LARGE LANGUAGE MODELS (LLMS) FOR WIRELESS NETWORK RESOURCE MANAGEMENT AND FORECASTING APPLICATIONS

MUHAMMAD UMAR KHAN

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

GRADUATE PROGRAM IN ELECTRICAL AND COMPUTER ENGINEERING
YORK UNIVERSITY
TORONTO, ONTARIO

May 2026

Abstract

Next-generation wireless networks require intelligent methods that can both optimize resource allocation and anticipate future network behavior under rapidly changing operating conditions. Conventional optimization and deep learning approaches have achieved strong performance in specific wireless tasks, but they often rely on expensive retraining, large task-specific datasets, and fixed model assumptions that limit adaptability across new environments. This thesis investigates whether large language models (LLMs) can serve as training-free, prompt-engineered reasoning agents for wireless systems, with the broader goal of developing a unified methodology that adapts to different wireless problems without gradient-based retraining.

The thesis first focuses on radio resource management (RRM), where constrained non-convex optimization problems arise in tasks such as beamforming, power control, and user association. To address these problems, this thesis proposes *LLM4RRM* (LLM for Radio Resource Management), a block-coordinate-descent-inspired multi-agent LLM framework in which specialized solver agents optimize subsets of decision variables and an evaluator agent iteratively refines their outputs through structured feedback. To improve feasibility, an LLM-guided adaptive differentiable projection mechanism is introduced, allowing the framework to handle coupled constraints while reducing manual hyperparameter tuning. A prompt-based uncertainty quantification

strategy is also developed to assess the trustworthiness and consistency of LLM-generated solutions.

The thesis then extends the same training-free agentic design philosophy to multi-variate time-series forecasting *LLM4TSF* (LLM for Time-Series Forecasting). In this setting, the forecasting problem is decomposed across specialized agents that model trend and fluctuation components of correlated temporal signals, while an evaluator agent enforces statistical consistency across multiple series. This demonstrates that prompt-engineered LLM systems are not limited to optimization tasks, but can also be adapted to structured time-series forecasting problems in wireless networks.

Overall, the results of this thesis show that multi-agent LLM frameworks can provide feasible, adaptable, and scalable wireless intelligence without task-specific retraining. The thesis thereby establishes a foundation for using prompt engineering, decomposition, evaluator-guided refinement, and uncertainty-aware reasoning as general design principles for future wireless optimization and forecasting systems.

Acknowledgments

First and foremost, all praise and gratitude belongs to Allah, the Most Gracious and Most Merciful. It is by His will, His guidance, and His provision alone that this work came to be. Every ability I have, every opportunity that found its way to me, and every person who supported me along this journey, all of it is from Him. Alhamdulillah.

I would like to express my deepest gratitude to my supervisor, Prof. Hina Tabassum, for welcoming me into the NGWN Research Lab and for everything that followed. Her mentorship went far beyond academic guidance, she created an environment where I felt genuinely supported, challenged, and encouraged to grow. Her expertise, her warmth, and her constant effort to open doors and create opportunities for me have shaped not just this thesis, but the researcher I am becoming. I am truly fortunate to have had her in my corner.

I would also like to thank my thesis committee members, Prof. Ping Wang and Prof. Divya Sharma, for their valuable time, thoughtful feedback, and constructive engagement with my work.

A heartfelt thank you goes to Dr. Omer Waqar, whose collaboration as a co-author was instrumental in helping me develop as a writer and researcher. His patience and insight throughout the paper writing process taught me lessons I will carry forward

in everything I write. I also want to acknowledge Dr. Saleem Shahid, Dr. Bahman Alyaei, and Dr. Ashiq Hussain, my professors from my undergraduate years, who laid the foundations of my understanding in telecommunications and whose teaching continues to echo in the work I do today.

To all the current and alumni members of the NGWN Research Lab, thank you for the conversations, the shared effort, and the sense of community that made the lab feel like home. And to Ehtisham ul Haq, my companion through the Master's journey, your friendship made every difficult stretch easier to navigate. To my bachelor group of friends, you know who you are, and you know what you mean to me. The bond we built across those years has been one of the most grounding things I have carried into this chapter of my life.

None of this would have been possible without my parents, Muhammad Waseem Khan and Raheema Waseem. From the very beginning, they believed in me without condition, pushed me to pursue my dreams rather than settle for comfort, and gave me the courage to leave home in search of something greater. Every page of this thesis carries a piece of their sacrifice, their prayers, and their love. I hope this is worthy of what they gave. I would also like to extend my sincere gratitude to my uncle, Rehan Aslam, who stood by me throughout my career and whose support and encouragement meant more to me than he may know. Having someone in your corner who believes in your path, not just your results, is a rare and precious thing, and I am grateful to have had that in him.

Whatever I have achieved, and whatever is still to come, is by the grace of Allah, and then because of the people He placed in my path. I am grateful beyond words.

Table of Contents

List of Tables

List of Figures

List of Acronyms

3GPP	3rd Generation Partnership Project
AI	Artificial Intelligence
AoI	Age of Information
API	Application Programming Interface
ARIMA	Autoregressive Integrated Moving Average
AWGN	Additive White Gaussian Noise
BW	Bandwidth
BCD	Block-Coordinate Descent
BS	Base Station
CDF	Cumulative Distribution Function
CNN	Convolutional Neural Network
CoT	Chain-of-Thought
CSI	Channel State Information
CVXPY	Convex Optimization in Python
DDPG	Deep Deterministic Policy Gradient
DEPNet	Deep Explicit Projection Network
DEUNet	Deep Explicit User Association Network
DNN	Deep Neural Network
DRL	Deep Reinforcement Learning
DUL	Deep Unsupervised Learning
DUQ	Pairwise L2-Norm Based Distance Uncertainty Quantification

EMF	Electromagnetic Field
GA	Genetic Algorithm
GPT	Generative Pre-trained Transformer
GSM	Global System for Mobile Communications
ICL	In-Context Learning
ISAC	Integrated Sensing and Communication
JCAS	Joint Communication and Sensing
JSON	JavaScript Object Notation
JUPNet	Joint User Association and Power Control using Neural Networks
LLM	Large Language Model
LLM4RRM	LLM Framework for Radio Resource Management
LLM4TSF	LLM Framework for Time-Series Forecasting
LSTM	Long Short-Term Memory
LoRA	Low-Rank Adaptation
LTE	Long-Term Evolution
MAD	Mean Absolute Deviation
MADUQ	Mean Absolute Deviation Uncertainty Quantification
MAPE	Mean Absolute Percentage Error
MEC	Mobile Edge Computing
mmWave	Millimeter Wave
MIMO	Multiple-Input Multiple-Output
MINLP	Mixed Integer Nonlinear Programming
MISO	Multiple-Input Single-Output
ML	Machine Learning

MoE	Mixture of Experts
MSE	Mean Square Error
ND	Normalized Deviation
NG-RAN	Next Generation Radio Access Network
NGWN	Next-Generation Wireless Networks
NF	Naïve Forecasting
NRMSE	Normalized Root-Mean-Square Error
QoS	Quality of Service
RAG	Retrieval-Augmented Generation
RAN	Radio Access Network
RIC	RAN Intelligent Controller
RIS	Reconfigurable Intelligent Surfaces
RL	Reinforcement Learning
RLHF	Reinforcement Learning from Human Feedback
RNN	Recurrent Neural Network
RRM	Radio Resource Management
SINR	Signal-to-Interference-plus-Noise Ratio
STAR-RIS	Simultaneously Transmitting and Reflecting RIS
TSF	Time-Series Forecasting
TSFM	Time-Series Foundation Model
UA-PC	User-Association and Power-Control
UAV	Uncrewed Aerial Vehicle
UQ	Uncertainty Quantification
WMMSE	Weighted Minimum Mean Square Error

Chapter 1

Introduction

Radio Resource Management (RRM), i.e., power allocation, beamforming, user association, and channel assignment, remains a core pillar of wireless system design. As cellular networks evolve through 5G-Advanced toward 6G, RRM capabilities continue to mature within the 3rd Generation Partnership Project (3GPP) standardization ecosystem. 3GPP Release 19 approaches functional freeze in late 2025 [1], while 3GPP Release 20, initiated in mid-2025 [2], addresses both advanced 5G enhancements and foundational 6G studies. At the Next Generation Radio Access Network (NG-RAN) architecture level [3], 3GPP increasingly leverages Artificial Intelligence (AI) and Machine Learning (ML) within the RAN Intelligent Controller (RIC) [4] for network slicing, resource scheduling, coverage optimization, mobility enhancements, and energy-aware resource allocation.

At the same time, wireless intelligence is not limited to instantaneous decision-making alone. Another important capability is time-series forecasting (TSF) [5], where future system behavior is predicted from historical observations. Forecasting plays a central role in many engineering domains because it enables proactive planning

rather than purely reactive control. In wireless systems, forecasting can support anticipatory resource allocation, traffic-aware management, network monitoring, anomaly detection, and reliability enhancement by providing estimates of future states before they are observed [6, 7].

1.1 Radio Resource Management (RRM) in 6G

The first normative 6G specification is expected in Release 21 by approximately 2029, establishing the architectural foundation for next-generation RRM mechanisms. The transition to 6G represents a fundamental reconfiguration of the wireless landscape. 6G networks are expected to deliver peak data rates approaching 1 Tbps [8], sub-millisecond end-to-end latency, centimeter-level localization accuracy, and connectivity densities an order of magnitude higher than current deployments. Achieving these targets necessitates unprecedented network densification, adoption of higher frequency bands, and integration of ultra-massive Multiple-Input Multiple-Output (MIMO) arrays with hundreds to thousands of antenna elements. The adoption of these emerging technologies, including higher transmission frequencies, massive antenna arrays, and reconfigurable intelligent surfaces (RIS) [9], pushes wireless channels toward significantly shorter coherence times than previous generations [10], resulting in highly dynamic channel conditions.

Unlike propagation at sub-6 GHz bands, higher frequencies exhibit higher path loss, high blockages, and rapid channel decorrelation. Consequently, core RRM tasks, beamforming, power control, interference management, and channel allocation, become far more dynamic, high-dimensional, and interdependent than in prior generations [11]. Emerging 6G architectures exacerbate this complexity: ultra-dense cell-free

massive MIMO systems [12] face challenges including scalable baseband processing, fronthaul constraints, and user-centric scheduling across distributed access points.

Reconfigurable intelligent surfaces [13] introduce additional optimization layers for phase control and cascaded Base Station–RIS–User channel estimation. Moreover, joint communication and sensing (JCAS) integration [14, 15] couples sensing accuracy with communication resource allocation, introducing dependency between radar performance and data transmission.

1.2 Challenges of RRM in 6G Networks

RRM in 6G networks is fundamentally shaped by extreme channel dynamics and architectural complexity. Operating in higher frequency bands leads to rapid channel decorrelation, where coherence time becomes exceedingly small due to high carrier frequencies, narrow beams, and mobility sensitivity [16]. Channel state information (CSI) acquired at one instant becomes outdated within milliseconds, creating a pronounced Age of Information (AoI) problem [17] where decisions based on stale CSI can degrade performance, violate Quality of Service (QoS) guarantees, or produce infeasible resource allocations.

The challenge of **outdated CSI** intensifies in cell-free massive MIMO and RIS-assisted networks [18, 19], where channels comprise cascaded structures involving hundreds of antennas or reflecting elements. At Millimeter Wave (mmWave) and Terahertz (THz) frequencies, blockage events, user mobility, and beam-squint effects further compress effective CSI lifetime [20], demanding resource allocation methods capable of producing reliable solutions with imperfect or partially outdated channel knowledge.

Scalability emerges as another critical challenge. Ultra-massive MIMO arrays, RIS panels with thousands of reconfigurable elements [21], and dense access point deployments create extremely high-dimensional decision spaces. Beamforming optimization must jointly determine transmit directions and power levels across hundreds of antenna elements, while RIS configuration requires setting phase shifts for thousands of passive reflectors. The resulting optimization problems exhibit complexity that grows rapidly with network size, often involving non-convex objectives and coupled constraints across multiple decision variables.

Traditional optimization approaches exhibit a **memoryless nature**, treating each resource allocation instance independently and restarting computation from scratch whenever CSI updates or system parameters change. This prevents exploitation of valuable information from previous optimization rounds, such as structural patterns in optimal solutions or near-optimal operating points that could serve as warm starts. In highly dynamic 6G environments where optimization must be performed in every coherence interval, this inability to leverage historical context leads to substantial computational redundancy.

Constraint satisfaction presents another major challenge for non-convex problems with coupled constraints. Many 6G scenarios involve multiple interdependent constraints: per-user QoS requirements, per-BS power budgets, per-RIS element phase limitations, and interference thresholds. Ensuring that all constraints are simultaneously satisfied becomes increasingly difficult as problem dimensionality grows and constraints become tightly coupled. Traditional penalty-based or barrier methods cannot guarantee feasibility at convergence.

Beyond these core challenges, hardware impairments [22], dynamic interference

patterns from narrow-beam transmissions, joint communication and sensing constraints [14], and heterogeneous service demands [23] further compound the difficulty of 6G resource allocation. Collectively, these challenges, rapid CSI aging, high-dimensional scalability, memoryless optimization, and complex constraint satisfaction, highlight fundamental limitations of conventional resource allocation approaches and motivate the need for new computational frameworks.

1.3 Deep Learning and its Challenges for RRM

Deep learning has emerged as a prominent approach to address the challenges outlined in Section 1.2, demonstrating success across wireless applications including channel estimation, localization, and end-to-end system design. For resource allocation, research has focused on sum-rate maximization through power control and user association using supervised, unsupervised, and reinforcement learning paradigms.

Supervised learning approaches train Deep Neural Networks (DNNs) to map channel state information to resource allocation decisions using labeled data from conventional solvers such as Weighted Minimum Mean Square Error (WMMSE) [24, 25], achieving performance comparable to iterative algorithms with orders-of-magnitude faster inference. Unsupervised methods eliminate labeled data dependency by directly optimizing system objectives [26,27], with graph neural networks demonstrating scalability for interference management [28]. Reinforcement learning enables adaptive policy learning through environmental interaction, proving effective in dynamic scenarios with outdated CSI [29, 30]. Notably, differentiable projection-based frameworks [31, 32] guarantee strict QoS feasibility through convex layers and gradient-based corrections, achieving zero constraint violation probability while maintaining

strong sum-rate performance.

Despite these achievements, deep learning exhibits critical limitations. The memoryless limitation persists: while DNNs learn from historical training data, each trained model remains frozen and cannot adapt to new network configurations without retraining. When system parameters change, QoS requirements, antenna configurations, or network sizes, models require complete retraining, even for projection-based methods. Training overhead is severe: supervised learning requires thousands of labeled examples from expensive optimization solvers, unsupervised methods demand hundreds of epochs with extensive hyperparameter tuning, and reinforcement learning necessitates long training episodes. The wireless domain lacks standardized datasets [33], requiring costly generation of realistic training data.

Trained models exhibit poor generalization across domain shifts in channel statistics, user density, or network topology, and suffer from catastrophic forgetting during adaptation. While projection-based methods have advanced constraint satisfaction, most approaches cannot guarantee feasibility at inference, relying on penalty terms or Lagrangian formulations with non-zero duality gaps [34]. Collectively, these limitations, i.e. inability to adapt without retraining, extensive training requirements, dataset scarcity, poor generalization, and catastrophic forgetting, reveal that deep learning, while addressing some 6G challenges, introduces new bottlenecks limiting practical deployment in highly dynamic next-generation wireless systems.

1.4 Time Series in Wireless Networks

Alongside resource allocation, forecasting is emerging as another important capability for intelligent wireless systems. Time-series forecasting in wireless networks aims to

predict the future evolution of relevant signals, measurements, or operating conditions from historical data. Examples include traffic load prediction, channel-quality evolution, mobility patterns, interference fluctuations, and energy-demand estimation. Such predictive capability is valuable because it allows the network to anticipate future conditions and support more informed planning, monitoring, and control. [35]

In practice, wireless forecasting problems are often multivariate and heterogeneous, since multiple signals, technologies, or measurements evolve jointly over time. Practical forecasting may therefore require predicting several correlated series simultaneously, while preserving both temporal structure within each series and statistical dependence across series [36]. This creates a need for forecasting frameworks that can handle multi-series interactions, changing operating conditions, and limited task-specific supervision in a unified manner.

1.5 Challenges of Time-Series Forecasting

Time-series forecasting is fundamentally challenging because real-world sequences often exhibit non-stationarity, multi-scale temporal structure, noise, missing observations, and sudden regime changes. Long-range dependencies may coexist with short-term fluctuations, making it difficult for a single model to simultaneously capture coarse trends, periodic behavior, and local variability [37]. In addition, forecasting performance often degrades as the prediction horizon increases, since small errors can accumulate over time [38]. These challenges are further complicated by the fact that many forecasting models must generalize across environments whose statistical properties shift over time, making robust adaptation without retraining a persistent difficulty.

These challenges become even more pronounced in wireless time-series forecasting, where the signals of interest are shaped by user behavior, traffic demand, mobility, interference, scheduling decisions, and propagation conditions [39]. In many cases, the problem is not only temporal but also multivariate, with correlated series evolving across frequencies, technologies, cells, or monitored variables. Daily cycles may interact with short-term fluctuations, and the signal statistics may change across sites, deployments, or environmental conditions. As a result, accurate wireless forecasting requires models that can capture both intra-series temporal patterns and inter-series dependence while remaining robust to noise, distribution shifts, and limited domain-specific data [40]. This motivates the need for adaptable forecasting frameworks that can leverage structure and feedback without depending on extensive task-specific retraining.

1.6 Why Large Language Models (LLMs) for RRM and TSF?

The emergence of Large Language Models (LLMs) [41, 42] represents a potential paradigm shift in addressing the resource allocation challenges identified in the preceding sections. By interpreting high-level problem descriptions expressed in natural language, LLMs demonstrate the potential of generating near-optimal solutions without explicit mathematical formulations or task-specific training datasets [43]. Recent research has begun exploring LLM integration into wireless optimization frameworks, revealing opportunities that directly address fundamental limitations of both classical and deep learning approaches.

A defining characteristic of LLMs is their ability to perform zero-shot and few-shot in-context learning without task-specific training. Unlike DNNs requiring thousands

of labelled examples and extensive offline training, LLMs can interpret optimization objectives and constraints expressed in natural language, reason about solution strategies through in-context learning, and generate candidate allocations using only task descriptions and minimal examples [43]. For wireless resource allocation, this enables tackling new network topologies, updated QoS requirements, or modified system parameters by simply reformulating the problem description, eliminating the retraining burden and enabling rapid adaptation to dynamic 6G conditions.

LLMs offer inherent interpretability through natural language reasoning. While traditional solvers produce numerical outputs without revealing decision logic and DNNs operate as black boxes, LLMs can generate step-by-step reasoning traces using chain-of-thought techniques, explicitly articulating how they decompose problems and evaluate trade-offs. This transparency is valuable in safety-critical applications where operators must validate constraint satisfaction and QoS guarantees, facilitating human-in-the-loop oversight without modifying architectures or retraining parameters. The generalization properties of LLMs address critical gaps in existing methods. Conventional algorithms struggle with high-dimensional decision spaces, while deep learning models trained for fixed network sizes fail to generalize when configurations change. LLMs process resource allocation through flexible, token-based representations that accommodate variable-length inputs. A single LLM can handle networks with varying numbers of users, access points, and frequency bands without fine-tuning, functioning as a general-purpose optimizer. Moreover, LLMs exhibit cross-domain adaptability; the same model can be repurposed for beamforming, power control, or user association by modifying only the task description.

Beyond optimization, these same properties make LLMs appealing for time-series

forecasting [44]. Because forecasting can be framed through natural-language descriptions [45], historical examples, and structured statistical summaries, LLMs can use in-context learning to reason about temporal patterns without task-specific retraining [46]. Their ability to incorporate retrieved examples, decompose forecasting into interpretable sub-tasks such as trend and fluctuation estimation, and iteratively refine outputs through evaluator feedback makes them especially attractive for multivariate wireless forecasting problems where data distributions and operating conditions may shift over time. In this sense, LLMs offer a common reasoning framework that can support both control-oriented tasks such as RRM and predictive tasks such as wireless time-series forecasting.

These opportunities position LLMs as a fundamentally new approach that combines optimization flexibility, forecasting adaptability, and interpretability while circumventing training overhead, constraint satisfaction challenges, and generalization limitations of existing paradigms. However, as discussed in the following section, LLMs also introduce their own limitations that must be carefully addressed for reliable deployment in wireless systems.

1.7 Limitations and Research Gaps

While LLMs demonstrate promising capabilities across various domains, their application to wireless resource allocation faces fundamental challenges. Understanding these limitations is essential for developing reliable frameworks that can meet the stringent requirements of next-generation wireless systems. LLMs exhibit several critical limitations when applied to optimization problems:

- *Hallucinations and constraint violations*: LLMs are trained to produce probable

text sequences rather than rigorously correct solutions, leading to outputs that may appear plausible but violate physical constraints or feasibility requirements.

- *Computational latency*: Autoregressive token generation introduces sequential delays (tens to hundreds of milliseconds) that may be incompatible with real-time wireless control requirements.

- *Output variability*: Stochastic decoding produces different solutions for identical inputs, making it difficult to ensure consistent performance across repeated executions.

- *Numerical precision limitations*: Token-based representation of continuous values inherently limits arithmetic precision, potentially affecting the quality of numerical optimization solutions.

Beyond these fundamental limitations, several critical research gaps exist in applying LLMs to wireless resource management. Current explorations remain largely proof-of-concept demonstrations on small-scale problems, lacking the rigour and comprehensiveness required for practical deployment. Existing approaches do not provide mechanisms to guarantee feasibility or quantify solution quality in terms of standard optimization metrics. Moreover, the inherent uncertainty in LLM outputs, stemming from stochastic decoding and sensitivity to prompt design, has not been systematically characterized or addressed. For complex wireless scenarios involving coupled decision variables, multiple non-convex constraints, and heterogeneous resource types, no existing framework demonstrates how LLMs can reliably produce high-quality solutions while ensuring all constraints are satisfied.

Similar gaps also arise in forecasting applications. Although LLMs show promise

as training-free forecasters, current approaches provide limited treatment of multivariate dependence, statistical consistency, horizon-wise reliability, and robustness under distribution shifts. In wireless forecasting problems, these issues become especially important because the forecaster must preserve both temporal structure and meaningful cross-series behavior while adapting to new monitoring environments. Consequently, there remains a need for LLM-based frameworks that move beyond one-shot prompting and instead incorporate decomposition, evaluator-guided refinement, structured statistical context, and reliability assessment for both optimization and forecasting tasks.

These limitations and gaps motivate the development of hybrid and agentic architectures that combine the adaptability and reasoning capabilities of LLMs with principled mechanisms for feasibility enforcement, statistical grounding, and uncertainty quantification.

1.8 Scope of the Thesis

Given the challenges of 6G resource allocation, the growing need for adaptive wireless forecasting, the limitations of classical and deep learning approaches, and the emerging opportunities offered by LLMs, this thesis develops a prompt-engineered, agentic framework for trustworthy and adaptive wireless intelligence. The primary focus is on constraint-aware wireless resource management, while a complementary forecasting study investigates how the same training-free design philosophy can be extended to multivariate time-series forecasting in wireless systems. Together, these two directions address a common set of gaps identified in previous sections: the inability to adapt without retraining, poor generalization across changing deployment

conditions, limited interpretability, and the need for trustworthy performance under practical wireless complexity.

We adopt a prompt-based optimization approach that leverages LLMs' ability to interpret optimization problems through natural language and perform in-context learning without task-specific training. This approach enables true zero-shot adaptability through prompt modifications alone, naturally supports iterative refinement through structured feedback, and provides interpretability through natural language reasoning traces. The proposed LLM4RRM framework combines multi-agent reasoning with differentiable projection to ensure rigorous constraint satisfaction. Our key contributions can thus be given as follows:

- **Multi-Agent LLM-based RRM Optimization:** We propose LLM4RRM, a novel training-free, block-coordinate descent (BCD)-inspired multi-agent LLM framework for constrained non-convex RRM optimization problems, in which the decision variables are decomposed across specialized solver agents and iteratively coordinated through structured feedback provided by an evaluator agent.

- **LLM-Guided Adaptive Projection for Constraint Satisfaction:** We introduce an LLM-guided adaptive differentiable projection mechanism, in which a dedicated hyperparameter-tuning agent dynamically adjusts projection parameters, such as step size, to enforce feasibility. This approach eliminates manual hyperparameter tuning and significantly accelerates convergence toward the feasible region.

- **Prompt-based Uncertainty Quantification:** We present the first prompt-based uncertainty quantification (UQ) framework for LLM-driven RRM optimization, characterizing both epistemic uncertainty (arising from limitations in

the LLM's pre-trained knowledge) and aleatoric uncertainty (resulting from inherent variability in the input data). We introduce two novel metrics, pairwise L2-norm based distance ($\mathcal{D}_{\mathrm{UQ}}$) and mean absolute deviation ($\mathrm{MAD}_{\mathrm{UQ}}$), that quantify output dispersion and consistency. The framework leverages input perturbations, prompt variations, and output sampling to provide measurable indicators of solution stability and reliability.

- **Training-Free Multi-Agent Time-Series Forecasting:** We extend the same prompt-engineered agentic paradigm to multivariate time-series forecasting in wireless systems. The resulting forecasting framework decomposes the prediction task into specialized agents for different temporal components and uses evaluator-guided refinement to improve forecast consistency across multiple correlated series, demonstrating that LLM-based wireless intelligence can be applied beyond optimization to structured time-series prediction.

- **Cross-Task Adaptability in Wireless Systems:** We validate the proposed methodology across both optimization and forecasting settings. For RRM, the framework is studied through two case studies classified as continuous non-convex optimization and mixed integer nonlinear programming (MINLP), respectively. For forecasting, the framework is applied to multivariate wireless time-series prediction. Together, these studies demonstrate that prompt-engineered multi-agent LLMs can adapt across distinct wireless problem classes without retraining.

- **Comprehensive Performance Evaluation:** We establish a rigorous evaluation framework assessing LLM-based optimization across network sum-rate,

QoS violation probability, convergence speed, and computational complexity. Numerical results demonstrate that LLM4RRM consistently achieves zero post-projection constraint violations, while baselines show 30-70% violations. The framework scales to varying network sizes without retraining, achieving up to 5x higher sum-rates and converging 2-4× faster compared to DNN approaches. Comparative evaluation across five LLM backbones reveals that GPT-5 Mini (Generative Pre-trained Transformer) achieves 5-7% faster convergence and 1-3% higher sum-rates than the baselines.

Beyond addressing technical challenges, this thesis establishes a methodological foundation for integrating large language models into wireless systems more broadly. The demonstration that pre-trained general-purpose models can be effectively adapted to constrained optimization through prompt engineering and projection mechanisms opens new research directions, while the forecasting study further shows that the same agentic design principles can be transferred to multivariate time-series prediction without gradient-based retraining. In this sense, the underlying ideas of multi-agent decomposition, iterative evaluator feedback, retrieval-enhanced prompting, adaptive projection, and uncertainty-aware reasoning extend naturally to a wider class of wireless tasks, including spectrum management, network slicing, mobility-aware resource allocation, and wireless time-series forecasting.

1.9 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 reviews LLM fundamentals and related work in wireless optimization. Chapter 3 presents the LLM4RRM framework design and validates it through two use-case studies: joint beamforming

and phase shift optimization in Simultaneously Transmitting and Reflecting RIS or STAR-RIS networks, and joint user association and power control in multi-BS networks. Chapter 4 presents a LLM framework for multivariate time series forecasting. Chapter 5 outlines future research directions, including mobility-aware optimization and time-series forecasting for wireless networks.

1.10 Research Outcomes

The research conducted in this thesis has led to two main paper-level outcomes corresponding to its two core adaptations: wireless resource optimization and time-series forecasting:

- LLM4RRM: A Multi-Agent Feedback-Driven LLM Framework for Constrained Resource Allocation in Wireless Networks (Submitted to IEEE Journal Transactions in Communications)
- LLM4TSF: A Multi-Agent LLM Framework for Multivariate Time-Series Forecasting (Will be Submitted to IEEE Wireless Communications Letter)

Chapter 2

Preliminaries and Literature Review

This section establishes the theoretical and empirical foundations for applying Large Language Models to wireless resource allocation. We first review the core architectural principles, training paradigms, and inference mechanisms of LLMs. We then survey recent research on LLM-based approaches for wireless resource management, examining both prompt-based optimization methods and code generation frameworks, and identify critical gaps that motivate the proposed LLM4RRM framework.

2.1 Introduction to LLMs

Large Language Models (LLMs) provide a new computational paradigm for optimization in 6G networks. Unlike task-specific deep neural networks (DNNs) that must be retrained for each topology and constraint set, LLMs are pre-trained, general-purpose sequence models that can be adapted to new problems purely through prompting. This subsection reviews the core principles of LLMs with emphasis on aspects most relevant to wireless resource allocation: (i) the Transformer architecture and attention

mechanisms, (ii) pre-training and emergent capabilities, (iii) inference mechanisms including in-context learning and chain-of-thought reasoning, (iv) numerical optimization with LLMs, (v) multi-agent LLM systems, and (vi) limitations and mitigation strategies.

2.1.1 Transformer-Based LLMs: Architectural Overview

Modern LLMs are built on the Transformer architecture [41], which departs from recurrent neural networks (RNNs) and convolutional neural networks (CNNs) by relying entirely on self-attention mechanisms to capture dependencies between any pair of tokens in a sequence. This enables efficient parallel processing and the modelling of long-range relationships across large contexts.

The core innovation is *multi-head self-attention*, which allows each token to attend to all other tokens in the sequence. For an input sequence represented as a matrix $Z \in \mathbb{R}^{n \times d_{\text{model}}}$, the attention mechanism computes queries (Q), keys (K), and values (V) through learned linear projections, then produces weighted combinations via:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V. \tag{2.1}$$

Multiple attention heads operate in parallel, each learning to capture different types of dependencies, such as local versus global patterns or different relational structures. This multi-head mechanism, combined with position-wise feed-forward networks and residual connections, allows Transformers to build increasingly abstract representations through dozens to hundreds of stacked layers.

Most state-of-the-art LLMs adopt a *decoder-only* architecture with masked self-attention that enforces causal constraints [47]: each token can only attend to previous

tokens. This design naturally supports autoregressive generation, where the model predicts one token at a time conditioned on all preceding tokens, forming the foundation for both pre-training and inference.

2.1.2 Pre-Training at Scale and Emergent Capabilities

The remarkable capabilities of LLMs arise from large-scale pre-training on diverse text corpora. LLMs are trained with a simple autoregressive objective: predict the next token given all previous tokens. For a sequence $x_1, \ldots, x_T$, the training loss is

$$\mathcal{L} = -\frac{1}{T} \sum_{t=1}^{T} \log p_\theta\big(x_t \mid x_1, \ldots, x_{t-1}\big), \tag{2.2}$$

where θ denotes all model parameters. This unsupervised objective can be applied to any text: books, scientific papers, code repositories, and technical documentation. Through this process, models internalize syntax, semantics, domain knowledge, and reasoning patterns that enable them to assign high probability to plausible continuations across many domains [48].

Three dimensions of scale are critical: (i) *parameter scale*: modern LLMs contain billions to trillions of parameters, (ii) *data scale*: training corpora span trillions of tokens and (iii) *compute scale*: training employs thousands of accelerators over weeks or months.

A key empirical observation is the emergence of qualitatively new abilities once model and data scale exceed certain thresholds [49]:

- *Zero-shot and few-shot generalization*: solving tasks not explicitly seen during training based purely on natural-language instructions or a handful of examples.

- *In-context learning (ICL)*: learning new input-output mappings at inference time from examples in the prompt without updating parameters.

- *Chain-of-thought (CoT) reasoning*: generating explicit intermediate reasoning steps that improve performance on multi-step problems requiring logical decomposition.

These emergent capabilities enable LLMs to approach complex optimization problems by decomposing them into subproblems, reasoning about constraint interactions, and iteratively refining solutions based on feedback, properties that are particularly valuable for non-convex resource allocation in wireless networks.

After pre-training, models undergo instruction tuning and reinforcement learning from human feedback (RLHF) [50] to align them with user intent, follow structured formatting requirements, and produce outputs suitable for downstream processing. For wireless optimization, this means LLMs can understand problem descriptions, interpret constraints expressed in natural language, and generate solutions in prescribed formats such as JSON (JavaScript Object Notation) or structured tables.

2.1.3 Inference Mechanisms: ICL, CoT, and Iterative Reasoning

Autoregressive Decoding: At inference time, LLMs operate autoregressively: given a prompt comprising instructions and context, the model repeatedly generates the next token by sampling from the predicted probability distribution until a stopping condition is met. Decoding strategies include greedy selection (argmax), low-temperature sampling for deterministic outputs, or nucleus sampling for exploration. For optimization tasks, low-temperature or deterministic decoding is typically preferred to ensure reproducibility and stability [51].

In-Context Learning (ICL) ICL is particularly important for wireless applications. Instead of retraining a model for each network configuration, practitioners provide few-shot examples directly in the prompt:

```
Example 1:  [CSI matrix] → [power allocation vector]
Example 2:  [CSI matrix] → [power allocation vector]
⋮
Example K: [CSI matrix] → [power allocation vector]
New problem:  [CSI matrix] → ?
```

The model infers the underlying task structure from these demonstrations and produces a mapping for the new instance without any parameter updates. Internally, self-attention mechanisms allow the model to compare the new input against the provided examples and extrapolate patterns. Empirically, larger models exhibit stronger ICL capabilities [49], often approximating policies that would otherwise require training specialized DNNs on thousands of labelled optimization solutions [46].

For resource allocation, ICL enables: (i) zero- or few-shot adaptation to new channel distributions, user densities, or QoS profiles, (ii) reuse of expert-designed heuristics or conventional solver outputs as demonstrations, and (iii) rapid prototyping of optimization strategies without expensive offline training.

Chain-of-Thought (CoT) Reasoning: Beyond direct input-output mapping, LLMs can be prompted to generate intermediate reasoning steps. By providing instructions such as "explain your reasoning step by step" or including few-shot examples with explicit reasoning traces, the model produces intermediate calculations, subproblem decompositions, and constraint checks before emitting final solutions [52]. For constrained optimization, CoT reasoning is valuable for:

- Decomposing joint problems (e.g., separating user association from power control),
- Making constraint checks explicit (e.g., verifying power budgets and QoS requirements),

Multi-Turn Iterative Prompting: Beyond single-round inference, LLMs can engage in multi-turn conversations where each response builds on previous interactions. In an optimization context, this enables iterative refinement: the LLM proposes an initial solution, receives feedback on constraint violations or objective value, and generates an improved solution in the next turn. This iterative prompting paradigm is particularly well-suited for complex constrained optimization, where finding feasible solutions requires multiple refinement steps [53]. By incorporating historical context and structured feedback into subsequent prompts, LLMs can progressively improve solution quality while learning from previous mistakes, a capability that motivates the multi-agent feedback-driven framework developed in this thesis.

2.1.4 Numerical Optimization with LLMs

A critical consideration for applying LLMs to wireless resource allocation is how they represent and manipulate numerical data. LLMs are fundamentally trained on discrete tokens, not continuous values, which introduces both opportunities and challenges for optimization tasks involving real-valued decision variables.

Tokenization of Numerical Data Numbers are typically tokenized digit-by-digit or using special numeric tokens. For example, the value "3.14159" might be split into tokens ["3", ".", "14159"] or represented as a single token if it appears frequently in

the training corpus. This tokenization affects numerical precision: LLMs may struggle with arithmetic operations requiring exact calculations or high-precision outputs. For optimization variables such as beamforming weights or power allocations, this means values are typically represented with limited decimal precision (e.g., 3-4 significant digits) [54].

Structured Output Generation To reliably extract numerical solutions from LLM outputs, prompts must specify structured output formats such as JSON, YAML (YAML Ain't Markup Language), or delimited tables. For instance, a power control problem might prompt the LLM to return:

```
{"power_vector": [2.45, 1.83, 3.12, 0.97],
 "constraint_satisfied": true}
```

This structured approach enables automated parsing and validation of LLM-generated solutions. Modern LLMs trained with instruction tuning and RLHF reliably follow format specifications, though parsing failures can still occur and must be handled through exception handling or retry mechanisms.

Precision and Representation Limits LLMs exhibit inherent limitations in numerical reasoning: they may produce approximate rather than exact values, struggle with operations requiring many significant digits, and occasionally generate syntactically valid but semantically incorrect numbers. For wireless optimization, this necessitates post-processing validation where LLM outputs serve as initialization points or candidate solutions that are subsequently refined through conventional numerical methods. This motivates hybrid architectures that leverage LLMs for high-level

reasoning while relying on specialized optimization layers for precise constraint enforcement and fine-grained numerical adjustments.

2.1.5 Multi-Agent LLMs

Beyond single-model inference, recent work has explored multi-agent LLM architectures where multiple language models collaborate to solve complex tasks. In such systems, individual LLMs are assigned specialized roles and communicate through structured exchanges of information. This paradigm is particularly promising for optimization problems that involve multiple coupled decision variables or require iterative refinement.

Multi-agent LLM systems offer several advantages over monolithic single-agent approaches. First, they enable *task decomposition*, where complex problems are partitioned into manageable subproblems, each handled by a specialized agent with tailored prompts and domain knowledge. Second, they support *iterative refinement* through agent interaction, where different agents can critique, validate, or improve upon each other's outputs across multiple rounds. Third, they provide natural *interpretability*, as the communication between agents reveals the reasoning process and decision-making logic [55].

For wireless optimization, multi-agent architectures can naturally map to problem structure: different agents might handle different resource types, different network layers, or different stages of the optimization process. Agents can also play distinct functional roles, such as solution generation versus quality assessment, enabling collaborative workflows that leverage the complementary strengths of different prompting strategies or model configurations.

2.2 Time-Series Forecasting Preliminaries

The same agentic principle is also relevant beyond optimization. In forecasting problems, separate agents can specialize in different temporal structures, for example slow-varying trend estimation, short-term fluctuation modeling, retrieval of analogous historical examples, or evaluator-based consistency checks. This makes multi-agent LLM systems a natural bridge between the two application threads of this thesis: wireless resource allocation and multivariate time-series forecasting.

Before reviewing the forecasting literature, it is useful to summarize the main concepts underlying time-series forecasting in wireless systems. In the context of this thesis, forecasting is treated as a multivariate time-series prediction problem grounded in wireless measurements. The objective is to use historical observations of relevant system quantities to predict future behavior in a manner that remains both statistically meaningful and operationally useful.

2.2.1 Basic Time-Series Representation

A forecasting problem begins with the representation of the observed process. Let $x(t)$ denote a scalar quantity sampled at discrete time intervals, and let $\mathbf{x}_t \in \mathbb{R}^M$ denote a multivariate observation consisting of M correlated quantities measured at time t. In wireless applications, these quantities may correspond to traffic loads, channel-quality indicators, interference levels, user counts, mobility traces, energy-demand measurements, or other related variables that evolve jointly over time. [5]

A forecast is formed by mapping a historical window of length L to a future trajectory over horizon H. If the past observations are written as $\mathbf{X}_{\text{past}} = (\mathbf{x}_{t-L+1}, \ldots, \mathbf{x}_t)$, then the objective is to estimate $\mathbf{X}_{\text{future}} = (\mathbf{x}_{t+1}, \ldots, \mathbf{x}_{t+H})$. This multivariate setup

is more demanding than univariate extrapolation because useful prediction often depends not only on temporal structure within each series but also on dependence across series. [56]

In practice, raw wireless measurements may be collected from multiple channels, cells, bands, sectors, sensors, or technologies. These measurements are often transformed into a lower-dimensional multivariate forecasting target by grouping, aggregation, feature extraction, or signal fusion. More generally, a forecasting framework should be flexible enough to operate on any set of correlated series derived from wireless system measurements.

2.2.2 Multivariate Forecasting Formulation

This formulation immediately raises several practical questions. How should the history be normalized before inference? Which summary statistics are most informative for the forecaster? How should inter-series correlation be represented? And how can forecast plausibility be checked when the model is prompted rather than trained directly on the dataset? These questions are especially important in LLM-based forecasting, because the model typically reasons over structured numerical summaries and retrieved examples rather than learning directly from gradient updates.

2.2.3 Statistical Features Relevant to Forecasting

Several statistical properties are particularly relevant to multivariate wireless forecasting. The first is the *trend*, which captures slow variation in the mean level of the observed process. The second is *periodicity*, often tied to daily traffic cycles and regular usage patterns. The third is *local fluctuation structure*, reflecting short-term

deviations around the trend. The fourth is *cross-series dependence*, since related variables can provide contextual information for one another. These properties motivate decomposition-based forecasting strategies, where the overall signal is interpreted as the combination of a coarse trend component and a fast-varying residual component [56].

2.3 State-of-the-Art: LLMs for Wireless Applications

Having established the fundamental limitations of deep learning methods and the core capabilities of LLMs, we now review recent research that explores LLMs for wireless resource allocation. This emerging literature can be broadly categorized into *prompt-based optimization approaches* and *LLM-driven code generation approaches*, each offering distinct perspectives on leveraging pre-trained language models for RRM without the extensive training overhead of fine-tuning the models.

2.3.1 Fine-Tuning and Specialized Adaptations

Several works adapt LLMs to wireless networks via domain-specific fine-tuning or architectural modifications. The framework in [57] employs a Mixture-of-Experts (MoE) with Low-Rank Adaptation (LoRA) to simultaneously handle tasks such as channel estimation, prediction, localization, and beam management. Similarly, [58] proposes *WirelessLLM*, a domain-adapted LLM using knowledge alignment, multimodal pre-training, and fine-tuning, demonstrated through case studies in power allocation, spectrum sensing, and protocol understanding. Other studies extend reinforcement learning (RL) with LLMs, such as [59], which augments base station deployment optimization via an LLM-enhanced DDPG (Deep Deterministic Policy

Gradient) framework, and [60], where LLMs act as gates to select specialized Deep Reinforcement Learning (DRL) experts in a MoE architecture. A more constrained setting is explored in [61], where an LLM-based solver addresses a two-link resource allocation problem, combining LLM inference with classical heuristics for reliability. Finally, [62] introduces *LLM-OptiRA*, a framework that explicitly transforms non-convex optimization problems into solvable forms using approximation, structured prompting, coupled with error correction and feasibility validation loops.

These approaches report performance improvements over classical deep learning or optimization baselines, with [62] achieving up to 96% execution success and 80% feasible solutions on non-convex problems. However, their reliance on large, domain-specific datasets is a critical drawback, given the scarcity of realistic wireless datasets and the high cost of simulation-based generation. Furthermore, the methods remain largely task-specific: while [57] supports multiple tasks, they are optimized independently rather than jointly. None of these works attempt true joint optimization of coupled tasks such as power control with user association or beamforming with RIS configuration. Finally, fine-tuning introduces high computational and energy costs, limiting deployment in real-time or edge environments.

2.3.2 Retrieval-Augmented Generation (RAG)

Another line of research leverages retrieval mechanisms. *CommLLM* [63] introduces a multi-agent reasoning pipeline with retrieval, planning, and reflection for semantic communication. Similarly, *WirelessAgent* [64] develops an LLM-based agent framework with perception, memory, planning, and action modules, applied to network slicing. Hierarchical collaboration appears in [65], where one LLM deployed on a

high-altitude platform controls Uncrewed Aerial Vehicle (UAV) access, while onboard LLMs manage motion planning, jointly reducing collisions and improving communication reliability.

While these frameworks improve adaptability, evaluations focus on high-level outcomes such as semantic quality, system reward, or collision rates, but neglect classical optimization metrics such as sum-rate, convergence steps, or violation percentages. In addition, none of these works tackle joint optimization of multiple resource allocation tasks; rather, they focus on isolated subtasks.

2.3.3 LLM-Driven Code Generation Approaches

An alternative paradigm employs LLMs as automated algorithm designers that generate executable optimization code rather than directly producing resource allocations. This approach leverages LLMs' ability to understand problem descriptions and synthesize code in languages such as Python or domain-specific modeling frameworks like CVXPY (Convex Optimization in Python).

[62] proposed LLM-OptiRA, which prompts an LLM to decompose non-convex wireless resource allocation problems into sequences of convex subproblems and generate corresponding CVXPY solver code. When syntax or logical errors occur, follow-up prompts trigger self-correction, enabling iterative refinement of the generated code. This demonstrates LLMs' potential for automated algorithmic decomposition and rapid solver prototyping.

More comprehensive frameworks extend this concept to multi-modal and multi-agent architectures. WirelessLLM [58] integrates domain-specific knowledge retrieval

Table 2.1: Comparison of prompt design features and evaluation methods across existing LLM-based wireless optimization methods and our proposed LLM4RRM.

Resource Allocation Task	**Prompt Features**					**Prompt Evaluation**	
	Physics Informed	**CoT Reasoning**	**Multi Agent**	**Feedback Type**	**Constraint Handling**	**UQ**	**Prompt Analysis**
Power Control [66]	✗	✗	✗	Static	reward function	✗	✗
Power Control [67]	✗	✗	✗	No Feedback Loop	fallback algorithm	✗	✗
Network Planning [68]	✗	✗	✗	Static	reward function	✗	✗
Frequency & User Allocation [69]	✗	✗	✗	Static	reward function	✗	✗
Task Offloading [70]	✗	✗	✗	Static	reward function	✗	✗
Channel Allocation [71]	✗	✓	✗	No Feedback Loop	bandit search	✗	✗
UAV Planning and power control [65]	✗	✗	✗	Static	reward function	✗	✗
Power Control [72]	✗	✗	✓	Static	reward function	✗	✗
UAV motion control and BS association [73]	✓	✓	✓	Static	reward function	✗	✗
LLM4RRM	✓	✓	✓	Dynamic	projection function	✓	✓

with LLM-based code generation to address spectrum sensing, interference mitigation, and resource allocation tasks. WirelessAgent [64] adopts a modular multi-agent design where individual LLM agents generate optimization code for subproblems such as network slicing, beamforming, or scheduling, with inter-agent communication coordinating the overall solution.

While code generation approaches offer flexibility in adapting to diverse problem formulations, they introduce distinct challenges. Generated code incurs runtime overhead from solver execution, is susceptible to hallucinations that produce syntactically valid but logically incorrect algorithms, and provides no inherent feasibility guarantees. The performance of these methods is highly sensitive to code correctness and prompt design, making them better suited as design assistants for offline algorithm development rather than real-time optimization engines for latency-critical 6G applications.

2.3.4 Prompt-Based Optimization Approaches

Recently, a handful of research works have begun exploring the potential of LLMs in wireless communications, leveraging prompt engineering and in-context learning (ICL) techniques to guide LLMs in solving RRM problems without task-specific model training. However most of these works have limitations in their approaches. For example, [66] proposed reinforced in-context learning for power control using reward-labeled examples, but relied on static prompts and soft penalty-based constraint handling. [67] employed few-shot prompting for two-user power control with a hybrid fallback scheme, without closed-loop refinement. [69] introduced a tool-assisted feedback loop for uplink resource allocation, where constraint violations are reported externally to guide iterative refinement.

In addition, [70] applied template-based prompting for satellite Mobile Edge Computing (MEC) task offloading with latency-based feedback and penalty mechanisms. Also, [65] integrated LLMs into evolutionary optimization for uncrewed aerial vehicle (UAV) assisted integrated sensing and communication (ISAC), using static fitness evaluation without semantic feedback. Moreover, [71] utilized chain-of-thought prompting to initialize bandit-based channel allocation, where optimization and constraint enforcement are handled outside the LLM. [68] embedded ray-tracing-based feedback into prompts for access point placement. [72] proposed decentralized multi-agent LLMs for power control using reward-driven coordination. Finally, [73] developed a hierarchical dual-LLM framework for joint UAV mobility and communication control with experience-based prompting.

Despite the recent advances, the prompts in the existing works suffer from certain limitations. First, the prompts used in them are not physics-informed, meaning

that they do not include mathematical models to compute physics-based received signal/interference powers as well as data rate, which can guide the LLM's reasoning process. Second, only a couple of the works considered implementing chain of thought (CoT) reasoning [71,73] and problem decomposition to multiple LLMs [72,73]. Third, all of the existing works use static feedback texts within prompts (which replace only the numerical result values in each iteration). Fourth, constraint satisfaction is typically handled by augmenting the constraint within reward functions or using search algorithms [71] or fallback mechanisms [67] without any feasibility guarantees. Finally, none of the existing works provide uncertainty quantification (UQ) of their LLM prompts and outputs. Table 2.1 shows these differences in terms of prompt structure, feedback mechanisms, constraint handling, and prompt evaluation.

2.3.5 Time-Series Forecasting and LLM-Based Forecasting Literature

The discussion above focuses on LLMs for wireless resource allocation, which forms the primary optimization thread of this thesis. However, the same training-free and agentic LLM paradigm is also relevant to forecasting problems, where the objective is not to optimize instantaneous decisions but to predict future system behavior under changing network conditions. This motivates a complementary review of the broader time-series forecasting literature and the emerging literature on LLM-based forecasting.

Traditional time-series forecasting methods have long relied on statistical models that explicitly capture trend, seasonality, and residual variation. Classical approaches such as autoregressive integrated moving average (ARIMA) and related decomposition-based methods provide interpretable forecasts under assumptions of

approximate linearity and stationarity [74]. These methods remain useful for well-structured signals, but their performance often degrades when the data exhibit strong nonlinearity, complex multivariate dependence, or rapidly changing dynamics.

Learning-based forecasting methods improve expressive power by modeling nonlinear temporal patterns directly from data. Recurrent models such as LSTMs have shown strong performance in sequential prediction tasks by capturing long-range dependencies [75], while convolutional temporal models can effectively identify recurring local patterns and short-term motifs in time-series data [76]. More recent probabilistic and sequence-modeling approaches incorporate uncertainty estimation, attention mechanisms, or generative forecasting pipelines to improve robustness and predictive flexibility [77, 78]. Despite their strong performance, these approaches generally rely on dataset-specific training and therefore inherit the usual challenges of retraining overhead, domain dependence, and limited adaptability to new deployment conditions.

Alongside these forecasting advances, recent work has shown that LLMs can also be used as training-free forecasters. [44] established that LLMs can perform zero-shot time-series forecasting through in-context prediction, demonstrating that pre-trained language models can extrapolate numerical sequences without explicit parameter updates. More recent agentic pipelines such as FLAIRR-TS further improved this paradigm by introducing retrieval, iterative refinement, and structured feedback into the forecasting loop [79]. At the same time, prior studies have highlighted important limitations of prompting-only forecasting, especially sensitivity to numerical noise and the need for stronger statistical grounding before values are passed to the model [80].

Taken together, the broader time-series forecasting literature and the emerging

literature on training-free LLM-based forecasting reveal a clear research opportunity. Classical and deep learning forecasters provide strong task-specific performance but require retraining, whereas agentic LLM methods offer adaptability and cross-task transferability but have not yet been systematically developed for multivariate wireless forecasting with structured decomposition and evaluator-guided refinement. This gap motivates the forecasting contribution of the thesis: extending the prompt-engineered, multi-agent LLM paradigm beyond optimization to multivariate time-series forecasting, while preserving the training-free and adaptable nature of the proposed LLM based framework.

Chapter 3

Multi-Agent Feedback-Driven LLM FOR Constrained RRM

In this chapter, we introduce LLM4RRM, a unified, training-free, multi-agent large language model framework for solving constrained and non-convex radio resource management problems. As illustrated in Fig. 3.1, the proposed framework adopts a feedback-driven, block-coordinate descent–inspired optimization architecture in which decision variables are decomposed across specialized solver agents and iteratively refined through structured feedback from an evaluator agent. To rigorously enforce feasibility, the framework incorporates an LLM-guided adaptive differentiable projection mechanism that dynamically adjusts projection hyperparameters, enabling efficient convergence to the feasible region without manual tuning. Furthermore, to address the inherent stochasticity and reliability concerns of LLM-generated solutions, we introduce a prompt-based uncertainty quantification post-analysis that systematically evaluates the robustness and trustworthiness of solutions.

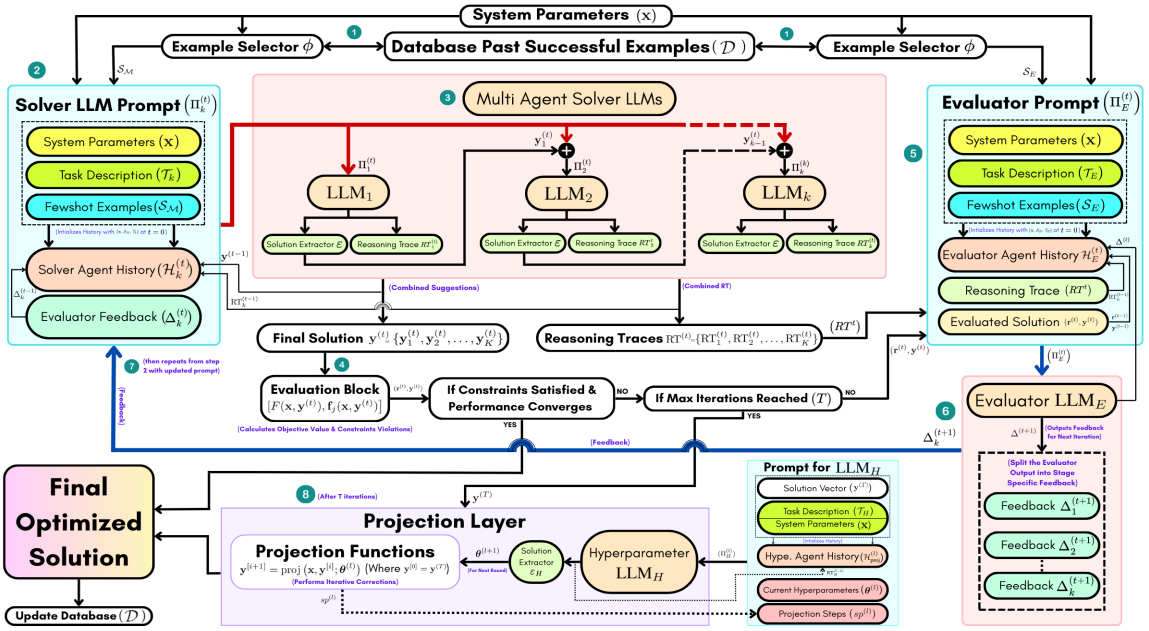


Figure 3.1: Overall architecture of the proposed LLM4RRM framework illustrating multi-agent reasoning, evaluator feedback, and adaptive projection for feasible and reliable resource optimization.

3.1 LLM4RRM Framework

This framework models the optimization process as a cooperative reasoning sequence among multiple LLMs, where each *solver agent* is responsible for optimizing a subset of decision variables under structured feedback from an *evaluator agent*. The overall system mimics a language-driven BCD method, where each agent updates its variable subset while other variables remain fixed.

3.1.1 Problem Definition:

Let $\mathbf{x}$ represent the system parameters (e.g., channel state information, noise variance, etc.), $\mathbf{y}$ denotes an Υ-dimensional decision vector, and $\mathcal{C}$ denotes the task specification, which defines the optimization problem as follows

$$\mathcal{C}: \quad \max_{\mathbf{y}} F(\mathbf{x}, \mathbf{y}) \quad \text{s.t.} \quad c_j(\mathbf{x}, \mathbf{y}) \triangleright_j 0,\ j = 1, 2, \ldots, J \tag{3.1}$$

where $F(\cdot)$ is any real-valued objective function (e.g., network sum-rate) and $c_j(\cdot)$ represents a real-valued j-th constraint (e.g., QoS requirement), where $\triangleright_j$ refers to either " $\geq$ ", " $\leq$ " or " $=$ " for each j-th constraint. Furthermore, it is important to note that some constraints can be eliminated via closed-form transformations, as demonstrated in [32].

3.1.2 Solver LLM Agents:

The decision vector is divided into K blocks, such as $\mathbf{y} = \{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_K\}$, where each $\mathbf{y}_k$ corresponds to a k-th subtask such as beamforming, phase control, or user association. The k-th block is optimized by a dedicated *solver* agent, LLM_k. As illustrated in Fig. 1, the proposed framework operates iteratively with $t = 0, 1, \cdots, T$, where T denotes the maximum number of iterations. For a given iteration t, each solver agent produces a text-based response, which is then parsed into machine-interpretable variables. Formally, the output of the k-th LLM agent at t-th iteration can be expressed as follows

$$\mathbf{y}_k^{(t)} = \mathcal{E}\left(\text{LLM}_k\big(\Pi_k^{(t)}\big)\right), \quad \text{for } t \geq 0, \tag{3.2}$$

where $\Pi_k^{(t)}$ is the prompt given to k-th LLM agent at t-th iteration and $\mathcal{E}(\cdot)$ denotes an extraction function that converts the textual output into structured numerical representations (e.g., vectors, matrices, or JSON objects).

3.1.3 Prompt Construction for Solver Agents:

For an iteration t, a prompt ($\Pi_k^{(t)}$) is constructed for each k-th solver agent, LLM_k as:

$$\Pi_k^{(t)} = \mathcal{P}_k\big(\mathbf{y}_{1:k-1}^{(t)},\, \Delta_k^{(t)},\, \mathcal{H}_k^{(t)}\big), \quad \text{for } t \geq 0, \tag{3.3}$$

where $\mathcal{P}_k(\cdot)$ denotes the prompt-construction function that maps the prior solver agents outputs $\mathbf{y}_{1:k-1}^{(t)}$, the evaluator's feedback available for the current iteration t, $\Delta_k^{(t)}$, and historical context $\mathcal{H}_k^{(t)}$ defined in (3.4) into a structured natural-language prompt. It is noteworthy that the inclusion of prior solver agents' outputs, $\{\mathbf{y}_1^{(t)}, \ldots, \mathbf{y}_{k-1}^{(t)}\}$ ensures that each solver operates with the most up-to-date contextual information from preceding blocks. Moreover, $\mathcal{H}_k^{(t)}$ comprises transcripts from previous iterations, e.g., past solutions and reasoning traces of k-th solver. Mathematically, historical context for LLM_k is given as follows

$$\mathcal{H}_k^{(t)} = \begin{cases} \{\mathbf{x},\, \mathcal{S}_M,\, \mathcal{T}_k\} & \text{for } t = 0, \\ \{\mathcal{H}_k^{(t-1)}, \Delta_k^{(t-1)},\, \mathbf{y}^{(t-1)},\, \text{RT}_k^{(t-1)}\}, & \text{for } t \geq 1, \end{cases} \tag{3.4}$$

where $\mathbf{y}^{(t-1)} = \{\mathbf{y}_1^{(t-1)}, \mathbf{y}_2^{(t-1)}, \ldots, \mathbf{y}_K^{(t-1)}\}$. The textual reasoning corresponding to k-th decision, including intermediate derivations and explanatory steps, is recorded as the *reasoning trace* $\text{RT}_k^{(t)}$, which is later examined by the evaluator agent.

Note that $\Delta_k^{(0)}$ is $\emptyset$ because there is no prior feedback in the first iteration (i.e.,

$t = 0$). Also, at $t = 0$, historical context for each solver agent contains the system parameters $\mathbf{x}$, a few-shot example set $\mathcal{S}_{\mathcal{M}}$, and task description, $\mathcal{T}_k$, as shown in (3.4). The few-shot example set $\mathcal{S}_{\mathcal{M}}$ is generated by retrieving the most relevant examples from $\mathcal{D}$ using any similarity function like in [81–83], where $\mathcal{D}$ denotes a memory bank of past successful optimization examples, each stored as a tuple $(\mathbf{x}, \mathbf{y}, F, \mathbf{c})$, where $\mathbf{c}(\cdot) := [c_1, c_2, \ldots, c_J]$. In addition, $\mathcal{T}_k$ represents a task description for k-th solver, which constitutes a role of solver agent and a specification of the task $\mathcal{C}$, given in (3.1).

3.1.4 Evaluator Agent and Feedback Generation ($\Delta_k^{(t)}$):

Once all k agents produce their reasoning traces and outputs, the evaluation stage begins. The role of *evaluator* agent LLM_E is to assess the joint solution, detect inconsistencies across each solver's reasoning, and generate structured feedback to guide the *next reasoning cycle* (i.e., for $t+1$ iteration). The evaluator agent's output is the high-level textual critique generated by the evaluator, i.e.,

$$\Delta^{(t+1)} = \text{LLM}_E\big(\Pi_E^{(t)}\big), \quad \text{for } t \geq 0, \tag{3.5}$$

where $\Pi_E^{(t)}$ is the prompt given to LLM_E at t-th iteration. After obtaining $\Delta^{(t+1)}$, a parsing function, $\mathcal{E}_E(\cdot)$ is applied to obtain structured sub-feedback for each solver agent, i.e.,

$$\{\Delta_k^{(t+1)}\}_{k=1}^{K} = \mathcal{E}_E\big(\Delta^{(t+1)}\big), \quad \text{for } t \geq 0. \tag{3.6}$$

The structured prompt for the evaluator agent, LLM_E is constructed as follows

$$\Pi_E^{(t)} = \mathcal{P}_E\big(\mathbf{r}^{(t)},\, \mathbf{y}^{(t)},\, \text{RT}^{(t)},\, \mathcal{H}_E^{(t)}\big), \text{for } t \geq 0, \tag{3.7}$$

where $\mathbf{r}^{(t)} \triangleq \{F(\mathbf{x}, \mathbf{y}^{(t)}), \mathbf{f}_j(\mathbf{x}, \mathbf{y}^{(t)})\}$, here $\mathbf{f}_j(\mathbf{x}, \mathbf{y}^{(t)})$ is a binary vector which states whether a constraint $\mathbf{c}_j$ is satisfied or violated, and
$\text{RT}^{(t)} \triangleq \{\text{RT}_1^{(t)}, \text{RT}_2^{(t)}, \ldots, \text{RT}_K^{(t)}\}$. Moreover, all prior evaluator interactions, i.e., the historical context of LLM_E is given as follows:

$$\mathcal{H}_E^{(t)} = \begin{cases} \{\mathbf{x},\, \mathcal{S}_E,\, \mathcal{T}_E\} & \text{for } t = 0, \\ \{\, \mathcal{H}_E^{(t-1)}, \Delta^{(t)},\, \mathbf{r}^{(t-1)}, & \\ \mathbf{y}^{(t-1)}, \text{RT}^{(t-1)}\,\}, & \text{for } t \geq 1, \end{cases} \tag{3.8}$$

where $\mathcal{S}_E$ denotes the few-shot demonstration set assigned to the evaluator agent, $\mathcal{T}_E$ represents a task description in which the evaluator agent is assigned a role of a *critic* or *evaluator*.

Special Case ($K = 1$): When $K = 1$, the framework reduces to a two-agent loop involving a single solver agent for all the optimization variables, i.e., LLM_1, and one evaluator agent, LLM_E. While structurally simpler, this framework still benefits from external feedback supervision, ensuring that reasoning errors or hallucinations are detected and corrected.

3.1.5 Projection Layer with Hyperparameter-Tuning LLM:

After T reasoning iterations, LLM4RRM produces solution, $\mathbf{y}^{(T)}$, which may exhibit constraint violations. Inspired by the work of [32], we propose employing a projection

layer that ensures feasibility by mapping $\mathbf{y}^{(T)}$ onto the feasible region, as shown in Fig. 3.1. However, unlike [32], which employs a conventional projection layer with *fixed* hyperparameters, we propose employing a *hyperparameter* LLM, LLM_H, which outputs the hyperparameters for projection at the end of each l-th round, where $l = 0, 1, \ldots, L$, here L is a predefined number which determines how many times LLM_H is used. The output of LLM_H can thus be defined as follows:

$$\boldsymbol{\theta}^{(l+1)} = \mathcal{E}_H\big(\text{LLM}_H(\Pi_H^{(l)})\big), \tag{3.9}$$

where $\boldsymbol{\theta}^{(l+1)}$ denotes the set of hyperparameter(s) such as step size $\mu^{(l+1)}$, momentum $\gamma^{(l+1)}$, or penalty weighting $\delta^{(l+1)}$, etc., which are used for the next round (i.e., $l+1$ round). Moreover, $\mathcal{E}_H(\cdot)$ denotes an operator that parses the textual output into numeric hyperparameters. It is important to mention here that $\boldsymbol{\theta}^{(0)}$ is initialized to any predefined value. Moreover, $\Pi_H^{(l)}$ denotes a structured prompt, which is constructed for LLM_H at the start of the l-th round. Formally, this prompt is given as follows:

$$\Pi_H^{(l)} = \mathcal{P}_H\big(\boldsymbol{\theta}^{(l)},\, sp^{(l)},\, \mathcal{H}_{\text{proj}}^{(l)}\big), \tag{3.10}$$

where $\mathcal{P}_H(\cdot)$ is the prompt-construction function. The term $sp^{(l)}$ represents the collection of projection-step records obtained with the current settings of $\boldsymbol{\theta}^{(l)}$, which includes the intermediate projected values (from eq. (3.12)) and the corresponding constraint evaluations. Moreover, $\mathcal{H}_{\text{proj}}^{(l)}$ represents the projection history, as defined below:

$$\mathcal{H}_{\text{proj}}^{(l)} = \begin{cases} \{\mathbf{x},\, \mathcal{T}_H,\, \mathbf{y}^{(T)}\} & \text{for } l = 0, \\ \{\mathcal{H}_{\text{proj}}^{(l-1)}, \boldsymbol{\theta}^{(l-1)},\, sp^{(l-1)}, \text{RT}_H^{(l-1)}\}, & \text{for } l \geq 1, \end{cases} \tag{3.11}$$

where $\mathcal{T}_H$ represents the task description assigned to LLM_H which includes the role of LLM_H, task objective and strategies. Moreover, RT_H represents the reasoning trace of LLM_H.

Let $\mathbf{y}^{[0]} = \mathbf{y}^{(T)}$ denote the initial point for the projection process. The projection process at lth round attempts to enforce feasibility through iterative updates of the form:

$$\mathbf{y}^{[i+1]} = \text{proj}\left(\mathbf{x}, \mathbf{y}^{[i]}; \boldsymbol{\theta}^{(l)}\right), \ \forall i = 0, 1, 2 \ldots I, \tag{3.12}$$

where $\text{proj}(\cdot)$ represents a projection function and I denotes the maximum number of projection steps. The process repeats until all constraints are satisfied or a maximum number of rounds, i.e., L are reached. The final feasible solution is obtained via eq. (3.12) for $\boldsymbol{\theta}^{(l)} = \boldsymbol{\theta}^*$, where $\boldsymbol{\theta}^*$ denotes the stable hyper-parameter configuration on convergence.

3.1.6 Uncertainty Quantification and Trustworthiness:

While the projection layer enforces feasibility and the evaluator LLM enhances trustworthiness by identifying implausible or hallucinated reasoning, these components do not yield *quantitative* measures of reliability. To address this limitation, we incorporate an *Uncertainty Quantification* component that characterizes both epistemic uncertainty (arising from limitations in the LLM's pre-trained knowledge) and aleatoric uncertainty (resulting from inherent variability in the input data) [84]. The component functions as a *post-processing uncertainty analysis*, operates independently of the main optimization framework, and generates stability indicators based on controlled perturbations of inputs, prompts, and outputs. Since we employ closed-source LLMs

accessed via APIs (Application Programming Interface), internal model weights, logits, and hidden representations are inaccessible; therefore, uncertainty must be estimated using black-box sampling strategies based on observable inputs, prompts, and outputs. To estimate uncertainty, we employ the following sampling strategies, each designed to probe a different source of variability:

- Input-based sampling: Perturbs the system parameters $\mathbf{x}$ provided to the LLM agent to evaluate robustness across varying operating conditions.
- Prompt-based sampling: Samples alternate prompt configurations for each Π_k (e.g., by modifying structure, phrasing, or description) to assess sensitivity to prompt formulation.
- Output-based sampling: Keeps the prompt and system parameters fixed while generating multiple outputs to capture stochastic variation induced by the LLM's decoding process.

For the first two sampling strategies, uncertainty is assessed using the objective values and constraint evaluations corresponding to the main optimization problem in (3.1). In contrast, for the third sampling strategy, statistical analysis is performed on the variability across multiple generated solutions using distance- or deviation-based measures. We define the following quantitative metrics to characterize both the dispersion and the consistency of repeated LLM-generated outputs.

3.1.6.1 Pairwise L2-norm Based Distance:

For a given prompt, we consider $\psi \in \{1, \ldots, \Psi\}$ different realizations of the input system parameters, resulting in Ψ prompt–input pairs for each prompt. For each such

pair, Λ outputs or $\lambda \in \{1, \ldots, \Lambda\}$ are generated by running the proposed optimization framework Λ times, yielding $\{\mathbf{y}_{\psi,1}^{(T)}, \mathbf{y}_{\psi,2}^{(T)}, \ldots, \mathbf{y}_{\psi,\Lambda}^{(T)}\}$. Our proposed pairwise L2-norm based distance metric can thus be computed as follows

$$\mathcal{D}_{\text{UQ}} = \frac{1}{\Psi} \sum_{\psi=1}^{\Psi} \left(\frac{2}{\Lambda(\Lambda-1)} \sum_{1 \leq \lambda < s \leq \Lambda} \left\| \mathbf{y}_{\psi,\lambda}^{(T)} - \mathbf{y}_{\psi,s}^{(T)} \right\|_2^2 \right), \tag{3.13}$$

where $\| \cdot \|_2$ denotes the L2 norm, λ and s represent indices of two distinct generated outputs for the same prompt–input pair. This metric performs pairwise comparisons across all Λ output vectors for each input realization, measuring the overall dispersion of the solution space. The constraint $\lambda < s$ ensures that each pair is counted only once, as the distance between outputs λ and s is symmetric (i.e., comparing outputs 1 and 2 is equivalent to comparing outputs 2 and 1). The normalization factor $\frac{2}{\Lambda(\Lambda-1)}$ accounts for the total number of unique pairwise combinations, which equals $\binom{\Lambda}{2} = \frac{\Lambda(\Lambda-1)}{2}$, thereby averaging over all possible pairs. The metric, $\mathcal{D}_{\text{UQ}}$, therefore represents the output dispersion, averaged over Ψ input realizations, and captures the stochastic variability of the LLM outputs.

3.1.6.2 Mean Absolute Deviation (MAD):

Let $y_{\psi,\lambda,\upsilon}^{(T)}$ denote the υ-th element of the vector $\mathbf{y}_{\psi,\lambda}^{(T)}$, then the proposed MAD metric, averaged over Ψ prompt–input pairs with a fixed prompt and different input realizations, is given as

$$\text{MAD}_{\text{UQ}} = \frac{1}{\Psi} \sum_{\psi=1}^{\Psi} \left(\frac{1}{\Lambda \Upsilon} \sum_{\lambda=1}^{\Lambda} \sum_{\upsilon=1}^{\Upsilon} \left| y_{\psi,\lambda,\upsilon}^{(T)} - \overline{y}_{\psi,\upsilon}^{(T)} \right| \right), \tag{3.14}$$

where $\overline{y}^{(T)}_{\psi,\upsilon} = \frac{1}{\Lambda}\sum_{\lambda=1}^{\Lambda} y^{(T)}_{\psi,\lambda,\upsilon}$ denotes mean of the υ-th element of $\mathbf{y}^{(T)}_{\psi,\lambda}$ over Λ generated outputs. It is important mentioning that $\mathrm{MAD}_{\mathrm{UQ}}$ captures the element-wise deviation of LLM-generated outputs from their empirical means, aggregated across both repeated decodings and multiple prompt-input pairs. Lower values of either $\mathcal{D}_{\mathrm{UQ}}$ or $\mathrm{MAD}_{\mathrm{UQ}}$ indicate greater consistency and reduced stochastic variability in the LLM-generated solutions. Their usage is demonstrated in Section 4.4 to assess uncertainty in the LLM outputs.

3.1.7 Theoretical Grounding of Uncertainty Quantification Metrics

This section provides a more detailed theoretical justification for the proposed uncertainty quantification metrics $\mathcal{D}_{\mathrm{UQ}}$ and $\mathrm{MAD}_{\mathrm{UQ}}$, including their statistical interpretation, a comparison with classical UQ approaches, and a discussion of their suitability for black-box LLM-driven RRM optimization.

3.1.7.1 Statistical Interpretation of $\mathcal{D}_{\mathrm{UQ}}$

The pairwise L2-norm based distance metric $\mathcal{D}_{\mathrm{UQ}}$, defined in equation (3.13) of the main manuscript, has a rigorous statistical foundation. When the LLM is queried Λ times with the same prompt-input pair, it produces Λ solution vectors $\{\mathbf{y}^{(T)}_{\psi,1}, \ldots, \mathbf{y}^{(T)}_{\psi,\Lambda}\}$. For any two such draws $\mathbf{y}_\lambda$ and $\mathbf{y}_s$, the following identity holds directly from expanding the squared norm and applying linearity of expectation:

$$\mathbb{E}\big[\|\mathbf{y}_\lambda - \mathbf{y}_s\|_2^2\big] = 2\sum_{\upsilon=1}^{\Upsilon} \mathrm{Var}(y_\upsilon). \tag{3.15}$$

Thus, $\mathcal{D}_{\mathrm{UQ}}$ is an unbiased estimator of twice the total output variance across all decision variable dimensions [85]. This result is distribution-agnostic and holds regardless

of whether the underlying channel is simulated or obtained from real measurements, making $\mathcal{D}_{\mathrm{UQ}}$ a statistically grounded reliability indicator applicable to both simulation and real-world deployment scenarios. Lower values of $\mathcal{D}_{\mathrm{UQ}}$ indicate that solutions cluster tightly around their mean, reflecting stable and consistent LLM reasoning.

3.1.7.2 Statistical Interpretation of $\mathrm{MAD}_{\mathbf{UQ}}$

The Mean Absolute Deviation metric $\mathrm{MAD}_{\mathrm{UQ}}$, defined in equation (3.14) of the main manuscript, is a direct element-wise extension of the classical Mean Absolute Deviation (MAD), which is a well-established robust alternative to variance and standard deviation in classical statistics [86]. In contrast to $\mathcal{D}_{\mathrm{UQ}}$, which captures global solution dispersion, $\mathrm{MAD}_{\mathrm{UQ}}$ provides element-wise consistency assessment, revealing which specific decision variables (e.g., individual beamforming weights, phase shifts, or power allocations) contribute most to output instability. This per-variable diagnostic capability enables targeted prompt refinement for unstable variables and is more informative than scalar variance metrics.

3.2 Use-Case Study I: Joint Beamforming and Phase Shift Optimization in STAR-RIS Networks

3.2.1 System Model

We consider a narrowband simultaneously transmitting and reflecting-RIS (STAR-RIS) aided downlink communication system in which an M-antenna base station (BS) simultaneously serves $U = U_t + U_r$ single-antenna users, where U_t users, indexed by $\mathcal{U}_t = \{1, \ldots, U_t\}$ and U_r users, indexed by $\mathcal{U}_r = \{U_t + 1, \ldots, U_t + U_r\}$ are present in transmission space and reflection space of the STAR-RIS, respectively. Moreover,

STAR–RIS comprises N elements, indexed by $\mathcal{N}=\{1,\ldots,N\}$ and the direct BS–user channels are assumed to be blocked due to obstacles. The received signal at the u-th user, with $u \in \mathcal{U}$ and $\mathcal{U} = \mathcal{U}_t \cup \mathcal{U}_r$, is written as $v_u = \mathbf{h}_u^H \boldsymbol{\Theta}_l \mathbf{G} \sum_{i\in\mathcal{U}} \mathbf{w}_i s_i + z_u$, where $\mathbf{G} \in \mathbb{C}^{N\times M}$ represents the BS–RIS channel, $\mathbf{h}_u \in \mathbb{C}^{N\times 1}$ denotes the RIS–user channel, and $z_u \sim \mathcal{CN}(0,\sigma_u^2)$ represents the additive white Gaussian noise (AWGN) with noise power σ_u^2. Each STAR–RIS element controls its transmission and reflection through $\boldsymbol{\Theta}_l = \mathrm{diag}(\beta_{1,l}e^{j\phi_{1,l}}, \beta_{2,l}e^{j\phi_{2,l}}, \ldots, \beta_{N,l}e^{j\phi_{N,l}})$, where $\beta_{n,l} \in [0,1]$ and $\phi_{n,l} \in [0,2\pi)$ denote the amplitude and phase shift of the n-th element for side $l \in \{t,r\}$, respectively. The BS employs beamforming vectors $\mathbf{w}_i \in \mathbb{C}^{M\times 1}$ to transmit symbols $s_i \in \mathbb{C}$ with unit average power, i.e., $\mathbb{E}[|s_i|^2] = 1$. The instantaneous signal-to-interference-noise-ratio (SINR) at user u is given as follows

$$\gamma_u = \frac{|\mathbf{h}_u^H \boldsymbol{\Theta}_l \mathbf{G} \mathbf{w}_u|^2}{\sum\limits_{i\in\mathcal{U}, i\neq u} |\mathbf{h}_u^H \boldsymbol{\Theta}_l \mathbf{G} \mathbf{w}_i|^2 + \sigma_u^2}, \tag{3.16}$$

and the corresponding sum-rate is $R_u = \log_2(1+\gamma_u)$.

3.2.2 Problem Formulation

Let $\mathbf{W} = [\mathbf{w}_1,\ldots,\mathbf{w}_U] \in \mathbb{C}^{M\times U}$ denote the beamforming matrix, $\mathbf{H} = [\mathbf{h}_1,\ldots,\mathbf{h}_U] \in \mathbb{C}^{N\times U}$ is the matrix of the STAR-RIS to user channels, $\boldsymbol{\phi}_l = [\phi_{1,l},\ldots,\phi_{N,l}]$, and $\boldsymbol{\beta}_l = [\beta_{1,l},\ldots,\beta_{N,l}]$ represent the RIS phase shift and amplitude vectors, respectively. Moreover, let $\mathbf{x} \triangleq \{\mathbf{G},\mathbf{H},\sigma_u^2\}$, and a decision variable vector be $\mathbf{y} = \{\mathbf{W},\boldsymbol{\phi}_t,\boldsymbol{\phi}_r,\boldsymbol{\beta}_t,\boldsymbol{\beta}_r\}$,

then the joint sum-rate maximization problem can be written as follows:

$$\max_{\mathbf{y}} \quad \sum_{u\in\mathcal{U}} R_u(\mathbf{x},\mathbf{y}), \tag{3.17a}$$

$$\text{s.t.} \quad \sum_{u\in\mathcal{U}} \|\mathbf{w}_u\|^2 \leq P_{\max}, \tag{3.17b}$$

$$R_u(\mathbf{x},\mathbf{y}) \geq R_{\min},\ \forall u \in \mathcal{U}, \tag{3.17c}$$

$$(\beta_{n,t})^2 + (\beta_{n,r})^2 = 1,\ \forall n \in \mathcal{N}, \tag{3.17d}$$

where $P_{\max}$ and $R_{\min}$ represent the maximum power budget of the BS and minimum rate requirement for each user. Moreover, constraint (3.17d) is due to the fact that the energy splitting mode is considered. Owing to the coupling of variables within the sum-rate formulation, the problem in (3.17) is inherently non-convex [32].

In order to eliminate the equality constraint given in (3.17d), we adopt the closed-form transformations $\beta_{n,t} = \sqrt{\text{Sig}(a_n)},\ \beta_{n,r} = \sqrt{1-\text{Sig}(a_n)},\ \forall n \in \mathcal{N}$, given in [32], where $\text{Sig}(\cdot) \triangleq 1/(1+e^{-(\cdot)})$ and $a_n \in \mathbb{R}$ is an *unconstrained* scalar. Moreover, the special structure of constraint (3.17b) allows us to handle this constraint by using the following transformation $\mathbf{W} = \frac{\hat{\mathbf{W}}}{\|\hat{\mathbf{W}}\|_F} \times \text{Sig}(\alpha) \times \sqrt{P_{\max}}$, where $\hat{\mathbf{W}} \in \mathbb{C}^{M\times U}$ denotes the *unconstrained* beamforming matrix. Thus, the closed-form transformations result in the following optimization problem:

$$\max_{\hat{\mathbf{y}}} \quad \sum_{u\in\mathcal{U}} R_u(\mathbf{x},\mathcal{T}(\hat{\mathbf{y}})) \tag{3.18a}$$

$$\text{s.t.} \quad R_u(\mathbf{x},\mathcal{T}(\hat{\mathbf{y}})) \geq R_{\min},\ \forall u \in \mathcal{U}, \tag{3.18b}$$

where $\mathcal{T}(.)$ denotes a combined transformation which takes unconstrained decision

vector $\hat{\mathbf{y}} \triangleq \{\hat{\mathbf{W}}, \alpha, \mathbf{a}, \boldsymbol{\phi}_t\, \boldsymbol{\phi}_r\}$ as input and returns constrained decision vector $\mathbf{y}$. Note that here $\mathbf{a} \triangleq [a_1, a_2, \ldots, a_N]$.

3.2.3 Implementation of the Proposed LLM Framework

We adopt the LLM4RRM framework to solve the problem (3.18). To reduce the prompt dimensionality and enable cooperative reasoning, the unconstrained decision vector $\hat{\mathbf{y}}$ is partitioned into two blocks, and at any t-th iteration, we have

$$\hat{\mathbf{y}}_1^{(t)} = \{\hat{\mathbf{W}}^{(t)},\, \alpha^{(t)}\}, \qquad \hat{\mathbf{y}}_2^{(t)} = \{\mathbf{a}^{(t)},\, \boldsymbol{\phi}_t^{(t)},\, \boldsymbol{\phi}_r^{(t)}\}. \tag{3.19}$$

It is evident from (3.19) that the solver agent 1 (LLM_1) focuses on active beamforming, while Agent 2 (LLM_2) optimizes the STAR–RIS amplitude and phase components. At each iteration t, the k-th solver agent produces its decision as:

$$\hat{\mathbf{y}}_k^{(t)} = \mathcal{E}\Big(\text{LLM}_k\big(\Pi_k^{(t)}\big)\Big), \quad k \in \{1, 2\}, \quad t \geq 0, \tag{3.20}$$

where $\mathcal{E}(\cdot)$ extracts the structured output from the LLM's textual response. The prompt $\Pi_k^{(t)}$ for the k-th solver agent is constructed as follows:

$$\Pi_k^{(t)} = \mathcal{P}_k\big(\hat{\mathbf{y}}_{1:k-1}^{(t)},\, \Delta_k^{(t)},\, \mathcal{H}_k^{(t)}\big), \quad t \geq 0, \tag{3.21}$$

where $\hat{\mathbf{y}}_{1:k-1}^{(t)}$ contains the outputs from prior solver agents, $\Delta_k^{(t)}$ is the evaluator's feedback, and $\mathcal{H}_k^{(t)}$ is the historical context analogous to (3.4).

After both solver agents produce their outputs, the evaluator agent LLM_E assesses

the joint solution $\hat{\mathbf{y}}^{(t)} = \{\hat{\mathbf{y}}_1^{(t)}, \hat{\mathbf{y}}_2^{(t)}\}$ by evaluating:

$$\mathbf{r}^{(t)} = \big\{R\big(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}}^{(t)})\big),\, \mathbf{f}_j\big(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}}^{(t)})\big)\big\}, \tag{3.22}$$

where $R(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}}^{(t)}))$ is the total sum-rate in (3.18a), and $\mathbf{f}_j(\cdot)$ is a binary vector indicating whether each QoS constraint in (3.18b) is satisfied. The evaluator's prompt is given as follows:

$$\Pi_E^{(t)} = \{\mathbf{r}^{(t)},\, \hat{\mathbf{y}}^{(t)},\, \mathrm{RT}^{(t)},\, \mathcal{H}_E^{(t)}\}, \quad t \geq 0, \tag{3.23}$$

where $\mathrm{RT}^{(t)} = \{\mathrm{RT}_1^{(t)}, \mathrm{RT}_2^{(t)}\}$ contains the reasoning traces from both solver agents. The evaluator generates feedback as in (3.5) which is then parsed into agent-specific feedback for the next iteration through (3.6). Appendix 5.3.4 shows how the prompts $\Pi_k^{(t)}$ and $\Pi_E^{(t)}$, are formed for iteration $t = 0$ and 1 followed by prompt $\Pi_H^{(l)}$.

For the projection process, we follow the work of [32] and employ a constraint violation measure function, V, given in (3.25), shown at the bottom of next page. In (3.25), $s(.)$ denotes the indicator function and is given as follows:

$$s(\mathbf{x}, \hat{\mathbf{y}}^{[i]}) \;=\; \begin{cases} 1, & R_u(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}}^{[i]}))\big) \;\geq\; R_{\min},\; \forall u \in \mathcal{U}, \\ 0, & \text{otherwise}, \end{cases} \tag{3.24}$$

Furthermore, we propose employing momentum-based gradient descent, as shown in (3.26) (given at bottom of next page), where $\mathbf{v}^{[i]}$ and $\hat{\mathbf{y}}^{[i]}$ denote momentum velocity vector and the value of $\hat{\mathbf{y}}$ at i-th iteration of the gradient-descent algorithm, respectively. Moreover, the hyperparameters for the momentum based gradient-descent algorithm, i.e., the learning rate ($\mu^{(l)}$) and momentum coefficient ($\gamma^{(l)}$) are provided by LLM_H for l-th round, by using (3.9). Furthermore, figure (**??**) shows how $\Pi_H^{(t)}$

is formed at round $l = 0$. The term $\nabla_{\hat{\mathbf{y}}} V(\mathbf{x}, \hat{\mathbf{y}}^{[i]})$ represents the gradient of V with respect to $\hat{\mathbf{y}}$, evaluated at $\hat{\mathbf{y}}^{[i]}$. Note that the initial point for the projection process is taken from the combined output of the solver agent(s), i.e., $\hat{\mathbf{y}}^{[0]} = \hat{\mathbf{y}}^{(T)}$ and $\mathbf{v}^{[0]} = \mathbf{0}$.

$$\begin{aligned} V\big(\mathbf{x}, \hat{\mathbf{y}}\big) = \big(1 - s(\mathbf{x}, \hat{\mathbf{y}}^{[i]})\big) \sum_{u=1}^{U} \mathrm{ReLU}\Big(e^{R_{\min} - R_u\left(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}})\right)} - 1\Big) \\ + s(\mathbf{x}, \hat{\mathbf{y}}^{[i]}) \sum_{u=1}^{U} \Big(e^{R_{\min} - R_k\left(\mathbf{x}, \mathcal{T}(\hat{\mathbf{y}})\right)} - 1\Big). \end{aligned} \tag{3.25}$$

$$\begin{aligned} &\hat{\mathbf{y}}^{[i+1]} = \mathrm{proj}\big(\mathbf{x}, \hat{\mathbf{y}}^{[i]}; \mu^{(l)}, \gamma^{(l)}\big) = \hat{\mathbf{y}}^{[i]} - \mu^{(l)} \mathbf{v}^{[i+1]}, \\ \text{where} \quad &\mathbf{v}^{[i+1]} = \gamma^{(l)} \mathbf{v}^{[i]} + \big(1 - \gamma^{(l)}\big) \nabla_{\hat{\mathbf{y}}} V\big(\mathbf{x}, \hat{\mathbf{y}}^{[i]}\big), \quad i = 0, 1, \ldots, I. \end{aligned} \tag{3.26}$$

It is worth mentioning that the proposed LLM4RRM framework can also be used to optimize $\hat{\mathbf{y}}$ using only a single block, i.e., setting $K = 1$.

3.3 Use-Case Study II: Joint User Association and Power Control in Multi-Antenna Multi-User Network

3.3.1 System Model and Problem Formulation

We consider a multi-cell downlink network with B base stations (BSs), U users, and Q orthogonal channels per BS. Each BS has a single transmit antenna and a total power budget $P_{b,\max}$ shared across its channels. We assume $U = BQ$ and each (b, q) pair serves at most one user, and each user can be assigned to at most one (b, q) combination. The bandwidth per channel is BW Hz. We denote the sets of users, BSs, and channels as $\mathcal{U} = \{1, \ldots, U\}$, $\mathcal{B} = \{1, \ldots, B\}$, and $\mathcal{Q} = \{1, \ldots, Q\}$, respectively. Moreover, we assume that the perfect CSI is available at the BS side. When a user u is served by BS b on channel q (i.e., $A_{b,q,u} = 1$), the corresponding

SINR and achievable rate for a user u are given, respectively, as

$$\begin{aligned} \gamma_{b,q,u}(\mathbf{x}, P) &= \frac{|H_{b,q,u}|^2 P_{b,q}}{\sum_{b' \neq b} |H_{b',q,u}|^2 P_{b',q} + \sigma_u^2}, \\ R_{b,q,u}(\mathbf{x}, P) &= BW \log_2\big(1 + \gamma_{b,q,u}(\mathbf{x}, P)\big), \end{aligned} \tag{3.27}$$

where $H_{b,q,u}$ denotes the complex channel q from BS b to user u, $P_{b,q}$ denotes the transmit power allocated by BS b on channel q, and σ_u^2 denotes the power of the noise at the user u. Let $P \in \mathbb{R}^{B \times Q}$ denote the matrix containing all $P_{b,q}$, $H \in \mathbb{C}^{B \times Q \times U}$ represents a tensor containing all $H_{b,q,u}$, and $\mathbf{x} \triangleq \{H, \sigma_u^2\}$. The binary association variables are given as:

$$A_{b,q,u} = \begin{cases} 1, & \text{if user } u \text{ is served by BS } b \text{ on channel } q, \\ 0, & \text{otherwise.} \end{cases} \tag{3.28}$$

Let the tensor $A \in \{0,1\}^{B \times Q \times U}$ denote the collection of all association variables. Then, the joint user-association and power-control (UA–PC) problem is formulated as:

$$\underset{\mathbf{P},\mathbf{A}}{\text{maximize}} \quad R(\mathbf{x},\mathbf{P},\mathbf{A}) = \sum_{u=1}^{U} R_u(\mathbf{x},\mathbf{P},\mathbf{A}), \tag{3.29a}$$

$$= \sum_{u=1}^{U}\sum_{q=1}^{Q}\sum_{b=1}^{B} A_{b,q,u} R_{b,q,u}(\mathbf{x},\mathbf{P}), \tag{3.29b}$$

$$\text{subject to} \quad P_{b,q} \geq 0, \forall b \in \mathcal{B}, \forall q \in \mathcal{Q}, \tag{3.29c}$$

$$\sum_{q=1}^{Q} P_{b,q} \leq P_{b,\max}, \forall b \in \mathcal{B}, \tag{3.29d}$$

$$A_{b,q,u} \in \{0,1\}, \forall b \in \mathcal{B}, \forall q \in \mathcal{Q}, \forall u \in \mathcal{U}, \tag{3.29e}$$

$$\sum_{u=1}^{U} A_{b,q,u} = 1, \forall b \in \mathcal{B}, \forall q \in \mathcal{Q}, \tag{3.29f}$$

$$\sum_{q=1}^{Q}\sum_{b=1}^{B} A_{b,q,u} = 1, \quad \forall u \in \mathcal{U}, \tag{3.29g}$$

$$R_u(\mathbf{x},\mathbf{P},\mathbf{A}) \geq R_{u,\min}, \quad \forall u \in \mathcal{U}. \tag{3.29h}$$

The objective is to maximize the total network sum-rate subject to the QoS constraints. Here, the first constraint, i.e., (3.29c), enforces non-negative power allocations, while the second constraint, i.e., (3.29d) limits the total transmit power of each BS. The third constraint, i.e., (3.29e), specifies the binary nature of the association variables. The fourth and fifth constraints, i.e., (3.29f) and (3.29g), ensure that each BS–channel pair serves exactly one user, and each user is assigned to exactly one BS-channel pair. Finally, the sixth constraint, i.e., (3.29h) guarantees that every user satisfies its minimum rate requirement, denoted by $R_{u,\min}$. The coupling of binary and continuous optimization variables renders the problem a MINLP, which is well known to be computationally intractable to solve within practical time limits. To this end, we first apply constraint transformations, then employ the LLM4RRM accordingly.

3.3.2 Constraint Transformation and Tensor Reshaping

To eliminate the explicit power constraints, we propose a transformation analogous to the beamforming case. Specifically, the power allocation constraints, (3.29c) and (3.29d), can be handled through the following closed-form transformation:

$$P_{b,q} = P_{b,\max} \cdot \mathrm{Sig}(\alpha) \cdot \mathrm{ReLU}\left(\frac{\hat{P}_{b,q}}{\sum_{q=1}^{Q} \hat{P}_{b,q}}\right), \tag{3.30}$$

where $\hat{P}_{b,q} \in \mathbb{R}$ denotes the *unconstrained* power variable for BS b on channel q, and $\alpha \in \mathbb{R}$ is a scalar controlling the total power consumption. To further facilitate efficient LLM-based reasoning, we reshape a three-dimensional tensor, $\mathbf{A} \in \{0,1\}^{B \times Q \times U}$ into a matrix form $\mathbf{A} \in \{0,1\}^{O \times U}$, where $O = B \times Q$ represents the total number of BS–channel pairs. The mapping for this transformation is defined as follows:

$$A_{o,u} = A_{b,q,u} \begin{cases} b = \lfloor \frac{o-1}{Q} \rfloor + 1 \\ q = ((o-1) \bmod Q) + 1 \end{cases}, \tag{3.31}$$

where $o \in \mathcal{O} = \{1, \ldots, O\}$ indexes the flattened BS–channel pairs. Additionally, we relax the binary constraint (3.29e) to enable continuous optimization within the LLM4RRM . Thus, applying the power transformation from (3.30), the mapping from (3.31) and relaxation of binary constraint, the optimization problem in (3.29b) can

be rewritten as follows:

$$\underset{\hat{\mathbf{P}},\alpha,\mathbf{A}}{\text{maximize}} \quad \sum_{u=1}^{U}\sum_{o=1}^{O} A_{o,u} R_{o,u}(\mathbf{x}, \mathcal{T}_P(\hat{\mathbf{P}}, \alpha)) \tag{3.32a}$$

$$\text{subject to} \quad A_{o,u} \in [0,1], \quad \forall o \in \mathcal{O}, \forall u \in \mathcal{U} \tag{3.32b}$$

$$\sum_{u=1}^{U} A_{o,u} = 1, \quad \forall o \in \mathcal{O} \tag{3.32c}$$

$$\sum_{o=1}^{O} A_{o,u} = 1, \quad \forall u \in \mathcal{U} \tag{3.32d}$$

$$R_u(\mathbf{x}, \mathcal{T}_P(\hat{\mathbf{P}}, \alpha), \mathbf{A}) \geq R_{u,\min}, \quad \forall u \in \mathcal{U} \tag{3.32e}$$

where $\mathcal{T}_P(\cdot)$ denotes the transformation, given in (3.30) and it takes the unconstrained power matrix $\hat{\mathbf{P}}$ and scalar α as inputs and returns the constrained power allocation $\mathbf{P}$. Note that in the above optimization problem, A has now been transformed into a matrix, i.e., $A \in [0,1]^{O \times U}$, and it contains a collection of all continuous variables $A_{o,u}$.

3.3.3 Adoption of LLM4RRM

To enable cooperative reasoning, the unconstrained decision vector $\hat{\mathbf{y}} = \{\hat{\mathbf{P}}, \alpha, \mathbf{A}\}$ is partitioned into two blocks, and at any t-th iteration, these two blocks are given as

$$\hat{\mathbf{y}}_1^{(t)} = \{\mathbf{A}^{(t)}\}, \qquad \hat{\mathbf{y}}_2^{(t)} = \{\hat{\mathbf{P}}^{(t)}, \alpha^{(t)}\}, \tag{3.33}$$

It is evident from (3.33) that the solver agent 1 (LLM_1) focuses on user association, generating a continuous association matrix which represents soft assignment preferences, while agent 2 (LLM_2) optimizes the unconstrained power variables $\hat{\mathbf{P}}^{(t)}$ and

$\alpha^{(t)}$. After T reasoning iterations, the projection process to enforce the QoS constraints is applied, but before this process, we first perform an association refinement step to handle the discrete assignment constraints in (3.32b)–(3.32d), specifically, (i) a QoS-driven feasibility mask (eq. 34) penalizes infeasible links, (ii) Sinkhorn normalization (eq. 35) projects the relaxed continuous matrix onto the Birkhoff polytope, and (iii) discretization is performed via greedy assignment (eq. 36) or the Hungarian algorithm (eq. 37), both are gradient-free combinatorial methods operating on the discrete feasible set.

3.3.4 Association Refinement

Given the current power allocation $\hat{\mathbf{P}}^{(T)}$, we apply a QoS-driven feasibility mask to penalize infeasible user–BS–channel links, thereby obtaining the updated continuous association matrix as follows:

$$\mathbf{A}' = \mathbf{A}^{(T)} + \mathbf{L}(\mathbf{x}, \hat{\mathbf{P}}^{(T)}, \alpha^{(T)}), \tag{3.34}$$

where $\mathbf{L} \in \mathbb{R}^{O \times U}$ is the mask matrix with elements:

$$L_{k,u}(\mathbf{x}, \hat{\mathbf{P}}^{(T)}, \alpha^{(T)}) \quad = \quad \begin{cases} 0, & R_{k,u}(\mathbf{x}, \mathcal{T}_P(\hat{\mathbf{P}}^{(T)}, \alpha^{(T)})) \geq R_{u,\min}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{3.35}$$

This masking operation effectively suppresses infeasible links by assigning them negligible weights prior to normalization. The masked matrix $\mathbf{A}'$ is subsequently projected onto the Birkhoff polytope via a truncated Sinkhorn normalization operator [87], $\mathcal{S}(\cdot)$,

implemented through T_s alternating column-row normalizations, as given below:

$$\begin{aligned} \mathcal{S}^{[0]}(\mathbf{A}') &= \exp(\mathbf{A}'), \\ \mathcal{S}^{[i+1]}(\mathbf{A}') &= N^r\big(N^c(\mathcal{S}^{[i]}(\mathbf{A}'))\big), \quad i = 0, \ldots, T_s - 1, \end{aligned} \tag{3.36}$$

where $N^c(\mathbf{A}') = \mathbf{A}' \oslash (\mathbf{A}'\mathbf{1}_U\mathbf{1}_U^\top)$ and $N^r(\mathbf{A}') = \mathbf{A}' \oslash (\mathbf{1}_O\mathbf{1}_O^\top\mathbf{A}')$ denote column and row normalization operators, respectively, with $\mathbf{1}_U \in \mathbb{R}^{U\times 1}$ and $\mathbf{1}_O \in \mathbb{R}^{O\times 1}$ as all-ones vectors and $\oslash$ representing element-wise division. After T_s steps, a continuous doubly-stochastic matrix $\tilde{\mathbf{A}} = \mathcal{S}^{[T_s]}(\mathbf{A}')$ is obtained, which satisfies constraints [87], given in (3.32b) - (3.32e). Now $\tilde{\mathbf{A}}$ can be discretized via the following greedy assignment:

$$\bar{A}_{o,u} = \begin{cases} 1, & \text{if } u = \arg\max_{u'} \tilde{A}_{o,u'}, \\ 0, & \text{otherwise.} \end{cases} \tag{3.37}$$

Although the greedy assignment is computationally efficient and significantly reduces the overall time complexity, it does not guarantee satisfaction of the quota constraint specified in (3.32d). In cases where the greedy assignment violates the quota constraints, the Hungarian algorithm is applied as a fallback, using the matrix $\tilde{\mathbf{A}}$ to ensure a valid solution. Specifically, the optimal binary assignment is obtained as

$$\bar{\mathbf{A}} = \arg\max_{\mathbf{A}^* \in \mathcal{P}} \sum_{o=1}^{O}\sum_{u=1}^{U} A^*_{o,u}\,\tilde{A}_{o,u}, \tag{3.38}$$

where $\mathbf{A}^*$ denotes a candidate binary assignment matrix, and the set of all feasible binary assignment matrices that satisfy the quota constraints is defined by

$\mathcal{P} = \left\{ \mathbf{A} \in \{0,1\}^{O \times U} \;\middle|\; \mathbf{A}\mathbf{1}_U = \mathbf{1}_O,\ \mathbf{A}^\top \mathbf{1}_O = \mathbf{1}_U \right\}$. Although one may apply the Hungarian algorithm directly, omitting the Sinkhorn and/or greedy refinement steps, doing so leads to a considerably higher computational cost. The final binary association matrix, obtained either from (3.37) or (3.38), is denoted by $\bar{\mathbf{A}}$.

3.3.5 Power Control Projection

Although both the greedy assignment and the Hungarian algorithm yield a *binary* doubly-stochastic matrix, the QoS constraints in 3.32e may no longer remain satisfied after discretization. To overcome this issue, we apply an iterative projection to refine variables $\hat{\mathbf{P}}$ and α using a constraint violation function, V, as defined in (3.25), with $\hat{\mathbf{y}}$ initialized as $\hat{\mathbf{y}}^{[0]} = \{\bar{\mathbf{A}}, \hat{\mathbf{P}}^{(T)}, \alpha^{(T)}\}$ and the transformation $\mathcal{T}_l(\cdot)$ in (3.25) is replaced by $\mathcal{T}_P(\cdot)$, which is given in (3.30). The hyperparameters $\mu^{(l)}$ and $\gamma^{(l)}$ are provided by LLM_H for round l, and the projection process continues until all QoS constraints in (3.32e) are satisfied (or maximum number of rounds reached) while simultaneously maximizing the achievable sum rate.

3.4 Numerical Results and Discussions

We now present and analyze the performance of the proposed LLM4RRM for both use-cases. Throughout the numerical results, the following notations are adopted. The average sum rate before and after projection is denoted by $\bar{R}_{\text{pre}}$ and $\bar{R}_{\text{post}}$, respectively. The QoS violation percentages before and after projection are represented by $\mathcal{V}_{\text{pre}}$ and $\mathcal{V}_{\text{post}}$, respectively. Moreover, the average number of projection steps corresponding to (3.12) required to achieve zero constraint violation is denoted by S_{ZV}, while the average number of projection steps required for convergence is given by

S_{conv}. Furthermore, the corresponding time to convergence of the entire optimization method is denoted by T_{conv}. Note that S_{ZV} is computed by averaging only over those instances for which the constraint violation has been successfully reduced to zero. In the numerical tables, bold entries indicate the best performance, while underlined entries denote the second-best results.

3.4.1 Results: Joint Beamforming and Phase Shift Optimization

3.4.1.1 Experimental Setup

We consider a single cell STAR RIS-assisted Multiple-Input Single-Output (MISO) downlink system comprising one BS, equipped with $M = 8$ antennas, and serving $U = 4$ single antenna users. The BS is positioned 200 meters from the STAR RIS, which is centrally located in the coverage area. Users are uniformly and randomly distributed within a 50-meter radius circle around the STAR RIS. The number of STAR RIS elements is varied as $N \in \{16, 32, 64\}$. The minimum rate requirement is set as $R_{\min} \in \{3, 5, 17.5, 20\}$ Mbps, while the transmit power constraint is chosen as $P_{\max} \in \{1, 1.5, 2.0, 5.0\}$. The noise power is fixed at -170 dBm. Path loss for all channels is calculated as $L(d) = (35.6 + 22.0 \log_{10}(d))$, where d denotes the distance in meters. The Rician factor is set to $\epsilon = 10$, and all channels follow the line of sight model as described in [32].

For each system parameter configuration, we generate 100 independent random channel realizations and report all performance metrics averaged over these realizations. The optimization is implemented using low-latency small-scale LLMs. In particular, our experiments employ GPT-5 Mini, GPT-4.1 Mini, Grok-4 Fast, Gemini-2.5 Flash, and Llama-4 Scout through their respective Application Programming Interfaces (APIs), with the temperature fixed at 1. Consistent prompt templates are used

Table 3.1: Table 3.1(a) shows varying QoS and power constraints with fixed $N = 16$; Table 3.1(b) shows scalability with different RIS element counts and with fixed $(R_{\min}, P_{\max}) = (3.0, 1.0)$.

System Constraints $(R_{\min}, P_{\max})$	Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
(3.0, 1)	LLM4RRM ($K = 1$)	10.4835	75.7081	47.14	0	3.3	31
	LLM4RRM	9.4345	75.1625	49	0	3.5	37.4
	Neural Network	8.972	74.5825	33	0	4.2	59.2
	Random Init	9.6605	71.1281	100	0	18	155.6
	GA	–	73.1894	–	–	–	–
(5.0, 1.5)	LLM4RRM ($K = 1$)	16.207	90.2583	69	0	8.8	33.6
	LLM4RRM	19.3355	91.034	61	0	7.6	32.4
	Neural Network	12.612	81.5907	89	5.6	10.9	49.2
	Random Init	10.0145	75.0613	100	11.6	40.3	196
	GA	–	88.5554	–	–	–	–
(17.5, 2.0)	LLM4RRM ($K = 1$)	60.406	105.1776	83	0	12.3	52.9
	LLM4RRM	65.697	106.0192	78	0	11	49.5
	Neural Network	49.9465	92.3363	100	25.4	23.4	55.7
	Random Init	29.2595	73.1088	100	59	81.2	174
	GA	–	101.8237	–	–	–	–
(20.0, 5.0)	LLM4RRM ($K = 1$)	69.6985	117.9326	89	0	15.9	68.7
	LLM4RRM	73.8905	118.3001	81	0	13.7	69.1
	Neural Network	57.5595	95.3631	100	31.2	29.6	81
	Random Init	33.216	71.5547	100	65.7	123.3	196.5
	GA	–	112.9114	–	–	–	–

RIS Elements N	Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}	T_{conv}
16	LLM4RRM ($K = 1$)	10.4835	75.7081	47.14	0	3.3	31	108 s
	LLM4RRM	9.4345	75.1625	49	0	3.6	37.4	89 s
	Neural Network	8.972	74.5825	33	0	4.2	59.2	3.97 h
	GA	—	73.1894	—	0	—	—	67 s
	Random Init	9.6605	71.1281	100	0	18	155.6	3.9 s
32	LLM4RRM ($K = 1$)	10.6435	101.2061	81	0	4.8	39.3	110 s
	LLM4RRM	9.674	101.8643	65	0	5.5	35.1	91 s
	Neural Network	9.476	100.4067	43	0	4.9	44	4.87 h
	GA	—	92.3444	—	0	—	—	134 s
	Random Init	9.121	79.7949	100	56	46.6	187	N-A
64	LLM4RRM ($K = 1$)	9.1135	126.3382	83	0	6.1	67.3	119 s
	LLM4RRM	11.3315	127.7738	76	0	5.7	47.2	95 s
	Neural Network	9.916	126.1286	45	0	7.4	74.3	5.48 h
	GA	—	112.3784	—	0	—	—	155 s
	Random Init	9.7835	24.8009	100	74.76	79.7	200	N-A

across all LLMs, and no task-specific fine-tuning or supervised training is performed. We set $T = 5$ for all simulations unless stated otherwise because LLM inference rounds yield the best performance when $T = 5$, as shown in Table 3.8. Similarly, three few-shot examples are provided per instance, selected from the database $\mathcal{D}$ and

constructed using the Frobenius norm as the similarity function. This choice is motivated by its strongest retrieval performance among all tested similarity metrics, as demonstrated in Table 3.9. Furthermore, since the adaptive projection mechanism consistently converges within $L_{\text{conv}} = 1$ to 2 rounds on average across all tested configurations, as reported in Table 3.10, we conservatively set the maximum number of projection rounds to $L = 5$ and the maximum number of projection steps to $I = 200$. Furthermore, GPT 5 Mini is used as an LLM model in all results, unless mentioned otherwise.

3.4.1.2 Baselines

We compare the proposed LLM4RRM against the following three baseline methods:

Deep Unsupervised Learning Framework This baseline implements the deep unsupervised learning (DUL) framework proposed in [32] for STAR RIS-assisted networks. This method employs two DNN models, i.e., WNet and PNet, that are trained using 90000 *feasible* channel instances to generate an unconstrained beamforming matrix and a phase shift vector, which are subsequently projected onto the feasible set using a differentiable projection layer.

Genetic Algorithm (GA) This baseline applies a standard GA with a population of 500 generations, directly optimizing the decision variables $\mathbf{y}$, without requiring projection.

Random Initialization This baseline follows the same structure as the DUL framework, except that the unconstrained beamforming matrix and phase-shift vector are

generated randomly instead of being computed using trained DNN models.

3.4.1.3 Results for Varying QoS and Power Constraints

Table 3.1 (a) compares the LLM4RRM framwork with the baselines under different $(R_{\min}, P_{\max})$ settings with RIS Elements set as $N = 16$. Here the DUL framework has been trained with $R_{\min} = 3.0$ Mbps and $P_{\max} = 1$W.

Table 3.1(a) shows that the LLM4RRM consistently achieves substantially higher pre- and post-projection sum rates across all scenarios when compared with the baseline schemes, while always reducing post-projection violations to zero. Although the pre-projection outputs of the LLM4RRM may exhibit constraint violations, for example, up to 89% at the operating point $(20.0, 5.0)$, these candidate solutions remain close to the feasible region. As a result, the projection procedure requires only a small number of corrective updates. This proximity leads to rapid convergence, typically requiring between 3 and 15 steps to restore feasibility and between 31 and 69 steps to reach full convergence, which is significantly faster than the DUL and random baselines.

By contrast, the DUL and random baselines yield poor initial solutions, with the random method performing the worst by producing the lowest post-projection sum rates and requiring the greatest number of projection steps for both violation reduction and convergence. These two baselines exhibit 100% pre-projection violations for some cases and generally struggle to eliminate constraint violations entirely, even after the projection process. This is due to the fact that the initial points generated by both baseline schemes are generally far from the feasibility region. On the other hand, although GA does not exhibit constraint violations, it generally achieves lower

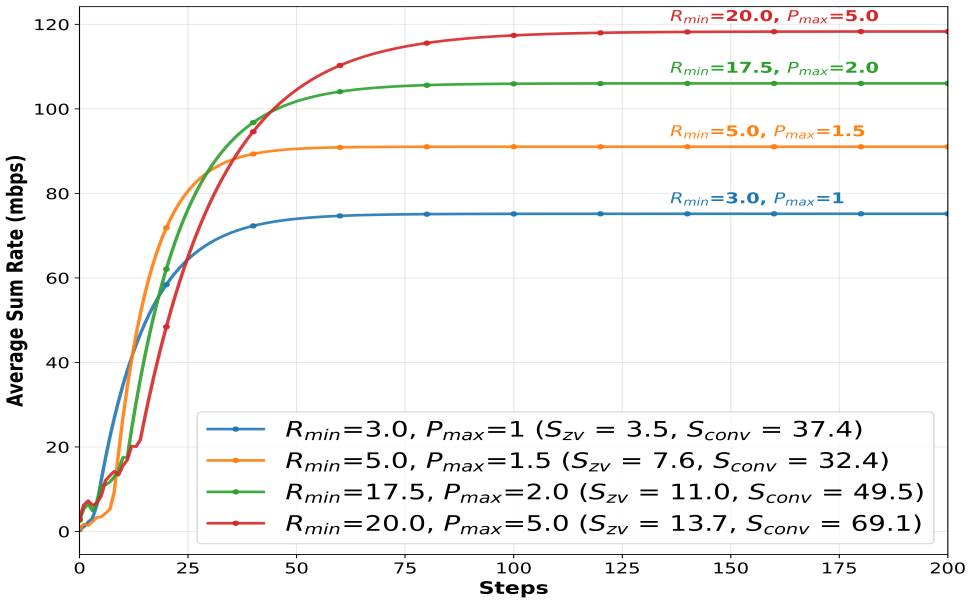
sum-rates compared to LLM4RRM. Moreover, this performance gap becomes more pronounced as the QoS constraints become more stringent.

3.4.1.4 Results for Varying Number of RIS Elements

Table 3.1(b) shows the effect of increasing RIS size from 16 to 64 elements with fixed $(R_{\min}, P_{\max}) = (3.0, 1.0)$. The LLM4RRM effectively scales with RIS size, maintaining zero violations after projection and consistently achieving the highest post-projection sum rates. The DUL baseline, while competitive in terms of sum rate, suffers from poor scalability. Specifically, its feasible dataset generation time and training time increase sharply, rising from approximately 3 hours for $N = 16$ to more than 5 hours for $N = 64$. Moreover, each configuration requires generating a new feasible dataset, which is a time-consuming process. In contrast, the random method exhibits a 100% pre-projection violation rate, and its sum rate deteriorates significantly for larger values of N. For example, when $N = 64$, the achieved sum rate drops to 24.8009 Mbps.

The GA achieves competitive sum rates for smaller RIS sizes but falls short of LLM4RRM as N increases, since its population-based search scales poorly with the growing dimensionality of the decision vector. Moreover, its runtime increases from 67 seconds at $N = 16$ to 155 seconds (surpasses the runtime of LLM4RRM) at $N = 64$, while still underperforming LLM4RRM in sum-rate across all configurations. In contrast, the LLM-based approaches are training-free and system size agnostic, thus are scalable. Moreover, Fig. 3.2 visualizes the post-projection convergence behaviour of LLM4RRM for the parameter settings in Table 3.1. For the most complex configuration ($N = 64$, $U = 4$, $T = 5$), the final prompt contained approximately 8,500–11,000

(a) Average Sumrate vs Projection Steps across different $(R_{\min}, P_{\max})$ settings $(N = 16)$.



(b) Average Sumrate vs Projection Steps across different numbers of RIS elements $(R_{\min} = 3,\ P_{\max} = 1)$.

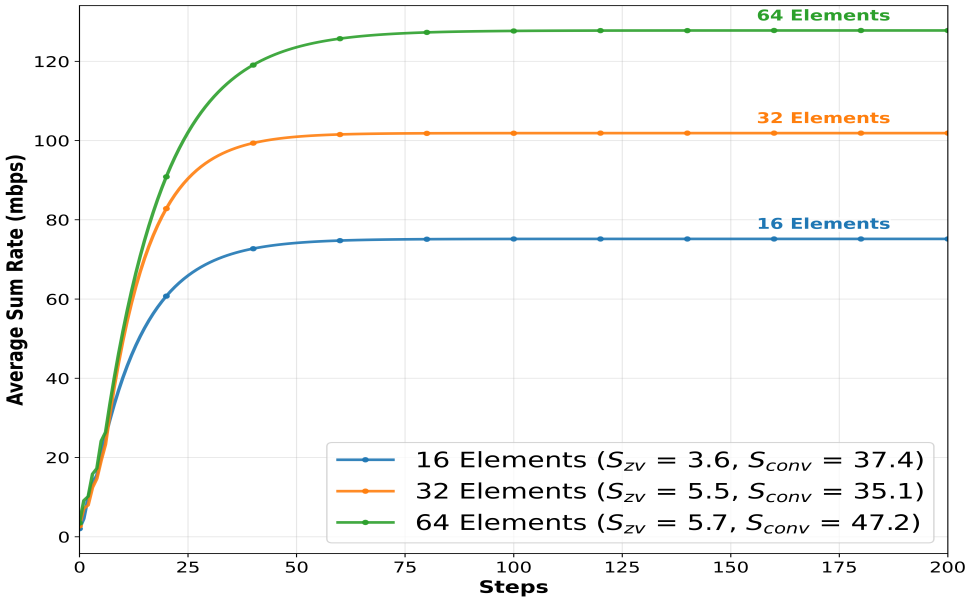


Figure 3.2: Post-projection sum rate vs. projection steps for LLM4RRM (Use-Case Study I), illustrating convergence across diverse network scenarios.

tokens across all solver and evaluator agents. This remains well within the context windows of modern LLMs (400k–1M tokens for GPT-based and other frontier models)

Table 3.3: Effect of varying No. of few-shot examples (for $R_{\min} = 5$ Mbps, $P_{\max} = 1.5$ W, $N = 16$) for Case-study-I.

Examples	Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
1	LLM4RRM ($K = 1$)	9.36	82.18	83	0	9.7	112.3
	LLM4RRM	9.00	80.25	76	0	5.0	88.9
3	LLM4RRM ($K = 1$)	17.34	90.26	53.9	0	3.7	37.8
	LLM4RRM	18.21	91.03	45	0	3.6	37.4
5	LLM4RRM ($K = 1$)	17.60	89.89	59	0	3.7	40.3
	LLM4RRM	16.00	87.02	44	0	3.7	42.1
7	LLM4RRM ($K = 1$)	12.30	82.45	53	0	5.5	88.0
	LLM4RRM	11.87	84.90	59	0	4.1	61.8

3.4.1.5 Impact of the Number of Few-Shot Examples

Table 3.3 shows the effect of varying the number of few-shot examples provided to the solver LLMs for a fixed configuration with 16 RIS elements, $R_{\min} = 5$ Mbps, and $P_{\max} = 1.5$W. Both variations of the LLM4RRM exhibit clear performance gains as the number of few-shot examples increases from one to three. In this range, the models achieve higher sum rates and require fewer feasibility correction steps, typically around three to four steps. Beyond this point, however, the benefit diminishes. With five examples, the improvement saturates, and with seven examples, performance degrades as the prompts become cluttered and less targeted. The sum rate follows the same pattern, as it peaks when three to five few-shot examples are provided and decreases when the number of examples is either too small (insufficient context) or too large (excessive or distracting information). These observations underscore the importance of prompt calibration. A well-chosen number of labelled examples accelerates convergence and improves solution quality, whereas overly long or unfocused

prompts hinder optimization efficiency.

3.4.1.6 UQ and Trustworthiness

As discussed in Section 3.1.6, our uncertainty analysis comprises three components: *input-based, prompt-based, and output-based uncertainty*. Table 3.1 demonstrated input-based robustness of the LLM4RRM . Table 3.4 and 3.5 evaluate the other two uncertainties by comparing robustness across prompt designs and by measuring dispersion across repeated outputs, respectively. For the analysis of prompt design variations, we define seven prompt styles: *Prompt 1* employs a simple direct instruction with constraints; *Prompt 2* incorporates few-shot examples; *Prompt 3* introduces physics aware formulations; *Prompt 4* adds confidence statements; *Prompt 5* adopts structured chain-of-thought reasoning; *Prompt 6* decomposes the task into multiple solver agents for modular optimization (although it is multiple LLM models and each model will be given its own separate prompt but for convention we can say that this breakdown of the task is also a variation of the prompt); and *Prompt 7* augments this decomposition with evaluator-guided feedback for refinement.

Table 3.4: Prompt-based UQ, (for $R_{\min} = 20$ Mbps, $P_{\max} = 5$ W, $N = 16$) for Case-study-I.

Prompt	$\bar{R}_{\text{post}}$ (Mbps)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
Prompt 1	76.2364	60.3	112.1	153.1
Prompt 2	85.3546	21.3	29.6	85.3
Prompt 3	90.5557	0	25.3	74.2
Prompt 4	92.5673	0	22.1	73.8
Prompt 5	98.2389	0	19.5	65.1
Prompt 6	103.3001	0	16.4	64.4
Prompt 7	118.3001	0	13.7	61.4

Table 3.4 shows gains as prompt richness increases. *Prompt 1* achieves only 76.23 Mbps with 60.3% violations and over 150 projection steps, reflecting high uncertainty. The demonstrations and mathematical structure in *Prompts 2 and 3* reduces

Table 3.5: Output-based UQ ($R_{\min} = 20$ Mbps, $P_{\max} = 5$ W, $N = 16$) for Case-study I.

Prompt	W_r	W_i	a	α	ϕ_r	ϕ_t
Prompt 1	2.9866	2.2899	2.7769	1.036	2.2878	2.5181
Prompt 2	1.0456	1.2884	1.8974	0.564	0.7480	0.8360
Prompt 3	0.5866	0.3864	0.7223	0.157	0.6230	0.5890
Prompt 4	0.4365	0.3479	0.6112	0.103	0.5970	0.5810
Prompt 5	0.3023	0.3335	0.1632	0.089	0.1460	0.0946
Prompt 6	0.1567	0.1899	0.1347	0.027	0.0899	0.0946
Prompt 7	0.0173	0.0262	0.0483	0.007	0.0697	0.0474

violations and achieves convergence in lesser steps. More structured reasoning and multi-agent decomposition in *Prompts 4–6* further stabilizes performance, reaching 103.30 Mbps with zero violations and achieving convergence in approximately 60–70 steps. Evaluator-guided refinement (*Prompt 7*) achieves the best performance, i.e., 118.30 Mbps, zero violations, and the fastest convergence ($S_{\text{ZV}} = 13.7$).

Table 3.5 presents output-based uncertainty using the pairwise L2-norm dispersion metric, as defined in (3.13) with $\Psi = 20$. For each prompt-input pair, multiple outputs are generated by running the LLM-based method 10 times ($\Lambda = 10$ in (3.13)), and their average pairwise distances are computed to quantify stability. Early prompt categories exhibit large dispersion values, e.g., dispersion values for W_r and ϕ_t are 2.9866 and 2.5181, respectively, for Prompt 1, indicating substantial stochastic variability. As the prompt structure becomes richer, dispersion decreases monotonically. For instance, a dispersion value for α reduces from 1.036 (Prompt 1) to 0.027

Table 3.6: LLM4RRM backbone comparison ($R_{\min} = 3$ Mbps, $P_{\max} = 1$, $N = 16$) for Case-study-I.

Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
GPT 5 Mini	9.43	75.16	49	0	3.5	37.4
GPT 4.1 Mini	9.32	74.89	53.9	0	3.7	39.8
Grok 4 Fast	8.71	74.63	60.1	0	4.1	42.1
Gemini 2.5 Flash	8.68	74.51	65.4	0	3.9	58.6
Llama 4 Scout	8.60	73.04	63.4	0	4.0	64.8

(Prompt 6). It is evident that Prompt 7 yields the smallest dispersion value for all variables, reflecting the highest degree of output consistency and lowest stochastic variability.

Table 3.7: Performance of LLM-based joint UA and Power PC optimization. Table 3.7(a) compares varying QoS and power constraints, while Table 3.7(b) analyzes scalability across different network configurations.

(a) Varying QoS and power constraints with $(B, Q, U) = (3, 2, 6)$

$(R_{u,\min}, P_{b,\max})$	Method	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{post}}$ (%)	S_{conv}
(2.5, 1.0)	LLM4RRM ($K = 1$)	29.7	0	16.1
	LLM4RRM	<u>27.9</u>	0	17.2
	JUPNET Neural Network	24.3	0	<u>16.3</u>
	GA	26.8	–	–
(5.0, 1.5)	LLM4RRM ($K = 1$)	<u>40.8</u>	0	<u>18.9</u>
	LLM4RRM	41.7	0	18.1
	JUPNET Neural Network	30.3	18.2	28.6
	GA	38.9	–	–
(7.5, 2.0)	LLM4RRM ($K = 1$)	<u>53.1</u>	0	18.8
	LLM4RRM	56.55	0	<u>19.1</u>
	JUPNET Neural Network	38.2	25.1	32.3
	GA	51.2	–	–
(12.5, 3.0)	LLM4RRM ($K = 1$)	<u>91.76</u>	0	<u>19.8</u>
	LLM4RRM	92.21	0	18.7
	JUPNET Neural Network	48.8	69.4	45.4
	GA	87.5	–	–

(b) Scalability across network configurations with $(R_{u,\min}, P_{b,\max}) = (2.5, 1.0)$

Configuration (B, Q, U)	Method	$\bar{R}_{\text{post}}$	S_{conv}	T_{conv}
(3, 2, 6)	LLM4RRM ($K = 1$)	29.7	16.1	19.2 s
	LLM4RRM	<u>27.9</u>	<u>17.2</u>	<u>23.07 s</u>
	JUPNET Neural Network	24.3	16.3	2.3 h
	GA	26.8	–	28 s
(4, 2, 8)	LLM4RRM ($K = 1$)	<u>67.2</u>	19.1	<u>25.322 s</u>
	LLM4RRM	68.1	<u>20.2</u>	24.105 s
	JUPNET Neural Network	65.9	19.5	2.7 h
	GA	63.5	–	38 s
(5, 2, 10)	LLM4RRM ($K = 1$)	<u>79.5</u>	<u>24.1</u>	<u>24.347 s</u>
	LLM4RRM	81.5	25.3	23.97 s
	JUPNET Neural Network	75.2	26.7	2.8 h
	GA	72.1	–	47 s
(6, 2, 12)	LLM4RRM ($K = 1$)	<u>95.8</u>	<u>30.5</u>	<u>25.33 s</u>
	LLM4RRM	96.1	30.2	25.212 s
	JUPNET Neural Network	94.7	31.1	3.1 h
	GA	89.3	–	59 s

3.4.1.7 Comparison Across Different LLM Models

Considering the proposed prompt design, Table 3.6 evaluates several state-of-the-art LLM backbones, i.e., GPT 5 Mini, GPT 4.1 Mini, Grok 4 Fast, Gemini 2.5 Flash, and Llama 4 Scout. GPT-based models clearly outperform other models. GPT 5 Mini consistently delivers the highest sum rates with the fewest steps to feasibility (3.5 steps) and the fastest convergence (37.4 steps). These results demonstrate consistent performance trends across different LLM backbones, with comparable performance gaps under the proposed prompt design, indicating that the observed trends are robust to architectural and training differences. Also, while projection guarantees feasibility for all models, the pre-projection solution quality strongly affects convergence speed and final performance.

Table 3.8: Effect of varying the number of LLM inference rounds T on optimization performance (Use-Case Study I, $R_{\min} = 5.0$ Mbps, $P_{\max} = 1.5$ W, $N = 16$).

T	Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
3	LLM4RRM ($K = 1$)	13.82	89.47	74	0	10.2	41.3
	LLM4RRM	**14.91**	**89.82**	**71**	0	**9.4**	**38.7**
5	LLM4RRM ($K = 1$)	16.21	90.26	69	0	8.8	33.6
	LLM4RRM	**19.34**	**91.03**	**61**	0	**7.6**	**32.4**
7	LLM4RRM ($K = 1$)	15.87	89.98	71	0	9.1	35.2
	LLM4RRM	**18.76**	**90.71**	**64**	0	**8.2**	**34.8**
9	LLM4RRM ($K = 1$)	14.93	89.35	75	0	10.5	38.9
	LLM4RRM	**17.82**	**90.21**	**68**	0	**9.3**	**37.6**

3.4.1.8 Effect of Number of LLM Inference Rounds (T)

Table 3.8 presents an ablation over the number of LLM inference iterations $T \in \{3, 5, 7, 9\}$ for Use-Case Study I under fixed system parameters ($R_{\min} = 5.0$ Mbps, $P_{\max} = 1.5$ W, $N = 16$). Performance improves substantially from $T = 3$ to $T = 5$, as the additional evaluator-guided refinement cycles allow the solver agents to address

constraint violations and improve solution quality. Beyond $T = 5$, performance saturates and degrades slightly for $T = 7$ and $T = 9$, likely due to context accumulation and diminishing quality of feedback when the solution is already near-feasible. Based on these results, $T = 5$ is adopted as the default throughout the main manuscript, providing the best trade-off between solution quality and computational overhead.

3.4.1.9 Sensitivity Analysis of Few-Shot Similarity Metrics

The few-shot examples presented to each solver agent are retrieved from the database $\mathcal{D}$ using a similarity function that measures the relevance of stored examples to the current channel realization. Table 3.9 compares four retrieval strategies: Frobenius norm, Cosine similarity, Manhattan distance, and random selection. This analysis is conducted for Use-Case Study I under a fixed system configuration ($R_{\min} = 5.0$ Mbps, $P_{\max} = 1.5$ W, $N = 16$).

Frobenius norm and Cosine similarity achieve comparable and superior performance relative to Manhattan distance and random selection. Random selection is the weakest strategy, yielding up to 5% lower post-projection sum rate and approximately 25% more steps to convergence, confirming that channel-aware similarity retrieval is beneficial for guiding LLM reasoning. The Frobenius norm is adopted as the default throughout the main manuscript due to its strong performance and computational simplicity, though the framework can readily accommodate any similarity metric.

3.4.1.10 Effect of Number of Projection Rounds (L)

Table 3.10 reports the average number of projection rounds actually used by the hyperparameter-tuning agent LLM_H, denoted L_{conv}, across different QoS and power

Table 3.9: Sensitivity of framework performance to different few-shot example similarity metrics (Use-Case Study I, $R_{\min} = 5.0$ Mbps, $P_{\max} = 1.5$ W, $N = 16$). Bold entries indicate best; underlined indicate second-best.

Metric	Method	$\bar{R}_{\text{pre}}$	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{pre}}$ (%)	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}
Frobenius	LLM4RRM ($K = 1$)	16.21	<u>90.26</u>	69	0	8.8	33.6
	LLM4RRM	<u>19.34</u>	**91.03**	**61**	0	**7.6**	<u>32.4</u>
Cosine	LLM4RRM ($K = 1$)	16.15	90.18	70	0	8.8	**31.8**
	LLM4RRM	**19.89**	90.95	<u>62</u>	0	<u>7.7</u>	32.6
Manhattan	LLM4RRM ($K = 1$)	15.23	88.95	74	0	9.6	36.5
	LLM4RRM	18.11	89.63	68	0	8.9	35.7
Random	LLM4RRM ($K = 1$)	13.45	85.72	81	0	11.5	42.3
	LLM4RRM	14.87	86.94	76	0	10.8	40.1

constraint settings in Use-Case Study I. Here, L_{conv} represents the average number of times LLM_H is invoked per channel instance, i.e., the number of rounds required before all projection constraints are satisfied. The maximum allowed number of rounds is fixed at $L = 5$.

Results show that the framework consistently uses only $L_{\text{conv}} \in \{1.2, 2.2\}$ rounds on average for $K = 1$ and $K = 2$ respectively, across all tested configurations, demonstrating that the adaptive projection mechanism converges rapidly and does not require manual tuning of the projection function. The maximum budget of $L = 5$ is thus a conservative and safe upper bound that does not need to be adjusted on a per-problem basis. Tighter QoS constraints (higher $R_{\min}$) require slightly more rounds, but the overhead remains modest throughout.

3.4.2 Results: Joint User Association and Power Control Optimization

3.4.2.1 Experimental Setup

We evaluate a multi-cell downlink network with $B \in \{3, 4, 5, 6\}$ base stations, each providing $Q = 2$ orthogonal channels and serving $U = B \times Q$ single antenna users.

Table 3.10: Average number of adaptive projection rounds L_{conv} used across different QoS and power constraints (Use-Case Study I, $N = 16$, $L = 5$, $T = 5$). Bold entries indicate best performance; underlined entries indicate second-best.

$(R_{\min}, P_{\max})$	Method	$\bar{R}_{\text{post}}$	$\mathcal{V}_{\text{post}}$ (%)	S_{ZV}	S_{conv}	L_{conv}
(3.0, 1.0)	LLM4RRM ($K = 1$)	**75.71**	0	**3.3**	**31.0**	**1.2**
	LLM4RRM	<u>75.16</u>	0	<u>3.5</u>	<u>37.4</u>	<u>1.3</u>
(5.0, 1.5)	LLM4RRM ($K = 1$)	<u>90.26</u>	0	<u>8.8</u>	<u>33.6</u>	<u>1.8</u>
	LLM4RRM	**91.03**	0	**7.6**	**32.4**	**1.9**
(17.5, 2.0)	LLM4RRM ($K = 1$)	<u>105.18</u>	0	<u>12.3</u>	<u>52.9</u>	<u>2.1</u>
	LLM4RRM	**106.02**	0	**11.0**	**49.5**	**2.2**
(20.0, 5.0)	LLM4RRM ($K = 1$)	<u>117.93</u>	0	<u>15.9</u>	<u>68.7</u>	<u>2.0</u>
	LLM4RRM	**118.30**	0	**13.7**	**69.1**	**2.1**

Base stations are placed within an $500 \times 500\,\text{m}$ area, and users are uniformly distributed. Each BS has a total power budget $P_{b,\max} \in \{1.0, 1.5, 2.0, 3.0\}$ W shared across its two channels. The system bandwidth is $BW = 5\,\text{MHz}$ and the noise power is -170 dBm. The per user QoS requirement is $R_{u,\min} \in \{2.5, 5.0, 7.5, 12.5\}$ Mbps. Channels follow distance based path loss $L(d) = d^{\gamma}$ with exponent $\gamma = 3.7$, log normal shadowing with standard deviation $\sigma_s = 2.0\,\text{dB}$, and Rayleigh fading (unit scale). For each setting, we generate 100 independent channel realizations using a feasibility-aware dataset generator and report averages across these realizations. The experiments are conducted using the same LLM models and settings as described previously in Section 3.4.1.

3.4.2.2 Baselines

To benchmark the performance of our proposed LLM4RRM, we consider two baselines: (i) *Joint UA+PC using Neural Netowrks (JUPNet) [87].* This baseline integrates two explicitly projected subnetworks, the Deep Explicit User Association Network (DEUNet) and the Deep Explicit Projection Network (DEPNet). During

training, the model alternates between DEUNet and DEPNet, where DEUNet applies Sinkhorn normalization to enforce user associations, and DEPNet performs differentiable projection for power control to satisfy QoS constraints. At inference, however, JUPNet executes a single forward pass without iterative alternation, directly producing both association and power allocations. Training is performed with 80,000 feasible examples. (ii) GA is employed as a second baseline with a population of 500 generations.

3.4.2.3 Results for Varying QoS Constraints

Table 3.7(a) compares LLM4RRM with the baseline for varying $(R_{u,\min}, P_{b,\max})$ constraints. The JUPNET has been trained for $(B, Q, U) = (3, 2, 6)$, $R_{u,\min} = 2.5$ Mbps and $P_{b,\max} = 1.0$W. The proposed LLM4RRM achieves considerably higher sum-rates for all cases in comparison to the baseline. Moreover, JUPNET fails to achieve zero constraint violations, especially for data points that were not included in its training set. For instance, for $(R_{u,\min}, P_{b,\max}) = (12.5, 3.0)$, the baseline exhibits a post-projection violation rate of 69.4%, whereas the LLM4RRM reduce the violation to zero. These findings demonstrate that the proposed LLM-based approach can adjust to new QoS conditions without any retraining, whereas JUPNet is constrained by its fixed training distribution and cannot reliably generalize beyond it. Convergence trends further support this advantage, as the LLM4RRM stabilizes in approximately 16–20 steps for considered constraint settings, while JUPNET requires 16–45 steps. Although GA adapts to new $(R_{u,\min}, P_{b,\max})$ values, it yields lower sum-rates than LLM4RRM across all settings, as it struggles to handle the coupled discrete and continuous variables.

3.4.2.4 Results for Varying Network Configurations

Table 3.7(b) evaluates the impact of network size by varying the number of users and BSs, i.e., $B \in \{3, 4, 5, 6\}$. The LLM4RRM sustains high performance while converging efficiently across all configurations. In the smallest configuration ($B = 3$, $U = 6$), LLM4RRM achieves a sum-rate of $27.9 - 29.7$ Mbps, within approximately 23 seconds, whereas JUPNet attains only 24.3 Mbps and requires more than two hours for training and feasible dataset generation. This gap in time complexity further widens as the network scales. That is, the sum-rates for the LLM4RRM converge in approximately 25 seconds, whereas JUPNET achieves slightly lower sum-rate and requires over 3 hours of retraining. The GA achieves competitive sum-rates for smaller configurations, but falls behind LLM4RRM as the network grows. Moreover, its runtime also scales unfavorably, increasing from 28 seconds at ($B = 3$) to 59 seconds at ($B = 6$), while still underperforming LLM4RRM in sum-rate across all configurations.

Table 3.11: Multi-Agent LLM backbone comparison for joint UA-PC optimization ($B = 3$, $Q = 2$, $U = 6$, $R_{\min} = 2.5$ Mbps, $P_{\max} = 1.0$ W).

Method	$\bar{R}_{\text{post}}$ (Mbps)	S_{conv}	T_{conv} (s)
GPT 5 Mini	27.98	16.1	23.07
GPT 4.1 Mini	<u>26.44</u>	<u>16.2</u>	<u>24.89</u>
Gemini 2.5 Flash	25.53	17.1	25.1
Grok 4 Fast	25.24	18.1	26.23
Llama 4 Scout	22.31	18.9	25.4

* ***Bold*** *= best,* <u>*underlined*</u> *= second-best.*

3.4.2.5 Comparison Across Different LLM Models

Table 3.11 analyzes the effect of model choice on UA+PC optimization with $R_{\min} = 2.5$ Mbps, $B = 3$, $Q = 2$, and $U = 6$. For both LLM frameworks, yet again, GPT-based models consistently deliver the best performance. More specifically, GPT 5

Mini delivers the strongest overall performance, achieving 29.8 Mbps in the Feedback-driven mode and 27.98 Mbps in the Multi-Agent mode, while also requiring the smallest number of steps and lowest time to reach convergence. GPT 4.1 Mini ranks second, achieving slightly lower sum-rates with slightly higher latencies. Gemini 2.5 Flash and Grok 4 Fast fall in the mid-tier category, producing similar sum rates, i.e., around 25–25.5 Mbps with modestly higher time for convergence. Llama 4 Scout consistently trails behind in terms of the achievable sum-rates. This analysis again shows that the solution quality is dependent upon the choice of the LLM model.

Table 3.12: UQ for Use-Case Study-II ($B = 3$, $Q = 2$, $U = 6$, $R_{u,\min} = 12.5$ Mbps, $P_{b,\max} = 3.0$ W).

(a) Prompt-based Uncertainty				(b) Output-based Uncertainty		
Prompt	$\bar{R}_{\text{post}}$ (Mbps)	$\mathcal{V}_{\text{post}}$ (%)	S_{conv}	Prompt	UA Cons. Rate	Power MAD
1	51.24	47.2	153.2	1	0.13	1.78
2	60.46	10.5	33.2	2	0.56	0.55
3	75.11	0	25.2	3	0.77	0.49
4	77.15	0	22.7	4	0.79	0.41
5	83.47	0	22.4	5	0.84	0.16
6	<u>85.11</u>	0	<u>21.5</u>	6	<u>0.88</u>	<u>0.089</u>
7	92.21	0	18.7	7	0.97	0.033

3.4.2.6 UQ and Trustworthiness

Similar to the case study I, we assess prompt-based and output-based uncertainty for the case study II in Tables 3.12(a) and 3.12(b). Since input-based robustness has already been demonstrated in earlier results (Table 3.7), we now focus on how prompt structure affects the reliability of the discrete user–association decisions and the continuous power–allocation variables for $R_{u,\min} = 12.5$ and $P_{b,\max} = 3.0$. We note a trend consistent with that observed in case study I. Weak prompting (Prompt 1) produces high violation rates (47.2%), and slow feasibility recovery, i.e., requires 153.2 steps for zero constraint violation. Adding structure through few-shot guidance,

equation-based reasoning, and modular decomposition through Prompts 2–7 reduces or removes violations, and accelerates convergence with improved sum-rates. The stability metrics in Table 3.12(b) quantify output-based uncertainty for both decision types, with $\Psi = 20$ and $\Lambda = 10$. User-association consistency is computed by applying the pairwise dispersion metric, $\mathcal{D}_{\mathrm{UQ}}$, given in (3.13) to the binary UA matrices and converting the resulting dispersion into a normalized consistency score $e^{-\mathcal{D}_{\mathrm{UQ}}}$, so values closer to 0 reflect high dispersion. Normalized consistency score increases from 0.13 (Prompt 1) to 0.97 (Prompt 7), indicating reduction in variability of a binary variable. Likewise, power-allocation stability, assessed via the $\mathrm{MAD}_{\mathrm{UQ}}$, given in (3.14), shows an analogous improvement.

3.5 Summary

This chapter proposes LLM4RRM, a novel BCD-inspired multi-agent feedback-driven LLM framework in which each LLM solver optimizes a subset of decision variables under a structured guidance provided by an evaluator LLM. An LLM-guided projection process is also introduced, which facilitates handling complicated constraints. Moreover, we introduce two metrics that quantify the trustworthiness of LLM-generated responses. Numerical results show that LLM4RRM not only delivers superior-quality solutions but also exhibits real-time scalability and adaptability, outperforming the baseline schemes. Furthermore, a prompt-based UQ is provided that demonstrates the reliability of LLM4RRM over heterogeneous network conditions. In addition, since LLM4RRM naturally accommodates additional decision variables through the block partitioning parameter K, it can be readily extended to scenarios such as multicast, mobility management, and network slicing.

Chapter 4

LLM4TSF: Multi-Agent LLM Framework for Multivariate Time-Series Forecasting

This chapter extends the thesis from LLM-based wireless decision-making to temporal prediction. In particular, it develops a general multi-agent large language model framework for multivariate time-series forecasting, referred to as *LLM4TSF*. Consistent with the design philosophy of Chapter 3, the framework is training-free, relies on structured prompt engineering rather than gradient-based optimization, decomposes the task into specialized sub-agents, and improves forecast quality through iterative evaluator feedback.

To preserve the same chapter logic used in Chapter 3, this chapter first introduces the *LLM4TSF* framework in a task-agnostic manner. This separation is intentional: the framework is meant to be reusable across different multivariate forecasting tasks and is therefore developed first at the methodological level before any application-specific instantiation is discussed.

4.1 System Model and Problem Formulation

Consider a multivariate time-series forecasting (TSF) problem in which each observation $\mathbf{x}_k \in \mathbb{R}^M$ denotes the joint values of M correlated channels at discrete time index k. Let $\mathbf{x}$ represent the system parameters, comprising the historical lookback window of length L,

$$\mathbf{X}_{\text{past}} = \big(\mathbf{x}_{-(L-1)}, \ldots, \mathbf{x}_0\big) \in \mathbb{R}^{L\times M}, \tag{4.1}$$

together with domain metadata such as cross-channel relationship priors and temporal event descriptors. The goal is to predict the future multivariate trajectory

$$\mathbf{y} \;\triangleq\; \mathbf{X}_{\text{future}} = \big(\mathbf{x}_1, \ldots, \mathbf{x}_H\big) \in \mathbb{R}^{H\times M}, \tag{4.2}$$

over a forecast horizon H, partitioned channel-wise as

$$\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_M\}, \qquad \mathbf{y}_m \in \mathbb{R}^H. \tag{4.3}$$

The forecasting task is subject to a set of J plausibility constraints $\{c_j(\mathbf{x}, \mathbf{y}) \triangleright_j 0\}_{j=1}^{J}$ encoding physical bounds, statistical consistency, and cross-channel coherence requirements, with $\triangleright_j \in \{\geq, \leq, =\}$.

Unlike trained forecasting models that explicitly learn the conditional distribution $p_\theta(\mathbf{X}_{\text{future}} \mid \mathbf{X}_{\text{past}})$ from task-specific labelled data, LLM4TSF approximates this mapping in a *training-free* manner through a cooperative multi-agent pipeline: (i) a time-series foundation model (TSFM) generates a probabilistic prior over future trajectories directly from $\mathbf{X}_{\text{past}}$ via zero-shot in-context inference; (ii) a large language

model (LLM) agent reasons over cross-channel physics and context statistics to produce a structured state-transition specification; and (iii) a Kalman filter layer fuses both outputs into a refined, statistically consistent multivariate forecast satisfying the plausibility constraints, without any parameter updates or dataset-specific retraining.

The joint evolution of the M channels over the forecast horizon is modelled as a linear discrete-time state-space system,

$$\mathbf{s}_t = \mathbf{F}\,\mathbf{s}_{t-1} + \boldsymbol{\eta}_t, \quad \boldsymbol{\eta}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{Q}), \tag{4.4}$$

$$\mathbf{z}_t = \mathbf{H}\,\mathbf{s}_t + \boldsymbol{\nu}_t, \quad \boldsymbol{\nu}_t \sim \mathcal{N}(\mathbf{0}, \mathbf{R}), \tag{4.5}$$

where $\mathbf{F} \in \mathbb{R}^{M \times M}$ is the LLM-defined state-transition matrix encoding cross-channel dependencies; $\mathbf{H} = \mathbf{I}_M$; $\mathbf{Q}$ and $\mathbf{R}$ are the process- and measurement-noise covariances; and $\mathbf{z}_t \in \mathbb{R}^M$ represents the TSFM-derived pseudo-observation at step t. The initial state $\mathbf{s}_0 = \mathbf{x}_0$ is subsequently adjusted by the LLM agent to account for anticipated short-term dynamics at the forecast boundary.

4.2 LLM4TSF Framework

In this section, we propose LLM4TSF, a novel training-free multi-agent framework for multivariate time-series forecasting, as illustrated in Fig. 4.1. The framework operates as a cooperative reasoning pipeline among two solver agents, a TSFM probabilistic forecasting agent LLM_1 and an LLM cross-channel reasoning agent LLM_2, coordinated by a shared evaluator agent LLM_E. Their joint output is subsequently passed to a Kalman filter layer that enforces cross-channel consistency and statistical plausibility.

4.2.1 LLM4TSF

4.2.1.1 Problem Decomposition and Agent Assignment

The forecast $\mathbf{y}$ is produced through two cooperative agent blocks. At iteration t, these blocks are given as:

$$\mathbf{y}_1^{(t)} = \hat{\mathbf{Y}}_{\text{TSFM}}^{(t)}, \qquad \mathbf{y}_2^{(t)} = \mathbf{\Phi}_{\text{LLM}}^{(t)}, \tag{4.6}$$

where $\hat{\mathbf{Y}}_{\text{TSFM}}^{(t)} \in \mathbb{R}^{M \times H}$ is the probabilistic trajectory produced by the TSFM agent LLM_1, and $\mathbf{\Phi}_{\text{LLM}}^{(t)}$ is the structured state-transition specification produced by the LLM reasoning agent LLM_2. Following standard agent notation, the output of the k-th solver agent at iteration t is:

$$\mathbf{y}_k^{(t)} = \mathcal{E}\big(\text{LLM}_k\big(\Pi_k^{(t)}\big)\big), \quad k \in \{1, 2\}, \quad t \geq 0, \tag{4.7}$$

where $\Pi_k^{(t)}$ is the structured prompt and $\mathcal{E}(\cdot)$ converts the agent's output into machine-interpretable numerical representations. LLM_2 receives $\mathbf{y}_1^{(t)}$ as part of its prompt context, ensuring it operates with the most current TSFM trajectory when reasoning about cross-channel dynamics.

4.2.1.2 TSFM Probabilistic Forecasting Agent (LLM_1)

TSFMs are large-scale neural networks pre-trained on heterogeneous time-series corpora spanning diverse domains and temporal granularities [88–90]. A particularly relevant capability of modern TSFMs is their ability to perform *zero-shot forecasting* via in-context learning: given a historical context window as input, the model directly

generates probabilistic forecasts over a specified horizon without any fine-tuning or gradient updates. Depending on the architecture, encoder-decoder, decoder-only, or encoder-only, TSFMs produce either autoregressive sample trajectories or direct quantile forecasts through a quantile regression head [88, 89].

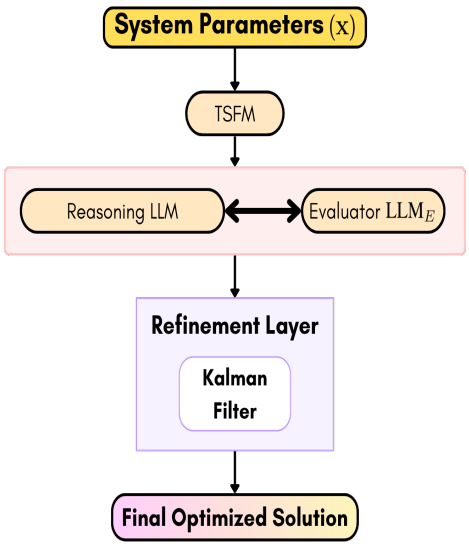


Figure 4.1: LLM4TSF Architecture

The TSFM agent LLM_1 takes the historical context $\mathbf{X}_{\text{past}}$ as its sole numerical input and applies per-channel instance normalization,

$$\tilde{x}_m(t) = \frac{x_m(t) - \mu_m}{\sigma_m}, \quad t = -(L-1), \ldots, 0, \tag{4.8}$$

where μ_m and σ_m are the sample mean and standard deviation of channel m over the lookback window. The normalised context is passed through the TSFM to obtain, for each channel, a probabilistic forecast in the form of either sample trajectories or quantile estimates. Specifically, the TSFM produces for each channel m a set of N_q quantile levels $\{\hat{q}_{m,\alpha_i}\}_{i=1}^{N_q}$ at each forecast step. A point estimate and uncertainty

envelope are extracted as:

$$\hat{\mu}_m = \text{median}\big\{\hat{q}_{m,\alpha_i}\big\}, \qquad \hat{\sigma}_m = \hat{q}_{m,\alpha_{\text{high}}} - \hat{q}_{m,\alpha_{\text{low}}}, \tag{4.9}$$

and the full TSFM output block, denormalised back to original units, is:

$$\mathbf{y}_1^{(t)} = \hat{\mathbf{Y}}_{\text{TSFM}} = \big\{\hat{\boldsymbol{\mu}},\, \hat{\boldsymbol{\sigma}},\, \hat{\mathbf{q}}_{\text{low}},\, \hat{\mathbf{q}}_{\text{high}}\big\} \in \mathbb{R}^{M\times H}. \tag{4.10}$$

Crucially, the TSFM agent LLM_1 operates entirely on numerical time-series data and does not accept natural-language prompts; it is not a text-based LLM but rather a numerical pre-trained model invoked through a standard inference API. Its output $\mathbf{y}_1^{(t)}$ serves two roles: the point estimate $\hat{\boldsymbol{\mu}}$ provides pseudo-observations $\mathbf{z}_t$ for the downstream Kalman update, while $\hat{\boldsymbol{\sigma}}$ provides per-channel uncertainty diagnostics that inform the LLM reasoning agent LLM_2.

4.2.1.3 LLM Cross-Channel Reasoning Agent (LLM_2)

LLM_2 translates domain knowledge, context-window statistics, and TSFM forecast diagnostics into a structured, numerically grounded specification of the cross-channel state-transition matrix $\mathbf{F}$ for the Kalman layer. Its prompt is constructed as:

$$\Pi_2^{(t)} = \mathcal{P}_2\big(\mathbf{y}_1^{(t)},\, \Delta^{(t)},\, \mathcal{H}_2^{(t)}\big), \quad t \geq 0, \tag{4.11}$$

where $\mathbf{y}_1^{(t)}$ is the current TSFM output provided as numerical context, $\Delta^{(t)}$ is the evaluator feedback (empty at $t = 0$), and $\mathcal{H}_2^{(t)}$ is the agent's accumulated history.

Formally:

$$\mathcal{H}_2^{(t)} = \begin{cases} \{\mathbf{x},\, \mathcal{S}_M,\, \mathcal{T}_2\}, & t = 0, \\ \{\mathcal{H}_2^{(t-1)},\, \Delta^{(t)},\, \mathbf{y}^{(t-1)},\, \mathrm{RT}_2^{(t-1)}\}, & t \geq 1, \end{cases} \tag{4.12}$$

where $\mathcal{T}_2$ is the LLM task description, $\mathcal{S}_M$ is the few-shot example set retrieved from a database $\mathcal{D}$ of past forecasting contexts, and $\mathrm{RT}_2^{(t)}$ is the agent's reasoning trace from the prior iteration. The prompt $\Pi_2^{(t)}$ integrates four structured information layers.

(L1) Domain context: Natural-language descriptions of cross-channel directional relationships with their characteristic lag values and strength priors, and known temporal events active at the current timestamp t_0 (e.g., working hours, weekday/weekend flag).

(L2) Context-window statistics: Per-channel descriptors $\mathbf{c}_m = [\mu_m,\, \sigma_m,\, \Delta x_m,\, x_m(0),\, \rho_{\text{active},m}]^\top$, where $\Delta x_m = x_m(0) - x_m(-(L-1))$ is the context trend and $\rho_{\text{active},m}$ is the fraction of non-zero samples, detecting inactive channels whose coupling coefficients should be suppressed.

(L3) TSFM forecast diagnostics: A compact per-channel numerical summary of $\mathbf{y}_1^{(t)}$,

$$\mathbf{d}_m = \left[\hat{\mu}_m^{\text{start}},\, \hat{\mu}_m^{\text{mid}},\, \hat{\mu}_m^{\text{end}},\, \bar{\sigma}_m,\, \sigma_m^{\max},\, \phi_m^{\text{null}}\right]^\top, \tag{4.13}$$

where $\hat{\mu}_m^{\text{start/mid/end}}$ are short-window averages at the beginning, middle, and end of the forecast horizon; $\bar{\sigma}_m$ and $\sigma_m^{\max}$ characterise forecast dispersion; and ϕ_m^{null} flags any sustained near-zero segment indicating channel inactivity. High dispersion $\bar{\sigma}_m$ on channel m signals low TSFM confidence, instructing LLM_2 to reduce the coupling coefficient magnitude for that channel and assign it lower reliability in the Kalman layer.

(L4) Few-shot in-context examples: Two representative examples retrieved

from $\mathcal{D}$: a *good-forecast example* where the dominant cross-channel relationship achieves Pearson correlation $\rho > 0.6$, and a *bad-forecast example* where the dependent channel remains flat despite a significant input trend ($\Delta > 8\%$), flagging a coupling breakdown the agent must counteract. A chain-of-thought directive instructs the agent to generate at minimum six $\mathbf{F}$-matrix entries, of which at least four must deviate from the static domain priors by more than 0.05 based on the specific forecast window.

The agent is constrained to emit a single self-contained JSON object defining the output block:

$$\mathbf{y}_2^{(t)} = \mathbf{\Phi}_{\text{LLM}} = \left\{\mathcal{E}_{\mathbf{F}},\ \boldsymbol{\delta}_0,\ \boldsymbol{\rho}_{\text{rel}},\ \mathcal{E}_{\text{events}}\right\}, \tag{4.14}$$

where $\mathcal{E}_{\mathbf{F}} = \{(j{\to}i,\ c_{ij},\ \ell_{ij})\}$ specifies transition coefficients $c_{ij} \in [0, 1]$ and lags $\ell_{ij} \geq 0$; $\boldsymbol{\delta}_0 \in \mathbb{R}^M$ are initial state corrections clipped to $\pm 10\%$ of the physical channel range; $\boldsymbol{\rho}_{\text{rel}} \in (0.1,\ 2.0]^M$ are per-channel TSFM reliability weights scaling the Kalman measurement noise $\mathbf{R}$; and $\mathcal{E}_{\text{events}}$ lists active domain events at t_0.

4.2.1.4 Evaluator Agent and Feedback Generation

Once both agents produce their outputs, the evaluator LLM_E assesses the joint solution $\mathbf{y}^{(t)} = \{\mathbf{y}_1^{(t)}, \mathbf{y}_2^{(t)}\}$ against the plausibility constraints. The evaluator prompt is:

$$\Pi_E^{(t)} = \mathcal{P}_E\big(\mathbf{r}^{(t)},\ \mathbf{y}^{(t)},\ \text{RT}_2^{(t)},\ \mathcal{H}_E^{(t)}\big), \quad t \geq 0, \tag{4.15}$$

where $\mathbf{r}^{(t)} \triangleq \{c_j(\mathbf{x}, \mathbf{y}^{(t)})\}_{j=1}^{J}$ is the constraint evaluation vector and $\mathrm{RT}_2^{(t)}$ is the LLM reasoning trace. The evaluator history is:

$$\mathcal{H}_E^{(t)} = \begin{cases} \{\mathbf{x},\, \mathcal{S}_E,\, \mathcal{T}_E\}, & t = 0, \\ \{\mathcal{H}_E^{(t-1)},\, \Delta^{(t)},\, \mathbf{r}^{(t-1)},\, \mathbf{y}^{(t-1)},\, \mathrm{RT}_2^{(t-1)}\}, & t \geq 1, \end{cases} \tag{4.16}$$

where $\mathcal{S}_E$ and $\mathcal{T}_E$ are the evaluator's few-shot set and task description assigning it the role of a statistical critic. The evaluator generates high-level textual critique:

$$\Delta^{(t+1)} = \mathrm{LLM}_E\big(\Pi_E^{(t)}\big), \tag{4.17}$$

which is parsed by an extraction function $\mathcal{E}_E(\cdot)$ into structured feedback $\Delta^{(t+1)}$ and injected into $\Pi_2^{(t+1)}$ for the next iteration to guide refinement of $\mathbf{\Phi}_{\mathrm{LLM}}$. Three plausibility constraints are enforced: **(C1)** forecast mean within δ_μ standard deviations of the historical mean; **(C2)** forecast-to-history volatility ratio within $[r_{\min}, r_{\max}]$; **(C3)** dominant sign structure of the cross-channel correlation matrix preserved in the predicted trajectories.

4.2.2 Kalman Filter Feasibility Layer

After T reasoning iterations, the joint solution $\mathbf{y}^{(T)} = \{\hat{\mathbf{Y}}_{\mathrm{TSFM}}^{(T)}, \mathbf{\Phi}_{\mathrm{LLM}}^{(T)}\}$ is passed to the Kalman filter feasibility layer, which fuses both agent outputs into a single cross-channel-consistent forecast:

$$\hat{\mathbf{y}}^{(T+1)} = \mathrm{KalmanFuse}\big(\hat{\mathbf{Y}}_{\mathrm{TSFM}}^{(T)},\ \mathbf{\Phi}_{\mathrm{LLM}}^{(T)},\ \mathbf{x}\big) \in \mathbb{R}^{M \times H}. \tag{4.18}$$

4.2.2.1 State-Transition Matrix Construction

$\mathbf{F}$ is assembled from $\mathbf{\Phi}_{\text{LLM}}^{(T)}$ in three steps. *Step 1, Domain prior seeding:* $\mathbf{F}$ is initialised to $\mathbf{I}_M$ with off-diagonal entries populated from static domain priors in $\mathbf{x}$. *Step 2, LLM-adaptive override:* Entries from $\mathcal{E}_{\mathbf{F}}$ replace domain priors with context-calibrated coefficients c_{ij}, subject to a bidirectional amplification guard,

$$|F_{ij} \cdot F_{ji}| \leq 0.25, \quad \forall\, i \neq j, \tag{4.19}$$

preventing circular state amplification over the horizon H. *Step 3, Spectral radius stabilisation:* Off-diagonal entries are iteratively damped until

$$\rho(\mathbf{F}) = \max |\lambda_i(\mathbf{F})| \leq 0.99, \tag{4.20}$$

guaranteeing bounded multi-step state rollout. Channels with $\hat{\sigma}_m/|\hat{\mu}_m| > \kappa$ have all associated off-diagonal $\mathbf{F}$ entries zeroed, preventing high-uncertainty channels from corrupting their cross-channel partners.

4.2.2.2 Noise Covariance and Kalman Fusion

The process noise is $\mathbf{Q} = q\,\mathbf{I}_M$, and the measurement noise is per-channel adaptive:

$$R_{mm} = \frac{R_0}{\max(\rho_{\text{rel},m},\, 0.1)}, \tag{4.21}$$

where R_0 is the base noise level, so that high LLM-assigned reliability reduces R_{mm} and shifts trust toward the TSFM pseudo-observation, while low reliability increases R_{mm} and shifts trust toward the transition model. Filtering is performed in the

normalised state space with the initial state incorporating the LLM correction:

$$\tilde{\mathbf{s}}_0 = \frac{(\mathbf{s}_0 + \boldsymbol{\delta}_0) - \boldsymbol{\mu}_{\text{ctx}}}{\boldsymbol{\sigma}_{\text{ctx}}}. \tag{4.22}$$

For each step $t = 1, \ldots, H$, the standard predict–update equations are applied:

$$\tilde{\mathbf{s}}_{t|t-1} = \mathbf{F}\,\tilde{\mathbf{s}}_{t-1|t-1}, \quad \mathbf{P}_{t|t-1} = \mathbf{F}\mathbf{P}_{t-1|t-1}\mathbf{F}^\top + \mathbf{Q}, \tag{4.23}$$

$$\mathbf{K}_t = \mathbf{P}_{t|t-1}(\mathbf{P}_{t|t-1} + \mathbf{R})^{-1}, \tag{4.24}$$

$$\tilde{\mathbf{s}}_{t|t} = \tilde{\mathbf{s}}_{t|t-1} + \mathbf{K}_t\big(\tilde{\mathbf{z}}_t - \tilde{\mathbf{s}}_{t|t-1}\big), \tag{4.25}$$

where $\tilde{\mathbf{z}}_t$ is the normalised channel-wise point estimate $\hat{\boldsymbol{\mu}}(:, t)$ from $\hat{\mathbf{Y}}^{(T)}_{\text{TSFM}}$ at step t. The Kalman gain $\mathbf{K}_t$ dynamically balances trust between the TSFM trajectory and the LLM-specified transition dynamics. After H steps, the filtered states are denormalised,

$$\hat{y}_m(t) = \tilde{s}_{m,t|t} \cdot \sigma_{\text{ctx},m} + \mu_{\text{ctx},m}, \quad t = 1, \ldots, H, \tag{4.26}$$

yielding the final multivariate forecast $\hat{\mathbf{y}}^{(T+1)} \in \mathbb{R}^{M \times H}$.

4.2.3 Uncertainty Quantification

To assess the reliability and trustworthiness of LLM4TSF outputs, we introduce a multi-component UQ framework operating at three levels: the TSFM probabilistic output, the Kalman feasibility layer, and the LLM reasoning agent.

4.2.3.1 TSFM Quantile Propagation

The TSFM agent LLM_1 generates N_s probabilistic sample trajectories per channel from which the 2.5th and 97.5th percentiles are extracted as lower and upper uncertainty bounds,

$$\hat{q}_{m,\text{low}}(t) = \text{P}_{2.5}\big\{\text{samples}_m(t)\big\}, \quad \hat{q}_{m,\text{high}}(t) = \text{P}_{97.5}\big\{\text{samples}_m(t)\big\}, \tag{4.27}$$

targeting a nominal 95% coverage interval for each channel m at each forecast step t. A per channel *quantile spread score* summarises predictive interval width across the horizon:

$$\text{QS}_m = \frac{1}{H}\sum_{t=1}^{H}\big(\hat{q}_{m,\text{high}}(t) - \hat{q}_{m,\text{low}}(t)\big). \tag{4.28}$$

Channels exhibiting large QS_m receive suppressed coupling coefficients in $\mathbf{F}$ and elevated measurement noise in $\mathbf{R}$, directly linking TSFM uncertainty to Kalman filter trust allocation.

4.2.3.2 Kalman Covariance Uncertainty

The Kalman filter maintains an internal error covariance matrix $\mathbf{P}_{t|t} \in \mathbb{R}^{M\times M}$ at every forecast step via (4.23)- (4.25). The diagonal entries $P_{mm}(t)$ represent per channel posterior variance, which are propagated into physical units to form 95% confidence bands:

$$\hat{y}_m(t) \pm 1.96\,\sqrt{P_{mm}(t)}\cdot\sigma_{\text{ctx},m}, \tag{4.29}$$

where $\sigma_{\text{ctx},m}$ is the standard deviation of channel m over the historical context window. To prevent numerical collapse of $\mathbf{P}$ over long horizons, a covariance floor $\mathbf{P} \leftarrow \max(\mathbf{P}, \epsilon\, _M)$ with $\epsilon = 10^{-4}$ is enforced after each predict and update step.

A per-channel *horizon uncertainty growth score*,

$$\mathrm{UG}_m = \frac{1}{H} \sum_{t=1}^{H} P_{mm}(t), \tag{4.30}$$

quantifies how rapidly the filter loses confidence as the horizon extends. Unlike the TSFM bands which reflect the pre-trained model's distributional uncertainty, the Kalman bands capture the compound uncertainty arising jointly from the LLM-specified transition dynamics and the TSFM measurement reliability.

4.2.3.3 LLM Reasoning Uncertainty

To assess the stability of the LLM reasoning agent LLM_2, we adopt the prompt-based and output-based UQ strategies from 3.1.6. Output-based stability is measured via the mean absolute deviation metric $\mathrm{MAD}_{\mathrm{UQ}}$ across Λ independent runs of the full pipeline with fixed inputs and temperature $T = 1.0$:

$$\mathrm{MAD}_{\mathrm{UQ}} = \frac{1}{\Lambda \Upsilon} \sum_{\lambda=1}^{\Lambda} \sum_{\upsilon=1}^{\Upsilon} \left| y_{\lambda,\upsilon}^{(T)} - \bar{y}_{\upsilon} \right|, \tag{4.31}$$

where $\Upsilon = M \times H$ is the total number of forecast elements and $\bar{y}_{\upsilon}$ is the element-wise mean across Λ runs. Lower $\mathrm{MAD}_{\mathrm{UQ}}$ indicates greater LLM output consistency. Prompt-based uncertainty is assessed by running the full pipeline under three prompt configurations of increasing richness, minimal statistics only (prompt 1), full domain knowledge (prompt 2), and addition of chain-of-thought (prompt 3) in prompt 2 reasoning, and comparing the resulting MSE (Mean Square Error) and plausibility constraint satisfaction rates across configurations.

4.2.4 Univariate Special Case

The general formulation naturally includes univariate forecasting as the special case $M = 1$. While the overall LLM4TSF pipeline is preserved: the TSFM agent LLM_1 provides a probabilistic prior, the LLM agent LLM_2 produces a scalar transition specification, and the evaluator LLM_E checks forecast plausibility. The only simplification is that no inter-channel dependencies are modeled, so the cross-channel coherence constraint (C3) is removed. The evaluator therefore enforces only the mean- and volatility-consistency constraints corresponding to (C1) and (C2), and the Kalman feasibility layer fuses the TSFM prior with a scalar transition model.

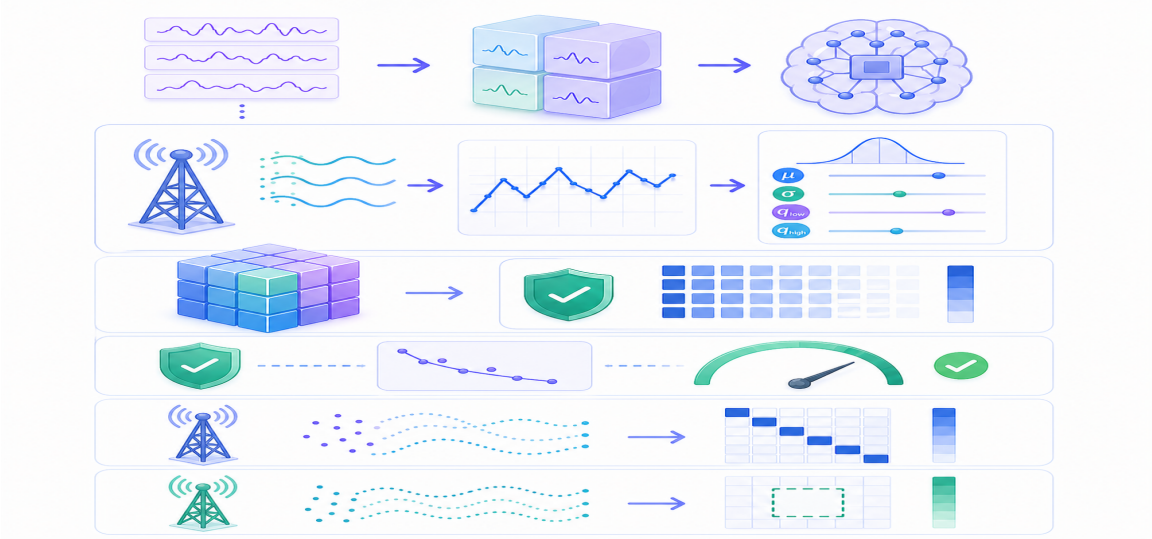


Figure 4.2: EMF Illustration

4.3 Use Case: Electromagnetic Field (EMF) Forecasting via LLM4TSF

4.3.1 EMF Signal Model and Grouped Representation

We instantiate LLM4TSF on frequency-selective EMF [91] forecasting across cellular technology bands. The monitoring dataset comprises $N_c = 29$ narrowband channels

resampled to a 7.5-minute grid. Received channel power $P_{\mathrm{dBm}}(f)$ is converted to electric field strength (V/m) as [92]:

$$E_f(t) = \sqrt{\frac{10^{(P_{\mathrm{dBm}}(f)-30)/10} Z_0}{A_e(f)}}, \tag{4.32}$$

where $Z_0 \approx 376.73\,\Omega$ and $A_e(f) = G_f \lambda_f^2/(4\pi)$ uses Keysight N6850A calibration factors. A Hampel outlier filter (window 11, threshold $3\,\sigma_{\mathrm{MAD}}$) is applied channel-wise to suppress sensor spikes prior to forecasting. The N_c channels are grouped into $M = 4$ technology classes $\mathcal{G} = \{g_{2\mathrm{G}}, g_{3\mathrm{G}}, g_{4\mathrm{G}}, g_{5\mathrm{G}}\}$ via root-sum-square fusion:

$$E_{g_m}(t) = \Big(\sum_{f \in \mathcal{F}_{g_m}} E_f^2(t) \Big)^{1/2}, \tag{4.33}$$

yielding the multivariate EMF state vector $\mathbf{E}(t) \in \mathbb{R}^4$ at each timestep t, which constitutes $\mathbf{X}_{\mathrm{past}}$ for LLM4TSF.

4.3.2 EMF Problem Instantiation

Given $\mathbf{X}_{\mathrm{past}} \in \mathbb{R}^{L \times 4}$, the forecasting task is instantiated with $M = 4$ and the plausibility constraints (C1)–(C3) specialised as:

$$c_1 : \big|\mu(\hat{\mathbf{E}}_{g_m}) - \mu(\mathbf{E}_{g_m}^{\mathrm{past}})\big| \le \delta_\mu\, \sigma(\mathbf{E}_{g_m}^{\mathrm{past}}), \quad \forall g_m, \tag{4.34}$$

$$c_2 : r_{\min} \le \sigma(\hat{\mathbf{E}}_{g_m})/\sigma(\mathbf{E}_{g_m}^{\mathrm{past}}) \le r_{\max}, \quad \forall g_m, \tag{4.35}$$

$$c_3 : \mathrm{sign}\big(\mathrm{corr}(\hat{\mathbf{E}}_{g_m}, \hat{\mathbf{E}}_{g_n})\big) = \mathrm{sign}(\rho_{g_m,g_n}), \quad \forall (g_m, g_n) \in \mathcal{P}_\rho, \tag{4.36}$$

where $\mathcal{P}_\rho$ is the set of dominant technology-band correlation pairs.

4.3.2.1 TSFM Agent for EMF

The TSFM agent takes the normalised grouped EMF history $\tilde{\mathbf{E}}_{g_m}^{\text{past}}$ per (4.8) as input for each of the four technology groups and produces a probabilistic forecast via zero-shot in-context inference. The resulting channel-wise point estimates and uncertainty envelopes form the output block $\mathbf{y}_1^{(T)} = \hat{\mathbf{Y}}_{\text{TSFM}}^{\text{EMF}} \in \mathbb{R}^{4\times H}$, whose columns serve as pseudo-observations $\tilde{\mathbf{z}}_t$ in the Kalman update step.

4.3.2.2 LLM Reasoning Agent for EMF

The system parameters $\mathbf{x}$ encode the following EMF-specific domain knowledge: adjacent 4G LTE (Long-Term Evolution) 800 MHz carriers co-vary at strength ≈0.7; 2G GSM (Global System for Mobile Communications) 900 MHz carriers correlate at ≈0.6; 4G 2600 MHz carriers exhibit the strongest coupling at ≈0.8; and a lag-1 relationship exists between 2G and 4G load patterns. Working-hour activity (09:00–17:00 on weekdays) drives the dominant diurnal EMF cycle. Using the context statistics $\{\mathbf{c}_{g_m}\}$ and TSFM diagnostics $\{\mathbf{d}_{g_m}\}$ from (4.13), the agent calibrates $\mathcal{E}_{\mathbf{F}}$ accordingly: high forecast dispersion on the 5G band suppresses its coupling coefficients and elevates its measurement noise in $\mathbf{R}$; the 4G band's known weak cross-technology correlation ($\rho \leq 0.19$ [92]) causes its off-diagonal $\mathbf{F}$ entries to be suppressed. The output $\mathbf{y}_2^{(T)} = \mathbf{\Phi}_{\text{LLM}}^{\text{EMF}}$ provides per-group $\mathbf{F}$-matrix entries, initial state corrections $\boldsymbol{\delta}_0$, and per-group reliability weights $\boldsymbol{\rho}_{\text{rel}}$.

4.3.2.3 Kalman Fusion for EMF

The 4×4 matrix $\mathbf{F}_{\text{EMF}}$ is assembled from $\mathbf{\Phi}_{\text{LLM}}^{\text{EMF}}$ subject to (4.19) and (4.20). After the H-step predict–update loop (4.23)–(4.25), with TSFM pseudo-observations $\tilde{\mathbf{z}}_t$ from

$\hat{\mathbf{Y}}^{\text{EMF}}_{\text{TSFM}}$, filtered states are denormalised per (4.26) to recover physical V/m units, yielding:

$$\hat{\mathbf{E}}^{\text{future}} = \left[\hat{\mathbf{E}}_{g_{2\text{G}}},\ \hat{\mathbf{E}}_{g_{3\text{G}}},\ \hat{\mathbf{E}}_{g_{4\text{G}}},\ \hat{\mathbf{E}}_{g_{5\text{G}}}\right] \in \mathbb{R}^{H\times 4}, \tag{4.37}$$

satisfying constraints (4.34)–(4.36) through the joint action of the evaluator feedback loop and the Kalman feasibility layer.

4.4 Results and Discussion

4.4.1 Experimental Setup

4.4.1.1 Dataset and Evaluation Protocol

Experiments are conducted on a real-world Italian Radio-Frequency Electromagnetic Field (RF-EMF) dataset collected at the University Hospital Tor Vergata rooftop site in Rome from July 2023 to March 2024, comprising $N_c = 29$ narrowband frequency channels spanning 9 kHz–6 GHz from four network operators (Iliad, TIM, VF, W3) [93]. Raw measurements are converted from dBm to V/m using Keysight N6850A calibration factors, resampled to a 7.5-minute grid (192 samples/day), and grouped into $M = 4$ technology-band series per (4.33). Performance is evaluated over $N_W = 100$ randomly selected rolling windows, each with a lookback of $L = 1344$ steps (7 days) and a forecast horizon of $H = 192$ steps (1 day). We report ND (normalized deviation), NRMSE (normalized root-mean-square error), and MAPE (mean absolute percentage error) as mean across windows at both the aggregated operator and technology-band levels.

4.4.1.2 LLM4TSF Configuration

The TSFM agent LLM_1 uses Amazon Chronos-2 [88], generating $N_s = 1000$ sample trajectories per channel via inverse-CDF (Cumulative Distribution Function) sampling from the model's native quantile outputs. The LLM reasoning agent LLM_2 and evaluator LLM_E both use gpt-5.1-mini with temperature 1.0 and 3000 maximum output tokens. The prompt includes $k_s = 2$ in-context examples per window: one good-forecast example ($\rho > 0.6$) and one bad-forecast example ($\Delta > 8\%$). Up to 3 re-attempts are made per window on JSON parse failure; windows for which all attempts fail are excluded from the aggregate statistics. The Kalman filter uses base sensor noise $R_0 = 2.0$, process noise scale $q = 0.01 \times \hat{\sigma}_{\text{TSFM}}$, and high-uncertainty flag threshold $\kappa = 5.0$.

4.4.1.3 Baselines

The primary benchmark is EMFusion [92], a conditional diffusion model for frequency-selective EMF forecasting evaluated in its working-hour-conditioned multivariate configuration. We consider a representative set of baselines across forecasting paradigms: Naive Forecast (NF) as a naive statistical reference, IQLSTM for recurrent sequence modeling, TimeGrad as a diffusion-based generative model, and EMForecaster [77] as a strong domain-specific deep learning baseline. To isolate the contribution of our framework, we also include a standalone TSFM baseline using Chronos-2 [88]. All baselines (except NF and Chronos) require dataset-specific training, whereas LLM4TSF operates with zero training overhead.

Table 4.1: Operator-wise forecasting performance comparison. Lower is better. Bold = best; underline = second best.

Model	Iliad			TIM			VF			W3		
	ND	NRMSE	MAPE	ND	NRMSE	MAPE	ND	NRMSE	MAPE	ND	NRMSE	MAPE
NF	0.1901	0.2271	22.0056	0.2674	0.3344	30.5366	0.1505	0.1899	16.3322	0.1732	0.2126	19.5926
IQLSTM	0.1742	0.2088	22.0082	0.2272	0.2905	23.7362	0.1484	0.2158	16.5989	0.1742	0.2179	19.3497
TimeGrad (29)	0.3405	0.8942	89.2536	0.4230	1.3310	52.4586	0.2997	1.0983	29.7099	0.3184	1.0497	48.2214
EMForecaster	0.1706	0.2019	21.5334	0.2042	0.2499	22.5550	0.1419	0.2031	16.0613	0.1585	0.1937	18.5124
EMFusion	**0.1304**	**0.1772**	**14.2795**	0.1433	0.2143	14.1847	**0.0997**	0.1781	**11.0273**	**0.1132**	0.1639	12.1131
Chronos (TSFM only)	0.1997	0.2345	23.5577	0.1062	0.1308	11.0043	0.1241	0.1481	12.9720	0.1294	0.1577	13.9718
LLM4TSF	0.1648	0.2107	19.6169	**0.0996**	**0.1237**	**10.1546**	0.1099	**0.1321**	11.5692	0.1147	**0.1405**	**11.9509**

Table 4.2: Technology-band-wise forecasting performance comparison. Lower is better. Bold = best; underline = second best.

Model	2G			3G			4G			5G		
	ND	NRMSE	MAPE	ND	NRMSE	MAPE	ND	NRMSE	MAPE	ND	NRMSE	MAPE
NF	0.1416	0.1773	14.3473	0.2153	0.2534	24.2409	0.1637	0.1983	18.1657	0.2103	0.3111	18.5137
IQLSTM	0.1342	0.1708	13.7036	0.1841	0.2256	20.9445	0.1488	0.1827	17.1466	0.1901	0.2994	18.1656
TimeGrad (29)	0.3761	2.2558	38.7363	0.3133	0.9559	61.6179	0.3375	0.9451	56.1075	0.3912	0.9879	40.6475
EMForecaster	0.1682	0.2156	19.0429	0.1688	0.2122	19.6655	0.1661	0.2102	19.3257	0.1802	0.2262	20.6762
EMFusion	**0.0821**	0.1135	**8.3460**	0.1153	0.1564	**13.1791**	**0.1109**	0.1495	11.8418	0.1518	0.4311	11.5743
Chronos (TSFM only)	0.1283	0.1543	13.2087	0.1647	0.1920	19.5078	0.1312	0.1591	14.0677	0.0556	**0.1205**	4.4152
LLM4TSF	0.0998	**0.1068**	9.7629	**0.1098**	**0.1401**	14.1276	0.1146	**0.1413**	**11.8159**	**0.0552**	0.1214	**4.3442**

4.4.2 Results across different technologies and operators:

Several observations follow from Tables 4.1 and 4.2. First, the proposed LLM4TSF framework substantially improves over the standalone Chronos baseline on most operator- and technology-level aggregates, which indicates that the LLM reasoning and Kalman fusion stages contribute meaningful value beyond the TSFM prior alone. This is especially evident for TIM, VF, and W3 at the operator level, where LLM4TSF achieves the lowest or near-lowest ND, NRMSE, and MAPE values.

Second, LLM4TSF is highly competitive with the strongest trained baselines. At the operator level, it achieves the best overall results for TIM and W3 and remains close to EMFusion on Iliad and VF. At the technology-band level, LLM4TSF achieves the best NRMSE on 2G, 3G, and 4G, and the best ND and MAPE on 5G. These results suggest that the proposed framework is particularly effective at preserving coarse multivariate dynamics while adapting the TSFM prior through structured

cross-band reasoning.

Third, the results indicate a complementary relationship between LLM4TSF and EMFusion. EMFusion remains strongest on some aggregates, particularly where diffusion-based training can exploit richer dataset-specific structure, whereas LLM4TSF often provides the best or second-best performance without any retraining. This highlights the practical value of the training-free setup: LLM4TSF offers competitive multivariate forecasting quality while avoiding the computational overhead and dataset dependence of fully trained generative models.

4.4.3 Uncertainty Quantification Results

4.4.3.1 TSFM and Kalman Coverage

Table 4.3 summarises the empirical coverage and uncertainty scores grouped by technology band. Both the Kalman 95% confidence bands (4.29) and the TSFM quantile bands (4.27) achieve near-nominal coverage across all bands, with mean Kalman coverage of 0.984 and mean TSFM coverage of 0.956. The 4G band exhibits the largest mean quantile spread score QS_m, reflecting higher TSFM predictive uncertainty on mid-band carriers, whereas 5G channels show near-zero QS_m and UG_m, indicating stable filter confidence across the full horizon. Channels with large QS_m automatically receive elevated measurement noise in $\mathbf{R}$ and suppressed $\mathbf{F}$ coupling coefficients, demonstrating that the UQ framework actively feeds back into forecast quality.

4.4.3.2 Prompt-Based and Output-Based UQ

Table 4.4 reports the prompt-based UQ results across technology bands. Chain-of-thought prompting consistently achieves the best performance across all bands, with

Table 4.3: Technology-band-grouped TSFM and Kalman UQ coverage. QS_m: mean quantile spread (4.28); UG_m: mean horizon uncertainty growth (4.30).

Band	**Kalman Cov.**	**TSFM Cov.**	$\overline{\mathbf{QS}}_m$	$\overline{\mathbf{UG}}_m$
2G	0.989	0.966	0.071	0.001
3G	0.994	0.965	0.052	0.001
4G	0.993	0.951	0.067	0.051
5G	0.970	0.955	0.002	0.000

Table 4.4: Prompt-based UQ: forecasting performance across three prompt configurations and technology bands. Lower is better.

Prompt Style	**Band**	**ND**	**NRMSE**	**MAPE**
Prompt 1	2G	0.1313	0.1635	13.1587
	3G	0.1566	0.1898	16.4532
	4G	0.1174	0.1471	12.6664
	5G	0.0497	0.1398	5.2403
Prompt 2	2G	0.1214	0.1464	12.8712
	3G	0.1326	0.1653	15.5841
	4G	0.1147	0.1469	11.9985
	5G	0.0664	0.1249	5.1948
Prompt 3	2G	0.0998	0.1068	9.7629
	3G	0.1098	0.1401	14.1276
	4G	0.1146	0.1413	11.8159
	5G	0.0552	0.1214	4.3442

notable gains on 2G (MAPE 9.76 vs 13.16 for minimal) and 3G (MAPE 14.13 vs 16.45), while improvements on 4G and 5G are more modest. This suggests that explicit step-by-step reasoning benefits lower-frequency bands where cross-channel coupling is stronger, but adds less value on higher-frequency bands where the TSFM prior is already highly accurate.

Table 4.5: Output-based UQ: mean MAD_{UQ} per technology band across $\Lambda = 20$ independent runs. Lower = more consistent LLM outputs.

Band	$\overline{\textbf{MAD}}_{\textbf{UQ}}$	**Most Variable Channel**
2G	0.00548	f955 (0.010)
3G	0.00876	f2150 (0.015)
4G	0.01296	f2635 (0.029)
5G	0.00085	f3610 (0.000)

4.5 Summary

This chapter introduced *LLM4TSF*, a training-free multi-agent LLM framework for general multivariate time-series forecasting combining a TSFM probabilistic agent, an LLM cross-channel reasoning agent, a shared evaluator enforcing statistical plausibility, and a Kalman feasibility layer. A multi-component UQ framework was further introduced across three levels: TSFM quantile propagation, Kalman covariance tracking, and LLM output stability via prompt-based and output-based sampling. Instantiated on EMF forecasting, LLM4TSF achieves competitive accuracy against trained baselines with zero training overhead, with UQ results confirming near-nominal 95% coverage across all technology bands and 4G channels exhibiting the highest forecast uncertainty.

Chapter 5

Future Directions and Conclusions

5.1 Concluding Remarks

This thesis investigated how large language models can be used as training-free decision-making engines for wireless systems through prompt engineering, task decomposition, and agent-based coordination. The central objective was not merely to apply LLMs to a single use case, but to develop a general methodology through which the same family of models can adapt to diverse optimization and forecasting problems without task-specific retraining. To this end, the thesis studied two complementary wireless applications: resource allocation and multivariate electromagnetic field (EMF) forecasting.

In the resource management setting, this thesis introduced LLM4RRM, a multi-agent framework inspired by block coordinate descent in which multiple solver agents optimize subsets of variables under the supervision of an evaluator agent. This decomposition enables the LLM system to handle coupled and constrained decision spaces more effectively than a single monolithic prompt. An LLM-guided projection mechanism was further incorporated to improve feasibility under complex constraints, and

trustworthiness was assessed through prompt-based and output-based uncertainty measures. The results demonstrate that the proposed framework can generate competitive solutions, maintain feasibility, and adapt to varying wireless network conditions while avoiding the retraining burden associated with conventional supervised or reinforcement learning pipelines.

In the forecasting setting, the thesis extends the same design philosophy to multivariate time-series prediction. The forecasting framework adopts a dual-agent or multi-stage prompting strategy in which different agents specialize in complementary sub-tasks such as trend extraction, fluctuation estimation, example retrieval, and iterative forecast refinement. Rather than relying on gradient-based training, the framework leverages structured prompts, few-shot retrieval, and evaluator feedback to exploit the in-context reasoning ability of LLMs. This direction further supports the broader thesis claim that prompt-engineered, agentic LLM systems can provide a unified and adaptable alternative for wireless problems that differ substantially in form, ranging from constrained optimization to temporal prediction.

Taken together, the contributions of this thesis establish an important proof of concept: LLMs can be organized as modular reasoning agents that solve domain-specific wireless tasks through language-based coordination alone. The key insight is that performance does not solely depend on larger models or additional fine-tuning, but also on how the problem is represented, partitioned, and iteratively refined through prompts. This makes the proposed paradigm attractive for wireless environments where objectives, constraints, operating conditions, and data distributions evolve rapidly. In such settings, the ability to adapt through prompting rather than retraining offers a practical advantage in latency, deployment flexibility, and cross-task

transferability.

5.2 Open Challenges

Despite the encouraging results, several important challenges remain before prompt-engineered LLM systems can be used as practical decision engines in next-generation wireless networks.

The first challenge is latency and instantaneous real-time decision making. Although the proposed frameworks avoid retraining and offer strong adaptability, practical wireless control often requires decisions within very short time scales. Resource allocation, scheduling, mobility handling, and interference management may need near-instantaneous responses, while forecasting pipelines may need to update continuously as new measurements arrive. In such settings, prompt construction, multi-agent interaction, iterative refinement, and API latency can become limiting factors [94]. Therefore, an important open challenge is how to preserve the reasoning and adaptability benefits of LLM-based agents while reducing end-to-end inference time to meet strict real-time requirements.

The second challenge is scalability to very large network sizes. The case studies in this thesis demonstrate the promise of decomposition and agent-based prompting, but future wireless systems may involve hundreds of users, many base stations, multiple frequency bands, dense sensor streams, and even large numbers of RIS elements or other controllable components [95]. As the problem dimension grows, the number of coupled variables, constraints, and interactions increases rapidly, making it difficult to represent the full system state efficiently within the prompt and to coordinate

multiple solver agents without excessive overhead. Extending the proposed methodology to such large-scale settings while maintaining solution quality, feasibility, and manageable latency remains a major challenge.

The third challenge is deployment in real online networks. Most current evaluations, including those in this thesis, are necessarily conducted in offline or semi-controlled settings. However, real wireless networks are dynamic, noisy, and operationally constrained. Measurements may be incomplete or delayed, traffic patterns may change abruptly, and decisions must interact with existing protocol stacks, hardware limitations, and network management policies [96]. For forecasting tasks, real deployments also require continual adaptation to changing environments and site-specific conditions. Consequently, moving from promising offline performance to robust online operation is still a significant challenge, and it requires not only better prompting strategies but also reliable system integration, monitoring, fallback mechanisms, and closed-loop validation.

In addition to these system-level challenges, several methodological limitations remain in the current study. First, all evaluated LLM backbones correspond to lightweight and low-latency model variants selected to maintain practical inference cost and deployment feasibility. Larger frontier-scale models may produce higher-quality pre-projection solutions, reduce the observed few-shot saturation effects, and handle higher-dimensional optimization spaces before encountering context-window limitations. However, this remains untested in the current work, and exploring such larger-scale models alongside their performance–latency trade-offs is an important direction for future investigation.

Second, the experiments were conducted using structured and systematically designed prompt templates, including variations involving physics-informed prompting, chain-of-thought reasoning, evaluator-guided refinement, decomposition strategies, and few-shot examples. However, no systematic study was conducted on how fine-grained phrasing changes, instruction ordering, or formatting variations affect performance. This absence of prompt sensitivity analysis limits the reproducibility claims of the current results, and further investigation is needed to fully characterize robustness across different prompting styles and LLM backbones.

Third, the few-shot retrieval memory used in both optimization and forecasting tasks was dynamically updated during execution by incorporating previously successful examples and reasoning traces. However, the retrieval pool itself was pre-built and fixed within the experimental pipeline, with no online update mechanism proposed for continuously evolving wireless environments. In live network deployments, channel statistics, traffic patterns, and environmental conditions drift over time, causing the memory bank to become stale. Future research should therefore investigate scalable online retrieval-update and memory management strategies suited to non-stationary wireless systems.

Finally, the optimization case studies in this thesis focused exclusively on sum-rate maximization under QoS constraints. Real 6G deployments, however, involve multi-objective trade-offs among competing metrics such as spectral efficiency, fairness, energy efficiency, and latency. The proposed framework has not been tested under such multi-objective settings, and extending it to handle these trade-offs remains an important open problem.

5.3 Future Directions

5.3.1 Adaptive Agent Architectures:

One important direction is the joint development of *adaptive agent architectures and unified cross-task prompting*. In the present work, decomposition and prompting strategies are designed manually according to the structure of each problem, such as block-wise optimization for resource management or trend–fluctuation separation for forecasting. A natural next step is to design a common prompting and orchestration framework that can adapt itself across different wireless tasks with minimal redesign [97]. Such a framework would determine how many agents are needed, what role each agent should play, how feedback should be exchanged, and which prompt modules should be activated for tasks such as optimization, forecasting, diagnosis, or control. By combining adaptive agent orchestration with reusable prompt primitives for constraint handling, temporal reasoning, self-evaluation, and decomposition, future systems can move closer to a general LLM-based wireless assistant that adapts to new problems without retraining.

5.3.2 Retrieval Augmented Memory

An additional direction is retrieval-augmented memory through RAG. As more operational data, solved instances, reasoning traces, and validated decision patterns are collected over time, retrieval-augmented generation can become a powerful mechanism for improving LLM-based wireless intelligence without retraining. For both RRM and forecasting tasks, a RAG-based memory can retrieve similar historical network states, forecast regimes, constraint templates, and previously successful agent interactions, then inject them into the prompt as relevant context. This would allow

the framework to accumulate experience over time, improve grounding, and make more informed decisions based on past scenarios while preserving the training-free nature of the overall approach [98].

5.3.3 Advanced Trustworthiness Analysis

Another key avenue is advanced trustworthiness analysis. Future research should develop richer uncertainty quantification methods, ensemble prompting strategies, self-consistency checks, and cross-agent verification protocols [99]. In forecasting, this could lead to calibrated predictive intervals and better handling of rare events or abrupt regime shifts. In resource allocation, it could support confidence-aware decision filtering, early detection of infeasible reasoning paths, and safer fallback to conventional optimization routines.

5.3.4 Real Time LLM Deployment

It is also important to study real-time deployment across broader wireless applications. Since the main appeal of the proposed framework is fast adaptation without retraining, future implementations should prioritize lower inference latency, reduced API cost, prompt compression, and smaller models that can operate closer to the network edge [100]. At the same time, these improvements should be validated across a wider range of wireless tasks beyond the case studies considered in this thesis, including multicast design, network slicing, interference coordination, semantic communications, anomaly detection, proactive network control, and joint forecasting–decision pipelines. Advancing along both dimensions together, namely, real-time deployability and broader task coverage, would provide stronger evidence that prompt-engineered

agentic LLM systems can serve as practical and versatile tools for real wireless networks.

In summary, the most compelling future vision is not a single specialized LLM solution for one problem, but a general agentic framework that uses prompt engineering, decomposition, retrieval, and verification to adapt across many wireless tasks. This thesis provides an initial step toward that vision by demonstrating the feasibility of the approach in both resource management and forecasting. The next stage is to make such systems more reliable, more efficient, and more principled so that they can operate as practical decision-support tools in real wireless networks.

Bibliography

[1] X. Lin, "The bridge toward 6g: 5g-advanced evolution in 3gpp release i9," *IEEE Communications Standards Magazine*, vol. 9, no. 1, pp. 28–35, 2025.

[2] W. Chen, X. Lin, J. Lee, A. Toskala, S. Sun, C. F. Chiasserini, and L. Liu, "5g-advanced toward 6g: Past, present, and future," *IEEE journal on selected areas in communications*, vol. 41, no. 6, pp. 1592–1619, 2023.

[3] X. Lin, L. Kundu, C. Dick, and S. Velayutham, "Embracing ai in 5g-advanced toward 6g: A joint 3gpp and o-ran perspective," *IEEE Communications Standards Magazine*, vol. 7, no. 4, pp. 76–83, 2023.

[4] B. Balasubramanian, E. S. Daniels, M. Hiltunen, R. Jana, K. Joshi, R. Sivaraj, T. X. Tran, and C. Wang, "Ric: A ran intelligent controller platform for ai-enabled cellular networks," *IEEE Internet Computing*, vol. 25, no. 2, pp. 7–17, 2021.

[5] C. Chatfield, *Time-series forecasting*. Chapman and Hall/CRC, 2000.

[6] S. Gowrishankar, "A time series modeling and prediction of wireless network traffic," *Computer Sciences and Telecommunications*, no. 2, pp. 40–52, 2008.

[7] X. Wang, J.-J. Ma, S. Wang, and D.-W. Bi, "Time series forecasting for energy-efficient organization of wireless sensor networks," *Sensors*, vol. 7, no. 9, pp. 1766–1792, 2007.

[8] I. F. Akyildiz, A. Kak, and S. Nie, "6g and beyond: The future of wireless communications systems," *IEEE access*, vol. 8, pp. 133 995–134 030, 2020.

[9] S. Bani Melhem, H. Tabassum, L. Chiaraviglio, and U. Trang Nguyen, "Joint coverage and emf compliance analysis in ris-assisted terahertz networks with optimized ris deployment," *IEEE Open Journal of the Communications Society*, vol. 6, pp. 8816–8834, 2025.

[10] M. Alizadeh and H. Tabassum, "Power control with qos guarantees: A differentiable projection-based unsupervised learning framework," *IEEE Transactions on Communications*, vol. 71, no. 8, pp. 4605–4619, 2023.

[11] Y. L. Lee, D. Qin, L.-C. Wang, and G. H. Sim, "6g massive radio access networks: Key applications, requirements and challenges," *IEEE Open Journal of Vehicular Technology*, vol. 2, pp. 54–66, 2020.

[12] H. He, X. Yu, J. Zhang, S. Song, and K. B. Letaief, "Cell-free massive mimo for 6g wireless communication networks," *Journal of Communications and Information Networks*, vol. 6, no. 4, pp. 321–335, 2021.

[13] P. Putranto, A. A. Hamza, S. Mabrouki, N. Armi, and I. Dayoub, "Reconfigurable intelligent surfaces for 6g and beyond: A comprehensive survey from theory to deployment," *arXiv preprint arXiv:2506.19526*, 2025.

[14] X. Fang, W. Feng, Y. Chen, N. Ge, and Y. Zhang, "Joint communication and sensing toward 6g: Models and potential of using mimo," *IEEE Internet of Things Journal*, vol. 10, no. 5, pp. 4093–4116, 2022.

[15] C. K. Sheemar, S. Solanki, E. Lagunas, J. Querol, S. Chatzinotas, and B. Ottersten, "Full duplex joint communications and sensing for 6g: Opportunities and challenges," *arXiv preprint arXiv:2308.07266*, 2023.

[16] M. A. Saeidi and H. Tabassum, "Resource allocation in cooperative mid-band/thz networks in the presence of mobility," *IEEE Transactions on Wireless Communications*, vol. 25, pp. 5046–5062, 2026.

[17] R. D. Yates, Y. Sun, D. R. Brown, S. K. Kaul, E. Modiano, and S. Ulukus, "Age of information: An introduction and survey," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 5, pp. 1183–1210, 2021.

[18] S. Elhoushy, M. Ibrahim, and W. Hamouda, "Cell-free massive mimo: A survey," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 1, pp. 492–523, 2021.

[19] E. Basar, "Reconfigurable intelligent surface-based index modulation: A new beyond mimo paradigm for 6g," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 3187–3196, 2020.

[20] C. Han, Y. Wang, Y. Li, Y. Chen, N. A. Abbasi, T. Kuerner, and A. F. Molisch, "Terahertz wireless channels: A holistic survey on measurement, modeling, and analysis," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 3, pp. 1670–1707, 2022.

[21] Z. Wang, J. Zhang, H. Du, W. E. Sha, B. Ai, D. Niyato, and M. Debbah, "Extremely large-scale mimo: Fundamentals, challenges, solutions, and future directions," *IEEE Wireless Communications*, vol. 31, no. 3, pp. 117–124, 2023.

[22] M. Wu, D. Wübben, A. Dekorsy, P. Baracca, V. Braun, and H. Halbauer, "Hardware impairments in millimeter wave communications using ofdm and sc-fde," in *WSA 2016; 20th International ITG Workshop on Smart Antennas*. VDE, 2016, pp. 1–8.

[23] H. Tataria, M. Shafi, A. F. Molisch, M. Dohler, H. Sjöland, and F. Tufvesson, "6g wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166–1199, 2021.

[24] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, 2018.

[25] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted mmse approach to distributed sum-utility maximization for a mimo interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331–4340, 2011.

[26] F. Liang, C. Shen, W. Yu, and F. Wu, "Towards optimal power control via ensembling deep neural networks," *IEEE Transactions on Communications*, vol. 68, no. 3, pp. 1760–1776, 2019.

[27] M. Eisen and A. Ribeiro, "Optimal wireless resource allocation with random edge graph neural networks," *ieee transactions on signal processing*, vol. 68, pp. 2977–2991, 2020.

[28] Y. Shen, Y. Shi, J. Zhang, and K. B. Letaief, "Graph neural networks for scalable radio resource management: Architecture design and theoretical analysis," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 1, pp. 101–115, 2020.

[29] H. Zhang, N. Yang, W. Huangfu, K. Long, and V. C. M. Leung, "Power control based on deep reinforcement learning for spectrum sharing," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4209–4219, 2020.

[30] Y. S. Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 10, pp. 2239–2250, 2019.

[31] M. Alizadeh and H. Tabassum, "Power control with qos guarantees: A differentiable projection-based unsupervised learning framework," *IEEE Transactions on Communications*, vol. 71, no. 8, pp. 4605–4619, 2023.

[32] M. Alizadeh, X. Mootoo, O. Waqar, and H. Tabassum, "Qos-aware deep unsupervised learning for star-ris assisted networks: A novel differentiable projection framework," *IEEE Wireless Communications Letters*, vol. 13, no. 9, pp. 2367–2371, 2024.

[33] Y. Zhang, L. Xiong, and J. Yu, "Deep learning based user association in heterogeneous wireless networks," *IEEE Access*, vol. 8, pp. 197 439–197 447, 2020.

[34] M. Chiang, P. Hande, T. Lan, C. W. Tan *et al.*, "Power control in wireless cellular networks," *Foundations and Trends® in Networking*, vol. 2, no. 4, pp. 381–533, 2008.

[35] Y.-A. Le Borgne, S. Santini, and G. Bontempi, "Adaptive model selection for time series prediction in wireless sensor networks," *Signal Processing*, vol. 87, no. 12, pp. 3010–3020, 2007.

[36] M. Di Mauro, G. Galatro, F. Postiglione, W. Song, and A. Liotta, "Multivariate time series characterization and forecasting of voip traffic in real mobile networks," *IEEE Transactions on Network and Service Management*, vol. 21, no. 1, pp. 851–865, 2023.

[37] M. Cheng, Z. Liu, X. Tao, Q. Liu, J. Zhang, T. Pan, S. Zhang, P. He, X. Zhang, D. Wang *et al.*, "A comprehensive survey of time series forecasting: Concepts, challenges, and future directions," *Authorea Preprints*, 2025.

[38] M. Kolambe and S. Arora, "Forecasting the future: A comprehensive review of time series prediction techniques," *Journal of Electrical Systems*, vol. 20, no. 2s, pp. 575–586, 2024.

[39] K. Cao, T. Hu, Z. Li, G. Zhao, and X. Qian, "Deep multi-task learning model for time series prediction in wireless communication," *Physical Communication*, vol. 44, p. 101251, 2021.

[40] V. Papastefanopoulos, P. Linardatos, T. Panagiotakopoulos, and S. Kotsiantis, "Multivariate time-series forecasting: A review of deep learning methods in

internet of things applications to smart cities," *Smart Cities*, vol. 6, no. 5, pp. 2519–2552, 2023.

[41] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[42] H. Naveed, A. U. Khan, S. Qiu, M. Saqib, S. Anwar, M. Usman, N. Akhtar, N. Barnes, and A. Mian, "A comprehensive overview of large language models," *ACM Transactions on Intelligent Systems and Technology*, vol. 16, no. 5, pp. 1–72, 2025.

[43] H. Zhou, C. Hu, Y. Yuan, Y. Cui, Y. Jin, C. Chen, H. Wu, D. Yuan, L. Jiang, D. Wu, X. Liu, J. Zhang, X. Wang, and J. Liu, "Large language model (llm) for telecommunications: A comprehensive survey on principles, key techniques, and opportunities," vol. 27, no. 3, pp. 1955–2005, 2025.

[44] M. Jin, S. Wang, L. Ma, Z. Chu, J. Y. Zhang, X. Shi, P.-Y. Chen, Y. Liang, Y.-F. Li, S. Pan *et al.*, "Time-llm: Time series forecasting by reprogramming large language models," *arXiv preprint arXiv:2310.01728*, 2023.

[45] Z. Pan, Y. Jiang, S. Garg, A. Schneider, Y. Nevmyvaka, and D. Song, "$\hat{s}$2 ip-llm: Semantic space informed prompt learning with llm for time series forecasting," in *Forty-first International Conference on Machine Learning*, 2024.

[46] Q. Dong, L. Li, D. Dai, C. Zheng, J. Ma, R. Li, H. Xia, J. Xu, Z. Wu, B. Chang *et al.*, "A survey on in-context learning," in *Proceedings of the 2024 conference on empirical methods in natural language processing*, 2024, pp. 1107–1128.

[47] J. Wu, S. Yang, R. Zhan, Y. Yuan, L. S. Chao, and D. F. Wong, "A survey on llm-generated text detection: Necessity, methods, and future directions," *Computational Linguistics*, vol. 51, no. 1, pp. 275–338, 2025.

[48] B. McKinzie, Z. Gan, J.-P. Fauconnier, S. Dodge, B. Zhang, P. Dufter, D. Shah, X. Du, F. Peng, A. Belyi *et al.*, "Mm1: methods, analysis and insights from multimodal llm pre-training," in *European Conference on Computer Vision*. Springer, 2024, pp. 304–323.

[49] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler *et al.*, "Emergent abilities of large language models," *arXiv preprint arXiv:2206.07682*, 2022.

[50] J. Dai, X. Pan, R. Sun, J. Ji, X. Xu, M. Liu, Y. Wang, and Y. Yang, "Safe rlhf: Safe reinforcement learning from human feedback," *arXiv preprint arXiv:2310.12773*, 2023.

[51] M. Stern, N. Shazeer, and J. Uszkoreit, "Blockwise parallel decoding for deep autoregressive models," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[52] J. Wei, X. Wang, D. Schuurmans, M. Bosma, F. Xia, E. Chi, Q. V. Le, D. Zhou *et al.*, "Chain-of-thought prompting elicits reasoning in large language models," *Advances in neural information processing systems*, vol. 35, pp. 24 824–24 837, 2022.

[53] C. Yang, X. Wang, Y. Lu, H. Liu, Q. V. Le, D. Zhou, and X. Chen, "Large language models as optimizers," in *The Twelfth International Conference on Learning Representations*, 2023.

[54] L. A. Mullen, K. Benoit, O. Keyes, D. Selivanov, and J. Arnold, "Fast, consistent tokenization of natural language text," *Journal of Open Source Software*, vol. 3, no. 23, p. 655, 2018.

[55] X. Li, S. Wang, S. Zeng, Y. Wu, and Y. Yang, "A survey on llm-based multi-agent systems: workflow, infrastructure, and challenges," *Vicinagearth*, vol. 1, no. 1, p. 9, 2024.

[56] P. J. Brockwell and R. A. Davis, *Introduction to time series and forecasting*. Springer, 2002.

[57] X. Liu, S. Gao, B. Liu, X. Cheng, and L. Yang, "Llm4wm: Adapting llm for wireless multi-tasking," *IEEE Transactions on Machine Learning in Communications and Networking*, vol. 3, pp. 835–847, 2025.

[58] J. Shao, J. Tong, Q. Wu, W. Guo, Z. Li, Z. Lin, and J. Zhang, "Wirelessllm: Empowering large language models towards wireless intelligence," *Journal of Communications and Information Networks*, vol. 9, no. 2, pp. 99–112, 2024.

[59] N. Sevim, M. Ibrahim, and S. Ekin, "Large language models (llms) assisted wireless network deployment in urban settings," in *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*. IEEE, 2024, pp. 1–7.

[60] H. Du, G. Liu, Y. Lin, D. Niyato, J. Kang, Z. Xiong, and D. I. Kim, "Mixture of experts for intelligent networks: A large language model-enabled approach," in *2024 International Wireless Communications and Mobile Computing (IWCMC)*, 2024, pp. 531–536.

[61] W. Lee and J. Park, "Llm-empowered resource allocation in wireless communications systems," *arXiv preprint arXiv:2408.02944*, 2024.

[62] X. Peng, Y. Liu, Y. Cang, C. Cao, and M. Chen, "Llm-optira: Llm-driven optimization of resource allocation for non-convex problems in wireless communications," *arXiv preprint arXiv:2505.02091*, 2025.

[63] F. Jiang, Y. Peng, L. Dong, K. Wang, K. Yang, C. Pan, D. Niyato, and O. A. Dobre, "Large language model enhanced multi-agent systems for 6g communications," *IEEE Wireless Communications*, vol. 31, no. 6, pp. 48–55, 2024.

[64] J. Tong, W. Guo, J. Shao, Q. Wu, Z. Li, Z. Lin, and J. Zhang, "Wirelessagent: Large language model agents for intelligent wireless networks," *arXiv preprint arXiv:2505.01074*, 2025.

[65] H. Li, M. Xiao, K. Wang, D. I. Kim, and M. Debbah, "Large language model based multi-objective optimization for integrated sensing and communications in UAV networks," *IEEE Wireless Commun. Lett.*, vol. 14, no. 4, pp. 979–983, 2025.

[66] H. Zhou, C. Hu, D. Yuan, Y. Yuan, D. Wu, X. Liu *et al.*, "Prompting wireless networks: Reinforced in-context learning for power control," *arXiv preprint arXiv:2506.06526*, 2025.

[67] W. Lee and J. Park, "LLM-empowered resource allocation in wireless communications systems," *IEEE Access*, vol. 14, pp. 15 260–15 272, 2026.

[68] K. Qiu, S. Bakirtzis, I. Wassell, H. Song, J. Zhang, and K. Wang, "Large language model-based wireless network design," *IEEE Wireless Commun. Lett.*, vol. 13, no. 12, pp. 3340–3344, 2024.

[69] H. Noh, B. Shim, and H. J. Yang, "Adaptive resource allocation optimization using large language models in dynamic wireless environments," vol. 74, no. 10, pp. 16 630–16 635, 2025.

[70] M. Sun, J. Hou, K. Qiu, K. Wang, X. Chu, and Z. Zhang, "LLM-based task offloading and resource allocation in satellite edge computing networks," pp. 1–6, 2026.

[71] S. Lian, J. Tong, J. Zhang, and L. Fu, "Intelligent channel allocation for ieee 802.11be multi-link operation: When MAB meets LLM," *IEEE Jrnl. on Sel. Areas in Commun.*, vol. 43, no. 11, pp. 3650–3665, 2025.

[72] H. Lee, M. Kim, S. Baek, W. Zhou, M. Debbah, and I. Lee, "Ai-driven decentralized network management: Leveraging multi-agent large language models for scalable optimization," vol. 63, no. 6, pp. 50–56, 2025.

[73] Z. Yan, H. Zhou, J. Pei, and H. Tabassum, "Hierarchical and collaborative LLM-based control for multi-uav motion and communication in integrated terrestrial and non-terrestrial networks," *arXiv preprint arXiv:2506.06532*, 2025.

[74] R. H. Shumway and D. S. Stoffer, "Arima models," in *Time series analysis and its applications: with R examples*. Springer, 2017, pp. 75–163.

[75] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "Lstm: A search space odyssey," *IEEE transactions on neural networks and learning systems*, vol. 28, no. 10, pp. 2222–2232, 2016.

[76] C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager, "Temporal convolutional networks for action segmentation and detection," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 156–165.

[77] X. Mootoo, H. Tabassum, and L. Chiaraviglio, "Emforecaster: A deep learning framework for time series forecasting in wireless networks with distribution-free uncertainty quantification," *IEEE Trans. on Network Science and Engineering*, vol. 13, pp. 1207–1225, 2026.

[78] D. Koutsoyiannis and A. Montanari, "Statistical analysis of hydroclimatic time series: Uncertainty and insights," *Water resources research*, vol. 43, no. 5, 2007.

[79] G. Jalori, P. Verma, and S. Ö. Arık, "Flairr-ts–forecasting llm-agents with iterative refinement and retrieval for time series," *arXiv preprint arXiv:2508.19279*, 2025.

[80] M. Tan, M. A. Merrill, V. Gupta, T. Althoff, and T. Hartvigsen, "Are language models actually useful for time series forecasting?" *Advances in Neural Information Processing Systems*, vol. 37, pp. 60 162–60 191, 2024.

[81] S. Medjkouh, E. Gönültaş, T. Goldstein, O. Tirkkonen, and C. Studer, "Unsupervised charting of wireless channels," in *2018 IEEE Global Communications Conference (GLOBECOM)*, 2018, pp. 1–7.

[82] P. Liang, J. Fan, W. Shen, Z. Qin, and G. Y. Li, "Deep learning and compressive sensing-based csi feedback in fdd massive mimo systems," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 8, pp. 9217–9222, 2020.

[83] R. He, B. Ai, A. F. Molisch, G. L. Stuber, Q. Li, Z. Zhong, and J. Yu, "Clustering enabled wireless channel modeling using big data algorithms," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 177–183, 2018.

[84] X. Liu, T. Chen, L. Da, C. Chen, Z. Lin, and H. Wei, "Uncertainty quantification and confidence calibration in large language models: A survey," in *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, 2025, pp. 6107–6117.

[85] L. Clarke, "Probability and measure, by patrick billingsley. pp 515.£ 15· 20. 1979. sbn 0 471 03173 9 (wiley)," *The Mathematical Gazette*, vol. 64, no. 430, pp. 293–294, 1980.

[86] R. A. Maronna, R. D. Martin, V. J. Yohai, and M. Salibián-Barrera, *Robust statistics: theory and methods (with R)*. John Wiley & Sons, 2019.

[87] M. Alizadeh, "Deep unsupervised learning for network resource allocation problems with convex and non-convex constraints," 2022-10-19.

[88] A. F. Ansari, L. Stella, C. Turkmen, X. Zhang, P. Mercado, H. Shen, O. Shchur, S. S. Rangapuram, S. P. Arango, S. Kapoor *et al.*, "Chronos: Learning the language of time series," *arXiv preprint arXiv:2403.07815*, 2024.

[89] A. Das, W. Kong, R. Sen, and Y. Zhou, "A decoder-only foundation model for time-series forecasting," *arXiv preprint arXiv:2310.10688*, 2023.

[90] X. Liu, J. Liu, G. Woo, T. Aksu, Y. Liang, R. Zimmermann, C. Liu, S. Savarese, C. Xiong, and D. Sahoo, "Moirai-moe: Empowering time series foundation models with sparse mixture of experts," *arXiv preprint arXiv:2410.10469*, 2024.

[91] S. B. Melhem, H. Tabassum, and L. Chiaraviglio, "Downlink coverage and exposure analysis in ris-assisted thz network with nakagami fading," in *ICC 2025 - IEEE International Conference on Communications*, 2025, pp. 6874–6879.

[92] Z. Yan, Y. Huang, J. Pei, H. Tabassum, and L. Chiaraviglio, "Emfusion: Conditional diffusion framework for trustworthy frequency selective emf forecasting in wireless networks," *arXiv preprint arXiv:2512.15067*, 2025.

[93] S. Adda, L. Chiaraviglio, D. Franci, C. Lodovisi, N. Pasquino, S. Pavoncello, C. Pedroli, and R. Pelosini, "High noon for mobile networks: Short-time emf measurements to capture daily exposure," *IEEE Transactions on Instrumentation and Measurement*, vol. 72, pp. 1–10, 2023.

[94] A. Agrawal, N. Kedia, A. Panwar, J. Mohan, N. Kwatra, B. Gulavani, A. Tumanov, and R. Ramjee, "Taming {Throughput-Latency} tradeoff in {LLM} inference with {Sarathi-Serve}," in *18th USENIX symposium on operating systems design and implementation (OSDI 24)*, 2024, pp. 117–134.

[95] S. Gupta, "Scalability and resilience in distributed llm training: A survey of modern techniques." *Journal of International Crisis & Risk Communication Research (JICRCR)*, vol. 8, 2025.

[96] U. Raza, A. Camerra, A. L. Murphy, T. Palpanas, and G. P. Picco, "Practical data prediction for real-world wireless sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 27, no. 8, pp. 2231–2244, 2015.

[97] B. Hayes-Roth, "An architecture for adaptive intelligent systems," *Artificial intelligence*, vol. 72, no. 1-2, pp. 329–365, 1995.

[98] M. Arslan, H. Ghanem, S. Munawar, and C. Cruz, "A survey on rag with llms," *Procedia computer science*, vol. 246, pp. 3781–3790, 2024.

[99] Y. Liu, Y. Yao, J.-F. Ton, X. Zhang, R. Guo, H. Cheng, Y. Klochkov, M. F. Taufiq, and H. Li, "Trustworthy llms: a survey and guideline for evaluating large language models' alignment," *arXiv preprint arXiv:2308.05374*, 2023.

[100] S. Sagi, "Optimizing llm inference: Metrics that matter for real time applications," *Journal of Artificial Intelligence & Cloud Computing*, vol. 446, pp. 2–4, 2025.

Appendix

This appendix presents the detailed prompt structures used in the LLM4RRM framework for Use-Case Study I (Joint Beamforming and Phase Shift Optimization in STAR-RIS Networks shown in section 3.2). Specifically, it illustrates the initial iteration ($t = 0$) prompt and output blocks for the solver agents LLM_1 and LLM_2, and the evaluator agent LLM_E, followed by the iterative refinement ($t = 1$) prompt and output blocks for the same agents and then the hyperparameter-tuning agent LLM_H is shown which is used after the final iteration.

$t = 0$, LLM_1 - Beamforming Agent - Prompt Block ($\Pi_1^{(0)}$)

Task Description $\mathcal{T}_1$:

You are OptiBeam–STAR, an expert in Wireless Communications and Resource Optimization specializing in STAR-RIS (Simultaneously Transmitting and Reflecting Reconfigurable Intelligent Surface) assisted MISO downlink systems.

Objective: Generate optimal network parameters that:

- Maximize total sum-rate (R_{total}). Satisfy QoS constraints: every user rate $\geq R_{\min}$ bps/Hz; Ensure all other constraints in (3.17)
- *Mathematical Framework Equations:* (3.16) and (3.17)

Optimization Variables (Outputs):

- $\mathbf{W}_r$: Beamforming matrix $M \times U$ real part, $\mathbf{W}_i$: Beamforming matrix $M \times U$ imaginary part, α: Power control transformation

Required Output Format:

{"Wr": [M*U real numbers], "Wi": [M*U real numbers], "alpha": [single real scalar]}

Response Format:

- Detailed reasoning showing optimization process. Channel analysis of the given **G** and **H** key entries. Strategy explanation for beamforming. Confidence statement about solution quality. JSON solution in exact format

System Parameters $\boldsymbol{x}$*:*

Base Station: M antennas, STAR-RIS: N elements, Users: U, BS→RIS channel **G**: shape $(N \times M)$, RIS→Users channel **H**: shape $(N \times U)$, Noise power: σ^2, Maximum transmit power: $P_{\max}$, Minimum rate requirement: $R_{\min}$

Few Shot Examples $\mathcal{S}_{\mathcal{M}}$*:*

1. For a certain **G** and **H**, the following solution achieved the best sum-rate:

{"Wr": [---], "Wi": [---], "alpha": [-], "a": [---], "phi_t": [---], "phi_r": [---]}

(Similarly, 3 or 4 more examples)

$t = 0$, LLM_1 - Beamforming Agent - Output Block

Output:

$\mathbf{y}_1^{(0)} = \{\mathbf{W}_r^{(0)}, \mathbf{W}_i^{(0)}, \alpha^{(0)}\}$

- {"Wr": [0.214, 0.062, -0.185, 0.491, -0.073, 0.331, -0.512, -0.208, 0.416, 0.309, 0.327, -0.294], "Wi": [-0.311, 0.102, 0.221, -0.377, 0.018, 0.205, 0.342, -0.194, 0.054, -0.211, -0.287, 0.163], "alpha": [0.812]}

Reasoning Trace:

$\text{RT}_0^{(1)}$: "The base station employs four transmit antennas (M=4) to serve U=3 users through the STAR-RIS. The channel matrix **G** shows dominant magnitudes at [0,3] and [4, 13], corresponding to rows counting toward antennas 3. Accordingly, the beamforming vectors $\mathbf{w}_u$ are aligned with the right singular vectors of **G** associated with the largest singular values. The power-scaling

factor $\alpha^{(0)} = 0.812$ ensures that $P_{\max} \leq 1$, while maintaining sufficient per-user SINR. Inter-user interference is mitigated by orthogonalizing the columns of $\mathbf{W}$ using QR-based normalization. The overall sum-rate improvement is projected at +12.4% compared to random initialization, with no QoS violations. *Confidence*: The beamforming vectors are well aligned with the dominant channel subspace, ensuring stable SINR distribution and balanced power usage across users."

$t = 0$, LLM_2 - RIS Configuration Agent - Prompt Block ($\Pi_2^{(0)}$)

Task Description $\mathcal{T}_2$:

You are OptiBeam–STAR, an expert in Wireless Communications and Resource Optimization specializing in STAR-RIS (Simultaneously Transmitting and Reflecting Reconfigurable Intelligent Surface) assisted MISO downlink systems.

Objective: Generate optimal RIS configuration parameters that:

- Maximize total sum-rate (R_{total}); Ensure all constraints in (3.17)
- *Mathematical Framework Equations:* (3.16) and (3.17)

Optimization Variables (Outputs):

- $\mathbf{a}$: Amplitudes (N values), ϕ_t: Transmit Phase Vector (N values in $[0, 2\pi]$), ϕ_r: Reflect Phase Vector (N values in $[0, 2\pi]$)

Required Output Format:

`{"a": [N real numbers], "phi_t": [N real numbers in [0,2pi]], "phi_r": [N real numbers in [0,2pi]]}`

Response Format:

- Detailed reasoning showing optimization process. Channel analysis of $\mathbf{G}$ and $\mathbf{H}$ key entries. Strategy explanation for RIS configuration. Confidence statement about solution quality. JSON solution in exact format

System Parameters $\boldsymbol{x}$:

Base Station: M antennas, STAR-RIS: N elements, Users: U, BS→RIS channel $\mathbf{G}$: shape $(N \times M)$, RIS→Users channel $\mathbf{H}$: shape $(N \times U)$, Noise power: σ^2, Maximum transmit power:

$P_{\max}$, Minimum rate requirement: $R_{\min}$

Few Shot Examples $\mathcal{S}_{\mathcal{M}}$*:*

1. For a certain $\mathbf{G}$ and $\mathbf{H}$, the following solution achieved the best sum-rate:

{"Wr": [---], "Wi": [---], "alpha": [-], "a": [---], "phi_t": [---], "phi_r": [---]}

(Similarly, 3 or 4 more examples)

$t = 0$, LLM_2 — RIS Configuration Agent — Output Block

Output:

$\mathbf{y}_2^{(0)} = \{a^{(0)}, \phi_t^{(0)}, \phi_r^{(0)}\}$

- {"a": [0.86, 0.79, 0.41, 0.55, 0.63, 0.72, 0.94, 0.88, 0.36, 0.47, 0.59, 0.62, 0.81, 0.91, 0.64, 0.58], "phi_t": [0.00, 0.94, 1.42, 2.97, 3.12, 0.25, 1.65, 0.81, 2.43, 1.98, 0.54, 0.77, 3.33, 2.56, 0.12, 1.90], "phi_r": [3.10, 0.88, 1.21, 0.54, 0.39, 2.84, 1.41, 2.93, 0.65, 1.22, 2.48, 3.05, 2.88, 1.70, 0.48, 0.31]}

Reasoning Trace:

$\text{RT}_0^{(2)}$: "The RIS comprises N=16 elements with significant gain observed in the 9th, 13th, and 14th columns of $\mathbf{G}$, and in the 11th and 1st rows of $\mathbf{H}$. Elements corresponding to these indices receive higher amplitude assignments ($a_n > 0.8$). Transmit and reflect phase profiles are designed to maximize $\mathfrak{Re}\{h_u^H \mathbf{\Theta} \mathbf{G} \mathbf{w}_u\}$ for all users simultaneously. Phases ϕ_t and ϕ_r are paired to satisfy the energy constraint $\phi_{t,n}^2 + \phi_{r,n}^2 = 1$ through a sigmoid projection. Phase alignment favors constructive addition toward user clusters on both sides of the STAR-RIS. Total per-user rate exceeds 3 bps/Hz with normalized energy efficiency of 0.97. *Confidence*: The selected amplitudes and phase shifts maximize coherent combining and maintain full energy conservation across transmission and reflection paths."

Evaluation

$$\mathbf{y}^{(0)} = [\mathbf{y}_1^{(0)}, \mathbf{y}_2^{(0)}], \quad r^{(0)} = \{F(\mathbf{x}, \mathbf{y}^{(0)}),\ \mathbf{f}_j(\mathbf{x}, \mathbf{y}^{(0)})\}$$

$t = 0$, LLM_E - Evaluator Agent - Prompt Block ($\Pi_E^{(0)}$)

Task Description $\mathcal{T}_E$:

You are OptiBeam–Evaluator, the expert assistant for STAR-RIS beamforming optimization. You analyze a STAR-RIS-aided MISO downlink system where:

- An M-antenna base station communicates with U single-antenna users via a STAR-RIS with N elements. All signals are relayed through the STAR-RIS; direct BS→user links are blocked. Each user u receives signals through the cascaded channel $\mathbf{h}_u^H \boldsymbol{\Theta} \mathbf{G}$ and its beamforming vector $\mathbf{w}_u$. Equations: (3.16) and (3.17)

Response Format:

1. ⟨**THOUGHT_PROCESS**⟩
   Analyze variable configurations, user rates, and violations; identify what is effective or problematic (e.g., power imbalance, phase misalignment, weak coupling). Indicate which parameters to adjust for improvement. Check the Reasoning Traces of each solver agent.
2. **FEEDBACK**
   Provide only a short, direct instruction (max 5 lines) or the phrase "No major changes needed." Do not restate variable values, rates, or channel data.

System Parameters $\boldsymbol{x}$:

Base Station: M antennas, STAR-RIS: N elements, Users: U, BS→RIS channel $\mathbf{G}$: shape $(N \times M)$, RIS→Users channel $\mathbf{H}$: shape $(N \times U)$, Noise power: σ^2, Maximum transmit power: $P_{\max}$, Minimum rate requirement: $R_{\min}$

Few Shot Examples $\mathcal{S}_\mathcal{E}$:

1. For a certain $\mathbf{G}$ and $\mathbf{H}$, the following solution achieved the best sum-rate:

`{"Wr": [---], "Wi": [---], "alpha": [-], "a": [---], "phi_t": [---], "phi_r": [---]}`

(Similarly, 3 or 4 more examples)

$t = 0$, LLM_E - Evaluator Agent - Output Block

Parsed Outputs:

$\Delta_1^{(1)}$: "User 3 exhibits a residual QoS gap of 0.21 bps/Hz. Increase beamforming gain toward $\mathbf{h}_3$ by redistributing $\approx 8\%$ of total power from the dominant eigenvector associated with User 1. Maintain $\|\mathbf{W}\|_F^2 = 1$ after rescaling and keep inter-user correlation below 0.15."

$\Delta_2^{(1)}$: "Phase alignment drift detected for RIS element indices $\{3, 7, 12\}$ with cumulative phase error $\approx 0.28\pi$. Rotate $\phi_t[3] + 10° \rightarrow \phi_t[12] \rightarrow \phi_r[12] - 8°$, and slightly increase β_t for elements $n \in \{7, 12\}$ by 0.05 to restore constructive coupling with $\mathbf{W}^{(1)}$."

$t = 1$, LLM_1 - Beamforming Agent - Prompt Block ($\Pi_1^{(1)}$)

Feedback:

$\Delta_1^{(1)}$: "User 3 exhibits a residual QoS gap of 0.21 bps/Hz. Increase beamforming gain toward $\mathbf{h}_3$ by redistributing $\approx 8\%$ of total power from the dominant eigenmode associated with User 1. Maintain $\|\mathbf{W}\|_F^2 = 1$ after rescaling and keep inter-user correlation below 0.15."

Previous Output and History:

and history $\mathcal{H}_1^{(1)} = \mathcal{H}_1^{(0)} \cup \{\Delta_1^{(1)},\ (\mathbf{y}^{(0)},\ \text{RT}_1^{(0)})\}$, where $\mathcal{H}_1^{(0)} = \{\mathbf{x},\ \mathcal{S}_M,\ \mathcal{T}_1\}$

$t = 1$, LLM_1 - Beamforming Agent - Output Block

Solution:

{ "Wr": [0.54, 0.26, 0.11, 0.39, 0.18, 0.27, 0.21, 0.09, 0.43, 0.16, 0.33, 0.22],
"Wi": [0.33, -0.42, 0.18, 0.47, 0.29, -0.36, -0.15, 0.11, 0.51, 0.12, 0.18, -0.27],
"alpha": 1.12 }

Reasoning Trace:

LLM_1 incorporated evaluator feedback $\Delta_1^{(1)}$ by reallocating approximately 8% of total transmit power toward User 3's subspace. Eigen-decomposition of $(G^H G)$ identified the top three dominant spatial modes with singular values $\{5.72, 4.31, 3.98\}$, where the third mode corresponded primarily to User 3's effective channel direction $\mathbf{h}_3^H G$. Columns of $\mathbf{W}$ were recomputed via the

updated sigmoid factor $\sigma(\alpha^{(1)}) = 0.75$.

$t = 1$, LLM$_2$ - RIS Configuration Agent - Prompt Block ($\Pi_2^{(1)}$)

Feedback:

$\Delta_2^{(1)}$: "Phase alignment drift detected for RIS element indices $\{3, 7, 12\}$ with cumulative phase error $\approx 0.28\pi$. Rotate $\phi_t[3] \rightarrow \phi_t[3]{+}10°$, $\phi_t[12] \rightarrow \phi_t[12]{-}8°$, and slightly increase β_t for elements $n \in \{7, 12\}$ by 0.05 to restore constructive coupling with $\mathbf{W}^{(1)}$."

Previous Output and History:

Current Preceding Agent Solution $\mathbf{y}_1^{(1)}$ and history $\mathcal{H}_2^{(1)} = \mathcal{H}_2^{(0)} \cup \{\Delta_2^{(1)},\ (\mathbf{y}^{(0)},\ \mathrm{RT}_2^{(0)})\}$, where $\mathcal{H}_2^{(0)} = \{\mathbf{x},\ \mathcal{S}_M,\ \mathcal{T}_2\}$

$t = 1$, LLM$_2$ - RIS Configuration Agent - Output Block

Output:

$\mathbf{y}_2^{(1)} = \{$

$a^{(1)} = [0.88, -0.09, 0.71, 0.27, -0.44, 0.76, 0.11, 0.61,$
$0.79, -0.12, 0.67, 0.14, 0.56, 0.40],$

$\phi_t^{(1)} = [1.12, 2.04, 0.86, 3.01, 2.33, 0.59, 1.78, 2.51,$
$0.95, 1.47, 2.82, 0.32, 1.16, 2.44, 0.73, 2.09],$

$\phi_r^{(1)} = [0.18, 1.74, 2.63, 0.71, 1.92, 2.48, 0.34, 1.51,$
$2.08, 0.45, 1.63, 2.22, 0.27, 1.95, 2.41, 0.66]\}$

Reasoning Trace:

LLM$_2$ applied evaluator feedback $\Delta_2^{(1)}$ by rotating $\phi_t[3] \rightarrow \phi_t[3]{+}10°$ and $\phi_t[12] \rightarrow \phi_r[12]{-}8°$, while increasing β_t for elements $\{7, 12\}$ by 0.05. The adjusted amplitudes maintained $\beta_{t,n}^2{+}\beta_{r,n}^2 = 1$ for all n and improved phase coherence for Users 2 and 3. As a result, the reflected and transmitted signals achieved stronger constructive alignment, raising the average SINR by about 1.5 dB and eliminating the previous QoS gap for User 3. All $R_u \geq R_{\min}$ were satisfied with minimal interference leakage.

Evaluation

$$\mathbf{y}^{(1)} = [\mathbf{y}_1^{(1)},\ \mathbf{y}_2^{(1)}], \quad r^{(1)} = \{F(\mathbf{x}, \mathbf{y}^{(1)}),\ \mathbf{f}_j(\mathbf{x}, \mathbf{y}^{(1)})\}$$

$t = 1$, LLM$_E$ - Evaluator Agent - Prompt Block ($\Pi_E^{(1)}$)

Updated solutions from both agents: $\mathbf{y}_1^{(1)} = \{\mathbf{W}^{(1)}, \alpha^{(1)}\}$ and $\mathbf{y}_2^{(1)} = \{a^{(1)}, \phi_t^{(1)}, \phi_r^{(1)}\}$, along with performance summary $r^{(1)} = [F(\mathbf{x}, \mathbf{y}^{(1)}),\ \mathbf{f}_j(\mathbf{x}, \mathbf{y}^{(1)})]$ and current Reasoning Traces $\mathrm{RT}^{(1)}$

Historical record: $\mathcal{H}_E^{(1)} = \mathcal{H}_E^{(0)} \cup \{\Delta^{(1)},\ (r^{(0)}, \mathbf{y}^{(0)}, \mathrm{RT}_E^{(0)})\}$, where $\mathcal{H}_E^{(0)} = \{\mathbf{x},\ \mathcal{S}_F,\ \mathcal{T}_E\}$

$t = 1$, LLM$_E$ - Evaluator Agent - Output Block

Output Feedback:

$\Delta^{(2)}$: "Overall convergence is stable; average QoS margin = +0.08 bps/Hz. Minor imbalance detected between reflected and transmitted paths: $\bar{\beta}_t - \bar{\beta}_r \approx 0.07$ leading to slightly weaker illumination for Users 1 and 3.

$\Delta_2^{(2)}$: Recommend a mild correction in the next iteration: increase β_t for high-angle elements $\{5, 9, 14\}$ by 0.03, and apply a small global phase offset $\phi_t \to \phi_t + 4°$ to restore symmetry across STAR-RIS sides."

LLM$_H$ - Hyperparameter Controller (OptiBeam—Hyper) - Prompt Block

Task Description:

You are OptiBeam—Hyper, the adaptive controller responsible for dynamically tuning the projection-layer hyperparameters of the STAR-RIS optimization framework. Your goal is to ensure that the iterative projection process (3.26) converges quickly, remains stable, and enforces all feasibility constraints $c_j(\mathbf{x}, \mathbf{y}) \geq p_j$. You must infer the next hyperparameter set $\theta^{(t+1)} = \{\mu^{(t+1)}, \gamma^{(t+1)}\}$ based on the provided projection steps, feasibility indicators, and convergence trends.

System Context:

- System parameters $\mathbf{x}$ describe the MISO downlink configuration (BS, RIS, and users). $\tilde{\mathbf{y}}$ is the approximate solution obtained after T reasoning iterations of the Multi-Agent LLM framework.

- (3.10) representing all prior parameter settings and feasibility outcomes. $sp^{(t)}$ represents the most recent few projection-step summaries, such as norm differences or constraint residuals.

Objectives:

- Analyze the projection behaviour up to iteration t. Determine if convergence is stable, oscillatory, or slow. Suggest updated hyperparameters $\{\mu^{(t+1)}, \gamma^{(t+1)}\}$ that stabilize oscillations (if detected), accelerate convergence when feasible, and preserve feasibility of $R_u(\mathbf{x}, \mathcal{T}(\tilde{y})) \geq R_{\min}$.

Response Format:

1. ⟨**THOUGHT_PROCESS**⟩: Provide a concise explanation of observed convergence (e.g., residual patterns, constraint violation magnitude, oscillation trends). Explain how each proposed hyperparameter addresses these behaviors.
2. ⟨**UPDATED_HYPERPARAMETERS**⟩: Output the new hyperparameter set in the format: { `"mu": value, "gamma": value` }

System Parameters x:

Base Station: M antennas, STAR-RIS: N elements, Users: U, BS→RIS channel $\mathbf{G}$: shape $(N \times M)$, RIS→Users channel $\mathbf{H}$: shape $(N \times U)$, Noise power: σ^2, Maximum transmit power: $P_{\max}$, Minimum rate requirement: $R_{\min}$, Iteration index: l, Projection-step observations: $sp^{(t)}$ (recent projection summaries, e.g., residual norms), Approximate solution from reasoning loop: $\tilde{\mathbf{y}}$

LLM$_H$ - Hyperparameter Controller (OptiBeam—Hyper) - Output Block

Output:

$\theta^{(t+1)} = \{\mu^{(t+1)},\ \gamma^{(t+1)}\}$

Generated JSON:

{ `"mu": 0.078, "gamma": 0.62` }

Reasoning Trace:

$\mathrm{RT}_H^{(t)}$: Residual trends in $sp^{(t)}$ indicate mild oscillations near feasibility: constraint violation $R_u(\mathbf{x}, \mathcal{T}(\tilde{y})) \geq R_{\min}$ plateaued between 0.012–0.018 with alternating overshoot. This suggests an excessive step size and low damping. To improve stability, the step size μ is slightly reduced (from 0.1 to 0.078), and momentum γ is moderately increased to 0.62 to preserve acceleration while suppressing divergence.