RMultiplex200K: Toward Reliable Multimodal Process Supervision for Visual Language Models on Telecommunications

Supplementary Material

Contents

A. Related Work

Large language models (LLMs) has the **step-by-step reasoning** ability [21, 38, 47, 49], which enables them to solve complex problems by generating a sequence of intermediate steps, where each step addresses a simpler subproblem, progressively advancing toward the final solution. For instance, with Chain-of-Thought (CoT) prompting [49], LLMs answer a question by generating intermediate reasoning steps, with each step represented as a thought — a sequence of textual descriptions detailing the solution of a sub-problem — leading toward a final solution. Encouraged by the progress made in LLMs, subsequent work [14, 38, 39, 63, 64] on multimodal large langauge models (VLMs) [4, 27, 28, 45] that receive both text and images as input has focused on enhancing reasoning in multimodal contexts. Therefore, ensuring the multi-step reasoning ability of VLMs is an urgent necessity to enable their application in complex tasks.

Toward this goal, apart from prompting methods, advancements in LLMs and VLMs can be categorized into three directions. First, pre-training mechanism [1, 3, 43] involves training models on extensive textual datasets to develop inherent reasoning abilities. Similarly, pre-training VLMs, such as Qwen-VL and LLaMA 3.2, equips them with reasoning skills from multimodal inputs, like visual-text pairs. Second, the fine-tuning methods [15, 56, 58, 60] are especially prominent, as they adapt the language model using targeted datasets, enabling specialization in specific reasoning tasks. For example, MetaMath [58] fine-tunes LLaMA-2 on a new dataset, MetaMathQA, which is crafted by rephrasing questions from multiple perspectives to support comprehensive mathematical reasoning. The core of fine-tuning usually lies in constructing high-quality question-response pair datasets with a chain-of-thought reasoning process.

Recent advancements in improving the reasoning abilities of LLMs for reliable problem-solving have focused on **verification**. This approach addresses the inherent issues of depending on the top-1 result, which often lacks reliability. By reranking candidate responses, verification models enhance the accuracy and consistency of outputs. Moreover, these models contribute valuable feedback, which is crucial for refining and boosting the performance of LLMs [24, 46, 53]. LLMs can be fine-tuned to serve as two types of verifiers: outcome reward models (ORMs) [23, 24, 30, 46, 53] and process reward models (PRMs). ORMs assign a confidence score to the entire solution, while PRMs assess the reasoning path step-by-step. For example, when an LLM generates an answer comprising multiple steps toward a solution, an ORM evaluates the complete answer and outputs a single score, whereas a PRM assigns a correctness score to each individual reasoning step. PRMs offer several advantages, including the ability to provide precise feedback by pinpointing the exact location of errors, which is highly valuable for reinforcement learning and automatic correction. Additionally, PRMs resemble human judgment: if a reasoning step contains an error, the likelihood of an incorrect final result increases.

A.1. Why Telecommunications Matter?

As discussed above, significant progress in developing both verification and complex reasoning abilities for problem-solving in large models is limited to (1) language models that process textual input and (2) applications within the mathematical domain. Thus, we focus on the multimodal large language models (VLMs) on *Telecom* for the following reasons.

First, as artificial intelligence (AI) for science receive more attention, many works have been proposed to applying large models in Mathematics, Physics [12, 40], Chemistry [50], Biology [61], Environmental Science [55], Neuroscience [6], and Healthcare [48]. To the best of our knowledge, no comprehensive analysis has been conducted on whether large models can be effectively applied to solve specific and practical scientific problems in *Telecom* [18], a crucial field with wide-ranging impact. Thus, our paper aims to contribute a first step by offering researchers a snapshot of whether large models can be applied to solve problems in *Telecom* and to what extent.

Second, existing research [5, 25, 34, 42, 52, 54, 65, 66] on the applications of large models in *Telecom* primarily focuses on general question answering rather than addressing specific problems through step-by-step reasoning. Additionally, these approaches train or fine-tune LLMs using textual datasets. For instance, SPEC5G [20] focuses on designing a knowledge-based Q&A system for network protocols, NetEval [37] trains LLMs for network operations, and TeleQnA [33] provides a benchmark dataset to assess LLM performance on *Telecom* knowledge. None of the existing work has kept pace with the recent advancements in the mathematical reasoning capabilities of LLMs. Specifically, for practical and specialized problems in *Telecom*, such as channel capacity calculation, signal-to-noise ratio computation, and waveform analysis, it remains unclear whether large models can perform step-by-step reasoning to reach a solution and whether process supervision can ensure reliable problem-solving.

Third, unlike mathematical problems that are typically expressed through textual sentences, *Telecom* problems often include supplementary images that provide essential information alongside the text. Consequently, this domain introduces the challenge of implementing step-by-step reasoning and verification in VLMs. Even though the MMMU [59] dataset includes the Tech & Engineering subject, where questions are paired with images, it lacks enhancements for multi-step reasoning and does not provide correctness

Table 4. Comparison of our *RMultiplex200K* dataset with other state-of-the-art related datasets or works.

	Text	Image	Step-by-step	Process Supervision	Annotation	Challenging
			Mathematics			
GSM8K [10]	✓					Easy
MATH [17]	✓		✓			Medium
TheoremQA [9]	✓	✓	✓			Hard
WizardMATH [32]	✓		✓			Medium
MathInstruct [60]	✓		✓			Medium
MetaMath [58]	✓		✓			Medium
PRM800K [24]	✓		✓	✓		Medium
MATH-SHEPHERD [46]	✓		✓	✓	Auto	Medium
MATHVISTA [31]	✓	✓				Hard
			Telecommunications			
MMMU [59]	✓	✓	✓$(partial)$			Hard
SPEC5G	✓					Easy
NetEval [37]	✓					Easy
TeleQnA [33]	✓					Easy
NetLLM [51]	✓	✓				Hard
Ours	✓	✓	✓	✓	Auto	Hard

scores for process supervision. Furthermore, the problems in MMMU may not be challenging enough to effectively evaluate VLMs in practical scientific applications. Therefore, by analyzing VLMs in the context of *Telecom* problem-solving, we aim to: 1) comprehensively evaluate VLMs in a domain that features challenging scientific problems, domain-specific concepts, and complex reasoning logic, 2) provide multimodal problems where both images and text are essential, making visual understanding a critical component of problem-solving, and 3) demonstrate the effectiveness of process supervision in enhancing VLM performance.

Finally, we present a comparison between our *RMultiplex200K* dataset and other state-of-the-art or related datasets in Tab. 4.

B. Data Details

B.1. Overview

RMultiplex200K comprises a total of 7,000 problems, with 5,000 designated for training and 2,000 for testing. These problems are categorized into five main areas: Wireless Communication (WC), Satellite Communication (SC), Network Theory and Optimization (NTO), Information Security and Encryption (ISE), and Communication Signal Processing (CSP). Specifically, WC includes five subcategories, SC has four subcategories, NTO features five subcategories, and both ISE and CSP are divided into five subcategories each. The number of training problems for the five categories is distributed as 1,300, 600, 1,400, 650, and 1,050, respectively. The corresponding number of test problems is 520, 240, 560, 260, and 420, respectively. The average number of reasoning steps required to solve each problem in these categories is 9, 8, 11, 9, and 12, respectively.

Wireless Communication (WC) contains five categories:

- Wireless Channel Characteristics and Modeling.
- Modulation and Coding Techniques.
- Advanced Antenna and Spatial Techniques.
- Multicarrier and Spread Spectrum Techniques.
- Multiuser and Networked Wireless Systems.

Satellite Communication (SC) contains four categories:

- Satellite Orbits, Launch, and Operations.
- Satellite Hardware and System Design.
- Satellite Communication and Link Design.
- Applications of Satellite Technology.

Network Theory and Optimization (NTO):

- Graph Theory in Networks.
- Resource Allocation and Scheduling.
- Optimization Techniques in Networks.
- Queuing Theory and Network Performance Analysis.
- Network Control and Stability.

Information Security and Encryption (ISE) contains five categories:

- Foundations of Cryptography and Number Theory.
- Symmetric and Asymmetric Encryption Techniques.
- Cryptographic Data Integrity and Authentication.
- Trust and Key Management Mechanisms.
- Network and Internet Security.

Communication Signal Processing (CSP) contains five categories:

- Signal Representation and Analysis.
- Transform Techniques and System Analysis.
- Filter Design and Implementation.
- Sampling, Conversion, and Multirate Signal Processing.
- Stochastic Signal Processing and System Design.

For these categories, the number of generated samples labeled with step-wise correctness scores for training is 48,000, 15,000, 63,000, 22,000, and 52,000, respectively. For the *RMultiplex200K* testbed, we generated 16,600, 5,400, 24,200, 8,700, and 21,800 samples for these categories, respectively.

B.2. Instructions

Data Privacy: Data is collected from open-source and publicly available resources, including quizzes, textbooks, and online documents. While using Mathpix [35] for data extraction, we strictly comply with the copyright and licensing regulations of each source. More importantly, the dataset is used solely for research purposes, with no financial gains involved.

Data Format: All textual data in *RMultiplex200K* is formatted in LaTeX style, while images are stored in the ".jpg" format. We ensure that each problem includes one or more images and that the difficulty level meets or exceeds college standards. Additionally, a dedicated researcher is responsible for verifying the accuracy of the data by comparing the collected LaTeX problems with the original sources.

Data Structure: The data structure includes two types: a solution sample and a regular sample. A solution sample comprises the question, the image, the raw answer, and the decomposed and generated reasoning steps, each with labeled correctness scores. In contrast, a regular sample contains the question, the image, one reasoning step, the previous steps, and a correctness label. Therefore, in our *RMultiplex200K* database, the data is stored as solution samples with the "json" format, shown as Fig. 11.

Data Description: To save the json file, the naming conventions follow "category-subfield-ID.json". The corresponding images are stored in a folder named "category-subfield-ID-images.json". It is important to note that most of the subfield components in the solution samples are missing. This is because, during data extraction from the documents, identifying the correct subfield for the data has proven challenging. This issue will be addressed in the future.

```
{
  # Source of this solution sample
  # TYPE should be a quiz, book, paper, or document.
  # NAME should be the name of the TYPE
  # ID is to distinguish the data source
  # URL is to access the data source
  "source": <TYPE>-<NAME>-<ID>-<URL>,
  # Data Collection Timeline
  "timestamp": <Year>-<Month>-<Day>-<H-M-S>,
  # Category:
  # Wireless Communication (WC), Satellite Communication (SC),
  # Network Theory and Optimization (NTO),
  # Information Security and Encryption (ISE), and183
  # Communication Signal Processing (CSP)
  "category": [Str],
  # Sub-field of the Category
  "sub-field": [Str],
  # Metadata of a solution sample
  "SolutionSamples": {
      # Text of the problem being solved
      "question": [Str],
      # Image of the problem being solved
      "image": [Str]
      # Ground truth solution
      "ground_truth_answer": [Str],
      # Ground truth reasoning step
      "ground_truth_steps": [List],
      # Ground truth images
      #   - Each step can contain many images
      "ground_truth_images": [[List]],
      # Ground truth solution
      "ground_truth_solution": [Str],
      # Large models used by ApPA
      "ApPA_model": [Str],

      # Generated reasoning paths
      #  - Each item is a nested list in which
      #   - each element is a list containing one or many
      #   - reasoning step for the corresponding step
      "reasoning_paths": [List[List]],
    },
  # Correctness scores generated by the ApPA
  "label": {
    # Scores for the Ground truth steps
    "ground_truth_scores": [List],
    # Generated correctness scores
    #  - Same structure as the 'reasoning_paths'
    #  - Each term is a score of the step
    "reasoning_scores": [List[List]],
  },
}
```

Figure 11. Illustration of the structure of the solution sample in the *RMultiplex200K*.

C. Details of the ApPA

C.1. Algorithm Design

The implementation of the automatic plan-based process annotation (*ApPA*) mechanism relies on the advanced VLMs, GPT-4o mini, which is prompted for answer decomposition, plan summarization, and reasoning generation. The algorithm of plan-based reasoning generation with Monte Carlo Tree Search (MCTS) is presented in Algorithm 1.

C.2. Prompts of ApPA

Prompt for the Answer decomposition $\mathbb{I}_1$:

- System Prompt. You are an expert in breaking down an answer to a *Telecom* question into multiple logical reasoning steps without changing the answer's words or sentences. Carefully read the given question and the answer, then decompose the answer into individual reasoning steps. Ensure each step is not small and thus contains a complete computation, analysis, or design process. Each step should be strictly extracted from the given answer without making any changes. Please only add the

Algorithm 1: Process annotation in *ApPA*

Input: VLM $p_{\boldsymbol{\theta}}$, Input $(\mathrm{Q}, \mathrm{A}, \mathrm{V})$.
Output: Optimized $\mathbf{P}$.

```
// Answer decomposition
```
$\boldsymbol{z}_{1\ldots n} \sim p_\theta\left(\boldsymbol{z}_{1\ldots n} | \mathbb{I}_1\left(\mathrm{Q}, \mathrm{A}, E\right)\right)$
```
// Plan generation
```
for $i \leftarrow 1$ ***to*** n **do**
  $\psi_i \sim p_\theta\left(\psi_i | \mathbb{I}_2\left(\mathrm{Q}, \mathrm{V}, \boldsymbol{z}_{1\ldots i}, S\right)\right)$
end
```
// Reasoning generation
```
$\boldsymbol{z}'_0 = (\mathrm{Q}, \mathrm{A}, \mathrm{V}),\ j = 0$
while *not* $\boldsymbol{z}'_j$ *reaches solution* **do**
  Get indexes $\boldsymbol{J}$ of child nodes for $\boldsymbol{z}'_j$
  if $|\boldsymbol{J}| == M$
    `// Selection of MCTS`
    Compute the UCT scores $c_{j+1'}, \forall j+1' \in \boldsymbol{J}$
    Select node $j+1^*$ with the best UCT score
    Add the selected node as $\boldsymbol{z}'_{j+1} = \boldsymbol{z}'_{j+1*}$
  else
    `// Expansion of MCTS`
    Generate M child nodes for j based on ψ_{j+1}
    $\boldsymbol{z}'_{j+1} \sim p_\theta\left(\boldsymbol{z}'_{j+1} | \mathbb{I}_3\left(\mathrm{Q}, \mathrm{V}, \boldsymbol{z}'_{1\ldots j}, \psi_{j+1}\right)\right)$
    Randomly select node from M child nodes as $\boldsymbol{z}'_{j+1}$
    break
  end
  $j = j + 1$
end
```
// SimulationRollOut of MCTS
```
while *not* $\boldsymbol{z}'_j$ *reaches solution* **do**
  `// Generate the next reasoning step with VLM`
  $\boldsymbol{z}'_{j+1} \sim p_\theta\left(\boldsymbol{z}'_{j+1} | \mathbb{I}_3\left(\mathrm{Q}, \mathrm{V}, \boldsymbol{z}'_{1\ldots j}, \psi_{j+1}\right)\right)$
  $j = j + 1$
end
▷ Backpropagate $\mathbf{1}(y, \boldsymbol{z}_n)$ to visited nodes.

Step ID before each decomposed step, such as 'Step 1:' or 'Step 2:'.

- $\mathbb{I}_1$: {Question}$\backslash n \backslash n${Question}$\backslash n \backslash n$Please decompose the given answer into logical reasoning steps. Please ensure: 1). Each step should not be small but large enough to only present the complete logic and contain a complete computation, analysis, or reasoning. 2). Use as few reasoning steps as possible. 3). Be careful not to make each step so small that it contains only a single calculation or a simple statement. 4). The content of each step should directly be extracted from the given answer without making any changes. 5). Do not change the words or sentences of the answer while decomposing it into steps.

Prompt for the Plan generation $\mathbb{I}_2$:

- System Prompt. You are an expert in identifying, extracting, and summarizing the plan that underpins one reasoning step. The summarized plan should be a general-purpose reasoning instruction and, thus, is a high-level, question-agnostic principle. Please get such a plan containing the highest-level ideas, principles, rules, or theorems from the given reasoning step. Start by reviewing the given question and any previous reasoning steps already taken, then directly summarize the plan of the given reasoning step. Please summarize the plan directly and briefly, avoiding including the specific contents of the given question or any reasoning steps.
- $\mathbb{I}_1$: {Question}$\backslash n \backslash n$Answer:$\backslash n${Existing Steps}$\backslash n \backslash n$Let's summarize the plan of the Step {} and directly generate the plan, which is a brief, high-level, question-agnostic principle, without including any question or reasoning step content.

Prompt for the Plan generation $\mathbb{I}_3$:

- System Prompt. You are an expert in following the given plan to generate the logical reasoning step for solving complex problems clearly and precisely. Your task is to produce a reasoning step based on a high-level, question-agnostic plan provided to you. Carefully review the given question and any prior reasoning steps. Then, generate a reasoning step that logically follows the plan, incorporating all necessary principles, calculations, or ideas to solve the problem effectively. Ensure your output is concise, focused, and directly follows the plan without adding unnecessary context or framing.
- $\mathbb{I}_1$: {question}$\backslash n \backslash n$Answer:$\backslash n${Existing Steps}$\backslash n$Plan: {plan}$\backslash n \backslash n$Follow the plan to directly generate reasoning step {} containing concise computation, exact equations, and a clear conclusion or an exact solution. Keep the reasoning step short and as brief as possible without introducing additional analysis or explanation.

C.3. Full Process of *ApPA*

In Fig. 12, we present the complete process of Fig. 4 from the main content. This figure illustrates the raw answer, the decomposed reasoning steps, and the corresponding plans summarized by GPT-4o mini. The details of this figure align with Algorithm 1, which illustrates one iteration of the MCTS process in our *ApPA*. As shown by the reasoning steps in the gray boxes, for the *Telecom* problem, even the advanced GPT-4o mini generally produces incorrect solutions when the plan is not included in the prompt. This highlights the importance of prompting VLMs with plans during process annotations.

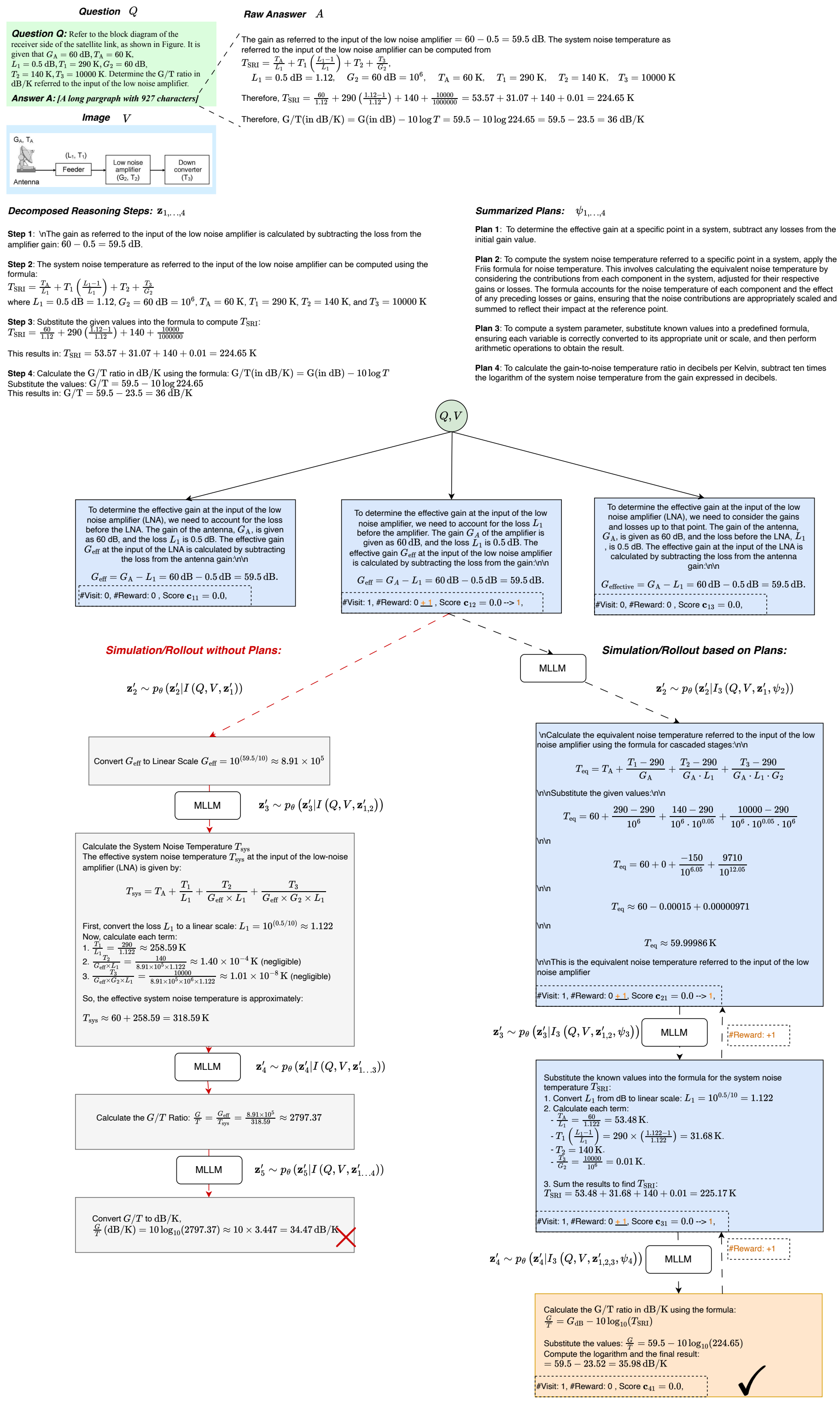


Figure 12. Illustration of an MCTS process of *ApPA*. We specifically present a reasoning path generated by GPT-4o mini, which does not utilize the plans in the prompt.

D. Details of TC-NAVIGATOR

D.1. Implementations

This section explains how to apply *TC-NAVIGATOR* to train a reward model using Qwen-VL-7B, Qwen2-VL-72B, and Llama-3.2-11B. To incorporate the QA-ViT [13] into the visual encoder as illustrated in Fig. 5, we heavily rely on the source code of the package *transformers* and the pre-trained models of the Hugging Face. The implementation details are presented as follows:

Language encoder. The language encoder of RA1 encodes each reasoning step, formatted as a textual description, into a single embedding. Specifically, it receives inputs, including the `reasoning_instruction`, which contains the question and the reasoning steps generated so far. This encoder can be flexibly chosen from options such as the pre-existing LLM's encoder, embeddings, or a designated language model. Here, we directly use the textual transformer as the base model. The encoder first uses the tokenizer of the base model to obtain the embeddings of the inputs, producing a tensor referred to as `reasoning_states` of size $(i+1)\times d$, where i is the number of reasoning steps generated so far, 1 corresponds to the question, and d is the visual feature dimension of the base model. For simplicity, batch size is ignored here.

Llama3.2-11B: We use the Llama3.2-11B-Vision model, identified as "unsloth/Llama-3.2-11B-Vision-Instruct" on Hugging Face, which is accessed through the unsloth package. To adapt this model as our *TC-NAVIGATOR*, for the **RA1**, we primarily implement `RA1VisionSdpaAttention`, which inherits from `MllamaVisionSdpaAttention` in the *mllama* module of the *transformers* library. Specifically, we add one additional argument `reasoning_states` along with the `reasoning_masks` to the *forward* function and make the modifications exactly following the `MMCLIPAttention` in QA-ViT code [†]. The main content of the *forward* function of the RA1VisionSdpaAttention is shown below:

Subsequently, `MllamaVisionSdpaAttention` in the `MllamaVisionEncoderLayer` of the Llama model is replaced with `RA1VisionSdpaAttention`. To achieve this, we inherit from `MllamaVisionEncoder` and modify it to accept the `reasoning_states` and `reasoning_states`. Finally, we inherit from `MllamaVisionModel` and update the code to incorporate the **language encoder** and handle the `reasoning_instruction`.

Therefore, in the *forward* function of `MllamaVisionModel`, the input textual `reasoning_instruction` is encoded by **language encoder** to get `reasoning_states` and `reasoning_states`. Then, the *self.transformer*, which is a `MllamaVisionEncoder`, receives these two additional inputs and processes them layer by layer.

RA2 is implemented in the *self.global_transformer* of the `MllamaVisionModel`, following similar operations described in **RA1**. The only difference is that the input textual instruction is not `reasoning_instruction` but `step_instruction`, which contains only the question and the reasoning step to be evaluated. Therefore, without additional textual encoding, we directly obtain the token embeddings of `step_instruction` from the base model, resulting in `step_states` with the shape $2\times L\times d$, where L is the padded length of the question and the reasoning step.

With these simple modifications, we can easily implement the Llama3.2-11B-Vision model as the base reward model of our *TC-NAVIGATOR*. For this QA-ViT component, only the gate projection layer is trainable.

Note that we implement *Late Fusion*, as described in QA-ViT [13], by applying **RA1** and **RA2** in the later layers. Specifically, **RA1** is applied to the last 10 layers of the `MllamaVisionEncoder`, i.e., *self.transformer*, while **RA2** is applied to the last 4 layers of the `MllamaVisionEncoder`, i.e., *self.global_transformer*.

Qwen-VL-7B: We use the Qwen-VL-7B model, specifically `Qwen/Qwen-VL-Chat` from Hugging Face. Following the exact operations discussed for **Llama3.2-11B**, we apply similar modifications for **RA1** to the `VisionTransformer` by changing the `VisualAttentionBlock` of the Qwen model. For the *Late Fusion*, we apply **RA1** to $30-40$ layers and **RA2** to $41-48$ layers.

D.2. Training

With the *RMultiplex200K* dataset, our *TC-NAVIGATOR* can be trained to be the outcome reward model (ORM) or the process reward model (PRM). For one solution sample $\mathbb{S}=\left(\mathrm{Q},\mathrm{V},\mathrm{A},\left\{\boldsymbol{z}^k_{1\ldots n},\boldsymbol{c}^k_{1\ldots n},y^k\right\}_{k=1}^{K}\right)$, ORM is trained by comparing the verification score on the whole reasoning process with the ground truth score, while PRM is optimized based on the loss calculated by comparing the ground truth score $\boldsymbol{c}_i$ with the predicted score $\widetilde{\boldsymbol{c}}_i$, where i is the step index.

ORM. Given Q, V, we have the reasoning process $\boldsymbol{z}_{1\ldots n}$, where the final step $\boldsymbol{z}_n$ contains a textual description of the solution, corresponding to the ground truth correctness score $\boldsymbol{c}_n$. With the Q, V, $\boldsymbol{z}_{1\ldots n}$ as the input, the ORM is to predict a correctness score $\boldsymbol{c}'_n$ presenting the confidence of large models on the whole reasoning process. Therefore, the ORM is trained with a cross-entropy loss:

$$\mathcal{L}_O=\boldsymbol{c}_n\log\boldsymbol{c}'_n+(1-\boldsymbol{c}_n)\log(1-\boldsymbol{c}'_n)$$

where $\boldsymbol{c}_n$ represents the correctness score of the final step. However, it is used here without error adjustments, as the

[†]https://github.com/amazon-science/QA-ViT

entire reasoning process is considered to verify the correctness of the final step.

PRM. We extract $(\mathrm{Q}, \mathrm{V}, \mathrm{A}, \boldsymbol{z}_{1\dots n}, \boldsymbol{c}_{1\dots n}, y)$ from a solution sample $\left(\mathrm{Q}, \mathrm{V}, \mathrm{A}, \left\{\boldsymbol{z}^k_{1\dots n}, \boldsymbol{c}^k_{1\dots n}, y^k\right\}_{k=1}^{K}\right)$. The PRM predicts the correctness scores for each reasoning step, outputting $\boldsymbol{c}'_{1\dots n}$. The loss is computed based on the verification of all steps, leading to:

$$\sum_{i=1}^{n} \boldsymbol{c}_i \log \boldsymbol{c}'_i + (1 - \boldsymbol{c}_i) \log (1 - \boldsymbol{c}'_i)$$

where $i \in 1, \dots, n$ is the index of the reasoning step. As discussed in the recent work [46], there is not much difference between the binary and the three classifications, and thus we can directly utilize the following equation to train the model as the binary classification. Compared to PRM800K [24], which relies on human annotations, our *ApPA* can automatically generate reliable annotations for each reasoning step. Furthermore, unlike another automatic annotation method, MATH-SHEPHERD [46], our *ApPA* achieves balanced annotation scores between positive and negative steps, thereby facilitating the robust training of PRMs. For instance, in challenging *Telecom* problems, MATH-SHEPHERD often generates incorrect reasoning steps, resulting in a training set dominated by negative samples.

Reinforce VLMs. After training *TC-NAVIGATOR* as a PRM, it can be used to supervise each reasoning step generated by VLMs during their training with reinforcement learning. Specifically, *TC-NAVIGATOR* provides rewards at the end of each reasoning step to facilitate step-by-step Proximal Policy Optimization (PPO), as introduced in prior work [46]. Therefore, the VLMs are optimized in real-time based on the reward for each reasoning step, and more importantly, this process significantly reduces the need for human effort, particularly in the challenging scientific domain of *Telecom*.

E. More Qualitative Illustrations

E.1. Examples of *RMultiplex200K*

In Fig. 14, Fig. 15, Fig. 16, and Fig. 17, we present three examples from the Wireless Communication (WC) and Satellite Communication (SC) categories of *RMultiplex200K*. Specifically, the question, the image, and the ground truth decomposed reasoning steps are illustrated. Additionally, we showcase the summarized plans of *ApPA* and two reasoning processes with step-wise correctness scores created by *ApPA*. As shown in each figure, every reasoning step is labeled with a score indicating whether the step is correct and leads to the correct solution. Scores higher than 0.5 are displayed in black, while scores lower than 0.5 are displayed in red to indicate that the reasoning step is largely incorrect, based on the annotations of *ApPA*.

First, in *Telecom* problems, the visual information provides essential context for problem-solving, making the challenges posed by *Telecom* more significant than those in Mathematics. For each problem, the VLM must comprehend both textual and visual inputs to generate the reasoning process. Moreover, solving these problems is inherently difficult, as they typically involve multiple reasoning steps, each requiring substantial computation and analysis to derive a solution. The number of reasoning steps required in the three figures is 6, 4, and 6, respectively.

Second, we verify that for step-by-step reasoning in VLMs, process supervision is crucial to ensure not only a correct solution but also a reliable reasoning process in which each step is accurate. For example, in the second solution sample presented in Fig. 14, both Step 2 and Step 4 are incorrect, yet the overall solution is correct. A similar phenomenon can be observed in the samples shown in Fig. 15 and Fig. 16. Due to the frequent occurrence of such cases—where the reasoning process contains errors but the final solution is correct—it becomes challenging to use VLMs for solving new problems and achieving high problem-solving success rates.

Third, our *ApPA* is capable of generating reliable correctness scores as labels for each reasoning step in any problem. After carefully reviewing the reasoning steps, we argue that the scores assigned by *ApPA* are reliable and accurately reflect the quality of the corresponding step. For instance, when a reasoning step is completely incorrect, the assigned score is typically around 0.1 or 0.2. In contrast, a reasoning step verified as correct by humans is assigned a higher score by *ApPA*.

More importantly, **based on the plan-based annotation process, *ApPA* achieves a balance between positive and negative samples**, where the number of lower correctness scores is approximately equal to the number of higher ones. This balance ensures that, when using our *RMultiplex200K* dataset for training reward models, the optimization process avoids the challenges of imbalanced learning, leading to improved performance.

E.2. Examples of *TC-NAVIGATOR*

After *TC-NAVIGATOR* is fine-tuned as a reward model using *RMultiplex200K*, it can be directly used as a verifier to evaluate the reasoning steps generated by any VLMs. The detailed operation is illustrated in Fig. 13. First, as shown in the upper subfigure, the VLM is prompted with the question Q, the corresponding image V, and the instruction R to generate a reasoning process A', which is then decomposed into individual reasoning steps, such as “Step 1:”, “Step 2:”, and so on. Subsequently, for any reasoning step $\boldsymbol{z}_i$, our *TC-NAVIGATOR* receives all existing reasoning steps $\boldsymbol{z}_{1\dots i}$ and is prompted to generate the correctness score $\boldsymbol{c}'_i$ for it. By

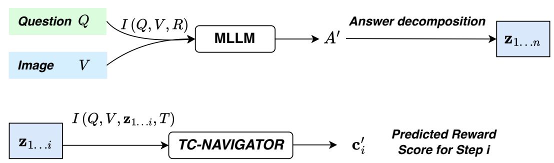


Figure 13. Illustration of how *TC-NAVIGATOR* works as a verifier to evaluate each reasoning step of an VLM during problem-solving. The I represents a prompt, which must include the question Q and the corresponding image V. Additionally, R serves as an instruction to guide the VLM in solving the problem through step-by-step reasoning, while T guides *TC-NAVIGATOR* in generating the correctness score for the reasoning step $\boldsymbol{z}_i$.

iteratively repeating this score prediction process, we obtain correctness scores $\boldsymbol{c}'_{1...n}$ for all n reasoning steps.

We present examples demonstrating how the *TC-NAVIGATOR*, trained as a PRM using the base model Qwen-2-VL-72B-instruct, verifies each reasoning step in the reasoning process generated by Llama-3.2-90B. Specifically, we present the predicted verification scores along with the corresponding attention maps, highlighting which parts of the image are important for the verification process.

Visible attention map. To illustrate the attention map, instead of using tools, such as Transformer-Explainability [‡], we mainly utilize the density map, which is a standard way to draw the attention on the image. First, we extract attention weights from layers that involves **RA2** which locates in the latter layers and build directly the relation between the image and the reasoning step to be verified, such as $\boldsymbol{z}_i$. After stacking these weights, we compute their average. Next, we average the attention tensor across heads and extract only the image tokens to obtain a tensor with the shape (batch_size, seq_len, img_tokens), where the batch_size and seq_len (only consider the generated textual score) are 1 in this case. Subsequently, we select the 30 image tokens with the largest scores, each corresponding to patches of the image. This allows us to create a blank map of the same size as the image, where each image patch corresponds to one point on the map, resulting in a total of 30 points. Finally, we apply a Gaussian filter with a sigma (standard deviation controlling the spread) of 30 to spread the intensity and create a density map. After normalizing the generated density map, we overlay it on the image to illustrate the **attention shifting** of our *TC-NAVIGATOR* when verifying different reasoning steps.

Therefore, in Fig. 18, Fig. 19, Fig. 20, and Fig. 21, we present four examples illustrating how the *TC-NAVIGATOR* functions as a verifier to evaluate the reasoning steps generated by Llama-3.2-90B. Specifically, we showcase two base models: Qwen-2-VL-72B (QW) and Llama-3.2-11B (Ll). For instance, when Qwen-2-VL-72B appears in an example, it indicates that *TC-NAVIGATOR* with Qwen-2-VL-72B has been fine-tuned as the PRM.

We make three observations. First, using larger VLMs as the base model for the verifier results in more precise correctness scores. Second, consistent with our argument for the *TC-NAVIGATOR*, VLM verifiers adjust their attention when evaluating different reasoning steps. This observation may inspire further exploration of *whether we can implement the Chain of Thought in the visual domain* to enable VLMs to follow a specific logic, adjusting attention across steps for more reliable reasoning and verification. Third, we confirm that the conclusion drawn in previous work [24] — that PRM can ensure reliable problem solving —– also holds true in the multimodal domain.

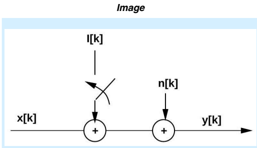
[‡] https://github.com/hila-chefer/Transformer-Explainability

Question

Time-Varying Interference: This problem illustrates the capacity gains that can be obtained from interference estimation, and how a malicious jammer can wreak havoc on link performance. Consider the following interference channel, shown in the image.
The channel has a combination of AWGN $n[k]$ and interference $I[k]$. We model $I[k]$ as AWGN. The interferer is on (i.e. the switch is down) with probability .25 and off (i.e. the switch is up) with probability .75 . The average transmit power is 10 mW , the noise spectral density is 10^{-8} W/Hz, the channel bandwidth B is 10 KHz (receiver noise power is N_oB), and the interference power (when on) is 9 mW .

Suppose now that the interferer is a malicious jammer with perfect knowledge of $x[k]$ (so the interferer is no longer modeled as AWGN). Assume that neither transmitter nor receiver have knowledge of the jammer behavior. Assume also that the jammer is always on and has an average transmit power of 10 mW . What strategy should the jammer use to minimize the SNR of the received signal?

Image



Ground Truth: Decomposed Reasoning Steps

Step 1: *The jammer should transmit* $-x(t)$ *to completely cancel off the signal.* $\bar{S} = 10\,\mathrm{mW}$, $N_0 = .001\mu\,\mathrm{W/Hz}$, $B = 10\mathrm{MHz}$. **1.0**

Step 2: *Now we compute the SNR's as:* $\gamma_j = \frac{|H_j|^2\bar{S}}{N_0B}$. *This gives:* **1.0**
$\gamma_1 = \frac{|1|^2 10^{-3}}{0.001\times10^{-6}10\times10^6} = 1, \gamma_2 = .25, \gamma_3 = 4, \gamma_4 = 0.0625.$

Step 3: *To compute* γ_0, *we use the power constraint as:* **1.0**
$\sum_j \left(\frac{1}{\gamma_0} - \frac{1}{\gamma_j}\right)_+ = 1.$
First, assume that $\gamma_0 < 0.0625$, *then we have:*
$\frac{4}{\gamma_0} = 1 + \left(\frac{1}{1} + \frac{1}{0.25} + \frac{1}{4} + \frac{1}{0.0625}\right)$, *which implies*
$\gamma_0 = 0.1798 > 0.0625.$
So, our assumption was wrong.

Step 4: *Now we assume that* $0.0625 < \gamma_0 < .25$, *then:* **1.0**
$\frac{3}{\gamma_0} = 1 + \left(\frac{1}{1} + \frac{1}{.25} + \frac{1}{4}\right)$
$\Rightarrow \gamma_0 = .48 > 0.25$
So, our assumption was wrong again.

Step 5: *Next we assume that* $0.25 < \gamma_0 < 1$, *then:* **1.0**
$\frac{2}{\gamma_0} = 1 + \left(\frac{1}{1} + \frac{1}{4}\right)$
$\Rightarrow \gamma_0 = .8889 < 1$
This time our assumption was right. So we get that only two sub-bands, each of bandwidth 10 MHz, are used for transmission, and the remaining two with lesser SNRs are left unused.

Step 6: *Now, we can find capacity as:* **1.0**
$C = \sum_{j:\gamma_j\geq\gamma_0} B\log_2\left(\frac{\gamma_j}{\gamma_0}\right)$

This gives us, $\mathrm{C} = 23.4\,\mathrm{Mbps}.$

Labeled Reasoning Steps Generated by ApPA

Step 1: The jammer should generate a signal $I[k] = -x[k]$ to achieve perfect destructive interference, fully canceling the transmitted signal and minimizing the SNR at the receiver. **0.9**

Step 2: *Calculate the noise power:* **0.84**
$\text{Noise Power} = N_0 \times B = 10^{-8}\,\mathrm{W/Hz} \times 10^4\,\mathrm{Hz} = 10^{-4}\,\mathrm{W}$. *Since the jammer transmits* $-x(t)$, *the effective signal power becomes zero, yielding* $\mathrm{SNR} = \frac{0}{10^{-4}} = 0$. *Thus, the jammer's strategy results in an SNR of zero.*

Step 3: *Assume the jammer transmits* $-x(t)$ *to perfectly cancel the signal* $x(t)$. *The resulting received signal power becomes zero, as the jammer's signal perfectly negates the transmitted signal. The SNR is then determined solely by the noise power, which is* $N_0B = 10^{-8}\,\mathrm{W/Hz} \times 10^4\,\mathrm{Hz} = 10^{-4}\,\mathrm{W}$. *Therefore, the SNR is* $\frac{0}{10^{-4}} = 0$. *This confirms that the jammer's strategy of transmitting* $-x(t)$ *minimizes the SNR to zero, effectively achieving the goal.* **0.66**

Step 4: *Since the assumption that* $\gamma_0 < 0.0625$ *was incorrect, we now assume* $\gamma_0 \geq 0.0625$. *Re-evaluate the power constraint equation:* **0.22**

$\sum_j \left(\frac{1}{\gamma_0} - \frac{1}{\gamma_j}\right)_+ = 1$

Given $\gamma_1 = 1$, $\gamma_2 = 0.25$, $\gamma_3 = 4$, *and* $\gamma_4 = 0.0625$, *we only consider terms where* $\gamma_0 < \gamma_j$: $\frac{1}{\gamma_0} - \frac{1}{0.0625} = 1$
Solving for γ_0: $\frac{1}{\gamma_0} = 1 + 16$, $\gamma_0 = \frac{1}{17} \approx 0.0588$

Since $\gamma_0 \approx 0.0588 < 0.0625$, *this assumption is consistent. Therefore,* $\gamma_0 \approx 0.0588$ *is the correct value.*

Step 5: *Now assume that* $0.25 < \gamma_0 < 1$. *Calculate* γ_0 *using the equation:* **0.2**
$\frac{2}{\gamma_0} = 1 + \left(\frac{1}{1} + \frac{1}{0.25}\right)$

Solving this gives:
$\gamma_0 = \frac{2}{1+1+4} = \frac{2}{6} = 0.3333$

Since $0.25 < 0.3333 < 1$, *this assumption is correct. Thus,* $\gamma_0 = 0.3333$.

Step 6: *Calculate the capacity for sub-bands with SNR* $\gamma_j \geq \gamma_0 = 0.8889$ *using* $C = B\log_2(1 + \mathrm{SNR})$. **0.1**

For $\gamma_1 = 1$: $C_1 = 10 \times 10^6 \log_2(2) = 10\,\mathrm{Mbps}$.
For $\gamma_3 = 4$: $C_3 = 10 \times 10^6 \log_2(5) \approx 23.22\,\mathrm{Mbps}$.

Total capacity: $C_{\text{total}} = 10 + 23.22 = 33.22\,\mathrm{Mbps}$.

Step 1. To minimize the SNR of the received signal, the jammer should employ a strategy of generating an interference signal that is the exact negative of the transmitted signal $x[k]$. Given that the jammer has perfect knowledge of $x[k]$, it can produce an interference signal $I[k] = -x[k]$. This results in the received signal being $y[k] = x[k] + I[k] + n[k] = x[k] - x[k] + n[k] = n[k]$. Thus, the received signal consists solely of the noise $n[k]$, effectively canceling out the transmitted signal and minimizing the SNR to the level of the noise power alone. **0.94**

Step 2: To calculate the SNR, we first determine the effective noise power. Since the jammer transmits $-x(t)$, the received signal is effectively canceled, and the only remaining noise is the receiver noise. The receiver noise power is given by $N_0B = 10^{-8}\,\mathrm{W/Hz} \times 10^4\,\mathrm{Hz} = 10^{-4}\,\mathrm{W}$. The SNR is then calculated as: **0.31**

$\mathrm{SNR} = \frac{\bar{S}}{N_0B} = \frac{10\times10^{-3}\,\mathrm{W}}{10^{-4}\,\mathrm{W}} = 100$

Thus, the jammer's strategy of transmitting $-x(t)$ results in an SNR of 100, effectively minimizing the received signal's SNR by canceling the signal component.

Step 3: Assume the jammer transmits $-x(t)$ to perfectly cancel the signal. Calculate the resulting SNR at the receiver. The received signal power is effectively zero due to perfect cancellation, while the noise power remains N_0B. Thus, the SNR is: **0.8**

$\mathrm{SNR} = \frac{0}{N_0B} = 0$

This confirms that the jammer's strategy of transmitting $-x(t)$ minimizes the SNR to zero, satisfying the constraint of minimizing the received signal's SNR.

Step 4: **0.27**
Since the assumption $\gamma_0 < 0.0625$ was incorrect, we now assume $\gamma_0 \geq 0.0625$. Re-evaluate the equation: $\sum_j \left(\frac{1}{\gamma_0} - \frac{1}{\gamma_j}\right)_+ = 1$

Given $\gamma_1 = 1$, $\gamma_2 = 0.25$, $\gamma_3 = 4$, and $\gamma_4 = 0.0625$, the terms where $\gamma_0 \geq \gamma_j$ contribute zero. Thus, $\frac{1}{\gamma_0} - \frac{1}{0.0625} = 1$

Solving for γ_0 gives: $\frac{1}{\gamma_0} = 1 + 16 = 17 \Rightarrow \gamma_0 = \frac{1}{17} \approx 0.0588$

Since $\gamma_0 \approx 0.0588 < 0.0625$, this assumption is consistent. Therefore, the correct range for γ_0 is $\gamma_0 \approx 0.0588$.

Step 5: **0.37**
Now assume that $0.25 < \gamma_0 < 1$, then: $\frac{2}{\gamma_0} = 1 + \left(\frac{1}{1} + \frac{1}{0.25}\right)$

$\Rightarrow \gamma_0 = 0.6667 < 1$ This assumption is correct, so $\gamma_0 = 0.6667$.

Step 6: **0.78**
To find the capacity of the communication channel, we use the sub-bands with SNRs greater than or equal to the threshold SNR $\gamma_0 = 0.8889$. The capacity C is calculated as: $C = \sum_{j\in\{1,3\}} B\log_2\left(\frac{\gamma_j}{\gamma_0}\right)$ where j indexes the sub-bands with $\gamma_j > \gamma_0$.
Thus,
$C = 10 \times 10^6 \log_2\left(\frac{1}{0.8889}\right) + 10 \times 10^6 \log_2\left(\frac{4}{0.8889}\right)$

Compute these terms to find the total channel capacity.

Figure 14. Illustration of the complete reasoning processes from Fig. 1. We present the decomposed ground truth reasoning steps along with two reasoning processes, each annotated with correctness scores generated by our *ApPA*.

Question

Consider a system with a transmitter, receiver, and scatterer as shown in the Figure. Assume the transmitter and receiver are both at heights $h_t = h_r = 4\,\mathrm{m}$ and are separated by distance d, with the scatterer at distance $0.5d$ along both dimensions in a two-dimensional grid of the ground, i.e., on such a grid the transmitter is located at $(0,0)$, the receiver is located at $(0,d)$, and the scatterer is located at $(0.5d, 0.5d)$. Assume a radar cross section of $20\,\mathrm{dB\,m}^2$. Find the path loss of the scattered signal for $d = 1, 10, 100$, and $1000\,\mathrm{m}$. Compare with the path loss at these distances if the signal is just reflected with reflection coefficient $R = -1$.

Image

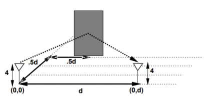


Figure 2.11: System with Scattering

Ground Truth: Decomposed Reasoning Steps

Step 1: **1.0**
$f_c = 900\,\mathrm{MHz}$
$\lambda = 1/3\,\mathrm{m}$
$G = 1$, *radar cross section* $20\,\mathrm{dBm}^2 = 10\log_{10}\sigma \Rightarrow \sigma = 100$
$d = 1$, $s = s' = \sqrt{(0.5d)^2 + (0.5d)^2} = d\sqrt{0.5} = \sqrt{0.5}$

Step 2: **1.0**
Path loss due to scattering

$$\frac{P_r}{P_t} = \left[\frac{\lambda\sqrt{G\sigma}}{(4\pi)^{3/2}ss'}\right]^2 = 0.0224 = -16.498\,\mathrm{dB}$$

Step 3: **1.0**
Path loss due to reflection (using 2-ray model):

$$\frac{P_r}{P_t} = \left(\frac{R\sqrt{G}}{s+s'}\right)^2\left(\frac{\lambda}{4\pi}\right)^2 = 3.52\times10^{-4} = -34.54\,\mathrm{dB}$$

For different values of d*:*
- $d = 10$: $P_{\text{scattering}} = -56.5\,\mathrm{dB}$, $P_{\text{reflection}} = -54.54\,\mathrm{dB}$
- $d = 100$: $P_{\text{scattering}} = -96.5\,\mathrm{dB}$, $P_{\text{reflection}} = -74.54\,\mathrm{dB}$
- $d = 1000$: $P_{\text{scattering}} = -136.5\,\mathrm{dB}$, $P_{\text{reflection}} = -94.54\,\mathrm{dB}$

Step 4: **1.0**
Notice that scattered rays over long distances result in tremendous path loss.

Summarized Plans

Plan 1:
Identify and define the relevant parameters and constants needed for the problem, such as frequency, wavelength, gain, and radar cross section. Calculate any necessary geometric distances or relationships based on the given configuration.

Plan 2:
To determine the path loss due to scattering, apply the radar equation for non-coherent scattering. Calculate the received power ratio by incorporating the wavelength, antenna gain, radar cross-section, and the distances from the transmitter to the scatterer and from the scatterer to the receiver. Use these parameters to compute the path loss in decibels.

Plan 3:
To determine the path loss due to reflection using the two-ray model, calculate the received power ratio by considering the reflection coefficient, antenna gain, and the distance between the transmitter and receiver. Use the formula that incorporates these factors along with the wavelength and the geometry of the system. Compare the calculated path loss with other scenarios to analyze the impact of reflection on signal propagation.

Plan 4:
Identify and analyze the trend or pattern in the results obtained from calculations or observations, particularly focusing on how a specific variable or condition affects the outcome over a range of scenarios.

Labeled Reasoning Steps Generated by ApPA

Step 1: To begin solving the problem, we need to calculate the geometric distances involved in the path of the scattered signal. Given the positions of the transmitter, receiver, and scatterer, we can determine the distances using the Pythagorean theorem. **0.73**

1. **Distance from Transmitter to Scatterer:**
The transmitter is at $(0,0)$ and the scatterer is at $(0.5d, 0.5d)$. The distance d_{ts} from the transmitter to the scatterer is:
$d_{ts} = \sqrt{(0.5d-0)^2 + (0.5d-0)^2} = \sqrt{0.5^2d^2 + 0.5^2d^2} = \sqrt{0.5d^2} = 0.5\sqrt{2}d$

2. **Distance from Scatterer to Receiver:**
The scatterer is at $(0.5d, 0.5d)$ and the receiver is at $(0,d)$. The distance d_{sr} from the scatterer to the receiver is:
$d_{sr} = \sqrt{(0.5d-0)^2 + (0.5d-d)^2} = \sqrt{0.5^2d^2 + (-0.5d)^2} = \sqrt{0.5d^2} = 0.5\sqrt{2}d$

These distances will be used in subsequent steps to calculate the path loss of the scattered signal.

Step 2: **0.87**
To calculate the path loss of the scattered signal, we use the bistatic radar equation for non-coherent scattering:
$P_r = \frac{P_tG_tG_r\lambda^2\sigma}{(4\pi)^3R_t^2R_r^2}$
where:
- P_t is the transmitted power,
- G_t and G_r are the transmitter and receiver antenna gains (both equal to 1 in this case),
- $\lambda = \frac{1}{3}\,\mathrm{m}$ is the wavelength,
- $\sigma = 100\,\mathrm{m}^2$ is the radar cross-section,
- R_t and R_r are the distances from the transmitter to the scatterer and from the scatterer to the receiver, respectively.

For the given setup, both R_t and R_r are equal to $\sqrt{0.5}d$.
Substituting these into the equation, the received power P_r becomes:
$P_r = \frac{P_t\cdot1\cdot1\cdot\left(\frac{1}{3}\right)^2\cdot100}{(4\pi)^3\cdot(\sqrt{0.5}d)^2\cdot(\sqrt{0.5}d)^2}$
Simplifying, we find:
$P_r = \frac{P_t\cdot\frac{1}{9}\cdot100}{(4\pi)^3\cdot0.5^2\cdot d^4}$
$P_r = \frac{P_t\cdot\frac{100}{9}}{(4\pi)^3\cdot0.25\cdot d^4}$
$P_r = \frac{P_t\cdot\frac{400}{9}}{(4\pi)^3\cdot d^4}$
The path loss L in decibels is given by:
$L = 10\log_{10}\left(\frac{P_t}{P_r}\right)$
$L = 10\log_{10}\left(\frac{(4\pi)^3\cdot d^4}{\frac{400}{9}}\right)$
$L = 10\log_{10}\left(\frac{(4\pi)^3\cdot d^4\cdot9}{400}\right)$
This expression can be used to compute the path loss for each given distance $d = 1, 10, 100$, and $1000\,\mathrm{m}$.

Step 3:
Path loss due to reflection using the two-ray model

For the reflection scenario with reflection coefficient $R = -1$, the path loss can be calculated using the formula for the two-ray ground reflection model:

$\frac{P_r}{P_t} = \left|\frac{\lambda}{4\pi d}\right|^2|1+R|^2$

Given $R = -1$, the term $|1+R|^2 = 0$, which implies that the direct and reflected paths cancel each other out perfectly, leading to a theoretical path loss of infinity (complete cancellation). However, in practical scenarios, this perfect cancellation is rarely achieved due to factors like imperfect reflection surfaces and antenna characteristics. Thus, the path loss due to reflection is significantly higher than the scattering scenario, especially at distances where the two-ray model is applicable. **0.85**

Step 4:
Analyze the trend in path loss for both scattering and reflection as the distance d increases.

For scattering:
- At $d = 1\,\mathrm{meter}$, path loss is $-16.498\,\mathrm{dB}$.
- At $d = 10\,\mathrm{meters}$, path loss is $-56.5\,\mathrm{dB}$.
- At $d = 100\,\mathrm{meters}$, path loss is $-96.5\,\mathrm{dB}$.
- At $d = 1000\,\mathrm{meters}$, path loss is $-136.5\,\mathrm{dB}$.

For reflection:
- At $d = 1\,\mathrm{meter}$, path loss is $-34.54\,\mathrm{dB}$.
- At $d = 10\,\mathrm{meters}$, path loss is $-54.54\,\mathrm{dB}$.
- At $d = 100\,\mathrm{meters}$, path loss is $-74.54\,\mathrm{dB}$.
- At $d = 1000\,\mathrm{meters}$, path loss is $-94.54\,\mathrm{dB}$. **0.93**

Conclusion: As d increases, the path loss for both scattering and reflection increases. However, the scattering path loss increases more rapidly compared to the reflection path loss, indicating that scattering becomes less efficient at larger distances compared to reflection.

Step 1: To solve the problem, we first need to calculate the path loss of the scattered signal using the radar cross section and the geometry of the system. The radar cross section (RCS) is given as 20 dB m^2, which is equivalent to a linear scale value of 100 m^2 (since $20\,\mathrm{dB} = 10\log_{10}(\mathrm{RCS})$, thus $\mathrm{RCS} = 10^{20/10} = 100\,\mathrm{m}^2$). **0.52**

The path loss for a scattered signal can be calculated using the bistatic radar equation, which involves the distances from the transmitter to the scatterer and from the scatterer to the receiver. The distances are:

1. Transmitter to scatterer:
$\sqrt{(0.5d)^2 + (0.5d)^2 + (h_t)^2} = \sqrt{0.5d^2 + 0.5d^2 + 16} = \sqrt{d^2/2 + 16}$.

2. Scatterer to receiver:
$\sqrt{(0.5d)^2 + (0.5d-d)^2 + (h_r)^2} = \sqrt{0.5d^2 + (-0.5d)^2 + 16} = \sqrt{d^2/2 + 16}$.

For each value of d (1, 10, 100, 1000 meters), calculate these distances and use them to determine the path loss using the formula for bistatic radar path loss:

$\text{Path Loss} = \left(\frac{4\pi}{\lambda}\right)^2\cdot\frac{(d_{ts}\cdot d_{sr})^2}{\sigma}$

where λ is the wavelength, d_{ts} is the distance from the transmitter to the scatterer, d_{sr} is the distance from the scatterer to the receiver, and σ is the radar cross section.

This step sets up the necessary calculations for determining the path loss for each specified distance d.

Step 2: **0.3**
To calculate the path loss due to scattering, use the radar equation for scattered signals:
$P_r = \frac{P_tG_tG_r\lambda^2\sigma}{(4\pi)^3(d_1^2)(d_2^2)}$
where P_t is the transmitted power, G_t and G_r are the transmitter and receiver gains (both equal to 1), λ is the wavelength, σ is the radar cross-section, and d_1 and d_2 are the distances from the transmitter to the scatterer and from the scatterer to the receiver, respectively.
Given $\lambda = \frac{1}{3}\,\mathrm{m}$, $\sigma = 100$, and $d_1 = d_2 = \sqrt{0.5}d$, the received power P_r can be expressed as:
$P_r = \frac{P_t\cdot1\cdot1\cdot\left(\frac{1}{3}\right)^2\cdot100}{(4\pi)^3\cdot(\sqrt{0.5}d)^2\cdot(\sqrt{0.5}d)^2}$

Simplifying, the path loss L in decibels is:
$L = 10\log_{10}\left(\frac{(4\pi)^3\cdot0.5^2\cdot d^4}{\left(\frac{1}{3}\right)^2\cdot100}\right)$
Calculate L for $d = 1, 10, 100$, and $1000\,\mathrm{m}$.

Step 3: **0.14**
Path loss due to reflection using the two-ray model:
The power ratio for the reflected signal is given by:
$\frac{P_r}{P_t} = |R|^2\left(\frac{\lambda}{4\pi d}\right)^2$
Given $R = -1$, the reflection coefficient magnitude is $|R| = 1$.
Substitute the values:
For $d = 1\,\mathrm{m}$:
$\frac{P_r}{P_t} = \left(\frac{1/3}{4\pi\times1}\right)^2 = \left(\frac{1}{12\pi}\right)^2 \approx 0.0007$
Path loss in dB:
$\text{Path loss} = -10\log_{10}(0.0007) \approx 31.55\,\mathrm{dB}$
Repeat for $d = 10, 100$, and $1000\,\mathrm{m}$:
For $d = 10\,\mathrm{m}$:
$\frac{P_r}{P_t} = \left(\frac{1/3}{4\pi\times10}\right)^2 = \left(\frac{1}{120\pi}\right)^2 \approx 0.000007$
Path loss in dB:
$\text{Path loss} = -10\log_{10}(0.000007) \approx 51.55\,\mathrm{dB}$
For $d = 100\,\mathrm{m}$:
$\frac{P_r}{P_t} = \left(\frac{1/3}{4\pi\times100}\right)^2 = \left(\frac{1}{1200\pi}\right)^2 \approx 0.00000007$
Path loss in dB:
$\text{Path loss} = -10\log_{10}(0.00000007) \approx 71.55\,\mathrm{dB}$
For $d = 1000\,\mathrm{m}$:
$\frac{P_r}{P_t} = \left(\frac{1/3}{4\pi\times1000}\right)^2 = \left(\frac{1}{12000\pi}\right)^2 \approx 0.0000000007$
Path loss in dB:
$\text{Path loss} = -10\log_{10}(0.0000000007) \approx 91.55\,\mathrm{dB}$
This completes the calculation of path loss due to reflection for the given distances.

Step 4:
Step 4: \nAnalyzing the results, we observe that as the distance d increases, the path loss for both scattering and reflection increases. Specifically, the path loss due to scattering increases by approximately 40 dB for each tenfold increase in distance, while the path loss due to reflection increases by approximately 20 dB for each tenfold increase in distance. This indicates that scattering results in a more significant increase in path loss with distance compared to reflection. Thus, at larger distances, the scattered signal experiences greater attenuation than the reflected signal. **0.78**

Figure 15. Illustration of a solution sample of a problem from the Wireless Communication (WC) category. Apart from the decomposed ground truth reasoning steps along with two reasoning processes, each annotated with correctness scores generated by our *ApPA*, we present the summarized plans of the *ApPA*.

Question

The angle formed by the slant ranges of two geostationary satellites from a certain Earth station as shown in Figure 7.20 is $5°$. Determine the longitudinal location of the two satellites given that the two slant ranges of satellites A and B are $42100\,\text{km}$ and $42000\,\text{km}$, respectively. Given that the radius of the geostationary orbit is equal to $42164\,\text{km}$.

Image

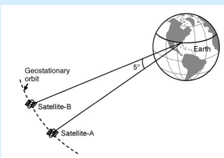


Figure 7.20 Figure for Problem 7.12

Ground Truth: Decomposed Reasoning Steps

***Step 1**: If β is the longitudinal separation of the two satellites, it can be computed from* **1.0**

$$\cos\theta = \left[\frac{d_A^2+d_B^2-2r^2(1-\cos\beta)}{2d_Ad_B}\right]$$

where:
$d_A =$ *slant range of satellite* $A = 42100\,\text{km}$,
$d_B =$ *slant range of satellite* $B = 42000\,\text{km}$,
$r =$ *geostationary orbit radius* $= 42164\,\text{km}$,
$\theta =$ *angular separation of the two satellites as viewed from the Earth station* $= 5°$.

***Step 2**: Therefore,* **1.0**

$$\cos 5° = \frac{(42100\times10^3)^2+(42000\times10^3)^2-2\times(42164\times10^3)^2\times(1-\cos\beta)}{2\times(42100\times10^3)\times(42000\times10^3)}$$

***Step 3**: or* **1.0**

$$0.996 = \frac{(177241\times10^{10})+(176400\times10^{10})-2\times(177780.3\times10^{10})\times(1-\cos\beta)}{2\times(176820\times10^{10})}$$

***Step 4**: or* **1.0**
$\cos\beta = 1 - 0.004 = 0.996$

***Step 5**: or* **1.0**
$\beta = 5.126°$

***Step 6**: Therefore, the longitudinal separation between the two satellites* $= 5.126°$. **1.0**

Summarized Plans

***Step 1**: To determine the angular separation between two points in space, use the cosine rule for spherical triangles. This involves calculating the cosine of the angle using the known distances between the points and the radius of the sphere.*

***Step 2**: Substitute the known values into the given equation to solve for the unknown variable.*

***Step 3**: Substitute known values into the equation to simplify it, and isolate the variable of interest by performing algebraic manipulations.*

***Step 4**: To find the value of a trigonometric function for an angle, rearrange the equation to isolate the desired trigonometric expression, substitute known values, and solve for the unknown.*

***Step 5**: To determine an angle from its cosine value, use the inverse cosine function (arccos) on the calculated cosine value to find the angle in degrees or radians.*

***Step 6**: Determine the final result by interpreting the calculated value from a previous step as the desired measurement or solution, ensuring it aligns with the context of the problem..*

Labeled Reasoning Steps Generated by ApPA

Step 1: To determine the angular separation between the two satellites, we can apply the cosine rule for spherical triangles. The cosine rule for spherical triangles states: **0.81**
$\cos(C) = \frac{\cos(a)-\cos(b)\cos(c)}{\sin(b)\sin(c)}$
However, in this context, we are dealing with a simpler form of the cosine rule for the angle between two vectors in space, which is:
$\cos(\theta) = \frac{r_1^2+r_2^2-d^2}{2r_1r_2}$
where r_1 and r_2 are the slant ranges of the satellites from the Earth station, and d is the distance between the satellites. Given that the angle θ between the slant ranges is $5°$, we can rearrange the formula to solve for d:
$d^2 = r_1^2 + r_2^2 - 2r_1r_2\cos(\theta)$
Substitute the given values $r_1 = 42100\,\text{km}$, $r_2 = 42000\,\text{km}$, and $\theta = 5°$:
$d^2 = 42100^2 + 42000^2 - 2\times42100\times42000\times\cos(5°)$
Calculate $\cos(5°) \approx 0.9962$:
$d^2 = 42100^2 + 42000^2 - 2\times42100\times42000\times0.9962$
Compute d to find the distance between the satellites.

***Step 2**: Substitute the known values into the equation:* **0.95**

$$\cos 5° = \left[\frac{42100^2+42000^2-2\times42164^2(1-\cos\beta)}{2\times42100\times42000}\right]$$

Calculate $\cos 5°$ *and simplify the equation to solve for* $\cos\beta$.

Step 3: Substitute the known values into the equation and simplify: **0.26**

$\cos 5° = \frac{(42100\times10^3)^2+(42000\times10^3)^2-2\times(42164\times10^3)^2\times(1-\cos\beta)}{2\times(42100\times10^3)\times(42000\times10^3)}$
Calculate $\cos 5° \approx 0.9962$.
Substitute and simplify:
$0.9962 = \frac{(42100^2+42000^2)\times10^6-2\times42164^2\times10^6\times(1-\cos\beta)}{2\times42100\times42000\times10^6}$
Simplify further:
$0.9962 = \frac{(1772410000+1764000000)-2\times1777760896\times(1-\cos\beta)}{3536400000}$
$0.9962 = \frac{3536410000-2\times1777760896+2\times1777760896\cos\beta}{3536400000}$

Multiply through by 3536400000 and solve for $\cos\beta$:
$0.9962\times3536400000 = 3536410000 - 2\times1777760896 + 2\times1777760896\cos\beta$
$3522726480 = 3536410000 - 3555521792 + 3555521792\cos\beta$
$3522726480 = -1911792 + 3555521792\cos\beta$
$3524638272 = 3555521792\cos\beta$
$\cos\beta = \frac{3524638272}{3555521792}$

Calculate $\cos\beta$:
$\cos\beta \approx 0.9913$
Finally, find β using the inverse cosine function:
$\beta \approx \cos^{-1}(0.9913) \approx 7.9°$

Thus, the longitudinal separation of the two satellites is approximately $7.9°$.

Step 4: Rearrange the equation to solve for $\cos\beta$: **0.78**
$0.996 = \frac{177241\times10^{10}+176400\times10^{10}-2\times177780.3\times10^{10}\times(1-\cos\beta)}{2\times176820\times10^{10}}$

Multiply both sides by $2\times176820\times10^{10}$:
$0.996\times2\times176820\times10^{10} = 177241\times10^{10} + 176400\times10^{10} - 2\times177780.3\times10^{10}\times(1-\cos\beta)$

Calculate the left side:
$353640\times10^{10}\times0.996 = 352226.64\times10^{10}$

Substitute and simplify:
$352226.64\times10^{10} = 353641\times10^{10} - 2\times177780.3\times10^{10}\times(1-\cos\beta)$

Rearrange to solve for $\cos\beta$:
$2\times177780.3\times10^{10}\times(1-\cos\beta) = 353641\times10^{10} - 352226.64\times10^{10}$
$2\times177780.3\times10^{10}\times(1-\cos\beta) = 1414.36\times10^{10}$

Divide by $2\times177780.3\times10^{10}$:
$1-\cos\beta = \frac{1414.36\times10^{10}}{2\times177780.3\times10^{10}}$

Calculate:
$1-\cos\beta = \frac{1414.36}{355560.6}$
$1-\cos\beta \approx 0.003978$
$\cos\beta \approx 1 - 0.003978 = 0.996022$

Thus, $\beta \approx \cos^{-1}(0.996022)$.

Step 5: To find the longitudinal separation β, use the inverse cosine function on the calculated value of $\cos\beta$: **0.07**
$\beta = \arccos(0.996)$

Calculating this gives:
$\beta \approx 5.74°$

Thus, the longitudinal separation between the two satellites is approximately $5.74°$.

Step 6: The longitudinal separation of the two satellites is determined to be approximately $5.126°$, which aligns with the calculated value of β. This is the final result for the longitudinal location difference between the two geostationary satellites. **0.97**

Step 1: To determine the longitudinal location of the two satellites, we use the cosine rule for spherical triangles. The cosine rule for spherical triangles states: **0.73**
$\cos C = \frac{\cos a-\cos b\cdot\cos c}{\sin b\cdot\sin c}$

In this context, C is the angle between the two slant ranges, which is $5°$, and a, b, and c are the sides of the spherical triangle, which correspond to the central angles subtended by the slant ranges and the radius of the geostationary orbit. The sides b and c are the angles subtended by the slant ranges of satellites A and B at the center of the Earth, calculated as:
$b = \cos^{-1}\left(\frac{42164^2+42100^2-42000^2}{2\times42164\times42100}\right)$
$c = \cos^{-1}\left(\frac{42164^2+42000^2-42100^2}{2\times42164\times42000}\right)$

Substitute these into the cosine rule to find the longitudinal separation between the satellites.

Step 2: Substitute the known values into the equation to solve for β: **0.79**
$\cos 5° = \left[\frac{42100^2+42000^2-2\times42164^2(1-\cos\beta)}{2\times42100\times42000}\right]$
Calculate each term:
1. $42100^2 = 1772410000$
2. $42000^2 = 1764000000$
3. $2\times42164^2 = 2\times1777746896 = 3555493792$

Substitute these into the equation:
$\cos 5° = \left[\frac{1772410000+1764000000-3555493792(1-\cos\beta)}{3536400000}\right]$
Simplify and solve for $\cos\beta$:
$\cos 5° = \frac{3536410000-3555493792+3555493792\cos\beta}{3536400000}$
Rearrange to isolate $\cos\beta$:
$3536400000\cos 5° = 3536410000 - 3555493792 + 3555493792\cos\beta$
$3555493792\cos\beta = 3536400000\cos 5° - 3536410000 + 3555493792$
Solve for $\cos\beta$:
$\cos\beta = \frac{3536400000\cos 5°-3536410000+3555493792}{3555493792}$

Calculate $\cos 5°$ and substitute to find β.

Step 3: Substitute the known values into the equation and simplify to solve for $\cos\beta$: **0.6**
$\cos 5° = \frac{(42100\times10^3)^2+(42000\times10^3)^2-2\times(42164\times10^3)^2\times(1-\cos\beta)}{2\times(42100\times10^3)\times(42000\times10^3)}$

Calculate each term:
1. $(42100\times10^3)^2 = 1.77241\times10^{15}$
2. $(42000\times10^3)^2 = 1.764\times10^{15}$
3. $2\times(42164\times10^3)^2 = 3.554\times10^{15}$
4. $2\times(42100\times10^3)\times(42000\times10^3) = 3.5364\times10^{15}$

Substitute these into the equation:
$\cos 5° = \frac{1.77241\times10^{15}+1.764\times10^{15}-3.554\times10^{15}\times(1-\cos\beta)}{3.5364\times10^{15}}$

Simplify and solve for $\cos\beta$:
$\cos 5°\times3.5364\times10^{15} = 1.77241\times10^{15} + 1.764\times10^{15} - 3.554\times10^{15} + 3.554\times10^{15}\cos\beta$
$3.554\times10^{15}\cos\beta = \cos 5°\times3.5364\times10^{15} - 1.77241\times10^{15} - 1.764\times10^{15} + 3.554\times10^{15}$

Solve for $\cos\beta$ to find the longitudinal separation β.

Step 4: Rearrange the equation to isolate $\cos\beta$: **0.25**
$0.996 = \frac{177241\times10^{10}+176400\times10^{10}-2\times177780.3\times10^{10}\times(1-\cos\beta)}{2\times176820\times10^{10}}$

Multiply both sides by $2\times176820\times10^{10}$:
$0.996\times2\times176820\times10^{10} = 177241\times10^{10} + 176400\times10^{10} - 2\times177780.3\times10^{10}\times(1-\cos\beta)$
Calculate the left side:
$353640\times10^{10}\times0.996 = 352226.64\times10^{10}$
Substitute back into the equation:
$352226.64\times10^{10} = 177241\times10^{10} + 176400\times10^{10} - 2\times177780.3\times10^{10} + 2\times177780.3\times10^{10}\,\text{cc}$
Simplify and solve for $\cos\beta$:
$352226.64\times10^{10} = 353641\times10^{10} - 2\times177780.3\times10^{10} + 2\times177780.3\times10^{10}\cos\beta$
$2\times177780.3\times10^{10}\cos\beta = 353641\times10^{10} - 352226.64\times10^{10}$
$2\times177780.3\times10^{10}\cos\beta = 1414.36\times10^{10}$
$\cos\beta = \frac{1414.36\times10^{10}}{2\times177780.3\times10^{10}}$
$\cos\beta = \frac{1414.36}{355560.6}$
$\cos\beta \approx 0.00398$
Thus, the longitudinal separation β can be found by taking the inverse cosine:
$\beta = \cos^{-1}(0.00398)$

Calculate β to find the longitudinal separation in degrees.

Step 5: To find the angle β, use the inverse cosine function: **0.1**
$\beta = \arccos(0.996)$
Calculating this gives:
$\beta \approx 5.73°$
Thus, the longitudinal separation between the two satellites is approximately $5.73°$.

Step 6: To find the longitudinal separation β, use the inverse cosine function on the computed cosine value: **0.96**
$\beta = \cos^{-1}(0.996)$

Calculating this gives:
$\beta \approx 5.126°$

Thus, the longitudinal separation between the two satellites is approximately $5.126°$.

Figure 16. Illustration of a solution sample of a problem from the Satellite Communication (SC) category.

Question

Consider the communication system shown in the Figure. The symbols S_n are independent and equally likely, drawn from a BPSK constellation with amplitude ± 2. The channel output can be written as:
$v_n = S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n$
where h_n is real-valued, and w_n denotes discrete-time, zero-mean real-valued AWGN with variance σ_w^2. The equalizer output can be written as:
$u_n = v_n g_0 + v_{n-1} g_1$

1. Compute the optimum linear MMSE equalizer coefficients g_0 and g_1 in terms of h_n and σ_w^2.
2. Compute the MMSE in terms of h_n and σ_w^2.

Image

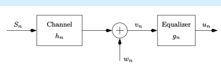


Figure 5.5: Figure for Example 5.1.1.

Ground Truth: Decomposed Reasoning Steps

***Step 1**: Since all the parameters in this problem are real-valued, we expect the equalizer coefficients also to be real-valued. Define* **1.0**
$e_n = S_n - u_n$
$e_n = S_n - v_n g_0 - v_{n-1} g_1$

We need to find g_0 and g_1 which minimize $E\left[e_n^2\right]$

***Step 2**: Differentiating $E\left[e_n^2\right]$ with respect to g_0 and equating to zero, we get:* **1.0**
$\frac{\partial}{\partial g_0} E\left[e_n^2\right] = E\left[2 e_n \frac{\partial e_n}{\partial g_0}\right] = -2E\left[e_n v_n\right] = 0$
$\Rightarrow E\left[(S_n - g_0 v_n - g_1 v_{n-1}) v_n\right] = 0$
$\Rightarrow g_0 R_{vv,0} + g_1 R_{vv,1} = h_0 P_{\mathrm{av}}$
where:
$R_{vv,m} = E\left[v_n v_{n-m}\right]$
$P_{\mathrm{av}} = E\left[S_n^2\right] = 4$

Observe the absence of the factor $1/2$ in the definition of $R_{vv,m}$, since v_n is a real-valued random process.

***Step 3**: Similarly, differentiating $E\left[e_n^2\right]$ with respect to g_1 and equating to zero, we get:* **1.0**
$\frac{\partial}{\partial g_1} E\left[e_n^2\right] = E\left[2 e_n \frac{\partial e_n}{\partial g_1}\right] = -2E\left[e_n v_{n-1}\right] = 0$
$\Rightarrow E\left[(S_n - g_0 v_n - g_1 v_{n-1}) v_{n-1}\right] = 0$

$\Rightarrow g_0 R_{vv,1} + g_1 R_{vv,0} = 0.$

***Step 4**: Solving for g_0 and g_1 from the equations, we get:* **1.0**

$g_0 = \frac{R_{vv,0} h_0 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}$
$g_1 = \frac{-R_{vv,1} h_0 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}$
where:
$R_{vv,0} = P_{\mathrm{av}}\left(h_0^2 + h_1^2 + h_2^2\right) + \sigma_w^2$
$R_{vv,1} = P_{\mathrm{av}}\left(h_0 h_1 + h_1 h_2\right)$

***Step 5**: Using the principle of orthogonality, the MMSE is:* **1.0**

$E\left[e_n^2\right] = E\left[e_n\left(S_n - g_0 v_n - g_1 v_{n-1}\right)\right]$
$= E\left[e_n S_n\right]$
$= E\left[(S_n - g_0 v_n - g_1 v_{n-1}) S_n\right]$
$= P_{\mathrm{av}} - g_0 E\left[v_n S_n\right]$
$= P_{\mathrm{av}} - g_0 E\left[S_n\left(h_0 S_n + h_1 S_{n-1} + h_2 S_{n-2} + w_n\right)\right]$
$= P_{\mathrm{av}} - g_0 h_0 P_{\mathrm{av}}$
$= P_{\mathrm{av}}\left[1 - \frac{R_{vv,0} h_0^2 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}\right].$

***Step 6**: Note that the MMSE could be reduced by:* **1.0**

1. Increasing the number of equalizer coefficients, and/or
2. Defining the error signal as $e_n = S_{n-\mathscr{D}} - u_n$, where $\mathscr{D}$ is a positive integer.

***Step 7**: From the above discussion, it is clear that computing the optimum equalizer tap weights is computationally complex, especially when L is large. In the next section, we discuss the steepest descent algorithm and the least mean square algorithm which are used in practice to obtain near-optimal equalizer performance at a reduced complexity.* **1.0**

Labeled Reasoning Steps Generated by ApPA

Step 1: To compute the optimum linear MMSE equalizer coefficients g_0 and g_1, we start by defining the error term e_n as the difference between the desired signal S_n and the equalizer output u_n: **0.88**
$e_n = S_n - u_n = S_n - (v_n g_0 + v_{n-1} g_1)$
Substitute the expressions for v_n and v_{n-1}:
$v_n = S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n$
$v_{n-1} = S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1}$
The error term becomes:
$e_n = S_n - ((S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n) g_0 + (S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1}) g_1)$
The objective is to minimize the expected value of the squared error $E[e_n^2]$. This is achieved by setting the partial derivatives of $E[e_n^2]$ with respect to g_0 and g_1 to zero and solving the resulting equations. This leads to a system of linear equations in terms of g_0 and g_1, which can be solved to find the optimum coefficients.

Step 2: To find the optimal coefficients g_0 and g_1 that minimize the mean squared error $E[e_n^2]$, we first express the error term: **0.72**
$e_n = S_n - (v_n g_0 + v_{n-1} g_1)$
The mean squared error is:
$E[e_n^2] = E\left[(S_n - v_n g_0 - v_{n-1} g_1)^2\right]$
To minimize this, we differentiate $E[e_n^2]$ with respect to g_0 and g_1, set the derivatives to zero, and solve the resulting equations. The derivatives are:
$\frac{\partial}{\partial g_0} E[e_n^2] = -2E[(S_n - v_n g_0 - v_{n-1} g_1) v_n] = 0$
$\frac{\partial}{\partial g_1} E[e_n^2] = -2E[(S_n - v_n g_0 - v_{n-1} g_1) v_{n-1}] = 0$
These lead to the normal equations:
$E[S_n v_n] = g_0 E[v_n^2] + g_1 E[v_n v_{n-1}]$
$E[S_n v_{n-1}] = g_0 E[v_n v_{n-1}] + g_1 E[v_{n-1}^2]$
Solving these equations will yield the optimal coefficients g_0 and g_1.

Step 3: To find the optimum linear MMSE equalizer coefficients, we differentiate the mean squared error with respect to g_1 and set it to zero: **0.41**
$\frac{\partial}{\partial g_1} E[e_n^2] = E\left[2 e_n \frac{\partial e_n}{\partial g_1}\right] = -2E[e_n v_{n-1}] = 0$
This leads to:
$E\left[(S_n - g_0 v_n - g_1 v_{n-1}) v_{n-1}\right] = 0$
Substituting the expression for e_n and simplifying, we get:
$g_0 R_{vv,1} + g_1 R_{vv,0} = h_1 P_{\mathrm{av}}$
Now, we have the system of equations:
1. $g_0 R_{vv,0} + g_1 R_{vv,1} = h_0 P_{\mathrm{av}}$
2. $g_0 R_{vv,1} + g_1 R_{vv,0} = h_1 P_{\mathrm{av}}$
These equations can be solved simultaneously to find the values of g_0 and g_1.

Step 4: Solve the system of linear equations obtained from the differentiation steps to find the optimum linear MMSE equalizer coefficients g_0 and g_1. **0.67**
The system of equations derived is:
1. $g_0 R_{vv,0} + g_1 R_{vv,1} = h_0 P_{\mathrm{av}}$
2. $g_0 R_{vv,1} + g_1 R_{vv,0} = 0$
Substitute $P_{\mathrm{av}} = 4$ and solve these equations simultaneously. From equation 2, express g_1 in terms of g_0:
$g_1 = -\frac{R_{vv,1}}{R_{vv,0}} g_0$
Substitute this expression for g_1 into equation 1:
$g_0 R_{vv,0} - \frac{R_{vv,1}}{R_{vv,0}} g_0 R_{vv,1} = h_0 \times 4$

Simplify and solve for g_0:
$g_0\left(R_{vv,0} - \frac{R_{vv,1}^2}{R_{vv,0}}\right) = 4h_0$
$g_0 = \frac{4h_0}{R_{vv,0} - \frac{R_{vv,1}^2}{R_{vv,0}}}$

Now, substitute g_0 back into the expression for g_1:
$g_1 = -\frac{R_{vv,1}}{R_{vv,0}} \times \frac{4h_0}{R_{vv,0} - \frac{R_{vv,1}^2}{R_{vv,0}}}$
This provides the explicit solutions for the coefficients g_0 and g_1 in terms of h_n and σ_w^2.

Step 5: To compute the MMSE, we use the expression for the mean square error: **0.34**
$E[e_n^2] = E\left[(S_n - g_0 v_n - g_1 v_{n-1})^2\right]$
Substituting the expressions for g_0 and g_1 from Step 4, we have:
$E[e_n^2] = P_{\mathrm{av}} - 2g_0 E[S_n v_n] - 2g_1 E[S_n v_{n-1}] + g_0^2 R_{vv,0} + 2g_0 g_1 R_{vv,1} + g_1^2 R_{vv,0}$
Using the orthogonality principle, $E[S_n v_n] = h_0 P_{\mathrm{av}}$ and $E[S_n v_{n-1}] = 0$, the MMSE simplifies to:
$\mathrm{MMSE} = P_{\mathrm{av}} - g_0 h_0 P_{\mathrm{av}}$
Substituting g_0 from Step 4:
$\mathrm{MMSE} = P_{\mathrm{av}} - \frac{R_{vv,0} h_0^2 P_{\mathrm{av}}^2}{R_{vv,0}^2 - R_{vv,1}^2}$
Thus, the MMSE is:
$\mathrm{MMSE} = \frac{P_{\mathrm{av}}\left(R_{vv,0}^2 - R_{vv,1}^2 - R_{vv,0} h_0^2 P_{\mathrm{av}}\right)}{R_{vv,0}^2 - R_{vv,1}^2}$

Step 6: To further reduce the MMSE, consider increasing the number of equalizer coefficients or introducing a delay in the error signal definition. However, given the current setup with two coefficients, the MMSE is already minimized with the derived coefficients g_0 and g_1. Therefore, the MMSE is given by: **0.83**

$E\left[e_n^2\right] = P_{\mathrm{av}}\left[1 - \frac{R_{vv,0} h_0^2 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}\right]$

This expression represents the minimized mean square error for the given system configuration, utilizing the optimal linear MMSE equalizer coefficients.

Step 7: To further optimize the equalizer coefficients g_0 and g_1 with reduced computational complexity, consider using iterative algorithms such as the Least Mean Squares (LMS) algorithm. This approach updates the coefficients iteratively based on the error signal e_n: **0.96**
1. Initialize g_0 and g_1 with small random values.
2. For each iteration k, update the coefficients as follows:
   $g_0^{(k+1)} = g_0^{(k)} + \mu e_n v_n$
   $g_1^{(k+1)} = g_1^{(k)} + \mu e_n v_{n-1}$
   where μ is the step size parameter controlling the convergence speed and stability.
3. Continue the iterations until the change in coefficients is below a predefined threshold or a maximum number of iterations is reached.

This iterative approach allows for practical implementation in real-time systems, providing near-optimal solutions with manageable computational resources.

Step 1: To compute the optimum linear MMSE equalizer coefficients g_0 and g_1, we first define the error term e_n as the difference between the desired signal S_n and the equalizer output u_n: **0.88**
$e_n = S_n - u_n = S_n - (v_n g_0 + v_{n-1} g_1)$
The objective is to minimize the expected value of the squared error, $\mathbb{E}[e_n^2]$. Substituting the expression for v_n and v_{n-1} into the error term, we have:
$e_n = S_n - (S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n) g_0 - (S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1}) g_1$
Expanding and rearranging terms, the error becomes:
$e_n = S_n(1 - h_0 g_0) - S_{n-1}(h_1 g_0 + h_0 g_1) - S_{n-2}(h_2 g_0 + h_1 g_1) - S_{n-3} h_2 g_1 - w_n g_0 - w_{n-1} g_1$
The MMSE solution requires setting the partial derivatives of $\mathbb{E}[e_n^2]$ with respect to g_0 and g_1 to zero. This results in a set of linear equations that can be solved to find the optimal coefficients g_0 and g_1.

Step 2: To find the optimal coefficients g_0 and g_1 that minimize the mean squared error $E[e_n^2]$, we first express the error e_n as: **0.72**
$e_n = S_n - (v_n g_0 + v_{n-1} g_1)$
Substituting the expression for v_n and v_{n-1}, we have:
$v_n = S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n$
$v_{n-1} = S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1}$
Thus, the error becomes:
$e_n = S_n - [(S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n) g_0 + (S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1}) g_1]$

To minimize $E[e_n^2]$, differentiate with respect to g_0 and g_1, set the derivatives to zero, and solve:
1. Differentiate $E[e_n^2]$ with respect to g_0:
$\frac{\partial}{\partial g_0} E[e_n^2] = -2E\left[e_n(S_n h_0 + S_{n-1} h_1 + S_{n-2} h_2 + w_n)\right] = 0$

2. Differentiate $E[e_n^2]$ with respect to g_1:
$\frac{\partial}{\partial g_1} E[e_n^2] = -2E\left[e_n(S_{n-1} h_0 + S_{n-2} h_1 + S_{n-3} h_2 + w_{n-1})\right] = 0$
Solving these equations will yield the optimal g_0 and g_1 in terms of h_n and σ_w^2.

Step 3: Differentiate the error expression with respect to g_1 and set it to zero: **0.72**
$\frac{\partial}{\partial g_1} E\left[e_n^2\right] = E\left[2 e_n \frac{\partial e_n}{\partial g_1}\right] = -2E\left[e_n v_{n-1}\right] = 0$
$\Rightarrow E\left[(S_n - g_0 v_n - g_1 v_{n-1}) v_{n-1}\right] = 0$
$\Rightarrow g_0 R_{vv,1} + g_1 R_{vv,0} = h_1 P_{\mathrm{av}}$
Now, solve the system of linear equations obtained from differentiating with respect to g_0 and g_1:
$g_0 R_{vv,0} + g_1 R_{vv,1} = h_0 P_{\mathrm{av}}$
$g_0 R_{vv,1} + g_1 R_{vv,0} = h_1 P_{\mathrm{av}}$

This system can be solved for g_0 and g_1 using matrix methods or substitution, where $R_{vv,m}$ are the autocorrelation terms of v_n.

Step 4: To solve for the optimum linear MMSE equalizer coefficients g_0 and g_1, we have the following system of linear equations derived from the conditions for minimizing the mean squared error: **0.24**
1. $g_0 R_{vv,0} + g_1 R_{vv,1} = h_0 P_{\mathrm{av}}$
2. $g_0 R_{vv,1} + g_1 R_{vv,0} = 0$
Given that $P_{\mathrm{av}} = 4$, we can express these equations as:
$g_0 R_{vv,0} + g_1 R_{vv,1} = 4h_0$
$g_0 R_{vv,1} + g_1 R_{vv,0} = 0$
To solve for g_0 and g_1, we can use substitution or elimination. From the second equation, solve for g_1:
$g_1 = -\frac{g_0 R_{vv,1}}{R_{vv,0}}$

Substitute this expression for g_1 into the first equation:
$g_0 R_{vv,0} - \frac{g_0 R_{vv,1}^2}{R_{vv,0}} = 4h_0$
Simplify and solve for g_0:
$g_0\left(R_{vv,0} - \frac{R_{vv,1}^2}{R_{vv,0}}\right) = 4h_0$
$g_0 = \frac{4h_0 R_{vv,0}}{R_{vv,0}^2 - R_{vv,1}^2}$
Substitute g_0 back into the expression for g_1:
$g_1 = -\frac{4h_0 R_{vv,1}}{R_{vv,0}^2 - R_{vv,1}^2}$

Thus, the optimum linear MMSE equalizer coefficients are:
$g_0 = \frac{4h_0 R_{vv,0}}{R_{vv,0}^2 - R_{vv,1}^2}, \quad g_1 = -\frac{4h_0 R_{vv,1}}{R_{vv,0}^2 - R_{vv,1}^2}$

Step 5: To compute the MMSE, we use the expression for the error $e_n = S_n - u_n$, where $u_n = v_n g_0 + v_{n-1} g_1$. The MMSE is given by: **0.11**
$\mathrm{MMSE} = E[e_n^2] = E\left[(S_n - v_n g_0 - v_{n-1} g_1)^2\right].$
Substituting the expressions for v_n and v_{n-1}, and using the orthogonality principle, we have:
$\mathrm{MMSE} = P_{\mathrm{av}} - (g_0 h_0 P_{\mathrm{av}} + g_1 h_1 P_{\mathrm{av}}),$
where $P_{\mathrm{av}} = 4$. Substituting the values of g_0 and g_1 from Step 4:
$g_0 = \frac{R_{vv,0} h_0 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}, \quad g_1 = \frac{-R_{vv,1} h_0 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}.$
Substitute these into the MMSE expression:
$\mathrm{MMSE} = P_{\mathrm{av}} - \left(\frac{R_{vv,0} h_0^2 P_{\mathrm{av}}^2}{R_{vv,0}^2 - R_{vv,1}^2} - \frac{R_{vv,1} h_0 h_1 P_{\mathrm{av}}^2}{R_{vv,0}^2 - R_{vv,1}^2}\right).$
Simplifying, we obtain:
$\mathrm{MMSE} = 4 - \frac{h_0^2 P_{\mathrm{av}}^2 (R_{vv,0} - R_{vv,1})}{R_{vv,0}^2 - R_{vv,1}^2}.$

This is the expression for the MMSE in terms of h_n and σ_w^2.

Step 6: To compute the MMSE using the derived expressions for g_0 and g_1, substitute these into the MMSE expression: **0.98**
$E\left[e_n^2\right] = P_{\mathrm{av}}\left[1 - \frac{R_{vv,0} h_0^2 P_{\mathrm{av}}}{R_{vv,0}^2 - R_{vv,1}^2}\right]$
Substitute the values of $R_{vv,0}$ and $R_{vv,1}$:
$R_{vv,0} = P_{\mathrm{av}}\left(h_0^2 + h_1^2 + h_2^2\right) + \sigma_w^2$
$R_{vv,1} = P_{\mathrm{av}}\left(h_0 h_1 + h_1 h_2\right)$

Thus, the MMSE becomes:
$E\left[e_n^2\right] = 4\left[1 - \frac{\left(P_{\mathrm{av}}\left(h_0^2 + h_1^2 + h_2^2\right) + \sigma_w^2\right) h_0^2 \cdot 4}{\left(P_{\mathrm{av}}\left(h_0^2 + h_1^2 + h_2^2\right) + \sigma_w^2\right)^2 - \left(P_{\mathrm{av}}(h_0 h_1 + h_1 h_2)\right)^2}\right]$
This expression gives the MMSE in terms of h_n and σ_w^2.

Step 7: To further optimize the equalizer coefficients with reduced computational complexity, iterative optimization algorithms such as the steepest descent or least mean square (LMS) algorithms can be employed. These algorithms iteratively adjust the coefficients g_0 and g_1 to minimize the mean square error $E[e_n^2]$. The LMS algorithm, for instance, updates the coefficients using the rule: **0.9**
$g_0^{(k+1)} = g_0^{(k)} + \mu e_n v_n$
$g_1^{(k+1)} = g_1^{(k)} + \mu e_n v_{n-1}$
where μ is the step size parameter that controls the convergence speed and stability of the algorithm. By iteratively applying these updates, the coefficients converge to values that approximate the optimal solution, balancing computational efficiency and accuracy.

Figure 17. Illustration of a solution sample of a problem from the Communication Signal Processing (CSP) category.

Question Q

Question Q: Compute the power spectral density of a random binary wave shown in Figure 4.7, with symbols $+A$ and 0 occurring with equal probability. Assume that the symbols are independent.

Image V

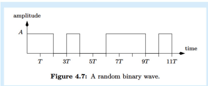


Figure 4.7: A random binary wave.

Ground Truth: Decomposed Reasoning Steps

Step 1: *Here, the transmit filter is defined as:*
$p(t) = 1$ *for* $0 < t < T$, *and* $p(t) = 0$ *elsewhere.*
Equivalently:
$p(t) \triangleq \text{rect}\left(\frac{t-T/2}{T}\right)$.
Hence, the magnitude of the Fourier transform of the transmit filter is:
$|\tilde{P}(F)| = T\left|\frac{\sin(\pi FT)}{\pi FT}\right| \triangleq T|\,\text{sinc}(FT)|$.

Step 2: The constellation has two points, namely $\{0, A\}$. Since the signal in this example is real-valued, we do not use the factor of half in the autocorrelation. Hence, we have:
For $R_{SS,l}$:
- $R_{SS,l} = A^2/2$ for $l = 0$,
- $R_{SS,l} = A^2/4$ for $l \neq 0$.

This can also be expressed as:
- $R_{SS,l} = \sigma_S^2 + m_S^2$ for $l = 0$,
- $R_{SS,l} = m_S^2$ for $l \neq 0$.

Or equivalently:
$R_{SS,l} = m_S^2 + \sigma_S^2\delta_K(l)$,
where m_S and σ_S^2 denote the mean and variance of the symbols, respectively.

In this example, assuming that the symbols are equally likely:
$m_S = E[S_k] = A/2$,
$\sigma_S^2 = E\left[(S_k - m_S)^2\right] = A^2/4$.

Step 3: The discrete-time Fourier transform of $R_{SS,l}$ is given by:
$S_{\mathscr{P},S}(F) = \sigma_S^2 + m_S^2\sum_{k=-\infty}^{\infty}\exp(-\text{j}2\pi FkT)$.
From the Fourier transform of the sampling theorem, we have:
$\sum_{k=-\infty}^{\infty}\exp(-\text{j}2\pi FkT) = \frac{1}{T}\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)$.
Hence, the power spectral density becomes:
$S_{\mathscr{P},S}(F) = \sigma_S^2 + \frac{m_S^2}{T}\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)$.

Step 4: The overall power spectrum is given by:
$S_{\tilde{S}}(F) = \frac{1}{T}\left(\sigma_S^2 + \frac{m_S^2}{T}\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)\right)|\tilde{P}(F)|^2$.

Substituting for $\tilde{P}(F)$, we have:
$S_{\tilde{S}}(F) = \frac{1}{T}\left(\sigma_S^2 + \frac{m_S^2}{T}\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)\right)T^2\,\text{sinc}^2(FT)$.

For $\delta_D\left(F - \frac{k}{T}\right)\text{sinc}^2(FT)$:
- It is 0 for $F = k/T, k \neq 0$,
- It becomes $\delta_D(F)$ for $F = 0$,
- It is 0 elsewhere.

Hence, the power spectrum simplifies to:
$S_{\tilde{S}}(F) = \frac{A^2T}{4}\text{sinc}^2(FT) + \frac{A^2}{4}\delta_D(F)$.

Step 5: Equation (4.30) suggests that to avoid the spectral lines at k/T (the summation term in (4.30)), the constellation must have zero mean. In this case, $R_{SS,l}$ becomes a Kronecker delta function.

Step 6: To find the power spectral density of the transmitted signal $s_p(t)$, once again consider the random process:
$\tilde{A}(t) = \tilde{S}(t)\exp\left(\text{j}\left(2\pi F_c t + \theta\right)\right)$
where $\tilde{S}(t)$ is the random process in (4.14) and θ is a uniformly distributed random variable in the interval $[0, 2\pi)$. Define the random process:
$S_p(t) \triangleq \Re\{\tilde{A}(t)\} = \frac{1}{2}\left(\tilde{A}(t) + \tilde{A}^*(t)\right)$.
Note that $s_p(t)$ in (4.6) is a particular realization of the random process in (4.35) for $\alpha = 0$ and $\theta = 0$. Now, the autocorrelation of the random process in (4.35) is given by (since $S_p(t)$ is real-valued, we do not use the factor $1/2$ in the autocorrelation function):
$R_{S_pS_p}(\tau) = E\left[S_p(t)S_p(t-\tau)\right]$
$= \frac{1}{4}E\left[\left(\tilde{A}(t) + \tilde{A}^*(t)\right)\left(\tilde{A}(t-\tau) + \tilde{A}^*(t-\tau)\right)\right]$
$= \Re\left\{\tilde{R}_{\tilde{S}\tilde{S}}(\tau)\exp\left(\text{j}2\pi F_c\tau\right)\right\}$.
We have used the fact that the random variables θ, α, and S_k are statistically independent, hence:
$E[\tilde{A}(t)\tilde{A}(t-\tau)] = E\left[\tilde{A}^*(t)\tilde{A}^*(t-\tau)\right] = 0$
and:
$\frac{1}{2}E\left[\tilde{A}(t)\tilde{A}^*(t-\tau)\right] = \tilde{R}_{\tilde{S}\tilde{S}}(\tau)\exp\left(\text{j}2\pi F_c\tau\right)$.

Step 7: Thus the power spectral density of $S_p(t)$ in (4.35) is given by
$S_{S_p}(F) = \frac{1}{2}\left[S_{\tilde{S}}\left(F - F_c\right) + S_{\tilde{S}}\left(-F - F_c\right)\right]$.

Decomposed Reasoning Step Generated by Llama-3.2-90B

Step	***Ll***	***QW:***
Step 1	**0.65**	**0.72**
Step 2	**0.57**	**0.79**
Step 3	**0.7**	**0.89**
Step 4	**0.72**	**0.81**
Step 5	**0.75**	**0.88**
Step 6	**0.8**	**0.92**
Step 7	**0.87**	**0.99**

Step 1: To compute the power spectral density (PSD) of the random binary wave, we first identify the transmit filter function. Since the symbols are binary with values $+A$ and 0, the transmit filter can be modeled as a rectangular pulse of duration T seconds. The Fourier transform of a rectangular pulse of amplitude A and duration T is given by:
$H(f) = AT\cdot\text{sinc}(fT)$
where $\text{sinc}(x) = \frac{\sin(\pi x)}{\pi x}$.
The magnitude of the Fourier transform is:
$|H(f)| = |AT\cdot\text{sinc}(fT)| = AT\cdot|\text{sinc}(fT)|$

This expression represents the magnitude of the Fourier transform of the transmit filter function, which is essential for determining the power spectral density of the random binary wave.

Step 2: The random binary wave consists of symbols $+A$ and 0, each occurring with equal probability. Therefore, the mean of the symbols is given by:
$\mu = \frac{1}{2}(A + 0) = \frac{A}{2}$
The variance of the symbols is calculated as:
$\sigma^2 = \frac{1}{2}((A-\mu)^2 + (0-\mu)^2) = \frac{1}{2}\left(\left(A - \frac{A}{2}\right)^2 + \left(0 - \frac{A}{2}\right)^2\right) = \frac{1}{2}\left(\left(\frac{A}{2}\right)^2 + \left(\frac{A}{2}\right)^2\right) = \frac{A^2}{4}$

The autocorrelation function $R(\tau)$ for the random binary wave is given by:
$R(\tau) = \sigma^2\cdot p(\tau) = \frac{A^2}{4}\cdot\text{rect}\left(\frac{\tau}{T}\right)$
This expression for $R(\tau)$ will be used in subsequent steps to compute the power spectral density.

Step 3: To compute the power spectral density (PSD), we take the discrete-time Fourier transform of the autocorrelation function $R_{SS,l}$. The PSD $S(F)$ is given by:
$S(F) = \sum_{l=-\infty}^{\infty}R_{SS,l}e^{-j2\pi Fl}$
Substituting the values of $R_{SS,l}$ from Step 2:
$S(F) = \left(m_S^2 + \sigma_S^2\delta_K(l)\right) * \delta(F)$
Since $m_S = A/2$ and $\sigma_S^2 = A^2/4$, we have:
$S(F) = \left(\left(\frac{A}{2}\right)^2 + \frac{A^2}{4}\delta_K(l)\right) * \delta(F)$
This simplifies to:
$S(F) = \frac{A^2}{4} + \frac{A^2}{4}\sum_{l=-\infty}^{\infty}\delta(F - l/T)$
Thus, the power spectral density of the random binary wave is:
$S(F) = \frac{A^2}{4}\left(1 + \sum_{l=-\infty}^{\infty}\delta(F - l/T)\right)$

Step 4: To find the overall power spectral density, multiply the power spectral density of the signal, $S_{\mathscr{P},S}(F)$, by the magnitude squared of the Fourier transform of the transmit filter, $|\tilde{P}(F)|^2$. Thus, the power spectral density $S_X(F)$ is given by:
$S_X(F) = \left(\sigma_S^2 + \frac{m_S^2}{T}\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)\right)\cdot T^2|\text{sinc}(FT)|^2$
Simplifying, we have:
$S_X(F) = \sigma_S^2T^2|\text{sinc}(FT)|^2 + m_S^2T\sum_{k=-\infty}^{\infty}\delta_D\left(F - \frac{k}{T}\right)$
This expression represents the power spectral density of the random binary wave, incorporating both the continuous spectrum due to the variance and the discrete spectrum due to the mean.

Step 5: To avoid spectral lines in the power spectral density, we need to ensure that the signal constellation has zero mean. This can be achieved by adjusting the mean of the symbols to zero. In this case, the symbols are $\{0, A\}$, and the mean is $m_S = A/2$. To make the mean zero, we can redefine the symbols as $\{-A/2, A/2\}$, which results in $m_S = 0$.
Consequently, the autocorrelation function becomes a Kronecker delta function, simplifying the power spectral density to:
$S_{\tilde{S}}(F) = \frac{A^2T}{4}\text{sinc}^2(FT)$
This eliminates unwanted frequency components, ensuring no spectral lines are present in the power spectral density.

Step 6: To finalize the power spectral density, we recognize that the power spectral density of the modulated signal is given by the Fourier transform of the autocorrelation function derived in Step 4.
The expression for the power spectral density is:
$S_{\tilde{S}}(F) = \frac{A^2T}{4}\text{sinc}^2(FT) + \frac{A^2}{4}\delta_D(F)$
This expression indicates that the power spectral density consists of a continuous part, $\frac{A^2T}{4}\text{sinc}^2(FT)$, which represents the energy spread over the frequency spectrum, and a discrete part, $\frac{A^2}{4}\delta_D(F)$, which represents the spectral line at zero frequency due to the non-zero mean of the signal. The presence of the $\text{sinc}^2(FT)$ function reflects the bandwidth limitation imposed by the rectangular pulse shape of the transmit filter.

Step 7: To find the power spectral density of the real-valued signal $S_p(t)$, we use the property that the power spectral density of the real part of a complex process is the average of the power spectral densities of the complex process shifted by the carrier frequency F_c and its negative. Thus, the power spectral density $S_{S_p}(F)$ is given by:
$S_{S_p}(F) = \frac{1}{2}\left(S_{\tilde{S}}(F - F_c) + S_{\tilde{S}}(F + F_c)\right)$
Substituting $S_{\tilde{S}}(F)$ from Step 4, we have:
$S_{S_p}(F) = \frac{1}{2}\left(\left(\frac{A^2T}{4}\text{sinc}^2((F - F_c)T) + \frac{A^2}{4}\delta_D(F - F_c)\right) + \left(\frac{A^2T}{4}\text{sinc}^2((F + F_c)T) + \frac{A^2}{4}\delta_D(F + F_c)\right)\right)$
This expression represents the power spectral density of the real-valued signal $S_p(t)$. ✓

QW Attention Map:

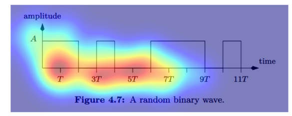
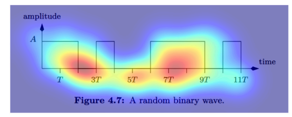
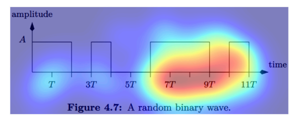
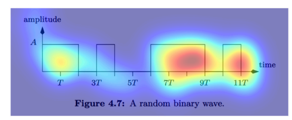
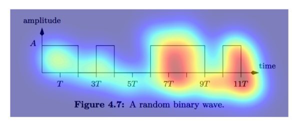
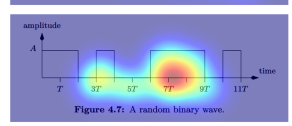
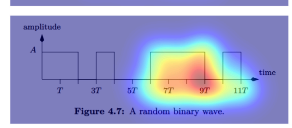
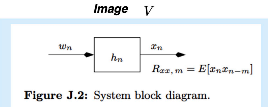








Figure 18. Illustration of the full version of Fig. 9, obtained by performing problem-solving with Llama-3.2-90B while using Qwen-2-VL-72B (QW) and Llama-3.2-11B (Ll) as verifiers. The problem belongs to the CSP category.

Question Q

Question Q: Consider the system in Figure. The input w_n is a real-valued, zero-mean white noise process with autocorrelation defined as:
$R_{ww,m} = E\left[w_n w_{n-m}\right] = 2\delta_K(m)$
where $\delta_K(\cdot)$ is the Kronecker delta function. The impulse response of the filter is given by:
$h_n = \delta_K(n) + 2\delta_K(n-1) + 3\delta_K(n-2)$
Using the Levinson-Durbin algorithm, compute the coefficients of the optimum 3rd -order forward predictor.

Image V



Ground Truth: Decomposed Reasoning Steps

***Step 1**: In general, the filter output x_n is given by the discrete-time convolution*
$x_n = \sum_{k=-\infty}^{\infty} h_k w_{n-k}$
Observe that all variables in this problem are real-valued. Therefore, the autocorrelation of x_n is given by
$R_{xx,m} = E\left[x_n x_{n-m}\right]$
$R_{xx,m} = E\left[\sum_{k=-\infty}^{\infty} h_k w_{n-k} \sum_{l=-\infty}^{\infty} h_l w_{n-m-l}\right]$
$= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} h_k h_l E\left[w_{n-k} w_{n-m-l}\right]$
$= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} 2 h_k h_l \delta_K(m+l-k)$
$= \sum_{k=-\infty}^{\infty} 2 h_k h_{k-m}$
$= 2R_{hh,m}$
where $R_{hh,m}$ denotes the autocorrelation of h_n. Clearly,
$R_{xx,0} = 28$
$R_{xx,1} = 16$
$R_{xx,2} = 6$
$R_{xx,m} = 0 \quad \text{for } |m| \geq 3$

***Step 2**: Therefore,*
$K_1 = -\frac{R_{xx,1}}{R_{xx,0}} = -0.5714 = a_{1,1}$
Hence,
$\mathscr{E}_1^f = \left(1 - K_1^2\right)\mathscr{E}_0^f$
$= \left(1 - K_1^2\right) R_{xx,0}$
$= 18.8571$

***Step 3**: Similarly,*
$K_2 = -\frac{\sum_{k=0}^{1} a_{1,k} R_{xx,2-k}}{\mathscr{E}_1^f} = 0.1666 = a_{2,2}$
Therefore, the second-order forward predictor coefficients are
$a_{2,0} = 1$
$a_{2,1} = a_{1,1}\left(1 + K_2\right)$
$= -0.6665$
$a_{2,2} = 0.1666$
and the variance of the prediction error at the output of the second-order predictor is
$\mathscr{E}_2^f = \left(1 - K_2^2\right)\mathscr{E}_1^f$
$= 18.3337$

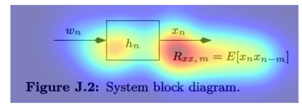
***Step 4**: Again,*
$K_3 = -\frac{\sum_{k=0}^{2} a_{2,k} R_{xx,3-k}}{\mathscr{E}_2^f} = 0.0727 = a_{3,3}$
and finally, the coefficients of the optimum 3rd-order forward predictor are
$a_{3,0} = 1$
$a_{3,1} = a_{2,1} + K_3 a_{2,2} = -0.6543$
$a_{3,2} = a_{2,2} + K_3 a_{2,1} = 0.1181$
$a_{3,3} = 0.0727$

Decomposed Reasoning Step Generated by Llama-3.2-90B | ***Ll*** | ***QW:*** | ***QW Attention Map:***

Step 1: We aim to compute the coefficients for a third-order predictor using the Levinson-Durbin recursion. The given autocorrelation values are:
$R_{xx,0} = 28, \quad R_{xx,1} = 16, \quad R_{xx,2} = 6, \quad R_{xx,3} = 0.$

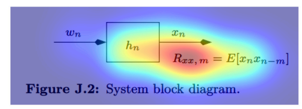
Start by initializing:
$E_0 = R_{xx,0} = 28.$

Ll: **0.79** QW: **0.91**



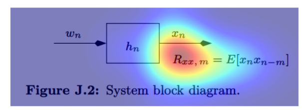
Step 2: The reflection coefficient k_1 is calculated as:
$k_1 = -\frac{R_{xx,1}}{E_0} = -\frac{16}{28} = -\frac{4}{7}.$
Update the prediction error:
$E_1 = (1 - k_1^2)E_0 = \left(1 - \left(-\frac{4}{7}\right)^2\right)28 = \left(1 - \frac{16}{49}\right)28$
$= \frac{33}{49} \cdot 28 = 18.86.$
The first predictor coefficient is:
$a_1^{(1)} = k_1 = -\frac{4}{7}.$

Ll: **0.46** QW: **0.39**



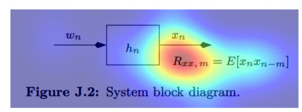
Step 3: The reflection coefficient k_2 is:
$k_2 = -\frac{R_{xx,2} + a_1^{(1)} R_{xx,1}}{E_1}.$
Substitute the values:
$k_2 = -\frac{6 + \left(-\frac{4}{7}\right)(16)}{18.86} = -\frac{6 - \frac{64}{7}}{18.86} = -\frac{\frac{42}{7} - \frac{64}{7}}{18.86} = -\frac{-\frac{22}{7}}{18.86}.$
Simplify:
$k_2 = \frac{22}{7 \cdot 18.86} = \frac{22}{131.02} \approx 0.168.$
Update the coefficients:
$a_2^{(2)} = k_2, \quad a_1^{(2)} = a_1^{(1)} + k_2 a_1^{(1)} = -\frac{4}{7} + (0.168)\left(-\frac{4}{7}\right).$
Simplify:
$a_1^{(2)} = -\frac{4}{7} - \frac{4 \cdot 0.168}{7} = -\frac{4}{7} - \frac{0.672}{7} = -\frac{4.672}{7} \approx -0.667.$
The error is updated as:
$E_2 = (1 - k_2^2)E_1 = (1 - 0.168^2)(18.86) \approx 18.40.$

Ll: **0.51** QW: **0.33**



Step 4: The reflection coefficient k_3 is:
$k_3 = -\frac{R_{xx,3} + a_1^{(2)} R_{xx,2} + a_2^{(2)} R_{xx,1}}{E_2}.$
Substitute the values:
$k_3 = -\frac{0 + (-0.667)(6) + (0.168)(16)}{18.40}.$
Simplify:
$k_3 = -\frac{-4.002 + 2.688}{18.40} = -\frac{-1.314}{18.40} = 0.0714.$
Update the coefficients:
$a_3^{(3)} = k_3, \quad a_2^{(3)} = a_2^{(2)} + k_3 a_1^{(2)}, \quad a_1^{(3)} = a_1^{(2)} + k_3 a_2^{(2)}.$
Substitute:
$a_2^{(3)} = 0.168 + (0.0714)(-0.667) = 0.168 - 0.0476 \approx 0.120.$
$a_1^{(3)} = -0.667 + (0.0714)(0.168) = -0.667 + 0.012 \approx -0.655.$
The final error is:
$E_3 = (1 - k_3^2)E_2 = (1 - 0.0714^2)(18.40) \approx 18.36.$

Ll: **0.59** QW: **0.27**



Step 5: The coefficients of the third-order predictor are:
$a_1 = -0.655,$ ✗ $a_2 = 0.120,$ ✓ $a_3 = 0.0714.$ ✗
The final prediction error is:
$E_3 \approx 18.36.$

Ll: **0.77** QW: **0.63**

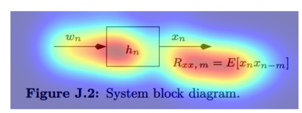


Figure 19. Illustration of the verification process obtained by performing problem-solving with Llama-3.2-90B while using Qwen-2-VL-72B (QW) and Llama-3.2-11B (Ll) as verifiers. The problem belongs to the CSP category.

Question Q

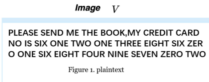
Question Q: Suppose in the multi-table substitution code

$$A=\begin{pmatrix}3&13&21&9\\15&10&6&25\\10&17&4&8\\1&23&7&2\end{pmatrix},\ B=\begin{pmatrix}1\\21\\8\\17\end{pmatrix}$$

encrypted as:
$C_i \equiv AM_i + B \pmod{26}$
The plain text is shown in the Figure.\
use the decryption transformation
$M_i \equiv A^{-1} + B \pmod{26}$
verify your result, where

$$A^{-1}=\begin{pmatrix}23&13&20&5\\0&10&11&0\\9&11&15&22\\9&22&6&25\end{pmatrix}$$

Image V



Ground Truth: Decomposed Reasoning Steps

***Step 1**: During encryption, first group the plaintext into blocks of four characters:*

$$M_1\equiv\begin{pmatrix}15\\11\\4\\0\end{pmatrix},\ M_2\equiv\begin{pmatrix}18\\4\\18\\4\end{pmatrix},\ M_3\equiv\begin{pmatrix}13\\3\\12\\4\end{pmatrix},\ M_4\equiv\begin{pmatrix}19\\7\\4\\1\end{pmatrix},$$
$$M_5\equiv\begin{pmatrix}14\\14\\10\\12\end{pmatrix},\ M_6\equiv\begin{pmatrix}24\\2\\17\\4\end{pmatrix},\ M_7\equiv\begin{pmatrix}3\\8\\19\\2\end{pmatrix},\ M_8\equiv\begin{pmatrix}0\\17\\3\\13\end{pmatrix},$$
$$M_9\equiv\begin{pmatrix}14\\8\\18\\18\end{pmatrix},\ M_{10}\equiv\begin{pmatrix}8\\23\\14\\13\end{pmatrix},\ M_{11}\equiv\begin{pmatrix}4\\19\\22\\14\end{pmatrix},\ M_{12}\equiv\begin{pmatrix}14\\13\\4\\19\end{pmatrix},$$
$$M_{13}\equiv\begin{pmatrix}7\\17\\4\\4\end{pmatrix},\ M_{14}\equiv\begin{pmatrix}4\\8\\6\\7\end{pmatrix},\ M_{15}\equiv\begin{pmatrix}19\\18\\8\\23\end{pmatrix},\ M_{16}\equiv\begin{pmatrix}25\\4\\17\\4\end{pmatrix},$$
$$M_{17}\equiv\begin{pmatrix}14\\13\\4\\18\end{pmatrix},\ M_{18}\equiv\begin{pmatrix}8\\23\\4\\8\end{pmatrix},\ M_{19}\equiv\begin{pmatrix}6\\7\\19\\5\end{pmatrix},\ M_{20}\equiv\begin{pmatrix}14\\20\\17\\13\end{pmatrix},$$
$$M_{21}\equiv\begin{pmatrix}8\\13\\4\\18\end{pmatrix},\ M_{22}\equiv\begin{pmatrix}4\\21\\4\\13\end{pmatrix},\ M_{23}\equiv\begin{pmatrix}25\\4\\17\\14\end{pmatrix},\ M_{24}\equiv\begin{pmatrix}19\\22\\14\end{pmatrix}.$$

***Step 2**: Substitute the groups into the encryption transformation* $C_i \equiv AM_i + B \pmod{26}$*, and get the calculation result:*

$$C_1\equiv\begin{pmatrix}13\\16\\23\\1\end{pmatrix},\ C_2\equiv\begin{pmatrix}1\\19\\22\\1\end{pmatrix},\ C_3\equiv\begin{pmatrix}3\\2\\9\\9\end{pmatrix},\ C_4\equiv\begin{pmatrix}8\\9\\3\\19\end{pmatrix}.$$
$$C_5\equiv\begin{pmatrix}23\\3\\2\\5\end{pmatrix},\ C_6\equiv\begin{pmatrix}24\\5\\18\\6\end{pmatrix},\ C_7\equiv\begin{pmatrix}11\\24\\6\\3\end{pmatrix},\ C_8\equiv\begin{pmatrix}12\\14\\23\\13\end{pmatrix}.$$
$$C_9\equiv\begin{pmatrix}11\\11\\6\\13\end{pmatrix},\ C_{10}\equiv\begin{pmatrix}7\\0\\15\\2\end{pmatrix},\ C_{11}\equiv\begin{pmatrix}16\\25\\25\\16\end{pmatrix},\ C_{12}\equiv\begin{pmatrix}25\\2\\17\\6\end{pmatrix}.$$
$$C_{13}\equiv\begin{pmatrix}25\\4\\25\\9\end{pmatrix},\ C_{14}\equiv\begin{pmatrix}20\\8\\4\\1\end{pmatrix},\ C_{15}\equiv\begin{pmatrix}17\\17\\18\\16\end{pmatrix},\ C_{16}\equiv\begin{pmatrix}13\\4\\12\\21\end{pmatrix}.$$
$$C_{17}\equiv\begin{pmatrix}16\\3\\9\\4\end{pmatrix},\ C_{18}\equiv\begin{pmatrix}12\\23\\13\\0\end{pmatrix},\ C_{19}\equiv\begin{pmatrix}8\\4\\17\\15\end{pmatrix},\ C_{20}\equiv\begin{pmatrix}23\\0\\10\\12\end{pmatrix}.$$
$$C_{21}\equiv\begin{pmatrix}24\\17\\1\\24\end{pmatrix},\ C_{22}\equiv\begin{pmatrix}19\\16\\5\\12\end{pmatrix},\ C_{23}\equiv\begin{pmatrix}13\\4\\12\\21\end{pmatrix},\ C_{24}\equiv\begin{pmatrix}14\\12\\4\end{pmatrix}.$$

***Step 3**: The ciphertext becomes:*

NQXB BTWB DCJJ IJDT XDCF YFSG
LYGD MOXN LLGN HAPC QZZQ ZCRG
ZEZJ UIEB RRSQ NEMV QDJE MXNA
IERP XAKM YRBY TQFM NEMV OME

***Step 4**: Similarly, for decryption, first group the ciphertext into blocks, then substitute it into the decryption transformation:*

$$M_i \equiv A^{-1} + B \pmod{26}$$

***Step 5**: The plaintext after decryption is:*

PLEA SESE NDME THEB OOKM YCRE
DITC ARDN OISS IXON ETWO ONET
HREE EIGH TSIX ZERO ONES IXEI
GHTF OURN INES EVEN ZERO TWO

***Step 6**: The decryption verification matches the original plaintext.*

Decomposed Reasoning Step Generated by Llama-3.2-90B

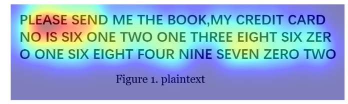
Step 1: The plaintext is "PLEASESENDME." Convert each letter to its numeric equivalent based on the rule $A=0, B=1, \ldots, Z=25$:
$P=15,\ L=11,\ E=4,\ A=0,\ S=18,\ E=4,\ S=18,\ E=4,\ N=13,\ D=3,\ M=12,\ E=4.$
The numeric representation of the plaintext is: $[15, 11, 4, 0, 18, 4, 18, 4, 13, 3, 12, 4]$.

Divide the plaintext into blocks of size 4 (since the matrix A is 4×4):

$$M_1=\begin{pmatrix}15\\11\\4\\0\end{pmatrix},\ M_2=\begin{pmatrix}18\\4\\18\\4\end{pmatrix},\ M_3=\begin{pmatrix}13\\3\\12\\4\end{pmatrix}.$$

LI: 0.56 *QW:* 0.11

QW Attention Map:



Step 2: The ciphertext is computed using the formula:
$C_i \equiv AM_i + B \pmod{26}$.

Block 1:

$M_1=\begin{pmatrix}15\\11\\4\\0\end{pmatrix}$. Compute: \(A M_1 = \begin{pmatrix} 3 13 21 9\ 15 10 6 25\ 10 17 4 8\ 1 23 7 2 \end{pmatrix} \begin{pmatrix} 15\ 11\ 4\ 0 \end{pmatrix}. \)

Step-by-step multiplication:
Row 1: $3(15)+13(11)+21(4)+9(0)=45+143+84+0=272,$
Row 2: $15(15)+10(11)+6(4)+25(0)=225+110+24+0=359,$
Row 3: $10(15)+17(11)+4(4)+8(0)=150+187+16+0=353,$
Row 4: $1(15)+23(11)+7(4)+2(0)=15+253+28+0=296.$

Now add $B=\begin{pmatrix}1\\21\\8\\17\end{pmatrix}$ to the result: $AM_1+B=\begin{pmatrix}272\\359\\353\\296\end{pmatrix}+\begin{pmatrix}1\\21\\8\\17\end{pmatrix}=\begin{pmatrix}273\\380\\361\\313\end{pmatrix}.$

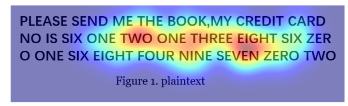
Take modulo 26: $C_1\equiv\begin{pmatrix}273 \bmod 26\\380 \bmod 26\\361 \bmod 26\\313 \bmod 26\end{pmatrix}=\begin{pmatrix}13\\16\\23\\1\end{pmatrix}.$

Block 2: $M_2=\begin{pmatrix}18\\4\\18\\4\end{pmatrix}$. Compute AM_2 and follow the same steps. After computations: $C_2\equiv\begin{pmatrix}10\\20\\3\\5\end{pmatrix}.$

Block 3: $M_3=\begin{pmatrix}13\\3\\12\\4\end{pmatrix}$. Compute AM_3 and follow the same steps. After computations: $C_3\equiv\begin{pmatrix}5\\22\\18\\9\end{pmatrix}.$

The full ciphertext is: $C=\begin{pmatrix}13\\16\\23\\1\end{pmatrix},\begin{pmatrix}10\\20\\3\\5\end{pmatrix},\begin{pmatrix}5\\22\\18\\9\end{pmatrix}.$

LI: 0.64 *QW:* 0.07



Step 3: The plaintext is recovered using: $M_i \equiv A^{-1}(C_i - B) \pmod{26}$.

Block 1: $C_1=\begin{pmatrix}13\\16\\23\\1\end{pmatrix}$. Subtract B: $C_1' = C_1 - B = \begin{pmatrix}13\\16\\23\\1\end{pmatrix}-\begin{pmatrix}1\\21\\8\\17\end{pmatrix}=\begin{pmatrix}12\\-5\\15\\-16\end{pmatrix}.$

Take modulo 26: $C_1'\equiv\begin{pmatrix}12\\21\\15\\10\end{pmatrix}.$

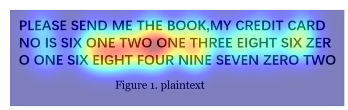
Now compute: $M_1 = A^{-1}C_1' \pmod{26}$.

Using A^{-1}: \(M_1 = \begin{pmatrix} 23 13 20 5\ 0 10 11 0\ 9 11 15 22\ 9 22 6 25 \end{pmatrix} \begin{pmatrix} 12\ 21\ 15\ 10 \end{pmatrix}. \)

After multiplication and modulo 26: $M_1=\begin{pmatrix}15\\11\\4\\0\end{pmatrix}.$

Repeat for C_2 and C_3. After decrypting all blocks, the plaintext is:
$M = \mathrm{PLEASESENDME}$.

LI: 0.78 *QW:* 0.15



Step 4: The encryption and decryption successfully verify the transformation, yielding the original plaintext: "PLEASESENDME."

LI: 0.8 *QW:* 0.1

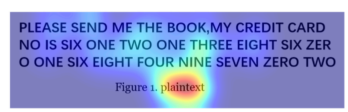
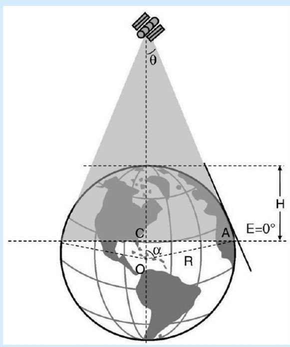


Figure 20. Illustration of the verification process obtained by performing problem-solving with Llama-3.2-90B while using Qwen-2-VL-72B (QW) and Llama-3.2-11B (Ll) as verifiers. The problem belongs to the ISE category.

Question Q

Question Q: Refer to the Figure. Determine the theoretical maximum area of the Earth's surface that would be in view from a geostationary satellite orbiting at a height of 35786 km from the surface of the Earth. Also determine the area in view for a minimum elevation angle of 10°. (Assume that the radius of the Earth is 6378 km.)

Image V



Ground Truth: Decomposed Reasoning Steps

Step 1*: Refer to the Figure. For the maximum possible coverage angle, the elevation angle* E *must be* 0°*. In that case, the coverage angle* α *is given by*
$\text{Coverage angle } \alpha = \sin^{-1}\left[\left(\frac{R}{R+H}\right)\cos E\right] = \sin^{-1}\left(\frac{R}{R+H}\right)$
where
R *= Earth's radius*
H *= Height of the satellite above the Earth's surface*
Thus,
$\alpha = \sin^{-1}\left(\frac{6378}{42164}\right) = 8.7^\circ$.

Step 2*: This gives*
$\theta = 90^\circ - \alpha - E = 90^\circ - 8.7^\circ = 81.3^\circ$.

In the right-angled triangle OAC, $\mathrm{OC} = \mathrm{OA} \times \sin 8.7^\circ$, *(Angle* $\mathrm{OAC} = 8.7^\circ$*) and* $\mathrm{OC} = 6378 \times 0.151 = 963.1\,\mathrm{km}$.

From the geometry, the covered surface area is given by

$2\pi R(6378 - 963.1) = 2\pi \times 6378 \times 5414.9 = 216997546.7\,\mathrm{km}^2$.

Step 3*: For* $E = 10^\circ$,
$\alpha = \sin^{-1}\left[\left(\frac{R}{R+H}\right)\cos E\right] = \sin^{-1}\left[\left(\frac{6378}{42164}\right)\cos 10^\circ\right] = 8.57^\circ$.

Step 4*: This gives*
$\theta = 90^\circ - 8.57^\circ - 10^\circ = 71.43^\circ$.
The new value of OC *is*
$6378 \sin 18.57^\circ = 2028.2\,\mathrm{km}$.

Covered area =
$2\pi R(6378 - 2028.2) = 2\pi \times 6378 \times 4349.8 = 174314563.3\,\mathrm{km}^2$
.

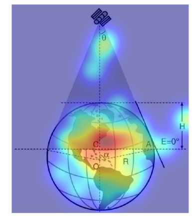
Decomposed Reasoning Step Generated by Llama-3.2-90B | ***Ll*** | ***QW:*** | ***QW Attention Map:***

Step 1: A geostationary satellite orbits the Earth at a height of $H = 35786\,\mathrm{km}$ above the surface of the Earth. The Earth's radius is $R = 6378\,\mathrm{km}$.

We need to calculate:
1. The **maximum theoretical area of the Earth's surface visible** from the satellite when the elevation angle is 0°.

2. The area visible for a **minimum elevation angle of 10°**.

Ll: **0.58** QW: **0.87**

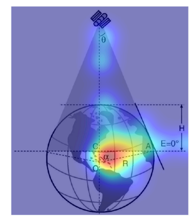


Step 2: From the geometry in the figure:
1. The satellite's orbital radius is:
$r_s = R + H = 6378 + 35786 = 42164\,\mathrm{km}$.

2. For any elevation angle E, the distance from the Earth's center to the edge of the visible area (point A) is given by: $d = R\cos\alpha$, where α is the half-angle of the cone formed by the visible region.

3. Using the law of cosines for the triangle OCA: $\cos\alpha = \frac{R}{r_s}$.

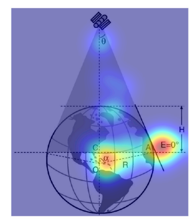
Ll: **0.66** QW: **0.24**



Step 3: For $E = 0^\circ$ (maximum theoretical visibility):
$\cos\alpha = \frac{R}{r_s} = \frac{6378}{42164}$.
$\cos\alpha \approx 0.1512$.
$\alpha = \cos^{-1}(0.1512) \approx 81.3^\circ$.

For $E = 10^\circ$:
From the geometry, the angle α is reduced by E:
$\cos(\alpha - E) = \frac{R}{r_s}$.
Let $\beta = \alpha - E$, so:
$\cos\beta = \frac{R}{r_s} = 0.1512$.
$\beta = \cos^{-1}(0.1512) \approx 81.3^\circ$.
Since $\alpha = \beta + 10^\circ$:
$\alpha \approx 81.3^\circ + 10^\circ = 91.3^\circ$.

Ll: **0.72** QW: **0.19**

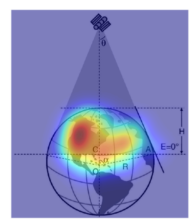


Step 4: The area visible is a spherical cap of the Earth. The formula for the surface area of a spherical cap is:
$A = 2\pi R^2(1 - \cos\alpha)$.

Case 1: Maximum Area ($E = 0^\circ$)
Substitute $\alpha = 81.3^\circ$:
$\cos\alpha = \cos(81.3^\circ) \approx 0.1512$.
$A = 2\pi(6378)^2(1 - 0.1512)$.
$A \approx 2\pi(40689684)(0.8488)$.
$A \approx 2\pi(34514632)$.
$A \approx 2.17 \times 10^8\,\mathrm{km}^2$.

Case 2: Minimum Elevation Angle ($E = 10^\circ$)
Substitute $\alpha = 91.3^\circ$:
$\cos\alpha = \cos(91.3^\circ) \approx 0.0017$.
$A = 2\pi(6378)^2(1 - 0.0017)$.
$A \approx 2\pi(40689684)(0.9983)$.
$A \approx 2\pi(40619680)$.
$A \approx 2.55 \times 10^8\,\mathrm{km}^2$.

Ll: **0.8** QW: **0.1**



Step 4: 1. The maximum theoretical area visible from the satellite is approximately:
$A_{\max} \approx 2.17 \times 10^8\,\mathrm{km}^2$.

2. The area visible for a minimum elevation angle of 10° is approximately:
$A_{10^\circ} \approx 2.55 \times 10^8\,\mathrm{km}^2$. ✗

Ll: **0.83** QW: **0.14**

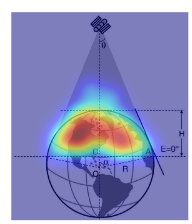


Figure 21. Illustration of the verification process obtained by performing problem-solving with Llama-3.2-90B while using Qwen-2-VL-72B (QW) and Llama-3.2-11B (Ll) as verifiers. The problem belongs to the SC category.