE in this setting. Each run uses 50 random warm-up evaluations followed by 50 UCB-driven iterations. Results over 9 recordings are summarized in Table 6.1.

Across all recordings, Search method matches or improves upon Expert value and never yields a lower score. The largest gain is obtained on Recording 7, where the score increases from 2791 to 2931, corresponding to a 5.0% improvement. Additional notable improvements are observed on Recordings 5 and 9, with score gains of 3.15% and 3.25%, respectively. These results indicate that Search consistently preserves baseline performance while providing measurable improvements on the most favorable recordings.

6.1.4 Conclusion

This appendix applies the Search method to an MPT receiver, which lies outside the UAC setting considered in the main chapters. The experiments indicate that the Search method can recover expert-level performance through a systematic, data-driven procedure that requires no prior knowledge of the channel, and it can yield measurable improvements on some recordings. In within-operation fine-tuning, gains are scenario-dependent. They are limited for operation (a), which is dominated by hallucinations, and more significant

Recording	**Method**	**Forgetting factor**	N_{GS}	N_{CS}	N_{sync}	$f^{(\text{replay})}$
Recording 1	Search method	0.784595632	89	1	264	1159
	Expert value	0.75	88	1	264	1149
Recording 2	Search method	0.915410707	192	0	452	2372
	Expert value	0.75	190	0	451	2351
Recording 3	Search method	0.925159972	204	1	497	2542
	Expert value	0.75	197	1	495	2470
Recording 4	Search method	0.917675682	90	0	200	1100
	Expert value	0.75	89	0	199	1089
Recording 5	Search method	0.865183656	172	1	469	2194
	Expert value	0.75	166	0	467	2127
Recording 6	Search method	0.75	150	2	429	1939
	Expert value	0.75	150	2	429	1939
Recording 7	Search method	0.860275639	223	7	656	2931
	Expert value	0.75	213	2	651	2791
Recording 8	Search method	0.694807409	136	0	468	1828
	Expert value	0.75	135	0	467	1817
Recording 9	Search method	0.976654358	111	1	601	1716
	Expert value	0.75	106	1	597	1662

Table 6.1 – Optimized Mud-Pulse Telemetry receiver versus expert values. Green blocks indicate the best performance for each recording. N_{sync} denotes is the number of frames synchronized, N_{CS} is the number of Corrected Surveys, and N_{GS} is the number of Good Surveys.

for operation (b), where a clearer region of high performance exists. In post-operation recovery, the Search method improves upon or matches the expert value across all tested recordings.

The main point is not that BO consistently and dramatically outperforms expert tuning in this limited offline study. Rather, it can reach comparable, and sometimes better, performance with substantially less manual effort. Expert tuning can require days of trial-and-error and domain expertise, whereas the Search method provides a repeatable procedure for exploring the hyperparameter space automatically. These results motivate larger-scale evaluation.

6.2 Chapter 5

We work under the standing assumptions in the main text: $\boldsymbol{\Theta} \subset [0, r]^d$ compact and convex; $k(\boldsymbol{\theta}, \boldsymbol{\theta}) \leq 1$; i.i.d. Gaussian noise with variance σ_n^2 within noisy observations. All agents are *peers*: they run the same X-KB-UCB schedules and broadcast their previous UCB-selected *arms* at gossip.

6.2.1 Proof of Lemma 1

Fix a gossip round t and agent a, and write $S := S_{a,t}$, $\boldsymbol{f}_S := [f_{t-1}(\boldsymbol{\theta})]_{(\boldsymbol{\theta},t-1)\in S}$, and $y_S = \boldsymbol{f}_S + \varsigma_S$ with $\varsigma_S \sim \mathcal{N}(0, \sigma_n^2 I)$ independent of everything. We have:

$$I\Big(f_t(\boldsymbol{\theta}); y_S \mid \boldsymbol{y}_{1:t-1}\Big) \leq I\Big(f_t(\boldsymbol{\theta}), \boldsymbol{f}_S; y_S \mid \boldsymbol{y}_{1:t-1}\Big) = I\Big(\boldsymbol{f}_S; y_S \mid \boldsymbol{y}_{1:t-1}\Big). \tag{6.4}$$

because, augmenting the first argument cannot decrease MI. Moreover $y_S \perp (f(\boldsymbol{\theta}, t), \boldsymbol{y}_{1:t-1}) \mid \boldsymbol{f}_S$ because $\boldsymbol{y}_S = \boldsymbol{f}_S + \varsigma_S$. Using $I(A; B \mid C) = H(B \mid C) - H(B \mid A, C)$ and $H(B \mid C) \leq H(B)$,

$$I\Big(\boldsymbol{f}_S; y_S \mid \boldsymbol{y}_{1:t-1}\Big) \leq I\Big(\boldsymbol{f}_S; y_S\Big). \tag{6.5}$$

$\forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, k(\boldsymbol{\theta}, \boldsymbol{\theta}) \leq 1$ and $k^{dyn}(\boldsymbol{\theta}, \boldsymbol{\theta}) \leq 1$. By (5.30) we have : $I\Big(\boldsymbol{f}_S; y_S\Big) = \frac{1}{2} \log\det\Big(I + \sigma_n^{-2} K_S\Big)$. Use $\log\det(I + M) \leq \mathrm{tr}(M)$ for $M \succeq 0$ and $\mathrm{tr}(K_S) \leq |S|$ and $|S| \leq |\mathcal{A}| - 1$:

$$\frac{1}{2} \log\det\Big(I + \sigma_n^{-2} K_S\Big) \leq \frac{1}{2} \sigma_n^{-2} \mathrm{tr}(K_S) \leq \frac{|S|}{2\sigma_n^2} \leq \frac{|\mathcal{A}| - 1}{2\sigma_n^2}. \tag{6.6}$$

6.2.2 Theorem 1 (Stationary reward)

Consider f under assumptions 4 and 6. We first recall the high-probability confidence sets for GP-UCB (Srinivas et al., 2012 [94]) and the batch inflation argument of GP-BUCB (Desautels et al., 2014 [130]). Using C_a provided with Lemma 1,

$$I\Big(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1}\Big) = \tfrac{1}{2}\log\frac{\sigma^2_{a,t}(\boldsymbol{\theta})}{\widehat{\sigma}^2_{a,t}(\boldsymbol{\theta})} \leq C_a \tag{6.7}$$

implies the band inclusion

$$\alpha_t^{1/2}\sigma_{a,t}(\boldsymbol{\theta}) \leq (e^{2C_a}\alpha_t)^{1/2}\widehat{\sigma}_{a,t}(\boldsymbol{\theta}) = \beta_t^{1/2}\widehat{\sigma}_{a,t}(\boldsymbol{\theta}), \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}, \tag{6.8}$$

so the classic and batch UCB bands are contained in our X-KB-UCB band at gossip rounds. At non-gossip rounds $\beta_t = \alpha_t$, and the KB variance does not enlarge the band, hence the inclusion. This inclusion enables the inheritance of regret bounds from both the GP-UCB and BUCB as highlighted in (Desautels et al., 2014 [130]).

GP-UCB and GP-BUCB

The single-agent GP-UCB bound the regret $R_T = \sum_{t=1}^{T}(f(x^*) - f(\boldsymbol{\theta}_t))$ as follows:

Theorem 3 ((Srinivas et al., 2012 [94])) *Let $\boldsymbol{\Theta} \subset [0, r]^d$ be compact and convex, $d \in \mathbb{N}$, $r > 0$. Let k be a kernel such that $f \sim \mathcal{GP}(0, k)$ satisfies Assumptions 4 and 6. Fix $\delta \in (0, 1)$, and define $\alpha_t = 2\log\left(\frac{4\pi t^2}{3\delta}\right) + 2d\log\left(t^2 dq_2 r\sqrt{\log\frac{8dq_1}{\delta}}\right)$,then with probability at least $1 - \delta/2$:*

$$R_T \leq \sqrt{C_1 T \alpha_T \gamma_T} + 2 \tag{6.9}$$

where $C_1 = \frac{8}{\log(1+\sigma_n^{-2})}$.

The centralized GP-BUCB, where at each iteration t, we choose a batch of B *arms* using KB and parallelize reward computation, bounds $R_{T_B} = \sum_{t=1}^{T_B}(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta}_t))$, $T_B = B \cdot T$, as follows.

Theorem 4 ((Desautels et al., 2014 [130])) *Let $\boldsymbol{\Theta} \subset [0, r]^d$ be compact and convex, $d \in \mathbb{N}$, $r > 0$. Let k be a kernel such that $f \sim \mathcal{GP}(0, k)$ satisfies Assumptions 4 and 6. Fix $\delta \in (0, 1)$, and $C_a > 0$ with $I(f(\boldsymbol{\theta}); \boldsymbol{y}_{S_{a,t}} \mid \boldsymbol{y}_{1:t-1}) \leq C_a$. Define $\alpha_t = 2\log\left(\frac{4\pi t^2}{3\delta}\right) + 2d\log\left(t^2 dq_2 r\sqrt{\log\frac{8dq_1}{\delta}}\right)$, then with probability at least $1 - \delta/2$:*

$$R_{T_B} \leq \sqrt{C_1 T_B e^{2C_a}\alpha_{T_B}\gamma_{T_B}} + 2. \tag{6.10}$$

Proof of Theorem 1

X-KB-UCB uses the single-agent GP-UCB rule at non-gossip rounds ($t \bmod t_g \neq 0$) and applies KB at gossip rounds ($t \bmod t_g = 0$). Using the regret of an augmented agent (5.26), we decompose the static regret into the sum over locally played tuples at non-gossip times and received tuples at gossip times:

$$Z_{ng} := \{(a, \boldsymbol{\theta}_{a,t}, t) : t \in \{1, \dots, T_{\text{iter}}\},\ t \bmod t_g \neq 0\}, \qquad Z_g := \bigcup_{t \in \{1,\dots,T_{\text{iter}}\},\ t \bmod t_g = 0} S_{a,t}. \tag{6.11}$$

$$R^{st}_{a,T} = \sum_{(i,\boldsymbol{\theta},\tau) \in Z_{ng}} \Big(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta})\Big) + \mathbb{E}\left[\sum_{(i,\boldsymbol{\theta},\tau) \in Z_g} \Big(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta})\Big)\right]. \tag{6.12}$$

The expectation $\mathbb{E}$ is over the number of received gossip tuples. The expected count

$$T := \mathbb{E}[|\mathcal{Z}_a|] = T_{\text{iter}} + N_g N_a, \tag{6.13}$$

and defining

$$T_B = T_{\text{iter}} + N_a T_{\text{iter}} = T + N_a(T_{\text{iter}} - N_g) =: T + \phi_g, \tag{6.14}$$

the band inclusion (6.8) implies that the classic GP-UCB confidence set is contained in our band at non-gossip rounds and the batch-inflated one is contained at gossip rounds. Therefore, under the events of Theorems 3 and 4 (each holding with probability $1 - \delta/2$), and using that cumulative regret is non-decreasing in the horizon,

$$E_1 : \quad \sum_{(i,\boldsymbol{\theta},\tau) \in Z_{ng}} \Big(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta})\Big) \leq R_T \leq \sqrt{C_1 T \alpha_T \gamma_T} + 2, \tag{6.15}$$

$$E_2 : \quad \mathbb{E}\left[\sum_{(i,\boldsymbol{\theta},\tau) \in Z_g} \Big(f(\boldsymbol{\theta}^*) - f(\boldsymbol{\theta})\Big)\right] \leq R_{T_B} \leq \sqrt{C_1 T_B e^{2C_a} \alpha_{T_B} \gamma_{T_B}} + 2. \tag{6.16}$$

Using $P(E_1 \cap E_2) \geq P(E_1) + P(E_2) - 1$, with probability at least $1 - \delta$,

$$R^{st}_{T,a} \leq \sqrt{C_1 T \alpha_T \gamma_T} + \sqrt{C_1 T_B e^{2C_a} \alpha_{T_B} \gamma_{T_B}} + 4. \tag{6.17}$$

Finally, by Cauchy-Schwarz inequality,

$$R^{st}_{T,a} \leq \sqrt{2C_1 \Big(T \alpha_T \gamma_T + (T + \phi_g) e^{2C_a} \alpha_{T+\phi_g} \gamma_{T+\phi_g}\Big)} + 4, \tag{6.18}$$

and since α_t and γ_t are non-decreasing, replacing both terms by their values at $T + \phi_g$ yields

$$R^{st}_{T,a} \leq \sqrt{2C_1 \Big([T + \phi_g] \alpha_{T+\phi_g} \gamma_{T+\phi_g} \Big(1 + e^{2C_a}\Big)\Big)} + 4. \tag{6.19}$$

This proves the theorem.

6.2.3 Theorem 2 (Time-Varying reward)

Consider f_t under assumptions 5 and 6. We first recall the high-probability upper bound of the regret in TV-GP-UCB (Bogunovic et al., 2016b [129]).

Single agent TV-GP-UCB

Regret in the single agent TV-GP-UCB framework is defined analogously to (5.27),

$$R_T^{single} = \sum_{t=1}^{T} f_t(\boldsymbol{\theta}_t^*) - f_t(\boldsymbol{\theta}_t) \tag{6.20}$$

highlighting that this is a tracking problem in contrast with Assumption 4.

Theorem 5 ((Bogunovic et al., 2016b [129])) *Let $\boldsymbol{\Theta} \subset [0, r]^d$ be compact and convex. Assume the time-varying model in Assumption 5 and the smoothness tail bound in Assumption 6 with constants (q_1, q_2). Fix $\delta \in (0, 1)$, and define $\alpha_T = 2\log\left(\frac{\pi^2 T^2}{\delta}\right) + 2d\log\left(rdq_2T^2\sqrt{\log\frac{dq_1\pi^2T^2}{\delta}}\right)$. Let $C_1 = \frac{8}{\log(1+\sigma_n^{-2})}$. Then the TV-GP-UCB algorithm satisfies, with probability at least $1 - \delta/2$,*

$$R_T^{single} \leq \sqrt{C_1 T \alpha_T \tilde{\gamma}_T} + 2, \tag{6.21}$$

and moreover, for any $\tilde{N} \in \{1, \ldots, T\}$,

$$R_T^{single} \leq \sqrt{C_1 T \alpha_T \left(\frac{T}{\tilde{N}} + 1\right)\left(\gamma_{\tilde{N}} + \tilde{N}^3\epsilon\right)} + 2. \tag{6.22}$$

Here

$$\tilde{\gamma}_T = \max_{X \subset \boldsymbol{\Theta}, |X| = T} \frac{1}{2}\log\det\left(\boldsymbol{I}_T + \sigma_n^{-2}\boldsymbol{K}_T^{dyn}\right), \tag{6.23}$$

with $\boldsymbol{K}_T^{dyn} = \left[k(\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)(1-\epsilon)^{|i-j|/2}\right]_{(i,j)\in[1,T]^2, (\boldsymbol{\theta}_i, \boldsymbol{\theta}_j)\in\boldsymbol{\Theta}^2}$, and $\gamma_{\tilde{N}}$ defined as in (5.29).

Proof of Theorem 2

Both (Bogunovic et al., 2016b [129]) and (Desautels et al., 2014 [130]) follow the same steps for bounding the regret as in (Srinivas et al., 2012 [94]): (i) build a time-dependent ε_t-net for D using the derivative tail bound (Assumption 6); (ii) apply Gaussian concentration on the grid and union-bound jointly over time and grid points. This is

where the expression of the schedule differ. (iii) lift the grid bound to the continuum; and (iv) convert $\sum_t \sigma_t^2(\cdot)$ into an information-gain term (static γ_T or time-varying $\tilde{\gamma}_T$). The exploration schedule is used only to certify the uniform confidence event $|f_t(\boldsymbol{\theta}) - \mu_t(\boldsymbol{\theta}, t)| \leq \alpha_t^{1/2} \sigma_t(\boldsymbol{\theta}, t)$; the maximum-information-gain bounds are schedule-independent.

$$\alpha_t := \max\left\{ \alpha_t^{\mathrm{UCB}},\ \alpha_t^{\mathrm{TV}} \right\}. \tag{6.24}$$

where

$$\alpha_t^{\mathrm{UCB}} = 2\log\left(\frac{4\pi t^2}{3\delta}\right) + 2d\log\left(t^2\, d\, q_2\, r\sqrt{\log\left(\frac{8d\, q_1}{\delta}\right)}\right), \tag{6.25}$$

$$\alpha_t^{\mathrm{TV}} = 2\log\left(\frac{\pi^2 t^2}{2\delta}\right) + 2d\log\left(r\, d\, q_2\, t^2\sqrt{\log\left(\frac{d\, q_1\, \pi^2 t^2}{2\delta}\right)}\right), \tag{6.26}$$

Since α_t is the pointwise maximum of two valid schedules, its band contains both, and by a union bound (splitting δ if desired) the same high-probability event holds. Consequently, all downstream steps (variance-to-information-gain conversion and MI inflation at gossip) go through unchanged, and we inherit the regret guarantees while paying only constant factors through $\sqrt{\alpha_T}$.

Similarly with section 6.2.2, we decompose the cumulative regret in (5.27) into gossip and non-gossip parts.

$$Z_{ng} := \{(\boldsymbol{\theta}_{a,t}, t) : t \in \{1, \ldots, T_{\mathrm{iter}}\},\ t \bmod t_g \neq 0\}, \qquad Z_g := \bigcup_{t\in\{1,\ldots,T_{\mathrm{iter}}\},\ t \bmod t_g = 0} S_{a,t}. \tag{6.27}$$

$$R_{a,T}^{dyn} = \mathbb{E}\left[\sum_{(\boldsymbol{\theta},\tau)\in Z_a} \Big(f_\tau(\boldsymbol{\theta}^*) - f_\tau(\boldsymbol{\theta})\Big)\right] = \sum_{(\boldsymbol{\theta},\tau)\in Z_{ng}} \Big(f_\tau(\boldsymbol{\theta}^*) - f_\tau(\boldsymbol{\theta})\Big) + \mathbb{E}\left[\sum_{(\boldsymbol{\theta},\tau)\in Z_g} \Big(f_\tau(\boldsymbol{\theta}^*) - f_\tau(\boldsymbol{\theta})\Big)\right]. \tag{6.28}$$

We use T and ϕ_g (6.14) from section 6.2.2. Under Assumptions 5 and 6, and using the schedule α_t we have, with probability at least $1 - \delta/2$:

$$E_1 : \sum_{(\boldsymbol{\theta},\tau)\in Z_{ng}} \Big(f_\tau(\boldsymbol{\theta}^*) - f_\tau(\boldsymbol{\theta})\Big) \leq R_T^{single} \leq \sqrt{C_1 T \alpha_T \tilde{\gamma}_T} + 2 \tag{6.29}$$

and

$$E_2 : \mathbb{E}\left[\sum_{(\boldsymbol{\theta},\tau)\in Z_g} \Big(f_\tau(\boldsymbol{\theta}^*) - f_\tau(\boldsymbol{\theta})\Big)\right] \leq R_{T_B} \leq \sqrt{C_1 T_B e^{2C_a} \alpha_{T_B} \tilde{\gamma}_{T_B}} + 2. \tag{6.30}$$

Note that the maximum mutual information $\tilde{\gamma}_T$ and $\tilde{\gamma}_{T_B}$ have the exact same expressions as (5.29). What changes is the kernel matrix in (5.30), where K_T^{dyn} and $K_{T_B}^{dyn}$ are used to compute the Mutual information. In (Bogunovic et al., 2016b [129]) They provide an

upper bound of such quantities using the maximum mutual information at each in the stationary domain, hence (6.22). This gives us, probability at least $1-\delta/2$:

$$E_1 : \sum_{(\boldsymbol{\theta},\tau)\in Z_{ng}} \Big(f_\tau(\boldsymbol{\theta}^*)-f_\tau(\boldsymbol{\theta})\Big) \leq R_T^{single} \leq \sqrt{C_1 T \alpha_T \Big(\frac{T}{\tilde{N}}+1\Big)\Big(\gamma_{\tilde{N}}+\tilde{N}^3\epsilon\Big)}+2, \qquad \forall \tilde{N} \in [1,T] \tag{6.31}$$

and

$$E_2 : \mathbb{E}\left[\sum_{(\boldsymbol{\theta},\tau)\in Z_g} \Big(f_\tau(\boldsymbol{\theta}^*)-f_\tau(\boldsymbol{\theta})\Big)\right] \leq R_{T_B} \leq \sqrt{C_1 T_B e^{2C_a} \alpha_{T_B} \Big(\frac{T_B}{\tilde{N}}+1\Big)\Big(\gamma_{\tilde{N}}+\tilde{N}^3\epsilon\Big)}+2, \qquad \forall \tilde{N} \in [1,T_B]. \tag{6.32}$$

Hence, with probability at least $1-\delta$, and choosing the same $\tilde{N}$ for both events, we get :

$$R_{a,T}^{dyn} \leq \sqrt{2C_1\Big(\gamma_{\tilde{N}}+\tilde{N}^3\epsilon\Big)\left[T\alpha_T\Big(\frac{T}{\tilde{N}}+1\Big)+(T+\phi_g)e^{2C_a}\alpha_{T+\phi_g}\Big(\frac{T+\phi_g}{\tilde{N}}+1\Big)\right]}+4, \qquad \forall \tilde{N} \in [1,T]. \tag{6.33}$$

6.2.4 Experimental Details

This section provides additional information complementing Section 5.5.

Synthetic Static Reward

The following benchmark functions are used to evaluate X-KB-UCB in static environments:

— **Rastrigin (d=4)** :

$$f_{\text{Ras}}(\boldsymbol{\theta}) = 10d + \sum_{i=1}^{d}\Big(\boldsymbol{\theta}_i^2 - 10\cos(2\pi\boldsymbol{\theta}_i)\Big), \boldsymbol{\theta} \in [-5.12,\, 5.12]^4. \tag{6.34}$$

— **Rosenbrock (d=3)**:

$$f_{\text{Ros}}(\boldsymbol{\theta}) = \sum_{i=1}^{d-1}\Big(100\Big(\boldsymbol{\theta}_{i+1}-\boldsymbol{\theta}_i^2\Big)^2 + (1-\boldsymbol{\theta}_i)^2\Big), \boldsymbol{\theta} \in [-5,\, 10]^3. \tag{6.35}$$

— **Styblinski-Tang (d=4)**:

$$f_{\text{ST}}(\boldsymbol{\theta}) = \tfrac{1}{2}\sum_{i=1}^{d}\Big(\boldsymbol{\theta}_i^4 - 16\boldsymbol{\theta}_i^2 + 5\boldsymbol{\theta}_i\Big), \quad \boldsymbol{\theta} \in [-5,\, 5]^4. \tag{6.36}$$

— **Schwefel 2.26 (d=3)**:

$$f_{\text{Sch}}(\boldsymbol{\theta}) = 418.9829\, d - \sum_{i=1}^{d} \boldsymbol{\theta}_i \sin\left(\sqrt{|\boldsymbol{\theta}_i|}\right), \boldsymbol{\theta} \in [-500,\, 500]^3. \tag{6.37}$$

— **Powell (singular, d=4)**:

$$f_{\text{Pow}}(\boldsymbol{\theta}) = (\boldsymbol{\theta}_1+10\boldsymbol{\theta}_2)^2+5(\boldsymbol{\theta}_3-\boldsymbol{\theta}_4)^2+(\boldsymbol{\theta}_2-2\boldsymbol{\theta}_3)^4+10(\boldsymbol{\theta}_1-\boldsymbol{\theta}_4)^4, \boldsymbol{\theta} \in [-5,\, 5]^4. \tag{6.38}$$

Further results with a number of agents $N = 4$, $N = 8$, and $N = 12$ respectively displayed in Table. 6.2 and Table. 6.4. These results further confirm the remarks in Section. **??**.

	Block A: vary N_a $N = 4$, $t_g = 1$			**Block B: vary t_g** $N = 4$, $N_a = N$		
	$N_a = N$	$0.8N$	$0.5N$	$t_g = 1$	$t_g = 30$	$t_g = 60$
rastrigin (d=4)	**60.140**	61.078	62.077	**60.140**	64.683	65.008
rosenbrock (d=3)	52.287	**49.284**	49.538	**52.287**	99.017	97.713
styblinski_tang (d=4)	**35.853**	37.522	36.509	**35.853**	54.346	55.485
schwefel (d=3)	**1190.122**	1204.829	1206.759	1190.122	1185.093	**1181.585**
powell (d=4)	3309.859	3021.473	**2951.404**	**3309.859**	6677.535	6457.541

Table 6.2 – Average instantaneous regret $rt_$ (lower is better) for $N = 4$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**.

The hyperparameters of the GPs (the lengthscale and the outputscale) of each agent are learned with every iteration $t \in [1, \ldots, T_{iter}]$ using maximum log-likelihood (see (Rasmussen et al., 2006 [87])). The acquisition function is maximized using a randoms sampling over 100000 candidates.

Synthetic Time-Varying Reward

Figure 6.6 illustrates the reward function, modeled according to Assumption 5, at three different time steps ($t = 10,\, 50,\, 90$) under drift levels $\epsilon \in \{0.001, 0.01, 0.03\}$, in a two-dimensional space. The plots highlight how the reward landscape evolves over time and demonstrate the pronounced sensitivity of the function to the drift parameter ϵ.

The acquisition function is optimized via grid sampling, using 50 points per dimension. The selected arm corresponds to the grid point that attains the highest acquisition value.

	Block A: vary N_a $N = 8$, $t_g = 1$			**Block B: vary t_g** $N = 8$, $N_a = N$		
	$N_a = N$	$0.8N$	$0.5N$	$t_g = 1$	$t_g = 30$	$t_g = 60$
rastrigin (d=4)	64.391	**63.432**	63.999	**64.391**	72.906	72.204
rosenbrock (d=3)	**85.623**	93.018	92.992	**85.623**	196.546	197.215
styblinski_tang (d=4)	**50.667**	61.470	61.311	**50.667**	103.180	93.570
schwefel (d=3)	120.060	1214.844	**1211.386**	**1210.060**	1219.815	1220.090
powell (d=4)	5460.591	5010.299	**4942.631**	**5460.591**	11958.663	11074.484

Table 6.3 – Average instantaneous regret $rt_$ (lower is better) for $N = 8$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**.

	Block A: vary N_a $N = 12$, $t_g = 1$			**Block B: vary t_g** $N = 12$, $N_a = N$		
	$N_a = N$	$0.8N$	$0.5N$	$t_g = 1$	$t_g = 30$	$t_g = 60$
rastrigin (d=4)	66.878	67.252	**66.840**	**66.878**	73.274	73.221
rosenbrock (d=3)	144.638	135.101	**132.080**	**144.638**	294.918	306.498
styblinski_tang (d=4)	**91.435**	92.701	98.948	**91.435**	134.189	128.418
schwefel (d=3)	**1224.084**	1237.489	1231.136	**1224.084**	1229.465	1228.699
powell (d=4)	**7687.572**	7791.842	7736.240	**7687.572**	20149.818	19473.558

Table 6.4 – Average instantaneous regret $rt_$ (lower is better) for $N = 12$ agents under varying gossip connectivity and communication budget. **Block A** varies the expected number of received tuples per round (N_a) at fixed $t_g = 1$. **Block B** varies the gossip period ($t_g \in \{1, 30, 60\}$) at fixed $N_a = N$. Per row, the best value *within each block* is shown in **bold**.

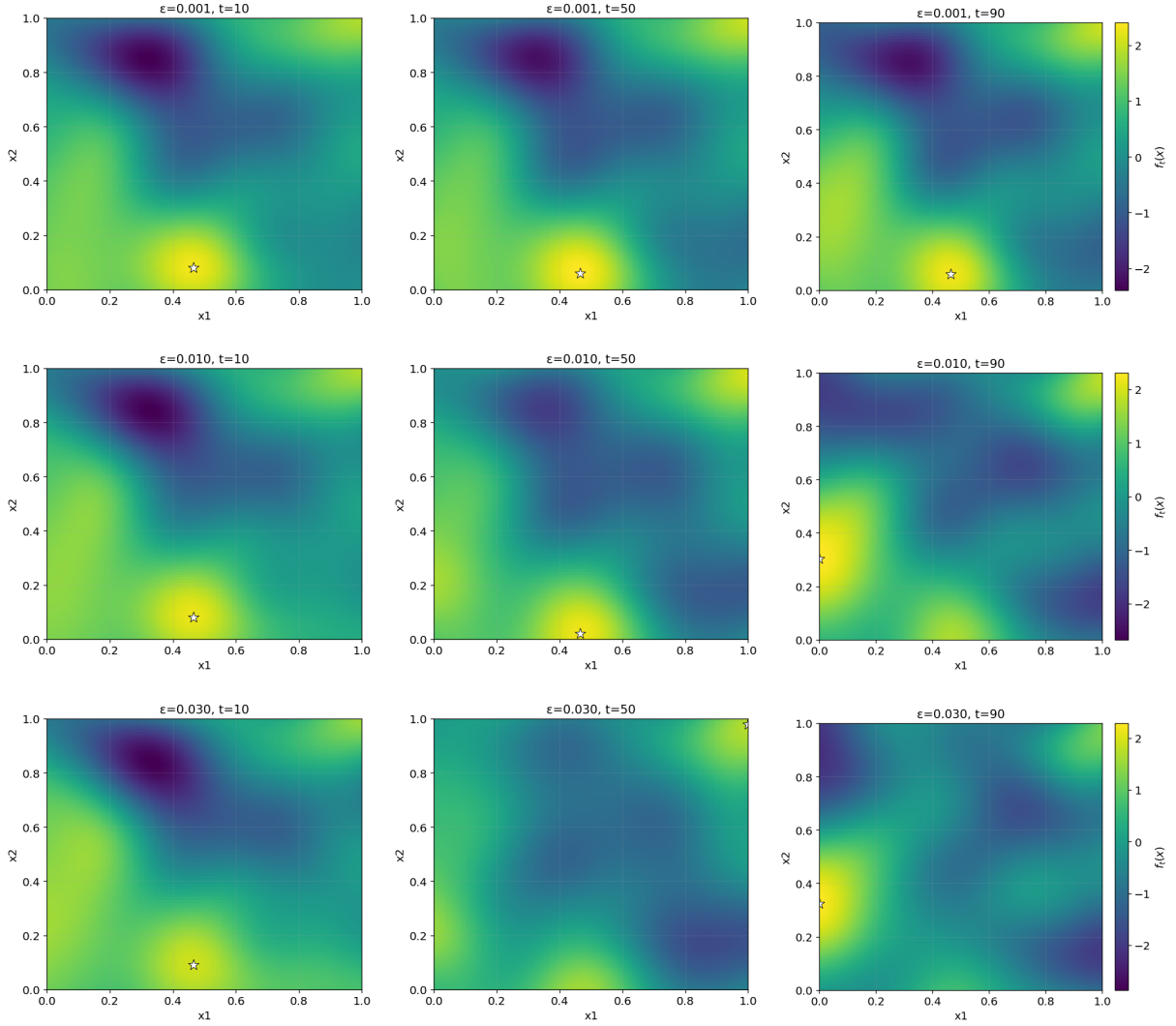
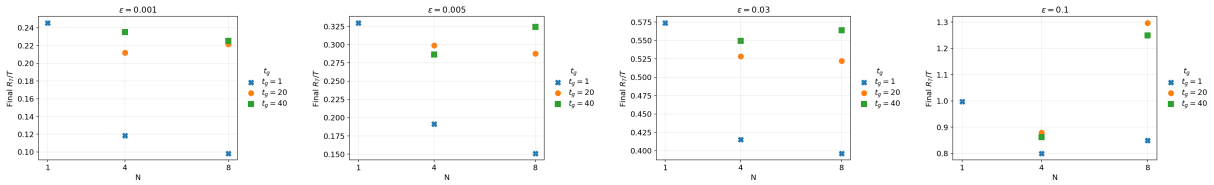


Figure 6.6 – Time-varying reward snapshots for drift levels $\varepsilon \in \{0.001, 0.01, 0.03\}$ at times $t \in \{10, 50, 90\}$. Each row fixes ε (top to bottom: 0.001, 0.01, 0.03) and each column fixes t (left to right: 10, 50, 90). The white star marks the instantaneous maximizer $\boldsymbol{\theta}_t^\star$ on the rendering grid in each panel.



(a) $\epsilon = 0.001$ (b) $\epsilon = 0.005$ (c) $\epsilon = 0.03$ (d) $\epsilon = 0.1$

Figure 6.7 – Time-varying experiments with $N_a = N$: final average regret as a function of the number of agents and the gossip period t_g for different drift levels ϵ. Lower is better.

While the optimal reward is the highest actual reward in the function. The observations are noisy, with a noise variance of $\sigma_n^2 = 0.01$.

Figure 6.7 reports the final average cumulative regret for $N_a = N$ across several drift levels ϵ. As expected, larger drift makes tracking harder. Across all ϵ, performance consistently improves with more agents and smaller gossip periods t_g. In short, more frequent communication with a larger team yields the lowest regret in the time-varying setting.

Chapter 7

Résumé en français

Cette thèse porte sur l'auto-configuration des récepteurs de télécommunication opérant dans des environnements difficiles et fortement variables. Elle s'intéresse principalement aux communications acoustiques sous-marines, pour lesquelles les performances du récepteur dépendent conjointement des conditions de propagation, des caractéristiques du signal transmis et de nombreux hyperparamètres associés aux traitements numériques. Ces hyperparamètres sont généralement réglés manuellement par des experts, à partir de connaissances partielles du canal ou de règles heuristiques. Une telle démarche est coûteuse, difficilement généralisable et peu adaptée aux environnements dans lesquels les conditions de propagation évoluent au cours de la communication.

L'objectif de cette thèse est de remplacer ces procédures manuelles par des méthodes d'optimisation capables de sélectionner automatiquement les configurations du récepteur. Le problème est formulé comme l'optimisation d'une fonction boîte noire : la relation entre les hyperparamètres et les performances finales du récepteur n'est pas disponible sous forme analytique, ses gradients ne peuvent pas être calculés de manière fiable et chaque évaluation nécessite l'exécution complète de la chaîne de réception. L'optimisation bayésienne est retenue comme cadre méthodologique principal en raison de son efficacité lorsque le nombre d'évaluations disponibles est limité.

Les contributions sont développées progressivement. Une première méthode, appelée *Search method*, recherche une configuration adaptée à des données enregistrées, à un canal rejoué ou à une simulation considérée comme stationnaire pendant l'optimisation. Une deuxième formulation, appelée *Tracking method*, permet de modifier les hyperparamètres en ligne lorsque le canal évolue d'un paquet au suivant. Cette formulation conduit à WandS, une architecture centralisée à deux agents associant exploration, exploitation prudente et modélisation temporelle de la fonction objectif. Enfin, la structure multi-agent est étendue à un cadre décentralisé avec la méthode X-KB-UCB, dans laquelle les agents échangent périodiquement des observations par un mécanisme de *gossip* sans dépendre d'un coordinateur central.

7.1 Chapitre 1 : Problème de réglage fin du récepteur

Ce chapitre présente le contexte des communications acoustiques sous-marines et formalise le problème de réglage automatique du récepteur. Dans un environnement sous-marin, les communications radio et optiques sont fortement limitées par l'atténuation et la diffusion dans l'eau. Les ondes acoustiques constituent donc la principale solution pour établir des communications à moyenne ou longue portée. Elles permettent d'atteindre des distances considérables, mais avec une bande passante réduite, une latence importante et des conditions de propagation particulièrement difficiles.

La vitesse du son dans l'eau dépend notamment de la température, de la salinité et de la pression. Elle n'est donc pas uniforme dans la colonne d'eau. Les variations du profil de célérité provoquent des phénomènes de réfraction qui modifient les trajectoires suivies par l'énergie acoustique. Ces trajectoires dépendent également de la profondeur de l'émetteur et du récepteur, de la surface de la mer, du fond marin et des différents obstacles présents dans l'environnement.

À ces effets s'ajoute la propagation par trajets multiples. Le signal reçu est constitué de plusieurs répliques du signal transmis, ayant des amplitudes, des phases et des retards différents. Lorsque les écarts de retard deviennent comparables à la durée des symboles, les symboles successifs se superposent et produisent de l'interférence intersymbole. La réponse impulsionnelle du canal est alors caractérisée par sa répartition de puissance en fonction du retard et, en particulier, par son étalement temporel quadratique moyen.

Les mouvements relatifs de l'émetteur, du récepteur ou des frontières du canal provoquent également des effets Doppler. La variation des longueurs des trajets entraîne une compression ou une dilatation temporelle du signal, tandis que les fluctuations des amplitudes des trajets produisent un étalement Doppler. Ces phénomènes sont particulièrement marqués dans les communications acoustiques en raison de la faible vitesse de propagation du son, proche de 1500 m/s. Ils perturbent la synchronisation temporelle et fréquentielle, font dériver les instants d'échantillonnage et entraînent des rotations de phase.

Les conditions de propagation évoluent à plusieurs échelles temporelles. Les variations saisonnières ou océanographiques lentes peuvent être négligées à l'échelle d'un paquet. À l'inverse, les vagues internes, la houle, les mouvements des plateformes et les variations de la surface peuvent modifier le canal sur des durées allant de quelques fractions de seconde à plusieurs minutes. Les fluctuations plus rapides, qui ne peuvent pas être suivies par les boucles de traitement du récepteur, sont assimilées à du bruit. Les hyperparamètres du récepteur doivent donc refléter les caractéristiques instantanées ou récentes du canal. Lorsque celles-ci évoluent, la région optimale de l'espace des hyperparamètres se déplace également.

Le système étudié est une liaison SISO organisée en paquets et utilisant une modulation

QPSK. Chaque paquet contient un préambule de 511 symboles destiné à la détection et à la synchronisation, une séquence d'apprentissage utilisée par l'égaliseur, les données utiles et une séquence CRC. Le signal est protégé par un code convolutif de taux 1/2. Quatre bandes passantes, comprises entre 4,5 kHz et 15 kHz, sont considérées. L'augmentation de la bande passante accroît le débit symbole mais augmente également la longueur apparente du canal en nombre de symboles, ce qui rend l'égalisation plus difficile.

La chaîne de réception comprend tout d'abord une banque de filtres adaptés testant différentes hypothèses de Doppler. Cette banque fournit une estimation grossière du début du paquet et de la dilatation temporelle. Le signal est ensuite rééchantillonné afin de compenser la composante principale du Doppler. La résolution de cette banque constitue elle-même un hyperparamètre : une résolution fine améliore la compensation, mais augmente la complexité de calcul.

Après cette première synchronisation, le signal est traité par un égaliseur à retour de décision, ou DFE. Celui-ci comprend un filtre direct de longueur L_A et un filtre de retour de longueur L_B. Le filtre direct traite notamment les interférences précédant le symbole courant, tandis que le filtre de retour utilise les décisions passées pour supprimer les interférences postérieures. Les coefficients sont adaptés à l'aide de l'algorithme IPNLMS, dont le fonctionnement dépend des pas d'adaptation μ_A et μ_B, ainsi que d'un paramètre β contrôlant le caractère proportionnel de la mise à jour.

Une boucle à verrouillage de phase est associée au DFE afin de corriger les décalages fréquentiels résiduels. Son comportement dépend principalement d'un gain proportionnel K_1 et d'un gain intégral K_2. Les principaux hyperparamètres étudiés dans cette thèse sont ainsi les longueurs L_A et L_B, les pas d'adaptation μ_A et μ_B, le paramètre β et les gains K_1 et K_2. Ces paramètres sont fortement couplés. Par exemple, une augmentation de l'étalement temporel nécessite généralement un filtre de retour plus long, tandis que les pas d'adaptation et les gains de la boucle de phase doivent être adaptés à la vitesse de variation du canal.

Afin d'automatiser leur réglage, une fonction objectif mesurant les performances du récepteur est définie. Lorsque les données transmises sont inconnues, le taux d'erreur binaire ne peut pas être calculé directement. La fonction objectif repose donc sur une estimation du rapport signal sur bruit à la sortie du récepteur. Toutefois, cette métrique présente une difficulté particulière liée à un état dégénéré du DFE, appelé dans cette thèse *hallucination*.

Lors d'une hallucination, la contribution du filtre direct tend vers zéro alors que le filtre de retour génère progressivement ses propres décisions. Le récepteur peut alors produire une suite de symboles apparemment cohérente, associée à une estimation artificiellement élevée du rapport signal sur bruit, alors que les données décodées sont incorrectes. Une telle situation peut notamment apparaître après une erreur de décision importante ou

après l'apparition brutale d'un nouveau trajet de propagation.

Une procédure de détection est donc proposée. Elle surveille l'énergie des dernières sorties du filtre direct. Lorsque cette énergie devient inférieure à un seuil donné, la configuration est considérée comme hallucinatoire et reçoit une valeur fortement pénalisée de -50 dB. Dans le cas contraire, la valeur moyenne du rapport signal sur bruit de sortie est utilisée. Cette fonction constitue la boîte noire optimisée dans les chapitres suivants.

L'étude de cette fonction montre qu'elle varie fortement avec le canal, la réalisation du bruit et la dimension de l'espace de recherche. Elle présente des régions de bonnes performances, des régions de faibles performances et des régions sujettes aux hallucinations, séparées par des transitions parfois abruptes. Les hyperparamètres sont en outre fortement corrélés, ce qui rend leur optimisation indépendante peu pertinente. Une recherche exhaustive devient rapidement impossible : avec seulement cinquante valeurs possibles pour cinq hyperparamètres, plus de 3×10^8 évaluations seraient nécessaires.

La formulation du réglage du récepteur comme un problème d'optimisation boîte noire, la métrique de performance proposée et la procédure de détection des hallucinations constituent les principales contributions de ce chapitre.

7.2 Chapitre 2 : Fonctions boîtes noires et optimisation bayésienne

Le deuxième chapitre présente les méthodes susceptibles d'optimiser la fonction définie dans le chapitre précédent. Le problème général consiste à rechercher une configuration $\boldsymbol{\theta}$ optimisant une fonction $f(\boldsymbol{\theta})$ qui ne peut être observée qu'à travers des évaluations ponctuelles et potentiellement bruitées. Aucune expression analytique exploitable, aucun gradient fiable et aucune information complète sur la structure de la fonction ne sont disponibles.

Plusieurs familles de méthodes sans dérivées sont d'abord examinées. Les méthodes de recherche directe ou de recherche par motifs explorent le voisinage du point courant selon un ensemble de directions. Elles acceptent les déplacements produisant une amélioration et réduisent progressivement la taille de leur maillage lorsque les points testés ne sont pas satisfaisants. Ces méthodes sont relativement simples et peuvent tolérer une certaine non-régularité, mais elles nécessitent souvent un grand nombre d'évaluations, notamment lorsque la dimension augmente.

Les méthodes fondées sur une région de confiance construisent un modèle local de la fonction à partir des points déjà évalués. Ce modèle, généralement polynomial ou fondé sur des fonctions de base radiale, est optimisé dans un voisinage du point courant. La

taille de ce voisinage est ajustée en fonction de l'accord entre l'amélioration prédite et l'amélioration réellement obtenue. Ces méthodes peuvent être efficaces près d'un optimum, mais leur modèle local devient moins pertinent lorsque la fonction est très bruitée ou évolue au cours du temps.

Les méthodes d'approximation stochastique estiment quant à elles un gradient à partir de perturbations aléatoires de la configuration. Elles sont potentiellement mieux adaptées aux espaces de grande dimension, mais leur convergence dépend fortement des niveaux de bruit et du choix des pas d'adaptation. Enfin, les méthodes évolutionnaires manipulent une population de solutions par mutation, croisement et sélection. Elles sont robustes et largement applicables, mais nécessitent généralement des milliers, voire des dizaines de milliers d'évaluations.

L'optimisation bayésienne se distingue de ces approches par son objectif d'efficacité en nombre d'évaluations. Elle construit un modèle probabiliste de la fonction à optimiser, puis utilise l'incertitude de ce modèle pour déterminer les configurations les plus informatives ou les plus prometteuses. Cette propriété est particulièrement adaptée au réglage des récepteurs, car chaque évaluation implique le traitement complet d'un paquet et peut donc être coûteuse.

Le modèle de substitution principalement utilisé dans cette thèse est un processus gaussien. Avant toute observation, celui-ci définit une distribution sur les fonctions possibles à partir d'une fonction moyenne et d'un noyau de covariance. Après chaque évaluation, la distribution est mise à jour afin de fournir, pour chaque configuration candidate, une moyenne prédictive et une variance. La moyenne représente la performance attendue, tandis que la variance quantifie l'incertitude associée à cette prédiction.

La corrélation entre deux configurations est définie par le noyau. Un noyau exponentiel quadratique est notamment utilisé dans les différentes méthodes proposées. Sa longueur de corrélation contrôle la vitesse avec laquelle la fonction peut varier dans l'espace des hyperparamètres. Une grande longueur produit un modèle lisse et étend l'influence des observations sur une large région. Une longueur plus faible permet de représenter des variations locales plus rapides, mais nécessite davantage d'observations.

À chaque itération, une fonction d'acquisition est construite à partir de la moyenne et de la variance du modèle. Son optimisation fournit la prochaine configuration à tester. Différentes politiques sont étudiées. L'amélioration espérée favorise les configurations pouvant à la fois améliorer la meilleure valeur connue et présenter une incertitude suffisante. La probabilité d'amélioration privilégie les points ayant une forte probabilité de dépasser la meilleure observation, sans tenir directement compte de l'amplitude de l'amélioration.

Les politiques de type borne de confiance combinent explicitement la moyenne et l'écart-type prédictifs. Une politique UCB favorise les régions à la fois prometteuses et

incertaines, ce qui encourage l'exploration. Une politique LCB peut au contraire privilégier les régions dont la bonne performance est prédite avec davantage de certitude, ce qui conduit à un comportement plus conservateur. Un paramètre d'exploration contrôle l'importance relative accordée à l'incertitude.

Le *Tree-structured Parzen Estimator*, ou TPE, constitue une autre approche. Au lieu de construire directement un modèle probabiliste de la valeur de la fonction, il sépare les configurations déjà testées en un groupe performant et un groupe peu performant. Il estime ensuite la densité des configurations dans chacun de ces groupes et recherche de nouveaux points ressemblant davantage aux bonnes configurations qu'aux mauvaises. Cette méthode est particulièrement adaptée aux espaces mixtes contenant des paramètres réels, entiers ou conditionnels.

L'optimisation bayésienne est ainsi retenue comme cadre méthodologique pour la suite du manuscrit. Elle permet d'utiliser progressivement les observations collectées, de quantifier l'incertitude et d'arbitrer entre l'exploitation des régions déjà identifiées et l'exploration de configurations encore mal connues.

7.3 Chapitre 3 : Méthode de recherche hors ligne

Ce chapitre introduit la *Search method*, destinée aux situations dans lesquelles les conditions opérationnelles peuvent être considérées comme fixes pendant une session d'optimisation. Il peut s'agir de données enregistrées qui n'ont pas été correctement décodées, d'un canal mesuré puis rejoué, ou encore d'une simulation représentative d'une opération future. Le même ensemble de données est traité à chaque itération, de sorte que la fonction objectif reste stationnaire pendant la recherche.

La méthode repose sur un agent d'optimisation bayésienne composé d'un modèle de substitution et d'une politique d'acquisition. Une phase initiale évalue aléatoirement plusieurs configurations afin de construire un premier ensemble de données. À chaque itération suivante, le modèle est ajusté aux observations disponibles, la politique propose une nouvelle configuration et le récepteur traite les données avec celle-ci. La performance mesurée est ensuite ajoutée à l'ensemble d'apprentissage. Ce processus est répété jusqu'à épuisement du budget d'évaluations.

Pour cette première preuve de concept, la recherche porte sur les longueurs L_A et L_B, les pas d'adaptation μ_A et μ_B et le paramètre β. La résolution de la banque Doppler est fixée suffisamment finement pour compenser précisément le Doppler simulé, ce qui permet de ne pas optimiser les gains de la boucle de phase dans ce chapitre.

Deux références sont utilisées pour évaluer la méthode. La première est un échantillon-

nage aléatoire uniforme de l'espace des hyperparamètres. La seconde utilise une estimation du canal obtenue à partir du préambule. L'étalement temporel estimé détermine alors les longueurs des filtres direct et de retour. Cette méthode dépend fortement de la qualité de l'estimation et de la durée du préambule. Si celui-ci est trop court par rapport au canal, l'étalement temporel est sous-estimé et les filtres obtenus ne couvrent pas l'ensemble des trajets significatifs.

Les premières expériences sont réalisées sur des canaux générés par un modèle à entropie maximale. Les politiques UCB, EI et TPE sont comparées à l'échantillonnage aléatoire et à la méthode fondée sur l'estimation du canal. Les agents d'optimisation bayésienne atteignent plus rapidement les régions de bonnes performances et évitent davantage les zones hallucinatoires. Dans une expérience où l'optimisation est déclenchée lorsque la méthode fondée sur l'estimation du canal échoue, environ 70 % des paquets concernés sont récupérés, avec un gain moyen proche de 3 dB sur la métrique de sortie.

La méthode est ensuite évaluée sur un canal issu d'une campagne de mesure réalisée au Loch Ness. Le profil de puissance mesuré présente plusieurs trajets dominants suivis d'une longue queue de réverbération. Afin de reproduire une évolution temporelle réaliste, ce profil et les facteurs de Rice estimés sont injectés dans le simulateur à entropie maximale, avec un étalement Doppler de 0,8 Hz.

Ce canal constitue un cas difficile pour le récepteur. La longue réverbération nécessite des filtres de grande taille, tandis que les configurations favorisant une représentation trop parcimonieuse risquent de négliger les trajets tardifs et de provoquer des hallucinations. À l'inverse, des réglages plus adaptés à un canal dispersif offrent une meilleure stabilité, mais peuvent réduire la performance maximale. La région de bonnes performances est donc étroite et entourée de nombreuses configurations hallucinatoires.

La référence fondée sur l'estimation du canal sous-estime l'étalement temporel et sélectionne des filtres beaucoup trop courts. L'échantillonnage aléatoire atteint rapidement un plateau, car la probabilité de sélectionner une configuration appartenant à la région optimale est faible. Les agents bayésiens, en particulier l'agent utilisant TPE, identifient plus efficacement cette région. Sur une évaluation portant sur plusieurs centaines de paquets et différents rapports signal sur bruit d'entrée, la Search method obtient le taux d'erreur paquet le plus faible sur l'ensemble des conditions testées. Le gain atteint environ vingt points de pourcentage par rapport à la méthode d'estimation du canal dans certaines conditions.

Ces expériences montrent que des différences relativement modestes sur la métrique de sortie peuvent produire des réductions importantes du taux d'erreur binaire ou du taux d'erreur paquet. Elles confirment également que la méthode peut être utilisée sans connaissance préalable détaillée de l'environnement.

La Search method est principalement adaptée à des applications hors ligne. Elle peut servir à récupérer des données importantes à partir d'enregistrements, ou à préparer une configuration avant une opération à l'aide de simulations et de mesures antérieures. Lorsque les données transmises sont connues, le taux d'erreur binaire peut directement remplacer la métrique proposée dans le premier chapitre.

La méthode présente néanmoins plusieurs limites. Le nombre d'hyperparamètres pouvant être optimisés conjointement reste limité par les capacités des modèles de substitution et des politiques d'acquisition. Le choix des hyperparamètres à inclure dans la recherche doit également être effectué avec soin, car certains peuvent avoir une influence faible dans de nombreux scénarios. Enfin, plusieurs exécutions du récepteur sont nécessaires avant la convergence. La méthode ne peut donc pas être appliquée directement à chaque nouveau paquet lorsque le canal évolue pendant la communication. Cette limitation conduit à la formulation du problème de suivi étudiée dans le chapitre suivant.

7.4 Chapitre 4 : Méthode de suivi en ligne

Le quatrième chapitre s'intéresse à la mise à jour des hyperparamètres pendant la communication. Contrairement au chapitre précédent, la fonction objectif dépend maintenant de l'indice du paquet. Les conditions de propagation peuvent changer entre deux paquets successifs, ce qui déplace la région optimale de l'espace de recherche. L'objectif n'est plus de déterminer une configuration unique, mais de suivre une succession de configurations adaptées au canal courant.

Une comparaison théorique entre la Search method et une méthode de suivi parfaite montre qu'une configuration unique optimisant les performances moyennes sur plusieurs contextes est généralement moins performante qu'une configuration choisie séparément pour chaque contexte. Cette différence devient stricte lorsque les hyperparamètres optimaux varient effectivement avec les conditions de propagation.

Pour respecter les contraintes temporelles, un mode de fonctionnement appelé *flywheel mode* est introduit. Après une phase d'initialisation, l'optimisation n'effectue qu'une itération pour chaque nouveau paquet. Les informations acquises sur les paquets précédents sont réutilisées pour sélectionner les hyperparamètres du paquet courant. L'algorithme accumule ainsi progressivement des connaissances tout en suivant l'évolution du canal.

Les méthodes sont testées sur un scénario simulé combinant le logiciel de tracé de rayons Bellhop et le modèle à entropie maximale. Un récepteur se déplace à 3 m/s entre des distances horizontales de 1900 m et 25 m par rapport à la source. L'émetteur est placé à 50 m de profondeur et le récepteur à 700 m. Le profil de célérité provient d'une mesure en Atlantique Nord. Les réponses impulsionnelles générées par Bellhop sont corrigées par

l'atténuation de Thorp, puis enrichies par des fluctuations statistiques dont l'étalement Doppler est compris entre 0,2 et 0,5 Hz.

La variation de la géométrie modifie principalement l'écart entre le trajet direct et les trajets réfléchis par la surface. L'étalement temporel évolue approximativement entre 37 et 74 ms, avec des zones d'ombre dans lesquelles certains trajets disparaissent. La durée totale du scénario est d'environ 10,4 minutes. Ce scénario est suffisamment long pour évaluer la convergence et la capacité de suivi, tout en conservant des régions de l'espace des hyperparamètres permettant un décodage correct.

Une première version de la Tracking method utilise un seul agent, un processus gaussien stationnaire et une politique UCB. Seules les longueurs L_A et L_B sont optimisées. Les résultats montrent que l'agent peut améliorer les performances par rapport à la référence fondée sur l'estimation du canal. Ils mettent également en évidence le déplacement de la région optimale lorsque la distance et l'étalement temporel varient.

Cette première version présente toutefois trois limites. Le processus gaussien est stationnaire alors que la fonction objectif évolue. Un agent unique doit arbitrer seul entre l'exploration, nécessaire pour détecter un changement de régime, et l'exploitation, nécessaire pour maintenir une communication fiable. Enfin, les régions hallucinatoires ne sont prises en compte qu'au moyen de la pénalité fixe définie dans le premier chapitre.

Pour résoudre ces difficultés, une méthode à deux agents, appelée WandS pour *Wise and Scout*, est proposée. L'agent Wise adopte un comportement prudent et exploite les configurations prédites comme performantes avec une faible incertitude. Il utilise une politique de type LCB. L'agent Scout utilise une politique UCB afin d'explorer les régions incertaines et de détecter les déplacements de la zone optimale.

Les deux agents utilisent un processus gaussien variant dans le temps. Celui-ci applique un facteur d'oubli aux observations passées. Plus une observation est ancienne, moins son influence sur la moyenne et la covariance du modèle est importante. Le facteur d'oubli peut être augmenté dans les environnements très dynamiques et réduit lorsque les conditions sont presque stationnaires. Le modèle reste ainsi sensible aux observations récentes sans être excessivement contraint par des états anciens du canal.

Une pénalité supplémentaire est intégrée aux politiques des agents afin de les éloigner des régions proches d'hallucinations déjà observées. Cette pénalité dépend de la distance entre la configuration candidate et l'hallucination la plus proche. Elle est forte au voisinage d'une hallucination récente, puis son influence diminue progressivement avec le temps. Ce mécanisme permet d'éviter les régions dangereuses tout en autorisant leur réexploration ultérieure lorsque le canal a suffisamment évolué.

Pour chaque paquet, le Wise et le Scout proposent chacun une configuration. Les deux propositions sont évaluées en parallèle. Une règle utilisant la métrique de sortie et

le résultat du CRC sélectionne ensuite la configuration appliquée. Par défaut, la proposition prudente du Wise est retenue. La proposition du Scout est sélectionnée lorsqu'elle présente une meilleure performance mesurée et satisfait les critères de validité. Les deux observations sont ajoutées à l'ensemble d'apprentissage afin de mettre à jour le modèle commun.

Les performances de WandS sont d'abord étudiées pour l'optimisation conjointe des longueurs L_A et L_B. La méthode converge généralement après quelques paquets et conserve une capacité de récupération rapide lors des changements abrupts du canal. Dans une zone d'ombre, le taux d'erreur peut augmenter fortement, mais l'exploration maintenue par le Scout permet de retrouver une configuration adaptée dès les paquets suivants.

À une bande passante de 15 kHz, WandS produit davantage de paquets ayant un taux d'erreur proche de zéro que les références fondées sur une recherche aléatoire exhaustive, sur la connaissance parfaite de l'étalement temporel ou sur l'estimation du canal avec un préambule fixe. La référence utilisant un préambule de 511 symboles reste compétitive à faible bande passante, mais se dégrade lorsque la bande augmente, car la durée du canal exprimée en symboles devient supérieure à la capacité d'observation du préambule. WandS demeure plus robuste, sans nécessiter d'ajustement préalable de cette durée.

La comparaison avec des méthodes de suivi à un seul agent utilisant UCB, EI ou PI montre également l'intérêt de séparer les fonctions d'exploration et d'exploitation. WandS obtient des taux d'erreur moyens et médians plus faibles et une distribution des performances plus concentrée vers les faibles taux d'erreur. Les essais de sensibilité indiquent que le facteur d'oubli ne doit être ni nul, ce qui empêcherait l'adaptation, ni trop élevé, ce qui supprimerait excessivement l'information acquise. Les résultats sont relativement peu sensibles aux paramètres des politiques dans une plage raisonnable, à condition que le Scout conserve un niveau d'exploration suffisant.

Une deuxième étude optimise conjointement les longueurs du DFE et les gains K_1 et K_2 de la boucle de phase. Lorsque la résolution de la banque Doppler est très fine, l'optimisation de la PLL est peu utile et le réglage des seules longueurs de filtres suffit. Lorsque cette résolution devient plus grossière, les décalages résiduels augmentent et le réglage de la PLL devient essentiel. Les expériences montrent que les gains de la boucle de phase et les longueurs des filtres doivent être optimisés conjointement. Une méthode qui ne règle que la PLL avec des filtres mal choisis reste fortement limitée.

La politique tenant compte des hallucinations améliore systématiquement les performances par rapport à sa variante sans pénalité dédiée. En moyenne sur les différentes résolutions Doppler étudiées, elle produit environ 30 % de paquets ayant un taux d'erreur proche de zéro. Ce résultat confirme qu'une pénalité fixe appliquée uniquement à la fonction objectif ne suffit pas et qu'il est utile d'intégrer explicitement la connaissance des régions hallucinatoires dans les politiques d'acquisition.

WandS est enfin évaluée par rejeu de deux canaux réels mesurés dans les ports de La Spezia et de Brest. Les enregistrements permettent respectivement de transmettre environ soixante-dix et vingt-quatre paquets, avec un rapport signal sur bruit fixé à 12 dB. Après une courte phase d'initialisation, la méthode se stabilise rapidement et dépasse la référence fondée sur l'estimation du canal dans le scénario de La Spezia. Ces résultats montrent qu'elle peut converger dans des conditions réalistes sans connaissance préalable de l'environnement.

Les principales contributions de ce chapitre sont la formulation du problème de suivi des hyperparamètres à l'échelle du paquet, l'introduction du mode *flywheel*, l'utilisation d'un modèle de substitution variant dans le temps, l'architecture à deux agents Wise et Scout et la pénalisation temporelle des régions hallucinatoires.

La méthode reste centralisée : un nœud commun synchronise les observations, ajuste le modèle partagé, coordonne les deux agents et sélectionne leur proposition. Cette architecture peut devenir difficile à mettre en œuvre lorsque les agents sont distribués entre plusieurs plateformes mobiles et disposent de liaisons de communication limitées. Cette difficulté motive l'étude d'une architecture décentralisée dans le dernier chapitre.

7.5 Chapitre 5 : Optimisation multi-agent décentralisée

Le dernier chapitre généralise la structure multi-agent à un cadre sans coordinateur central. Ce type d'architecture est pertinent pour les équipes de robots, les réseaux de capteurs et les essaims de véhicules sous-marins autonomes. Dans ces systèmes, la centralisation peut être limitée par la bande passante, l'énergie disponible, la portée des liaisons et les délais de propagation.

Dans un réseau de véhicules sous-marins, chaque plateforme peut optimiser localement ses décisions, mais une coordination est nécessaire pour éviter que tous les agents explorent simultanément les mêmes régions. Les communications acoustiques étant lentes, coûteuses en énergie et de faible capacité, les échanges doivent rester peu fréquents et contenir un volume réduit d'informations.

Le problème est formulé dans le cadre des bandits à processus gaussiens. À chaque itération, un agent sélectionne une action dans un espace continu et observe une récompense bruitée. Dans l'application au réglage du récepteur, une action correspond à une configuration d'hyperparamètres et la récompense à la performance obtenue. Dans un environnement stationnaire, tous les agents cherchent le même maximum fixe. Dans un environnement variable, ils doivent suivre un maximum évoluant avec le temps.

La performance d'une politique est mesurée à l'aide du regret cumulé. Le regret instantané correspond à la différence entre la récompense de la meilleure action et celle de l'action réellement sélectionnée. Une politique est dite sans regret lorsque son regret cumulé moyen tend vers zéro. Dans le cas variable, le regret dynamique compare l'action sélectionnée à la meilleure action disponible à chaque instant.

La méthode proposée, appelée X-KB-UCB, combine une politique UCB locale, des communications périodiques par *gossip* et une adaptation inter-agents du mécanisme de *Kriging–Believer*. Chaque agent conserve son propre processus gaussien et sélectionne ses actions localement. En dehors des instants de communication, son fonctionnement correspond à GP-UCB dans le cas stationnaire ou à TV-GP-UCB dans le cas variable.

Les communications ont lieu toutes les t_g itérations. À chaque tour de *gossip*, un agent transmet uniquement sa dernière action, la récompense observée et l'indice temporel correspondant. Une matrice de communication définit la probabilité qu'un message émis par un agent soit reçu par chacun de ses voisins. Le paramètre t_g contrôle directement le compromis entre la quantité d'information partagée et le coût des communications.

Si tous les agents utilisent les mêmes observations, le même modèle et la même politique, ils peuvent sélectionner des actions identiques et produire des mesures redondantes. Pour éviter cette situation, X-KB-UCB adapte le principe du Kriging–Believer, habituellement utilisé pour sélectionner plusieurs points non redondants lors d'une optimisation bayésienne par lots.

Lorsqu'un agent reçoit les actions choisies par ses voisins, il conserve temporairement sa moyenne prédictive, ajoute les actions reçues à la matrice utilisée pour calculer la covariance et recalcule sa variance. L'incertitude diminue alors autour des régions déjà explorées par les autres agents. L'agent maximise ensuite une borne de confiance supérieure utilisant cette variance modifiée. Il est ainsi encouragé à choisir une action différente de celles reçues tout en conservant le compromis exploration-exploitation de la politique UCB.

Après la sélection de sa nouvelle action, l'agent incorpore les véritables récompenses communiquées par ses voisins et sa propre nouvelle observation dans son ensemble de données. La décision reste donc locale, mais les informations disponibles sont enrichies périodiquement par les échanges. Ce fonctionnement distingue X-KB-UCB des méthodes centralisées de sélection par lots, dans lesquelles un décideur unique planifie simultanément toutes les actions.

L'ajout artificiel des actions reçues dans le calcul de la variance peut rendre le modèle excessivement confiant. Une correction théorique est donc introduite à l'aide de l'information mutuelle apportée par les observations reçues. Aux tours de *gossip*, la largeur de la borne de confiance est augmentée afin de garantir que la fonction réelle reste contenue dans l'intervalle avec une probabilité élevée.

L'analyse s'appuie sur la notion d'agent augmenté. Un agent augmenté représente un agent local ainsi que l'ensemble des observations qu'il a reçues. Cette abstraction permet de comparer directement différents régimes de communication. En l'absence de *gossip*, l'algorithme retrouve le comportement et les garanties d'un agent GP-UCB isolé. Lorsque les échanges ont lieu à chaque itération et que le réseau est fortement connecté, le comportement se rapproche de celui d'une méthode centralisée ou d'une optimisation par lots.

Des bornes de regret sont établies pour les fonctions stationnaires et pour les fonctions variant selon un modèle markovien. Dans le cas stationnaire, X-KB-UCB possède une propriété de non-regret avec une forte probabilité. Dans le cas variable, une borne de regret dynamique est obtenue en fonction de la vitesse de dérive, de la fréquence des communications, du nombre moyen de messages reçus et du gain maximal d'information. Lorsque la dérive décroît suffisamment avec l'horizon, le regret dynamique moyen tend également vers zéro.

Les expériences stationnaires utilisent notamment la fonction de Styblinski–Tang. Elles montrent que l'augmentation du nombre d'agents permet d'approcher une accélération linéaire lorsque les agents communiquent fréquemment. Une fréquence de *gossip* plus élevée réduit le regret, de même qu'une connectivité plus importante du réseau. Une légère perte peut apparaître lorsque de nombreux agents partagent les mêmes informations, en raison de la réduction de variance produite par le Kriging–Believer, mais le gain lié à l'exploration parallèle reste dominant.

Dans les expériences variables, plusieurs agents suivent une fonction synthétique dont le maximum se déplace au cours du temps. L'augmentation du nombre d'agents et la réduction de la période de communication améliorent nettement le suivi. La méthode demeure relativement robuste lorsque le facteur de dérive ou la variance du bruit utilisés dans les modèles des agents diffèrent modérément des valeurs réelles. Une dernière expérience utilise des mesures de température provenant de quarante-six capteurs. L'objectif est de suivre, au cours du temps, le capteur présentant la température la plus élevée. Sur cette fonction construite à partir de données réelles, le regret diminue lorsque le nombre d'agents augmente et lorsque les communications deviennent plus fréquentes. Ces résultats confirment que le partage d'observations améliore l'efficacité en nombre d'évaluations et la capacité de suivi.

Les contributions principales de ce chapitre sont l'introduction de X-KB-UCB, l'adaptation inter-agents du Kriging–Believer aux échanges par *gossip*, la formalisation de l'agent augmenté et l'établissement de garanties théoriques dans les cas stationnaire et variable. Le cadre proposé dépasse l'application particulière des communications acoustiques et peut être appliqué à d'autres systèmes distribués dans lesquels plusieurs agents optimisent une fonction commune avec un budget de communication limité.

7.6 Conclusion générale et perspectives

Cette thèse développe une chaîne complète de méthodes pour l'auto-configuration des récepteurs de télécommunication. Elle part de la définition d'une métrique de performance et d'une formulation boîte noire, puis traite successivement la recherche hors ligne, le suivi en ligne dans un environnement variable et la coordination décentralisée de plusieurs agents.

La Search method montre que l'optimisation bayésienne peut remplacer avantageusement les recherches exhaustives, l'échantillonnage aléatoire et certaines règles fondées sur l'estimation du canal. La Tracking method étend cette approche aux fonctions variant d'un paquet au suivant. WandS répartit les rôles d'exploration et d'exploitation entre deux agents et utilise un modèle temporel ainsi qu'une gestion explicite des hallucinations. X-KB-UCB généralise enfin cette structure aux réseaux décentralisés et fournit un cadre théorique permettant de quantifier les bénéfices des communications inter-agents.

Bien que les communications acoustiques sous-marines constituent l'application principale, les méthodes proposées ne dépendent pas fondamentalement de ce domaine. Leur généralité est notamment illustrée par l'application de la Search method à la télémétrie par impulsions de boue. Elles peuvent également être envisagées pour d'autres récepteurs dont les paramètres sont difficiles à régler, notamment dans les communications véhiculaires, les objets connectés et les systèmes de synchronisation.

Plusieurs perspectives peuvent être considérées. La fonction objectif pourrait intégrer un terme de complexité afin de pénaliser les configurations nécessitant des filtres très longs ou des calculs trop coûteux. Des méthodes de réduction de dimension ou de décomposition fonctionnelle pourraient permettre d'optimiser un nombre plus important d'hyperparamètres. WandS pourrait être étendue à plus de deux agents et utiliser des politiques nécessitant moins de paramètres de réglage.

La taille croissante de l'ensemble de données des processus gaussiens constitue également une limitation pour les communications longues. Des redémarrages périodiques, des fenêtres glissantes ou des critères explicites de suppression des observations devenues peu informatives pourraient maintenir une complexité bornée. Une campagne expérimentale dédiée en mer permettrait par ailleurs de valider WandS sur une communication réelle de longue durée.

Dans le cadre décentralisé, il serait pertinent de considérer des agents hétérogènes, des fonctions objectif différentes, des communications asynchrones, des messages retardés ou perdus et des contraintes de confidentialité. Enfin, les méthodes de suivi pourraient être étendues à l'optimisation conjointe de l'émetteur et du récepteur. Elles pourraient alors sélectionner simultanément un schéma de modulation et de codage, une configuration de réception et un compromis entre le débit, la fiabilité et la complexité à l'aide d'une

optimisation bayésienne multiobjectif.

BIBLIOGRAPHY

[1] Cullen, W. Douglas (1990), *The Public Inquiry into the Piper Alpha Disaster*, Cm 1310, HMSO, URL: https://www.hse.gov.uk/offshore/piper-alpha-disaster-public-inquiry.htm (visited on 03/07/2026).

[2] BP Deepwater Horizon (2011), *Deep Water: The Gulf Oil Disaster and the Future of Offshore Drilling*, U.S. Government Printing Office, URL: https://www.govinfo.gov/content/pkg/GPO-OILCOMMISSION/pdf/GPO-OILCOMMISSION.pdf (visited on 03/07/2026).

[3] Borthwick, David (June 2010), *Report of the Montara Commission of Inquiry*, Australian Government, URL: https://www.industry.gov.au/sites/default/files/2018-11/montara-commission-of-inquiry-report-june-2010.pdf (visited on 03/07/2026).

[4] Phillips, Brennan T. et al. (2021), « The Fiber Optic Reel System: A Compact Deployment Solution for Tethered Live-Telemetry Deep-Sea Robots and Sensors », *in*: *Sensors* 21.7, ISSN: 1424-8220, DOI: 10.3390/s21072526, URL: https://www.mdpi.com/1424-8220/21/7/2526.

[5] Nguyen, Dong Trong et al. (2025), « Enabling scalable inspection of offshore mooring systems using cost-effective autonomous underwater drones », *in*: *Frontiers in Robotics and AI* Volume 12 - 2025, ISSN: 2296-9144, DOI: 10.3389/frobt.2025.1655242, URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2025.1655242.

[6] Nauert, Franka and Peter Kampmann (2023), « Inspection and maintenance of industrial infrastructure with autonomous underwater robots », *in*: *Frontiers in Robotics and AI* Volume 10 - 2023, ISSN: 2296-9144, DOI: 10.3389/frobt.2023.1240276, URL: https://www.frontiersin.org/journals/robotics-and-ai/articles/10.3389/frobt.2023.1240276.

[7] Walree, Paul A. van (2011), « Channel sounding for acoustic communications: techniques and shallow-water examples, FFI-rapport 2011/00007 », *in*: URL: https://ffi-publikasjoner.archive.knowledgearc.net/bitstream/handle/20.500.12242/2456/11-00007.pdf.

[8] Pottier, Antony (Nov. 2018), « Méthodes décentralisées d'allocation des ressources dans le canal d'interférence acoustique sous-marin », Theses, Ecole nationale supérieure Mines-Télécom Atlantique, URL: https://theses.hal.science/tel-02095995.

[9] Jarrot, Arnaud et al. (2021), « High-speed underwater acoustic communication for multi-agent supervised autonomy », *in*: *2021 Fifth Underwater Communications and*

Networking Conference (UComms), pp. 1–4, DOI: 10.1109/UComms50339.2021.9598036.

[10] Shimura, Takuya et al. (2021), « High-rate underwater acoustic communication at over 600 kbps × km for vertical uplink data transmission on a full-depth lander system », *in*: *2021 Fifth Underwater Communications and Networking Conference (UComms)*, pp. 1–4, DOI: 10.1109/UComms50339.2021.9598067.

[11] Li, Zhengnan, Mandar Chitre, and Milica Stojanovic (2025), « Underwater acoustic communications », *in*: *Nature reviews electrical engineering* 2.*2*, pp. 83–95.

[12] Garnett, Roman (2023a), *Bayesian Optimization*, Cambridge University Press, DOI: 10.1017/9781108348973.

[13] Zuberi, Habib Hussain et al. (2023), « Deep-Neural-Network-Based Receiver Design for Downlink Non-Orthogonal Multiple-Access Underwater Acoustic Communication », *in*: *Journal of Marine Science and Engineering* 11.*11*, p. 2184, DOI: 10.3390/jmse11112184.

[14] Zhang, Y., F. Meriaudeau, et al. (2022a), « Deep learning aided OFDM receiver for underwater acoustic communications », *in*: *Applied Acoustics* 192, p. 108515, DOI: 10.1016/j.apacoust.2021.108515.

[15] Zhang, Y. et al. (2023), « Deep learning and expert knowledge based underwater acoustic OFDM receiver », *in*: *Physical Communication*, DOI: 10.1016/j.phycom.2023.102041.

[16] Riera-Palou, F., J.M. Noras, and D.G.M. Cruickshank (2000a), « Variable length equalizers for broadband mobile systems », *in*: *Vehicular Technology Conference Fall 2000. IEEE VTS Fall VTC2000. 52nd Vehicular Technology Conference (Cat. No.00CH37152)*, vol. 5, 2478–2485 vol.5, DOI: 10.1109/VETECF.2000.883307.

[17] Kar, Asutosh, Ravinder Nath, and Alaka Barik (2011), « A VLMS based pseudo-fractional optimum order estimation algorithm », *in*: *Proceedings of the 2011 International Conference on Communication, Computing & Security*, ICCCS '11, Rourkela, Odisha, India: Association for Computing Machinery, pp. 119–123, ISBN: 9781450304641, DOI: 10.1145/1947940.1947966, URL: https://doi.org/10.1145/1947940.1947966.

[18] Zhou, Jiaqiong et al. (2024), « A novel variable tap-length adaptive decision feedback equalization in underwater wireless communication », *in*: *Physical Communication* 66, p. 102424, ISSN: 1874-4907, DOI: https://doi.org/10.1016/j.phycom.2024.102424, URL: https://www.sciencedirect.com/science/article/pii/S1874490724001423.

[19] Liu, Changxin, Jun Tao, and Weimin Huang (2024), « A Recursive Gaussian Newton Based Variable Parameter Adaptive Receiver for Single-Carrier Underwater Acoustic Communications », *in*: *IEEE Transactions on Vehicular Technology* 73.*12*, pp. 19275–19286, DOI: 10.1109/TVT.2024.3448197.

[20] Mathews, V John and Zhenhua Xie (2002), « A stochastic gradient adaptive filter with gradient adaptive step size », *in*: *IEEE transactions on Signal Processing* 41.*6*, pp. 2075–2087.

[21] Jiang, Weihua and Roee Diamant (2022), « Sparse channel estimation for long range underwater acoustic communication », *in*: *2022 Sixth Underwater Communications and Networking Conference (UComms)*, pp. 1–4, DOI: 10.1109/UComms56954.2022.9905683.

[22] Song, HC et al. (2011), « Long-range acoustic communication in deep water using a towed array », *in*: *The Journal of the Acoustical Society of America* 129.*3*, EL71–EL75.

[23] Roux, Philippe, W.A. Kuperman, and Karim G. Sabra (2011), « Ocean acoustic noise and passive coherent array processing », en, *in*: *Comptes Rendus. Géoscience* 343.*8-9*, pp. 533–547, DOI: 10.1016/j.crte.2011.02.003.

[24] Theocharidis, Theocharis and Ergina Kavallieratou (2025), « Underwater Communication Technologies: A Review », *in*: *Telecommunication Systems* 88.*2*, p. 54, URL: https://doi.org/10.1007/s11235-025-01279-x.

[25] Bjørnø, Leif, Thomas Neighbors, and David Bradley (2017), *Applied underwater acoustics*, Elsevier.

[26] Worzel, John Lamar, Maurice Ewing, and Chaim Leib Pekeris (1948), *Propagation of Sound in the Ocean*, vol. 27, Geological Society of America.

[27] Bourré, Arnaud et al. (2013), « A robust OFDM modem for underwater acoustic communications », *in*: *2013 MTS/IEEE OCEANS - Bergen*, pp. 1–5, DOI: 10.1109/OCEANS-Bergen.2013.6608003.

[28] Socheleau, Francois-Xavier et al. (2012), « Information-theoretic analysis of underwater acoustic OFDM systems in highly dispersive channels », *in*: *Journal of Electrical and Computer Engineering* 2012.*1*, p. 716720.

[29] Bocus, Mohammud Junaid, Angela Doufexi, and Dimitris Agrafiotis (2020), « Performance of OFDM-based massive MIMO OTFS systems for underwater acoustic communication », *in*: *IET Communications* 14.*4*, pp. 588–593.

[30] Walree, Paul A. van and Roald Otnes (2013), « Ultrawideband Underwater Acoustic Communication Channels », *in*: *IEEE Journal of Oceanic Engineering* 38.*4*, pp. 678–688, DOI: 10.1109/JOE.2013.2253391.

[31] Thorp, William H. (1967), « Analytic description of the low-frequency attenuation coefficient », *in*: *The Journal of the Acoustical Society of America* 42.*1*, pp. 270–270, DOI: 10.1121/1.1910566.

[32] Cheikh Melainine, Cheikh M., François-Xavier Socheleau, and Arnaud Jarrot (2024), « Self-Configurable Receiver for Underwater Acoustic Communications Using Bayesian Optimization », *in*: *2024 Seventh Underwater Communications and Networking Conference (UComms)*, pp. 1–5, DOI: 10.1109/UComms64662.2024.10847986.

[33] Socheleau (Nov. 2019), « Traitement du signal pour l'acoustique sous-marine : des communications numériques à la surveillance passive », Accreditation to supervise research, Université de bretagne occidentale, URL: https://theses.hal.science/tel-02511194.

[34] Walree, Paul A. van et al. (2017), « The Watermark Benchmark for Underwater Acoustic Modulation Schemes », *in*: *IEEE Journal of Oceanic Engineering* 42.*4*, pp. 1007–1018, DOI: 10.1109/JOE.2017.2699078.
[35] Underwater Acoustic Channels Group (2026), *Underwater Acoustic Channels*, https://uwa-channels.github.io/, Documentation site for the Underwater Acoustic Channel Library, (visited on 03/07/2026).
[36] Badiey, M. et al. (2000), « Signal variability in shallow-water sound channels », *in*: *IEEE Journal of Oceanic Engineering* 25.*4*, pp. 492–500, DOI: 10.1109/48.895356.
[37] Walree, Paul A. van (2013), « Propagation and Scattering Effects in Underwater Acoustic Communication Channels », *in*: *IEEE Journal of Oceanic Engineering* 38.*4*, pp. 614–631, DOI: 10.1109/JOE.2013.2278913.
[38] Tomasi, Beatrice, James Preisig, and Michele Zorzi (2011), « On the predictability of underwater acoustic communications performance: The KAM11 data set as a case study », *in*: *Proceedings of the 6th International Workshop on Underwater Networks*, pp. 1–7.
[39] Worcester, Peter F et al. (1999), « A test of basin-scale acoustic thermometry using a large-aperture vertical array at 3250-km range in the eastern North Pacific Ocean », *in*: *The Journal of the Acoustical Society of America* 105.*6*, pp. 3185–3201.
[40] Haziq, Muhammad et al. (2025), « Modulation techniques for underwater acoustic communication: A comprehensive survey », *in*: *IEEE Access*.
[41] Stojanovic, Milica, Josko A Catipovic, and John G Proakis (2002), « Phase-coherent digital communications for underwater acoustic channels », *in*: *IEEE journal of oceanic engineering* 19.*1*, pp. 100–111.
[42] Zia, Muhammad Yousuf Irfan, Javier Poncela, and Pablo Otero (2021), « State-of-the-art underwater acoustic communication modems: Classifications, analyses and design challenges », *in*: *Wireless personal communications* 116.*2*, pp. 1325–1360.
[43] Pranitha, B and L Anjaneyulu (2020), « Analysis of underwater acoustic communication system using equalization technique for ISI reduction », *in*: *Procedia Computer Science* 167, pp. 1128–1138.
[44] Hsieh, Yung-Ting, Zhuoran Qi, and Dario Pompili (2023), « Full-duplex underwater acoustic communications via self-interference cancellation in space », *in*: *Journal of Communications and Networks* 25.*2*, pp. 167–181.
[45] Maia, Lussac P., Antonio Silva, and Sergio M. Jesus (2018), « Sustained underwater acoustic communications with environmental-based time-reversal », *in*: *The Journal of the Acoustical Society of America* 144.*4*, EL262–EL267, DOI: 10.1121/1.5058119.
[46] Unal, Deniz et al. (2024), « Design and performance evaluation of SEANet, a software-defined networking platform for the Internet of Underwater Things », *in*: *Computer Networks* 250, p. 110579, DOI: 10.1016/j.comnet.2024.110579.
[47] Johnson, Mark, Lee Freitag, and Milica Stojanovic (1997a), « Improved Doppler tracking and correction for underwater acoustic communications », *in*: *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, IEEE, pp. 575–578.

[48] Stojanovic, Milica, Josko Catipovic, and John G. Proakis (1993), « Adaptive multichannel combining and equalization for underwater acoustic communications », *in*: *The Journal of the Acoustical Society of America* 94.*3*, pp. 1621–1631, DOI: 10.1121/1.408135.

[49] Cannelli, Loris et al. (2013), « Adaptive turbo equalization for underwater acoustic communication », *in*: *2013 MTS/IEEE OCEANS - Bergen*, pp. 1–9, DOI: 10.1109/OCEANS-Bergen.2013.6608150.

[50] Eynard, Goulven and Christophe Laot (2008), « Extended linear phase detector characteristic of a software PLL », *in*: *2008 3rd International Symposium on Communications, Control and Signal Processing*, pp. 62–67, DOI: 10.1109/ISCCSP.2008.4537193.

[51] Oh, Dae Sun et al. (1996), « Convergence analysis of a PLL for a digital recording channel with an adaptive partial response equalizer », *in*: *Proceedings of GLOBECOM'96. 1996 IEEE Global Telecommunications Conference*, vol. 2, pp. 979–983, DOI: 10.1109/GLOCOM.1996.587577.

[52] Cuji, Diego A., Zhengnan Li, and Milica Stojanovic (2024a), « High-Rate Information Transmission for Underwater Channels: Path-Specific Beamforming and Doppler Compensation », *in*: *2024 Seventh Underwater Communications and Networking Conference (UComms)*, pp. 1–5, DOI: 10.1109/UComms64662.2024.10847973.

[53] Johnson, M., L. Freitag, and M. Stojanovic (1997b), « Improved Doppler tracking and correction for underwater acoustic communications », *in*: *1997 IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. 1, pp. 575–578, DOI: 10.1109/ICASSP.1997.599703.

[54] Pelekanakis, Konstantinos and Mandar Chitre (2010), « Comparison of sparse adaptive filters for underwater acoustic channel equalization/Estimation », *in*: *2010 IEEE International Conference on Communication Systems*, pp. 395–399, DOI: 10.1109/ICCS.2010.5686514.

[55] Duttweiler, D. L. (2000), « Proportionate normalized least-mean-squares adaptation in echo cancelers », *in*: *IEEE Transactions on Speech and Audio Processing* 8.*5*, pp. 508–518, DOI: 10.1109/89.861368.

[56] Riera-Palou, F., J. M. Noras, and D. G. M. Cruickshank (2000b), « Variable length equalizers for broadband mobile systems », *in*: *Vehicular Technology Conference Fall 2000*, vol. 5, pp. 2478–2485, DOI: 10.1109/VETECF.2000.883307.

[57] Kusuma, Julius et al. (2018), « Pragmatic Performance Optimization of a Multichannel DFE System for a Wideband 100 kbps 1-km Subsea Acoustic Modem », *in*: *2018 Fourth Underwater Communications and Networking Conference (UComms)*, pp. 1–4, DOI: 10.1109/UComms.2018.8493222.

[58] Bae, Ki Tack, Won Gi Jeon, and Yong Soo Cho (1997), « Convergence analyses of timing and carrier recovery loops for a digital communication channel with an adaptive equalizer », *in*: *Proceedings of 40th Midwest Symposium on Circuits and Systems*, vol. 2, pp. 1366–1369, DOI: 10.1109/MWSCAS.1997.662336.

[59] Ye, Zi, Tarak Arbi, and Benoit Geller (2021), « Adaptive Step-size Iterative Equalization for Underwater Communication », *in*: *OCEANS 2021: San Diego – Porto*, pp. 1–5, DOI: 10.23919/OCEANS44145.2021.9705715.

[60] Mahmutoglu, Yigit, Kadir Turk, and Emin Tugcu (2016), « Particle swarm optimization algorithm based decision feedback equalizer for underwater acoustic communication », *in*: *2016 39th International Conference on Telecommunications and Signal Processing (TSP)*, pp. 153–156, DOI: 10.1109/TSP.2016.7760848.

[61] Yang, Xiaoxia et al. (2013), « Soft decision feedback equalizer for channels with low SNR in underwater acoustic communications », *in*: *2013 MTS/IEEE OCEANS - Bergen*, pp. 1–5, DOI: 10.1109/OCEANS-Bergen.2013.6608050.

[62] Deguchi, Mitsuyasu, Yukihiro Kida, and Takuya Shimura (2023), « Low-dimensional approximation of recursive least squares algorithm applied to decision feedback equalizer on underwater acoustic communication », *in*: *OCEANS 2023 - MTS/IEEE U.S. Gulf Coast*, pp. 1–9, DOI: 10.23919/OCEANS52994.2023.10337095.

[63] Cuji, Diego A., Zhengnan Li, and Milica Stojanovic (2024b), « High-Rate Information Transmission for Underwater Channels: Path-Specific Beamforming and Doppler Compensation », *in*: *2024 Seventh Underwater Communications and Networking Conference (UComms)*, pp. 1–5, DOI: 10.1109/UComms64662.2024.10847973.

[64] Selmi, Mehrez et al. (2010), « Block versus adaptive MIMO equalization for coherent PolMux QAM transmission systems », *in*: *36th European Conference and Exhibition on Optical Communication*, pp. 1–3, DOI: 10.1109/ECOC.2010.5621462.

[65] Chen, Jiawei et al. (2024), « Adaptive Receiver Design for Long-Range Underwater Acoustic Communications », *in*: *OCEANS 2024 - Halifax*, pp. 1–6, DOI: 10.1109/OCEANS55160.2024.10753790.

[66] Socheleau, Francois-Xavier, Christophe Laot, and Jean-Michel Passerieux (2010), « Concise Derivation of Scattering Function from Channel Entropy Maximization », *in*: *IEEE Transactions on Communications* 58.*11*, pp. 3098–3103, DOI: 10.1109/TCOMM.2010.091310.090247.

[67] Wu, Y. et al. (2024), « Race to the bottom: Bayesian optimisation for chemical problems », *in*: *Digital Discovery*, DOI: 10.1039/D3DD00234A.

[68] Frazier, Peter I. (2015), « Bayesian optimization for materials design », *in*: *Information Science for Materials Discovery and Design*, Springer, pp. 45–75.

[69] Trabucco, Brandon et al. (2022), « Design-Bench: Benchmarks for Data-Driven Offline Model-Based Optimization », *in*: *Proceedings of the 39th International Conference on Machine Learning*, vol. 162, Proceedings of Machine Learning Research, PMLR, pp. 21658–21676, URL: https://proceedings.mlr.press/v162/trabucco22a.html.

[70] Kawabata, Takanori et al. (2025), « Black-box optimization in immunology and beyond: A practical guide to algorithms and future directions », *in*: *Allergology International* 74.*4*, pp. 549–562, DOI: 10.1016/j.alit.2025.08.006.

[71] Rosa, S. S., D. Nunes, et al. (2022), « Maximizing mRNA vaccine production with Bayesian optimization », *in*: *Biotechnology and Bioengineering*, DOI: 10.1002/bit.28216.

[72] Kim, Minsu et al. (2025), « Offline Model-Based Optimization: Comprehensive Review », *in*: *arXiv preprint arXiv:2503.17286*, URL: https://arxiv.org/abs/2503.17286.

[73] Franceschi, Luca et al. (2024), « Hyperparameter Optimization in Machine Learning », *in*: *arXiv preprint arXiv:2410.22854*, URL: https://arxiv.org/abs/2410.22854.

[74] Shahriari, Bobak et al. (2016), « Taking the Human Out of the Loop: A Review of Bayesian Optimization », *in*: *Proceedings of the IEEE* 104.*1*, pp. 148–175, DOI: 10.1109/JPROC.2015.2494218.

[75] Young, M. Todd et al. (2020), « Distributed Bayesian optimization of deep reinforcement learning algorithms », *in*: *Journal of Parallel and Distributed Computing* 139, pp. 43–52, DOI: 10.1016/j.jpdc.2019.07.008.

[76] Eimer, Theresa, Marius Lindauer, and Roberta Raileanu (2023), « Hyperparameters in Reinforcement Learning and How To Tune Them », *in*: *Proceedings of Machine Learning Research*, URL: https://proceedings.mlr.press/v202/eimer23a.html.

[77] Larson, Jeffrey, Matt Menickelly, and Stefan M. Wild (2019), « Derivative-free optimization methods », *in*: *Acta Numerica* 28, pp. 287–404, DOI: 10.1017/S0962492919000060.

[78] Nelder, J. A. and R. Mead (Jan. 1965), « A Simplex Method for Function Minimization », *in*: *The Computer Journal* 7.*4*, pp. 308–313, ISSN: 0010-4620, DOI: 10.1093/comjnl/7.4.308, eprint: https://academic.oup.com/comjnl/article-pdf/7/4/308/1013182/7-4-308.pdf, URL: https://doi.org/10.1093/comjnl/7.4.308.

[79] Kiefer, Jack and Jacob Wolfowitz (1952), « Stochastic estimation of the maximum of a regression function », *in*: *The Annals of Mathematical Statistics*, pp. 462–466.

[80] Michalewicz, Zbigniew (2013), *Genetic Algorithms + Data Structures = Evolution Programs*, Springer Science & Business Media, DOI: 10.1007/978-3-662-03315-9.

[81] Kennedy, J. and R. Eberhart (1995), « Particle swarm optimization », *in*: *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, pp. 1942–1948, DOI: 10.1109/ICNN.1995.488968.

[82] Dorigo, Marco, Mauro Birattari, and Thomas Stützle (2006), « Ant colony optimization », *in*: *IEEE Computational Intelligence Magazine* 1.*4*, pp. 28–39, DOI: 10.1109/MCI.2006.329691.

[83] He, C. et al. (2023), « A review of surrogate-assisted evolutionary algorithms for expensive optimization problems », *in*: *Expert Systems with Applications*, DOI: 10.1016/j.eswa.2022.119495.

[84] Katoch, Sourabh, Sumit Chauhan, and Vijay Chahar (2021), « A review on genetic algorithm: past, present, and future », *in*: *Multimedia Tools and Applications* 80, pp. 8091–8126, DOI: 10.1007/s11042-020-10139-6.

[85] Hernández-Lobato, José Miguel et al. (2016), « A General Framework for Constrained Bayesian Optimization using Information-based Search », *in*: *Journal of Machine Learning Research* 17.*160*, pp. 1–53, URL: http://jmlr.org/papers/v17/15-616.html.

[86] Bischl, Bernd et al. (2023), « Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges », *in*: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 13.*2*, e1484.

[87] Rasmussen, Carl Edward and Christopher K. I. Williams (2006), *Gaussian Processes for Machine Learning*, The MIT Press, ISBN: 9780262256834, DOI: 10.7551/mitpress/3206.001.0001.

[88] Hutter, Frank, Holger H. Hoos, and Kevin Leyton-Brown (2011), « Sequential Model-Based Optimization for General Algorithm Configuration », *in*: *Learning and Intelligent Optimization*, ed. by Carlos A. Coello Coello, vol. 6683, Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, pp. 507–523, DOI: 10.1007/978-3-642-25566-3_40.

[89] Eggensperger, Katharina et al. (2013), « Towards an empirical foundation for assessing bayesian optimization of hyperparameters », *in*: *NIPS workshop on Bayesian Optimization in Theory and Practice*, vol. 10, 3, pp. 1–5.

[90] Bergstra, James et al. (2011), « Algorithms for Hyper-Parameter Optimization », *in*: *Advances in Neural Information Processing Systems*, vol. 24, URL: https://proceedings.neurips.cc/paper_files/paper/2011/file/86e8f7ab32cfd12577bc2619bc635690-Paper.pdf.

[91] Watanabe, S. (2023), « Tree-structured Parzen estimator: Understanding its algorithm components and their roles for better empirical performance », *in*: *arXiv preprint arXiv:2304.11127*, URL: https://arxiv.org/abs/2304.11127.

[92] Snoek, Jasper et al. (2015), « Scalable Bayesian optimization using deep neural networks », *in*: *Proceedings of the 32nd International Conference on Machine Learning*, URL: https://proceedings.mlr.press/v37/snoek15.html.

[93] Li, Y. et al. (2023a), « A Study of Bayesian Neural Network Surrogates for Bayesian Optimization », *in*: *arXiv preprint arXiv:2305.20028*, URL: https://arxiv.org/abs/2305.20028.

[94] Srinivas, Niranjan et al. (2012), « Information-Theoretic Regret Bounds for Gaussian Process Optimization in the Bandit Setting », *in*: *IEEE Transactions on Information Theory* 58.*5*, pp. 3250–3265, DOI: 10.1109/TIT.2011.2182033.

[95] Garnett, Roman (2023b), « Computing Policies with Gaussian Processes », *in*: *Bayesian Optimization*, Cambridge: Cambridge University Press, pp. 157–200.

[96] Song, Jiaming et al. (2022), « A General Recipe for Likelihood-free Bayesian Optimization », *in*: *Proceedings of the 39th International Conference on Machine Learning*, vol. 162, Proceedings of Machine Learning Research, PMLR, pp. 20384–20404, URL: https://proceedings.mlr.press/v162/song22b.html.

[97] Socheleau, Francois-Xavier, Christophe Laot, and Jean-Michel Passerieux (2011), « Stochastic replay of non-WSSUS underwater acoustic communication channels recorded at sea », *in*: *IEEE Transactions on Signal Processing* 59.*10*, pp. 4838–4849.

[98] Porter, Michael B. (2011), « The BELLHOP manual and user's guide: Preliminary draft », *in*: *Heat, Light, and Sound Research, Inc., Tech. Rep.* 260, URL: http://oalib.hlsresearch.com/Rays/HLS-2010-1.pdf.

[99] Brossier, Jean-Marc, Pierre-Olivier Amblard, and B. Geller (Sept. 2002a), « Self-adaptive PLL for general QAM constellations », *in*: *EUSIPCO 2002 - 11th European Signal Processing Conference (EUSIPCO)*, Toulouse, France, URL: https://hal.science/hal-03764853.

[100] Brossier, JM, PO Amblard, and B Geller (2002b), « Self adaptative PLL for multi-carrier local loop transmission », *in*: *Proc. Eusipco*, pp. 631–634.

[101] Geller, B. et al. (1996), « Equalizer for video rate transmission in multipath underwater communications », *in*: *IEEE Journal of Oceanic Engineering* 21.*2*, pp. 150–155, DOI: 10.1109/48.486790.

[102] Gerzaguet, Robin et al. (2018), « On multiplicative update with forgetting factor adaptive step size for least mean-square algorithms », *in*: *2018 25th International Conference on Telecommunications (ICT)*, IEEE, pp. 588–592.

[103] Shu, Huaqiang, Eric Pierre Simon, and Laurent Ros (2015), « On the use of tracking loops for low-complexity multi-path channel estimation in OFDM systems », *in*: *Signal Processing* 117, pp. 174–187.

[104] El Husseini, Ali Houssam, Eric Pierre Simon, and Laurent Ros (2019a), « Second-order autoregressive model-based Kalman filter for the estimation of a slow fading channel described by the Clarke model: Optimal tuning and interpretation », *in*: *Digital Signal Processing* 90, pp. 125–141.

[105] Shu, Huaqiang, Laurent Ros, and Eric Pierre Simon (2014), « Third-order complex amplitudes tracking loop for slow flat fading channel online estimation », *in*: *IET Communications* 8.*3*, pp. 360–371.

[106] Amis, Karine, Nicolas Le Josse, and Christophe Laot (2007), « Efficient Frequency-Domain MMSE Turbo Equalization Derivation and Performance Comparison with the Time-Domain Counterpart », *in*: *2007 Third International Conference on Wireless and Mobile Communications (ICWMC'07)*, p. 65, DOI: 10.1109/ICWMC.2007.38.

[107] Unal, Deniz et al. (2023), « Field Experiments with Doppler Compensation in High-Frequency Underwater Acoustic Communication System », *in*: *ICC 2023 - IEEE International Conference on Communications*, pp. 2185–2190, DOI: 10.1109/ICC45041.2023.10279776.

[108] Cuji, Diego A., Zhengnan Li, and Milica Stojanovic (2024c), « High-Rate Information Transmission for Underwater Channels: Path-Specific Beamforming and Doppler Compensation », *in*: *2024 Seventh Underwater Communications and Networking Conference (UComms)*, pp. 1–5, DOI: 10.1109/UComms64662.2024.10847973.

[109] Imbert, Stéphane, Guillaume Ansel, and Christophe Laot (2019), « Frame Detection and Synchronization for a SIMO Receiver in presence of Doppler Shift for Underwater Acoustic Communications », *in*: *OCEANS 2019 - Marseille*, pp. 1–4, DOI: 10.1109/OCEANSE.2019.8867462.

[110] Jensen, Johan Ludwig William Valdemar (1906), « Sur les fonctions convexes et les inégalités entre les valeurs moyennes », *in*: *Acta mathematica* 30.*1*, pp. 175–193.

[111] Bogunovic, Ilija, Jonathan Scarlett, and Volkan Cevher (2016a), « Time-varying Gaussian process bandit optimization », *in*: *Artificial Intelligence and Statistics*, PMLR, pp. 314–323, URL: https://proceedings.mlr.press/v51/bogunovic16.pdf.

[112] Gardner, Jacob et al. (2018), « GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration », *in*: *Advances in Neural Information Processing Systems* 31, URL: https://proceedings.neurips.cc/paper/2018/file/27e8e17134dd7083b050476733207ea1-Paper.pdf.

[113] Pottier, Antony and Beatrice Tomasi (2021), « Online adaptive power allocation and channel state feedback strategies in underwater acoustic networks », *in*: *OCEANS 2021: San Diego – Porto*, pp. 1–7, DOI: 10.23919/OCEANS44145.2021.9706008.

[114] Zhao, Peng et al. (2020), « A simple approach for non-stationary linear bandits », *in*: *International Conference on Artificial Intelligence and Statistics*, PMLR, pp. 746–755, URL: https://proceedings.mlr.press/v108/zhao20a.html.

[115] Zhou, Xingyu and Ness Shroff (2021), « No-regret algorithms for time-varying Bayesian optimization », *in*: *2021 55th Annual Conference on Information Sciences and Systems (CISS)*, IEEE, pp. 1–6, URL: https://arxiv.org/abs/2102.06296.

[116] Bardou, Anthony, Patrick Thiran, and Giovanni Ranieri (2024a), « This Too Shall Pass: Removing Stale Observations in Dynamic Bayesian Optimization », *in*: *Advances in Neural Information Processing Systems*, NeurIPS 2024. Also available as arXiv:2405.14540, URL: https://papers.nips.cc/paper_files/paper/2024/file/4b32c2943a02331792877cc6b5205f49-Paper-Conference.pdf.

[117] Boyd, Stephen et al. (2006), « Randomized Gossip Algorithms », *in*: *IEEE Transactions on Information Theory* 52.*6*, pp. 2508–2530, DOI: 10.1109/TIT.2006.874516, URL: https://web.stanford.edu/~boyd/papers/gossip.html.

[118] Gielis, Jennifer, Ajay Shankar, and Amanda Prorok (2022), « A critical review of communications in multi-robot systems », *in*: *Current Robotics Reports* 3.*4*, pp. 213–225, DOI: 10.1007/s43154-022-00077-5.

[119] Halsted, Trevor et al. (2021), « A survey of distributed optimization methods for multi-robot systems », *in*: *arXiv preprint arXiv:2103.12840*, URL: https://arxiv.org/abs/2103.12840.

[120] Li, Ning et al. (2016), « A survey on underwater acoustic sensor network routing protocols », *in*: *Sensors* 16.*3*, p. 414, DOI: 10.3390/s16030414.

[121] Imbert, Steven et al. (2022), « Performance Optimization of Underwater Communication Links at Different Ranges for AIS Relay to AUV », *in*: *Applied Sciences* 12.*9*,

p. 4166, DOI: 10.3390/app12094166, URL: https://www.mdpi.com/2076-3417/12/9/4166.

[122] Anderson, J. et al. (2021), « Communication Planning for Cooperative Terrain-Based Navigation of Multiple AUVs », *in*: *Sensors* 21.*5*, p. 1675, DOI: 10.3390/s21051675, URL: https://www.mdpi.com/1424-8220/21/5/1675.

[123] Fattah, Salmah et al. (2020), « A survey on underwater wireless sensor networks: Requirements, taxonomy, recent advances, and open research challenges », *in*: *Sensors* 20.*18*, p. 5393, DOI: 10.3390/s20185393.

[124] Gussen, Camila M. G. et al. (2021), « Optimization of Acoustic Communication Links for a Swarm of AUVs: The COMET and NEMOSENS Examples », *in*: *Applied Sciences* 11.*17*, p. 8200, DOI: 10.3390/app11178200, URL: https://www.mdpi.com/2076-3417/11/17/8200.

[125] Bouneffouf, Djallel, Irina Rish, and Charu Aggarwal (2020), « Survey on Applications of Multi-Armed and Contextual Bandits », *in*: *2020 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1–8, DOI: 10.1109/CEC48606.2020.9185782.

[126] Bubeck, Sébastien and Nicolo Cesa-Bianchi (2012), « Regret analysis of stochastic and nonstochastic multi-armed bandit problems », *in*: *arXiv preprint arXiv:1204.5721*.

[127] Agrawal, Shipra and Navin Goyal (29 Apr–01 May 2013), « Further Optimal Regret Bounds for Thompson Sampling », *in*: *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, ed. by Carlos M. Carvalho and Pradeep Ravikumar, vol. 31, Proceedings of Machine Learning Research, Scottsdale, Arizona, USA: PMLR, pp. 99–107, URL: https://proceedings.mlr.press/v31/agrawal13a.html.

[128] Parker-Holder, Jack, Vu Nguyen, and Stephen J. Roberts (2020), « Provably Efficient Online Hyperparameter Optimization with Population-Based Bandits », *in*: *Advances in Neural Information Processing Systems (NeurIPS)*, URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/c7af0926b294e47e52e46cfebe173f20-Paper.pdf.

[129] Bogunovic, Ilija, Jonathan Scarlett, and Volkan Cevher (2016b), « Time-varying Gaussian process bandit optimization », *in*: *Artificial Intelligence and Statistics*, PMLR, pp. 314–323, URL: https://proceedings.mlr.press/v51/bogunovic16.html.

[130] Desautels, Thomas, Andreas Krause, and Joel W. Burdick (2014), « Parallelizing exploration-exploitation tradeoffs in Gaussian process bandit optimization », *in*: *Journal of Machine Learning Research* 15.*1*, pp. 3873–3923, URL: https://www.jmlr.org/papers/v15/desautels14a.html.

[131] Madden, Samuel (2004), *Intel Lab Data*, https://db.csail.mit.edu/labdata/labdata.html, Data collected from 54 sensors deployed in the Intel Berkeley Research Lab, February 28–April 5, 2004. Accessed: 2026-03-23.

[132] Wang, Ziyu et al. (2016), « Bayesian optimization in a billion dimensions via random embeddings », *in*: *Journal of Artificial Intelligence Research* 55, pp. 361–387, URL: https://jair.org/index.php/jair/article/view/11252.

[133] Bardou, Anthony, Patrick Thiran, and Thomas Begin (May 2024b), « Relaxing the Additivity Constraints in Decentralized No-Regret High-Dimensional Bayesian Optimization », *in*: *The Twelfth International Conference on Learning Representations*, Vienna, Austria, URL: https://hal.science/hal-04629193.

[134] Goldsmith, A.J. and S.-G. Chua (1998), « Adaptive coded modulation for fading channels », *in*: *IEEE Transactions on Communications* 46.*5*, pp. 595–602, DOI: 10.1109/26.668727.

[135] Cooper-Baldock, Zachary, Eirini Panteli, and Paulo Santos (2025), « A state-of-the-art survey and benchmarking of Adaptive Modulation and Coding for Underwater Acoustic Communications », *in*: *Authorea Preprints*.

[136] Zhang, Yuzhi et al. (2022b), « Deep Reinforcement Learning-Based Adaptive Modulation for Underwater Acoustic Communication with Outdated Channel State Information », *in*: *Remote Sensing* 14.*16*, ISSN: 2072-4292, DOI: 10.3390/rs14163947, URL: https://www.mdpi.com/2072-4292/14/16/3947.

[137] Daulton, Samuel et al. (Jan. 2022), « Multi-objective Bayesian optimization over high-dimensional search spaces », *in*: *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, ed. by James Cussens and Kun Zhang, vol. 180, Proceedings of Machine Learning Research, PMLR, pp. 507–517, URL: https://proceedings.mlr.press/v180/daulton22a.html.

[138] Paria, Biswajit, Kirthevasan Kandasamy, and Barnabás Póczos (22–25 Jul 2020), « A Flexible Framework for Multi-Objective Bayesian Optimization using Random Scalarizations », *in*: *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, ed. by Ryan P. Adams and Vibhav Gogate, vol. 115, Proceedings of Machine Learning Research, PMLR, pp. 766–776, URL: https://proceedings.mlr.press/v115/paria20a.html.

[139] Suzuki, Shinya et al. (13–18 Jul 2020), « Multi-objective Bayesian Optimization using Pareto-frontier Entropy », *in*: *Proceedings of the 37th International Conference on Machine Learning*, ed. by Hal Daumé III and Aarti Singh, vol. 119, Proceedings of Machine Learning Research, PMLR, pp. 9279–9288, URL: https://proceedings.mlr.press/v119/suzuki20a.html.

[140] El Husseini, Ali Houssam, Eric Pierre Simon, and Laurent Ros (2019b), « Second-order autoregressive model-based Kalman filter for the estimation of a slow fading channel described by the Clarke model: Optimal tuning and interpretation », *in*: *Digital Signal Processing* 90, pp. 125–141, ISSN: 1051-2004, DOI: https://doi.org/10.1016/j.dsp.2019.04.008, URL: https://www.sciencedirect.com/science/article/pii/S1051200419300466.

[141] Nasr, Imen et al. (2016), « Time synchronization in IoT networks: Case of a wireless body area network », *in*: *2016 International Symposium on Signal, Image, Video and Communications (ISIVC)*, pp. 297–301, DOI: 10.1109/ISIVC.2016.7894004.

[142] Imamura, Hideaki et al. (2020), « Time-varying Gaussian Process bandit optimization with non-constant evaluation time », *in*: *arXiv preprint arXiv:2003.04691*.

[143] Hariri, Negar, Bamshad Mobasher, and Robin Burke (2015), « Adapting to User Preference Changes in Interactive Recommendation. », *in*: *IJCAI*, vol. 15, pp. 4268–4274.

[144] Jarrot, Arnaud, Andriy Gelman, and Julius Kusuma (2018), « Wireless Digital Communication Technologies for Drilling: Communication in the Bitss Regime », *in*: *IEEE Signal Processing Magazine* 35.*2*, pp. 112–120, DOI: 10.1109/MSP.2017.2781288.

[145] *High-Speed Wired Drill Pipe and its Corresponding Manufacture & Test Technology Research* (Nov. 2021), vol. Abu Dhabi International Petroleum Exhibition Conference, Abu Dhabi International Petroleum Exhibition and Conference, D022S185R001, DOI: 10.2118/207435-MS, eprint: https://onepetro.org/SPEADIP/proceedings-pdf/21ADIP/21ADIP/D022S185R001/2538792/spe-207435-ms.pdf, URL: https://doi.org/10.2118/207435-MS.

[146] Li, Chenxi and Zhengguang Xu (2023b), « A review of communication technologies in mud pulse telemetry systems », *in*: *Electronics* 12.*18*, p. 3930.

[147] *Deep electrode: A game-changing technology for electromagnetic telemetry* (Aug. 2017), vol. SEG Technical Program Expanded Abstracts 2017, SEG Technical Program Expanded Abstracts, pp. 1059–1063, DOI: 10.1190/segam2017-17731164.1, eprint: https://pubs.geoscienceworld.org/segeab/proceedings-pdf/SEGEAB.36/1/1059/7591711/segam2017-17731164.1.pdf, URL: https://doi.org/10.1190/segam2017-17731164.1.

[148] Pelekanakis, Konstantinos et al. (2014), « Performance of channel coding and equalization for acoustic telemetry along drill strings », *in*: *2014 IEEE International Conference on Communication Systems*, pp. 610–614, DOI: 10.1109/ICCS.2014.7024875.

[149] Spinnler, R F and F A Stone (July 1978), *Mud pulse logging while drilling telemetry system: design, development, and demonstrations*, tech. rep., Teleco Oilfield Services, Inc., Middletown, CT (USA), DOI: 10.2172/6900992, URL: https://www.osti.gov/biblio/6900992.

[150] Chin, Wilson C, Xiaoying Zhuang, and Jamie A Chin (2024), « High Data Rate MWD Mud Pulse Telemetry–From Mysteries to Discovery », *in*: *Journal of Energy and Power Technology* 6.*4*, pp. 1–36.

COLLEGES SCIENCES
BRETAGNE POUR L'INGENIEUR
LOIRE ET LE NUMERIQUE

IMT Atlantique
Bretagne-Pays de la Loire
École Mines-Télécom

Titre : Autoconfiguration des récepteurs de télécommunication en environnements sévères par optimisation bayésienne

Mot clés : Communications acoustiques sous-marines, Réglage des hyperparamètres, Optimisation bayésienne

Résumé : Cette thèse s'intéresse à la configuration automatique de récepteurs de télécommunications évoluant dans des environnements contraints et non stationnaires, avec un accent particulier sur les communications acoustiques sous-marines. Dans ce type de systèmes, les performances du récepteur dépendent de nombreux paramètres de réglage, généralement ajustés manuellement par des experts à partir de règles empiriques et d'une connaissance partielle du milieu de propagation. Une telle approche est coûteuse, difficilement transposable d'un scénario à un autre, et peu adaptée à des contextes où les propriétés du canal, notamment les effets Doppler et le multi-trajet, varient continuellement. Ce travail propose de modéliser le réglage des hyperparamètres du récepteur comme un problème d'optimisation de boîte noire soumis à de fortes contraintes pratiques : chaque évaluation de performance est coûteuse, les capacités de calcul embarquées sont limitées, les informations a priori sur l'environnement sont réduites, et la fonction objectif évolue dans le temps. Dans ce cadre, l'optimisation bayésienne apparaît comme un outil pertinent. Une contribution centrale de cette thèse réside dans l'adaptation de l'optimisation bayésienne à des scénarios en ligne et non stationnaires. Une architecture d'optimisation en ligne reposant sur plusieurs agents coopérants est proposée afin d'ajuster les hyperparamètres du récepteur en temps réel et de suivre un optimum mobile. La méthodologie est aussi étendue à des architectures décentralisées. Enfin, sa généralisabilité est illustrée par une application à la télémétrie par impulsions de boue.

Title: Self-Configuration of Telecommunication Receivers in Adverse Environments with Bayesian Optimization

Keywords: Underwater Acoustic Communications, Hyperparameter tuning, Bayesian Optimization

Abstract: This PhD addresses the automatic configuration of telecommunication receivers operating in adverse and highly dynamic environments, with a primary focus on Underwater Acoustic Communications. In such systems, receiver performance depends on many configuration parameters, traditionally tuned by experts using heuristic rules and partial knowledge of propagation conditions. This manual process is costly, difficult to generalize across environments, and poorly suited to contexts where channel characteristics such as multipath propagation and Doppler effects evolve over time. The thesis formulates receiver hyperparameter tuning as a black-box optimization problem under strict practical constraints: evaluations are expensive, computational resources are limited, prior knowledge is scarce, and the objective function is time-varying. Bayesian Optimization is therefore a suitable methodology. A key contribution is the extension of Bayesian Optimization to online, time-varying scenarios. The thesis proposes an online optimization architecture based on multiple interacting agents that adapt receiver hyperparameters in real time. This approach tracks a moving optimum while respecting practical constraints. The methodology is also extended to decentralized architectures, such as networks of Autonomous Underwater Vehicles using gossip-based information sharing. Its generality is finally illustrated through an application to Mud-Pulse Telemetry.