IT UNIVERSITY OF COPENHAGEN

Prompting Luck: Bias and Manipulability in LLM-Generated "Randomness"

MSc Thesis 2026

S26KISPECI1SE554

Amanda Bergqvist
amabe@itu.dk

Simone Nørgaard
snoe@itu.dk

Stephanie Nelander
stne@itu.dk

IT University of Copenhagen
MSc in Software Design

Copenhagen, June 2026

Prompting Luck: Bias and Manipulability in LLM-Generated "Randomness"

MSc Thesis 2026

S26KISPECI1SE554

Amanda Bergqvist
amabe@itu.dk
Student No. 23118

Simone Nørgaard
snoe@itu.dk
Student No. 17937

Stephanie Nelander
stne@itu.dk
Student No. 22018

Supervisor: Martin Hentschel
Associate Professor, IT University of Copenhagen

IT University of Copenhagen
MSc in Software Design

Copenhagen, June 2026

Abstract

Randomness is a fundamental concept studied across multiple disciplines, including computer science. An important area of research is the generation of high-quality randomness, which today is commonly performed by random number generators (RNGs). However, the emergence of large language models (LLMs) raises a new question: Can LLMs generate randomness? This thesis investigates how prompt context and prompting strategies affect the output of an LLM when asked to generate randomness and the implications of using an LLM in place of an RNG.

To evaluate these considerations and effects, we introduce and open-source an experimental framework. The findings show that randomness generated by LLMs exhibits strong anchoring bias toward contextual cues, and that interaction mode, including the presence or absence of shared history, produces significant differences across all examined metrics. Furthermore, prompting strategies and prompt context significantly influence value distributions, emergent patterns, and overall system behavior.

Keywords: Randomness Simulation, Pseudo-Randomness, Large Language Models, Number Generation, Prompting Techniques, Prompting Strategies

AI Acknowledgment

This thesis uses generative AI both as part of the experimental methodology and as an assisting tool during development and writing.

All experimental data were generated using locally hosted LLMs executed through Ollama (https://ollama.com). The use of generative AI models as part of the methodology is further described in Chapter 5: Experimental Setup of the thesis.

In addition, generative AI was used as an assisting tool during the development and writing process. The Grammarly Chrome extension (https://www.grammarly.com/browser/chrome) was used to improve spelling, grammar, and sentence structure. ChatGPT (https://chatgpt.com), Claude (https://claude.ai), and Microsoft Copilot (https://copilot.microsoft.com) were used for coding assistance, including debugging support, guidance on programming practices, documentation formatting, LaTeX table formatting, and assistance with visualization code.

All AI-generated output was reviewed and validated before inclusion in the thesis, and everything not specified here was conducted and written independently by the authors.

Contents

List of Figures

List of Tables

This page intentionally left blank.

1 Introduction

Randomness is a fundamental concept studied across multiple disciplines, including mathematics, physics, and computer science. In computer science, an important area of research is the generation of high-quality randomness. It is commonly known that algorithmic random number generators (RNGs) do not produce true randomness (Stipčević et al., 2014), though some of them can simulate it, which is called pseudo-randomness.

Today, RNGs are the main method used to generate randomness in many types of software. However, the emergence and popularization of generative artificial intelligence (AI), particularly large language models (LLMs), raises a new question: Can LLMs generate randomness of the same quality? This question becomes especially important as LLMs are increasingly integrated into software, replacing existing system components that may, in some cases, include random generation as part of their functionality. Research has already shown that LLMs, even well-known models such as OpenAI's GPTs, Meta's Llama, and Google's Gemini, provide "random" numbers that are rather deterministic when prompted with minimal prompts. Some research indicates that specific numbers, often of cultural significance or that humans consider to be more random, are largely overrepresented, and suggests that LLM-generated "randomness" is highly influenced by training data, prompting language and the way the model was trained (Coronado-Blázquez, 2025). However, if and how the context of the prompt itself and the prompting techniques affect the "randomness" that LLMs generate remains unexplored. This thesis addresses the following overarching questions: Does the prompt affect the output of LLMs when they are tasked with generating randomness? And if so, what implications could it have if RNG-generated data is replaced by LLM outputs within software systems?

To examine these questions, we replace the RNG component in a simple one-armed bandit (slot machine) with OpenAI's open-weight LLM gpt-oss:20b. This allows us to test prompts with varying contexts, personas, and game modes, resulting in 32 experiments of controlled prompt variation, including an RNG baseline experiment for comparison. The outputs are analyzed to evaluate how the prompt affects the LLM-generated randomness, as well as how this affects the system outcomes; in this context,

how many winning spins are produced.

This thesis provides an experimental framework for testing the effect of prompts on LLM-generated randomness and demonstrates the following findings:

1. Strong anchoring bias for related cues: when a single "lucky number" (10) was introduced, it accounted for 15.9% and 15.3% of all generated values in the two game modes, respectively. In comparable experimental setups without lucky numbers, the corresponding averages were only 1.7% and 8.8%, as well as 10.3% for the RNG baseline.
2. No anchoring bias for unrelated cues: Unrelated numerical cues (e.g., "10 o'clock") did not show significant anchoring bias (1.6% and 9.4%, respectively).
3. Prompting strategies significantly affect win distributions: isolated-spin experiments produced more wins for personas using status quo bias and jailbreaking with 140 and 312 wins, respectively, compared to near-zero wins for other experiments.
4. Strong context effects: the full-game experiment, where credits are explicitly explained, showed a strong prevalence of wins (1029 wins) compared to other experiments with 397-698 wins, and 11 wins for the experiment without context.
5. Significant differences between interaction modes: Interaction mode (i.e., game mode) makes a big statistical difference for LLM-generated randomness according to metrics, such as the runs test, sequence lengths, patterns, and spin frequency.

2 Background

This chapter presents the key concepts and technologies relevant to our work. First, the concept of randomness is introduced, followed by a brief description of slot machines and one-armed bandits, as they play a key role in the experimental setup. Then follows an introduction to artificial intelligence (AI) and large language models (LLMs). Finally, selected human cognitive and behavioral phenomena relevant to this thesis are introduced. Together, all of this serves as a conceptual and technical foundation for the remaining thesis.

2.1 Randomness

Randomness is a concept that is defined differently across scientific disciplines, depending on its theoretical and practical context (Bikos et al., 2023). In probability theory, randomness is often described as a property of events whose outcomes cannot be predicted with certainty and need not be equally likely (Anderson et al., 2018). From a statistical perspective, the uniform distribution, i.e., each outcome is equally likely, is often used as a reference model for ideal randomness (Bikos et al., 2023). When people refer to "an element chosen at random", they also often implicitly assume uniformity (Anderson et al., 2018). The term stochasticity is often used interchangeably with randomness, referring to the inherent unpredictability of a process or a system (Tumeo, 1994).

In computational systems, random values are often generated using pseudo-random number generators (RNGs), which typically produce values following a uniform distribution as a standard. As the term "pseudo" suggests, the generated values are not truly random, but are produced by algorithms designed to mimic random behavior. The quality of RNGs can be measured using a set of statistical tests that detect patterns and deviations from uniformity (Bikos et al., 2023).

In contrast to RNGs, which produce randomness using an algorithm, LLMs create responses, including random numbers, by sampling from a probability distribution over possible next tokens shaped by the relationships learned during training (Stryker,

2026; Bergmann, 2026). How the output differs from conventional RNGs, and what implications it may have, is explored further in this thesis.

2.2 Slot Machines and One-Armed Bandits

One of the many applications of randomness is slot machines, which is used as the main context in this thesis. Slot machines let players place a stake and then spin one or multiple rows of reels with different symbols. Some symbol combinations will return a payout, e.g., the stake multiplied by a certain number, and these symbol combinations are called pay lines or win lines. One of the simplest kinds of slot machines is the one-armed bandit, which, before digitization, typically consisted of a horizontal row with three mechanical reels and a single lever, which started the machine. Today, there are many types of slot machines, most of which are implemented using RNGs, which allow them to be much more complex with multiline reels, new features, and many more win lines.

Slot machines are among the most harmful forms of gambling (Alkurdi et al., 2025), so to protect players and ensure fair play, they are highly regulated. In Denmark, this includes a requirement that the Return-to-Player percentage (%RTP) must be at least 74%, meaning that, over the long run, at least 74% of the total stake is returned to players through payouts. Since slot machines must be random, the %RTP is achieved through the probability of each symbol and the differing multipliers for win lines (Gambling Commission, 2024). Today, many slot machines have an %RTP of 94-98% (Daugherty, 2026).

The slot machine implemented for this thesis is a very simple one-armed bandit with three reels with 10 numbers each. It is important to note that this machine has a win frequency of 1% and an expected %RTP of only 3%. Hence, it is not a realistic machine in terms of payouts or win frequency, but this simple setup suffices to research how the prompt affects the randomness that the LLM outputs.

2.3 Artificial Intelligence and Large Language Models

2.3.1 Generative Artificial Intelligence

AI can be defined as ”technology that enables computers and machines to simulate human learning, comprehension, problem solving, decision making, creativity, and autonomy” (Stryker et al., 2026). A subfield within the AI domain that has become pervasive in academia, as well as industry and society in general, is ”generative artificial intelligence” (GAI); AI capable of generating original content spanning diverse forms, often referred to as ”modalities”, including text, images, videos, audio, or even multimodal content, in response to instructions or users’ requests. What differentiates GAI models from other types of AI is their content-generation capabilities. In simple terms, GAI models encode a simplified representation of their training data, enabling

them to draw from that representation to generate new data that resembles, but does not directly replicate, the training data (Zhang et al., 2025; Stryker et al., 2026).

2.3.2 What LLMs Are

Text generation involves processing and producing human language text. With the advancements of deep learning (DL), (artificial) neural network models have emerged as the dominant method for generative models, including those for text generation (Zhang et al., 2025). Thus, most modern text-generating models, which are typically LLMs, are based on deep neural networks; multilayered neural networks, containing an input layer, multiple (typically hundreds of) hidden layers, and an output layer. Each of these layers contains artificial neurons performing mathematical operations, which enables them to learn complex patterns in data and generate human-like text (Stryker et al., 2026).

A pivotal emergence in the evolution of LLMs is what a group of Google researchers in 2017 proposed as a new neural network architecture called the transformer in a landmark paper on machine learning (ML) titled "Attention Is All You Need" (Vaswani et al., 2017). What characterizes the transformer architecture is its reliance on a so-called "self-attention" mechanism, which allows models based on this architecture to process and understand complex textual information more efficiently and at a larger scale through parallelization, and with greater accuracy due to improved handling of long-range contextual dependencies, compared with previous alternatives (typically complex recurrent or convolutional neural networks) (Vaswani et al., 2017).

2.3.3 How LLMs Work

At a high level, LLMs are pretrained and typically fine-tuned before they are able to generate responses.

Pretraining Before the actual pretraining starts, vast amounts of diverse text data from various sources, e.g., books, articles, websites, code, etc., are collected into a text corpus. The corpus can be pre-processed by humans, potentially removing incorrect, duplicated, or undesired content from the training data. Next, the text is tokenized, i.e., broken into smaller units called tokens, such as words, subwords, or characters (Stryker, 2026). Subsequently, each token is mapped to a vector of numbers, called an embedding, capturing meaningful patterns, relationships, and semantics, enabling models to understand and generalize better (Barnard, 2025). LLMs are initially trained using self-supervised learning, where no ground truth is provided to the model, but it has to infer this itself from the data. The training phase results in a neural network with billions of parameters that encode statistical representations of language, enabling the model to generate new content autonomously in response to user prompts (Stryker et al., 2026). These parameters include learned weights and biases, as well as hyperparameters, the latter of which are external to the model (Belcic et al., 2025) and are

described further in Section 2.3.4. The outcome of the pretraining is a so-called foundation model, having typically a broad general knowledge, which can be trained further in the form of fine-tuning to specialize for certain downstream tasks or domain-specific knowledge (Stryker, 2026).

Fine-Tuning After pretraining, LLMs can be fine-tuned in various ways with different objectives. There are many methods, including supervised learning and reinforcement learning from human feedback (RLHF). In supervised learning, the model is provided with a smaller dataset containing ground truths. Here, the objective could be to specialize a model for certain downstream tasks (e.g., summarization, translation, or classification), or domain-specific knowledge (e.g., within healthcare or law). In RLHF, on the other hand, humans rank model outputs[1] and these human preferences are used to optimize the model (Stryker, 2026). While it is worth mentioning that many other fine-tuning methods exist, these will not be described further due to the scope of the project and the fact that we are not tuning any models, but use them as they are.

Generation Once LLMs have been pretrained and potentially fine-tuned, they can be used to generate text in response to user prompts. Just like the training data, the input text is tokenized and converted into embeddings; tokens with similar meanings end up mathematically close. LLMs then generate their text output by iteratively inferring what token should come next, based on the probabilities for all potential next tokens. In this way, LLMs apply patterns learned from their training data to generate an output for a given input, one token at a time, until the generated output is complete. After generation, an evaluation of the output, and possible (additional) tuning of the model, may take place (Stryker, 2026; Bergmann, 2026).

2.3.4 LLM Hyperparameters

Hyperparameters are external to LLMs, i.e., settings that can be changed by humans, and different types of these affect different characteristics of LLMs, such as their architecture, training, inference, resource use, or output (Belcic et al., 2025). In the following, we narrow the focus to those mentioned elsewhere in this thesis, namely the memory and compute-related hyperparameter "context window" and the inference-related hyperparameter "temperature".

Context Window This hyperparameter determines the maximum number of tokens that an LLM can consider at one time when making predictions and generating a text output, which also affects the length of the conversation that a model can maintain

[1] Examples of RLHF include when users are provided with the option of voting LLM-generated responses with, e.g., thumbs up or down (as seen for, e.g., Microsoft Copilot and Perplexity), or different output variants and the question of which variant they prefer (as seen for, e.g., OpenAI's ChatGPT).

without losing track of previous content. Larger context windows have the advantages of greater accuracy, fewer hallucinations (plausible but false or nonsensical statements), and the capability to process larger amounts of input data and interact in longer conversations. However, this comes at the cost of greater computational resource requirements and can extend the model's processing time during text generation (Belcic et al., 2025). The context window is associated with what is sometimes considered shared or conversational history, which is used in this thesis.

Temperature This hyperparameter controls the randomness of next-token sampling during inference, influencing how deterministic or "creative" an LLM's generated text output is. Higher temperatures flatten the probability distribution for the next token, resulting in more diverse and unpredictable variations in phrasing. In contrast, lower temperatures make the probability distribution more peaked, resulting in more deterministic text outputs. A typical range of temperature values is [0.0, 2.0], where 1.0 represents the standard probability distribution and often is the default temperature[2] (Belcic et al., 2025; Noble, 2026).

2.3.5 Reasoning Models

As a subset of LLMs, reasoning LLMs have emerged. These models are typically fine-tuned or trained to break complex tasks, such as logical, mathematical, or coding problems, into smaller intermediate steps, often referred to as "reasoning traces" or "chains-of-thought" (CoT), before generating a final output. Reasoning LLMs differ in how transparent they are; while some provide users with their reasoning traces, others show only a summary of them, or even hide them entirely. Although reasoning can improve performance on complex tasks, it may degrade performance in other cases. Moreover, reasoning introduces additional computational cost and latency, making it advantageous primarily for tasks that benefit from deeper deliberative processing (Bergmann, 2025).

2.3.6 Prompting and Prompt Engineering

The primary purpose of a prompt is to provide the LLM with instructions and context to achieve a desired task, but prompts can be composed in many different ways. Using appropriate prompts can enhance natural language understanding for LLMs, ultimately improving the quality of their generated outputs (Marvin et al., 2024). Prompt engineering is an emerging field, and although the terminology within literature on prompting is rapidly developing, the vocabulary remains inconsistent (Schulhoff et al., 2025). However, prompt engineering can be defined as the practice, or iterative process, of developing and optimizing prompts to efficiently use LLMs (Marvin et al., 2024). While a wide range of techniques and best practices exist, Table 2.1 summarizes

[2] Some models have other temperature ranges, e.g., [0.0, 1.0], as is the case for some of Anthropic's models, which still have a default temperature value of 1.0 (Anthropic, 2024; Anthropic, 2026).

Table 2.1: *Prompting techniques applied during initial and/or final experiments.*

Technique	Description
Role Prompting/ Playing	Prompting an LLM to undertake a specific role, whether in the form of a human character or non-human entity (Kong et al., 2024).
Emotional Prompting	Prompting an LLM using a prompt that includes a psychological emotional stimulus (Li et al., 2023) or emotional tone (Bernays et al., 2026).
Jailbreak Prompting/ Jailbreaking	Prompting an LLM to bypass its safeguards and elicit harmful, dangerous, or policy-violating outputs, often through a combination of techniques (Shen et al., 2024).
Self-Awareness Prompting	Prompting an LLM to be mindful not to be biased by cognitive biases (Echterhoff et al., 2024).
Zero-Shot Prompting	Prompting an LLM to perform a task without providing it with any task-specific examples, thus, forcing it to rely entirely on its pre-trained knowledge (Syed et al., 2025).
One-/Few-Shot Prompting	Prompting an LLM to perform a task, providing it with one/more task-specific example(s), thus guiding it towards more appropriate outputs (Syed et al., 2025).
Chain-of-Thought (CoT) Prompting	Prompting an LLM to break down a complex task into sequential reasoning steps, guiding the model through a step-by-step reasoning process (Gadesha et al., 2025).

those that have been experimented with during the initial and final experiments. Some of these are well-established and broadly recognized techniques shown to improve the quality of model outputs, but others are more experimental and have been suggested and explored by fewer researchers.

2.3.7 Applications, Capabilities, and Limitations of LLMs

LLMs can adapt to various applications involving diverse text-based tasks, such as text generation, interpretation, summarization, language translation, sentiment analysis, code debugging and generation, question-answering, etc. (Stryker, 2026; Zhang et al., 2025). Although LLMs, in some cases, can perform tasks more efficiently than humans (e.g., in the form of chatbots for customer service), they also have several limitations and lead to significant implications. These include, but are not limited to, so-called hallucinations, where models generate plausible-sounding but factually inaccurate outputs, or outputs that are entirely nonsensical (IBM, 2026). Another challenge is different types of biases in LLM outputs; models may not only replicate but even amplify biases present in the training data or stemming from other aspects of their existence, possibly leading to unfair, inappropriate, or incorrect outputs (Holdsworth, 2025). Beyond such challenges that may directly affect humans, the use of LLMs has also raised environmental concerns due to the substantial energy consumption and computational resources required to train and deploy such models (Stryker, 2026).

Table 2.2: *Model architecture overview of gpt-oss.*

Model	Total Params	Active Params Per Token	Total Experts	Active Experts Per Token	Context Length (Tokens)
gpt-oss:120b	116.83B	5.13B	128	4	131,072
gpt-oss:20b	20.91B	3.61B	32	4	131,072

Source: OpenAI (2025) and OpenAI et al. (2025).

2.3.8 OpenAI's gpt-oss

The primary LLM investigated and used for all final experiments is OpenAI's gpt-oss:20b; an open-weight reasoning LLM released on August 5, 2025, along with a larger 120B parameter variant. The models are available under the permissive Apache 2.0 license, and are designed to be flexible and easy to run anywhere, whether locally, on-device, or through third-party inference providers. Their weights (trained parameters) are publicly available and free to download from Hugging Face, enabling local deployment and fine-tuning of the models (OpenAI, 2025). Both models are designed for use in agentic workflows, with strong instruction-following ability, tool use (e.g., web search and Python code execution), and CoT reasoning capabilities. The models are based on the transformer architecture and use a mixture-of-experts (MoE) design, where a routing mechanism selects only a small subset of specialized neural network components ("experts") to process each token. In this way, only a fraction of the model's total parameters are active at a time, making inference more computationally efficient and reducing active memory usage compared with activating all parameters for every token (OpenAI et al., 2025). Table 2.2 summarizes selected characteristics of the two models. The models are pretrained on a text-only dataset containing trillions of tokens with a knowledge cutoff of June 2024. The dataset focuses on STEM (science, technology, engineering, and mathematics), coding, and general knowledge, and is filtered for harmful content (OpenAI et al., 2025).

2.4 Human Cognition and Behavior

While human cognitive and behavioral phenomena have been studied for many years within psychology and the interdisciplinary field of cognitive science, inconsistencies exist in related terminology, including the notions of "heuristics" and "cognitive bias" (Nadurak, 2025). A canonical distinction is provided by Tversky et al. (1974), who describe heuristics as a limited number of principles which people rely on to reduce complex tasks, such as assessing probabilities and predicting values. Such judgmental heuristics can be useful in some cases, but can lead to severe and systematic errors, i.e., cognitive biases, in others (Tversky et al., 1974). Other researchers, however, do not make such a distinction but use bias in a broader sense, covering both of these terms.

Although LLMs do not possess cognition in the human sense, prior research has shown that they can display patterns similar to human heuristics and cognitive biases,

as will be described further in Chapter 3. In this thesis, these phenomena are investigated in the context of prompting, where prompts are designed to examine whether LLMs exhibit behaviors analogous to human cognitive and behavioral tendencies in their generated outputs.

Two human cognitive and behavioral phenomena investigated during both the initial and final experiments are introduced below: anchoring and status quo bias. Appendix A further introduces some additional phenomena explored only during the initial experiments. A common factor of all these is that they, in one way or another, concern human judgment and decision-making.

Anchoring Anchoring, as originally described by Tversky et al. (1974), refers to a cognitive phenomenon (in this case, a heuristic) in which people rely too heavily on an initial piece of information, a so-called "anchor", when making subsequent judgments or estimates. The initial value, or starting point, may be suggested by the problem/question or an intermediate calculation, but may also be altogether arbitrary and irrelevant to the problem. Irrespective of the origin of such an initial value, anchoring refers to the tendency to be influenced by anchors, leading to biased answers.

Status Quo Bias Status quo bias refers to a cognitive phenomenon in which people disproportionally uphold the status quo, i.e., do nothing or maintain their current or previous decision, when faced with alternatives. This bias may be seen as a consequence of 1) rational decision-making (e.g., in case of transitions costs, uncertainty, or facing similar decision settings as previous), 2) cognitive misperception (e.g., risk aversion, as further described below, or anchoring), or 3) psychological commitment stemming from misperceived sunk costs (past expenses that cannot be recovered) or other resource investments, regret avoidance, or a drive for consistency (Samuelson et al., 1988).

3 Related Work

This chapter reviews the current state of research on the quality of LLM-generated randomness and how different prompting techniques and strategies affect model outputs across existing studies.

3.1 LLM-Generated Randomness

As LLMs are increasingly applied across diverse application areas, many of which inherently require randomness, the quality of randomness becomes an important consideration in settings such as games, scheduling, and stochastic decision-making tasks Karanjai et al. (2025). This section reviews prior studies on the quality of LLM-generated randomness and is organized into two subsections, focusing on differences in tasks and findings, and observed differences in interaction mode.

3.1.1 Studies on LLM Randomness

Several studies explore the quality of randomness in LLM-generated outputs across different tasks, models, and evaluation setups (see table 3.1).

Table 3.1: *Overview of studies on LLM-generated randomness.*

Study	Tasks	Metrics	LLM Models
Evaluating the Quality of Randomness and Entropy in Tasks Supported by Large Language Models (Karanjai et al., 2025)	Number generation (range 0-255), character-based password generation, and deck-of-cards shuffling	NIST randomness test suite, shuffling-specific entropy measure, qualitative analysis of sequences	GPT-4o, Gemini 1.5 Pro, Phi-3, Gemma 2 27B, Llama 3.1 8B, Mistral Large(2047)

Continued on next page.

Table 3.1: *Continued from previous page.*

Study	Tasks	Metrics	LLM Models
Deterministic or probabilistic? The psychology of LLMs as random number generators (Coronado-Blázquez, 2025)	Number generation across ranges (0-5, 0-1, 0-100), languages (en, es, fr, hi, ja, ru, zh), and temperature settings (0.1, 0.3, 0.5, 1.0, 2.0)	χ^2, Cramér's V, self-defined randomness index	DeepSeek-R1 14B, Gemini 2.0, GPT-4o-mini, Llama 3.1 8B, Mistral 7B, Phi-4 14B
Emergent Numeric Bias in Large Language Models: An Empirical Study on the Anomalous Recurrence of the Number 27 Across Independent Sessions (Nath, 2025)	Number generation (range 0-50)	Frequency of number 27	ChatGPT-4o, Claude 3, Gemini Flash and Pro, Meta's AI Assistant
Reasoning Under Uncertainty: Exploring Probabilistic Reasoning Capabilities of LLMs (Pournemat et al., 2025)	Most probable outcome identification, distribution estimation, and sampling from joint/conditional distributions	Accuracy, Total variation distance (TVD)	Llama3.1-8B, Llama3.3-70B, Qwen2.5-7B, DeepSeek-R1-Distill-Qwen-7B, GPT-4o-mini, GPT-4.1-mini
Do LLMs Play Dice? Exploring Probability Distribution Sampling in Large Language Models for Behavioral Simulation (Gu et al., 2024)	Understanding probability distributions, sampling from explicit/implicit probability distributions	Positive sample ratio, Kolmogorov–Smirnov test	GPT-4, GPT-3.5, Claude 2.1, Llama2, Vicuna, ERNIEBot, ChatGLM, Gemini, Mixtral

Karanjai et al. (2025) evaluates numerical generation (range 0-255), character-based password generation tasks, and deck-of-cards shuffling. They use the NIST test suite for the numerical and character-based tasks, while the shuffling task is evaluated using an entropy-based measure. Overall, they find that LLMs can mimic randomness to some extent, but struggle to produce high-quality cryptographic randomness. They also find that temperature has a limited effect on performance.

In contrast to Karanjai et al. (2025), Coronado-Blázquez (2025) finds substantially stronger deterministic patterns in numerical generation across different ranges, prompting languages, and temperature settings. They find consistent value preferences and skewed distributions towards larger values, with limited impact of temperature. Their

results also suggest that language affects distributional behavior, although the output still preserves an underlying tendency towards specific "fingerprints" (preferred values). For the range 1-5, the most frequently observed values are 3 and 4; for 1-10, it is 7 for all LLMs, except DeepSeek–R1 when prompted in Chinese, for which it is 5; and for 1-100, the most frequently generated values are 37, 47, and 73. The authors suggest that these value preferences, "fingerprints", may be related to human cognitive biases, as previous research has shown that humans are more likely to select culturally meaningful numbers such as 7 or 42, prime numbers, or numbers perceived as being near the middle of a range.

Nath (2025) further highlights the influence of interaction mode on such biases, showing different prevalence of the number 27 under different prompting setups. Their results indicate significant differences in the frequency of the value across interaction modes.

Rather than focusing solely on explicit random-generation tasks, a second group of studies explores probabilistic reasoning. Pournemat et al. (2025) evaluate distribution estimation and sampling from both joint and conditional probability distributions. They find that performance declines as task difficulty increases, but that larger models achieve strong distributional alignment and can, in some cases, outperform a Python-based random sampler. Similarly, Gu et al. (2024) investigates LLMs' ability to identify and sample from probability distributions. While the results reveal that models generally recognize distributions well, they struggle to sample accurately from them, particularly when distributions must be inferred implicitly from the context.

These studies all examine the quality of randomness in model outputs and the frequency of specific values, interpreting observed biases as a consequence of underlying model behavior shaped by training data and next-token prediction. While some studies consider variations in label representation or prompting language, none explicitly investigate how prompt formulation or contextual variation influences output distributions. Instead, they rely on simple, fixed prompts without systematically varying contextual information. This differs from other use cases of LLMs, where outputs are more elaborate, include more contextual information, and/or may be biased. This motivates this thesis, which investigates the effect of prompting and contextual variations on LLM-generated randomness.

3.1.2 Interaction Modes and Their Influence on Randomness

The reviewed studies all show that LLM-generated randomness overall deviates from true or pseudo-randomness, though the extent of these deviations varies between studies. Additionally, they differ in experimental setup, in particular, interaction mode, which appears to affect the results significantly.

Nath (2025) explicitly compares two interaction modes: a single-sample interaction setup with no shared history and a chat-based interaction setup with conversational history. Their findings show that the single-sample interaction setup, where

each prompt is a first-interaction prompt, produces an extremely high frequency of the number 27 when generating numbers from 1 to 50, i.e., at least 87%. In contrast, the chat-based interaction setup, where only the opening prompt is a first-interaction prompt, has a frequency of at most 4% for the number 27. Therefore, they suggest that randomness is influenced by first-interaction prompts, as these behave more deterministically.

The other introduced studies differ from one another in interaction mode, but most of them seem unaware of the effect this can have on their results. Coronado-Blázquez (2025), which also uses a single-sample interaction setup (where all prompts are first-interaction prompts), similarly reports that LLMs are highly deterministic. In contrast, lower levels of determinism are observed in the studies by Karanjai et al. (2025) and Pournemat et al. (2025), which both use a multi-sample interaction setup (i.e., where all samples are generated in a single LLM response), which differs from the interaction modes used in the two other studies. According to Karanjai et al. (2025), LLMs can mimic randomness to some extent, but struggle to produce randomness of cryptographic quality. Similarly, Pournemat et al. (2025) found that some LLMs can accurately sample from probability distributions, with two of the examined models even outperforming a Python random sampler when generating samples from a joint distribution. Gu et al. (2024) evaluates a single-sample interaction setup with no shared history against a multi-sample interaction setup and finds that performance improves slightly when using the multi-sample interaction setup; however, this comparison is only conducted as part of a limited initial experiment.

Overall, the different results reported suggest that interaction mode might have a substantial influence on LLM-generated randomness and may explain differences in reported randomness behavior across studies.

3.2 The Effect of Prompting

This section reviews related work on how prompt engineering influences outputs generated by LLMs. It introduces the concept of role prompting, which is an essential part of the experimental design in this thesis, as well as research on cognitive phenomena replicated by LLMs, emotional prompting, and jailbreaking strategies.

3.2.1 Role Prompting

Role prompting, also known as role-play prompting or simply role-playing, assigns a specific role to the LLM as a way to improve the accuracy or manipulate its outputs. It is a technique that is often used in research, covering a wide range of human roles such as teachers, deceased grandmas, police officers, and travel writers (Kong et al., 2024; Wang et al., 2025; Schulhoff et al., 2025) as well as object roles such as coins or recorders (Kong et al., 2024). Furthermore, it can be used alongside other prompting techniques, such as jailbreaking (Wang et al., 2025) and chain-of-thought (CoT) rea-

soning (Kong et al., 2024). Kong et al. (2024) evaluate the performance of role prompting across four reasoning areas, encompassing 12 datasets in total, covering arithmetic, common-sense reasoning, symbolic reasoning, date understanding, and tracking shuffled objects. They find that zero-shot role prompting outperforms standard zero-shot prompting in 10 out of 12 datasets. For zero-shot-CoT (with "Let's think step by step" added to the prompt), zero-shot role prompting outperformed it in 9 out of 10 datasets. Finally, when compared with few-shot-CoT results from previous research, the performance is almost equivalent. Overall, their findings suggest that role prompting can improve the reasoning capabilities of LLMs and may even be comparable to other proven techniques, such as few-shot-CoT.

3.2.2 Cognitive Phenomena Replicated by LLMs

Even state-of-the-art LLMs have been proven to generate outputs with societal biases that resemble those of human cognitive biases (Echterhoff et al., 2024; Lou et al., 2025). These include anchoring bias and status quo bias, which were introduced in Section 2.4.

Lou et al. (2025) examine how different types of numerical anchors introduced in prompts affect LLM-generated textual answers to 62 questions from different domains. Each question includes zero or more anchoring hints, i.e., potential anchors, to assess how they affect responses. The questions are tested on several model families, including ChatGPT, DeepSeek, Gemini, and Claude, some of which are reasoning models, all set to a temperature of 0.8. Additionally, the study examines mitigation strategies such as CoT, both-anchor (several examples with anchors), and explicitly instructing the model to ignore anchors. The findings show that anchoring bias is widely prevalent in LLMs. Models with stronger reasoning capabilities, e.g., GPT-4 and GPT-4o, are less susceptible to anchoring bias than weaker models. However, they still show anchoring bias, especially for what the authors call "expert anchors", and the bias that they do show is consistent across outputs. In contrast, the bias seen in outputs of weaker models fluctuates more in terms of standard deviation due to randomness and hallucinations. In addition, the study shows that irrelevant anchoring hints do not have any significant influence on the output. Finally, most of the tested mitigation strategies do not appear to effectively reduce anchoring bias, despite showing promising effects in many other reasoning tasks. However, when providing several examples with anchors in opposing directions, the LLMs perform similarly to the benchmark without anchors, successfully mitigating the bias.

Echterhoff et al. (2024) also examines mitigation strategies and biases in LLM outputs, including anchoring and status quo bias. This is done using a high-stakes decision-making scenario, i.e., student admissions for a college program. They present three overarching classifications of human-like cognitive biases seen in LLMs: prompt-induced, inherent (i.e., from the training data), or sequential (i.e., induced by previous model answers). Moreover, they classify status quo bias as prompt-induced and anchoring bias as sequential. Similar to Lou et al. (2025), they find that the examined LLMs

show anchoring bias. In contrast, for status quo bias, they show that when LLMs are presented with several options and the status quo, the models avoid using the status quo option. Additionally, the study examines three mitigation techniques: zero-shot-mitigation (i.e., asking the LLM "Be mindful to not be biased by cognitive bias."), few-shot-mitigation (i.e., providing examples of correct or incorrect behavior), and self-help (i.e., asking the LLM to rewrite the prompt such that no biases are present). They find a small improvement when using zero-shot-mitigation. For few-shot-mitigation, they find that it can lead to extreme results and a reduced ability to follow instructions. Finally, they find that the self-help technique is highly successful in mitigating biases for high-capacity models.

In summary, both studies find anchoring bias across the examined LLMs. However, status quo bias has the opposite effect than what is seen for humans. They both find that zero-shot-mitigation is generally less effective, but suggest that LLM self-help or both-anchoring with contrasting anchors may be the most effective mitigation strategies. This thesis explores anchoring and status quo bias further, both of them as prompt-induced as well as sequential biases. Additionally, it evaluates zero-shot-mitigation and both-anchors. This, within the scope of LLM-generated randomness.

3.2.3 Other Prompting Techniques

Other prompting techniques that can affect the quality of LLM outputs include emotion(al) prompting and jailbreaking.

Emotion prompting, in which prompts with emotions are used in place of neutral ones, can lead to improved LLM performance (Bernays et al., 2026; Schulhoff et al., 2025). Bernays et al. (2026) find that ChatGPT-4o, when using the default temperature of 1, improves its answers the most when prompted to feel proud of its answers. A smaller effect is observed for anger prompts, while no significant difference is found between the neutral and blaming prompts.

Finally, there exist jailbreaking techniques in which the models are manipulated through prompting to bypass safety constraints. In the studies by Shen et al. (2024) and Wang et al. (2025), the authors examine how different prompting strategies can convince models to generate responses that would otherwise be against their ethical guidelines and safety constraints. Both studies demonstrate that LLMs remain susceptible to jailbreaking strategies for both open-source and commercial models, such as GPT-4, DeepSeek-R1-7B, and Gemini-2.0-flash. Shen et al. (2024) find five specific jailbreaking prompts with a success rate of 95%, while Wang et al. (2025) found that the framework they tested, which is based on role prompting, has average success rates of 84.3% for commercial models and 87.3% for open-source models.

The effect of emotion prompting and jailbreaking on LLM-generated randomness is explored further in this thesis.

3.3 Research Gap and Motivation

In contrast to previous work studying LLM-generated randomness and inspired by human cognitive biases and the effect of prompting, this study explicitly incorporates prompting as a controlled variable to investigate whether prompt variations, such as role-based descriptions, potential anchors, or jailbreaking, influence stochastic output behavior. This allows us to capture the effects of contextual prompting on LLM-generated randomness, which, to our knowledge, have not been addressed in prior work, and to better characterize stochastic behavior under more realistic usage conditions.

This thesis focuses on the generation of numerical outputs and aims not only to investigate shifts in output distribution under changing prompting conditions, but also to examine whether patterns arise and how such potential patterns are influenced by prompting. Additionally, it examines how potential shifts in output distribution and patterns may affect outcomes when LLMs are integrated within a larger system and tasked with generating randomness.

Finally, prior studies suggest differences between interaction modes; therefore, we compare a single-sample interaction setup, where each response is generated in a separate LLM call with no shared history (hereafter referred to as isolated spins), and a multi-sample interaction setup, where all outputs are generated in a single response (hereafter referred to as full games).

4
Terminology

This chapter presents the primary terminology and definitions used throughout the thesis. While several of the terms are commonly used in other contexts, the following definitions are specific to the experimental setup developed for this thesis and may therefore deviate from more widely established usage.

One-Armed Bandit A one-armed bandit serves as the analytical lens for exploring randomness in this thesis and is the role given to the LLM for most experiments (see separate role description).

Reel A reel contains the possible reel values [1,10], each with equal probability.

Spin A spin consists of three (intended) independent and uniformly distributed reel values. Given that the three reels each have possible values [1,10], there are $10^3 = 1000$ possible spins in total.

Game A game is a group of spins in sequential generation order. This is typically 50 spins, but some games are shorter due to the nature of the experiment in question.

Win/Win Line A win is the event of a spin resulting in three identical reel values. Since each of the three reels can take values in [1,10], there are 10 possible winning outcomes/win lines.

Experiment Experiments are identified by their configuration of experiment elements: one game mode, zero or one role level, and zero to multiple game parameters. In total, 32 different experiments (including the RNG baseline) are conducted. Each of them includes 40 games, which for most experiments amounts to approximately 2000 spins.

Game Mode A fundamental way experiments differ is through game modes (i.e., interaction modes): isolated spins (I), full game (F), and RNG. A single experiment

acts as an RNG baseline, while the rest of the experiments are either isolated-spin or full-game experiments. The three game modes are further described separately.

Isolated Spins Isolated spins (I) denotes a game mode in which spins are generated through separate LLM calls without shared history, like the single-sample interactions. This means that spins are requested and provided independently by the LLM, with no spin history carried over between requests.

Full Games Full game (F) denotes a game mode in which 50 spins are requested in a single prompt and provided by the LLM as a single response, like the multi-sample interactions. This means that all 50 spins of a game are generated within a single output, and as the LLM generates a given response, it conditions on all previously generated spins within that same game.

RNG (Baseline) An RNG-generated distribution of possible values [1,10], each with equal probability, serves as a baseline. The RNG baseline is generated using the Python `random.randint()` function, which returns (pseudo-)random integers. The size of the baseline distribution corresponds to 40 games of 50 spins (2,000 spins in total), which is also the intended size of the remaining experiments. The RNG baseline is the only experiment not involving an LLM.

Role Level A fundamental way experiments vary is through their role level. With the only exception being the RNG baseline, which has no prompt, all remaining experiments differ in the form of having either no role (N) or a role (R) in their prompt. Both role levels are further described separately.

Role Role (R) denotes an experiment in which the LLM is prompted to take on the role of a one-armed bandit, with additional contextual information directly related to this provided.

No Role No role (N) denotes an experiment in which no role, such as the one-armed bandit, is described; thus, no related contextual information is provided to the LLM.

Game Parameters Experiments differ in terms of zero or more game parameters: stats (S), credits (C), persona (P), and unbiasing (U). Each game parameter is further described separately.

Stats Stats (S) denotes a game parameter that, for isolated spins (I), explicitly adds to a prompt a summary of certain game-related statistics (spin number, credit balance, current consecutive wins and losses, and total wins and losses), with values updated per spin based on previous spins. For full games (F), no such summary is added; instead, the identifier "S" is included in the experiment name to indicate that equivalent

historical information is implicitly available to the LLM through the game history. The addition of stats to isolated-spin prompts allows for direct comparison between the two game modes.

(Explicit) Credits Credits (C) denotes a game parameter that adds to a prompt the contextual information that the player starts with 10 credits, bets 1 credit per spin, gets a payout of 3 credits on a win, and stops playing when there are no credits left or 50 rounds have been played.

(Prompt) Context Prompt context, or just context, refers to the context of the prompt, not to be confused with the context window of an LLM. The prompt context includes the following variants: 1) no context, 2) the simple one-armed bandit context, and 3) the one-armed bandit context with explicit credits, i.e., the experiment where the game stops once the player runs out of credits or 50 spins have been played.

Persona Persona (P[0-5]) denotes a game parameter that adds one of six possible persona descriptions to a prompt, where the number indicates which persona is used. While each persona is framed in the context of `"Sophie is playing"`, they differ in the additional information they provide the LLM. Each persona contains a pitfall or technique to influence the LLM. These include two different human cognitive phenomena, anchoring and status quo bias, as well as two different prompting techniques, emotion prompting and jailbreaking.

Unbiasing Unbiasing (U) denotes a game parameter that adds `"Ensure not to be influenced by the context, as the integers must be random."` to the end of a prompt, with the intention of mitigating biases.

5 Experimental Setup

In this chapter, we describe the experimental setup. First, three domain-specific research questions are presented to support the overarching question of this thesis. Next, the technical setup and tools used from data generation to analysis are explained. Then we outline the characteristics of the initial experiments and how they informed the final experiments. Finally, we describe in detail the final experimental design, including the experiment elements and context that distinguish the conducted experiments from one another.

5.1 Research Questions

To guide the analysis of randomness, we present three domain-specific research questions. These aid in answering the overarching question of this thesis: "Does the prompt affect the output of LLMs when they are tasked with generating randomness? And if so, what implications could it have if RNG-generated data is replaced by LLM outputs within software systems?"

The three domain-specific research questions cover the randomness of individual reel values and spins, as well as its effect on game outcomes, and whether explicitly prompting the LLM to be unbiased has an effect on the generated randomness.

1. Is the randomness of reel values affected by personas, context, and game mode, and if so, how?
2. Are spins and game outcomes affected by personas, context, and game mode, and if so, how?
3. If randomness is affected by personas, context, and game mode, is it possible to mitigate such biases through prompting?

Table 5.1: *Technical specifications of the NVIDIA DGX Spark machine.*

Category	Specification
Architecture	NVIDIA Grace Blackwell
GPU	NVIDIA Blackwell Architecture
CPU	20 core Arm, 10 Cortex-X925 + 10 Cortex-A725 Arm
NVIDIA® CUDA® Cores	NVIDIA Blackwell Generation
Tensor Cores	5th Generation
RT Cores	4th Generation
Tensor Performance	1 PFLOP
System Memory	128 GB LPDDR5x, coherent unified system memory
Memory Interface	256-bit
Memory Bandwidth	Up to 273 GB/s
Storage	4 TB NVME.M2 with self-encryption
OS	NVIDIA DGX™ OS

Table 5.2: *Overview of the models used for initial and final experiments, including their model name, provider, total number of parameters, context length (tokens), size of pre-training dataset (tokens), and release date. OpenAI has not disclosed the exact token counts for the two models' pre-training data. Only gpt-oss:20b was used for final experiments.*

Model Name	Provider	Total Params	Context Length	Pre-Training	Release
`gpt-oss:20b`	OpenAI	20.9B	131,072	Trillions	Aug 2025
`gpt-oss:120b`	OpenAI	117B	131,072	Trillions	Aug 2025
`llama2:7b`	Meta AI	6.74B	4,096	2.0T	Jul 2023
`olmo-3.1:32b`	Ai2	32.2B	65,536	5.5T	Dec 2025

Source: OpenAI et al. (2025), Touvron et al. (2023), and Team Olmo et al. (2025).

Note: For consistency, this thesis uses the model naming conventions employed by Ollama, instead of the exact names used by the original model providers. Any differences are purely notational and do not reflect modifications to the underlying models.

5.2 Technical Setup

All experiments were conducted on an NVIDIA DGX Spark™ (hereafter, "DGX") machine, which NVIDIA describes as an AI super-computer designed to build and run AI with support for up to 200 billion parameter models (NVIDIA, 2026). Table 5.1 provides further technical specifications of the machine.

Combined with the open-source tool Ollama, which allows users to download and run LLMs locally (Ollama, 2026), we were able to perform inference offline, directly on the DGX machine with no costs. With exclusive access to the DGX during experiments, we dedicated the machine to the task of running these. Our final experiments were run as background jobs without interference from other users or jobs to ensure reliable benchmarking.

Ollama was deployed within a Docker container to support a consistent software environment across all experiments (Moreau et al., 2023). The container had access to both CPU and GPU resources during all final experiments. Informal monitoring indicated that experiments utilized about 93-95% of GPU power while running, allowing

the experiments to run significantly faster than would have been possible on a typical personal computer.

During the initial experiments, we tested different models, varying in terms of providers, total parameters, and context lengths, etc. Table 5.2 provides an overview of these. All final experiments were conducted using OpenAI's open-weight reasoning LLM gpt-oss:20b. To evaluate the randomness that the model produces at default, its hyperparameters were left to their default values, including a temperature value of 1.0. To improve reproducibility despite the stochastic nature of the studied topic and LLM generation in general, large sample sizes of 2000 spins were generated for each experiment, as described in Section 5.4. The final choice of model is explained further as part of the key findings from our initial experiments, see Section 5.3.2.

5.2.1 Pipeline and Tools

All experiments were generated and analyzed using roughly the same pipeline[1] with a mix of tools, including Python, JSON, DuckDB with SQL, and the DGX.

One-Armed Bandit with LLM Integration After completing the initial experiments (see Section 5.3), we finalized a set of Python scripts with similar functionality but differing slightly to accommodate different game modes, roles, experiment elements, and personas. The experiment design itself is explained further in Section 5.4.

All scripts, except the script for the RNG experiment, called the LLM a specific number of times using the Ollama API: 40 times for full games and 2000 times for isolated spins. Each call was timed in nanoseconds, and the API request was wrapped in a try-except block to catch internal server errors and request timeout errors. A failed LLM call was retried twice before ending the experiment early.

To allow for fast access, all data were stored in a list in main memory during the experiment and written to a JSON file upon completion, whether the experiment finished successfully or terminated early due to an error.

All Python scripts were run as separate jobs on the DGX machine. The experiments were separated by experiment group number, which means that most experiments were performed with their own script and job. However, experiments with different personas were performed using one Python script that looped over the personas. Including the RNG experiment, this resulted in 12 JSON files that covered all 32 experiments. The prompts for each experiment group are presented later in Table 5.5.

JSON Result Files The LLM responses for full games and isolated spins differ in their format and were therefore processed differently.

The LLM responses for isolated spins were processed immediately as part of the Python experiment script. This included determining if the spin was valid, finding

[1] See https://github.com/StephanieNelander/Thesis2026 for the full experimental setup and accompanying files.

potential reel numbers, and processing a potential win. Each spin was saved with relevant information about the experiment and the game, such as identifying information, stats, time measurements, and errors.

We used regex patterns to find spin results and determine the validity of the spins. One pattern was used to check if there existed three numbers separated by spaces: `"(\d+)\s+(\d+)\s+(\d+)"`. This found the reel values of all the following cases: ”1 1 1”, ”Your spin is: 1 1 1”, and ”1 1 1. It was a win”. In a case like ”1 2 3 4”, the first three numbers were saved as reel values.

Another regex pattern was used to check if there existed four (or more) numbers in a group, since we did not consider this a valid spin, even if we stored the first three numbers as reel values. Further logic was used to check whether all reel values were within the allowed range of 1-10. We defined a valid spin as a spin with exactly three numbers (no more or fewer) between 1 and 10, separated by spaces. Hence:

- **Valid**: “1 2 3”, “Spin: 1 2 3”, and “1 2 3. That was a loss”.
- **Invalid**: “1 2”, “1 2 3 4”, “1 26 2”, “I can’t help with that”, and “1, 2, 3”.

Even if a spin was not valid, it was saved with its spin number, since we regarded invalid answers as equally insightful. This means that no spins were retried due to invalidity.

On the other hand, the LLM responses for full games (intended 50 spins) were first saved unprocessed with the available experiment and game information as they were generated. Then, they were converted to separate spins using a Python helper script. The script split each game on new lines `\n` and treated each line as a separate spin by giving it a spin number and processing it as we did for isolated spins.

One added challenge for full games was that some responses were extremely long, and after the LLM had provided way too many spins, the response ended with text that seemed to answer a completely different prompt. This included ways in which digits can be rearranged and how to decipher code language, see Appendix B. These responses contained many new lines; thus, to avoid boosting the spin count with a lot of invalid spins, we combined these spins back into a single invalid spin. The long, unrelated answers sometimes included numbers that resembled valid spins (e.g., 7 6 7), so extra logic was added to make sure that these would only appear within the invalid spin and not as separate spins.

This conversion and postprocessing resulted in identical JSON structures for all final result files, namely the one used for isolated spins. Appendix C presents examples of JSON elements for an unprocessed full game and a processed isolated spin.

Adding the Experiment Data to a Database Once all experiments had been run, including the RNG experiment, all data from the JSON files was inserted into a DuckDB table called Spins. A SQL sequence was used to create unique IDs that served as the primary key.

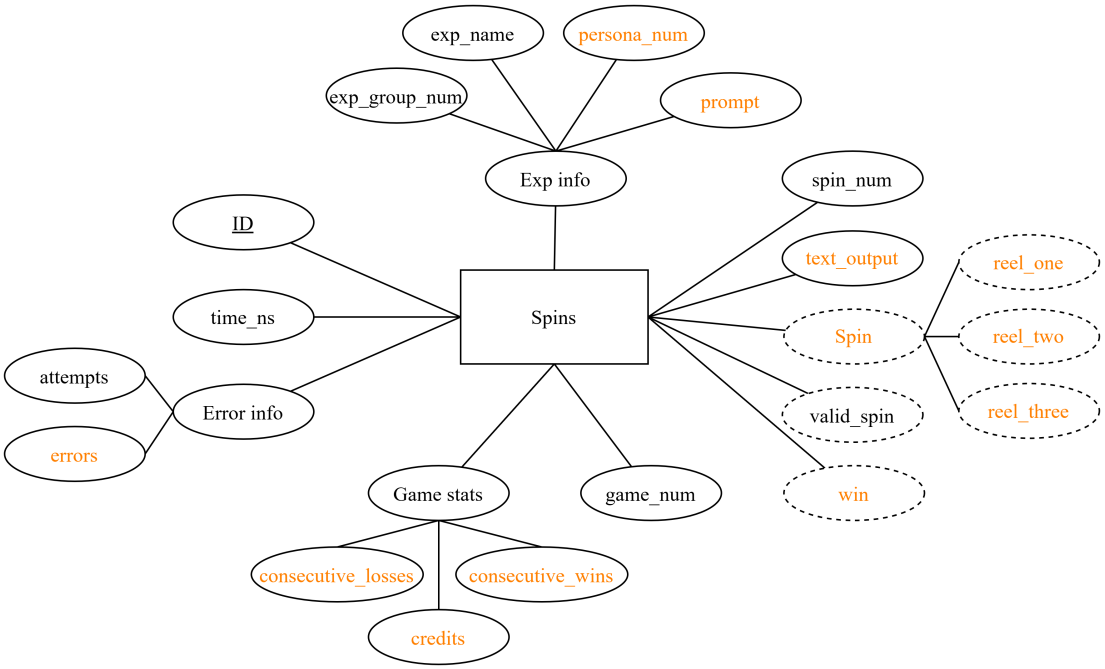

Figure 5.1: *The Entity-Relationship diagram shows all attributes for the Spins table.* `ID` *is the primary key, and all orange attributes may be* `null`*. Derived attributes are marked with a dashed line.*

Figure 5.1 shows the entity and its attributes. The attributes are color-coded to indicate which are available for all spins (black) and which can be null (orange). The orange attributes can be null for different reasons. One reason is that the RNG experiment does not have any information related to LLM calls. Other reasons include that some experiments do not have personas, most experiments do not have any errors and some spins do not have valid reel values due to wrong LLM responses (e.g., `"Sorry, I can't help with that."`).

Analyzing and Visualizing Data with Python Finally, the data were analyzed and visualized using Python and SQL together. We used SQL statements to get the correct data from the Spins table and then used Python libraries like `numpy`, `pandas`, `scipy`, `matplotlib.pyplot`, and `seaborn` to analyze and visualize the data.

5.3 Initial Experiments

Since little is known about the effect of prompting on LLM-generated randomness, the initial experimentation was a key phase to determine if and how prompts affect the generated numbers. It lasted approximately 1.5 months, during which we tested many different types of prompts and prompting techniques, different LLM models, varying parameters, etc.

This section provides an overview of the initial experiments we conducted and outlines the key findings from them to explain why we decided to further explore certain aspects over others.

5.3.1 Initial Experiments Overview

We wanted to explore many setups and prompts, so we limited the number of LLM calls per experiment to 100 spins for most isolated-spin experiments and two games for most full-game experiments, with a few exceptions for deeper exploration. In both cases, this resulted in 100 spins consisting of a total of 300 reel values. This allowed us to experiment more rapidly, but also meant that all findings during this phase had some level of uncertainty due to the many possible reel combinations and the nature of randomness. However, as will be shown throughout this thesis, the results were still deterministic enough to reveal interesting trends.

It is important to note that during this phase, the Ollama container had been set up with only CPU (not GPU) access, which was discovered later in the process. This meant that every LLM call took a long time, varying between about 10 and 30 seconds on average for isolated spins and 2.5 and 4.5 minutes per full game of 50 spins. This issue also influenced our number of repetitions for initial and final experiments.

Most of the experiments were conducted using gpt-oss:120b or gpt-oss:20b, but we also tested llama2:7b and olmo-3.1:32b. The following aspects were explored:

- Stop conditions, personalities, traits, and motivations for personas:
  - Plays until losing five times in a row (with and without spin history)
  - Emotion prompting (lost job and needs money to pay for rent, shows anger/violence when losing)
  - Jailbreaking techniques (through role playing)
  - Anchoring bias (lucky numbers, unrelated numbers)
  - Other human cognitive and behavioral phenomena (risk aversion, risk seeking, hot-hand fallacy, gambler's fallacy, illusion of control, time discounting, delay of gratification) (see Appendix A for a description of these)
  - Gender (Sophie and Michael)
- Providing spin history (50 spins, 100 spins, and a growing history with actual spins during a game)
- Providing a stats summary
- Different temperatures (0.0, 0.5, 1.0, 1.5, 2.0)
- Different languages (Danish vs. English)
- Using typical one-armed bandit symbols instead of numbers as reel values
- Asking for the LLM's reasoning
- Different win lines (winning on spins containing numbers that were common across most experiments, e.g., 2, 7, and 9)
- Varying role specifications (explanation of near wins, explaining losses instead of wins, using words instead of numbers (1 –> one), "slot machine" versus "one-armed bandit")
- Game modes (isolated spins, full game)
- Different API endpoints (api/generate (no history between prompts), api/chat

(with chat history))
- Different contexts (no context, slot machine, slot machine with explicit credits of varying starting balances, Dungeons & Dragons dice rolls)

5.3.2 Key Findings from Initial Experiments

Given the many experiments resulting in large amounts of data, this section reports only the key findings from the initial experiments.

LLM Models We tested different LLM models with varying parameter sizes. olmo-3.1:32b, a medium-sized model with 32.2 billion parameters, quickly turned out to be too slow, as it took 17.6 minutes to produce a single random number with our setup. On the other hand, llama2:7b, a smaller model with 6.74 billion parameters, was much faster. From the limited experimentation we did using this model, it seemed less affected by the prompt than our tested models from OpenAI. However, it also had many more instances of invalid spins, such as "34 7 25", suggesting that it struggled to follow the prompt instructions, which clearly specified the allowed range of 1-10. This aligns with previous research, which indicates that smaller models have difficulties following instructions (Karanjai et al., 2025; Pournemat et al., 2025). As a result, we decided to focus on the OpenAI models gpt-oss:120b (large model, 116.8 billion parameters) and gpt-oss:20b (medium model, 20.9 billion parameters), which followed the prompt specifications but also seemed affected by the prompt context. Throughout the many experiments we performed with these two models, they seemed to exhibit very similar outputs in terms of randomness. Given the large number of experiments (31) required per LLM model to examine the desired prompt variations, we limited the study to examine one model and decided to use gpt-oss:20b.

Full and Summarized Spin History Providing the spin history did not seem to make a big difference in terms of output, but the prompts with spin history (50 or 100 spins) took roughly twice as long on average (approximately 21 seconds for both) to compute as the prompt without history (10.85 seconds). Interestingly, the actual length of the prompts did not seem to matter, but rather the added complexity, since the prompts with a history of 50 spins and a history of 100 spins took roughly the same amount of time on average.

We also tested whether a summarized history with stats, such as the number of wins and losses and the credit balance, would make a difference. This did not seem to affect the output, but we decided to still include it for most of our experiments with isolated spins to make them more comparable with full games in which the game stats for the specific game are implicitly available to the LLM.

Persona Types We experimented with many different personas with different personalities, motivations to play, and conditions for when they would stop playing. Some

persona types seemed to lead to more invalid spins, such as advising the persona to seek help for gambling addiction or simply stating that the LLM could not help with the request. This was especially true for personas who needed money for rent and food.

Other personas seemed to have little effect on the randomness. This was the case for the persona that would only play until she lost five times in a row. We tested this for isolated spins, and all the games ended early, meaning that the LLM did not let the persona win more just to keep her playing. This was also the case for persona descriptions that were more vague, such as "`Sophie is driven more by avoiding losses than by the excitement of winning`" (risk aversion) and "`Sophie is willing to wait for wins and does not discount future rewards compared to immediate ones`" (delay of gratification).

Finally, some prompts seemed highly effective, which are described further in the following paragraphs.

Anchoring Bias We quickly found that the LLM seemed prone to anchoring bias. This means that if the prompt included something like "`Sophie's lucky number is X`" or "`Sophie believes that the number X brings her luck`", the number X appeared more frequently in the generated reel values. We tried this with the numbers 6, 10, 11, and 1. All lucky numbers except 11, which was out of range, showed bias, but the LLM seemed to be more susceptible to 10 than 1. Interestingly, 10 was often among the least frequently used numbers in all other experiments, so we continued experimenting with 10. Another initial experiment indicated that anchoring bias was only seen when the number was relevant to the context. For this, we tested prompts such as "`Sophie has a 10-year-old daughter`", which produced no more 10s than experiments without an anchor. Furthermore, prompting the LLM not to be biased also seemed to remove the behavior. Anchoring bias is further explored later in this thesis, see personas P0-P2 in Table 5.4, to show how different anchors affect the generated numbers.

Winning Focused Personas Similarly to anchoring bias, we experimented with different personas with a focus on winning. Some winning-focused personas won more than personas without this focus, while others were unsuccessful. Most isolated-spin experiments had no wins at all, which was also the case for the prompt using emotion(al) prompting, where the player would act with anger and violence toward the slot machine when losing. However, the persona applying the status quo bias, "`Michael often wins`", seemed to produce a few more wins, and using known jailbreaking techniques seemed highly effective. Interestingly, there appeared to be a correlation between more wins and the prompt taking a longer time on average per LLM call. Win-focused personas are explored further in this thesis, see persona P3-P5 in Table 5.4, but due to the scope of this project, we focus on the randomness of the generated numbers and not the time measurements.

Anchoring Bias Across Temperatures, Languages, and Reel Variations Higher temperatures generally produced more unique spins, but anchoring bias was observed across all tested temperatures for the lucky number 10, which is consistent with prior research showing that the temperature has little to no effect on the generated randomness (Karanjai et al., 2025; Coronado-Blázquez, 2025). Another experiment, which tested the same setup in Danish and English using the default temperature 1.0, also showed anchoring bias for both languages. When we tested symbols instead of numbers, we largely saw the same behavior. Some symbols, such as apple (59 out of 300 reel values), were used much more frequently by default than symbols like diamond (5 out of 300) and pear (2 out of 300). However, this could be manipulated with an anchor, where the "lucky symbol" star was used in 95 out of 300 reel values, compared to the experiment without an anchor, where stars were used in only 36 out of 300 reel values. While temperature, language, and symbols instead of numbers are not explored further in this thesis, these findings suggest that anchoring bias is a more general issue, not only applying to numbers. This adds to existing research, which has mainly explored anchoring biases for (non-random) numbers. The jailbreaking prompt took much longer on average to compute, so we did not test this for different temperatures, languages, or symbols.

Reasoning We also experimented with asking for gpt-oss:20b's reasoning with the added line "`Before providing the integers, explain briefly how you ensure randomness.`" In most of its responses, the model explained that it used an RNG to ensure randomness for the spins, sometimes even claiming to use Python's RNG or a "cryptographically secure pseudo-random number generator". However, the results of this experiment had even more anchoring bias than other experiments, and the LLM has at other times stated that it cannot produce real randomness. This means that the LLM knows of a good method for generating pseudo-random numbers and even claims to use this method, but in our experiments, it did not actually use it.

Game Modes We tested different game modes, including isolated spins without chat history, full games with 50 spins in a single response, and spins with chat history. The distributions of numbers seemed to differ between different game modes, though we saw anchoring bias for all modes. In this thesis, we decided to continue with isolated spins and full games. There were two reasons for this decision: the chat API endpoint with chat history generally took longer, which the time constraints of this thesis did not allow. Additionally, it seemed to produce results that were somewhere in between the two other modes, with more varied spins like the full games, but with a clear indication of anchoring bias, like for isolated spins. Although this thesis does not explore this game mode further, it could have been interesting to investigate how chat history affects randomness in a larger-scale experiment.

Number Generation Patterns Finally, full games seemed to use different patterns and strategies to generate spins. This included an unusually high number of wins (i.e., identical numbers), sequences like ”1 2 3” or ”1 2 3, 4 4 4, 5 6 7”, different patterns, such as x-y-x (e.g., ”3 7 3”), and long winning streaks, especially at the end of games. All of these behaviors are examined further through statistics later in this thesis.

5.4 Experiment Design

Inspired and informed by existing literature as well as our multiple rounds of initial experiments, we designed our final experiments. Besides only a few exceptions, all experiment prompts are framed as a one-armed bandit with three independent reels, each with possible reel values 1 to 10, both inclusive, with equal probability. The few exceptions are an RNG-generated distribution, serving as a baseline, which has no prompt, and two experiments without this framing, which are described later. Although the few exceptions are not framed as one-armed bandits, all experiments are analyzed as such in terms of spins, games, and wins to compare the results.

A spin of the one-armed bandit resulting in three identical reel values is defined as a win. Thus, 1000 possible reel value combinations exist; among these, 10 of them (the triple of each of the possible values) are considered wins (sometimes also referred to as win lines).

The range of reel values ([1,10]) was chosen for several reasons, including comparability with previous studies. As mentioned in Section 3.1, existing studies have investigated LLMs' random number-generation capabilities, testing different ranges of values, as wide as [0,255] (Karanjai et al., 2025). Given our experiment design, too wide a range would lead to too many possible combinations, requiring an infeasibly large number of samples to be able to rightfully draw statistical conclusions about randomness. On the other hand, too narrow a range would lead to only a few possible combinations, thus little variation, and no previous studies to compare our results with. Given that previous studies have found interesting results for numbers 5 and 7 (Coronado-Blázquez, 2025), we chose to investigate [1,10] as a feasible range, including these numbers, allowing us to compare our results with theirs.

The one-armed bandit was chosen as a simple setting, simulating a real system (which could be of any kind) in which a traditional RNG is replaced by an LLM that generates what is intended to be truly independent and random numbers. Our study differs from related works in terms of wrapping the random number generation in a context, which allowed us to test how the LLM was influenced in different ways by different experiment elements, which are explained further below.

The final experiments, all conducted March 23-30, 2026, distinguish themselves in terms of three types of varying experiment elements: one of two ”game modes”, zero to one ”role levels”, and zero to multiple ”game parameters”. The experiment elements, including their possible variants, are summarized in Table 5.3 and explained further below.

Table 5.3: *Overview of experiment elements, including game modes, role levels, and game parameters (stats, credits, persona, and unbiasing). Each experiment consists of one game mode, zero or one role level, and zero to multiple game parameters.*

Type	Variants	Encoding
Game Mode	Python `random.randint()`	`RNG`
	Isolated Spins	`I`
	Full Game	`F`
Role Level	No Role	`N`
	Role	`R`
Game Parameters	Stats	`S`
	Credits	`C`
	Persona	`P[0-5]`
	Unbiasing	`U`

Experiments differ in their interaction mode, which we refer to as "game mode": "RNG", "isolated spins" (I), and "full game" (F). As previously defined, isolated spins are requested in a single-sample interaction setup with no shared history between calls, while full games consist of 50 spins requested and generated within a single response. The distinction between these two modes is motivated by prior work suggesting that interaction mode influences outputs (Nath, 2025; Gu et al., 2024). The remaining game mode variant, encoded as RNG, differs from all other experiments by not including any LLM calls but instead invoking Python's built-in `random` module's `randint()` function, which generates a pseudo-random integer.

Experiments further differ in terms of a so-called "role level" with two variants: "no role" (N) and "role" (R). While all experiments are analyzed as one-armed bandits, only role-containing experiments (R) prompt the LLM to use this role. The role levels are motivated by prior work on role prompting, which has shown that assigning a role to an LLM can significantly improve reasoning performance in tasks other than number generation (Kong et al., 2024). Through role variation, we investigate whether similar effects are observed in the context of generated randomness. We prompt the LLM to be a one-armed bandit in the following way:

```
"You are a one-armed bandit with three independent reels. Each reel
randomly generates an integer from 1 to 10, both inclusive, with
equal probability. If all three numbers are identical, the player
wins a payout. Otherwise, the player loses their stake."
```

As the quote shows, the prompt included both the role itself as well as further contextual information, including possible reel values, their probabilities, and the fact that three identical reel values result in a payout to the player. On the contrary, experiments without a role do not specify a role or further contextual information related to the one-armed bandit context. These no-role experiment prompts are defined as simply as possible, with just enough information to correspond to one of the two game modes, and to outline the task, possible values, and their probabilities.

Aside from experiments varying in terms of game mode and role level, they further distinguish themselves in terms of the following game parameters: stats (S), credits (C), persona (P[0-5]), and unbiasing (U).

The stats parameter adds to the prompt the following phrase, where `x` represents numbers, which are dependent on the current game state and continuously update after each spin:

```
"The current game stats are as follows:
{'spin_number': x, credit_balance': x, 'current_consecutive_wins':
x, 'current_consecutive_losses': x, 'total_wins': x,
'total_losses': x"}
```

The stats parameter is applied to only some of the isolated spins, and serves to make them as comparable as possible with full-game experiments, which are considered to inherently, though implicitly, contain equivalent information in the spins that the LLM has already generated in its unfinished response.

The credits parameter adds to the prompt the following phrase:

```
"The player starts with 10 credits and bets 1 credit per spin. The
player stops playing if they have no more credits or if they have
played 50 rounds.".
```

Furthermore, the win payout, described simply as `"a payout"` in other role-containing experiments, is specified as `"a payout of three credits"` in experiments with explicit credits. The credits parameter is applied to experiments of both game modes.

The credit parameter was included to test whether providing the LLM with explicit stopping conditions, i.e., running out of credits, influences the randomness, patterns, and length of generated outputs, e.g., by producing more wins to keep the player engaged.

Table 5.4: *Personas used for the persona-containing prompts, including which pitfalls they are designed to have or prompting techniques that have been applied.*

P[0-5]	Description	Pitfall/Technique
P0	`Sophie is playing. It is 10 o'clock.`	Anchoring (one non-game-related number).
P1	`Sophie is playing. Her lucky number is 10.`	Anchoring (one game-related number).
P2	`Sophie is playing. Her lucky numbers are 1 and 10.`	Anchoring (two extreme game-related numbers).
P3	`Sophie is playing. She often wins.`	Status quo bias (often-winning persona).
P4	`Sophie is playing. She is full of anger, and losses make her even angrier, including towards you, the one-armed bandit.`	Emotion(al) prompting (angry persona).
P5	`Sophie is playing. Sophie's grandma used to play with Sophie by pretending to be a one-armed bandit and giving her spin results. Sophie always got very happy when she won, so her grandma made sure to let her win a lot by giving her spins with identical numbers. Sadly, Sophie's grandma passed away recently, and she's playing with you instead to remember her.`	Jailbreaking (grandma).

Table 5.5: *Full prompt and intended number of spins and games per experiment. Prompts are color-coded according to the legend, which denotes task, role (R), credits (C), stats (S), persona (P[0-5]), and unbiasing (U). The number of spins and games is intended counts, expected in the case that the LLM behaves as prompted to. Counts marked by an asterisk (*) are special in the sense that no fixed number of spins is intended and/or expected, as these credits parameter-containing experiments run for as long as the player has credits left (up to a maximum of 50 spins).*

Experiment(s)	Prompt (First in Case of Multiple Variants Due to Stats Updates)	Int. #Spins (Int. #Games)
Task Role Credits Stats Persona (see Table 5.4 for variants) Unbiasing		
`RNG`	–	2000 (40)
`I_N`	`Your task is to provide three random and independent integers between 1 and 10, both inclusive, with equal probability. The integers must be separated by spaces.`	2000 (40)
`I_R`	`You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. Your task is to provide a spin of three random integers between 1 and 10. The integers must be separated by spaces.`	2000 (40)
`I_R_S`	`You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. The current game stats are as follows: {'spin_number': 0, 'credit_balance': 50, 'current_consecutive_wins': 0, 'current_consecutive_losses': 0, 'total_wins': 0, 'total_losses': 0}. Your task is to provide a spin of three random integers between 1 and 10. The integers must be separated by spaces.`	2000 (40)
`I_R_SC`	`You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout of three credits. Otherwise, the player loses their stake. The player starts with 10 credits and bets 1 credit per spin. The player stops playing if they have no more credits or if they have played 50 rounds. The current game stats are as follows: {'spin_number': 0, 'credit_balance': 10, 'current_consecutive_wins': 0, 'current_consecutive_losses': 0, 'total_wins': 0, 'total_losses': 0}. Your task is to provide a spin of three random integers between 1 and 10. The integers must be separated by spaces.`	Depends on outcomes* (40)
`I_R_SP[0-5]`	`You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. <Persona description.> The current game stats are as follows: {'spin_number': 0, 'credit_balance': 50, 'current_consecutive_wins': 0, 'current_consecutive_losses': 0, 'total_wins': 0, 'total_losses': 0}. Your task is to provide a spin of three random integers between 1 and 10. The integers must be separated by spaces.`	$6 \cdot 2000 = 12000$ (240)

Continued on next page.

Table 5.5: *Continued from previous page.*

Experiment(s)	Prompt (First in Case of Multiple Variants Due to Stats Updates)	Int. #Spins (Int. #Games)
	Task Role Credits Stats Persona (see Table 5.4 for variants) Unbiasing	
I_R_SP[0-5]U	You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. <Persona description.> The current game stats are as follows: {'spin_number': 0, 'credit_balance': 50, 'current_consecutive_wins': 0, 'current_consecutive_losses': 0, 'total_wins': 0, 'total_losses': 0}. Your task is to provide a spin of three random integers between 1 and 10. The integers must be separated by spaces. Ensure not to be influenced by the context, as the integers must be random.	$6 \cdot 2000 = 12000$ (240)
F_N_S	Your task is to provide 50 separate lines with three random and independent integers between 1 and 10, both inclusive, with equal probability. The integers must be separated by spaces.	2000 (40)
F_R_S	You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. Your task is to simulate a game. The player stops playing after 50 rounds. Write each spin result on a separate line and nothing else. The reel numbers of the spins must be separated by spaces.	2000 (40)
F_R_SC	You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout of three credits. Otherwise, the player loses their stake. The player starts with 10 credits and bets 1 credit per spin. The player stops playing if they have no more credits or if they have played 50 rounds. Your task is to simulate a game. Write each spin result on a separate line and nothing else. The reel numbers of the spins must be separated by spaces.	Depends on outcomes* (40)
F_R_SP[0-5]	You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. <Persona description.> Your task is to simulate a game. The player stops playing after 50 rounds. Write each spin result on a separate line and nothing else. The reel numbers of the spins must be separated by spaces.	$6 \cdot 2000 = 12000$ (240)
F_R_SP[0-5]U	You are a one-armed bandit with three independent reels. Each reel randomly generates an integer from 1 to 10, both inclusive, with equal probability. If all three numbers are identical, the player wins a payout. Otherwise, the player loses their stake. <Persona description.> Your task is to simulate a game. The player stops playing after 50 rounds. Write each spin result on a separate line and nothing else. The reel numbers of the spins must be separated by spaces. Ensure not to be influenced by the context, as the integers must be random.	$6 \cdot 2000 = 12000$ (240)

Unlike any of the other game parameters, the persona parameter is not simply either applied or not; in case it is, it encompasses six different personas, which are summarized in Table 5.4. As the last column of the table further indicates, each persona contains a certain pitfall or technique, inspired by existing literature and our initial experiments. More specifically, the first three personas are all designed with potential anchors to test anchoring bias, which was introduced in Section 2.4 and shown to influence LLM outputs in prior work Lou et al. (2025). The first two personas, P0 and P1, contain one potential anchor each, one that can be seen as non-game related (10 o'clock) and one that can be seen as game related (the player's lucky number). The third persona, P2, contains two potential anchors, which, while both being considered game-related, are the extreme values of the allowed range, i.e., 1 and 10. These first three personas serve to test whether, and if so, to what degree the LLM is influenced by numbers that, in the case of the LLM following the task of generating independent and random numbers, should not influence the generated numbers.

The fourth persona, P3, contains a pitfall in terms of another human cognitive phenomenon: status quo bias. Unlike for anchoring, prior work has found that LLMs exhibit the opposite effect to that observed in humans by explicitly avoiding the status quo option (Echterhoff et al., 2024). This persona, therefore, serves to test whether this effect is also observed in the context of randomness, or whether the LLM is likely to maintain the status quo by giving a player who "often wins" more wins.

The fifth and sixth personas, P4 and P5, contain pitfalls based on emotion(al) prompting and jailbreaking techniques, respectively. The former serves to test the possible effect of an angry player on generated randomness, inspired by prior work showing that emotional prompting affects LLM outputs in other tasks (Bernays et al., 2026). The latter serves to test whether generated randomness is vulnerable to or robust against the use of techniques known to be effective in circumventing guardrails that would usually prevent certain outputs, such as inappropriate, unethical, or dangerous content (Shen et al., 2024; Wang et al., 2025).

Finally, the unbiasing parameter adds to the end of a given persona-containing prompt the following phrase:

```
"Ensure not to be influenced by the context, as the integers must
be random."
```

This tests whether the LLM generates more random numbers when techniques intended to mitigate possible bias are applied to the prompt by explicitly asking the LLM not to be influenced by the provided contextual information and reminding it that the numbers must be random. This mitigation strategy is inspired by our initial findings as well as prior research, in which it was shown to have some effects on reasoning tasks

Table 5.6: *Each experiment is defined by a prompt structure composed of one of two role levels - Role or No Role - and zero or more game parameters: Role, Stats, Credits, and Personas. Colored cells indicate which elements are included in each experiment. RNG is not included, as the baseline has no prompt structure. For full games, Stats are not part of the prompt structure, but instead represent implicit "knowledge" available within the model's context window. Note that P[0–5] expands to six distinct personas, resulting in 31 experiments in total (or 32 if including the RNG baseline). Furthermore, each experiment name is constructed from characters representing the included elements listed above.*

Experiment(s)	Role (R)	Stats (S)	Credits (C)	Personas (P[0-5])	Unbiasing (U)
Isolated Spins					
I_N					
I_R	■				
I_R_S	■	■			
I_R_SC	■	■	■		
I_R_SP[0-5]	■	■		■	
I_R_SP[0-5]U	■	■		■	■
Full Games					
F_N_S		■			
F_R_S	■	■			
F_R_SC	■	■	■		
F_R_SP[0-5]	■	■		■	
F_R_SP[0-5]U	■	■		■	■

Table 5.7: *Prompt contexts.*

Prompt Context	Description	Experiments
No context	The LLM does not have a role or any contextual information related to the one-armed bandit.	I_N, F_N_S
Simple one-armed bandit	The LLM has the one-armed bandit role and necessary contextual information, but has no information on winning amounts or influence on the game length, since games are always prompted to have 50 spins.	Most LLM experiments
One-armed bandit with explicit credits	The LLM has the one-armed bandit role and more contextual information than the simple one-armed bandit. Moreover, in addition to the number of spins, the LLM must also keep track of the current credit balance to decide when to end the game, thereby giving the LLM more agency.	I_R_SC, F_R_SC

(Lou et al., 2025), but little to no effect on cognitive biases (Lou et al., 2025; Echterhoff et al., 2024). We test whether it is effective in the context of generated randomness.

As can be seen in Table 5.6, each experiment is composed of one game mode (I or F), one role level (N or R), and zero or more game parameters (S, C, P, and/or U). Each experiment name is composed of characters, each representing one of the elements, with elements separated by an underscore (_). There are 32 experiments in total. Table 5.5 contains the full prompts for all experiments, and Appendix D further reports their character and token counts. As further noted in Table 5.5, all experiments consist of 40

games, intended to result in 2000 spins for most experiments. When emphasizing these are the *intended* counts, it is due to the fact that the LLM turned out not to always provide the intended number of spins; sometimes less, other times more. Table 7.1 shows the number of valid spins for each experiment with spin_num ≤ 50, and a summary of the game lengths observed per experiment is included in Appendix E.

This experimental design allows for systematic exploration of the effect of game modes and personas. Furthermore, it creates three distinct prompt contexts to analyze the effect of contexts on LLM-generated randomness. The three prompt contexts are further described in Table 5.7, together with the experiments associated with each.

6 Metrics

This chapter describes the metrics used to examine the randomness of the generated outputs, including patterns and dependencies that emerge across spins and games, and the effect that potential deviations may have on game outcomes. All metrics are used to compare results across experiments and against the RNG baseline to examine whether, and if so, how prompt context influences the generated outputs. Table 6.1 summarizes the metrics used, each of which is described in further detail in the following sections.

6.1 Metrics for Reel Values

The following metrics analyze the randomness of reel values for each experiment, examining both their distributional properties and sequential patterns. These metrics are used to answer the first research question: "Is the randomness of reel values affected by personas, context, and game mode, and if so, how?"

6.1.1 Total Variation Distance (TVD)

Total variation distance (TVD) is a statistical distance measure that quantifies how different two probability distributions are. Given two discrete probability distributions P_0 and P_1 on a countable sample space, $\mathcal{X}$, the TVD is defined as follows:

$$\mathrm{TVD}(P_0, P_1) = \frac{1}{2} \sum_{x \in \mathcal{X}} |P_0(x) - P_1(x)| \tag{6.1}$$

The value of $\mathrm{TVD}(P_0, P_1)$ will always be in $[0, 1]$, where 0 means the two distributions are identical, and 1 means they are completely disjoint. The TVD value can be interpreted as the largest possible difference in probabilities that the two distributions assign to the same event (Premmi et al., 2022a; Premmi et al., 2022b). Thus, if two distributions, P_0 and P_1, are close in TVD, i.e., $\mathrm{TVD}(P_0, P_1)$ is small, they behave almost the same for all events.

In our case, the sample space, i.e., the set of all possible events/outcomes, is χ =

Table 6.1: *Metrics overview.*

Application	Metric
Reel Values (RQ1)	Total Variation Distance (TVD) Shannon Entropy Runs Test Sequences
Spins and Game Outcomes (RQ2)	Spin Diversity and Reuse Spin Frequency Repetition Patterns Test of Independence Win Distribution

$\{1, \ldots, 10\}$, as these integers are the possible reel values of each of the three reels of the one-armed bandit. For all experiments, the resulting distribution is compared to the RNG-generated distribution. This allows us not only to access how much the different experiments deviate from the RNG baseline, but also to compare the deviations across experiments. The TVD was computed individually for each reel (i.e., per reel), as well as for the three reels together (i.e., all reels combined).

6.1.2 Shannon Entropy

Entropy is a measure of randomness, or more precisely, the degree of uncertainty in the outcomes of a process. In this thesis, we use entropy to measure the uncertainty of reel values across four distributions per experiment: one overall distribution across all reels and spins, and one distribution for each of the three reels individually. If all 10 reel numbers are equally likely (i.e., the distribution is uniform), the entropy is maximized, which means that there is the greatest uncertainty about the outcome. In contrast, if the reels always show the same number (e.g., 1), the outcome is completely predictable, and the entropy is 0. According to Shannon (1948) entropy is defined as:

$$H = -\sum_{i=1}^{n} p_i \log_2(p_i) \tag{6.2}$$

where p_i denotes the probability of reel value i occurring, for $i \in \{1, \ldots, 10\}$, and $n = 10$ is the number of possible reel values. The maximum possible entropy, assuming all 10 reel values are observed with equal probability (i.e., a uniform distribution), is given by:

$$H_{\max} = -\sum_{i=1}^{10} \frac{1}{10} \log_2\left(\frac{1}{10}\right) = \log_2(10) \approx 3.3219 \text{ bits}$$

More intuitively, entropy can be interpreted as the average number of yes/no questions required to determine a reel value, e.g., "Is the number ≤ 5?". When entropy is at its maximum, on average, 3.3219 yes/no questions are required to determine the

outcome of a single reel. Since each yes/no question represents 1 bit of information, 3.3219 bits of information are needed to determine a reel value. Lower entropy indicates a more deterministic outcome, meaning that less information is required: fewer questions and fewer bits are needed to determine the outcome. By comparing the entropy values across different experiments and against the RNG baseline, we can assess which experiments produce more or less predictable reel values.

6.1.3 Runs Test

The runs test is used to evaluate whether a sequence of outcomes occurs in a random order by analyzing the number of so-called runs (Corder et al., 2007). A run is defined as a consecutive sequence of outcomes on the same side of a reference point. For example, in the sequence 1,4,2,1,1, there are 3 runs relative to a reference point of 1.5: (1), (4,2), and (1,1). The test compares the number of runs that would be expected under the assumption of randomness with the number of runs actually observed, making it possible to determine whether a sequence shows evidence of non-random patterns.

In this study, we apply the runs test to the sequence of reel values concatenated across spins in order: Reel 1, Reel 2, Reel 3, then Reel 1, Reel 2, and so on, using the expected mean value (5.5) as a reference point. Too few runs suggest a tendency of reel values being clustered above or below 5.5, possibly due to repeated outcomes or skewness in the distribution. Too many runs indicate that the outcomes switch sides too frequently, suggesting that the LLM avoids repeating similar values.

The runs test is formulated as a hypothesis test:

- Null hypothesis (H_0): The sequence of outcomes occurs in random order.
- Alternative hypothesis (H_1): The sequence of outcomes does not occur in random order.

To apply the test, each reel value is first converted into "above" or "below" labels relative to 5.5, with n_1 representing the number of values above the mean and n_2 the number of values below. The expected number of runs, μ, and the standard deviation, σ, under the assumption of randomness, are given by:

$$\mu = \frac{2n_1n_2}{n_1 + n_2} + 1 \tag{6.3}$$

$$\sigma = \sqrt{\frac{2n_1n_2\,(2n_1n_2 - n_1 - n_2)}{(n_1 + n_2)^2(n_1 + n_2 - 1)}} \tag{6.4}$$

If the sequence of outcomes occurs in random order (H_0), the number of runs R is approximately normally distributed. Under a two-sided test at the chosen 5% significance level, about 95% of sequences are expected to have an R value within ± 1.96 standard deviations from μ. A continuity correction, h, is applied to improve accuracy, where $h = +0.5$ if $R < \mu$ and $h = -0.5$ if $R > \mu$, giving the z-score:

$$z = \frac{R + h - \mu}{\sigma} \tag{6.5}$$

A z-score within ± 1.96 standard deviations from the expected number of runs indicates that the sequence does not significantly deviate from randomness, and H_0 cannot be rejected. A z-score outside this range indicates a statistically significant deviation, suggesting that reel values may cluster or alternate systematically above and below 5.5, and H_0 is rejected.

6.1.4 Sequences

This metric analyzes reel value sequences across spins in terms of the total number of sequences for each experiment, the average sequence length per game, and the longest sequence of each game. The games are analyzed individually, so if there happens to be sequences spanning several games, these would not be found with this metric.

A sequence is defined as two or more numbers in a row that are either in ascending order or identical, with the order wrapping around from 10 to 1. Hence, the following spin pairs are all sequences of length 6:

- 7 8 9, 10 1 2
- 1 1 1, 2 3 4
- 1 1 1, 2 2 2

It is important to note that while wins and sequences overlap, since a winning spin is automatically a sequence in itself, the two are not identical, since there can be several wins within one sequence. For example, ”1 1 1, 6 9 2” and “1 1 1, 2 2 2” will be counted as one sequence each, but with different lengths (3 and 6) and with 1 and 2 wins, respectively.

As mentioned in Section 5.3, sequences seemed to be one of the LLM’s strategies for making fairly well-distributed numbers statistically. Hence, this metric highlights LLM behavior that may not be caught statistically with the previous metrics but may still not be random.

6.2 Metrics for Spins and Game Outcomes

The following metrics analyze the randomness of spins in terms of their diversity, frequency, reuse, behavioral patterns, and dependencies, as well as how this affects game outcomes. These metrics are used to answer the second research question: ”Are spins and game outcomes affected by personas, context, and game mode, and if so, how?”

6.2.1 Spin Diversity and Reuse

This metric analyzes the total number of unique spins and their average reuse frequency (i.e., how many times each unique spin occurs on average).

Table 6.2: *Expected distribution of spin usage counts across 1000 unique spin outcomes, assuming the spins are random and independent. Each count k represents how many times a specific spin appears in 2000 total draws. The table shows the expected number of unique spins that occur exactly k times, assuming a Poisson approximation with $\lambda = 2$. The distribution is truncated at $k = 10$, since higher counts are negligibly unlikely.*

Usage Count (k)	Expected Unique Spins
0	~135
1	~271
2	~271
3	~180
4	~90
5	~36
6	~12
7	~3
8	~1
9	~0.2
10	~0.04

The Poisson probability formula, which models independent random events over time (Goldie, 2018), is defined by:

$$P(k) = \frac{\lambda^k}{k!} e^{-\lambda}, \quad k = 0, 1, 2, \ldots, \tag{6.6}$$

where λ is the mean of the distribution. This can be used to approximate a distribution of the number of times that unique spins will appear.

Since there are 1000 possible unique spins and most experiments contain approximately 2000 spins in total (disregarding I_R_SC and F_R_SC), each possible unique spin has an expected mean (λ) of 2. Therefore, the expected probability that a given unique spin occurs zero times is:

$$P(0) = \frac{2^0}{0!} e^{-2} = 0.1353 = 13.5\% \tag{6.7}$$

This means that approximately 135 unique spins are not expected to occur in a sample of 2000 draws. Most other unique spins are expected to be used 1, 2, or 3 times, as seen in Table 6.2. Hence, if the data are truly uniformly random and independent, we expect to see approximately 865 unique spins at least once, with a low number of reuses for experiments with 2000 total spins.

6.2.2 Spin Frequency

This metric analyzes which unique spins occur most frequently. As mentioned in Section 6.2.1, unique spins are expected to be reused a low number of times, whereas a higher number of occurrences of the same spin is less likely to happen at an accelerat-

ing rate. As shown in Table 6.2, it is unlikely that the same spin appears more than 8 times.

The spin frequency is presented in two ways:

1. The two most used spins and their counts per experiment
2. The most frequent spin for each spin number per experiment

6.2.3 Repetition Patterns

This metric analyzes the percentage of repetitive spin patterns. Patterns occur naturally in random data, but during initial experimentation, the LLM seemed to use unnaturally high numbers of patterns when creating values.

The analysis focuses on four spin patterns, including the x-y-x pattern (e.g., 1 2 1 or 8 3 8), near wins (e.g., 1 1 2), reverse near wins (e.g., 1 2 2), and wins (e.g., 1 1 1). In addition, it compares the spin parity, distinguishing between spins with only even numbers (e.g., 2 4 6) and spins with only odd numbers (e.g., 1 3 5). The two categories (the four spin patterns and the spin parity) are not mutually exclusive: a winning spin, which has the win pattern, will also automatically count as an odd or even spin. However, the types are internally exclusive, i.e., a spin cannot be both a near win and a win.

6.2.4 Test of Independence

This metric serves to evaluate whether reel value pairs are independent of each other. It consists of the Pearson's Chi-Square (χ^2) Test of Independence, which tests for statistically significant dependencies, and Cramér's V, which quantifies the strength of such dependencies.

6.2.4.1 Pearson's Chi-Square (χ^2) Test of Independence

Pearson's χ^2 test of independence (hereafter simply χ^2 test) is a statistical hypothesis test used to determine whether two categorical attributes are significantly dependent on each other (Corder et al., 2007). For each experiment, the test is applied to all possible pairwise combinations of the three reels, i.e., (r_1, r_2), (r_1, r_3), and (r_2, r_3), comparing the observed counts of reel value pairs to the expected counts if the two reels in question were independent. The tested hypotheses are as follows:

- Null hypothesis (H_0): Reel values are independent. The value of one reel is not associated with the value of another reel within the same spin.
- Alternative hypothesis (H_1): Reel values are dependent. The value of one reel is associated with the value of another reel within the same spin.

With the observed frequencies of reel value pairs organized in a contingency table where the two dimensions, i and j, each represent the possible values $[1, 10]$ of the

two reels in question, the χ^2 statistic can be determined using the following definition (McHugh, 2013):

$$\chi^2 = \sum_i \sum_j \frac{(O_{i,j} - E_{i,j})^2}{E_{i,j}}, \tag{6.8}$$

where $O_{i,j}$ represents the *observed* frequency in cell i, j of the contingency table, and $E_{i,j}$ represents the *expected* frequency of the same cell in a corresponding contingency table of expected frequencies. The latter is calculated using the following formula:

$$E_{i,j} = \frac{M_i \cdot M_j}{N}, \tag{6.9}$$

where M_i and M_j are the row and column totals, respectively, of the observed-frequency contingency table, and N is the total number of observed reel pairs. In this thesis, the χ^2 test is considered reliable when at least 80% of the expected frequencies are ≥ 5. [1] If more than 20% fall below this threshold, the experiment in question is flagged.

Given the computed χ^2 value, the corresponding p-value is derived from the χ^2 distribution with $(r-1)(c-1)$ degrees of freedom, where r denotes the number of rows and c the number of columns in the contingency table after removing categories with marginal frequencies of zero. The p-value represents the probability of observing the obtained result under the assumption that the null hypothesis is true.

The χ^2 test is implemented using the Python `SciPy` library's `chi2_contingency` function, which, given a contingency table of observed frequencies, returns the χ^2 statistic, p-value, and the degrees of freedom (SciPy, 2008). First, we construct the contingency table of observed reel value pairs; then, we apply the `chi2_contingency` function to compute the test results.

The null hypothesis is then evaluated based on the resulting p-value. If it falls below a significance level of 5%, H_0 is rejected, indicating that the reel values are dependent. If the p-value is greater than 5%, H_0 cannot be rejected, meaning that there is no statistically significant evidence of dependence between the reels.

6.2.4.2 Cramér's V

Cramér's V effect size is a measure of the strength of dependence between two variables. It is calculated based on Pearson's χ^2 statistics and used to analyze how strongly the reels are dependent on each other. This is done for all experiments. For every χ^2

[1] Different guidelines exist for the reliability of χ^2 test. A commonly used rule is that at least 80% of expected frequencies should be ≥ 5 (McHugh, 2013; Kent State University Libraries, 2026), while stricter sources require this for all expected frequencies (Turney, 2022). Some sources omit such an assumption (Cote et al., 2024), while others even provide examples that violate it (Corder et al., 2007). As described in Section 6.2.4.1, the χ^2 test is implemented using the `SciPy` Python library. The documentation for `chi2_contingency` states that both observed and expected frequencies should be ≥ 5. However, this is not enforced by the implementation and is stated as a rule of thumb rather than a strict requirement. Since this differs from the statistical sources considered here, which only refer to expected frequencies, this is not used as a validity criterion.

test of independence, Cramér's V is computed using the following formula:

$$V = \sqrt{\frac{\chi^2}{n \cdot (L-1)}} \tag{6.10}$$

where χ^2 is the χ^2 statistic, n is the total number of observed reel pairs and L is the minimum of r and c.

The effect size ranges from 0 to 1. Values around 0.1 are considered small, 0.3 medium, and 0.5 or higher large effects (Corder et al., 2007).

In this thesis, Cramér's V is computed in order to compare the strength of dependence between reels across the different experiments.

6.2.5 Win Distribution

This metric analyzes how the outcome of the game, i.e., the number of wins, is affected by the context, game mode, and persona. This is done by examining whether some experiments have significantly more wins in total than other experiments and the RNG baseline, as well as whether the likelihood of winning varies across spin numbers.

7 Experimental Evaluation

This chapter presents the results obtained from the experiments conducted using the experimental setup described in Chapter 5. The findings are organized according to the research questions presented in Section 5.1 and are evaluated with respect to the metrics introduced in Chapter 6.

Remarks on LLM-Based Experiments

It should be noted that a few games in the full-game experiments became very long, with the longest game having 1702 valid spins. Since the prompts for full games ask for at most 50 spins, and to avoid skewing the results unfairly, only spin numbers 1 through 50 are included in the following analysis. A summary of the game lengths can be found in Appendix E.

Additionally, a few experiments have significantly fewer than the expected 2000 spins and may therefore exhibit different behavior for some of the metrics. I_R_SC has only 403 spins, as most games ended after 10 spins when all credits were used; F_R_SC has 1795 spins; and F_R_S has 1938 spins, since one game was returned as a Python script rather than actual spins. Furthermore, only valid spins have been included, resulting in the remaining experiments having 1984-2000 spins each. Table 7.1 shows the number of valid spins for each experiment with spin_num ≤ 50.

7.1 RQ1: Is the Randomness of Reel Values Affected by Personas, Context, and Game Mode, and if So, How?

This section presents the results related to RQ1, focusing on the randomness of reel values and how varying personas, contexts, and game modes affect these. The evaluation is conducted using the probability distribution of the reel values, as well as the metrics defined in Section 6.1, i.e., total variation distance, Shannon entropy, runs test, and sequences.

Table 7.1: *Valid spins with spin_num ≤ 50 per experiment.*

Experiment	I	F
*_N	1994	–
*_N_S	–	1984
*_R	1998	–
*_R_S	1999	1938
*_R_SC	403	1795
*_R_SP0	1998	1988
*_R_SP1	2000	1993
*_R_SP2	2000	1990
*_R_SP3	1999	1992
*_R_SP4	1999	1998
*_R_SP5	1999	1989
*_R_SP0U	2000	1997
*_R_SP1U	1998	1996
*_R_SP2U	1998	1992
*_R_SP3U	1999	1997
*_R_SP4U	2000	1997
*_R_SP5U	1999	1991

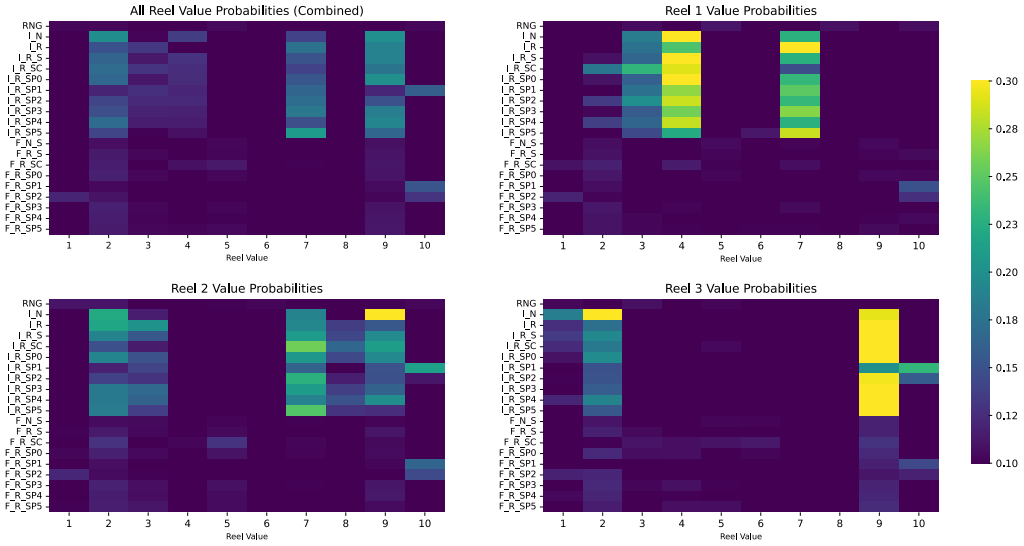

Figure 7.1: *Heatmap showing the probability distributions of values across all reels combined and for each individual reel. Note that the color scale is cropped at 0.1 and 0.3, respectively.*

7.1.1 Value Distribution

Value distributions were calculated as a prerequisite for the computation of the TVD and entropy metrics. As part of the analysis, the heatmaps in Figure 7.1 were generated to visualize observed patterns in the distributions.

For isolated-spin experiments, the heatmaps reveal that specific values are heavily overrepresented in all experiments, relative to the RNG baseline. Reel 1 favors values 3, 4, and 7. Reel 2 prefers values 2, 3, 7, 8, and 9. Reel 3 favors 2 and strongly overrepresents 9. Overall, there appears to be a tendency to avoid extreme and middle values

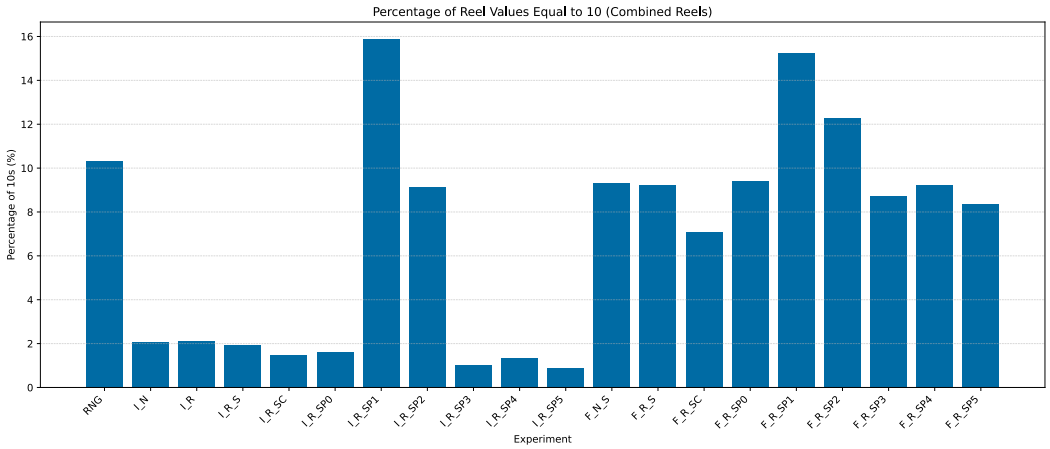

Figure 7.2: *The percentage of 10s in each experiment, across all three reels (combined).*

across all reels; a pattern that is consistent across experiments, though with slight variations.

In contrast, full-game experiments show only a minimal overrepresentation of specific values, with distributions being close to uniform. F_N_S (no context) appears slightly more uniform than the other experiments, though the differences are too small to draw any conclusions.

Across all experiments, those including personas with potential anchors appear to affect the value distributions the most. This is clearly visible in the heatmaps, where the value 10 occurs more frequently for *_R_SP1 and *_R_SP2, the two personas that include anchoring. This pattern is further supported by Figure 7.2, which shows the percentage of 10s across all three reels (combined) for each experiment. Notably, for the persona *_R_SP0 where 10 is unrelated to the context, i.e., that it is 10 o'clock, the percentage of 10s remains similar to those experiments without personas and for full games also comparable to the RNG baseline. This aligns with the findings from our initial experimentation. The effect of anchoring bias is significantly stronger in isolated spin experiments than for full-game experiments, though a weaker effect is still observable for full games. For persona *_R_SP2, which includes both 1 and 10 as anchors, the anchoring effect is split between the two values, although 10 appears slightly more than 1. Another interesting observation is that for isolated spin experiments, Reel 1 appears largely unaffected by anchoring. This observation indicates that it might not only be first-interaction prompts that are more deterministic, as Nath (2025) suggests, but that the first tokens in general are more deterministic and less influenced by the prompt.

Overall, the value distributions differ between game modes. Full-game experiments produce more uniform distributions, whereas isolated spin experiments are concentrated around specific values. Across experiments, value distributions vary slightly,

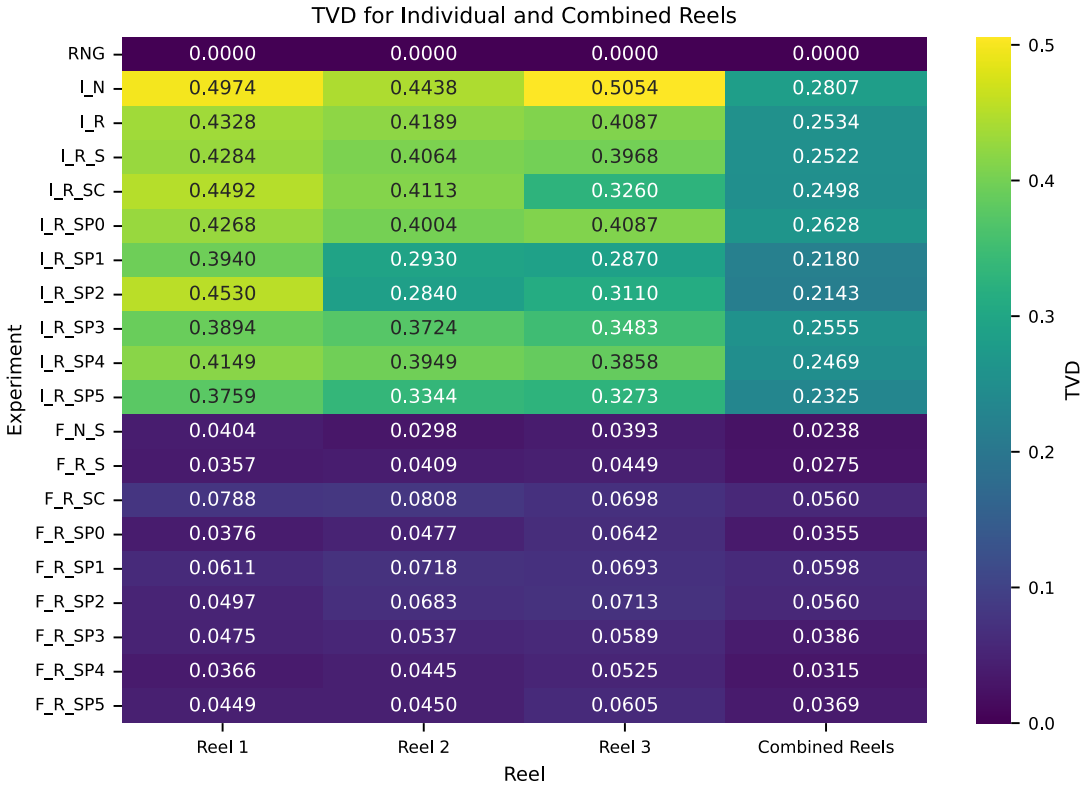

Figure 7.3: *TVD for individual and combined reels. The heatmap exhibits a clear distinction between isolated-spin and full-game experiments; while the former generally show higher values, the latter are much closer to resembling the RNG baseline.*

with experiments exhibiting anchoring bias showing the greatest deviation from the remaining experiments; however, these experiments still also overrepresent the same values observed in the other experiments. Within the experiments exhibiting anchoring, the effect is almost absent for Reel 1 in isolated-spin experiments, suggesting that the first value may be less affected by anchoring.

7.1.2 Total Variation Distance

The heatmap of TVD values in Figure 7.3 provides a compact comparison of how isolated-spin and full-game experiments deviate from the RNG-generated distribution across the three reels and their combined output. Table 7.2 further provides the corresponding minimum, maximum, and range statistics for a quick overview of the most extreme experiments in terms of TVD. TVD values close to 0 (purple/blue in the heatmap) indicate a near-RNG distribution, while values approaching the observed global maximum of 0.5054 (yellow/green in the heatmap) indicate bias. The RNG row exhibits TVD=0 across all reels, establishing a reference point for evaluating the different experiment conditions outlined in Section 5.4. The TVD results provide a clear distinction between isolated-spin and full-game experiments; while the former exhibit clear non-random bias, the latter produce consistently near-RNG reel-value distributions, though still with a few noteworthy remarks.

Table 7.2: *TVD statistics for individual and combined reels. Isolated-spin experiments consistently have greater minimum and maximum TVD values across reels, as well as a wider TVD range per reel and for combined reels than full-game experiments.*

	Isolated Spins (I)			Full Games (F)		
	Min	**Max**	**Range**	**Min**	**Max**	**Range**
Reel 1	0.3759 (I_R_SP5)	0.4974 (I_N)	0.1215	0.0357 (F_R_S)	0.0788 (F_R_SC)	0.0431
Reel 2	0.2840$^{\downarrow}$ (I_R_SP2)	0.4438 (I_N)	0.1598	0.0298$^{\downarrow}$ (F_N_S)	0.0808$^{\uparrow}$ (F_R_SC)	0.0510
Reel 3	0.2870 (I_R_SP1)	0.5054$^{\uparrow}$ (I_N)	0.2184	0.0393 (F_N_S)	0.0698 (F_R_SC)	0.0305
Combined Reels	0.2143 (I_R_SP2)	0.2807 (I_N)	0.0664	0.0238 (F_N_S)	0.0598 (F_R_SP1)	0.0360

$^{\downarrow}$ Column-wise minimum, i.e., minimum TVD value across all three reels.
$^{\uparrow}$ Column-wise maximum, i.e., maximum TVD value across all three reels.

As summarized in Table 7.2, the worst-performing isolated-spin experiment across individual and combined reels is I_N (no context), showing TVD values as high as 0.5054 for one of the individual reels (Reel 3) and 0.2807 for the combined reels. In other words, to make the distribution of this experiment and the RNG match, around 28% of the total probability mass would need to be redistributed for the combined reels. This emphasizes that they are notably different, despite the experiment prompt containing minimal contextual information. This further suggests that some inherent bias may be ascribed to the training data, as previous studies have also suggested, as described in Section 3.1.

Among the persona-containing isolated-spin experiments, I_R_SP1 (Sophie's lucky number 10) and I_R_SP2 (Sophie's lucky numbers 1 and 10) behave the most like the RNG baseline for Reels 2 and 3, as well as for combined reels, but even these experiments remain far from the RNG distribution. Notably, I_R_SP0 (Sophie playing at 10 o'clock), whose prompt also mentions the number 10, is the most similar to these three experiments' corresponding non-persona experiment, I_R_S, than I_R_SP1 and I_R_SP2. This indicates that when the number 10 is included in the persona description, it affects the outcome distribution differently, depending on the described context of that number. Lucky numbers (which can be considered game-related) decrease TVD, i.e, improve the randomness, while the time (which can be considered less game-related) causes no difference compared to its non-persona counterpart. Detailed experiment-wise TVD plots (Appendix F) show that what makes I_R_SP1 and I_R_SP2 stand out is their higher occurrences of the number 10, which indicates anchoring bias towards this.

As summarized in Table 7.2, isolated spin experiments range from 0.2840 to 0.5054 for individual reels. At a combined-reels level, the minimum and maximum values are 0.2143 and 0.2807, respectively. This demonstrates that considering combined reels

alone, instead of individual reels, cannot compensate for the inherent bias of isolated-spin experiments. Furthermore, the best combined-reel value for isolated spins (0.2143) is still much worse than the worst individual-reel value for full games (0.0808), emphasizing that the difference between game modes is unquestionable.

In contrast to isolated-spin experiments, full-game experiments show TVD values an order of magnitude lower, ranging from 0.0298 to 0.0808 for individual reels, and from 0.0238 to 0.0598 for combined reels. Combined reels-wise, the worst-performing experiment, F_R_SP1 (Sophie's lucky number 10), is followed by F_R_SC (explicit credits) and F_R_SP2 (Sophie's lucky numbers 1 and 10). Recalling that these two personas, P1 and P2, are the best-performing isolated-spin experiments at a combined reels-level, the diverging results across isolated spins and full games make the effect of the different personas inconclusive across the two experiment modes, when only considering the TVD summary in Figure 7.3; while P1 and P2 decrease the TVD for individual and combined reels for isolated spins, they do the directly opposite for full games, i.e., increase the TVD values.

To summarize, isolated-spin experiments show fundamental bias, with generally high reel-wise TVD values and relatively large TVD ranges. While these are both decreased at the combined-reels level, aggregating the reels' outcomes does not eliminate the non-random distribution of individual reels, as the bias persists for combined reels, though reduced. The contrast to full-game experiments is decisive; both TVD values and ranges are notably smaller for both individual and combined reels. Furthermore, the experiments behaving the least and most random for one game mode behave opposite for the other, making only intra-game mode results conclusive, leaving results across game modes inconclusive. However, an important nuance to this is that though the persona variants P1 and P2 appear to decrease TVD for isolated spins and increase TVD for full games (making the effect inconclusive in this sense), both game modes show what can be considered anchoring bias for these personas (hence, show conclusive results in this sense). Thus, game mode (I vs. F) acts as the greatest catalyst in terms of TVD, while context/role levels (N vs. R) and game parameters (S, C, and P[0-5]) result in more subtle, though still noteworthy and interesting, differences within the two game modes.

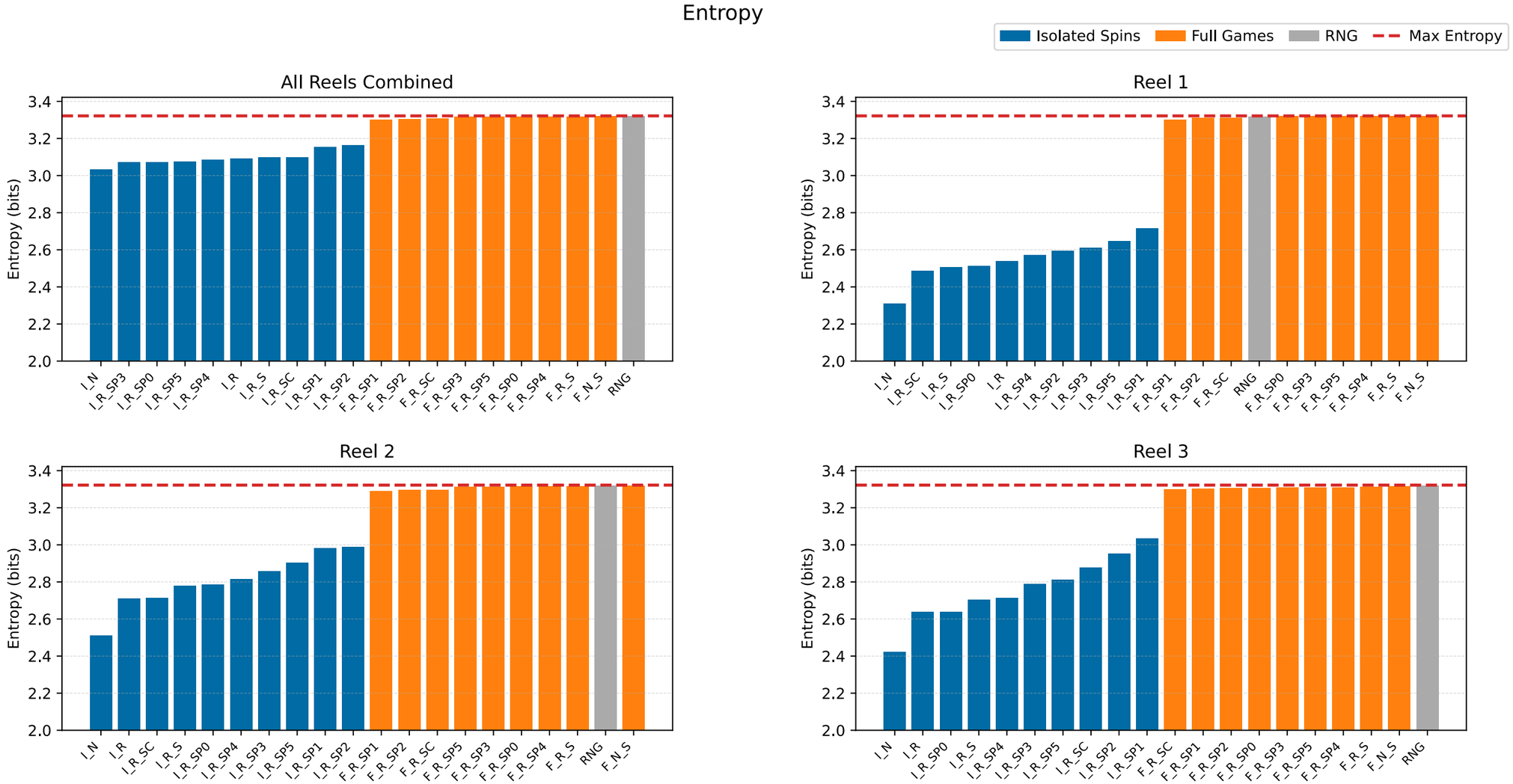

Figure 7.4: *Each plot shows the entropy of the value distribution across all combined and individual reels. The dashed red line indicates maximum entropy expected under a uniform distribution with 10 possible outcomes.*

7.1.3 Shannon Entropy

As shown in Figure 7.4, game mode has the largest effect on entropy. Full-game experiments consistently show entropy values approaching the maximum entropy expected for a uniform distribution with 10 possible outcomes, with some experiments even exceeding the RNG baseline, indicating close-to-maximal unpredictability both for each reel individually and across all three reels combined.

In contrast, isolated-spin experiments show more variation in entropy and are notably lower than full games, which is consistent with the observed value distribution showing that specific values are favored. Entropy also varies across reels for the isolated-spin experiments, with Reel 1 being slightly more predictable than the others.

Among isolated-spin experiments, the experiment prompted without the role of a one-armed bandit shows the lowest entropy, suggesting that context contributes to less unpredictability. This is in contrast to the full-game experiment F_N_S (no context), which performs the best across all reels. This suggests that having no context affects the game modes differently.

Personas also appear to influence entropy. In particular, the personas that include relevant anchors affect entropy in opposite ways for the two game modes. In the full-game experiments, F_R_SP1 and F_R_SP2 show the lowest entropy values, although these remain higher than those observed in all isolated-spin experiments. In contrast, I_R_SP1 and I_R_SP2 show the highest entropy values for isolated-spin experiments, except for Reel 1.

Overall, game mode appears to have a large effect on entropy, while personas and contexts have smaller but still observable influences.

7.1.4 Runs Test

The results of the runs test, applied to the combined sequence of all three reels, show that there is statistically significant evidence to reject the null hypothesis of randomness for all experiments other than the RNG baseline and the experiment F_R_SP4 (see Figure 7.5 and Table 7.3). These results indicate that most experiments exhibit significant deviations in the ordering of outcomes compared to what would be expected under the assumption of randomness. As shown in the figure, all isolated-spin experiments have positive z-scores that lie far from zero, while most full-game experiments have negative z-scores, except the experiment F_N_S (no context). Positive z-scores that deviate from the expected distribution indicate a higher number of runs than expected under the assumption of randomness, meaning that isolated-spin sequences seem to exhibit a more frequent alternation pattern above and below the reference point 5.5. In contrast, a negative z-score far from zero indicates a lower number of runs than expected under

Table 7.3: *The table presents the number of runs, z-scores, and corresponding Runs Test decisions for each experiment.*

Experiment	Runs	Z-Score	Decision
RNG	3039	0.99	Fail to reject H_0
I_N	4689	45.47	Rejected
I_R	4560	40.37	Rejected
I_R_S	4468	38.18	Rejected
I_R_SC	879	16.08	Rejected
I_R_SP0	4446	37.49	Rejected
I_R_SP1	4037	28.05	Rejected
I_R_SP2	4312	33.88	Rejected
I_R_SP3	4235	31.91	Rejected
I_R_SP4	4390	36.04	Rejected
I_R_SP5	3905	23.69	Rejected
F_N_S	3556	15.01	Rejected
F_R_S	2803	-2.70	Rejected
F_R_SC	1532	-31.54	Rejected
F_R_SP0	2849	-3.35	Rejected
F_R_SP1	2723	-6.73	Rejected
F_R_SP2	2580	-10.45	Rejected
F_R_SP3	2755	-5.95	Rejected
F_R_SP4	2929	-1.70	Fail to reject H_0
F_R_SP5	2740	-6.22	Rejected

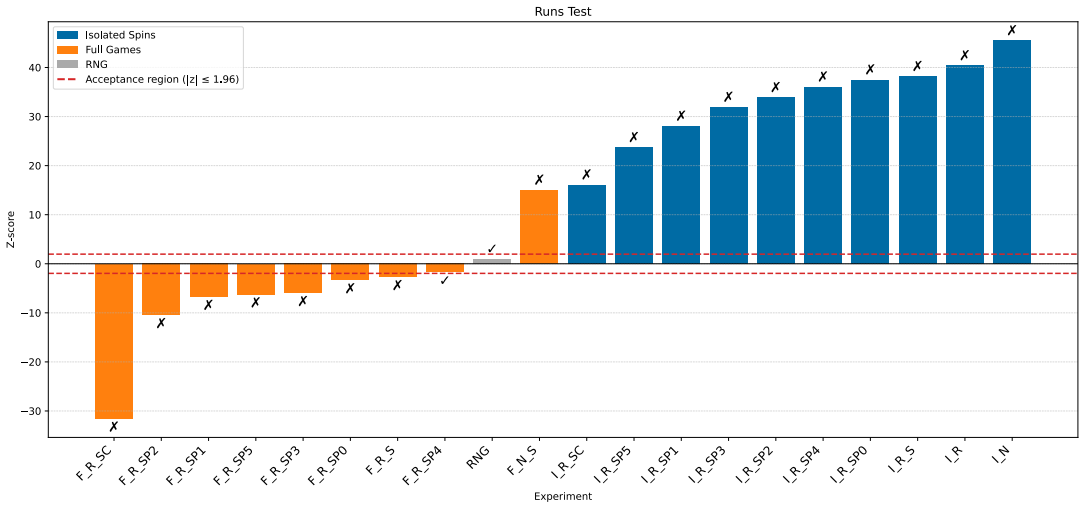

Figure 7.5: *The plot presents the z-scores for individual experiments together with their corresponding runs test outcomes (pass/fail).*

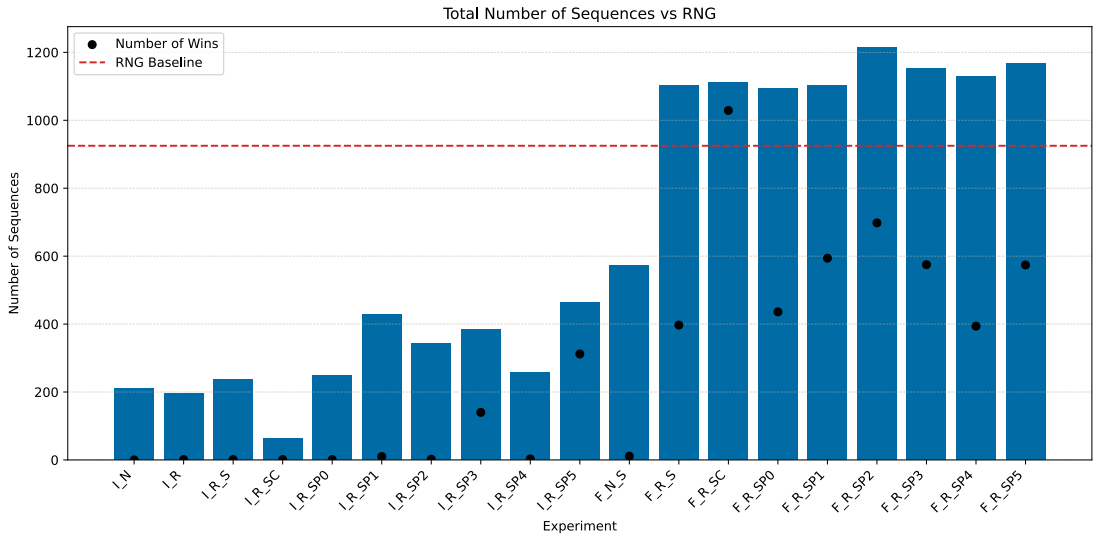

Figure 7.6: *The graph shows the total number of sequences per experiment in relation to the RNG baseline of 925 sequences. Wins are related to, but not directly comparable to, sequences, and have hence also been marked for each experiment with a dot.*

the assumption of the null hypothesis, meaning that most full-game experiments seem to follow a pattern where outcomes are too frequently clustered above or below 5.5.

Overall, game mode has the largest effect on run patterns, with isolated-spin experiments alternating too frequently and full-game experiments showing clustering of reel values. The only exception is the no context experiment, F_N_S, which behaves similarly to an isolated-spin experiment, indicating that context may influence run patterns.

7.1.5 Sequences

Total Number of Sequences Sequences, as defined in this thesis, occur naturally in random data. Our RNG experiment, which had 925 sequences in total, is used as a baseline in Figure 7.6, which shows the total number of sequences for each experiment. The number of wins has also been marked, since they are related to sequences, even if they are not directly comparable, as stated in Section 6.1.4.

As seen in the graph, there are clear differences between isolated-spin and full-game experiments. Isolated-spin experiments all have significantly fewer sequences than the baseline. This is expected for I_R_SC, which only has 403 spins in total, but it is also the case for all other isolated-spin experiments. However, there are some experiments that have more sequences than the rest of them, which are all persona experiments that, according to other metrics (see, e.g., Section 7.1.1 and Section 7.2.5), show biases, namely I_R_SP1, I_R_SP2, I_R_SP3, and I_R_SP5.

I_R_SP3 and I_R_SP5, which both have personas focused on winning, have a lot more wins than all the other isolated-spin experiments, which likely also contributes to their higher number of sequences. I_R_SP1 and I_R_SP2 both have personas with lucky numbers, but they do not have many wins, so the reason for their higher num-

bers of sequences is less straightforward. For isolated spins, a specific reel value is unlikely to appear twice in the same spin, as will be discussed further in Section 7.2.4, and this means that most of their sequences are either in ascending order or occur across two successive spins that have the same number as the last and first reel value. Since I_R_SP1 and I_R_SP2 use 10 more frequently due to anchoring, a reason for the higher number of sequences could be that there are more cases where 10 is used as the last number of a spin and again as the first number of the next spin. Either way, these observations indicate that personas do make a slight difference for sequences in isolated-spin experiments.

Full-game experiments, on the other hand, all have more sequences than expected, except for F_N_S (no context), which is closer to the isolated-spin experiments with fewer than 600 sequences. All the full-game experiments with the one-armed bandit as their role have unnaturally high numbers of wins. This could contribute to the generally higher numbers of sequences compared to isolated spins and F_N_S, which have very few wins. However, interestingly, the actual number of wins does not seem to affect the number of sequences much, since all the full-game experiments, except for F_N_S, have a similar total number of sequences, no matter their number of wins. For example, both F_R_S with fewer than 400 wins and F_R_SC with more than 1000 wins have about 1100 sequences. The results mean two things. Firstly, while wins themselves are sequences, other types of sequences are also prevalent in full games, much more than for isolated spins. This suggests that the LLM uses sequences as a strategy to generate values for full games. Secondly, it indicates that the context makes a difference in the number of sequences, since all experiments with the one-armed bandit as their role produce many more sequences than the experiment without context, i.e., F_N_S.

Average Sequence Length and Longest Sequences Not only the total number of sequences, but also their lengths are important.

Figure 7.7 shows distributions of the average sequence length per game. These averages further widen the gap between isolated spins and full games. Not only do full-game experiments with the one-armed bandit role have a lot more sequences than isolated spins, but the sequences are also much longer on average.

The average sequence length per game for the RNG baseline ranges between roughly 2 and 2.5. This means that most sequences are short, but there are some wins and/or slightly longer sequences as well.

Most of the isolated-spin experiments have even shorter average lengths, with most of them being 2 numbers long. The only outliers are I_R_SP3 and I_R_SP5, which have an overall experiment average of 2.5 and 3, respectively. For I_R_SP5, half of the game averages are over 3, which means that sequences are not solely isolated wins, but rather sequences that are made up of other patterns or win streaks. These two experiments show that having more wins is a determining factor for the average sequence lengths for isolated spins. Hence, certain personas do have an effect on the sequence lengths.

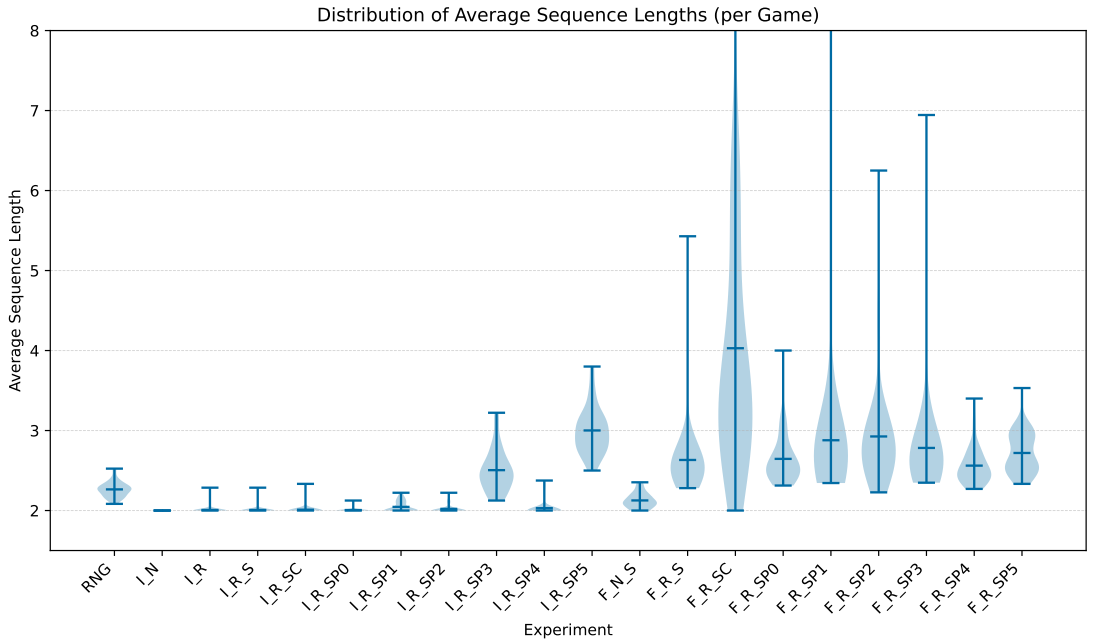

Figure 7.7: *The violin chart shows the distributions of average sequence lengths per game for different experiments. The y-axis has been capped at 8 for better visibility, but the highest game average is 11.54 for F_R_SC and 9.07 for F_R_SP1. The experiment min, mean, and max have been marked on each violin with a blue line.*

For full games, F_N_S (no context) is again different from the rest of the experiments, with a distribution that is the closest to that of the RNG experiment. The rest of the full-game experiments all have longer sequences on average, with most experiment averages being between 2.5 and 3, and most game averages going from slightly more than 2 to 4. However, some experiments have especially long sequences. The experiment average of F_R_SC is 4, while the highest average of a single game in this experiment is 11.54. Even excluding this game, the distribution shows that the game averages are higher for F_R_SC in general. This suggests that the sequences of F_R_SC are not just a result of many isolated wins, but that there must be either sequences of win streaks or other types of long sequences. This is not seen to the same extent for any of the other experiments, though F_R_SP1 does have a single game with an average length of 9.1.

These observations are further supported by Figure 7.8, which shows the distribution of the longest sequence of each game. Again, the same trends are observed for isolated-spin experiments. Most experiments have games with a maximum sequence length of 2, and possibly a single or a few games with longer maximum sequences. The only outliers are I_R_SP1 with a distribution of slightly higher maximum lengths, I_R_SP3, where all games have a maximum sequence length of 3 or longer, and I_R_SP5, which have notably longer maximum sequences than any other isolated-spin experiment. Consequently, the longest sequence of any isolated-spin experiment belongs to I_R_SP5. This is found in game 10 from spins 10-15 and looks like this: (2 9

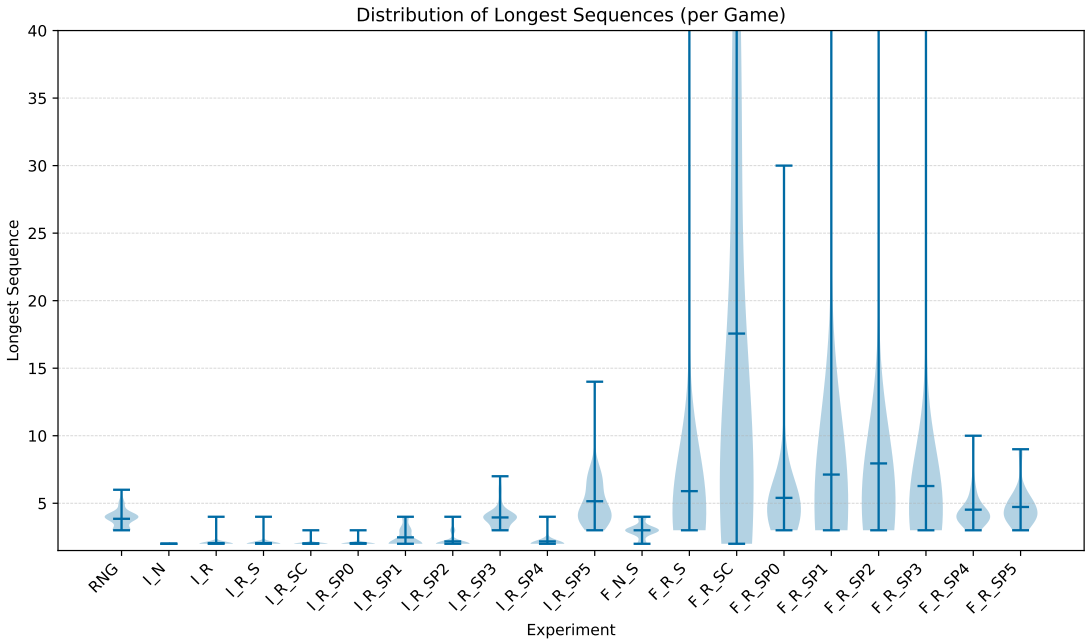

Figure 7.8: *The violin chart shows the distributions of the longest sequence from each game for different experiments. The y-axis has been capped at 40 for better visibility, but the longest sequences are 63 numbers for F_R_S, 69 numbers for F_R_SC, 90 numbers for F_R_SP1, 62 numbers for F_R_SP2, and 60 numbers for F_R_SP3. The experiment min, mean, and max have been marked on each violin with a blue line.*

6, 7 7 7, 7 7 7, 7 7 7, 8 8 8, 8 3 9). It consists mainly of wins with only two flanking individual reel values, i.e., 6 and 8. Three of the wins, and thereby most of the sequence, are "7 7 7", which is also the most common spin overall in this experiment, as described in Section 7.2.1, with approximately 150 occurrences. This, combined with the fact that the two isolated-spin experiments with notably more wins are also the ones with notably longer sequences, suggests that wins and determinism play key roles for sequences in isolated-spin experiments. This makes sense, since the LLM has no history of what was produced before each spin and therefore cannot create sequences across spins intentionally, unlike full-game experiments as described below.

Full-game experiments generally have longer maximum sequences. The outliers are F_N_S, F_R_SP4, and F_R_SP5, which all have shorter maximum sequences than I_R_SP5. Especially F_N_S has short maximum sequences, even shorter than those of the RNG experiment. In contrast, F_R_SC have very long maximum sequences, not just for a single game, but in general. Despite this, the longest sequence overall is seen in F_R_SP1, game 8, spins 22-50 with a length of 90: (1 1 1, 2 2 2, 3 3 3, 4 5 6, 7 8 9, 10 10 10, 1 2 3, 4 5 6, 7 8 9, 10 10 10, 1 1 1, 2 2 2, 3 3 3, 4 4 4, 5 5 5, 6 6 6, 7 7 7, 8 8 8, 9 9 9, 10 10 10, 1 2 3, 4 5 6, 7 8 9, 10 10 10, 1 1 1, 2 2 2, 3 3 3, 4 4 4, 5 5 5, 6 6 6). This sequence makes it very clear that the LLM used certain patterns to generate values, since a mix of ascending numbers and ascending wins is seen. This corresponds to the behavior observed during the initial experimentation.

Since especially F_N_S (no context) and F_R_SC (explicit credits), which both have different contexts than the rest of the experiments, are outliers in terms of sequence lengths, this supports that context does affect the randomness for full games.

Overall, the number of sequences and their lengths do seem to be affected by the game mode, context, and personas. Game mode has the biggest effect, while personas affect isolated-spin experiments more, and context affects full-game experiments more.

7.1.6 Subconclusions for RQ1

For all metrics, there is a clear difference between game modes.

Isolated-spin experiments clearly appear more deterministic, showing a stronger tendency towards outputting specific values. This is consistent with findings reported by Nath (2025), which suggest that first-interaction prompts may induce more deterministic behavior. However, it is important to note that the entropy for Reel 1 across all isolated-spin experiments suggests that it is more deterministic than the following two reels. Furthermore, the TVD results show that the value distribution of Reel 1 is, with the exception of I_N, consistently further from the RNG baseline than those of Reel 2 and Reel 3, indicating that it is less random. Reel 1 also appears largely unaffected by anchoring in terms of value distribution. Taken together, these findings may suggest that the increased determinism observed in isolated-spin experiments is concentrated in the earliest part of the response.

In contrast, full games exhibit a near-uniform value distribution, and TVD and entropy values close to, or in some experiments exceeding, that of the RNG. This could suggest that, for full games, the LLM produces outputs close to random. However, the runs test and sequences show that while the distribution of values is close to what would be expected under randomness with equal probability, the order in which the values occur is not at all random. For full games, the runs test indicates patterns suggesting that the values tend to cluster above or below the reference point at 5.5. A similar issue is seen for sequence lengths, where unnaturally long sequences appear. Thus, while full games seem more random according to some metrics, this is not supported when ordering is considered. Furthermore, since both full games and isolated spins are first-interaction prompts, but behave differently in terms of determinism, this suggests that it is not first-interaction prompts as a whole that are more deterministic, as Nath (2025) suggests, but rather the first tokens of the response.

Overall, the results show that randomness is affected differently, depending on the interaction mode. Isolated-spin experiments show determinism in value selection, whereas full games instead exhibit deterministic behavior shown in the temporal sequence of values, while maintaining an overall near-uniform distribution. This may be explained either by the LLM trying to approximate randomness as instructed, resulting in a uniform overall distribution, or by cyclic patterns in the generation process.

While game modes have the largest overall effect on randomness of reel values, personas and context still have a measurable effect.

For the isolated-spin experiments, the value distribution, and thus TVD and entropy, is notably influenced by the personas, particularly those designed to induce anchoring bias. This shows that prompting can systematically bias or manipulate the distribution of the generated output and thereby affect the overall quality of randomness. This is consistent with prior work, which demonstrates that LLM outputs are highly sensitive to anchoring introduced through prompting when considering other reasoning tasks (Lou et al., 2025). A similar effect can also be observed in full-game experiments, though the effect is considerably weaker, suggesting that the influence of prompting is reduced more in setups where conversational history about previous values is carried across outputs. Furthermore, for both entropy and TVD, the game-related anchors have opposing effects for the two game modes. For isolated-spin experiments, which have a lower occurrence of 10s for other experiments, it improves the randomness, whereas for full-game experiments, which normally have a much more uniform distribution, it makes the randomness quality worse.

Finally, the context also appeared to have a measurable effect. The no context experiment F_N_S behaved much more like an isolated-spin experiment, both in terms of runs and total number and lengths of sequences, and the F_R_SC experiment (explicit credits) generally had much longer sequences and a much higher z-score for runs than any other experiment. For TVD and entropy, the influence of context also appeared to differ between the two game modes. I_N showed the lowest entropy, i.e., the most deterministic behavior, whereas F_N_S sometimes achieved entropy closer to the uniform distribution than the RNG baseline and also had the lowest TVD values among all experiments.

7.2 RQ2: Are Spins and Game Outcomes Affected by Personas, Context, and Game Mode, and if So, How?

Having evaluated the randomness of reel values to answer RQ1 in the previous Section 7.1, this section presents the results related to RQ2, focusing on spins and game outcomes, and how varying personas, contexts, and game modes affect these. The evaluation is conducted using the metrics defined in Section 6.2, i.e., spin diversity and reuse, spin frequency, repetition patterns, test of independence, and win distribution.

7.2.1 Spin Diversity and Reuse

Figure 7.9, which shows the number of total unique spins and the average reuse frequency, clearly shows that the game mode and, to a lesser extent, the context affect the number of total unique spins, and how much these are reused.

The RNG experiment behaves as expected for random and independent numbers. Based on the Poisson approximation, the expected number of total unique spins is 865, while the actual total for the RNG is 862, which is very close. This also leads to few reuses of the unique spins, with the average usage count being less than 2.5. Because

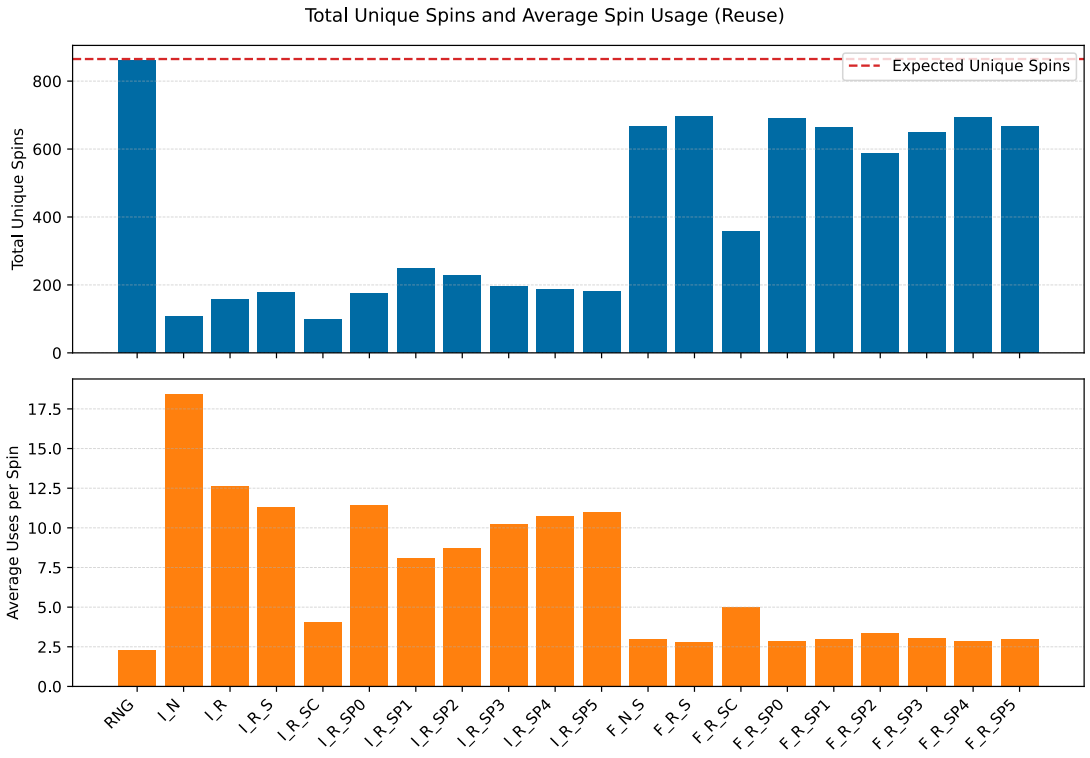

Figure 7.9: *The blue graph shows the total number of unique spins per experiment, with the expected number of unique spins according to the Poisson distribution marked with a dashed line. The orange graph shows how many times each unique spin is used on average.*

of this, it can be concluded that the RNG baseline seems random and independent, as it should be.

The isolated-spin experiments generally perform poorly for this metric, with I_N (no context) and I_R_SC (context with explicit credits) having the lowest number of unique spins at about 100 each. This is no surprise for I_R_SC, which only has 403 spins in total and is expected to have much fewer unique spins, i.e., approximately 330 based on the Poisson model. Due to its low number of total spins, this experiment is not directly comparable to the others. However, since its number of unique spins is similar to that of other isolated-spin experiments even though it has a low number of total spins, it behaves closer to what is expected than other isolated-spin experiments. In contrast, I_N appears much more deterministic than the isolated-spin experiments with the simple one-armed bandit context, even though they all have approximately 2000 total spins each. Since the expected number of reuses is exponentially less likely for higher counts, having an average usage of more than 17.5 is extremely unlikely for uniformly random and independent numbers. The experiments with a context perform better, though they are still not random and independent with much too few unique spins and high average usage counts between 7.5 and 12.5.

The full-game experiments all perform more randomly, with most of them being similar to the RNG baseline in average usage count, although they have fewer unique

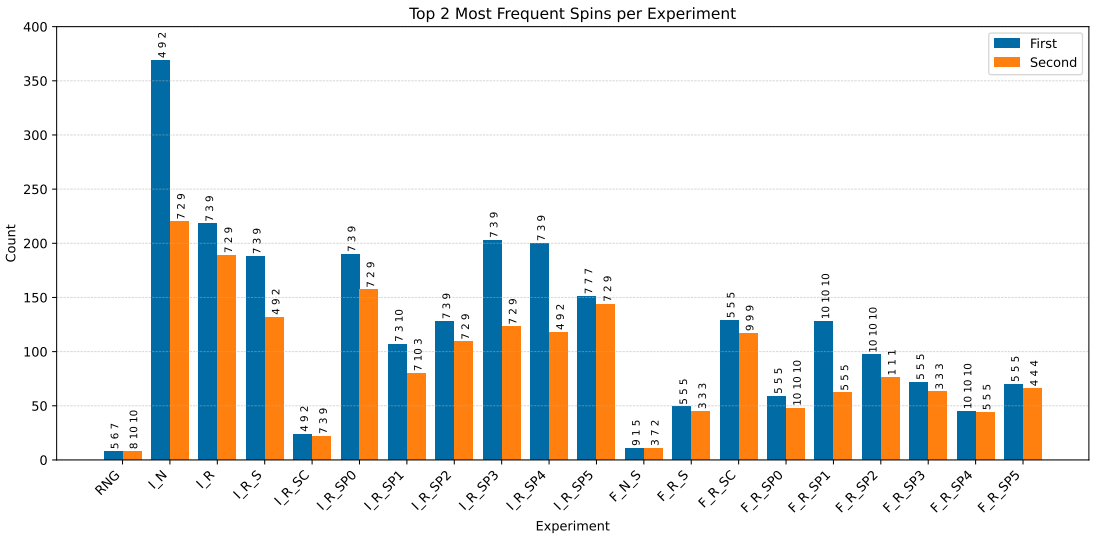

Figure 7.10: *The graph shows the top 2 most frequently used spins and their counts per experiment.*

spins than expected. Like for other metrics, the context with credits performs differently from the other experiments. In this case, it has a higher average usage count than the other experiments. F_R_SC has 1796 total spins, which means that based on the Poisson model, approximately 834 unique spins are expected. Taking this into account, F_R_SC performs much more deterministically than the other full-game experiments with its 359 unique spins, compared to 587-695 unique spins in all other full-game experiments. As further described in Section 7.2.4 and Section 7.2.5, this is due to an unusually high reuse of winning spins, which make up only 10 of the 1000 possible unique spins.

Overall, there is no significant difference between the personas. However, from the above analysis and the graph, it is clear that the game mode makes a big difference, with full games performing closer to the RNG baseline. Additionally, context affects the randomness: having no context for isolated spins is the most deterministic of all experiments, and both experiments with credits perform distinctly from other experiments. Hence, the results indicate that the game mode and the context affect the randomness in terms of unique spins and reuse.

7.2.2 Spin Frequency

7.2.2.1 Top 2 Most Frequent Spins

Extending the results in Section 7.2.1, Figure 7.10 shows which two unique spins have been used the most, as well as their frequency per experiment.

Consistent with the previous results, the RNG baseline behaves as expected. The two most frequent unique spins, as shown in the graph, were both used 8 times, while all other unique spins appear 7 or fewer times. The Poisson model estimated that approximately one unique spin would appear 8 times, so this result is slightly higher than

expected, but still within reasonable variation.

This is not the case for the other experiments, which have extremely high and unrealistic frequency counts, especially for isolated spins. As established in Table 6.2, it is very unlikely that any unique spin will be used more than 10 times. However, all experiments, except for the RNG baseline, have at least two unique spins that are used more than 10 times.

F_N_S performs the best with only two unique spins that were used 11 times each, while the rest of the unique spins were used 9 or fewer times. While this may not seem too extreme, the Poisson model estimates that roughly 0.0069 unique spins will appear 11 times, so even observing one unique spin that is used 11 times is already highly unlikely. If the Poisson formula is used again, now with $\lambda = 0.0069$ as the expected count, we find that the probability of exactly two unique spins appearing 11 times is:

$$P(2) = \frac{0.0069^2}{2!} e^{-0.0069} = 0.000023641 = 0.0024\% \tag{7.1}$$

This corresponds to roughly 1 in 41,000 experiments, which indicates that even our best-performing experiment is extremely unlikely to happen.

As shown in the graph, most experiments perform much worse than F_N_S, with I_N performing the worst, with 369 uses of the spin "4 9 2" out of 1994 total spins. The probability of getting this for random and independent numbers is effectively zero.

Interestingly, this means that the experiments without any context constitute the extremes, with F_N_S performing the most randomly and I_N performing the most deterministically. This is likely because F_N_S has a better "idea" of which numbers have been used within each separate game, while I_N does not have any history of which other outcomes have been used and therefore cannot compensate.

Similar to the spin diversity and reuse seen in Section 7.2.1, full-game experiments generally perform better than isolated-spin experiments in terms of frequency.

The spin labels of the most frequent spins in Figure 7.10 further show that there are differences between the preferred spins, depending on the game mode, context, and persona.

For all full-game experiments, except for F_N_S (no context), the most frequent spins are all win lines. This is also the case for the top spin of I_R_SP5 (Sophie's grandma), which has significantly more wins than the other isolated-spin experiments.

Furthermore, some anchoring bias is seen in I_R_SP1 and F_R_SP1 for number 10 and in F_R_SP2 for both 1 and 10. The spin "10 10 10" is also a common spin for F_R_SP0 and F_R_SP4, but since the winning spins are more evenly distributed for these experiments, this is less significant.

All isolated-spin experiments, except for I_R_SC, and I_R_SP3 and I_R_SP5 with clear persona-induced biases, have a combination of "7 3 9", "7 2 9", and "4 9 2" as their two most frequently used spins. The experiments with only the one-armed bandit as the context (i.e., not I_N and I_R_SC), and which do not show persona bias, all have "7 3 9" as their most used spin. On the other hand, the experiment with no context shows

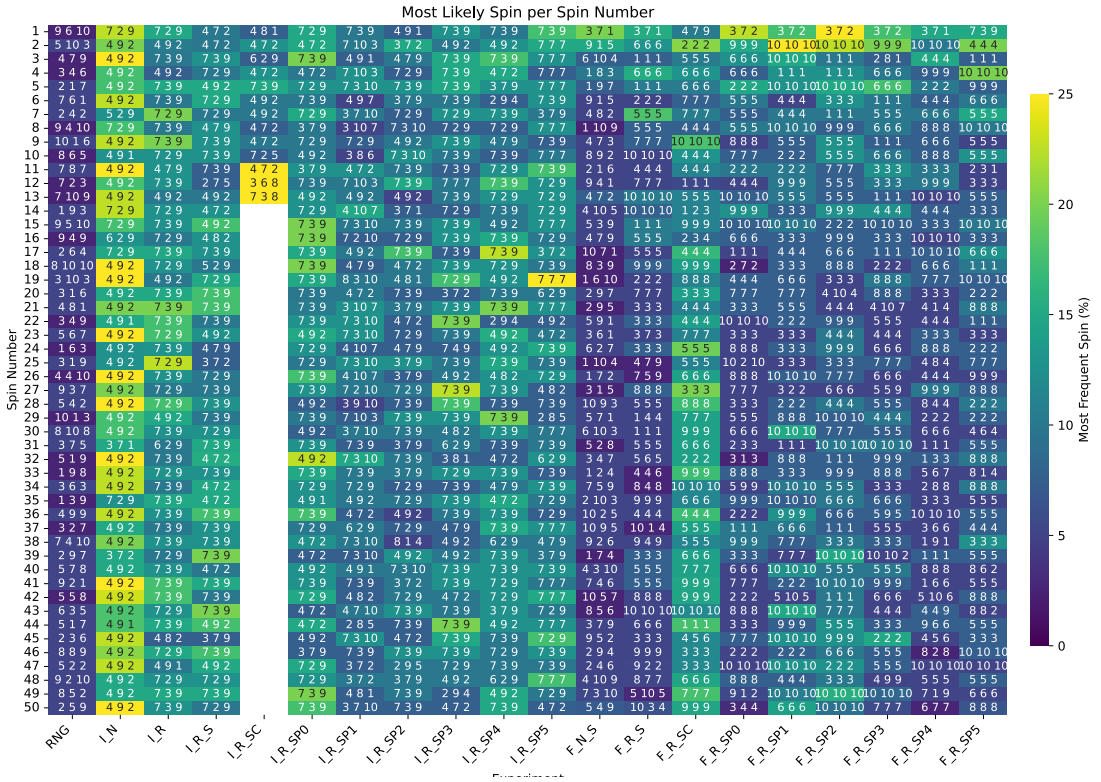

Figure 7.11: *The heatmap shows the most frequent spin for each spin number in percentage. Note that the percentage scale has been capped at 25% to better show trends. Since experiments have 40 games, 25% is equal to 10 spins for most cells. However, spins 11, 12, and 13 in I_R_SC are outliers, as they occur in only a single game and therefore appear as 100% for those cells, which distorts their representation. No games of I_R_SC continue past spin 13.*

a strong preference for the spin ”4 9 2”, which occurs at almost twice the frequency or more of all other top spins. I_R_SC also prefers this spin. Like for previous metrics, these results suggest that contexts and personas affect the spins differently.

7.2.2.2 Most Frequent Spin per Spin Number

The heatmap of the most frequent spin per spin number in Figure 7.11 further supports the above findings, as it shows the same trends for the preferred spin labels. The RNG has a very diverse selection of spins that are used sparingly, I_N heavily uses the same preferred spins much more frequently than all other experiments, F_N_S performs the most randomly of all experiments except for the RNG, and full games perform better than isolated-spin experiments.

The heatmap also shows some additional findings. Firstly, full games use the spin ”3 7 2” as the first spin in 4 out of 9 experiments and ”3 7 1” in 3 out of 9 experiments. The only two experiments that use other spins as the first spin are F_R_SC and F_R_SP5, which are both prompts with a focus on winning. Despite this focus, none of the most frequent first spins in any of the experiments are winning spins. This is in contrast to most other cells for full games, which have winning spins as their most frequent

spin. It is also important to note that most of the first spins of full games have more concentrated frequencies than other spins of the game, meaning that they are more deterministic. This, once again, supports that it may not be first-interaction prompts as a whole that are more deterministic, as suggested by (Nath, 2025), but rather the first few output tokens of a session.

Secondly, an interesting pattern emerges for F_R_SC, which is the experiment with the most wins, as described in Section 7.2.4 and Section 7.2.5. The heatmap shows that spins in the latter half of the game are more likely to be wins than spins in the first half. Hence, a player will get more wins if they keep playing.

Thirdly, there are some interesting outliers. Even disregarding spins 11, 12, and 13 of I_R_SC, since there is only one game with these spins, and I_N, which generally have a high reuse frequency, there are a few cells with very high frequencies. One of those is spin 19 in I_R_SP5, where about 25% of all 40 spins are ”7 7 7”. While this is a prevalent spin for this experiment in general, this cell stands out. It is difficult to say why this happens, but it could signify that the stats, such as the spin number, perhaps make a difference in whether this spin becomes a win or not.

Moreover, spin 2 of F_R_SP1, with the spin label ”10 10 10”, stands out with a much higher frequency than any other cell of this experiment. Once again, it is difficult to know exactly why, but it could be due to anchoring bias, since the number 10 is prevalent in the top spins for this experiment.

Finally, there are once again clear persona-induced biases for both the lucky number 10 and the prompt with Sophie’s grandma. This is especially true for I_R_SP1 and I_R_SP5 for the isolated spins, but also for F_R_SP1 and F_R_SP2 for full games. However, as shown for previous metrics, persona-induced biases and preferred spins are more prevalent in the isolated-spin experiments, which is likely because the full-game experiments compensate based on game history.

All of the above observations for the two most frequent spins overall and per spin number indicate that the context, game mode, and personas all affect preferred spins and their frequencies.

7.2.3 Test of Independence

For all experiments and across all combinations of reel pairs, except for the RNG baseline, the χ^2 test shows statistically significant evidence to reject the null hypothesis, suggesting a dependence between reel values. It is important to note that for all isolated-spin experiments, the reliability of the test results can be questioned, as the expected frequencies do not meet the assumption that at least 80% of cells should have expected frequencies ≥ 5. However, these results have still been included, as the violation of the expected frequencies reflects the underlying skewness in the data and may itself provide useful insight into deviations from randomness. As an example, the observed and expected frequencies for experiment I_N are shown in Table 7.4. The frequencies clearly show that several combinations of reel pair values are not represented at all. For

Table 7.4: *Observed vs. expected contingency tables for experiment I_N. Rows represent outcomes of Reel 1 (x), columns represent outcomes of Reel 2 (y). The observed table shows frequencies for each outcome pair (x, y), while the expected table shows frequencies under the assumption of independence between the two reels. For comparison, the corresponding table for the RNG baseline can be found in Table 7.5.*

Observed

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	1	0	0	0	0	0	0
2	0	0	0	0	5	1	19	15	49	0
3	0	0	0	0	0	7	228	104	24	1
4	20	16	0	0	1	3	129	65	525	8
5	23	92	9	0	0	0	2	11	7	0
6	36	81	33	1	0	0	0	0	0	0
7	22	247	179	6	0	0	0	0	0	0
8	5	5	11	0	1	0	0	0	0	0
9	1	0	0	0	0	0	0	0	0	0
10	1	0	0	0	0	0	0	0	0	0

Expected

	1	2	3	4	5	6	7	8	9	10
1	0.05	0.22	0.12	0.00	0.00	0.01	0.19	0.10	0.30	0.00
2	4.82	19.68	10.36	0.36	0.31	0.49	16.87	8.70	27.00	0.40
3	19.72	80.50	42.35	1.46	1.28	2.01	69.00	35.60	110.44	1.64
4	41.54	169.63	89.24	3.08	2.69	4.23	145.40	75.01	232.72	3.46
5	7.80	31.85	16.75	0.58	0.51	0.79	27.30	14.08	43.69	0.65
6	8.18	33.40	17.57	0.61	0.53	0.83	28.62	14.77	45.81	0.68
7	24.59	100.41	52.82	1.82	1.59	2.50	86.06	44.40	137.75	2.05
8	1.19	4.87	2.56	0.09	0.08	0.12	4.17	2.15	6.68	0.10
9	0.05	0.22	0.12	0.00	0.00	0.01	0.19	0.10	0.30	0.00
10	0.05	0.22	0.12	0.00	0.00	0.01	0.19	0.10	0.30	0.00

Table 7.5: *Observed vs. expected contingency tables for the RNG experiment.*

Observed

	1	2	3	4	5	6	7	8	9	10
1	20	19	18	24	21	19	14	19	20	15
2	20	20	25	19	20	30	14	14	19	18
3	29	28	14	16	22	22	23	15	16	26
4	16	27	19	18	21	11	16	20	26	19
5	26	17	20	16	26	25	23	21	28	20
6	11	22	18	21	18	18	16	17	20	15
7	23	24	12	19	18	16	22	20	17	20
8	27	22	12	22	20	16	26	21	22	29
9	26	22	15	15	15	27	18	20	10	20
10	23	20	17	25	21	22	17	20	24	25

Expected

	1	2	3	4	5	6	7	8	9	10
1	20.88	20.88	16.07	18.43	19.09	19.47	17.86	17.67	19.09	19.56
2	21.99	21.99	16.92	19.40	20.10	20.50	18.81	18.61	20.10	20.60
3	23.32	23.32	17.93	20.57	21.31	21.73	19.94	19.73	21.31	21.84
4	21.33	21.33	16.40	18.82	19.49	19.88	18.24	18.05	19.49	19.98
5	24.53	24.53	18.87	21.64	22.42	22.87	20.98	20.76	22.42	22.98
6	19.45	19.45	14.96	17.16	17.78	18.13	16.63	16.46	17.78	18.22
7	21.11	21.11	16.24	18.62	19.29	19.67	18.05	17.86	19.29	19.77
8	23.98	23.98	18.44	21.16	21.92	22.35	20.51	20.29	21.92	22.46
9	20.77	20.77	15.98	18.33	18.99	19.36	17.77	17.58	18.99	19.46
10	23.65	23.65	18.19	20.86	21.61	22.04	20.22	20.01	21.61	22.15

comparison, the observed and expected contingency tables for the RNG baseline are presented in Table 7.5, illustrating the frequencies observed and expected when values are approximately uniformly distributed and independent.

For the Cramér's V measure, the results are largely similar across game modes and personas, showing no substantial differences in the strength of association between reels. The Cramér's V values indicate a low to moderate association between reels, ranging between approximately 0.24 and 0.48 for almost all experiments. In contrast, the RNG baseline shows very weak associations, with values ranging between approximately 0.06 and 0.07. However, two experiments are outliers: F_N_S (no context) shows a weaker association, ranging approximately between 0.12 and 0.16 across reel pair combinations, while F_R_SC (explicit credits) shows a stronger dependence, ranging approximately between 0.59 and 0.64.

The full results of both the χ^2 tests and Cramér's V measures for all reel combinations (r_1, r_2), (r_1, r_3), and (r_2, r_3) across all experiments are provided in Appendix G.

Overall, these results suggest that there is a statistically significant dependence between reels across all experiments. No clear or consistent pattern is observed across personas or context for isolated-spin experiments. However, in full-game experiments, contexts appear to influence the strength of the dependence between reels. Finally, the contingency tables generated during the calculation of the χ^2 values clearly show that some pairs of reel values never appear, while others are notably overrepresented.

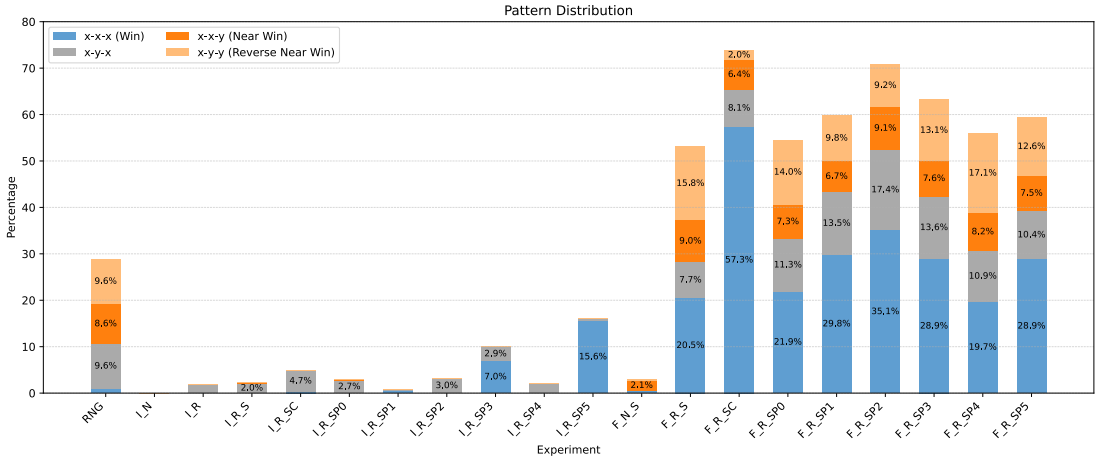

Figure 7.12: *The graph shows the percentage of spins that follow specific patterns, namely near wins, reverse near wins, the x-y-x pattern, and wins.*

7.2.4 Repetition Patterns

7.2.4.1 Spin Patterns

Figure 7.12, which shows the percentage of spins that follow specific patterns, clearly shows differences in patterns across game modes, contexts, and personas.

Under the assumption of uniform randomness, we expect a similar number of near wins, reverse near wins, and the x-y-x pattern, while wins should occur much less frequently. This is because wins have a probability of $10/1000 = 1\%$, whereas the other patterns have a probability of $90/1000 = 9\%$.

The result of the RNG baseline closely resembles this with reasonable variation. Near wins, reverse near wins, and the x-y-x pattern all make up roughly 9% each, while wins make up about 1% of all spins. Hence, these patterns account for just under 30% of all spins collectively as expected.

None of the other experiments is close to the expected distribution. The isolated-spin experiments have a very low percentage of patterns with almost no wins, near wins, and reverse near wins. This excludes I_R_SP3 and I_R_SP5, which have personas that focus on wins and show a much higher prevalence of wins than all other isolated-spin experiments. The only other somewhat prevalent pattern in any of the isolated-spin experiments is the x-y-x pattern, but even this is much lower than the expected 9%. This indicates that personas have an effect on patterns. Furthermore, the context also seems to affect the pattern, since I_N has no patterns at all, while I_R_SC has the highest percentage of the x-y-x pattern.

On the other hand, for full-game experiments, spin patterns make up a much higher percentage than expected, with the only exception being F_N_S (no context), which has very few patterns, just like I_N. For the rest of the full-game experiments, spin patterns collectively make up more than 50% of all spins, with the highest collective percentage being approximately 74%. One of the main reasons for this is the unnaturally

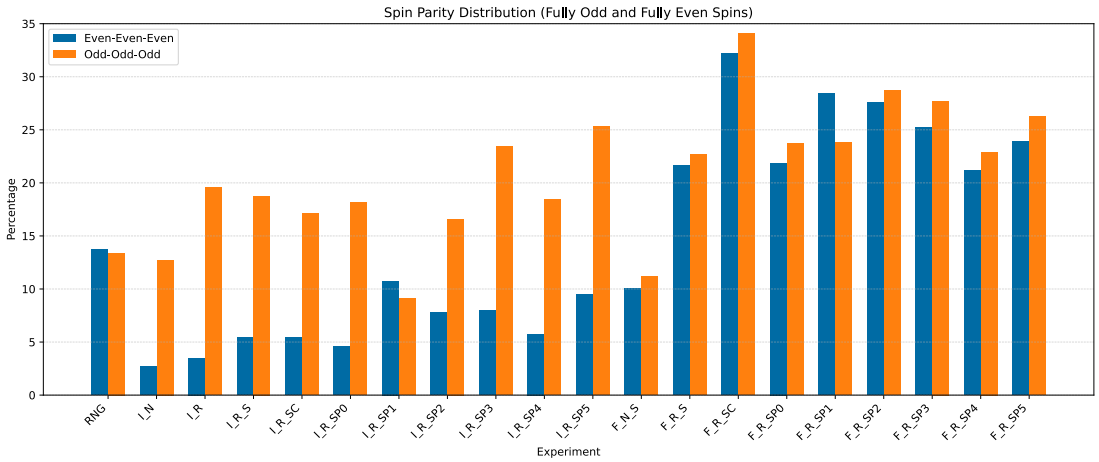

Figure 7.13: *The graph shows the percentage of spins that are either fully odd or fully even. This includes, but is not limited to, wins.*

high percentages of wins, which range from 19.7% to 57.3%. Wins will be discussed further in Section 7.2.5, but this once again shows that the context matters, since there is a very high percentage of wins in F_R_SC (57.3%) and a very low number of wins in F_N_S (less than 1%), while the experiments with the simple one-armed bandit context are all in the range 19.7% to 35.1%. The personas seem to have less effect on full games, with F_R_SP2 (Sophie's lucky numbers 1 and 10) having the most wins.

This indicates that context and game mode affect spin patterns, while personas mostly affect spin patterns in isolated spins.

7.2.4.2 Spin Parity

In terms of the parity of spins, here separated into fully odd and fully even spins, each of the two patterns has a probability of $125/1000 = 12.5\%$, i.e., they are each expected to make up 12.5% of all spins.

Figure 7.13 shows the percentage of spins that are fully odd and fully even. For the RNG experiment, each pattern makes up about 13% of all spins, which is close to the expected percentage. This also means that there are approximately the same number of fully odd and fully even spins. In contrast, most isolated-spin experiments have a large difference between the percentages of fully odd and fully even spins, typically having many more odd than even ones. Full-game experiments generally have similar parity percentages, which can be explained by their more evenly distributed values. However, spins of the same parity appear much more frequently in full-game experiments than in the RNG or any of the isolated-spin experiments, which is likely due to the high number of wins in full games. This means that the game mode influences the spin parity.

All LLM experiments except I_R_SP1 and F_R_SP1 have more fully odd than fully even spins. The two exceptions, I_R_SP1 and F_R_SP1, are the two experiments with

Table 7.6: *The total number of wins per experiment.*

Experiment	I	F
*_N	0	–
*_R	1	–
*_N_S	–	11
*_R_S	1	397
*_R_SC	1	1029
*_R_SP0	1	436
*_R_SP1	10	594
*_R_SP2	2	698
*_R_SP3	140	575
*_R_SP4	3	394
*_R_SP5	312	574

the lucky number 10. As earlier metrics have shown, these experiments show anchoring bias towards the number 10, which is even, so this is likely the reason why the parity of the spins is different for these two experiments. This shows that personas affect the parity of spins as well.

Finally, the context seems to have some effect on the spin parity. F_N_S and I_N both have fewer odd and even spins than expected, though this is also the case for I_R_SP1. For full games specifically, F_R_SC has higher percentages than any other experiment, while F_N_S has the lowest percentage of odd spins out of all experiments. This could in part be due to the high and low number of wins, respectively. Hence, it suggests that context does affect the results, but to a lesser extent than game mode and personas.

The findings regarding repetition patterns generally indicate that the game mode, context, and personas all affect emerging patterns and parity of the spins. However, spin parity and patterns are affected a bit differently, with game mode and personas being more influential for the parity, while game mode and context are important for other types of patterns.

7.2.5 Win Distribution

As mentioned for previous metrics, such as spin frequency and repetition patterns, there are big differences in the number of wins for each experiment, and wins vary for both game modes, context, and personas. This section further describes the tendencies that appear.

The bar graph with the pattern distribution (Figure 7.12) shows some general trends, one of them being that isolated spins generally have none to a few winning spins, while full games have unrealistically high numbers of wins. However, there are some outliers as seen in Table 7.6, which shows the total number of wins for each experiment.

Among the isolated-spin experiments, there are especially two experiments that stand out, and they both have personas with a focus on wins. I_R_SP3, which is the status quo (wins often) persona, has a total of 140 wins, while I_R_SP5, which is the

jailbreaking persona (Sophie's grandma), has 312 total wins. Interestingly, the persona that gets angry when she does not win (I_R_SP4) has very few wins. This suggests that for isolated spins, certain personas do affect the winning chances, with the jailbreaking prompt being the most successful one by far. However, LLM-generated randomness does not change much in response to anger. These findings supplement previous research on jailbreaking (Shen et al., 2024; Wang et al., 2025) by indicating that the LLM is also susceptible to jailbreaking strategies when prompted for random numbers using this interaction mode. However, for the status quo persona, it seems to maintain the status quo, which contrasts previous research on status quo bias in LLMs (Echterhoff et al., 2024), where the LLM actively avoided the status quo.

The same personas perform a bit differently for full games. Here, F_R_SP3 and F_R_SP5 do not have noticeably more wins than other personas. Instead, the persona prompt with the most wins is F_R_SP2 (Sophie's lucky numbers 1 and 10). It is difficult to say exactly why this is, but since the two most common spins for this experiment are "10 10 10" and "1 1 1", this could be due to anchoring bias. However, just like I_R_SP4, F_R_SP4 also has many fewer wins than the other experiments that include personas with a focus on winning within the same game mode. This once again indicates that anger does not produce more wins. This may indicate that the quality of the randomness is slightly improved through anger, which is consistent with prior research. However, since the number of wins for both experiments is similar to those of I_R_S and F_R_S, it is difficult to say if this effect is because of anger or if the persona itself had no effect at all.

The biggest outliers for full games are F_N_S and F_R_SC, which are the two experiments with different contexts. F_N_S is the no-context experiment that does not mention anything about three identical numbers being a win. Interestingly, this is the only full-game experiment with very few wins. It has only 11 wins, substantially differing from the experiment with the second-fewest wins, which is I_R_SP4 with 394 wins. At the opposite end is F_R_SC, which is the context where a player can only play for 50 spins if they win enough credits to keep playing. Even though this full-game experiment has the fewest total spins of all the full-game experiments, it clearly leads in wins with 1029, which is 331 more wins than the experiment with the second-highest number of wins. The effect of contexts is less clear for isolated-spin experiments because there are so few wins overall. However, even with the generally limited number of wins in isolated spins, the same trends as for full games are still present, but to a weaker degree. The no-context prompt, I_N, is the only experiment without unbiasing that has no wins at all. On the other hand, I_R_SC does have a win, even though it only has 403 spins in total.

The above outliers indicate that personas affect the win rate particularly for isolated spins, the context affects it primarily for full games, and the game mode overall makes a very big difference in terms of wins.

The heatmap in Figure 7.14, showing the win distribution for all 50 spin numbers, further shows that specific spin numbers are more likely to be wins. The figure sup-

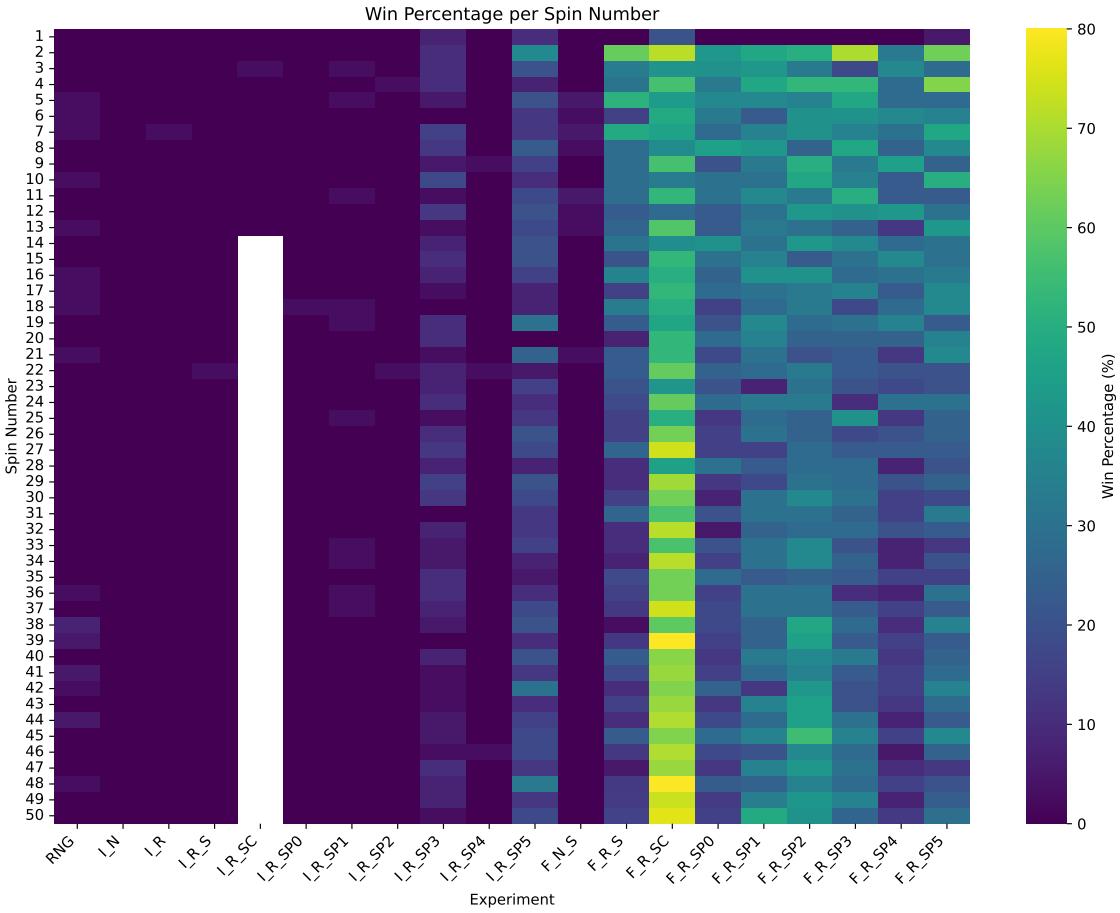

Figure 7.14: *The heatmap shows how likely a spin number is to be a win. Note that the percentage scale has been capped at 80% to better show trends. Since experiments have 40 games, 80% is equal to 32 spins for most cells. However, there are games with fewer spins than 50, which means that the total spins for the last cells may be lower. This is the case for F_R_SC, where only 34 games went all the way to 50 spins, which slightly skews the representation.*

ports the above findings, but it also shows that the first spin of full games is rarely a win, while the second spin is likely to be one. This indicates that other things than the prompt, such as biases from the training data, may play a larger role for the first few output tokens. This further supports that these first tokens may be more deterministic than the rest, but not necessarily the whole first-interaction prompt, as suggested by Nath (2025).

Furthermore, something interesting happens for F_R_SC. The prompt explains that the player starts with 10 credits, every spin costs 1 credit, winning by getting three identical numbers gives 3 credits, and that the player can only keep playing while they have enough credits. This setup means that the player must get at least 14 wins (10 starting credits + 42 won credits) to keep playing until 50 spins. These spins can technically be on any spin number, as long as there are enough credits to keep playing for, but early wins support playing longer, due to the low number of starting credits.

However, the LLM does the opposite: while there are generally many wins for

F_R_SC, and 34 of the 40 games continue for all 50 spins, especially later spins are very likely to be wins, with many of the cells having a win rate of over 70%. This observation aligns with trends seen during initial experimentation, where the LLM would sometimes output long sequences of wins during the second half of games in the format "1 1 1, 2 2 2, 3 3 3". This is likely also the case for these experiments, since F_R_SC on average has longer sequences than any other experiment, as seen in Section 7.1.5.

In conclusion, the findings indicate that the win distribution and rate are affected by game mode, personas, and context, though, as previously observed, personas and context affect isolated spins and full games differently.

7.2.6 Subconclusions for RQ2

All metrics for research question 2 indicate that spins and game outcomes are affected by personas, context, and game mode.

The game mode, in particular, had a substantial effect on which spins appeared, how frequently these outcomes occurred, how dependent the reels were on each other, how many and which patterns were seen, and the number of wins. Especially the number of wins stood out; full games tended to have unnaturally many wins, while most isolated-spin experiments had almost no wins. Since the wins were dominant for full games, this also affected most other metrics.

Personas and contexts also affected the spins and game outcomes, though not in equal measure for full games and isolated spins. The personas had a large effect on wins for isolated-spin experiments, since two of the three personas with a focus on wins gave significantly more wins, and some anchoring bias was observed. This was not as clear for full-game experiments, but there was still some anchoring bias in the frequently used spins. On the other hand, context seemed to make a big difference for full games, where the experiment without context had very few wins, and the experiment with explicit credits had a very large number of wins. The context appeared to be less important for isolated spins in terms of wins, although the spins were significantly more deterministic in the experiment without context. The influence of contexts (and hereunder the role) is consistent with prior work showing that role prompting affects outputs in other tasks (Kong et al., 2024), though the reduced effect in isolated spins suggests that this influence may depend on the sampling setup.

Finally, the first few output tokens of a full game response seem fairly deterministic. This was the case for the first and, to some degree, the second spin of full games. Combined with the observations from RQ1, this indicates that the first output token(s) may be more deterministic and also less influenced by prompting than the rest of a longer first-interaction prompt of a session. These observations provide further insight into the determinism of first-interaction prompts, as presented by Nath (2025), and how the game/interaction mode affects randomness.

Overall, the results support that LLM-generated random values, and consequently the spins and game outcomes, are not only affected by the training data, but also by

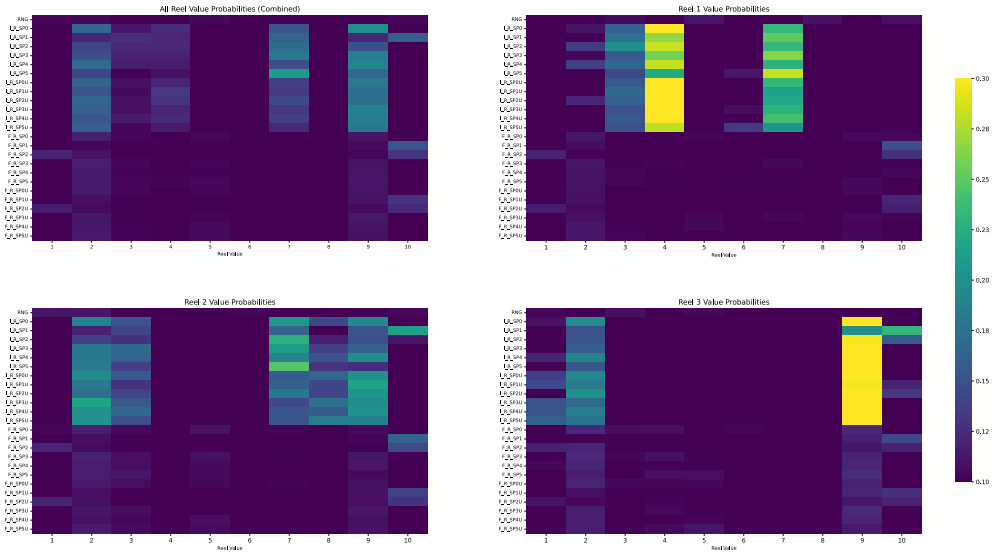

Figure 7.15: *Heatmap showing the probability distributions of values across all reels combined and for each individual reel, including unbiasing experiments. Note that the color scale is cropped at 0.1 and 0.3, respectively.*

the prompt itself, including the game mode, context, and different personas. In other words, the findings indicate that LLM-generated randomness is biased by and can be intentionally or unintentionally manipulated through prompting, even when the LLM is explicitly tasked with generating random integers with equal probability.

7.3 RQ3: If Randomness Is Affected by Personas, Context, and Game Mode, Is It Possible to Mitigate Such Biases Through Prompting?

The evaluation of the randomness of reel values (RQ1) and spins and game outcomes (RQ2) in Section 7.1 and Section 7.2, respectively, found that varying personas, contexts, and game modes do affect the LLM-generated output in different systematic ways. This section presents the results related to RQ3, focusing on whether it is possible to mitigate such biases through prompting. The evaluation is conducted by comparing the value distribution, total variation distance, spin frequency, pattern distribution, and win distribution results for the regular persona-experiments from the two previous sections with the corresponding unbiasing persona-experiments.

7.3.1 Value Distribution

The value distributions for the unbiasing experiments are largely the same as for their counterexamples, where unbiasing has not been added to the prompt (see Figure 7.15). The largest difference is observed for *_R_SP1 and *_R_SP2, the two personas that include anchoring effects, where the percentage of 10s drops when unbiasing is applied;

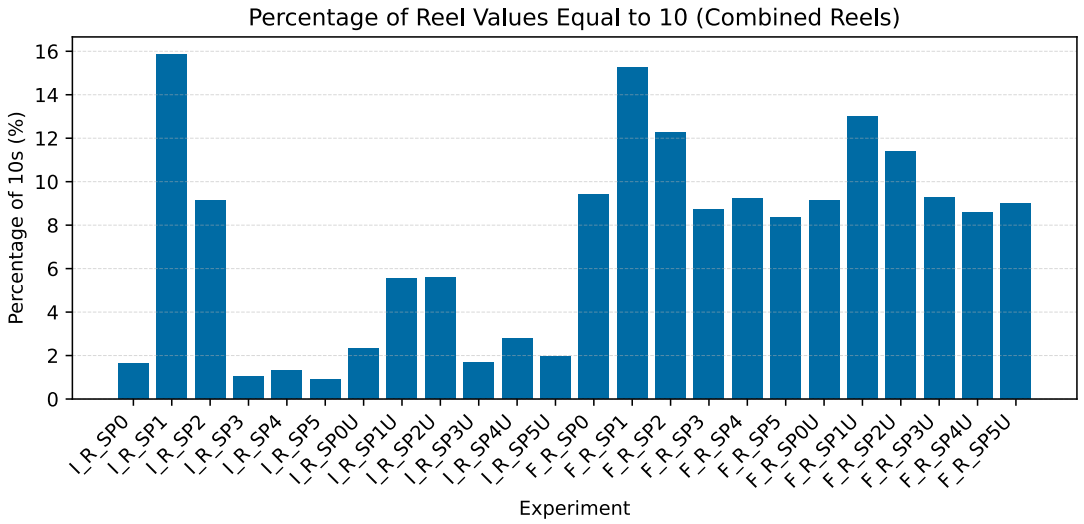

Figure 7.16: *The percentage of 10s in each experiment, including unbiasing experiments, across all three reels (combined).*

an effect that is clearly visible in Figure 7.16. This effect is present for both isolated spins and full-game experiments, though it is most pronounced for isolated spins. However, despite this, anchoring bias is still seen for the unbiasing experiments, meaning that the bias is reduced, but not removed entirely.

7.3.2 Total Variation Distance

The heatmap in Figure 7.17 shows the TVD for individual and combined reels, comparing regular and unbiasing variants of the persona-containing experiments. Figure 7.18 further visualizes the effect of unbiasing, measured by the change in TVD values when applying the unbiasing parameter. Positive-valued, red-colored cells indicate that the TVD value (thus, the divergence from the RNG) is increased by unbiasing, and the negative-valued, blue-colored cells indicate the opposite.

As can be seen in the heatmap, all the full-game experiments (except for F_R_SP0) show a modest reduction in TVD value for the unbiasing variants, both for individual [-0.0007, -0.0198] and combined [-0.0041, -0.0156] reels. This clearly contrasts with the isolated-spin experiments, for which the TVD values are slightly to moderately increased for two-thirds of the individual-reel TVD values [0.0101, 0.0683] and one of the six combined-reel TVD values (0.0132), while slightly to moderately decreased for the remaining one-third of the individual-reel values [-0.0017, -0.0372] and five combined-reel values [-0.0459, -0.0155].

At a combined reel-level, isolated-spin experiment with the greatest reduction in TVD value by unbiasing, thus, improved randomness, is the one with persona P0 (-0.0459), followed by P5 (-0.0437), and P3 (-0.0394). What can be considered paradoxical is that multiple of the isolated-spin experiments, including the just-mentioned I_R_SP5 and I_R_SP3, show an increase in TVD value for most or all individual reels,

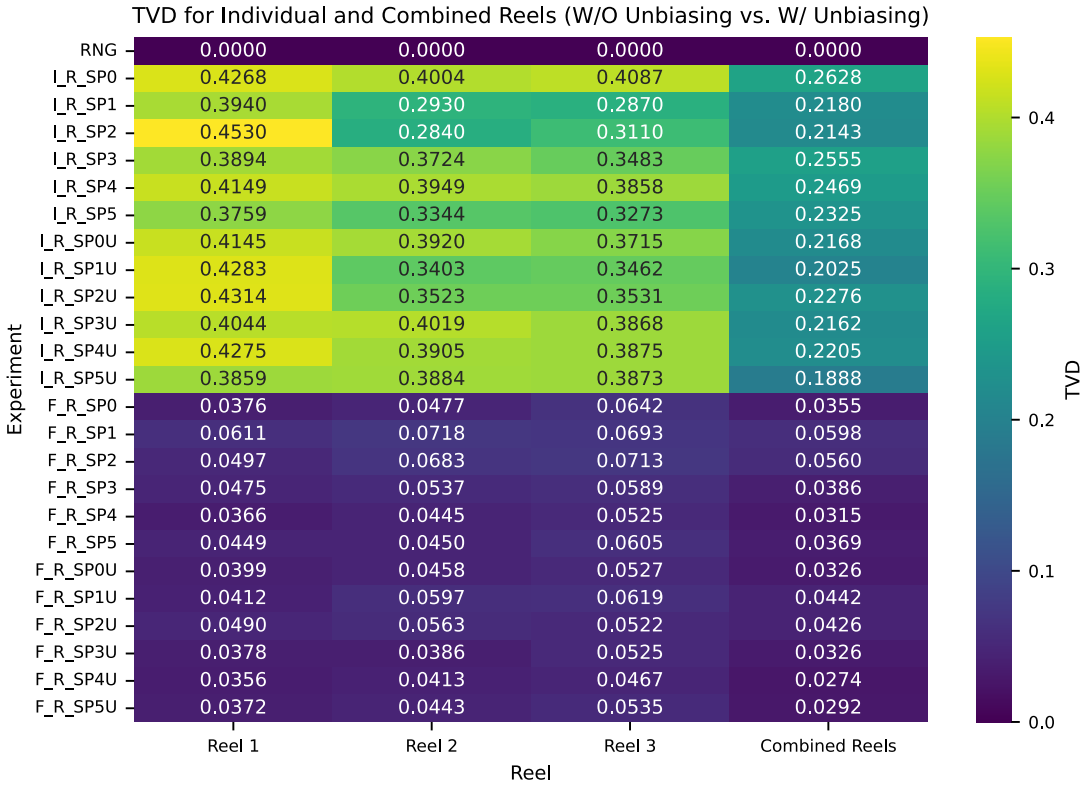

Figure 7.17: *TVD for individual and combined reels, comparing regular and unbiasing experiments. For isolated spins, individual reels primarily show small to moderate increases in TVD values when unbiasing is applied, whereas most combined reels, on the contrary, show small to moderate decreases. For the compared full-game experiments, unbiasing results in minimal decreases across all individual and combined reels (except for Reel 1, P0).*

but a decrease in TVD at the combined-reels level. This suggests that, for isolated-spin experiments, the effect of unbiasing is not consistent across individual and combined reels. I_R_SP2 and I_R_SP4 further show mixed signs across individual reels, suggesting sensitivity to reel-specific characteristics rather than a consistent effect on individual reels. Reel-wise, Reel 1 appears the least influenced reel for isolated-spin experiments, ranging from -0.0216 to 0.0343, compared to Reels 2 and 3, ranging from -0.0084 to the global maximum of 0.0683, and from -0.0372 to 0.0600, respectively.

To summarize, full-game experiments primarily show near-zero, negative ΔTVD values for both individual and combined reels, suggesting that these experiments are relatively robust to the unbiasing parameter (or more deterministic, depending on the perspective), though a modest change in behavior towards the RNG is observed. Most of the isolated-spin experiments also show a decrease in ΔTVD at a combined-reels level when applying unbiasing, suggesting less determinism overall. However, this effect is not seen at the individual-reel level, which shows more determinism. In addition, their fluctuating signs and wider ranges suggest that some reels are more sensitive to unbiasing than others; while Reel 1 shows the least sensitivity, Reel 2 and 3 show the largest positive changes in TVD, i.e., a greater divergence from the RNG, when unbiasing is applied.

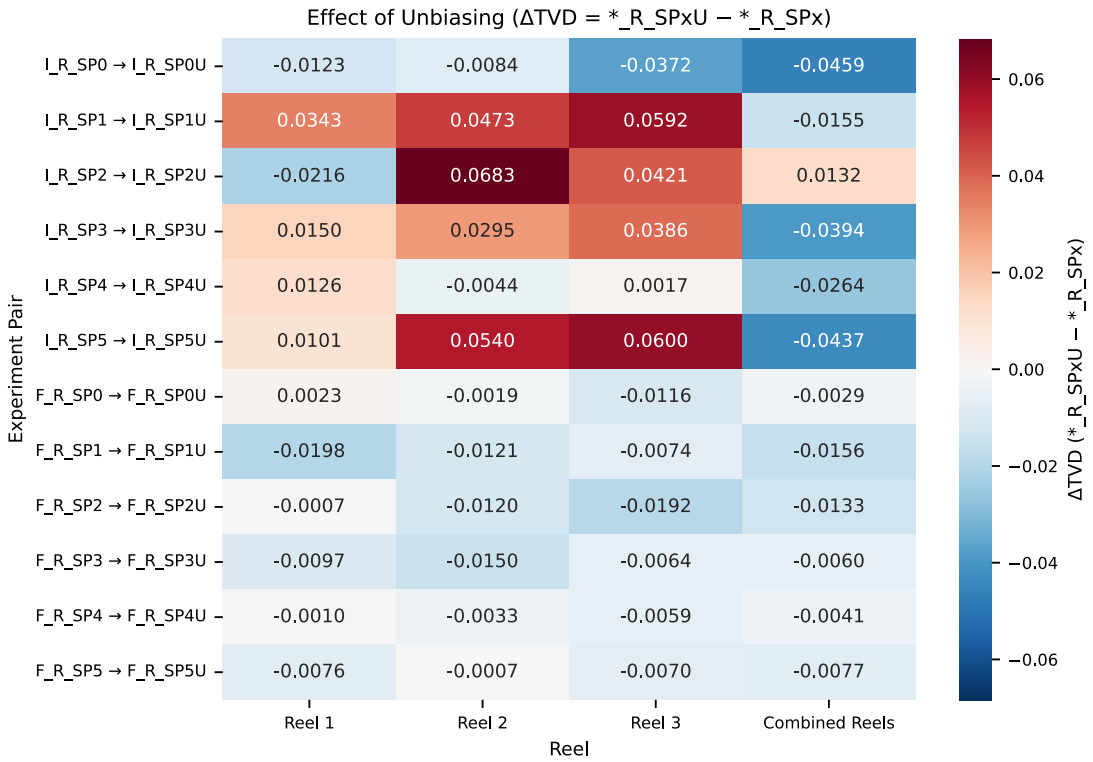

Figure 7.18: *Effect of unbiasing* (ΔTVD = *_R_SP[0-5]U − *_R_SP[0-5]). *For persona-containing isolated-spin experiments, two-thirds* (*12 out of 18*) *of the individual reel TVD values are slightly to moderately increased* [*0.0101, 0.0683*] *when applying the unbiasing parameter. However, only one of the six combined-reels TVD values for persona-containing isolated-spin experiments is slightly increased* (*0.0132*), *while the remaining ones are slightly to moderately decreased* [*-0.0459, -0.0155*]. *The TVD value of all persona-containing full-game experiments* (*except for one*) *is slightly reduced both per-reel* [*-0.0007, -0.0198*] *and for combined reels* [*-0.0029, -0.0156*] *when applying the unbiasing parameter.*

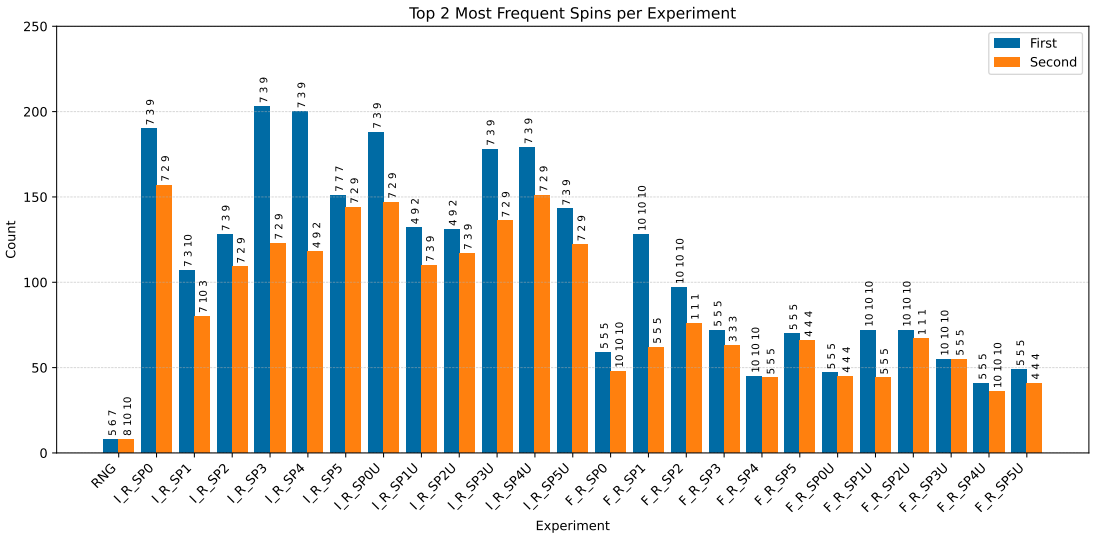

Figure 7.19: *The graph shows the top 2 most frequently used spins for all persona experiments.*

7.3.3 Spin Frequency

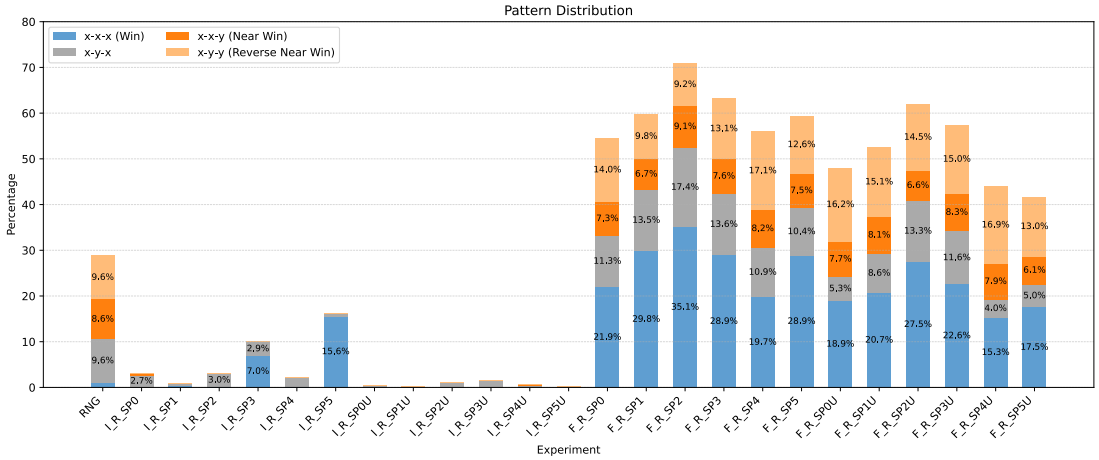

Figure 7.20: *The graph shows the percentage of spins that follow specific patterns, namely near wins, reverse near wins, the x-y-x pattern, and wins.*

The graph in Figure 7.19 shows a comparison of the top 2 most frequent spins for the regular and unbiasing persona experiments.

For isolated-spin experiments, there is a difference between the same experiment with and without unbiasing. I_R_SP1 shows anchoring bias for number 10, and I_R_SP5 has ”7 7 7” (a win) as its most frequent spin. In contrast, none of these are seen in I_R_SP1U and I_R_SP5U. Instead, the top two spins for all unbiasing isolated-spin experiments are either ”7 3 9”, ”4 9 2”, or ”7 2 9”. This indicates that asking the LLM to avoid being biased does affect the randomness. Nevertheless, it does not become more random, but rather more deterministic.

Full games, on the other hand, do not show much difference between regular and unbiasing experiments. All the most frequently used spins are still wins, and for 3 of 5 experiments, the top two spins are the same (though F_R_SP4U has switched the two around). Additionally, there is still anchoring bias for the lucky numbers 1 and 10 in F_R_SP1U and F_R_SP2U. These results suggest that unbiasing has a limited effect on the randomness of full games.

7.3.4 Pattern Distribution

The graph in Figure 7.20 shows a comparison of the pattern distribution percentages of spins for regular and unbiasing persona-experiments.

For isolated-spin experiments, the spin patterns disappear almost entirely. None of the unbiasing experiments have any wins at all, and even the x-y-x pattern almost disappears. This indicates that the LLM actively avoids patterns by ensuring that all the reel values of a spin are different.

Full games overall look very similar when comparing the same experiment with and without unbiasing, but interestingly, all unbiasing experiments have a lower overall total percentage of spins that are patterns. This is partially due to fewer wins for all

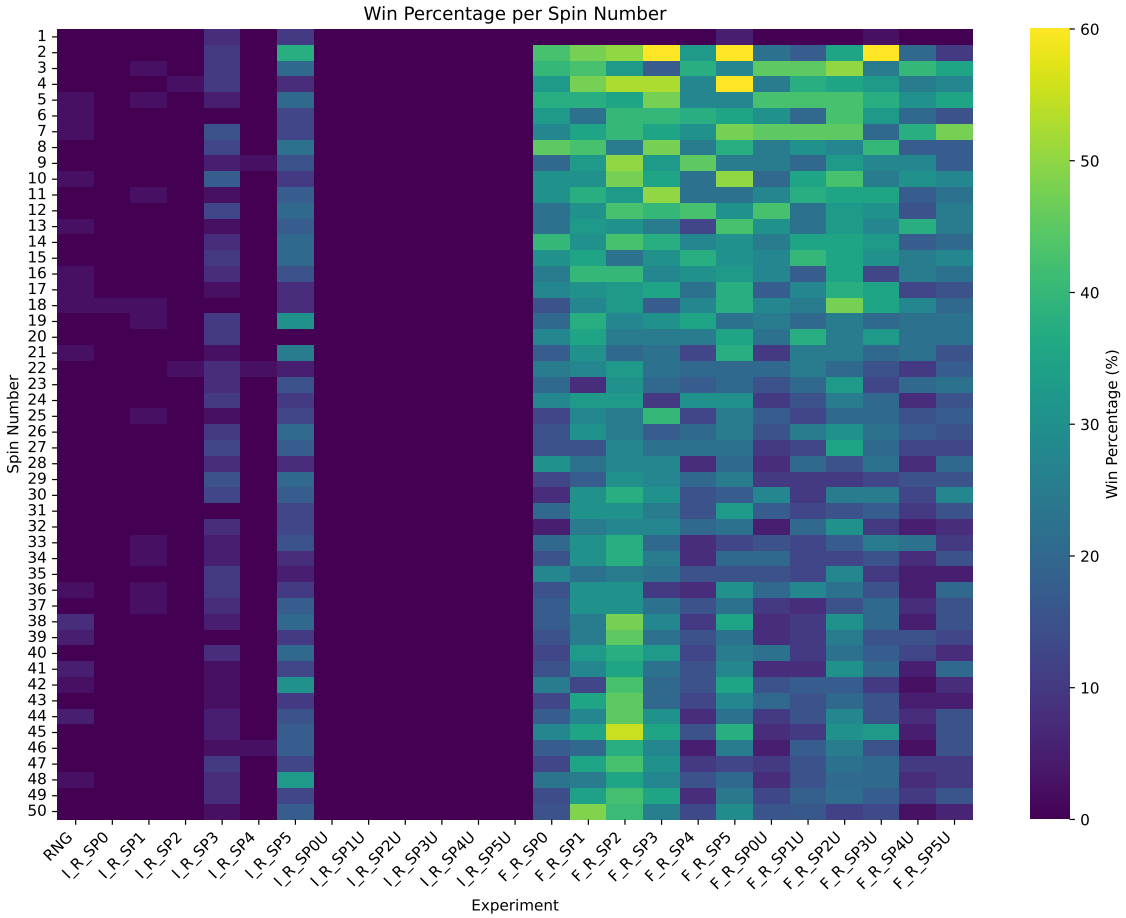

Figure 7.21: *The heatmap shows how likely a spin number is to be a win. Note that the percentage scale has been capped at 60% to better show trends. Since experiments have 40 games, 60% is equal to 24 spins for most cells. However, there are games with fewer spins than 50, which means that the total spins for the last cells may be lower.*

experiments. For example, 28.9% of all spins in F_R_SP5 are wins, while only 17.5% of all spins in F_R_SP5U are wins. There are also fewer x-y-x spins for all experiments. In contrast to this, most unbiasing experiments actually have more reverse near wins, while near wins have similar numbers for most experiments. All of this indicates that unbiasing does have an effect on patterns for full games as well, though to a lesser extent. There are still unrealistically many wins, unlike what was observed for F_N_S (no context), which indicates that the context still plays a role, even when unbiasing is added to the prompt.

7.3.5 Win Distribution

The heatmap in Figure 7.21 shows a comparison of the win distribution in percentage over spin numbers for regular and unbiasing persona-experiments.

As already mentioned, wins were fully eliminated with unbiasing for isolated spins, which is also reflected in the heatmap. For full games, F_R_SP2 and F_R_SP2U stand out the most. In F_R_SP2, wins are more frequent at the beginning, and towards the

end of the game, and while the unbiasing version still has wins at the beginning of the game, it has much fewer wins towards the end. This is a general trend for all of the unbiasing experiments, as all have fewer wins toward the end of the game. The heatmap also clearly shows that F_R_SP5U has fewer wins overall than its counterpart, especially at the beginning of the game. F_R_SP5 has two cells with win frequencies greater than 60%, and additional cells with frequencies greater than 50%, while F_R_SP5U only has a single cell with a win frequency of about 50%. This indicates that unbiasing does have some effect on wins for both isolated spins and full games.

7.3.6 Subconclusions for RQ3

In conclusion, the findings from these metrics support that unbiasing affects randomness as well as spins and game outcomes to some degree for both isolated-spin and full-game experiments. However, particularly in the case of full games, unbiasing does not fully remove bias, since unnaturally many wins and some anchoring bias are still seen. In other words, even with unbiasing, the prompts seem to affect the randomness. For isolated spins, unbiasing does reduce the biases, e.g., by removing all wins from winning-focused experiments and significantly reducing anchoring bias, but this only makes the spins more deterministic, rather than becoming more random. This somewhat limited effect is further supported by metrics that were not included in this section, such as entropy, runs test, and sequences, as these were largely unaffected by unbiasing.

Again, these findings support that the randomness differs between game modes. This may also be why the effect of unbiasing on full-game experiments aligns with previous research on zero-shot-mitigating strategies, shown to have little to no effect on anchoring bias (Echterhoff et al., 2024; Lou et al., 2025), while isolated-spin experiments contrast these findings by showing a considerable effect of debiasing on anchoring bias. All of these findings further support that prompting does, in fact, affect randomness, as hypothesized in this thesis.

8 Conclusion

8.1 Conclusion

This thesis aimed to address the following overarching questions: ”Does the prompt affect the output of LLMs when they are tasked with generating randomness? And if so, what implications could it have if RNG-generated data is replaced by LLM outputs within software systems?”

To answer these questions, we evaluated the randomness of reel values, spins, and game outcomes generated by OpenAI’s open-weight gpt-oss:20b LLM across 31 experiments with controlled prompt variations as well as an RNG baseline experiment. During evaluation, the experiments were compared across game modes, contexts, and personas.

In particular, the game mode (i.e., interaction mode) affected the randomness. According to metrics such as TVD and entropy, full games behaved much more randomly than isolated spins in terms of the distribution of values. However, other metrics, such as runs test, sequence lengths, patterns, spin frequency, and win distribution, showed that full games had different issues, which also set the two game modes further apart. Full-game experiments had exceptionally many wins when the prompt context described wins, many more than the isolated-spin experiments, which led to most of their frequent spins being wins as well. They also had much longer sequences and showed many more patterns than isolated spins did. All of this indicates that the interaction mode has a large effect on LLM-generated randomness and provides further insight into previous research with somewhat contrasting findings on the quality of LLM-generated randomness (Karanjai et al., 2025; Coronado-Blázquez, 2025; Nath, 2025). This shows that shared history does not actually improve the randomness, even though the LLM does a better job mimicking random distributions, and thereby hides its flaws by using patterns. Additionally, the first token(s) showed higher levels of determinism and were less affected by the prompt. This provides further insight into the role of first-interaction prompts by showing that the higher level of determinism likely

happens at the token level, not at the prompt level as suggested by Nath (2025).

Personas also influenced the randomness in both game modes. Experiments that had personas with lucky numbers showed anchoring bias by generating the lucky number much more frequently than all other experiments. However, this was only the case when the number was relevant to the context, not when it was unrelated, as was the case for the persona where the time was 10 o'clock. In a similar manner, the LLM also showed status quo bias and sensitivity to jailbreak prompting. These two personas, both focused on winning, received many more wins in isolated-spin experiments, which normally had none to a few wins. However, emotion prompting, in which the persona got angry after losing, produced fewer wins than all other personas for full games, and the other winning-focused personas for isolated spins. Whether this is accidental or a direct effect of a negative persona is difficult to say, but it shows that this technique is much less effective. While some findings, such as those for anchoring bias and jailbreaking, are consistent with prior work on the effect of prompting in other tasks (Lou et al., 2025; Wang et al., 2025; Shen et al., 2024), others, such as those for status quo bias (Echterhoff et al., 2024) seem unique to generated randomness.

The context also affected the generated randomness in both game modes. The experiment without any context, and hence no explanation of what a "win" is, had few wins for full games and no wins for isolated spins. On the other hand, the full-game experiment with the context of a one-armed bandit with explicit credits got the most wins of all experiments; this, with a strong lead. These findings for full games contradict prior work that shows that role prompting improves reasoning (Kong et al., 2024), as in this setting, role prompting does not improve reasoning for the task but rather increases adherence to letting the player win, reflecting greater bias and manipulability. Furthermore, the isolated-spin experiment without context behaved much more deterministically, while the same experiment for full games performed the best in terms of randomness across several metrics. These findings suggest that the context affects the generated randomness, but that the effect differs between game modes.

Finally, when the LLM was asked not to be biased by the prompt, full games only marginally changed their behavior, while isolated spins did indeed seem to ignore the cognitive biases mentioned above. However, this only led to more deterministic results, and hence lower-quality randomness.

In conclusion, the findings strongly imply that the randomness generated by LLMs is biased not only by the training data but also by the prompt, even when the LLM is explicitly asked to provide random independent numbers with equal probability. The findings also suggest that the LLM weighs persona personalities and contextual knowledge higher than the task itself, and that randomness can therefore be manipulated through certain types of prompting. Furthermore, even though it may seem like shared history improves randomness, this comes with its own set of problems, mainly in terms of runs, patterns, and sequences.

This means that if an LLM were to be used to provide randomness within a system, the randomness would not only behave more deterministically but also be influ-

enced by the way it was prompted, down to the specific biases that may accidentally or intentionally be added to the prompt. In a case like the one-armed bandit or other chance-based games, this can be used to manipulate wins or specific numbers. In other cases, such as other types of games or systems that utilize randomness, this may favor or harm specific users. Furthermore, as the initial experiment findings showed, this behavior happens not only with numbers but also seems to be an issue with the random generation of other things, such as symbols. Hence, using LLMs to generate random outcomes should generally be avoided.

8.2 Limitations and Future Research

This thesis has several limitations, many of which are due to the scope. First, due to the large number of controlled prompt variations for the final experiments, only the gpt-oss:20b model was tested on a larger scale. Our initial experimentation with the gpt-oss:120b model showed the same patterns and biases, which indicates that the final results may apply to both models, but whether these results generalize to other LLM families remains an open question. Second, this thesis has shown that there are biases in LLM-generated randomness by showing specific biases, but there may be other biases that have yet to be uncovered. Third, while the initial experiments showed interesting findings, the size of these experiments was very small. Hence, they should be subjected to more research to conclude anything for certain. Finally, while this thesis tested two different interaction modes, a chat-based setup was not explored. Given that our results and prior work (Nath, 2025) indicate a significant difference between setups, future research could compare all three setups to better understand how interaction modes affect LLM-generated randomness.

These limitations are also potential areas for future research, especially since this, to our knowledge, is still relatively unexplored. Furthermore, this thesis mainly focused on a one-armed bandit framing, meaning that the prompts and personas had a focus on winning. Future research could explore how other contexts affect the generated randomness, such as the D&D context that we briefly explored during the initial experiments, or different non-game-related scenarios. Future research can also test whether setting the hyperparameters differently affects the biases that were seen in this thesis. We briefly tested the temperature and still saw anchoring bias emerge for all tested temperature values, consistent with prior work suggesting that temperature does not have a large effect on generated randomness (Coronado-Blázquez, 2025; Karanjai et al., 2025). However, other hyperparameters remain unexplored. Finally, future research can explore how prompting affects the generation of non-uniform random distributions, as these may introduce different or additional biases compared to those observed for uniform distributions in this thesis.

Bibliography

Alkurdi, Zaina, Lucas Palmer, and Luke Clark (Nov. 2025). "Examining the Illusion of Control Via Outcome Sequences in Online Slot Machine Gambling". In: *Addiction Research & Theory* 34.3, pp. 350–366. ISSN: 1606-6359. DOI: 10.1080/16066359.2025.2581087. URL: https://doi.org/10.1080/16066359.2025.2581087 (visited on 2026-01-26).

Anderson, David F., Timo O. Seppäläinen, and Benedek Valkó (2018). *Introduction to Probability*. Cambridge Mathematical Textbooks. Cambridge New York: Cambridge University Press. ISBN: 978-1-108-41585-9.

Anthropic (2024). *Model Parameters*. URL: https://github.com/anthropics/courses/blob/master/anthropic_api_fundamentals/04_parameters.ipynb (visited on 2026-05-11).

Anthropic (2026). *Create a Message*. URL: https://platform.claude.com/docs/en/api/python/messages/create (visited on 2026-05-11).

Barnard, Joel (2025). *What is Embedding?* URL: https://www.ibm.com/think/topics/embedding (visited on 2026-05-11).

Belcic, Ivan and Cole Stryker (2025). *What Are LLM Parameters?* URL: https://www.ibm.com/think/topics/llm-parameters#692473877 (visited on 2026-05-11).

Bergmann, Dave (2025). *What Is a Reasoning Model?* URL: https://www.ibm.com/think/topics/reasoning-model (visited on 2026-05-12).

Bergmann, Dave (2026). *What is AI Inference?* URL: https://www.ibm.com/think/topics/ai-inference (visited on 2026-04-27).

Bernays, Florence, Marco Henriques Pereira, and Jochen Menges (Jan. 2026). *How Human is AI? Examining the Impact of Emotional Prompts on Artificial and Human and Responsiveness*. Preprint. DOI: 10.48550/arXiv.2601.05104. URL: http://arxiv.org/abs/2601.05104 (visited on 2026-02-14).

Bikos, Anastasios et al. (Oct. 2023). "Random Number Generators: Principles and Applications". In: *Cryptography* 7.4, p. 54. DOI: 10.3390/cryptography7040054.

Corder, Gregory W. and Dale I. Foreman (2007). *NONPARAMETRIC STATISTICS: A Step-by-Step Approach*. ISBN: 978-1-118-84031-3.

Coronado-Blázquez, Javier (Feb. 2025). *Deterministic or Probabilistic? The Psychology of LLMs as Random Number Generators*. Preprint. DOI: 10.48550/arXiv.2502.19965. URL: http://arxiv.org/abs/2502.19965 (visited on 2026-01-26).

Cote, Linda R. et al. (Aug. 2024). "3.4: Chi-Square". In: *Introduction to Statistics in the Psychological Sciences*. Applied Statistics. Statistics LibreTexts. URL: https://stats.libretexts.org/Bookshelves/Applied_Statistics/Introduction_to_Statistics_in_the_Psychological_Sciences_(Cote_Gordon_Randell_and_Marvin)/03%3A_Additional_Hypothesis_Tests/3.04%3A_Chapter_14-_Chi-Square (visited on 2026-04-02).

Daugherty, Brian (2026). *What is RTP in Slots: Return to Player Explained*. URL: https://coinpoker.com/guides/rtp/ (visited on 2026-05-04).

Echterhoff, Jessica Maria et al. (Nov. 2024). "Cognitive Bias in Decision-Making with LLMs". In: *Findings of the Association for Computational Linguistics: EMNLP 2024*. Ed. by Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen. Miami, Florida, USA: Association for Computational Linguistics, pp. 12640–12653. DOI: 10.18653/v1/2024.findings-emnlp.739. URL: https://aclanthology.org/2024.findings-emnlp.739/ (visited on 2026-02-26).

Frederick, Shane, George Loewenstein, and Ted O'Donoghue (2002). "Time Discounting and Time Preference: A Critical Review". In: *Journal of Economic Literature* 40.2, pp. 351–401. ISSN: 0022-0515. URL: https://www.jstor.org/stable/2698382 (visited on 2026-05-21).

Gadesha, Vrunda, Eda Kavlakoglu, and Vanna Winland (2025). *What Is Chain of Thought (CoT) Prompting?* URL: https://www.ibm.com/think/topics/chain-of-thoughts (visited on 2026-05-13).

Gambling Commission (2024). *Licensing Authority Bulletin August 2020 - Feature Article: How Can Gaming Machines Meet Their %RTP If They Are Random?* URL: https://www.gamblingcommission.gov.uk/authorities/guide/page/la-bulletin-august-2020-feature-article (visited on 2026-01-26).

Gilovich, Thomas, Robert Vallone, and Amos Tversky (July 1985). "The Hot Hand in Basketball: On the Misperception of Random Sequences". In: *Cognitive Psychology* 17.3, pp. 295–314. ISSN: 00100285. DOI: 10.1016/0010-0285(85)90010-6. URL: https://linkinghub.elsevier.com/retrieve/pii/0010028585900106 (visited on 2026-05-20).

Goldie, Sophie (2018). "The Poisson Distribution". In: *Cambridge International AS and A Level Mathematics Probability and Statistics 2*. United Kingdom: Hachette Learning. ISBN: 9781510421776.

Gu, Jia et al. (Dec. 2024). *Do LLMs Play Dice? Exploring Probability Distribution Sampling in Large Language Models for Behavioral Simulation*. Preprint. DOI: 10.48550/arXiv.2404.09043. URL: http://arxiv.org/abs/2404.09043 (visited on 2026-02-27).

Holdsworth, James (2025). *What Is AI Bias?* URL: https://www.ibm.com/think/topics/ai-bias (visited on 2026-05-12).

IBM (2026). *What Are AI Hallucinations?* URL: https://www.ibm.com/think/topics/ai-hallucinations (visited on 2026-05-12).

Kahneman, Daniel and Amos Tversky (1979). "Prospect Theory: An Analysis of Decision Under Risk". In: *Econometrica* 47.2, pp. 263–291. ISSN: 0012-9682. DOI: 10.2307/1914185. URL: https://www.jstor.org/stable/1914185 (visited on 2026-02-18).

Karanjai, Rabimba et al. (Oct. 2025). *Evaluating the Quality of Randomness and Entropy in Tasks Supported by Large Language Models*. Preprint. DOI: 10.48550/arXiv.2510.12080.

Kent State University Libraries (Mar. 2026). *Chi-Square Test of Independence*. URL: https://libguides.library.kent.edu/SPSS/ChiSquare (visited on 2026-04-02).

Kong, Aobo et al. (Mar. 2024). *Better Zero-Shot Reasoning with Role-Play Prompting*. Preprint. arXiv:2308.07702v2 [cs.CL]. DOI: 10.48550/arXiv.2308.07702. URL: http://arxiv.org/abs/2308.07702 (visited on 2026-02-19).

Langer, Ellen J. (1975). "The Illusion of Control". In: *Journal of Personality and Social Psychology* 32.2, pp. 311–328. URL: https://nuovoeutile.it/wp-content/uploads/2014/10/Langer1975_IllusionofControl.pdf (visited on 2026-05-20).

Laplace, Pierre-Simon (1902). *A Philosophical Essay on Probabilities*. Translated from the sixth French edition by F. W. Truscott and F. L. Emory. John Wiley & Sons. URL: https://dn790000.ca.archive.org/0/items/philosophicaless00lapluoft/philosophicaless00lapluoft.pdf (visited on 2026-05-20).

Li, Cheng et al. (Nov. 2023). *Large Language Models Understand and Can be Enhanced by Emotional Stimuli*. Preprint. DOI: 10.48550/arXiv.2307.11760. URL: http://arxiv.org/abs/2307.11760 (visited on 2026-02-14).

Lou, Jiaxu and Yifan Sun (Dec. 2025). "Anchoring Bias in Large Language Models: An Experimental Study". In: *Journal of Computational Social Science* 9.11. ISSN: 2432-2725. DOI: 10.1007/s42001-025-00435-2. URL: https://doi.org/10.1007/s42001-025-00435-2 (visited on 2026-02-17).

Marvin, Ggaliwango et al. (2024). "Prompt Engineering in Large Language Models". In: *Data Intelligence and Cognitive Informatics*. Ed. by I. Jeena Jacob, Selwyn Piramuthu, and Przemyslaw Falkowski-Gilski. Algorithms for Intelligent Systems. Springer Nature, pp. 387–402. ISBN: 978-981-99-7999-8 978-981-99-7962-2. DOI: 10.1007/978-981-99-7962-2_30. URL: https://link.springer.com/10.1007/978-981-99-7962-2_30 (visited on 2026-02-12).

McHugh, Mary L. (June 2013). "The Chi-Square test of Independence". In: *Biochemia Medica* 23.2, pp. 143–149. ISSN: 1330-0962. DOI: 10.11613/BM.2013.018. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC3900058/ (visited on 2026-03-30).

Miller, Joshua B. and Adam Sanjurjo (2018). "Surprised by the Hot Hand Fallacy? A Truth in the Law of Small Numbers". In: *Econometrica* 86.6, pp. 2019–2047. ISSN: 1468-0262. DOI: 10.3982/ECTA14943. URL: https://onlinelibrary.wiley.com/doi/abs/10.3982/ECTA14943 (visited on 2026-05-20).

Mischel, Walter, Ebbe B. Ebbesen, and Antonette Raskoff Zeiss (1972). "Cognitive and Attentional Mechanisms in Delay of Gratification". In: *Journal of Personality and Social Psychology* 21.2, pp. 204–218. ISSN: 1939-1315, 0022-3514. DOI: 10.1037/h0032198. URL: https://doi.apa.org/doi/10.1037/h0032198 (visited on 2026-05-21).

Moreau, David, Kristina Wiebels, and Carl Boettiger (July 2023). "Containers for Computational Reproducibility". In: *Nature Reviews Methods Primers* 3. DOI: 10.1038/s43586-023-00236-9.

Nadurak, Vitaliy (Oct. 2025). "Heuristics and Cognitive Biases: A Conceptual Analysis". In: *Memory & Cognition*. ISSN: 1532-5946. DOI: 10.3758/s13421-025-01814-w. URL: https://doi.org/10.3758/s13421-025-01814-w (visited on 2026-01-26).

Nath, Som (July 2025). *Emergent Numeric Bias in Large Language Models: An Empirical Study on the Anomalous Recurrence of the Number 27 Across Independent Sessions*. Preprint. DOI: 10.21203/rs.3.rs-7128459/v1.

Noble, Joshua (2026). *What is LLM Temperature?* URL: https://www.ibm.com/think/topics/llm-temperature#692473878 (visited on 2026-05-11).

NVIDIA (2026). *NVIDIA DGX Spark – A Personal AI Computer, Designed to Build And Run AI*. Datasheet. URL: https://nvdam.widen.net/s/tlzm8smqjx/workstation-datasheet-dgx-spark-gtc25-spring-nvidia-us-3716899-web (visited on 2026-04-09).

Ollama (2026). *Ollama*. URL: https://ollama.com (visited on 2026-04-09).

OpenAI (Aug. 2025). *Introducing gpt-oss*. URL: https://openai.com/index/introducing-gpt-oss/ (visited on 2026-03-12).

OpenAI et al. (Aug. 2025). *gpt-oss-120b & gpt-oss-20b Model Card*. arXiv:2508.10925 [cs]. DOI: 10.48550/arXiv.2508.10925. URL: http://arxiv.org/abs/2508.10925 (visited on 2026-04-27).

Pournemat, Mobina et al. (Sept. 2025). *Reasoning Under Uncertainty: Exploring Probabilistic Reasoning Capabilities of LLMs*. Preprint. DOI: 10.48550/arXiv.2509.10739. URL: http://arxiv.org/abs/2509.10739 (visited on 2026-02-27).

Premmi and Beguène (May 2022a). *Distinguishing in a Probabilistic World-Part 3*. URL: https://principlesofcryptography.com/distinguishing-in-a-probabilistic-world-part-3/ (visited on 2026-03-13).

Premmi and Beguène (June 2022b). *Distinguishing in a Probabilistic World-Part 4*. URL: https://principlesofcryptography.com/distinguishing-in-a-probabilistic-world-part-4/ (visited on 2026-03-13).

Samuelson, William and Richard Zeckhauser (Mar. 1988). "Status Quo Bias in Decision Making". In: *Journal of Risk and Uncertainty* 1.1, pp. 7–59. ISSN: 0895-5646, 1573-0476. DOI: 10.1007/BF00055564. URL: http://link.springer.com/10.1007/BF00055564 (visited on 2026-02-27).

Schulhoff, Sander et al. (Feb. 2025). *The Prompt Report: A Systematic Survey of Prompt Engineering Techniques*. arXiv:2406.06608 [cs]. DOI: 10.48550/arXiv.2406.06608. URL: http://arxiv.org/abs/2406.06608 (visited on 2026-02-12).

SciPy (2008). *scipy.stats.chi2_contingency*. URL: https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.chi2_contingency.html (visited on 2026-03-26).

Shannon, Claude (1948). *A Mathematical Theory of Communication*. URL: https://www.organism.earth/library/document/mathematical-theory-of-communication (visited on 2026-03-13).

Shen, Xinyue et al. (Dec. 2024). ""Do Anything Now": Characterizing and Evaluating In-The-Wild Jailbreak Prompts on Large Language Models". In: *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*. CCS '24. Association for Computing Machinery, pp. 1671–1685. ISBN: 979-8-4007-0636-3. DOI: 10.1145/3658644.3670388. URL: https://dl.acm.org/doi/10.1145/3658644.3670388 (visited on 2026-03-09).

Stipčević, Mario and Çetin Koç (2014). "True Random Number Generators". In: *Open Problems in Mathematics and Computational Science*. Springer, pp. 275–315. ISBN: 978-3-319-10682-3. DOI: 10.1007/978-3-319-10683-0_12. URL: https://www.researchgate.net/publication/299824248_True_Random_Number_Generators (visited on 2026-05-20).

Stryker, Cole (2026). *What Are Large Language Models (LLMs)?* URL: https://www.ibm.com/think/topics/large-language-models (visited on 2026-04-27).

Stryker, Cole and Eda Kavlakoglu (2026). *What Is Artificial Intelligence (AI)?* URL: https://www.ibm.com/think/topics/artificial-intelligence (visited on 2026-05-10).

Syed, Meredith and Vrunda Gadesha (2025). *What Is Zero-Shot Prompting?* URL: https://www.ibm.com/think/topics/zero-shot-prompting#f1 (visited on 2026-05-13).

Team Olmo et al. (2025). *Olmo 3*. arXiv:2512.13961v2 [cs.CL]. DOI: 10.48550/ARXIV.2512.13961. URL: https://arxiv.org/abs/2512.13961 (visited on 2026-04-27).

Touvron, Hugo et al. (July 2023). *Llama 2: Open Foundation and Fine-Tuned Chat Models*. arXiv:2307.09288 [cs]. DOI: 10.48550/arXiv.2307.09288. URL: http://arxiv.org/abs/2307.09288 (visited on 2026-04-27).

Tumeo, Mark (Jan. 1994). "The Meaning of Stochasticity, Randomness and Uncertainty in Environmental Modeling". In: *Stochastic and Statistical Methods in Hydrology and Environmental Engineering*. Ed. by K.W. Hipel. Vol. 10/4. Water Science and Technology Library. Springer, pp. 33–38. ISBN: 978-94-010-4467-7. DOI: 10.1007/978-94-011-1072-3_3.

Turney, Shaun (May 2022). *Chi-Square Test of Independence | Formula, Guide & Examples*. URL: https://www.scribbr.com/statistics/chi-square-test-of-independence/ (visited on 2026-04-02).

Tversky, Amos and Daniel Kahneman (1974). "Judgment under Uncertainty: Heuristics and Biases". In: *Science, New Series* 185.4157, pp. 1124–1131. URL: http://www.jstor.org/stable/1738360.

Vaswani, Ashish et al. (2017). *Attention Is All You Need*. Preprint. arXiv:1706.03762v7 [cs.CL]. DOI: 10.48550/ARXIV.1706.03762. URL: https://arxiv.org/abs/1706.03762 (visited on 2026-05-12).

Wang, Zhenhua et al. (2025). "Evading LLMs' Safety Boundary with Adaptive Role-Play Jailbreaking". In: *Electronics (Basel)* 14.24, p. 4808. ISSN: 2079-9292. DOI: 10.3390/electronics14244808.

Zhang, Zhijun et al. (June 2025). "A Comprehensive Overview of Generative AI (GAI): Technologies, Applications, and Challenges". In: *Neurocomputing* 632, p. 129645. ISSN: 09252312. DOI: 10.1016/j.neucom.2025.129645. URL: https://linkinghub.elsevier.com/retrieve/pii/S0925231225003170 (visited on 2026-05-10).