Neuromorphic Computing with Spin Hall Oscillators: Modelling and Leveraging of nonlinear magnetization dynamics for classification and prediction computational tasks

（スピンホール発振器を使用した脳型計算処理:
分類および予測タスクのための非線形磁化ダイナミクスの
モデル化と活用）

MOHAN JOHN REX

Advisory Committee

Prof. Yasuhiro Fukuma, PhD Supervisor, Chair

Prof. Edmud Soji Otabe

Prof. Om Prakash Sinha

Prof. Masonori Takabayashi

ABSTRACT

Neuromorphic Computing with Spin Hall Oscillators: Modelling and Leveraging of nonlinear magnetization dynamics for classification and prediction computational tasks

(スピンホール発振器を使用した脳型計算処理**:** 分類および予測タスクのための非線形磁化ダイナミクスのモデル化と活用）

The landscape of information technology has been profoundly reshaped by the emergence of Artificial Intelligence (AI) and Machine Learning (ML), catalyzing transformative shifts across industries and enhancing human interactions [1]. These advancements mark a shift from conventional information processing to a new era of intelligent computing. Central to this transformation is the capability of computers to analyze voluminous datasets, extracting significant insights that empower informed decision-making. Notably, OpenAI's ChatGPT exemplifies this paradigm shift, a language model adept at contextual comprehension and human-like responsiveness. Brain-inspired Artificial Neural Networks (ANNs) strive to emulate human brain information processing through multiple processing layer computational models that can learn representations of data at various levels of abstraction. Executed on Von Neumann architectures, these ANNs employ algorithms like backpropagation to fine-tune weights and replicate learning mechanisms [2]. Nonetheless, the journey of ANNs is obstructed by scalability challenges that demand innovative solutions to bridge the gap between artificial and biological intelligence. The limitations of the Von Neumann architecture, where processing and memory exist as distinct entities, constrain traditional ANN implementations, leading to processing power limitations and functional constraints. The renowned "von Neumann bottleneck" obstructs data-intensive operations, hindering parallelism and inducing inefficiencies in real-time data processing and AI inference.

The evolution beyond Von Neumann architecture investigates different computing paradigms like neuromorphic, quantum, and unconventional methods. Spiking neural networks and memristors are two examples of neuromorphic devices that attempt to combine memory and processing to mimic the unified functionality of the human brain [3]. These devices simulate synapses and neurons found in biological systems, allowing for unified communication and parallel processing. The offloading of intensive computational tasks from the conventional computing architecture is where tailored neuromorphic components show promise for real-time computations. Such components have great potential for real-time computations and are ideal for memory-constrained gadgets like wearables, Internet of Things (IoT) devices, and embedded systems [4]. While ANNs excel in classification and pattern recognition tasks, incorporating dedicated neuro-inspired computing units mandates efficient signal processing, seamless Complementary Metal-Oxide-Semiconductor (CMOS) circuit integration, and adaptability with the existing machine learning algorithms. Integrating specialized computing elements with CMOS technology is pivotal in bridging the gap between conventional and unconventional computing paradigms. The implementation of specialized inference or feature extraction computing units holds the potential to significantly mitigate energy costs associated with feature mapping, a substantial proportion of current ANN expenditures [5]. The promise of spintronic devices, with their inherent nonlinear magnetization dynamics, as prospective candidates for neuromorphic hardware and unconventional computing components is compelling [6,7]. Spin torque oscillators, comprising spin transfer torque oscillators and spin Hall oscillators, showcase remarkable capabilities in classification and recognition tasks.

This thesis investigates the realm of information processing capability of Spin Hall Oscillators (SHOs) using macrospin-level (micromagnetic) simulations. SHOs emerge as generators of high-frequency microwave signals and nonlinear magnetization dynamics, presenting opportunities in simple signal processing endeavors. The research aims to model SHO(s) as specialized computing component(s), adept at efficient signal processing, reduced computations, embracing real-time inference capabilities, and serving memory-constrained devices. Furthermore, the investigation extends into Reservoir Computing (RC) strategies, bolstering SHOs' information-handling prowess. To achieve these objectives, certain restrictions are imposed, guiding the course of the research:1. Designing computing components to offload computational complexity while minimizing memory utilization, 2. Seamless integration with conventional signal processing techniques to align with current computing architectures, 3. Ensuring real-time operation and suitability for memory-constrained devices to cater to diverse application scenarios.

The study commences by showcasing SHOs' capability in classification tasks, adaptable for processing binary data inputs nonlinearly, enabling real-time feature extraction and classification. When combined with frequency domain filtering, input driven magnetization dynamics can be used to classify 4-bit binary digit patterns with a single floating-point output. This novel methodology, which eradicates the need for weight storage in the initial layer of computation, shows the capability of SHO's self-computation based on the order of inputs in the pattern. The methodology is applied to classify handwritten digit images from the Modified National Institute of Standards and Technology database [8]. In a simple linear regression model, the model achieves an accuracy of 83.1%, demonstrating the effectiveness of the SHO for real-time and on-device neuromorphic framework.

Furthermore, the research also investigates the use of a single SHO in reservoir computing, a machine learning framework that uses recurrently connected nodes to effectively process sequential data. Memory capacity (MC) of a reservoir is a measure of the amount of data it can store and use over time [9]. It is important for a variety of reservoir computing tasks, such as time series prediction, nonlinear data transformation, and temporal pattern identification. We show that the reservoir's memory capacity and its use for temporal tasks are directly related. When SHO output magnetization dynamics include both transient state and limit cycle oscillations, the best reservoir computing results are obtained. The effectiveness of temporal tasks is revealed to be significantly correlated with reservoir memory capacity. The effect of input current pulse parameters on the memory capacity of SHOs is investigated. The results show an improvement trend with increasing pulse amplitude and width, peaking in the 4.5–5.0 range. Nonlinear Autoregressive Moving Average Mode 2 (NARMA2) time series prediction task and the three-bit parity task are used to test SHO's performance as a reservoir computing system, confirming a strong correlation between memory size and temporal task performance.

Finally, the nonlinear dynamics of the magnetization, high-frequency oscillations, and cooperative behavior enabled by dipolar coupling of SHOs (dSHO) is investigated. The use of an array of dSHOs is a novel approach to enhance memory capacity in the spatial and temporal domains. Dipolar coupling introduces a cooperative behavior component, allowing interaction and storing and retrieving of complex temporal patterns. The systems' memory capacity can effectively be increased to 10 by using dSHOs for spatial domain extension, which also improves their ability to predict. Significantly, the approach substantially expedites large-scale data processing, speeds up prediction and classification. This accelerated functionality holds the promise of immediate decision-making in domains such as self-driving vehicles and financial predictions.

In conclusion, the integration of Spin Hall Oscillators (SHOs) marks a pivotal point in computing by combining neuromorphic computing with existing computing architecture. This results in computing that is effective, flexible, and memory-efficient. To maximize the computing potential of SHOs, we concentrated on machine learning adaptability and the efficient signal processing capability of CMOS integration. By enabling more effective, adaptable systems that go in reservoir computing and beyond conventional approaches, these devices have the potential to fundamentally alter the computing landscape. Intelligent computing is made possible by the adaptability of SHOs to machine learning algorithms, enabling pattern recognition and decision-making in applications like image recognition, robotics, and autonomous vehicles.

References:

[1] B. Goertzel, " *J. Artif. Gen. Intell.*, vol. 5, no. 1, pp. 1–48, Dec. 2014.

[2] Y. LeCun, Y. Bengio, and G. Hinton, *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015.

[3] D. Ielmini and S. Ambrogio, *Nanotechnology*, vol. 31, no. 9, pp. 092001, Feb. 2020.

[4] C. D. Schuman *et al.*, *Nat. Comput. Sci.*, vol. 2, no. 1, pp. 10–19, Jan. 2022.

[5] W. Zhang *et al.*, *Nat. Electron.*, vol. 3, no. 7, pp. 371–382, Jul. 2020.

[6] J. Grollier *et al., Nat. Electron.*, vol. 3, no. 7, pp. 360–370, Mar. 2020.

[7] J. Zhou and J. Chen, *Adv. Electron. Mater.*, vol. 7, no. 9, pp. 2100465, Sep. 2021.

[8] Y. LeCun, C. Cortes, and C. J. C. Burges, The MNIST Database of Handwritten Digits.

[9] H. Jaeger, German Nat. Res. Centre Inf. Technol., St. Augustin, Germany, GMD Rep. 148, 2001.

LIST OF PUBLICATIONS

This thesis is based on the results derived from the simulations reported in the following papers:

1. Classification Tasks Using Input Driven Nonlinear Magnetization Dynamics in Spin Hall Oscillator; John Rex Mohan, Arun Jacob Mathew, Kazuma Nishimura, Ruoyan Feng, Rohit Medwal, Surbhi Gupta, Rahdeep Singh Rawat, and Yasuhiro Fukuma. *Sci Rep* vol. 13, no. 7909, (2023).

2. Evaluation of Memory Capacity and Time Series Prediction Using a Spin Hall Oscillator as Reservoir; Arun Jacob Mathew, John Rex Mohan, Ruoyan Feng, Rohit Medwal, Surbhi Gupta, Rajdeep Singh Rawat, and Yasuhiro Fukuma. *IEEE Transactions on Magnetics* vol. 59, no. 6, pp. 1-5, (2023).

Other publications not related to this thesis:

1. Anisotropy-Assisted Bias-Free Spin Hall Nano-Oscillator; Manna, Sourabh, Rohit Medwal, Surbhi Gupta, John Rex Mohan, Yasuhiro Fukuma, and Rajdeep Singh Rawat; *Appl. Phys. Lett.*, vol. 122, no. 7, pp. 72401, (2023).

2. Piezoelectric Strain Control of Terahertz Spin Current; Chaurasiya, Avinash, Ziqi Li, Rohit Medwal, Surbhi Gupta, John Rex Mohan, Yasuhiro Fukuma, Hironori Asada, Elbert E. M. Chia, and Rajdeep Singh Rawat. A*dv. Opt. Mater.*, vol. 10, no. 24, (2022).

3. Nonstoichiometric FePt Nanoclusters for Heated Dot Magnetic Recording Media; Mohan, John Rex, Rohit Medwal, Surbhi Gupta, Kriti Gogia, Joseph Vimal Vas, Rekha Gupta, Angshuman Deka, Rajdeep Singh Rawat, Annapoorni Subramanian, and Yasuhiro Fukuma. *ACS Appl. Nano Mater.*, vol. 4, no. 7, pp. 7079–7085 (2021).

CONTENTS

Chapter 1. Introduction

Chapter 1

Introduction

1. Introduction

In the current digital age, access to knowledge and data has become more prevalent than ever before. The advent of the internet and other digital technologies has resulted in an astonishing amount of data being generated, making information more accessible and transforming human interactions. The emergence of platforms like Google, Wikipedia, and Social Network Sites (SNS) has enabled people to contribute to and access information seamlessly, transcending geographical boundaries. In 2023, it is projected that an average of 120 zettabytes (120×10^{12} GB) of data will have been generated and consumed globally[1]. Amid these digital advancements, our society is experiencing a transformative journey from an information-centric state to an era characterized by intelligence and innovation. This transformation revolves around the realms of decision-making and problem-solving, where computing plays a pivotal role in society's evolution towards greater intelligence. With unprecedented precision, intelligent algorithms are now capable of deciphering complex patterns and optimizing solutions, thanks to the computing power of Artificial Neural Networks (ANNs)[2]–[10]. These applications span a wide range of tasks, from real-time data analysis to artificial intelligence (AI) processing, revolutionizing how society advances by enabling intelligence augmentation and automation in various fields, including self-driving cars, robotics, healthcare, and large-scale operations like urban planning and environmental monitoring[11]–[19]. For example, AI-enabled disease recognition using bio-signal patterns such as electrocardiogram (ECG), electroencephalogram (EEG), and x-ray imaging has revolutionized healthcare by enabling early disease detection and diagnosis [11], [20], [21].

1.1. The emergence of Intelligent algorithms and Artificial Neural Networks

The computing paradigm, centered around information processing, revolves around handling, storing, and analyzing enormous volumes of data. Its limitations, including data security, scalability, real-time processing, and data storage, have been addressed through technologies such as distributed computing frameworks, cloud computing, and data preprocessing techniques [22]–[25]. These tools empower individuals and organizations to efficiently manage the flow of information, store and process data, and derive valuable insights from large datasets. Figure 1.1 displays a typical data processing flow from edge devices to data storage and analysis. The goal of the new era of intelligent computing, powered by AI and machine learning (ML), is to equip systems with the capacity to comprehend, learn from, and draw conclusions from data [26].

AI refers to the development of computer systems capable of performing tasks that typically require human intelligence. By using sophisticated algorithms to find patterns, trends, and correlations, AI leverages the vast amount of big data available. Machines can now simulate decision-making processes similar to those of humans through techniques like deep learning, natural language processing, and reinforcement learning [3], [27]–[30]. A striking example of this transition is embodied by OpenAI's ChatGPT—a language model designed to understand context and respond with human-like proficiency [31]. This technology goes beyond presenting information; it simulates conversations, offers nuanced advice, and engages users in meaningful interactions.

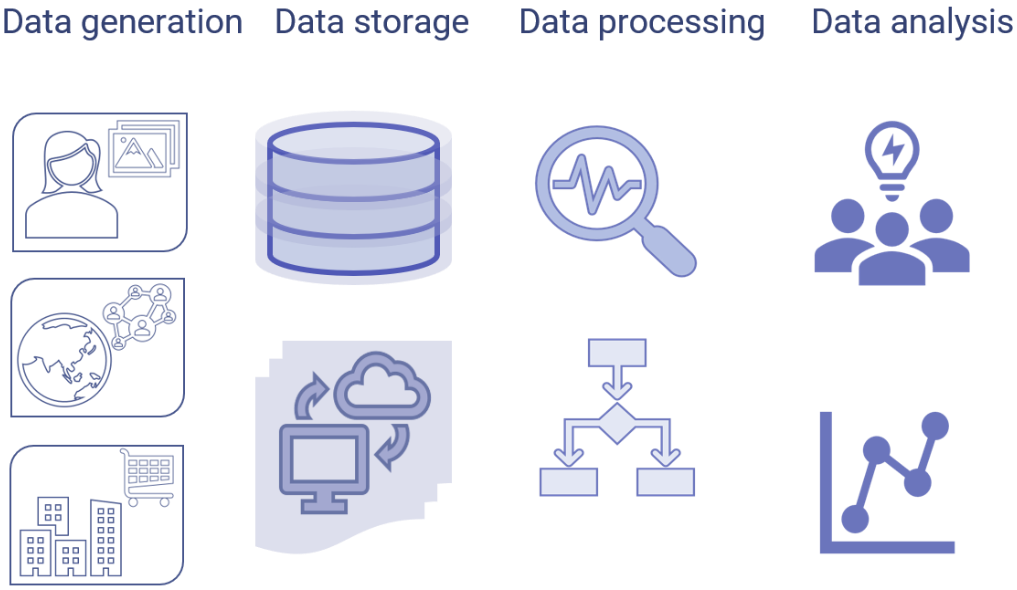


Fig. 1.1. Large data processing and analysis pipeline. Adapted from [32].

The true potential of AI and ML lies in brain-inspired neural networks, the ANNs, which aim to replicate the information processing seen in the human brain. They have revolutionized the way we process information. These networks consist of nodes connected to one another, resembling neurons in how they transmit signals and perform computations. The weight of each neuronal connection governs the strength of signal transmission. ANNs can change these weights based on learning from data, enabling them to perform tasks more effectively. Their strength lies in their ability to recognize intricate patterns, adapt to changes in data, and simultaneously process multiple pieces of information. They can automatically learn relevant features from raw data, including feature extraction and feature mapping. Additionally, ANNs enable end-to-end learning, where input data is directly mapped to output, eliminating the need for intermediary representations. In fields like image recognition and language translation, complex algorithms like convolutional neural networks (CNNs) and multilayer feedforward neural networks (FNNs) have demonstrated impressive capabilities. In ML, feature engineering is a crucial step. It involves selecting, transforming, or creating relevant features from the input data to help the model make accurate predictions[29][19]. Feature engineering often relies on domain knowledge of the experts. Deep learning (DL) models, particularly CNNs and recurrent neural networks (RNNs), can automatically learn relevant features from raw data. This reduces the need for manual feature engineering and makes DL suitable for tasks like image and speech recognition[33][19]. A simple data processing structure which shows the differences in the ML and DL models are shown in Fig.1.2.

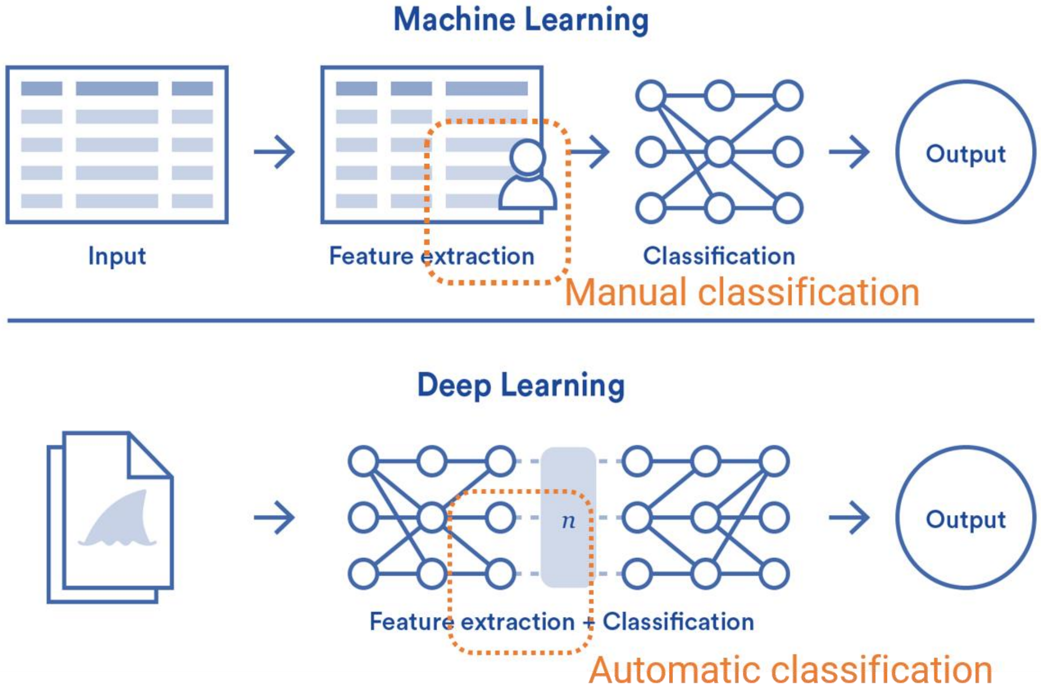


Fig. 1.2. Concept and data processing structures of artificial neural networks implemented in machine learning and deep learning models.

1.2. Challenges in ANN Computing: Algorithmic Disparities and Scalability implications

As AI and machine learning technology advances, a variety of new challenges arise, particularly in the areas of computation algorithms and hardware constraints. To recognize images, CNNs and FNNs interpret and analyze visual data in layers and features, closely resembling how humans perceive visual information. However, the complexity that makes these algorithms powerful also presents formidable difficulties. Extensive computational resources are required to train and optimize these algorithms due to the intricate network and connections they contain. Large datasets are necessary to train recognition models effectively, preventing overfitting and ensuring generalization to unexplored data. As the depth of feature extraction layer increases, deep neural networks demand significant processing power and storage capability.

Beyond algorithms and data, the underlying hardware that powers AI and ML also faces challenges. Current hardware based on complementary metal-oxide semiconductors (CMOS) is limiting the potential of AI and ML applications. Traditional computing's deterministic and rule-based structure struggles to mimic the cognitive adaptability and context awareness characteristic of human intelligence. The Von Neumann architecture, the foundation of digital computation, is approaching its limits. It features sequential data processing, memory hierarchies, and binary logic, which have limitations in addressing modern computing-related problems[10][34].

1. ***Processing Speed*****:** Deep learning models are computationally intensive, and the density scaling of transistors on Integrated Chips (ICs) is related to processing power. While multi-core structures with high-density transistors can perform parallel processing, power constraints limit their full-speed operation, slowing down chip performance.

2. ***Energy Conservation*****:** Power-hungry ANN algorithms pose a challenge for energy-efficient computing, especially for complex AI models such as Transformers used in natural language processing.

3. ***Memory Bandwidth*****:** Handling ANN algorithms, which involve complex calculations and large data volumes, requires substantial memory bandwidth. Conventional memory architectures like Dynamic Random-Access Memory (DRAM) and Static Random-Access Memory (SRAM) can become bottlenecks for data-intensive workloads.

4. ***Heat Dissipation*****:** Intensive computations generate heat, impacting hardware functionality and durability. CMOS technology struggles to dissipate the heat generated by such workloads.

To address these limitations, dedicated hardware ANN accelerators with parallel architectures, such as Graphical Processing Units (GPUs), Tensor Processing Units (TPUs), and Application Specific Integrated Circuits (ASICs), are being developed. GPUs, with their support for parallelism, are particularly well-suited for running Vector Matrix Multiplication (VMM) operations essential for training neural networks. TPUs and ASICs take a step further by optimizing circuits for specific neural network workloads, significantly improving processing efficiency[35][36].

1.3. Innovative Solutions: Neuromorphic Computing and Beyond

The journey towards mitigating these disparities involves a two-fold approach: refining the algorithmic architecture of Artificial Neural Networks (ANNs) and exploring innovative hardware paradigms that mirror biological intricacies.

1.3.1. Algorithmic Refinements

Efforts are directed towards designing ANNs that closely align with the biological essence of neural networks. This entails embracing spiking neural networks and event-driven computation, thus capturing the temporal nuances of neural communication. The objective is to minimize the gap between artificial and biological neural processing to enhance scalability and efficiency.

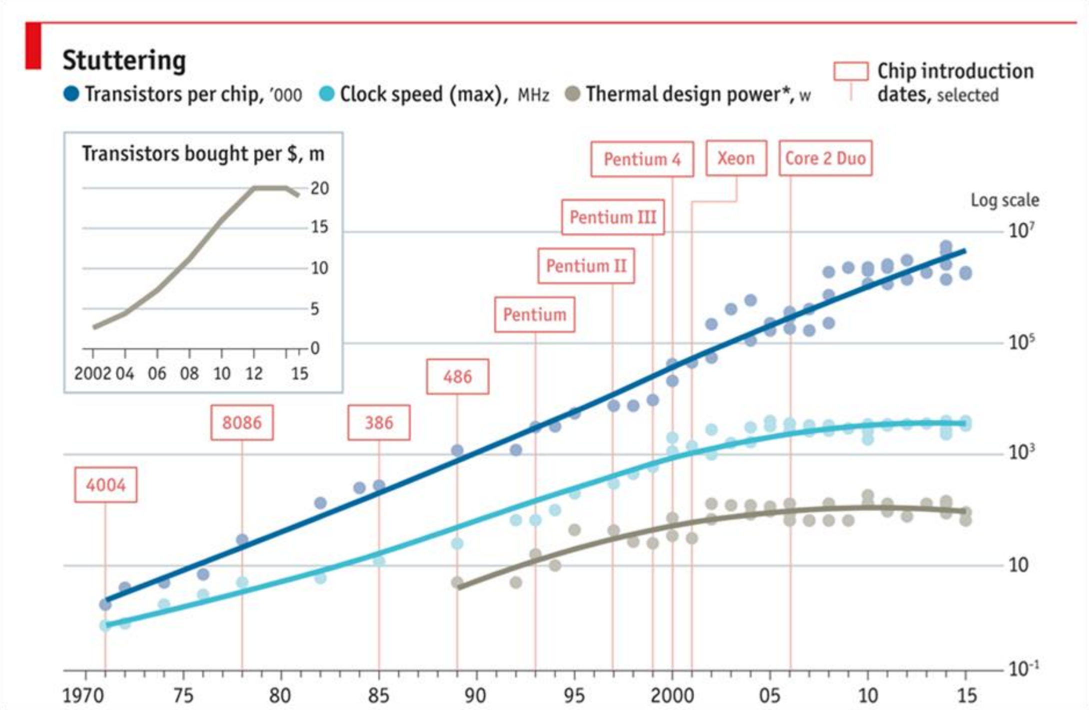


Fig 1.3. Transistor scaling and performance gap yearly trend [37].

1.3.2. Beyond Binary Logic - A Spectrum of Possibilities

On the hardware limitations front, researchers are exploring unconventional computing paradigms such as quantum computing, optical computing, Stochastic computing, Neuromorphic computing, and Unconventional computing. These paradigms offer unique capabilities and challenges.

1. *Quantum Computing*:

Quantum computing exploits the principles of quantum mechanics to perform complex calculations at extraordinary speeds. Quantum bits or "qubits" can exist in multiple states simultaneously, ushering in the potential for exponential speedup in certain computations. However, the nascent nature of quantum technology presents formidable challenges in maintaining qubit stability, error correction, and large-scale integration.

2. *Unconventional Computing*:

Unconventional computing paradigms explore computation beyond the confines of binary logic. These encompass approaches like DNA computing, optical computing, and analog computing. While they hold the promise of energy efficiency and novel problem-solving capabilities, they often grapple with issues of scalability, robustness, and compatibility with existing CMOS infrastructure.

3. *Neuromorphic Computing*:

Neuromorphic computing draws inspiration from the human brain's structure and function. It uses artificial neural networks with specialized hardware designed to emulate the behavior of neurons and synapses. This approach is particularly suitable for AI and ML tasks due to its ability to process complex data patterns efficiently and with low power consumption.

1.4. Neuromorphic Computing as a Forerunner.

Among alternative computing strategies, neuromorphic computing stands out as one that can successfully handle the difficulties presented by AI applications. It emulates the brain's architecture, which excels at tasks like pattern recognition and sensory perception. There are different ways in which the hardware approaches are implemented.

1. *Specialized Hardware for ANNs*

In this approach, specialized neuromorphic hardware architecture is developed to handle ANN tasks. Prime examples include Loihi by Intel and TrueNorth by IBM[38][39][40]. These chips use spiking neural networks to carry out operations like image recognition quickly and effectively while using less power than conventional hardware.

2. ***Combining Neuromorphic and Traditional Computing***

Neuromorphic computing can also be used to offload particular tasks from existing computing architecture. A hybrid system might, for instance, use conventional hardware for general-purpose computations and neuromorphic hardware for things like image processing or pattern recognition.

3. ***Enhancing ANN Operations with Memristors***

Memristors are resistors with memory, capable of storing and processing information in the same device. They hold promise for accelerating certain computations, particularly vector multiplications, which are essential for neural network operations[41].

4. ***Accelerating computational tasks with dedicated Feature Extraction layers***

Feature extraction is a pivotal step in AI tasks such as image recognition. The computational demands associated with feature extraction account for approximately 80% of the computational cost in implementing ANN algorithms. Take, for instance, a scenario in which a CPU is used for convolutional operations on high-resolution images in image recognition tasks. During this procedure, a filter is applied to a few discrete areas of the image to extract features from them. A number of data transmission from memory are required as the filter moves across the image. Although CPUs are incredibly fast at computation, the time needed to retrieve data from memory adds a lot of overhead and reduces performance as a whole. Convolutional operations can be carried out concurrently by FPGAs, which significantly reduces the need for data fetching[42]. By utilizing the parallel processing abilities of FPGAs, this method accelerates the feature extraction process overall. However, integrating FPGAs with CPUs for neuromorphic computing necessitates addressing both hardware and software limitations. It can be challenging to get the FPGA and CPU to communicate and transfer data effectively. Data synchronization, memory management, and communication protocols all need to be carefully considered when coordinating the execution of tasks between these two different architectures. Furthermore, FPGA programming calls for specialized knowledge and abilities. FPGAs require familiarity with hardware description languages and low-level hardware design, in contrast to traditional CPUs, which are easier to use and have well-established programming paradigms.

1.5. Innovations in Neuromorphic Hardware Implementations

Implementations of neuromorphic hardware draw inspiration from various scientific fields and specializations, including biology, chemistry, physics, nanomaterials, and spintronics [36], [43]–[50]. The diverse landscape of neuromorphic hardware research offers unique benefits and challenges in each approach. Here, we summarize some key principles of these implementations.

1. *Biomaterials-based Systems*:

Recent research demonstrates the potential of biomaterial-based memories for non-volatile data storage to simulate synaptic functions. Molecular self-assembly allows for the programming of molecular behavior by selecting optimal DNA sequences to encode information and recognize patterns. While these models accurately represent neural behavior, effectively implementing these intricate biological mechanisms remains a challenge.

2. *Chemical Neuromorphic Systems*:

Chemical systems mimic neural behaviors by utilizing reactions and diffusion. Computation in chemical neuromorphic hardware relies on chemical interactions. Controlling reaction rates and ensuring noise stability are formidable tasks.

3. *Photonic Neuromorphic Systems*:

Photonic neuromorphic systems employ light to encode and transmit information, enabling high-speed communication and parallel processing. Photonic neural networks excel at complex calculations with minimal energy consumption. Challenges include integrating photonics with electronic systems and addressing signal noise.

4. *Physical Substrates and Nanomaterials*:

Neuromorphic systems can be realized using physical substrates like memristors or phase-change materials, effectively mimicking synapses and storing weights. Nanomaterials such as graphene and carbon nanotubes offer high conductivity and mechanical flexibility. However, challenges lie in reproducibility, variability in manufacturing, precise control of physical properties, and integration with conventional electronics.

5. *Spintronics-Based Systems*:

In spintronics, electron spin encodes and processes information. Components like magnetic hard disks and magnetic tunnel junctions (MTJs) have proven their adaptability to current CMOS technology, offering a promising route to neuromorphic hardware implementation.

1.6. Research Motivations and Thesis Focus

Expedited research is currently underway to find innovative ways to close the gap between these algorithms' computational requirements and the constraints of conventional hardware architectures. Specialized or self-contained computing units based on physical or neuromorphic principles that can be integrated into current computing architectures represent one promising direction. For example, specialized computing units within the CPU or adaptable to CPU structure can be designed to offload complex calculations, reducing processing times and freeing up CPU resources for other tasks. The goal of this integration is to take advantage of the advantages of both conventional CMOS systems and specialized components. Due to this, these components should be created to operate in perfect harmony with current CMOS fabrication techniques and signal processing tools. This focused approach may reduce overall computational complexity, improve computing system performance, and speed up real-time computations on memory-constrained devices.

In the context of image recognition and pattern classifications, implementation of specialized inference or feature extraction computing units holds the potential to significantly mitigate energy costs associated with feature mapping, a substantial proportion of current ANN expenditures [5]. The promise of spintronic devices, with their inherent nonlinear magnetization dynamics, as prospective candidates for neuromorphic hardware and unconventional computing components is compelling [6,7]. Experiments and theoretical models, based spin torque oscillators, comprising spin transfer torque oscillators and spin Hall oscillators have shown success in classification and recognition tasks.

Thesis Focus: Investigating Spin Hall Oscillators (SHOs) for information processing capabilities

This thesis investigates the realm of information processing capability of Spin Hall Oscillators (SHOs) using micromagnetic simulations. SHOs emerge as generators of high-frequency microwave signals and nonlinear magnetization dynamics, presenting opportunities in simple signal processing endeavors. The primary aim is to model SHO(s) as specialized computing component(s), adept at efficient signal processing, reduced computations, embracing real-time inference capabilities, and serving memory-constrained devices. Furthermore, the investigation extends into Reservoir Computing (RC) strategies, a category of ANN, bolstering SHOs' information-handling prowess.

To achieve these objectives, certain restrictions are imposed, guiding the course of the research:

1.Designing computing components to offload computational complexity while minimizing memory utilization,

2. Seamless integration with conventional signal processing techniques to align with current computing architectures.

3. Ensuring real-time operation and suitability for memory-constrained devices to cater to diverse.

1.7. Thesis organization

This thesis is structured as follows;

Chapter 2: Background: Neuromorphic Computing and Spintronics

In this chapter, the theoretical foundations and general aspects of spintronics are discussed. The physical principles of spin-orbit interaction leading to the spin Hall effect and anisotropic magnetoresistance are introduced. Additionally, an overview of the principles of neural networks and models relevant to this work is provided.

Chapter 3: Methods and Materials

This section describes the methods and materials used for modeling and investigating magnetic devices and neural networks in this thesis.

Chapter 4: Classification of binary digit patterns using spin Hall oscillators

Chapter 4 presents the strategies and results of the self-computing unit-based spin Hall oscillator model for binary digit input pattern classification. The methodology is applied to classify handwritten digit images from the Modified National Institute of Standards and Technology database, and its impact on image recognition is discussed.

Chapter 5: Spin Hall Oscillators in Reservoir Computing

Chapter 5 explores the use of spin Hall oscillators in reservoir computing for processing sequential data. It delves into the generation of input-driven magnetization dynamics and analyzes its impact on reservoir computing metrics.

Chapter 6: Enhancing information processing capability of SHO – magnetic dipolar coupling approach

Chapter 6 investigates dipolar-coupled spin Hall oscillators in an array structure. The chapter explores variations in magnetization dynamics and discusses their impact on reservoir computing.

Summary and Conclusion

This section serves as the conclusion of this thesis, summarizing the contributions of this research for the neuromorphic computing.

References

[1] Taylor Petroc, "Total Data Volume Worldwide 2010-2025," *Statista*, 2022. https://www.statista.com/statistics/871513/worldwide-data-created (accessed Aug. 14, 2023).

[2] B. Goertzel, "Artificial General Intelligence: Concept, State of the Art, and Future Prospects," *J. Artif. Gen. Intell.*, vol. 5, no. 1, pp. 1–48, Dec. 2014, doi: 10.2478/jagi-2014-0001.

[3] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[4] C. C. Aggarwal, *Neural Networks and Deep Learning*. Cham: Springer International Publishing, 2018.

[5] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo *et al.*, "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems." 2015, [Online]. Available: https://www.tensorflow.org/.

[6] L. Buitinck *et al.*, "API design for machine learning software: experiences from the scikit-learn project," pp. 1–15, 2013, [Online]. Available: http://arxiv.org/abs/1309.0238.

[7] A. Paszke *et al.*, "PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems," *NeurIPS*, no. NeurIPS, pp. 8026–8037, 2019.

[8] M. H. Huang and R. T. Rust, "Artificial Intelligence in Service," *J. Serv. Res.*, vol. 21, no. 2, pp. 155–172, 2018, doi: 10.1177/1094670517752459.

[9] S. Haykin, "Neural Networks and Learning Machines," in *Prentice Hall*, vol. 1–3, Elsevier, 2008, pp. 612–620.

[10] K.-L. Du and M. N. S. Swamy, *Neural Networks and Statistical Learning*, no. 2007. London: Springer London, 2014.

[11] P. Rajpurkar, E. Chen, O. Banerjee, and E. J. Topol, "AI in health and medicine," *Nat. Med.*, vol. 28, no. 1, pp. 31–38, Jan. 2022, doi: 10.1038/s41591-021-01614-0.

[12] H. Wang *et al.*, "Scientific discovery in the age of artificial intelligence," *Nature*, vol. 620, no. 7972, pp. 47–60, 2023, doi: 10.1038/s41586-023-06221-2.

[13] P. Pataranutaporn *et al.*, "AI-generated characters for supporting personalized learning and well-being," *Nat. Mach. Intell.*, vol. 3, no. 12, pp. 1013–1022, 2021, doi: 10.1038/s42256-021-00417-9.

[14] I. Grossmann, M. Feinberg, D. C. Parker, N. A. Christakis, P. E. Tetlock, and W. A. Cunningham,

"AI and the transformation of social science research," *Science (80-.).*, vol. 380, no. 6650, pp. 1108–1109, 2023, doi: 10.1126/science.adi1778.

[15] D. Tuia *et al.*, "Perspectives in machine learning for wildlife conservation," *Nat. Commun.*, vol. 13, no. 1, p. 792, Feb. 2022, doi: 10.1038/s41467-022-27980-y.

[16] Z. Yao *et al.*, "Machine learning for a sustainable energy future," *Nat. Rev. Mater.*, vol. 8, no. 3, pp. 202–215, 2023, doi: 10.1038/s41578-022-00490-5.

[17] R. Vinuesa *et al.*, "The role of artificial intelligence in achieving the Sustainable Development Goals," *Nat. Commun.*, vol. 11, no. 1, pp. 1–10, 2020, doi: 10.1038/s41467-019-14108-y.

[18] D. Silvestro, S. Goria, T. Sterner, and A. Antonelli, "Improving biodiversity protection through artificial intelligence," *Nat. Sustain.*, vol. 5, no. 5, pp. 415–424, 2022, doi: 10.1038/s41893-022-00851-6.

[19] I. H. Sarker, "Machine Learning: Algorithms, Real-World Applications and Research Directions," *SN Comput. Sci.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.

[20] S. C. Huang, A. Pareek, M. Jensen, M. P. Lungren, S. Yeung, and A. S. Chaudhari, "Self-supervised learning for medical image classification: a systematic review and implementation guidelines," *npj Digit. Med.*, vol. 6, no. 1, 2023, doi: 10.1038/s41746-023-00811-0.

[21] D.-J. Choi, J. J. Park, T. Ali, and S. Lee, "Artificial intelligence for the diagnosis of heart failure," *npj Digit. Med.*, vol. 3, no. 1, p. 54, Apr. 2020, doi: 10.1038/s41746-020-0261-3.

[22] X. Sun, Y. He, D. Wu, and J. Z. Huang, "Survey of Distributed Computing Frameworks for Supporting Big Data Analysis," *Big Data Min. Anal.*, vol. 6, no. 2, pp. 154–169, Jun. 2023, doi: 10.26599/BDMA.2022.9020014.

[23] J. Dean and S. Ghemawat, "MapReduce," *Commun. ACM*, vol. 51, no. 1, pp. 107–113, Jan. 2008, doi: 10.1145/1327452.1327492.

[24] Y. Demchenko, F. Turkmen, C. de Laat, C.-H. Hsu, C. Blanchet, and C. Loomis, "Chapter 2 - Cloud Computing Infrastructure for Data Intensive Applications," in *Big Data Analytics for Sensor-Network Collected Intelligence*, H.-H. Hsu, C.-Y. Chang, and C.-H. Hsu, Eds. Academic Press, 2017, pp. 21–62.

[25] A. Reeve, "Chapter 21 - Big Data Integration," in *Managing Data in Motion*, A. Reeve, Ed. Boston: Morgan Kaufmann, 2013, pp. 141–156.

[26] J. Fulcher and L. C. Jain, *Computational Intelligence: A Compendium*, vol. 115. Berlin,

Heidelberg: Springer Berlin Heidelberg, 2008.

[27] H. Li, “Deep learning for natural language processing: advantages and challenges,” *Natl. Sci. Rev.*, vol. 5, no. 1, pp. 24–26, Jan. 2018, doi: 10.1093/nsr/nwx110.

[28] M. Botvinick, J. X. Wang, W. Dabney, K. J. Miller, and Z. Kurth-Nelson, “Deep Reinforcement Learning and Its Neuroscientific Implications,” *Neuron*, vol. 107, no. 4, pp. 603–616, 2020, doi: 10.1016/j.neuron.2020.06.014.

[29] L. Alzubaidi *et al.*, *Review of deep learning: concepts, CNN architectures, challenges, applications, future directions*, vol. 8, no. 1. Springer International Publishing, 2021.

[30] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions,” *SN Comput. Sci.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.

[31] “Chatgpt,” ChatGPT, https://openai.com/chatgpt (accessed Dec. 6, 2023).

[32] S. Daily, “Components of modern data pipelines,” Software Engineering Daily, https://softwareengineeringdaily.com/2020/04/30/components-of-modern-data-pipelines/ (accessed Aug., 2021).

[33] B. Scholkopf *et al.*, “Toward Causal Representation Learning,” *Proc. IEEE*, vol. 109, no. 5, pp. 612–634, 2021, doi: 10.1109/JPROC.2021.3058954.

[34] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, “Memory devices and applications for in-memory computing,” *Nat. Nanotechnol.*, vol. 15, no. 7, pp. 529–544, Jul. 2020, doi: 10.1038/s41565-020-0655-z.

[35] C. D. Schuman, S. R. Kulkarni, M. Parsa, J. P. Mitchell, P. Date, and B. Kay, “Opportunities for neuromorphic computing algorithms and applications,” *Nat. Comput. Sci.*, vol. 2, no. 1, pp. 10–19, Jan. 2022, doi: 10.1038/s43588-021-00184-y.

[36] D. V Christensen *et al.*, “2022 roadmap on neuromorphic computing and engineering,” *Neuromorphic Comput. Eng.*, vol. 2, no. 2, p. 022501, Jun. 2022, doi: 10.1088/2634-4386/ac4a83.

[37] “After Moore’s law | technology quarterly,” The Economist, https://www.economist.com/technology-quarterly/2016-03-12/after-moores-law (accessed Aug., 2023).

[38] J. Pei *et al.*, “Towards artificial general intelligence with hybrid Tianjic chip architecture,” *Nature*, vol. 572, no. 7767, pp. 106–111, Aug. 2019, doi: 10.1038/s41586-019-1424-8.

[39] Y. Zhang *et al.*, "A system hierarchy for brain-inspired computing," *Nature*, vol. 586, no. 7829, pp. 378–384, Oct. 2020, doi: 10.1038/s41586-020-2782-y.

[40] P. A. Merolla *et al.*, "A million spiking-neuron integrated circuit with a scalable communication network and interface," *Science (80-.).*, vol. 345, no. 6197, pp. 668–673, Aug. 2014, doi: 10.1126/science.1254642.

[41] E. Chicca and G. Indiveri, "A recipe for creating ideal hybrid memristive-CMOS neuromorphic processing systems," *Appl. Phys. Lett.*, vol. 116, no. 12, 2020, doi: 10.1063/1.5142089.

[42] P. Antonik, *Application of FPGA to Real-Time Machine Learning*. Cham: Springer International Publishing, 2018.

[43] A. Adamatzky, *Advances in Unconventional Computing*, vol. 22. Cham: Springer International Publishing, 2017.

[44] P. Kumar, M. Jin, T. Bu, S. Kumar, and Y.-P. Huang, "Efficient reservoir computing using field programmable gate array and electro-optic modulation," *OSA Contin.*, vol. 4, no. 3, p. 1086, 2021, doi: 10.1364/osac.417996.

[45] B. J. Shastri, A. N. Tait, T. F. de Lima, M. A. Nahmias, H. T. Peng, and P. R. Prucnal, "Principles of Neuromorphic Photonics," *arXiv*, 2017, doi: 10.1007/978-1-4939-6883-1_702.

[46] V. K. Ingle and J. G. Proakis, *Digital Signal Processing Using MATLAB®, 3rd Edition*, vol. 3, no. 1. Cengage Learning, 2011.

[47] G. Finocchio, M. Di Ventra, K. Y. Camsari, K. Everschor-Sitte, P. Khalili Amiri, and Z. Zeng, "The promise of spintronics for unconventional computing," *J. Magn. Magn. Mater.*, vol. 521, p. 167506, Mar. 2021, doi: 10.1016/j.jmmm.2020.167506.

[48] J. Torrejon *et al.*, "Neuromorphic computing with nanoscale spintronic oscillators," *Nature*, vol. 547, no. 7664, pp. 428–431, Jul. 2017, doi: 10.1038/nature23011.

[49] J. Zhou and J. Chen, "Prospect of Spintronics in Neuromorphic Computing," *Adv. Electron. Mater.*, vol. 7, no. 9, p. 2100465, Jun. 2021, doi: 10.1002/aelm.202100465.

[50] J. Grollier, D. Querlioz, K. Y. Camsari, K. Everschor-Sitte, S. Fukami, and M. D. Stiles, "Neuromorphic spintronics," *Nat. Electron.*, vol. 3, no. 7, pp. 360–370, Jul. 2020, doi: 10.1038/s41928-019-0360-9.

Chapter 2

Background: Neuromorphic Computing and Spintronics

In this chapter, the theoretical foundations and an overview of the principles of neural networks and models relevant to this work are discussed. The physical principles micromagnetism and spintronics principles are introduced.

2.1. Artificial Neural Networks

An artificial neural network (ANN) is a mathematical representation of a network of neurons that can perform intricate functional mapping similar to the human brain. The brain, which mirrors the neural (nerve) network in the human nervous system, acts as the central processing unit, receiving and processing information to make decisions. The nervous system, with its various levels of organization as illustrated in the Fig, contributes to the overall functionality of the neural system [1]–[3]. This system encompasses individual glial cells and neurons at the molecular and cellular levels, which process and transmit information through electrical and chemical signals. Neurons form interconnected networks known as microcircuits or neural circuits, collaborating to perform specific functions. These local circuits, also termed neural networks, consist of interconnected brain areas, each contributing to the processing of specific types of information, such as sensory inputs, motor commands, and complex cognitive processes like perception, logic, language processing, and conscious awareness.

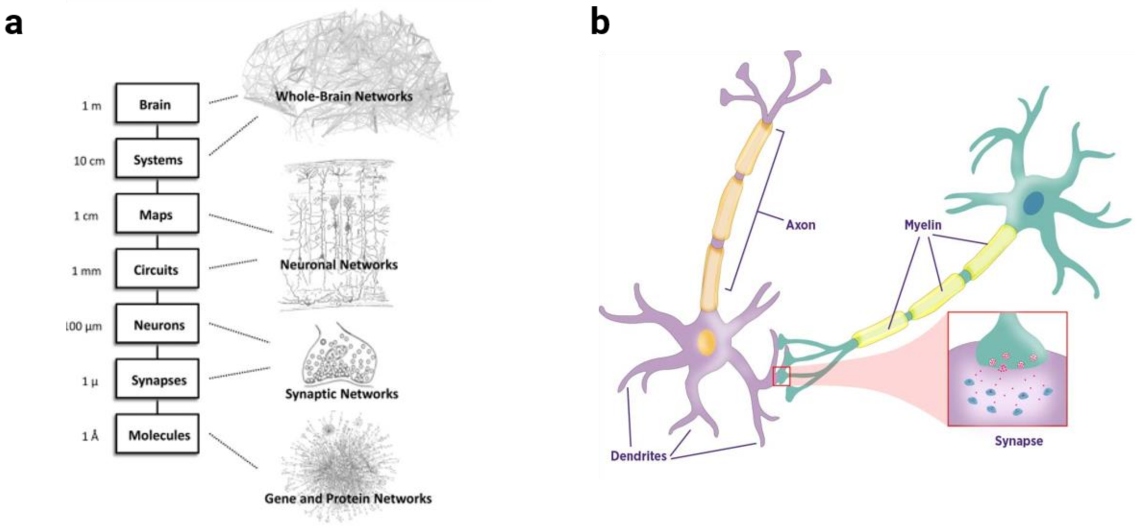


Fig. 2.1 a Schematics of structural level of nervous system, adapted from [1], **b** components of biological neuron.

2.1.1. Biological Neuron

Biological neurons are the fundamental units of the nervous system, responsible for processing and transmitting information through electrical impulses. They consist of three main components: dendrites, a cell body, and an axon. Dendrites receive input signals from other neurons, which are integrated by the cell body to determine whether the neuron should transmit an output signal through the axon. Synaptic plasticity, the ability of synapses to strengthen or weaken over time, plays a crucial role in learning and memory within the biological neural network. The human brain is estimated to comprise approximately 100 billion neurons and around 600 trillion synapses due to the intricate interconnections between neurons [1], [4], [5].

2.1.2. Artificial Neuron

Artificial neurons serve as fundamental building blocks within artificial neural networks, abstract representations inspired by the information processing and decision-making mechanisms of biological brains. These neurons encompass various models designed to capture essential neuronal features while being computationally tractable. They can be categorized into biologically plausible, biologically-inspired, neuron component models, spiking models, Integrate-and-fire models, and McCulloch–Pitts models. The McCulloch–Pitts neuron model, proposed by Warren McCulloch and Walter Pitts in 1943, forms the foundation for modern artificial neural networks and computational neuroscience [6]. The simplest form of an artificial neural network, the perceptron, consists of a single neuron unit with multiple inputs and a single output [7]–[11] Each neuron includes several key components:

(a) Inputs: These represent external environmental values or samples, serving as the neuron's inputs, often multiple binary values (0 or 1).

(b) Weights: Each input signal is associated with a synaptic weight that signifies its influence on the neuron's output.

(c) Linear Aggregator: This component computes the weighted sum of input signals through a dot product.

(d) Activation Threshold or Bias (h): The activation threshold determines the point at which the neurons activate.

(e) Activation Potential (u): Calculated as the difference between the linear aggregator output and the activation threshold, it decides whether the neuron produces an excitatory or inhibitory response.

(f) Activation Function (g): This function transforms the activation potential into an output signal, introducing nonlinearity.

(g) Output Signal (y): The final value produced by the neuron after applying the activation function, serving as input for other neurons in subsequent layers, facilitating the construction of multi-layered neural networks.

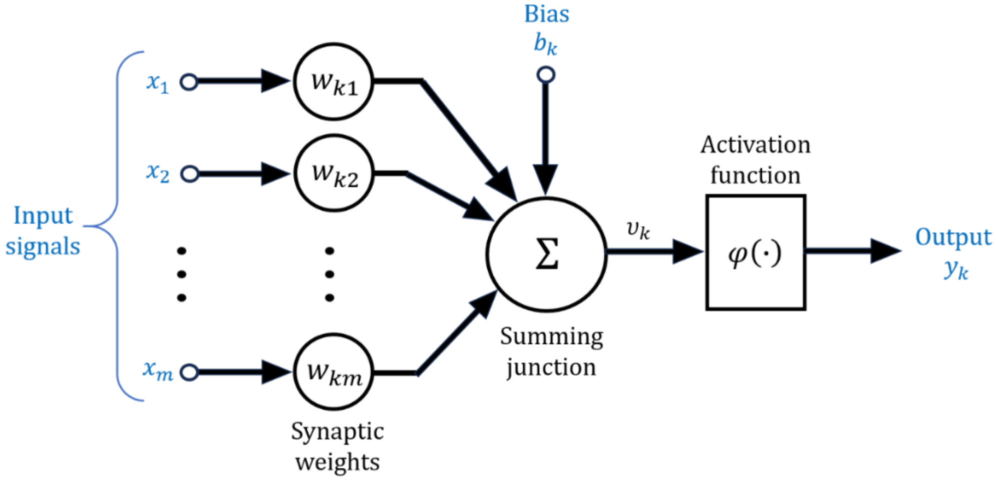


Fig. 2.2 Schematics of a perceptron model artificial neuron.

In mathematical terms, the neuron *n* can be written as,

$$u_n = \sum_{j=1}^{m} w_{nj} x_j \ , \tag{2.1}$$

$$y_n = f(u_n + b_n), \tag{2.2}$$

Where, $x_1, x_2, x_3, \dots x_m$ are the input signals, $w_{n1}, w_{n2}, w_{n3}, \dots w_{nm}$ are the synaptic weights of neuron n, u_n is the linear aggregator output, b_n is the bias, f is the activation function and y_n is the output signal of the neuron.

2.2. Perceptron and Activation functions

The perceptron model, a fundamental concept in artificial neural networks (ANNs), employs activation functions to introduce non-linearity. Activation functions determine a neuron's output based on its weighted inputs [8], [12], [13]. Common activation functions used in perceptron models include:

1. Step Function (Binary Threshold): The step function yields binary output (0 or 1) depending on whether the weighted sum of inputs surpasses a threshold. It is often employed in binary classification problems.

$$f(x) = \begin{cases} 1 \text{ if } x \ \geq 0 \\ 0 \ \text{ if } x \ < \ 0 \end{cases} \tag{2.3}$$

2. Sigmoid Function (Logistic Function): The sigmoid function compresses the weighted sum of inputs into the (0, 1) range, offering smooth, differentiable outputs. It suits tasks necessitating a smooth class transition.

$$f(x) = \frac{1}{1 + e^{-x}} \tag{2.4}$$

3. Hyperbolic Tangent Function (Tanh): Similar to the sigmoid, the tanh function maps inputs to the (-1, 1) range, addressing the vanishing gradient problem more effectively due to its zero-centered nature.

$$f(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \tag{2.5}$$

4. Rectified Linear Unit (ReLU): ReLU is the predominant activation function in deep learning. It enhances training efficiency and mitigates the vanishing gradient problem, enabling faster learning. Variants like Leaky ReLU, Parametric ReLU, and Exponential Linear Unit (ELU) further address gradient issues. The basic ReLU can be expressed as:

$$f(x) = \max(0, x) \tag{2.6}$$

5. Softmax Function: The softmax function is a commonly used activation function in multi-class classification problems. It takes as input a vector of real numbers and transforms them into a probability distribution over multiple class. Mathematically, the sigmoid function as below where $\boldsymbol{Z}$ is vector of inputs to output layer and j indexes the output units from 1, 2, 3,..,k.

$$\sigma(\boldsymbol{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} k} \tag{2.7}$$

The selection of an activation function hinges on the specific problem, network architecture, and empirical testing. Experimentation often guides the choice of the most suitable activation function for a given task as illustrated in Fig. 2.4.

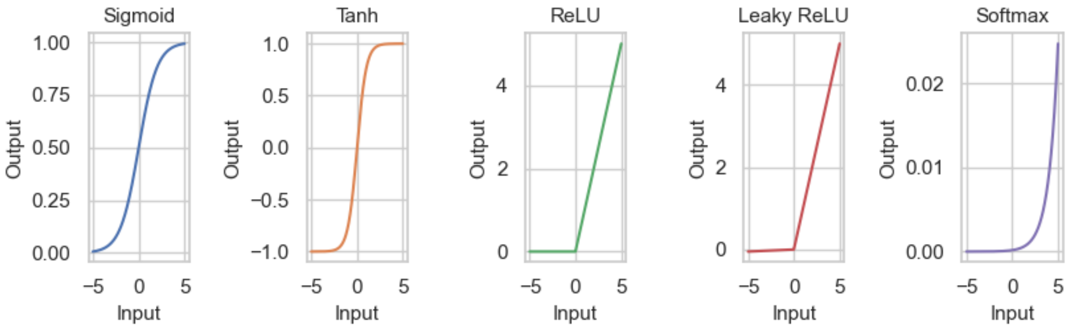


Fig. 2.3 Graphical representation of various activation functions.

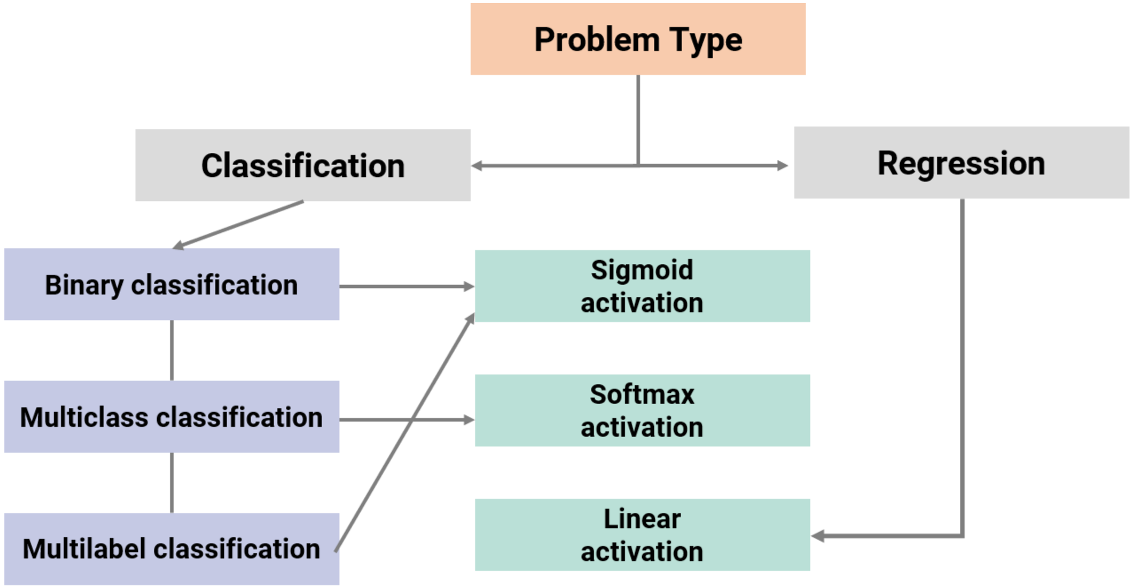


Fig. 2.4 Classes of activation functions and their specific usage to the computational problems.

2.3. Artificial Neural Network Architectures

ANNs can adopt diverse architectures and models tailored to address specific problem types. The organization of network neurons is closely tied to the learning algorithm employed for network training. Some relevant network architectures include:

2.3.1. Feedforward Networks (FNNs)

FNNs are the simplest form of neural networks, rooted in the perceptron model. They comprise an input layer, one or more hidden layers, and an output layer. These networks are devoid of loops or feedback connections, following a "feedforward" mechanism. The number of hidden layers can be adjusted based on problem complexity. When multiple hidden layers are involved, it is termed a Multilayer Perceptron (MLP) [8], [12]. In MLPs, data flows unidirectional from the input layer through the hidden layers to the output layer. Each neuron within an FNN applies an activation function to the weighted sum of its inputs. FNNs are typically trained via backpropagation and gradient descent algorithms, aiming to minimize a loss function measuring the divergence between predicted and target outputs. FNNs exhibit versatility and find applications in various machine learning tasks, including classification (e.g., sentiment analysis, image recognition) and pattern recognition (e.g., speech recognition, natural language processing) [14][15]. Importantly, FNNs excel in approximating complex functions. To ease FNN development and training, deep learning frameworks such as TensorFlow, PyTorch, and Keras offer pre-implemented layers and optimization algorithms [16]–[18]. However, large FNNs with numerous parameters often demand substantial computational power, with Graphics Processing Units (GPUs) and specialized hardware like Tensor Processing Units (TPUs) commonly utilized for accelerated training [19][20].

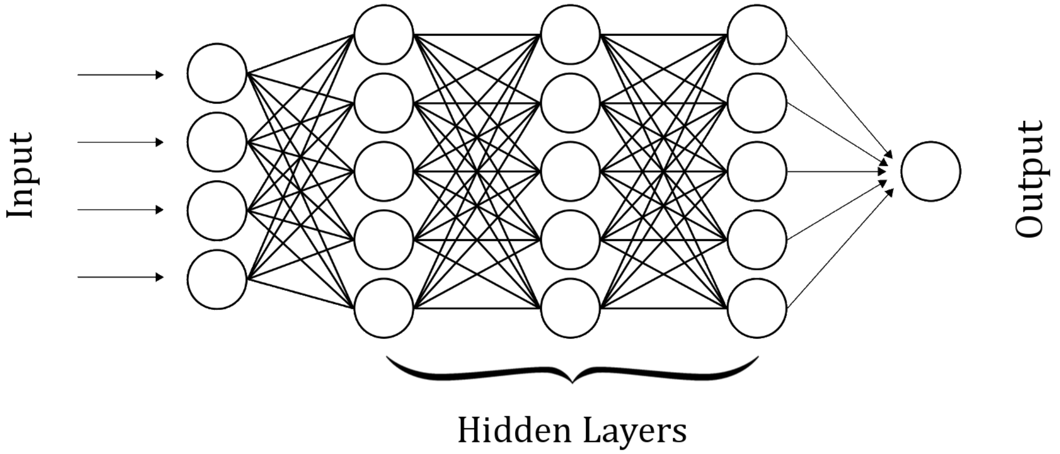


Fig. 2.5 Network architecture of Feedforward Neural Network.

2.3.2. Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) represent a class of artificial neural networks designed specifically for handling sequential data. In contrast to feedforward neural networks, RNNs possess recurrent connections, allowing them to maintain memory-like capabilities, making them particularly well-suited for tasks involving sequences or time-series data. Perceptron units can be incorporated into recurrent networks by introducing a temporal factor in the computations, assuming that the activation of each unit consumes one-time unit. For instance, if the input arrives at time t, the result is produced at time t + 1. This process is illustrated in Fig. 2.6. Essentially, any feed-forward network with instantaneous computation at the nodes can be replicated using networks of units with delays. The defining characteristic of RNNs is their recurrent connections. Each neuron within an RNN maintains a hidden state that encodes information about prior elements in the sequence. This hidden state is continually updated at each time step, effectively serving as a form of short-term memory [21][22]. RNNs are invaluable when the order of data elements carries significance. They excel in a variety of tasks such as natural language processing (NLP), speech recognition, video analysis, and time series forecasting. The simplest form of RNN is often referred to as a Vanilla RNN, where the hidden state at each time step depends on the current input and the preceding hidden state. However, Vanilla RNNs are plagued by the vanishing gradient problem, constraining their ability to capture long-range dependencies. Long Short-Term Memory (LSTM) networks, a more advanced form of RNN, effectively tackle the vanishing gradient issue by using gated units to regulate information flow through the network [23] [24]. This enables LSTMs to capture long-range dependencies in data. Gated Recurrent Units (GRUs), while similar to LSTMs, feature a simpler architecture with fewer gates [25]. They are computationally less demanding than LSTMs while still delivering strong performance.

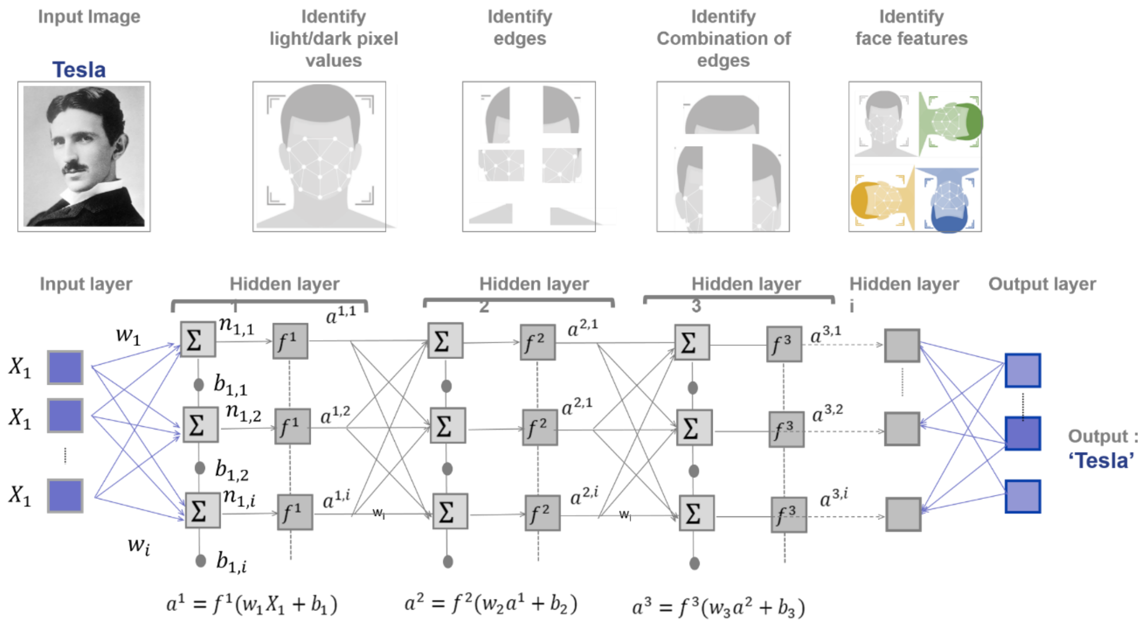


Fig. 2.6 Input data flow and processing structure in a recurrent neural network.

2.3.3. Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specialized category of artificial neural networks primarily designed for tasks related to computer vision, image analysis, and spatial data processing [26]–[30]. Inspired by the visual processing mechanisms in the human brain, CNNs have proven highly effective in various tasks involving grid-like data, such as images. Furthermore, they have demonstrated utility in domains like natural language processing. CNNs were conceived to mimic the visual perception process in animals, particularly the human visual system [25], [26]. In the human visual cortex, neurons respond to specific regions in the visual field, enabling them to detect patterns, edges, and textures in images. CNNs employ convolutional layers to autonomously and adaptively learn spatial hierarchies of features from input data. These layers comprise filters or kernels that slide over the input data, performing convolution operations to detect patterns and features. Typically, lower layers focus on recognizing basic features like edges and corners, while higher layers combine these features to recognize more complex patterns, shapes, and objects. Pooling layers are used to reduce the spatial dimensions of feature maps generated by convolutional layers while preserving essential information. Common pooling operations include max-pooling and average-pooling. Following the convolutional and pooling layers, CNNs typically employ one or more fully connected layers, akin to those found in feedforward neural networks, for making predictions or classifications [28], [29], [31]. Parameter sharing is a key feature of CNNs, significantly reducing the number of learnable parameters. In convolutional layers, the same set of filters is applied to different regions of the input, allowing for the detection of the same feature irrespective of its location.

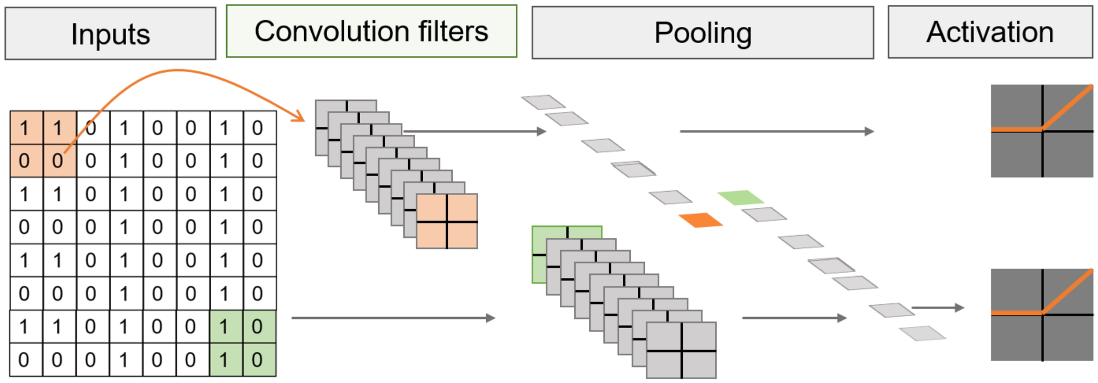


Fig. 2.7 Network architecture of a convolution neural network (CNN) implemented for image recognition.

2.4. Reservoir Computing

Reservoir Computing (RC) is a paradigm within the domain of RNNs that specializes in processing sequential data [32]–[34]. Unlike traditional RNNs, where the training process involves adjusting the weights across all time steps, RC separates the training of the reservoir (the recurrent layer) from the readout layer. This decoupling simplifies training and often leads to improved performance. In RC, the reservoir consists of a large number of recurrently connected neurons known as reservoir, forming a dynamic system that can process sequences. The key feature of the RC is its ability to generate complex, high-dimensional temporal dynamics. These dynamics result from the interplay of the connections within the reservoir and are sensitive to the input sequences presented to the network. The reservoir is typically designed to be a random and fixed structure, meaning that its connections are randomly initialized and do not change during training. This is in contrast to the readout layer, which is trained to map the high-dimensional reservoir dynamics to the desired output. This readout layer can be a simple linear regression, a neural network, or any suitable function approximation. The two major types of RC are briefly explained below sections.

2.4.1. Liquid State Machines (LSMs)

Liquid State Machines, introduced by Wolfgang Maass and Thomas Natschläger, are inspired by the behavior of neurons in the brain's cortical microcircuits [35]. LSMs use continuous streams of data (such as spike trains) to perform real-time computations. The "liquid" in an LSM refers to a randomly connected recurrent neural network, where each neuron receives input from others with random weights. This liquid process input data by allowing information to flow and interact through its recurrent connections. A readout layer, typically a memory-less readout map, generates the output based on the relevant temporal patterns and relationships within the liquid. The equations for LSMs are as follows [36],

$$x^M(t) \; = \; (L^M \, u)(t), \tag{2.8}$$

where t denotes continuous time, x^M represents the reservoir neuronal activation patterns, $u(.)$ is the input encoded as a spike sequence and L^M is the filter for transforming the input into the reservoir state.

$$y(t) = f^M\big(x^M(t)\big), \tag{2.9}$$

where $y(t)$ is the output, f^M is a memory-less readout map.

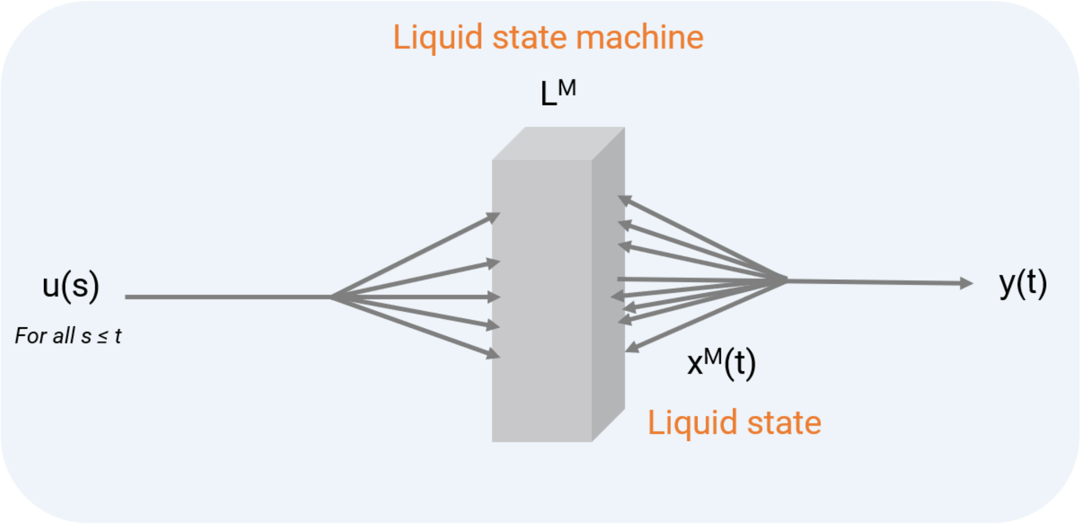


Fig. 2.8 Structure of Liquid state machine proposed by W. Maass. Adapated from [36].

2.4.2. Echo State Networks (ESNs)

Echo State Networks, proposed by Herbert Jaeger, are discrete-time neural networks constructed from a randomly initialized fixed-sized reservoir and a trainable readout layer [37]. The reservoir captures temporal dependencies in data, while the readout layer learns to map these dynamics to the desired output. The dynamics of ESNs are described by the following equations:

$$x(n) = f\big(W_{in}\, u(n) + W_{res}\, x(n-1)\big), \tag{2.10}$$

where n denotes discrete time, $x(n)$ is the state vector of the reservoir, $u(n)$ is the input vector, W_{in} is the input weight matrix and W_{res}represents the reservoir's internal recurrent connections. The output is given as,

$$y(n) = W_{out}\, x(n), \tag{2.11}$$

where, $y(n)$ is the output vector, W_{out} is the weight matrix in the readout. The readout weights can be learned using various linear regression techniques.

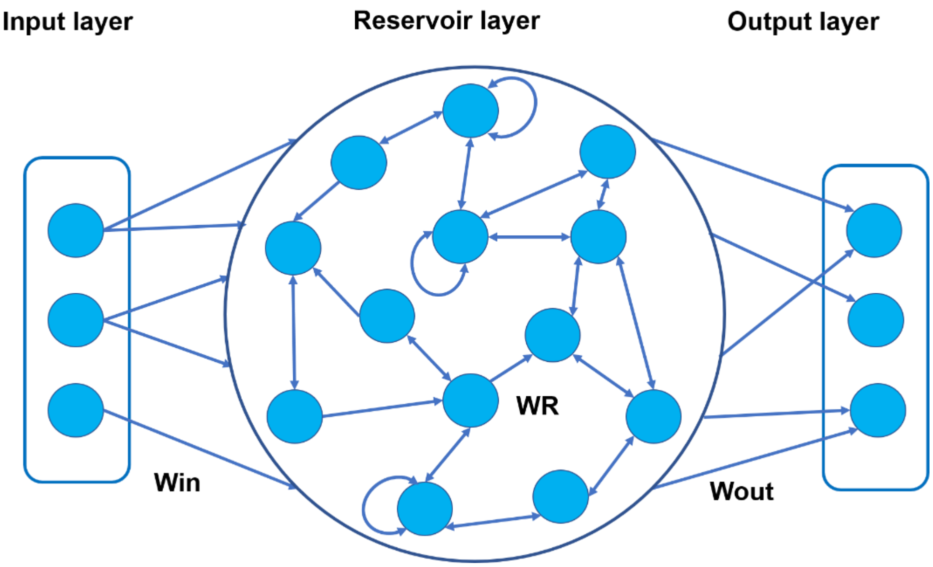


Fig. 2.9 Structure of Echo state Reservoir Computing approach.

2.4.3. Requirements of a Reservoir

To efficiently solve various computational tasks, a reservoir is expected to have specific capabilities listed below [32], [34], [38], [39].

1. ***High Dimensionality***: High-dimensional reservoirs have a large number of interconnected nodes. This high dimensionality allows the reservoir to represent complex patterns and relationships in the input data. It facilitates efficient mapping of input patterns to reservoir states, crucial for capturing intricate features.

2. ***Nonlinearity***: Reservoirs need to exhibit nonlinear dynamics, which enable them to capture and process nonlinear relationships in data. Nonlinearity results from activation functions applied to node inputs and outputs.

3. ***Separation Property***: Reservoirs should be able to separate different classes of input stimuli. This separation ensures that similar inputs are classified into the same class, simplifying the task of finding appropriate readout weights during training.

2.4.4. Benchmark Tasks in Reservoir Computing

Benchmark tasks are essential for evaluating the performance and capabilities of reservoir computing systems [32], [34], [38], [39]. Among these tasks, Memory Capacity, the Non-linear Auto Regressive Moving Average (NARMA) task, time series prediction tasks are prominent benchmarks.

2.4.4.1 Memory Capacity

Memory capacity (MC) serves as a foundational benchmark for reservoir computing. It assesses how well a reservoir can store and recall information from past inputs. The MC is quantitatively measured by evaluating the variance of the delayed input that can be accurately recovered across various delays. The MC is determined using the equation [34],

$$MC = \sum_{k=1}^{k_{max}} MC_k = \sum_{k=1}^{k_{max}} \frac{cov^2(u(t-k), y_k(t))}{var(u(t)) \cdot var(y_k(t)}. \tag{2.12}$$

Here, K_{max} is the maximum reservoir's size for MC calculations , $u(t - k)$ represents delayed inputs, and $y_k(t)$represents current output, cov and var denotes covariance and variance, respectively. This task determines the reservoir's effectiveness in retaining and retrieving past input information.

2.4.4.2. NARMA Task (Non-linear Auto Regressive Moving Average)

The NARMA task, a fundamental benchmark in reservoir computing, evaluates a reservoir's ability to model highly non-linear dynamical systems with nth-order dependencies. In this task, the system's state depends not only on incoming inputs but also on its own historical inputs, creating long-term dependencies. The challenge lies in accurately modelling these dependencies, especially in high-order systems.

2.4.4.3. Predicting Chaotic Attractors (Mackey-Glass)

Predicting chaotic attractors, such as those in the Mackey-Glass system, is another benchmark task for reservoir computing. The goal is to train the reservoir to predict the future trajectory of chaotic systems, usually one time-step ahead. Chaotic systems pose significant challenges due to their sensitivity to initial conditions.

2.4.5. Physical Implementation of Reservoir Computing

To apply reservoir networks practically, dedicated hardware implementations can substantially enhance computational speed compared to traditional optimization methods for RNNs [34], [38], [39]. One innovative approach is the use of a single dynamical node with time multiplexing delay to build a reservoir for tasks involving temporal dependencies [40]. This approach simplifies the structure by employing a single node with memory and nonlinear dynamics instead of networks of interconnected nodes. The critical element that enables this single node to function as a reservoir is the introduction of time multiplexing delay. This feedback mechanism allows the node to exhibit temporal dynamics, effectively acting as a dynamic memory element. Reservoir computing has found applications in various physical devices and properties, each offering unique advantages. Some of the notable implementations are listed in Table 1.

Category	Type of application	References
Photonic Reservoir computing	Time-series prediction, Pattern recognition	[41]–[46]
Optoelectronic Reservoir computing	Chaotic time-series prediction, High-speed data processing	[47]–[49]
Memristor-Based Reservoir Computing	Pattern recognition, Adaptive filtering	[50]–[55]
Spintronics-Based Reservoir Computing	Various information processing, including pattern recognition	[56], [57], [66], [58]–[65]

Table 1. A comprehensive list of physical systems for reservoir computing hardware implementation and the types of applications implemented.

2.5. Magnetization dynamics and spintronics oscillators

2.5.1. Micromagnetic theory

Micromagnetic theory, introduced by W.F. Brown, is a powerful framework for comprehending the behavior of magnetic materials at the microscale [67]. This theory becomes particularly crucial when we consider the magnetic properties of individual domains, grains, or even atomic moments. It provides a solid foundation for simulating and predicting various magnetic phenomena, including domain formation, hysteresis, and magnetization dynamics, across a wide spectrum of magnetic materials, such as ferromagnetic, antiferromagnetic, and ferrimagnetic materials [67][68]. At the heart of micromagnetic theory lies the concept of treating magnetization as a continuous function of space, denoted as $\boldsymbol{M}(\boldsymbol{r})$. This approach allows us to consider the average magnetization at each point over a small but sufficiently uniform volume $\Delta V,$ which contains elementary magnetic moments. It operates under the assumption that magnetic moments within volume ΔV quickly reach thermodynamic equilibrium value $M.$ Key to this approach is the concept of spatially constant saturation magnetization M_s [67]. This normalized magnetization is described by the unit vector,

$$\mathbf{m}(\mathbf{r},t) = \frac{\mathbf{M}(\mathbf{r},t)}{M_s}. \tag{2.13}$$

Micromagnetic energies play a pivotal role in understanding and modelling the behavior of magnetic materials at nanoscales. These energies arise from various physical phenomena and interactions that govern the behavior of magnetic moments within a material. They are crucial for comprehending the fundamental physics of magnetism in condensed matter systems.

2.5.1.1. Magnetostatic energies

***Exchange Energy*:**

Exchange energy is a fundamental magneto static energy that arises due to the exchange interaction between neighboring magnetic moments within a material. This energy tends to align neighboring magnetic moments parallel to each other, promoting a ferromagnetic ordering. The exchange energy density (E_{ex}) is given by the following equation:

$$E_{ex} = -\frac{1}{2}\sum_{i<j} J_{ij}\mathbf{S}_i \cdot \mathbf{S}_j, \tag{2.14}$$

where, J_{ij} is the exchange integral indicating the energy difference between the parallel and anti-parallel arrangement of the nearest neighbor spins $\mathbf{S}_i$ and $\mathbf{S}_j$. In the micromagnetic continuum description, the exchange energy is treated as a continuous vector field and the exchange energy can be written as,

$$E_{ex} = \frac{A_{ex}}{M_s^2} \int (\nabla \cdot \mathbf{M})^2 \, dV, \qquad (2.15)$$

where, A_{ex} is the exchange stiffness constant which can be calculated from $A_{ex} = \frac{k2JS^2}{a}$ in units J/m, a is the lattice constant, and k depends on the type of lattice whether simple cubic ($k = 1$), body centered cubic ($k = 3$), face centered cubic ($k = 2$) and **M** is the local magnetization vector.

***Zeeman energy*:**

The Zeeman energy arises from the interaction of magnetic moments with an external magnetic field $\mathbf{H}_{\text{ext}}$. It tends to align magnetic moments with the external field and is given by an integral over magnetized body as,

$$E_{zeeman} = = \mu_0 \int_V \mathbf{H}_{\text{ext}} \cdot \mathbf{M} \, dV, \qquad (2.16)$$

where μ_0 is the vacuum permeability, Zeeman energy influences the local magnetization changes and domain reconfigurations in response to external magnetic fields.

***Demagnetization energy*:**

Demagnetization energy accounts for the energy required to demagnetize a magnetic material. This energy arises from the material's geometry and shape. The demagnetization energy can be expressed as,

$$E_{demag} = -\frac{\mu_0}{2} \int_V \mathbf{H}_{\text{D}} \cdot \mathbf{M} \, dV. \qquad (2.17)$$

The demagnetization field $\mathbf{H}_{\text{D}}$ is caused by the divergence of the magnetic field arising from the poles located at the surface of the sample where $\nabla \cdot \mathbf{M} \neq 0$. In the Eqn.2.17, a factor 2 is introduced to avoid the double counting of interaction between two magnetic poles. The dependence of the sample geometry on the demagnetization field is accounted by a tensor $\widehat{\mathbf{N}}$, such that $\mathbf{H}_{\text{D}} = -\widehat{\mathbf{N}} \cdot \mathbf{M}$. For an ellipsoid sample with principle coordinate axes x, y, z, the demagnetization tensor $\widehat{\mathbf{N}}$ can be written in its diagonal form[69],

$$\widehat{N} = \begin{pmatrix} N_x & 0 & 0 \\ 0 & N_y & 0 \\ 0 & 0 & N_z \end{pmatrix}, \qquad (2.18)$$

where, $N_x + N_y + N_z = 1$. For a spherical sample geometry with all axis of equal length, $N_x = N_y = N_z = 1/3$, for an infinite cylinder where on axis is infinite, $N_x = 0,\ N_y = N_z = 1/2$ and a plane with two infinite axes $N_x = \ N_y = 0,\ N_z = 1$.

***Anisotropy Energy*:**

Anisotropy energy arises from the inherent preference of magnetic moments to align in certain crystallographic directions within a material. It can be described using various forms, including shape anisotropy, magneto-crystalline anisotropy, and strain-induced anisotropy. The magneto-crystalline anisotropy results from the spin-orbit interaction, and depends on the orientation of the electron spins relative to the crystallographic axes of the material. In cubic systems the energy densities due to crystal anisotropy can be written as,

$$E_{anis} = K_1\left(\alpha_x^2\alpha_y^2 + \alpha_y^2\alpha_z^2 + \alpha_z^2\alpha_x^2\right) + K_2\alpha_x^2\alpha_y^2\alpha_z^2, \tag{2.19}$$

Where, α_i are the directional cosines of the normalized magnetization $\mathbf{m}$ with respect to the Cartesian axes of the lattice. K_1 and K_2 are the crystalline anisotropy constants of first order and second order, respectively.

$$E_{anis} \approx K_1(\sin^4\theta\sin^2\phi\cos^2\phi + \sin^2\theta\sin^2\phi\cos^2\theta + \sin^2\theta\cos^2\phi\cos^2\theta). \tag{2.20}$$

For crystals with uniaxial anisotropy constant K_u, the energy density is

$$E_{anis} = K_u\alpha_x^2. \tag{2.21}$$

2.5.2. Dynamic Magnetism

In micromagnetics, understanding the dynamic behavior of magnetization within magnetic materials is crucial for various applications, from data storage devices to sensors. This dynamic magnetism is described by the Landau-Lifshitz-Gilbert (LLG) equation, which governs the time evolution of the magnetization vector $\mathbf{m}$ within a magnetic material[70][71],

$$\frac{d\mathbf{m}}{dt} = -\gamma(\mathbf{m}\times\mathbf{H}_{\mathrm{eff}}) + \alpha(\mathbf{m}\times\mathbf{m}\times\mathbf{H}_{\mathrm{eff}}), \tag{2.22}$$

where, γ is the gyromagnetic ratio, a fundamental constant with a value of approximately $1.76\times 10^{11}\ \mathrm{T}^{-1}\mathrm{s}^{-1}$. It characterizes the coupling between magnetic moments and external magnetic fields. $\mathbf{H}_{\mathrm{eff}}$ is the effective magnetic field including magnetic anisotropy, Zeeman, exchange and external fields, α is the Gilbert damping constant which characterizes the strength of dissipation.

The first term on the right-hand side corresponds to the conservative precessional torque. This term is responsible for causing the magnetic moment to precess around the direction of the effective magnetic field. It accounts for the Larmor precession of magnetic moments and is the primary driver of the motion of magnetization within the material. The second on the right-hand side represents the non-conservative damping torque. This torque aligns the magnetic moment with the direction of the effective field and describes how the magnetization vector relaxes. It reduces oscillations and ensures that the magnetization ultimately reaches its equilibrium state.

In ferromagnetic materials, the exchange interaction is a fundamental short-range interaction that plays a pivotal role in the alignment of neighboring atomic magnetic moments. This interaction tends to align adjacent magnetic moments parallel to each other, promoting ferromagnetic ordering. The energy associated with this interaction is expressed in terms of the exchange length (l_{ex}). It represents the characteristic length scale over which the exchange interaction dominates. It is typically on the order of nanometers and depends on material properties. The formula for calculating the exchange length is given by,

$$l_{ex} = \sqrt{\frac{2A_{ex}}{\mu_0 M_s^2}}. \tag{2.23}$$

The exchange length helps determine the scale over which the exchange interaction affects the magnetic behavior of the material. In materials with smaller exchange lengths, the exchange interaction dominates over longer distances, leading to distinct magnetic properties.

2.5.3. Spin Transfer Torque (STT) and Spin-Orbit Torques (SOT)

2.5.3.1. Spin Transfer Torque (STT)

STT is a fundamental phenomenon in spintronics introduced by Berger and Slonczewski. It occurs when spin-polarized electrons transfer their angular momentum to a magnetic layer, leading to changes in the magnetization direction. A common setup involves two uniform magnetic layers separated by a thin non-magnetic layer as shown in Fig. 2.9, with current flowing perpendicular to the layer stack. When an electrical current J passes through the bottom magnetic layer, electrons become spin-polarized. This means that their spins align predominantly in a particular direction. These spin-polarized electrons interact with the magnetization of the upper magnetic layer. The non-magnetic spacer layer isolates the two magnetic layers, preventing mutual exchange interaction.

The magnetization dynamics in the free layer, influenced by the spin-polarized current, can be described using an additional torque equation considering both in-plane and perpendicular torques as,

$$\boldsymbol{\tau}_{STT} = \frac{|g|}{2} \frac{\mu_B J P}{M_s |e| d} \left(\mathbf{m} \times (\mathbf{m} \times \mathbf{m}_P) - \alpha \mathbf{m} \times \mathbf{m}_p\right), \tag{2.24}$$

where, g is the g-factor, μ_B is the Bohr magneton, J is the electrical current density, P represents the current polarization ($0 < P < 1$). $\boldsymbol{m}_p$ is the magnetization of the fixed layer and determines the polarization direction of the spin current.

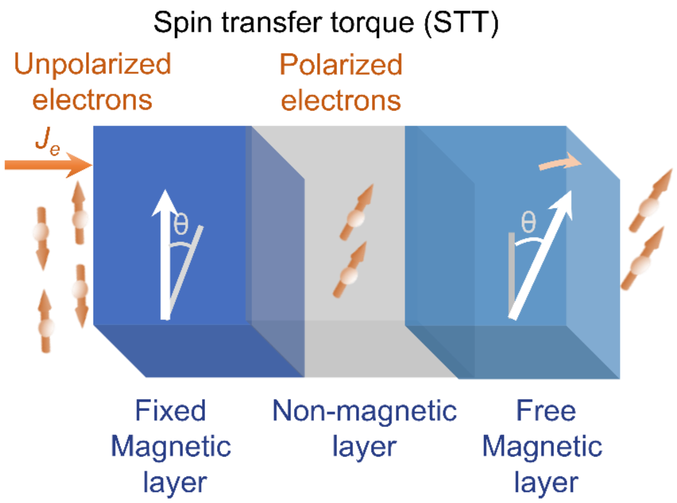


Fig. 2.9 Spin polarized charge current and spin transfer torque mechanism.

2.5.3.2. Spin-Orbit Torques (SOTs)

SOT is another important effect in spintronics, distinct from STT, and arises from the interaction between an electron's spin and its orbital motion as it moves through a material with strong spin-orbit coupling. Two major phenomena contributing to SOT are the Spin Hall Effect (SHE) and Rashba Effect.

Spin Hall Effect

The SHE is a fascinating phenomenon in condensed matter physics and spintronics, where an electrical current passing through a material can generate a transverse spin current, without the need for a magnetic field. This effect was first proposed theoretically by M.I. Dyakonov and V.I. Perel in 1971 and later fully described and named by J.E. Hirsch in 1999[72]–[74]. It has since become a significant area of research and has important implications for the development of spintronic devices. The SHE originates from the relativistic spin-orbit coupling (SOC) present in materials. Spin-orbit coupling is a relativistic effect that couples the motion of electrons' charge (orbital motion) to their intrinsic spin angular momentum. It results from the interaction of an electron's electric charge with the electric field due to the positively charged atomic nucleus. This coupling becomes prominent in heavy metals and materials with strong atomic numbers, where the electrons move at relativistic speeds. The fundamental mechanism behind the SHE is the spin-dependent scattering of charge carriers. When charge carriers, typically electrons, move through a material with strong spin-orbit coupling, they experience spin-dependent deflections due to the interaction between their spin and the crystal lattice's electric field. This deflection leads to the generation of a net spin current that flows perpendicular to the applied electric field.

The generation of a pure spin current $\boldsymbol{j}_s$ in response to an applied electric field $\boldsymbol{E}$ can be described mathematically using the following equation,

$$\boldsymbol{j}_s = \theta_{SH}\boldsymbol{E} \times \hat{\boldsymbol{z}}, \tag{2.25}$$

where, θ_{SH} is the spin Hall angle, a material-specific parameter, $\boldsymbol{E}$ is the applied electric field and $\hat{\boldsymbol{z}}$ is the unit vector perpendicular to the plane of the film. The spin Hall angle quantifies the strength of the SHE in a particular material. It depends on the material's intrinsic properties, such as its crystal structure and atomic number. The spin Hall angle quantifies the strength of the SHE in a particular material. It depends on the material's intrinsic properties, such as its crystal structure and atomic number [75]–[78].

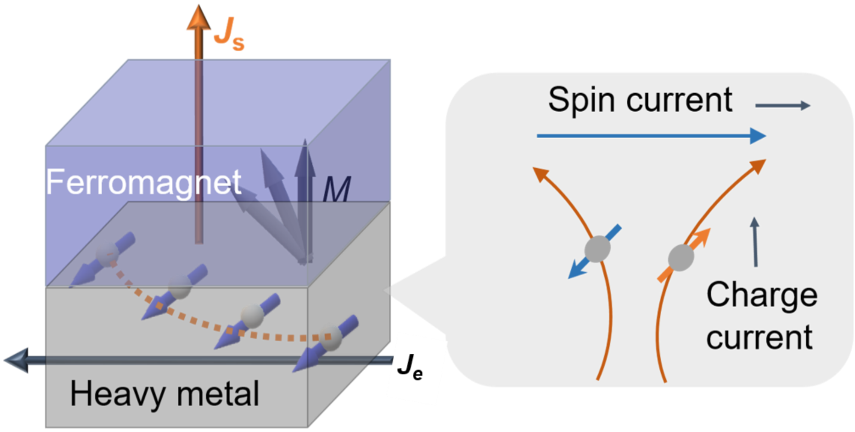


Fig. 2.9 Pure spin transport and spin Hall effect mechanism.

Rashba Effect

The Rashba Effect is another spin-orbit coupling phenomenon that can be observed at the interface between materials with strong SOC[79]–[81]. It arises when the inversion symmetry of a crystal lattice is broken at the interface. The Rashba Effect results in a momentum-dependent splitting of electronic energy bands, causing the energy levels for electrons with opposite spin orientations to separate. This energy splitting can lead to the formation of a spin texture, where the electron spins rotate in a two-dimensional plane. The Rashba Hamiltonian, which describes the energy splitting due to the Rashba Effect, can be written as follows,

$$H_{Rashba} = \alpha(\sigma_x p_y - \sigma_y p_x), \tag{2.25}$$

where, α is the Rashba coefficient, determined by the strength of the spin-orbit interaction at the interface. σ_x and σ_y are the Pauli matrices and p_x and p_y are the momentum operators.

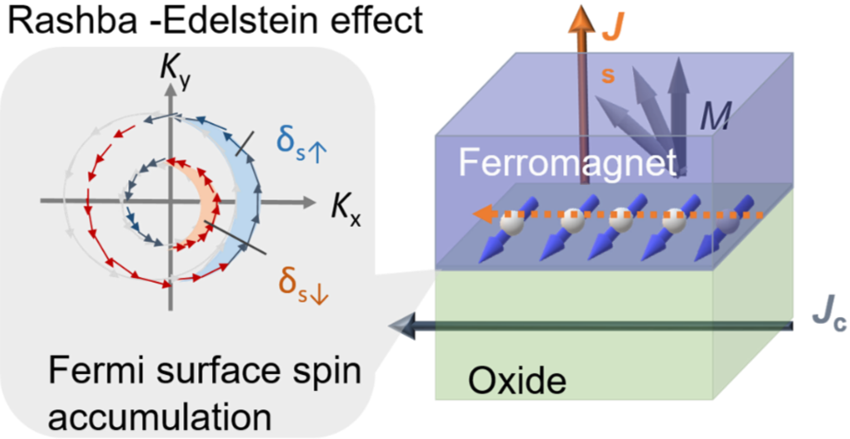


Fig. 2.10 Spin accumulation and Rashba -Edelstein effect.

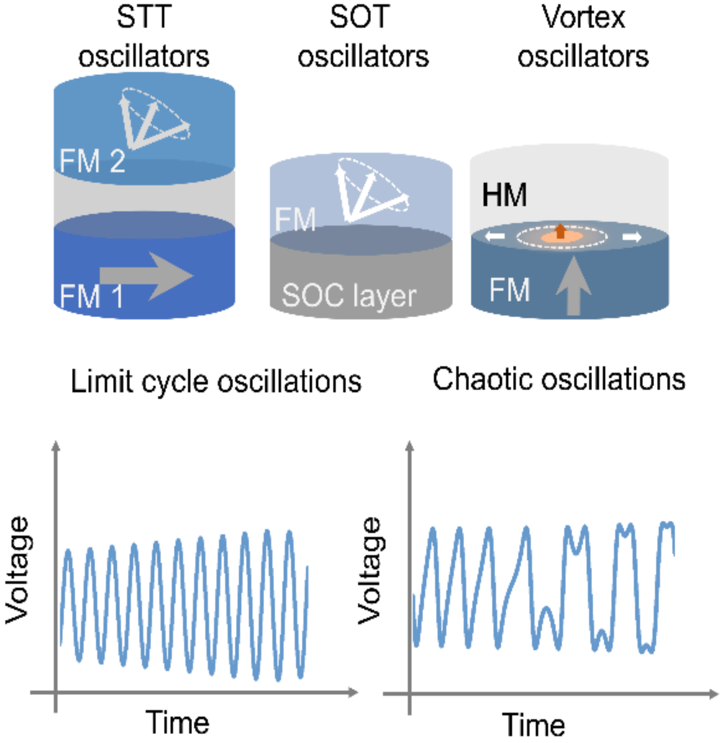


Fig. 2.11 Spintronics based oscillators based on STT and SOT principles.

2.5.4. Spin torque oscillators and spin Hall oscillators

Spintronic oscillators such as spin-torque oscillators (STOs) and spin Hall oscillators (SHOs) have been widely studied for their use as feature filters and computational tasks such as pattern recognition and classification tasks. In STOs, the spin transfer torque (STT) is induced by the electric current flowing through a multilayer structure consisting of a free magnetic layer and a fixed magnetic layer separated by a non-magnetic conductor or an insulator [82], [83]. The electric current passing through the fixed layer is spin-polarized in the direction of magnetization and exerts a torque on the free layer magnetization. The SHO consists of a ferromagnetic (FM) and heavy metal (HM) bilayer structure in which the generation of a pure spin current in the HM via the spin Hall effect (SHE) or the Rashba-Edelstein effect induces spin orbit torque (SOT) [72], [75], [84], [85]. The SHOs have a few additional benefits despite STOs having a higher oscillatory output power. First, they are easier to fabricate due to their simpler bilayer structure [86]–[89]. Second, the SOT which is caused by the pure spin current from the HM electrode, can be exerted over extended areas in SHOs, whereas the STT in STOs is a localized effect [88], [89]. SOT can be successfully used to excite and regulate a variety of magnetization oscillations [90]–[93]. Furthermore, the development of charge-to-spin conversion efficiency through material engineering and the significant contribution of a field-like term to the SOT, caused by interfacial effects, provide an interesting opportunity to investigate SHO-based neuromorphic hardware [76], [78], [94], [95].

References

[1] S. E. Petersen and O. Sporns, “Brain Networks and Cognitive Architectures,” *Neuron*, vol. 88, no. 1, pp. 207–219, 2015, doi: 10.1016/j.neuron.2015.09.027.

[2] K. Gurney, “*An introduction to neural networks*.,” CRC Press, 2018.

[3] C. M. Bishop and others, “*Neural networks for pattern recognition*.,” Oxford university press, 1995.

[4] G. M. Shepherd, “*The Synaptic Organization of the Brain*.,” Oxford University Press, 2004. doi: 10.1093/acprof:oso/9780195159561.001.1.

[5] C. C. H. Petersen, “The functional organization of the barrel cortex.,” *Neuron*, vol. 56, no. 2, pp. 339–355, 2007, doi: 10.1016/j.neuron.2007.09.017.

[6] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bull. Math. Biophys.*, vol. 5, no. 4, pp. 115–133, Dec. 1943, doi: 10.1007/BF02478259.

[7] K.-L. Du and M. N. S. Swamy, “*Neural Networks and Statistical Learning.*,” no. 2007. London: Springer London, 2014. doi: 10.1007/978-1-4471-5571-3_13.

[8] B.D.Ripley, “*Pattern Recogonition and Neural Networks.*,” vol. 01. Cambridge University Press, 1996.

[9] A. M. Andrew, “The Handbook of Brain Theory and Neural Networks,” *Kybernetes*, vol. 28, no. 9, pp. 1084–1094, 1999, doi: 10.1108/k.1999.28.9.1084.1.

[10] G. Kumar, R. Banerjee, D. Kr Singh, N. Choubey, and Arnaw, “Mathematics for Machine Learning,” *J. Math. Sci. Comput. Math.*, vol. 1, no. 2, pp. 229–238, 2020, doi: 10.15864/jmscm.1208.

[11] S. W. Ellacott, J. C. Mason, and I. J. Anderson, Eds., “*Mathematics of Neural Networks*.,” vol. 8. in Operations Research/Computer Science Interfaces Series, vol. 8. Boston, MA: Springer US, 1997. doi: 10.1007/978-1-4615-6099-9.

[12] K.-L. Du and M. N. S. Swamy, “*Neural Networks and Statistical Learning*.,” Springer London, 2014. doi: 10.1007/978-1-4471-5571-3.

[13] R. Rojas, “*Neural Networks.*,” vol. 53, no. 9. Berlin, Heidelberg: Springer Berlin Heidelberg, 1996.

[14] Bharadwaj, K. B. Prakash, and G. R. Kanagachidambaresan, “*Pattern Recognition and Machine Learning*.,” 2021. doi: 10.1007/978-3-030-57077-4_11.

[15] B. Scholkopf *et al.*, “Toward Causal Representation Learning.,” *Proc. IEEE*, vol. 109, no. 5, pp. 612–634, 2021, doi: 10.1109/JPROC.2021.3058954.

[16] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015. [Online]. Available: https://www.tensorflow.org/

[17] D. J. Rao, “Deep Learning Using Keras,” *Keras to Kubernetes*®, pp. 111–129, 2019, doi: 10.1002/9781119564843.ch4.

[18] A. Paszke *et al.*, “PyTorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems.,” *NeurIPS*, no. NeurIPS, pp. 8026–8037, 2019.

[19] R. Raina, A. Madhavan, and A. Y. Ng, “Large-scale deep unsupervised learning using graphics processors.,” *ACM Int. Conf. Proceeding Ser.*, vol. 382, 2009, doi: 10.1145/1553374.1553486.

[20] N. P. Jouppi *et al.*, “In-datacenter performance analysis of a tensor processing unit,” *Proc. - Int. Symp. Comput. Archit.*, vol. Part F1286, pp. 1–12, 2017, doi: 10.1145/3079856.3080246.

[21] L. Gonon, L. Grigoryeva, and J. P. Ortega, “Memory and forecasting capacities of nonlinear recurrent networks,” *Phys. D Nonlinear Phenom.*, vol. 414, no. April, p. 132721, 2020, doi: 10.1016/j.physd.2020.132721.

[22] K. Mehrotra, K. M. Chilukuri, and S. Ranka, “Elements of Artificial Neural Networks (Complex Adaptive Systems),” p. 400, 1996.

[23] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, 1997, doi: 10.1162/neco.1997.9.8.1735.

[24] C. C. Aggarwal, “Neural Networks and Deep Learning.,” Cham: Springer International Publishing, 2018. doi: 10.1007/978-3-319-94463-0.

[25] J. Schmidhuber, “Deep Learning in neural networks: An overview,” *Neural Networks*, vol. 61, pp. 85–117, 2015, doi: 10.1016/j.neunet.2014.09.003.

[26] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015, doi: 10.1038/nature14539.

[27] I. H. Sarker, “Machine Learning: Algorithms, Real-World Applications and Research Directions.,” *SN Comput. Sci.*, vol. 2, no. 3, p. 160, May 2021, doi: 10.1007/s42979-021-00592-x.

[28] S. Ahlawat, A. Choudhary, A. Nayyar, S. Singh, and B. Yoon, “Improved handwritten digit

recognition using convolutional neural networks (CNN).," *Sensors (Switzerland)*, vol. 20, no. 12, pp. 1–18, 2020, doi: 10.3390/s20123344.

[29] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions.," vol. 8, no. 1. Springer International Publishing, 2021. doi: 10.1186/s40537-021-00444-8.

[30] H. Li, "Deep learning for natural language processing: advantages and challenges.," *Natl. Sci. Rev.*, vol. 5, no. 1, pp. 24–26, Jan. 2018, doi: 10.1093/nsr/nwx110.

[31] Y. Zhang *et al.*, "A system hierarchy for brain-inspired computing," *Nature*, vol. 586, no. 7829, pp. 378–384, Oct. 2020, doi: 10.1038/s41586-020-2782-y.

[32] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Comput. Sci. Rev.*, vol. 3, no. 3, pp. 127–149, Aug. 2009, doi: 10.1016/j.cosrev.2009.03.005.

[33] J. Dambre, D. Verstraeten, B. Schrauwen, and S. Massar, "Information Processing Capacity of Dynamical Systems," *Sci. Rep.*, vol. 2, no. 1, p. 514, Jul. 2012, doi: 10.1038/srep00514.

[34] Farkaš, R. Bosák, and P. Gergeľ, "Computational analysis of memory capacity in echo state networks," *Neural Networks*, vol. 83, pp. 109–120, Nov. 2016, doi: 10.1016/j.neunet.2016.07.012.

[35] W. W. Maass, H. Markram, T. Natschläger, W. W. Maass, and H. Markram, "The" liquid computer": A novel strategy for real-time computing on time series," *Spec. Issue Found. Inf. Process. Telemat.*, vol. 8, no. 1, pp. 39–43, 2002, [Online]. Available: http://infoscience.epfl.ch/record/117806

[36] W. Maass, "Liquid state machines: Motivation, theory, and applications," *Comput. Context Comput. Log. Real World*, pp. 275–296, 2011, doi: 10.1142/9781848162778_0008.

[37] H. Jaeger, "The 'echo state' approach to analysing and training recurrent neural networks," 2001. [Online]. Available: http://www.faculty.jacobs-university.de/hjaeger/pubs/EchoStatesTechRep.pdf

[38] K. Nakajima, "Physical reservoir computing—an introductory perspective," *Jpn. J. Appl. Phys.*, vol. 59, no. 6, p. 060501, Jun. 2020, doi: 10.35848/1347-4065/ab8d4f.

[39] K. Nakajima and I. Fischer, *Reservoir Computing: Theory, Physical Implementations, and Applications*. in Natural Computing Series. Singapore: Springer Singapore, 2021. doi: 10.1007/978-981-13-1687-6.

[40] L. Appeltant *et al.*, "Information processing using a single dynamical node as complex system,"

Nat. Commun., vol. 2, no. 1, p. 468, Sep. 2011, doi: 10.1038/ncomms1476.

[41] F. Duport, A. Smerieri, A. Akrout, M. Haelterman, and S. Massar, “Fully analogue photonic reservoir computer,” *Sci. Rep.*, vol. 6, no. March, pp. 1–12, 2016, doi: 10.1038/srep22381.

[42] A. Katumba, M. Freiberger, P. Bienstman, and J. Dambre, “A Multiple-Input Strategy to Efficient Integrated Photonic Reservoir Computing,” *Cognit. Comput.*, vol. 9, no. 3, pp. 307–314, 2017, doi: 10.1007/s12559-017-9465-5.

[43] K. Harkhoe, G. Verschaffelt, A. Katumba, P. Bienstman, and G. van der Sande, “Demonstrating delay-based reservoir computing using a compact photonic integrated chip,” *arXiv*, no. October, 2019, doi: 10.1364/oe.382556.

[44] P. Antonik, M. Hermans, F. Duport, M. Haelterman, and S. Massar, “Towards pattern generation and chaotic series prediction with photonic reservoir computers,” *Real-time Meas. Rogue Events, Emerg. Appl.*, vol. 9732, p. 97320B, 2016, doi: 10.1117/12.2210948.

[45] J. Dong, M. Rafayelyan, F. Krzakala, and S. Gigan, “Optical Reservoir Computing Using Multiple Light Scattering for Chaotic Systems Prediction,” *arXiv*, pp. 1–15, 2019, doi: 10.1109/jstqe.2019.2936281.

[46] G. Van Der Sande, D. Brunner, and M. C. Soriano, “Advances in photonic reservoir computing,” *Nanophotonics*, vol. 6, no. 3, pp. 561–576, 2017, doi: 10.1515/nanoph-2016-0132.

[47] L. Sun *et al.*, “In-sensor reservoir computing for language learning via two-dimensional memristors,” *Sci. Adv.*, vol. 7, no. 20, 2021, doi: 10.1126/sciadv.abg1455.

[48] Y. Paquot *et al.*, “Optoelectronic reservoir computing,” *Sci. Rep.*, vol. 2, pp. 1–6, 2012, doi: 10.1038/srep00287.

[49] P. Kumar, M. Jin, T. Bu, S. Kumar, and Y.-P. Huang, “Efficient reservoir computing using field programmable gate array and electro-optic modulation,” *OSA Contin.*, vol. 4, no. 3, p. 1086, 2021, doi: 10.1364/osac.417996.

[50] R. Midya *et al.*, “Reservoir Computing Using Diffusive Memristors,” *Adv. Intell. Syst.*, vol. 1, no. 7, p. 1900084, Nov. 2019, doi: 10.1002/aisy.201900084.

[51] J. Moon *et al.*, “Temporal data classification and forecasting using a memristor-based reservoir computing system,” *Nat. Electron.*, vol. 2, no. 10, pp. 480–487, 2019, doi: 10.1038/s41928-019-0313-3.

[52] M. Kulkarni, “Memristor-based Reservoir Computing.,” (2012). Dissertations and Theses. Paper

899. doi: 10.15760/etd.899

[53] Y. Zhong, J. Tang, X. Li, B. Gao, H. Qian, and H. Wu, “Dynamic memristor-based reservoir computing for high-efficiency temporal signal processing,” *Nat. Commun.*, no. 2021, pp. 1–9, doi: 10.1038/s41467-020-20692-1.

[54] C. Du, F. Cai, M. A. Zidan, W. Ma, S. H. Lee, and W. D. Lu, “Reservoir computing using dynamic memristors for temporal information processing,” *Nat. Commun.*, vol. 8, no. 1, pp. 1–10, 2017, doi: 10.1038/s41467-017-02337-y.

[55] X. Zhang *et al.*, “Hybrid memristor-CMOS neurons for in-situ learning in fully hardware memristive spiking neural networks,” *Sci. Bull.*, vol. 66, no. 16, pp. 1624–1633, Aug. 2021, doi: 10.1016/j.scib.2021.04.014.

[56] W. Jiang *et al.*, “Physical reservoir computing using magnetic skyrmion memristor and spin torque nano-oscillator,” *Appl. Phys. Lett.*, vol. 115, no. 19, p. 192403, Nov. 2019, doi: 10.1063/1.5115183.

[57] T. Yamaguchi, N. Akashi, S. Tsunegi, H. Kubota, K. Nakajima, and T. Taniguchi, “Periodic structure of memory function in spintronics reservoir with feedback current,” *Phys. Rev. Res.*, vol. 2, no. 2, p. 023389, Jun. 2020, doi: 10.1103/PhysRevResearch.2.023389.

[58] A. Kurenkov, S. Fukami, and H. Ohno, “Neuromorphic computing with antiferromagnetic spintronics,” *J. Appl. Phys.*, vol. 128, no. 1, 2020, doi: 10.1063/5.0009482.

[59] M.-K. Lee and M. Mochizuki, “Reservoir Computing with Spin Waves in Skyrmion Crystal,” pp. 1–16, 2022, [Online]. Available: http://arxiv.org/abs/2203.02160

[60] W. Jiang *et al.*, “Physical reservoir computing built by spintronic devices for temporal information processing,” *arXiv*, 2019.

[61] S. Tsunegi *et al.*, “Physical reservoir computing based on spin torque oscillator with forced synchronization,” *Appl. Phys. Lett.*, vol. 114, no. 16, 2019, doi: 10.1063/1.5081797.

[62] D. Marković *et al.*, “Reservoir computing with the frequency, phase, and amplitude of spin-torque nano-oscillators,” *Appl. Phys. Lett.*, vol. 114, no. 1, 2019, doi: 10.1063/1.5079305.

[63] H. Nomura, K. Tsujimoto, M. Goto, N. Samura, R. Nakatani, and Y. Suzuki, “Reservoir computing with two-bit input task using dipole-coupled nanomagnet array,” *Jpn. J. Appl. Phys.*, vol. 59, no. SE, 2020, doi: 10.7567/1347-4065/ab4f0f.

[64] W. Jiang *et al.*, “Physical reservoir computing using magnetic skyrmion memristor and spin torque

nano-oscillator," *Appl. Phys. Lett.*, vol. 115, no. 19, 2019, doi: 10.1063/1.5115183.

[65] D. Pinna, G. Bourianoff, and K. Everschor-Sitte, "Reservoir Computing with Random Skyrmion Textures," *Phys. Rev. Appl.*, vol. 14, no. 5, p. 054020, Nov. 2020, doi: 10.1103/PhysRevApplied.14.054020.

[66] T. Taniguchi, A. Ogihara, Y. Utsumi, and S. Tsunegi, "Spintronic reservoir computing without driving current or magnetic field," *Sci. Rep.*, vol. 12, no. 1, pp. 1–11, 2022, doi: 10.1038/s41598-022-14738-1.

[67] W. F. Brown, "Thermal Fluctuations of a Single-Domain Particle," *Phys. Rev.*, vol. 130, no. 5, pp. 1677–1686, Jun. 1963, doi: 10.1103/PhysRev.130.1677.

[68] R. Skomski, *Simple Models of Magnetism*. Oxford University Press, 2008

[69] A. Aharoni, "Demagnetizing factors for rectangular ferromagnetic prisms," *J. Appl. Phys.*, vol. 83, no. 6, pp. 3432–3434, Mar. 1998, doi: 10.1063/1.367113.

[70] L. LANDAU and E. LIFSHITZ, "On the theory of the dispersion of magnetic permeability in ferromagnetic bodies," in *Perspectives in Theoretical Physics*, L. P. PITAEVSKI, Ed., Amsterdam: Elsevier, 1992, pp. 51–65. doi: 10.1016/B978-0-08-036364-6.50008-9.

[71] T. L. Gilbert, "A phenomenological theory of damping in ferromagnetic materials," *IEEE Trans. Magn.*, vol. 40, no. 6, pp. 3443–3449, 2004, doi: 10.1109/TMAG.2004.836740.

[72] J. E. Hirsch, "Spin Hall Effect," *Phys. Rev. Lett.*, vol. 83, no. 9, pp. 1834–1837, Aug. 1999, doi: 10.1103/physrevlett.83.1834.

[73] M. I. Dyakonov, "Spin Hall effect," *Int. J. Mod. Phys. B*, vol. 23, no. 12–13, pp. 2556–2565, 2009, doi: 10.1142/s0217979209061986.

[74] M. I. Dyakonov and V. I. Perel, "Current-induced spin orientation of electrons in semiconductors," *Phys. Lett. A*, vol. 35, no. 6, pp. 459–460, 1971, doi: 10.1016/0375-9601(71)90196-4.

[75] J. Sinova, S. O. Valenzuela, J. Wunderlich, C. H. Back, and T. Jungwirth, "Spin Hall effects," *Rev. Mod. Phys.*, vol. 87, no. 4, pp. 1213–1260, Oct. 2015, doi: 10.1103/RevModPhys.87.1213.

[76] A. Hoffmann, "Spin Hall Effects in Metals," *IEEE Trans. Magn.*, vol. 49, no. 10, pp. 5172–5193, Oct. 2013, doi: 10.1109/TMAG.2013.2262947.

[77] U. Shashank *et al.*, "Enhanced Spin Hall Effect in S-Implanted Pt," *Adv. Quantum Technol.*, vol. 4, no. 1, pp. 1–6, 2021, doi: 10.1002/qute.202000112.

[78] U. Shashank *et al.*, "Highly dose dependent damping-like spin–orbit torque efficiency in O-

implanted Pt," *Appl. Phys. Lett.*, vol. 118, no. 25, p. 252406, Jun. 2021, doi: 10.1063/5.0054779.

[79] Y. A. Bychkov and E. I. Rashba, "Properties of a 2D electron gas with lifted spectral degeneracy," *JETP Letters*, vol. 39. pp. 78–81, 1984.

[80] A. Manchon, H. C. Koo, J. Nitta, S. M. Frolov, and R. A. Duine, "New perspectives for Rashba spin-orbit coupling," *Nat. Mater.*, vol. 14, no. 9, pp. 871–882, 2015, doi: 10.1038/nmat4360.

[81] D. Bercioux and P. Lucignano, "Quantum transport in Rashba spin–orbit materials: a review," *Reports Prog. Phys.*, vol. 78, no. 10, p. 106001, Oct. 2015, doi: 10.1088/0034-4885/78/10/106001.

[82] J. C. Slonczewski, "Current-driven excitation of magnetic multilayers," *J. Magn. Magn. Mater.*, vol. 159, no. 1–2, pp. L1--L7, Jun. 1996, doi: 10.1016/0304-8853(96)00062-5.

[83] S. I. Kiselev *et al.*, "Microwave oscillations of a nanomagnet driven by a spin-polarized current," *Nature*, vol. 425, no. 6956, pp. 380–383, Sep. 2003, doi: 10.1038/nature01967.

[84] I. M. Miron *et al.*, "Current-driven spin torque induced by the Rashba effect in a ferromagnetic metal layer," *Nat. Mater.*, vol. 9, no. 3, pp. 230–234, Jan. 2010, doi: 10.1038/nmat2613.

[85] A. R. Mellnik *et al.*, "Spin-transfer torque generated by a topological insulator," *Nature*, vol. 511, no. 7510, pp. 449–451, Jul. 2014, doi: 10.1038/nature13534.

[86] V. E. Demidov *et al.*, "Magnetization oscillations and waves driven by pure spin currents," *Phys. Rep.*, vol. 673, pp. 1–31, Feb. 2017, doi: 10.1016/j.physrep.2017.01.001.

[87] T. Chen *et al.*, "Spin-Torque and Spin-Hall Nano-Oscillators," *Proc. IEEE*, vol. 104, no. 10, pp. 1919–1945, Oct. 2016, doi: 10.1109/JPROC.2016.2554518.

[88] Z. Duan *et al.*, "Nanowire spin torque oscillator driven by spin orbit torques," *Nat. Commun.*, vol. 5, no. 1, Dec. 2014, doi: 10.1038/ncomms6616.

[89] Z. Duan *et al.*, "Spin-wave modes in permalloy/platinum wires and tuning of the mode damping by spin Hall current," *Phys. Rev. B*, vol. 90, no. 2, Jul. 2014, doi: 10.1103/physrevb.90.024427.

[90] V. E. Demidov, S. Urazhdin, E. R. J. Edwards, M. D. Stiles, R. D. McMichael, and S. O. Demokritov, "Control of Magnetic Fluctuations by Spin Current," *Phys. Rev. Lett.*, vol. 107, no. 10, Sep. 2011, doi: 10.1103/physrevlett.107.107204.

[91] V. E. Demidov *et al.*, "Magnetic nano-oscillator driven by pure spin current," *Nat. Mater.*, vol. 11, no. 12, pp. 1028–1031, Dec. 2012, doi: 10.1038/nmat3459.

[92] H. Fulara *et al.*, "Spin-orbit torque–driven propagating spin waves," *Sci. Adv.*, vol. 5, no. 9, Sep. 2019, doi: 10.1126/sciadv.aax8467.

[93] M. Dvornik, A. A. Awad, and J. Åkerman, “Origin of Magnetization Auto-Oscillations in Constriction-Based Spin Hall Nano-Oscillators,” *Phys. Rev. Appl.*, vol. 9, no. 1, Jan. 2018, doi: 10.1103/physrevapplied.9.014017.

[94] U. Shashank *et al.*, “Enhanced Spin Hall Effect in S-Implanted Pt,” *Adv. Quantum Technol.*, vol. 4, no. 1, p. 2000112, Dec. 2020, doi: 10.1002/qute.202000112.

[95] K.-W. Kim, K.-J. Lee, J. Sinova, H.-W. Lee, and M. D. Stiles, “Spin-orbit torques from interfacial spin-orbit coupling for various interfaces,” *Phys. Rev. B*, vol. 96, no. 10, Sep. 2017, doi: 10.1103/physrevb.96.104438.

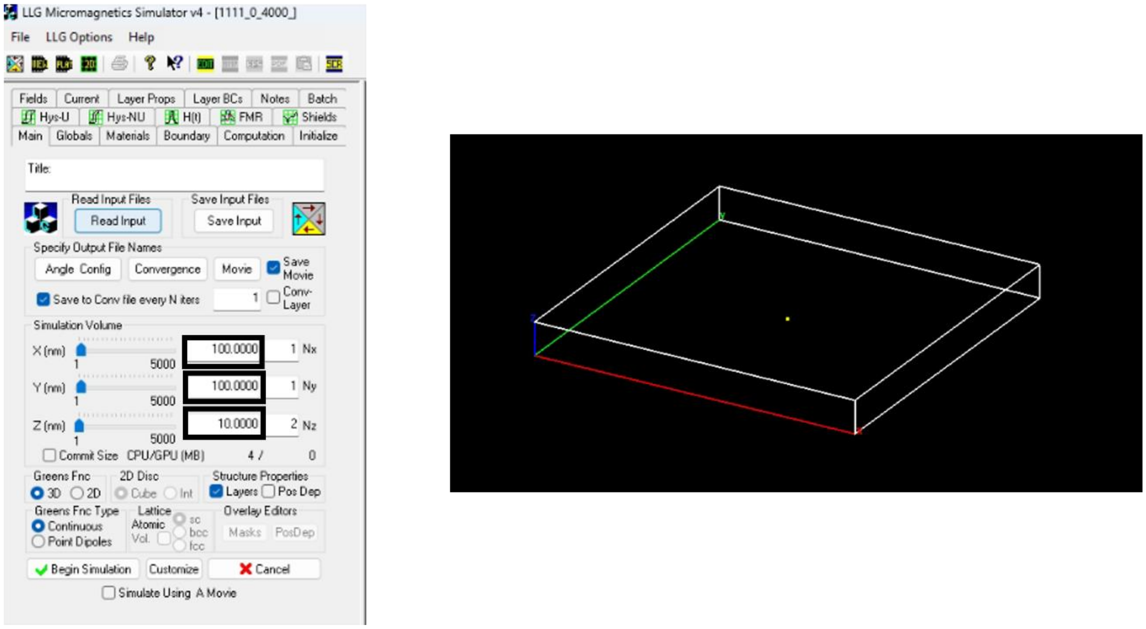
Chapter 3

Materials and Methods

3.1. Micromagnetic simulations

3.1.1. LLG simulator – Simulation & Data processing

In this section, we briefly explain how we perform micromagnetic simulations using the LLG Micromagnetics Simulator [1]. Fig.3.1 shows the displayed user interface when starting a new simulation. We set the dimensions of the simulation model by changing the values in *Simulation Volume* dialog box along X, Y and Z directions. *Nx, Ny, Nz* values refer to number of computation cells along each direction. We activate multiple layers by checking the *Layers* box.



☑ Save Movie
☑ Save to Conv file every N iters 1
□ Conv-Layer

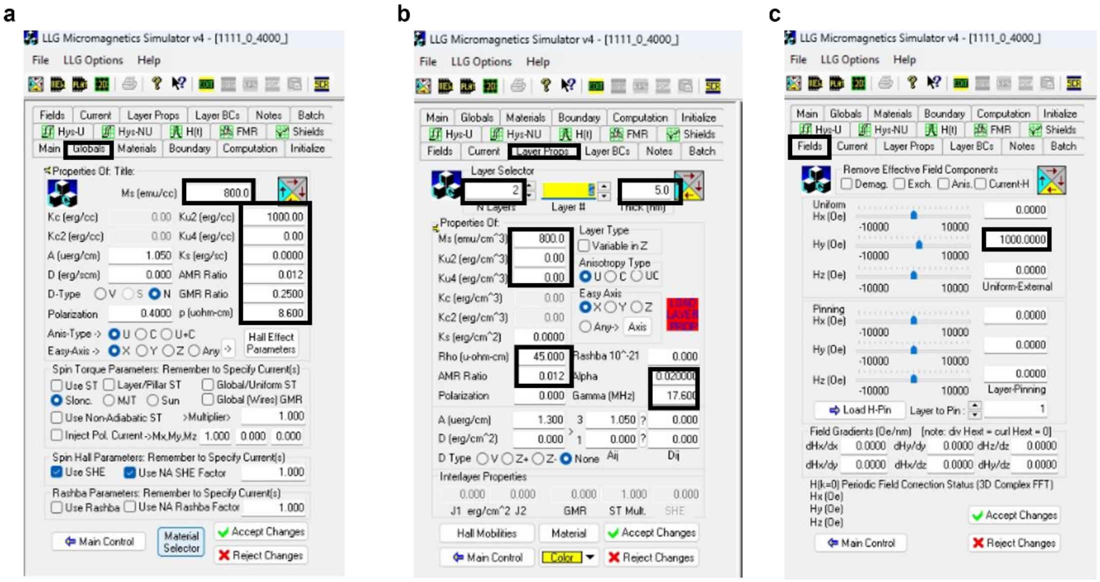
X (nm) 1 5000 100.0000 1 Nx
Y (nm) 1 5000 100.0000 1 Ny
Z (nm) 1 5000 10.0000 2 Nz
□ Commit Size CPU/GPU (MB) 4 / 0

Greens Fnc ● 3D ○ 2D
2D Disc ○ Cube ○ Int
Structure Properties ☑ Layers □ Pos Dep
Greens Fnc Type ● Continuous ○ Point Dipoles
Lattice Atomic Vol. □ ○ sc ○ bcc ○ fcc
□ Simulate Using A Movie

Fig. 3.1. Specifying the simulation volume and multiple layers.

As shown in Fig. 3.2 (a), in the *Globals* page, we define the material properties for each cell in the simulation volume. Here we can define the values of saturation magnetization (M_s), 2nd and 4th order uniaxial anisotropies (Ku_2, Ku_4), cubic anisotropy (K_c), exchange stiffness(A), surface anisotropy (K_s), resistivity(ρ), anisotropic magnetoresistance (AMR), as well as the types and directions of the easy axis. To load the predefined material specific parameters (e.g., Fe, Co, Permalloy), we can click the *Material*

Selector. All parameters are set in CGS units. Alternatively, when modelling multiple layers, we can change the thickness and material properties of each layer separately, in the *Layer Props* page (Fig. 3.2 (b)). In the *Fields* page (Fig. 3.2 (c)), we set the strength and orientation of the external magnetic field (H_{ext}).


a
LLG Micromagnetics Simulator v4 - [1111_0_4000_]
File LLG Options Help
Fields Current Layer Props Layer BCs Notes Batch
Hys-U Hys-NU H(t) FMR Shields
Main Globals Materials Boundary Computation Initialize
Hall Effect Parameters
Main Control
Material Selector
Accept Changes
Reject Changes
b
LLG Micromagnetics Simulator v4 - [1111_0_4000_]
File LLG Options Help
Main Globals Materials Boundary Computation Initialize
Hys-U Hys-NU H(t) FMR Shields
Fields Current Layer Props Layer BCs Notes Batch
Axis
Hall Mobilities
Material
Accept Changes
Main Control
Color
Reject Changes
c
LLG Micromagnetics Simulator v4 - [1111_0_4000_]
File LLG Options Help
Main Globals Materials Boundary Computation Initialize
Hys-U Hys-NU H(t) FMR Shields
Fields Current Layer Props Layer BCs Notes Batch
Load H-Pin
Accept Changes
Main Control
Reject Changes

a

Properties Of: Title:

Ms (emu/cc) 800.0

Kc (erg/cc)	0.00	Ku2 (erg/cc)	1000.00
Kc2 (erg/cc)	0.00	Ku4 (erg/cc)	0.00
A (uerg/cm)	1.050	Ks (erg/sc)	0.0000
D (erg/scm)	0.000	AMR Ratio	0.012
D-Type ○ V ○ S ● N		GMR Ratio	0.2500
Polarization	0.4000	p (uohm-cm)	8.600

Anis-Type -> ● U ○ C ○ U+C

Easy-Axis -> ● X ○ Y ○ Z ○ Any ->

Spin Torque Parameters: Remember to Specify Current(s)

□ Use ST □ Layer/Pillar ST □ Global/Uniform ST

● Slonc. ○ MJT ○ Sun □ Global (Wires) GMR

□ Use Non-Adiabatic ST >Multiplier> 1.000

□ Inject Pol. Current ->Mx,My,Mz 1.000 0.000 0.000

Spin Hall Parameters: Remember to Specify Current(s)

☑ Use SHE ☑ Use NA SHE Factor 1.000

Rashba Parameters: Remember to Specify Current(s)

□ Use Rashba □ Use NA Rashba Factor 1.000

b

Layer Selector

N Layers 2 Layer # Thick (nm) 5.0

Properties Of:

Ms (emu/cm^3)	800.0	Layer Type □ Variable in Z	
Ku2 (erg/cm^3)	0.00	Anisotropy Type ● U ○ C ○ UC	
Ku4 (erg/cm^3)	0.00		
Kc (erg/cm^3)	0.00	Easy Axis ● X ○ Y ○ Z	
Kc2 (erg/cm^3)	0.00	○ Any->	
Ks (erg/cm^2)	0.0000		
Rho (u-ohm-cm)	45.000	Rashba 10^-21	0.000
AMR Ratio	0.012	Alpha	0.020000
Polarization	0.000	Gamma (MHz)	17.600

A (uerg/cm)	1.300	3	1.050 ?	0.000
D (erg/cm^2)	0.000	1	0.000 ?	0.000
			Aij	Dij

D Type ○ V ○ Z+ ○ Z- ● None

Interlayer Properties

J1	erg/cm^2 J2	GMR	ST Mult.	SHE
0.000	0.000	0.000	1.000	0.000

c

Remove Effective Field Components

□ Demag. □ Exch. □ Anis. □ Current-H

Uniform

Hx (Oe)	-10000	10000	0.0000
Hy (Oe)	-10000	10000	1000.0000
Hz (Oe)	-10000	10000	0.0000

Uniform-External

Pinning

Hx (Oe)	-10000	10000	0.0000
Hy (Oe)	-10000	10000	0.0000
Hz (Oe)	-10000	10000	0.0000

Layer-Pinning

Layer to Pin : 1

Field Gradients (Oe/nm) [note: div Hext = curl Hext = 0]

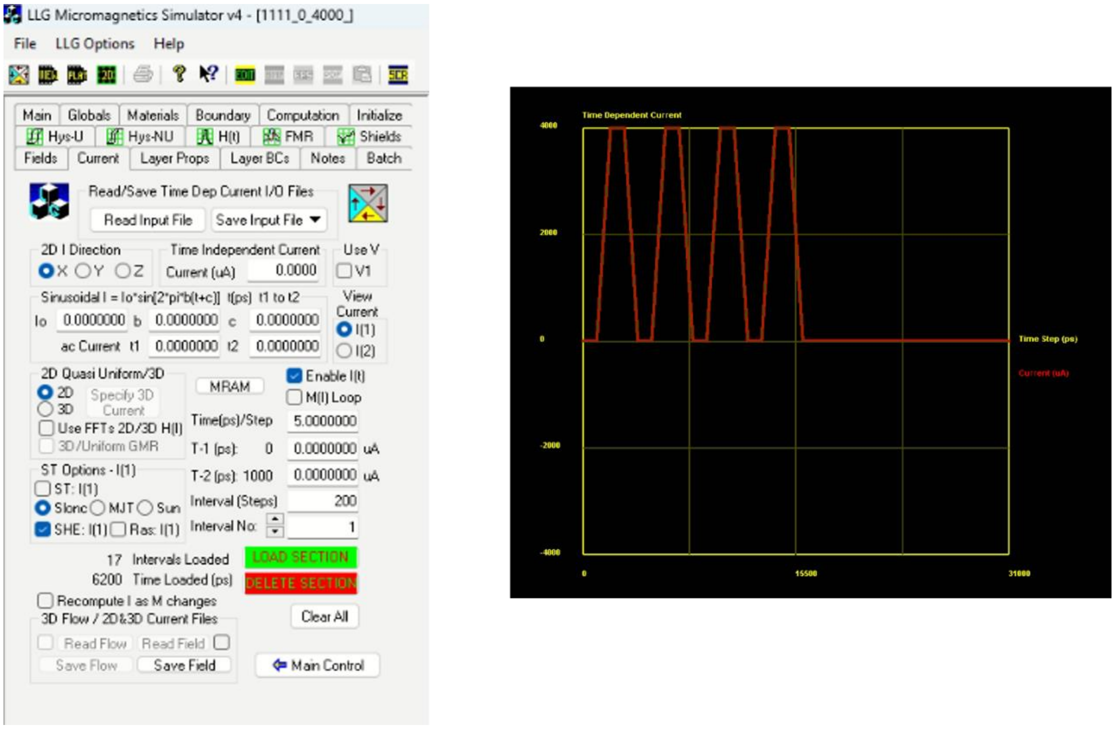
dHx/dx	0.0000	dHy/dy	0.0000	dHz/dz	0.0000
dHx/dy	0.0000	dHx/dz	0.0000	dHy/dz	0.0000

H(k=0) Periodic Field Correction Status (3D Complex FFT)

Hx (Oe)
Hy (Oe)
Hz (Oe)

Fig. 3.2 The material parameters can be set **a,** in the *Globals* page, or **b,** layer wise in the *Layer Props* page. **C,** the external magnetic field is set in the *Fields* page.

In the *Current* page (Fig. 3.3), we specify the input currents. We choose from the *2D I Direction* to specify the direction of current flow. Checking *Enable I(t)* allows us to inject a time-dependent current. The current pulse can be seen on the adjoining viewing panel. Depending on the spin torque term of interest, we choose appropriate options from the *ST Options – I(1)* box. Since we are utilizing the torque arising from the spin Hall effect, we choose the *SHE* box. We can also load predefined current sequences or save the current sequences we have created, by using the *Read Input File* or *Save Input File* options.



LLG Micromagnetics Simulator v4 - [1111_0_4000_]

Read/Save Time Dep Current I/O Files

2D I Direction: ● X ○ Y ○ Z

Time Independent Current: Current (uA) 0.0000

Use V: □ V1

Sinusoidal I = Io*sin[2*pi*b(t+c)] t(ps) t1 to t2

Io 0.0000000 b 0.0000000 c 0.0000000

ac Current t1 0.0000000 t2 0.0000000

View Current: ● I(1) ○ I(2)

2D Quasi Uniform/3D: ● 2D ○ 3D

□ Use FFTs 2D/3D H(I)

□ 3D/Uniform GMR

☑ Enable I(t)

□ M(I) Loop

Time(ps)/Step 5.0000000

T-1 (ps): 0 0.0000000 uA

T-2 (ps): 1000 0.0000000 uA

Interval (Steps) 200

Interval No: 1

ST Options - I(1)

□ ST: I(1)

● Slonc ○ MJT ○ Sun

☑ SHE: I(1) □ Ras: I(1)

17 Intervals Loaded

6200 Time Loaded (ps)

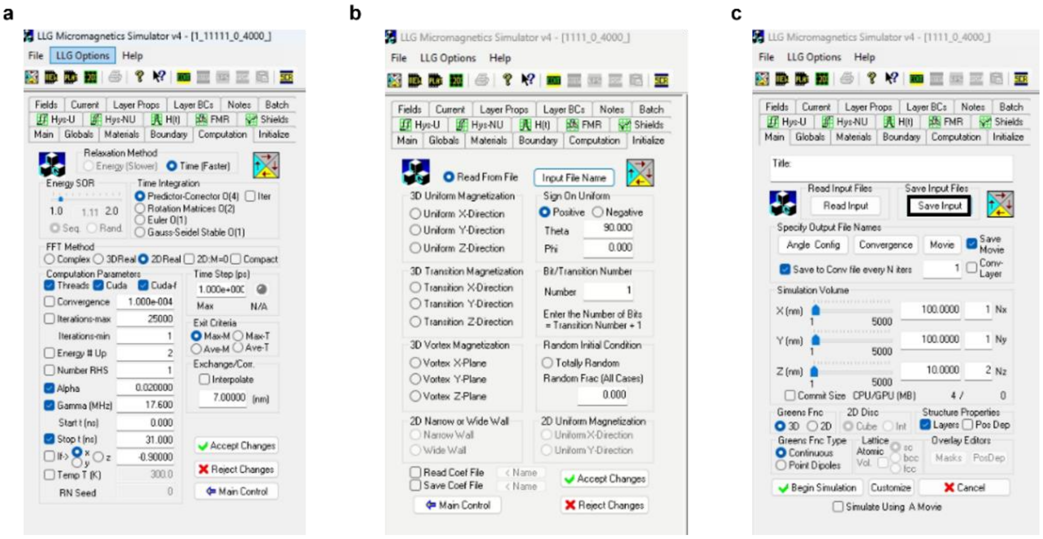
□ Recompute I as M changes

3D Flow / 2D&3D Current Files

Fig. 3.3 The *Current* page to specify the input current pulse and choice of spin torque.

In the *Computation* page (Fig. 3.4 (a)), we specify the computational scheme for solving the LLG equation. Here, we also set value for *Alpha* (damping constant) and *Stop t (ns)* which limits the total processing time of the simulation. Finally, in the *Initialize* page (Fig. 3.4 (b)), we specify the initial state of the system. We can click *Read From File* and *Input File Name* to load an initial state from previous simulations, or specify the initial magnetization configuration by selecting the axis or angle values. Having set up the simulation, we return to the *Main* page and click *Save Input* to save the created simulation parameters. We click on *Begin Simulation* to run the simulation.

In the *Simulation* page, we click on *Start To Compute* to begin the simulation. The *Turn Graphics On/Off* button allows us to turn on/off the display of live update of any of the selected parameters in the viewing panel. We can choose the parameter to be viewed from the available options, on the *Views* page (Fig. 3.5). For example, turning on *M(t)* allows us to see how the components of magnetization along the three Cartesian axes evolve with time.



a

LLG Micromagnetics Simulator v4 - [1_11111_0_4000_]

Relaxation Method: ○ Energy (Slower) ● Time (Faster)

Energy SOR: 1.0 1.11 2.0 ● Seq. ○ Rand.

Time Integration: ● Predictor-Corrector O(4) □ Iter ○ Rotation Matrices O(2) ○ Euler O(1) ○ Gauss-Seidel Stable O(1)

FFT Method: ○ Complex ○ 3DReal ● 2DReal □ 2D:M=0 □ Compact

Computation Parameters: ☑ Threads ☑ Cuda ☑ Cuda-f

□ Convergence	1.000e-004
□ Iterations-max	25000
Iterations-min	1
□ Energy # Up	2
□ Number RHS	1
☑ Alpha	0.020000
☑ Gamma (MHz)	17.600
Start t (ns)	0.000
☑ Stop t (ns)	31.000
□ If-> ● x ○ y ○ z	-0.90000
□ Temp T (K)	300.0
RN Seed	0

Time Step (ps): 1.000e+00C Max N/A

Exit Criteria: ● Max-M ○ Max-T ○ Ave-M ○ Ave-T

Exchange/Corr.: □ Interpolate 7.00000 (nm)

b

LLG Micromagnetics Simulator v4 - [1111_0_4000_]

● Read From File

3D Uniform Magnetization: ○ Uniform X-Direction ○ Uniform Y-Direction ○ Uniform Z-Direction

Sign On Uniform: ● Positive ○ Negative; Theta 90.000; Phi 0.000

3D Transition Magnetization: ○ Transition X-Direction ○ Transition Y-Direction ○ Transition Z-Direction

Bit/Transition Number: Number 1; Enter the Number of Bits = Transition Number + 1

3D Vortex Magnetization: ○ Vortex X-Plane ○ Vortex Y-Plane ○ Vortex Z-Plane

Random Initial Condition: ○ Totally Random; Random Frac (All Cases) 0.000

2D Narrow or Wide Wall: ○ Narrow Wall ○ Wide Wall

2D Uniform Magnetization: ○ Uniform X-Direction ○ Uniform Y-Direction

□ Read Coef File < Name
□ Save Coef File < Name

c

LLG Micromagnetics Simulator v4 - [1111_0_4000_]

Title:

Read Input Files; Save Input Files

Specify Output File Names: ☑ Save Movie

☑ Save to Conv file every N iters 1 □ Conv-Layer

Simulation Volume:

X (nm)	1 5000	100.0000	1	Nx
Y (nm)	1 5000	100.0000	1	Ny
Z (nm)	1 5000	10.0000	2	Nz

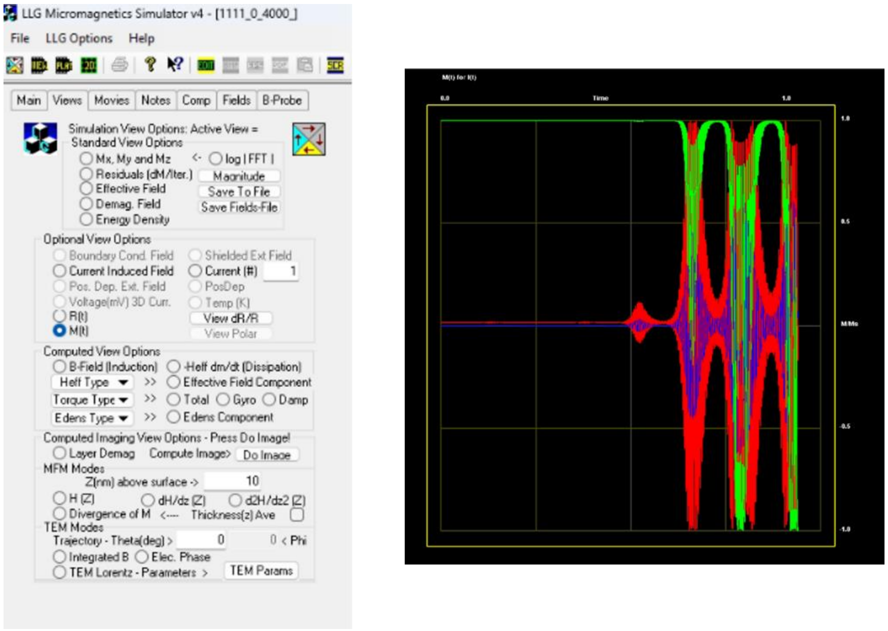
□ Commit Size CPU/GPU (MB) 4 / 0

Greens Fnc: ● 3D ○ 2D; 2D Disc: ○ Cube ○ Int; Structure Properties: ☑ Layers □ Pos Dep

Greens Fnc Type: ● Continuous ○ Point Dipoles; Lattice Atomic Vol. □ ○ sc ○ bcc ○ fcc; Overlay Editors

□ Simulate Using A Movie

Fig. 3.4 a. The ***Computation*** page for specifying the method of solving the LLG equation and other computation parameters **b,** the ***Initialize*** page for specifying the initial condition of the modelled system. **c,** the simulation parameters that have been set up are saved using the ***Save Input*** option.



LLG Micromagnetics Simulator v4 - [1111_0_4000_]

Simulation View Options: Active View = Standard View Options

○ Mx, My and Mz <- ○ log | FFT |
○ Residuals (dM/Iter.)
○ Effective Field
○ Demag. Field
○ Energy Density

Optional View Options

○ Boundary Cond. Field ○ Shielded Ext Field
○ Current Induced Field ○ Current (#) 1
○ Pos. Dep. Ext. Field ○ PosDep
○ Voltage(mV) 3D Curr. ○ Temp (K)
○ R(t)
● M(t)

Computed View Options

○ B-Field (Induction) ○ -Heff dm/dt (Dissipation)
>> ○ Effective Field Component
>> ○ Total ○ Gyro ○ Damp
>> ○ Edens Component

Computed Imaging View Options - Press Do Image!

○ Layer Demag Compute Image>

MFM Modes

Z(nm) above surface -> 10

○ H (Z) ○ dH/dz (Z) ○ d2H/dz2 (Z)
○ Divergence of M <---- Thickness(z) Ave □

TEM Modes

Trajectory - Theta(deg) > 0 0 < Phi
○ Integrated B ○ Elec. Phase
○ TEM Lorentz - Parameters >

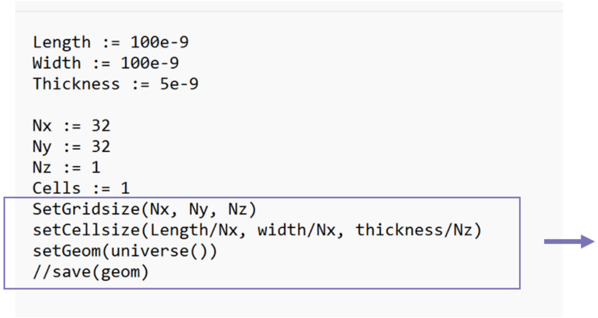
Fig. 3.5 When running the simulation, we can choose to view different parameters. The time evolution of M(t) separated into the three Cartesian components is shown as an example.

3.1.2. Mumax3 – Simulation & Data processing.

Mumax3 is a script-based simulator and has the convenience to modify add user defined parameters in the open source code [2]. Here we briefly explain the script-based model and execution of simulations.

Simulation file and parameters:

1. The simulation parameters such as the sample size, thickness can be defined and loaded in to simulation model as mumax3 API.

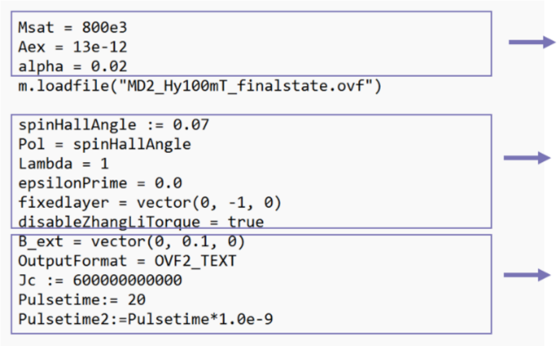


```
Length := 100e-9
Width := 100e-9
Thickness := 5e-9

Nx := 32
Ny := 32
Nz := 1
Cells := 1
SetGridsize(Nx, Ny, Nz)
setCellsize(Length/Nx, width/Nx, thickness/Nz)
setGeom(universe())
//save(geom)
```

User defined parameters:
Editable naming conventions

Simulation size definition
Non editable source command

2. Magnetic and STT/SOT input parameters.



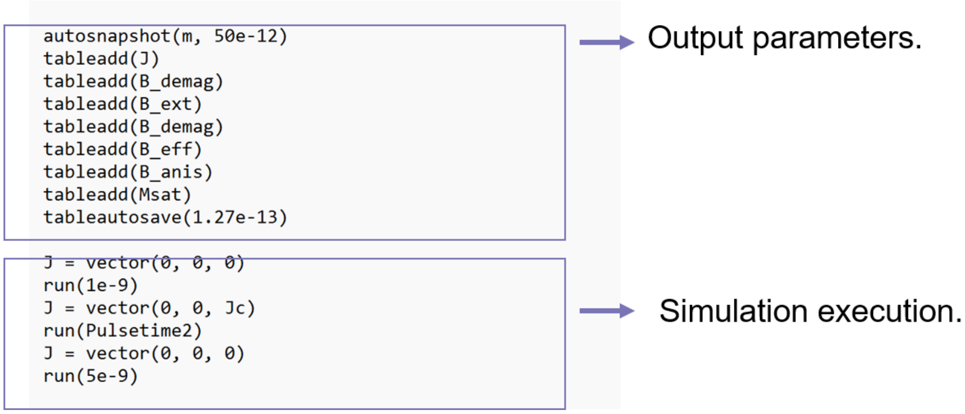
```
Msat = 800e3
Aex = 13e-12
alpha = 0.02
m.loadfile("MD2_Hy100mT_finalstate.ovf")

spinHallAngle := 0.07
Pol = spinHallAngle
Lambda = 1
epsilonPrime = 0.0
fixedlayer = vector(0, -1, 0)
disableZhangLiTorque = true
B_ext = vector(0, 0.1, 0)
OutputFormat = OVF2_TEXT
Jc := 600000000000
Pulsetime:= 20
Pulsetime2:=Pulsetime*1.0e-9
```

Magnetic material parameters.

STT/SOT parameters.
Fixed layer vector set for
SOT condition.

Applied magnetic field and current
density parameters.

3. Output and running simulations.



```
autosnapshot(m, 50e-12)
tableadd(J)
tableadd(B_demag)
tableadd(B_ext)
tableadd(B_demag)
tableadd(B_eff)
tableadd(B_anis)
tableadd(Msat)
tableautosave(1.27e-13)
```

```
J = vector(0, 0, 0)
run(1e-9)
J = vector(0, 0, Jc)
run(Pulsetime2)
J = vector(0, 0, 0)
run(5e-9)
```

Output parameters.

Simulation execution.

3.2. Artificial neural network implementation using Python and MATLAB.

For the image recognition tasks, the Modified National Institute of Standards and Technology database (MNIST) [3] images are converted to binary black and white images, following which each image is divided into 4-bit patterns striding horizontally, resulting in a 196 × 4 matrix. In the readout layer, we use both linear and logistic regression functions executed in Matlab software. During learning process, for a particular image, the output amplitudes are mapped into a one dimensional feature vector of 196 elements. This process is repeated for all 60000 training images, creating a 60000 × 196 output matrix **O**. The 196 × 10 weight matrix **W** (where each column corresponds to each digit from 0 to 9) is calculated using the output matrix **O** and a label matrix **L** (60000 × 10) containing the true labels for each training image. In each row of the label matrix, the (l+1)th column has a value 1 and the remaining columns have value 0, where l is the true digit (l=0, 1...9).

In linear regression, assuming a linear relationship between the output matrix **O** and the label matrix **L**,

$$\boldsymbol{OW} = \boldsymbol{L}. \tag{8}$$

We find the weight matrix **W** using the pseudo inverse $\mathbf{O}^{\dagger}$.

$$\boldsymbol{W} = \boldsymbol{O}^{\dagger}\boldsymbol{L}. \tag{9}$$

In logistic regression, we assume the relation between **O** and **L** to be of the form,

$$\boldsymbol{g_W(O)} = \frac{\mathbf{1}}{\mathbf{1} + \boldsymbol{e^{-OW}}} = \boldsymbol{L}. \tag{10}$$

We find weight matrix **W** by the minimization of a cost function **J(W)** given by,

$$\boldsymbol{J(W) = \frac{1}{m}\sum_{i=1}^{m}\left[-L^{(i)}\,log\left(g_W\left(O^{(i)}\right)\right) - \left(1 - L^{(i)}\right)log\left(1 - g_W\left(O^{(i)}\right)\right)\right]}, \tag{11}$$

where **m** is the number of iterations. We ran 12000 iterations for the weight matrix **W** optimization. The cost function **J(W)** is minimized by the method of gradient descent, given by,

$$\boldsymbol{\frac{\partial J(W)}{\partial W_j} = \frac{1}{m}\sum_{i=1}^{m}(g_W\left(O^{(i)}\right) - L^{(i)})O_j{}^{(i)}}. \tag{12}$$

References

[1] M. R. Scheinfein and E. A. Price, “LLG Micromagnetic Simulator,” *LLG Micromagnetic Simulator*, 2015.

[2] A. Vansteenkiste, J. Leliaert, M. Dvornik, M. Helsen, F. Garcia-Sanchez, and B. Van Waeyenberge, “The design and verification of Mumax3,” *AIP Adv.*, vol. 4, no. 10, p. 107133, 2014, doi: 10.1063/1.4899186.

[3] Y. LeCun, C. Cortes, and C. J. C. Burges, “The MNIST Database of Handwritten Digits,” *MNIST Database Handwrit. Digit.*, [Online]. Available: http://yann.lecun.com/exdb/mnist/.

Chapter 4

Classification task using spin Hall oscillators

In this chapter, we explore the utilization of a Spin Hall oscillator (SHO) device for the classification of binary input patterns. We introduce a novel approach involving the spectral filter technique to directly classify multibit binary data, simplifying feature extraction. Our primary objective is to reduce feature map dimensionality, facilitating rapid and efficient information processing with minimal training costs. By manipulating the magnetization dynamics of the SHO through input pulse pattern configuration, we demonstrate the classification capability of input sequences containing 4-binary digit data.

4.1. Simulation Model

A conceptual schematic of the hardware is illustrated in Figure 1a, comprising a pulse input circuit, SHO (Spin Hall Oscillator), and an electrical output circuit [1]. The modeled SHO is constructed using a platinum/ permalloy bilayer (Pt/NiFe) with lateral dimensions of 100 nm x 100 nm, and each layer has a thickness of 5 nm. The input binary data is represented by a current pulse, where "1" and "0" are encoded as distinct current values, I_1 and I_0, respectively. An in-plane magnetic field ($\mu_0 \mathrm{H}_{ext}$) of 100 mT is applied at an angle $\varphi = 90^0$ (+Y direction) to align the magnetization perpendicular to the current direction. The temporal dynamics of the ferromagnetic layer was solved by the Landau-Lifshitz-Gilbert (LLG) equation with the spin transfer torque term,

$$\frac{d\widehat{\boldsymbol{m}}}{dt} = -\gamma\widehat{\boldsymbol{m}} \times \mu_0 \boldsymbol{H}_{eff} + \alpha\widehat{\boldsymbol{m}} \times \frac{d\widehat{\boldsymbol{m}}}{dt} - \gamma\frac{\hbar}{2|e|}\frac{\theta_{SH}\ |j_c|}{M_s t_{FM}}\left(\widehat{\boldsymbol{m}} \times (\widehat{\boldsymbol{m}} \times \widehat{\boldsymbol{\sigma}})\right) \tag{4.1}$$

where $\widehat{\mathbf{m}} = \frac{\mathbf{M}}{\mathrm{M_s}}$ is the normalized magnetization vector, γ is the gyromagnetic ratio, α is the Gilbert damping parameter, μ_0 is the vacuum permeability, $\mathrm{M_s}$ is the saturation magnetization, $\hbar$ is the reduced Planck constant, e is the electron charge, and $\mathrm{t_{FM}}$ is the thickness of the magnetic layer. The effective field $\mathbf{H}_{\mathrm{eff}}$ includes the external magnetic field, the magneto-crystalline anisotropy field, and the demagnetization field. θ_{SH} is the spin Hall angle, which characterizes the conversion efficiency of charge current density $\hat{\mathbf{j}}_{\mathbf{c}}$ to spin current density $\hat{\mathbf{j}}_{\mathbf{s}}$ in the heavy metal layer. $\widehat{\boldsymbol{\sigma}} = -\mathrm{sgn}\ \theta_{\mathrm{SH}}\ (\hat{\mathbf{z}} \times \hat{\mathbf{j}}_{\mathbf{c}})$ is the orientation of spin injected into the ferromagnet, where $\hat{\mathbf{z}}$ and $\hat{\mathbf{j}}_{\mathbf{c}}$ are the unit vectors in the direction of surface normal and the electrical current, respectively. The $-$sgn factor changes with the position of ferromagnet, i.e., if the ferromagnet is atop the HM, $\hat{\mathbf{z}}$ would face into the HM, or if the ferromagnet is below the HM, $\hat{\mathbf{z}}$ would face into the ferromagnet but it's sign would be opposite to the prior case. In accordance with experiments, the material parameters used in the simulations are listed in Table.1 [2]

Parameters	**Symbol**	**Value**
Saturation magnetization	$\mu_0 M_s$	1.0 T
Exchange constant	A_{ex}	$1.13 \times \ 10^{-12}$J m^{-3}
Damping	α	0.02
Thickness of NiFe	t_{NiFe}	5 nm
Thickness of Pt	t_{Pt}	5 nm
Resistivity of NiFe	ρ_{NiFe}	$4.5 \times \ 10^{-7}\ \Omega$ m
Resistivity of Pt	ρ_{pt}	$2.0 \times \ 10^{-7}\ \Omega$ m
Spin Hall angle	θ_{SH}	0.07

Table 1. Simulation material parameters.

The working principle of the SHO is as follows: when a current (Ic) flows in the +X direction, the Spin Hall Effect (SHE) in Pt causes spin-dependent scattering of electrons, leading to spin accumulation at the Pt/NiFe interface. This accumulation results in the transfer of spin angular momentum to NiFe, inducing a transverse flow of spin current (+Z direction), characterized by the spin Hall conversion efficiency θ_{SH} (spin Hall angle) [3], [4]This spin current gives rise to two Spin-Orbit Torques (SOTs): damping-like torque (DLT) and field-like torque (FLT) [5], [6]. In this context, we consider only the role of DLT due to the negligible effect of FLT for the 5 nm thick Pt Layer. By increasing Jc (current density), DLT can be adjusted to compensate for the natural damping of NiFe, achieving auto-oscillations in the gigahertz frequency range [7]–[9].

The electrical detection of the SHO's oscillatory dynamics relies on the oscillating anisotropic magnetoresistance (AMR) of the ferromagnet [10], [11]. The dependence of the FM resistance on the angle θ_{M} between current and magnetization is $R(\theta_{\mathrm{M}}) = R_{\perp} + (R_{\parallel} - R_{\perp})cos^2\theta_M = R_0 + \Delta R_{AMR}\, cos^2\theta_M$, where $R_{\perp(\parallel)}$ is the device resistance when the current and the magnetization are oriented perpendicular (parallel) to each other. $\Delta R_{AMR} = R_{\parallel} - R_{\perp}$is the AMR resistance and R_0 is the minimum resistance of the device. The microwave voltage signal across the device is,

$$V(t) = I_c \times R(t) = \ I_c \times\ [R_0 + \Delta R_{AMR}\, cos^2\theta_M(t)]. \tag{4.2}$$

Since the magnetization dynamics occur in unit sphere and precess around the effective field (H_{eff}), the time dependent $\theta_{\mathrm{M}}(t)$ consists of an in-plane component θ_{in} and an out-of-plane component θ_{out} and can be decomposed as[41],

$$\cos(\theta_M(t)) = \cos(\theta_{in}(t))\cos(\theta_{out}(t)), \tag{4.3}$$

$$\theta_{in}(t) = \ \varphi_0 + \ \varphi_c \cos(2\pi f)\,, \tag{4.4}$$

$$\theta_{out}(t) = \ \vartheta_c \sin(2\pi f)\,, \tag{4.5}$$

where φ_0 is the static angle between the current and the magnetization **M** defined by the effective field $\mathbf{H}_{\mathrm{eff}}$, φ_c and ϑ_c are the in-plane and out-of-plane precession cone angles of the magnetization, respectively. *f* is the frequency of oscillation in gigahertz. For the Cartesian coordinate axes in Fig. 4.1, $\varphi_0 = \ \varphi$ (when $H_{eff} = H_{ext}$) and $\sin(\theta_{in}) \approx \frac{M_X}{M_0}$ (normalized magnetization component), such that the Mx^2 is related to the detected output voltage V(t). The simulated time dependent Mx is converted into frequency spectra via fast Fourier transform (FFT) to represent the collective behavior of the SHO for the given input signal. The main advantage of frequency domain analysis is the reduction in the amount of output data for further computations.

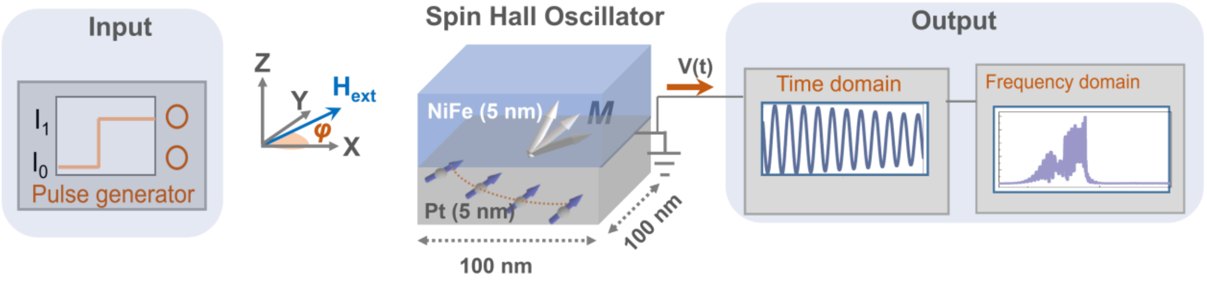


Fig. 4.1 Schematics of the spin Hall oscillator simulation model [1].

4.1.1. Single domain vs multidomain model

Single-cell simulations are computationally less demanding compared to their multicell counterparts. This computational efficiency makes them particularly suitable for preliminary or exploratory studies, especially in cases where computational resources are limited. The reduced complexity allows for quicker simulations and facilitates an initial understanding of the system's behavior. However, it's worth noting that single-cell calculations come with certain limitations. They may not capture small-scale effects such as the presence of domain walls, defects, or localized variations in magnetic properties. Nonetheless, single-cell simulations are invaluable for statistical analysis and for capturing the average behavior of a material. They provide a macroscopic view of the system and can yield insights into its overall characteristics. This makes them particularly useful for practical applications, especially in the design of magnetic devices. In such cases, where the goal is to consider the behavior of the entire system, single-domain models are often necessary. Single-domain models treat the entire magnetic material as a single domain, simplifying the representation but preserving essential features.

In light of these considerations, it becomes imperative to validate simulations by comparing single-cell (single-domain) and multicell (multi-domain) models to detect any nonlinear behavioral changes. In our context, we are primarily interested in the macroscopic behavior of NiFe. Assumptions are made that all magnetic moments are uniform and rotations are coherent, given the small magnetocrystalline anisotropy in NiFe. Thus, the magnetic moments predominantly align with the applied field direction. This alignment is confirmed by our simulations, which maintain the single-domain state in NiFe even under the influence of a 100 mT Y-axis applied field, as depicted in Fig 4.1a and b.

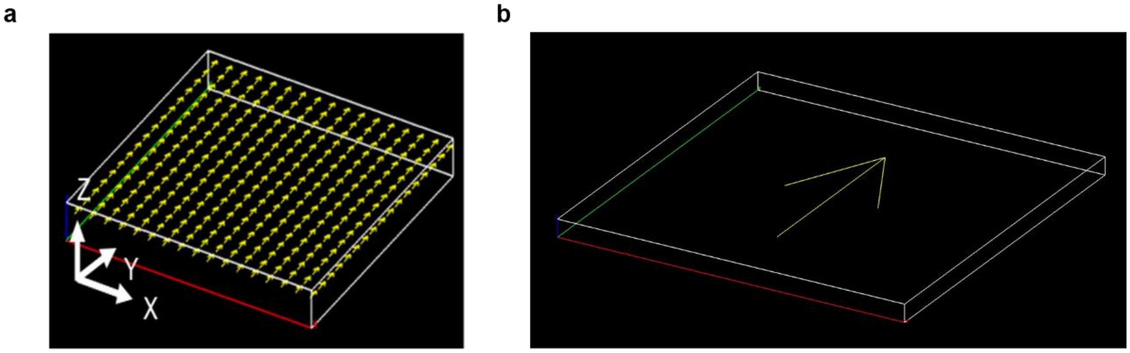


Fig. 4.2 a Multicell simulation model, **b**. single cell simulation model.

In the MD model (multicell), as shown in Fig. 4.3a, the excitation at $I_1 = 2.0$ mA initiates approximately at the center of the pulse width (5 ns). Conversely, in the SD model (single-cell), the excitation occurs at a time period greater than half of the pulse width as can be seen in the Fig.4.3b. Furthermore, the amplitude of excitation appears stronger in the SD model, indicating differences in the response of the two approaches. This behavior persists for the $I_1 = 3.0$ mA excitation, as depicted in the comparison between MD and SD models in Figs. 4.3c and d. In both cases, oscillations eventually reach a steady-state stage, but the initiation time varies between the two approaches.

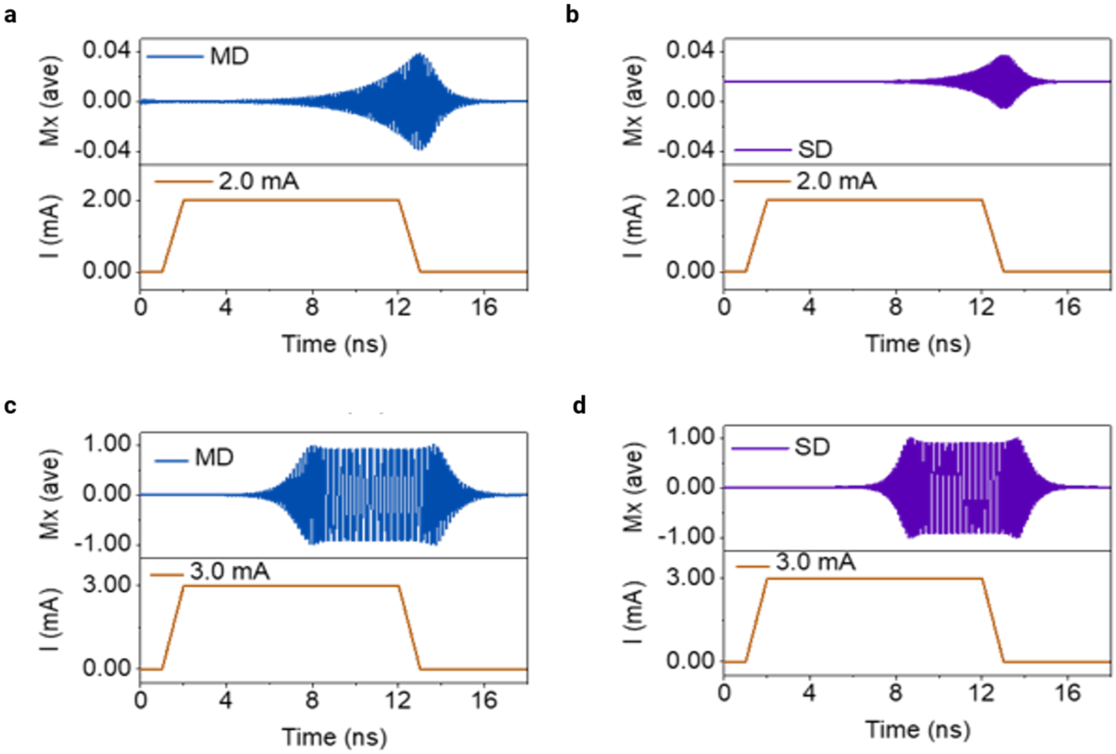


Fig. 4.3 a Single cell simulation magnetization dynamics Mx for $I_1 = 2.0$ mA **b**. Multicell simulation dynamics for $I_1 = 2.0$ mA, **c** multicell simulation dynamics for $I_1 = 3.0$ mA **d** multicell simulation dynamics for $I_1 = 3.0$ mA.

To further explore this, Fig 4.4 provides comparative plot of the oscillation frequencies as a function of I_1 for pulse width, $\tau = 10$ ns, respectively, in both MD and SD models. Notably, a reduction in oscillation frequency is observed in the MD model when compared to the SD model for both pulse widths. This reduction in frequency can be attributed to the inclusion of the exchange field A_{ex} in the effective field $\mathbf{H}_{\mathrm{eff}}$ within the multicell model. The exchange energy contributions tend to reduce the frequency of the oscillation, influencing the system's dynamics. In both SD and MD models, once steady-state oscillations are achieved, the oscillation frequency exhibits a nonlinear redshift. This behavior aligns with expectations for an in-plane magnetized film. It's important to note that this nonlinearity is primarily due to the demagnetization factor when other nonlinearities are disregarded. Consequently, considering the computational efficiency of single-cell simulations and their ability to provide valuable insights into the system's macroscopic behavior, the decision was made to continue with single-cell approximation (single-domain model) for the subsequent simulations. This choice helps streamline computational demands while retaining essential characteristics of the SHO system.

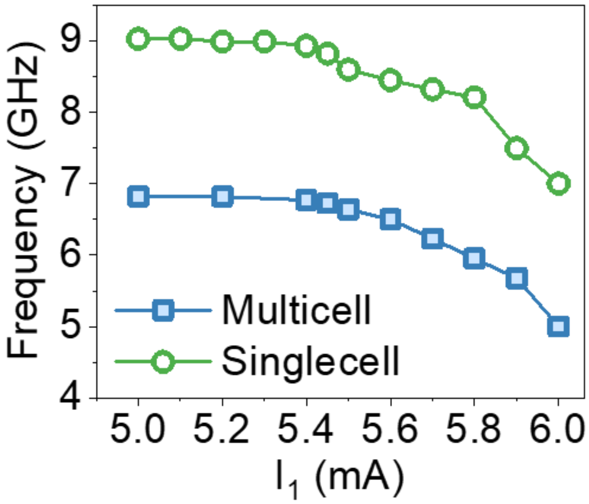


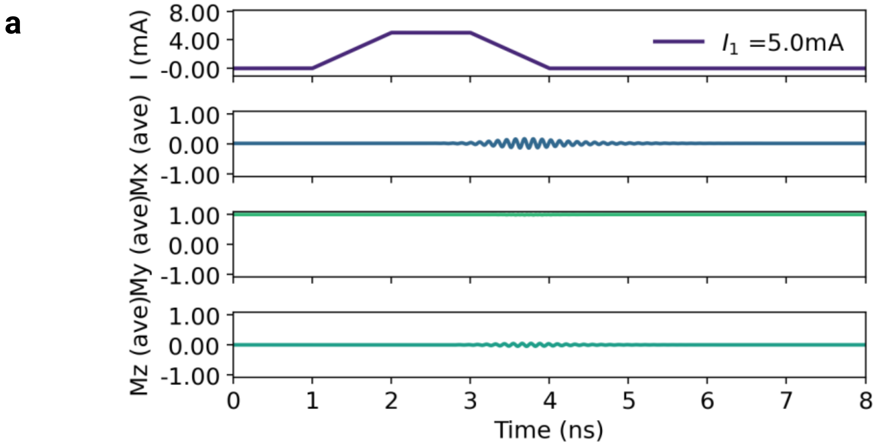
Fig. 4.4 Oscillation frequency vs applied pulse current I_1 for single cell and multicell simulations.

4.2. Input driven magnetization dynamics

We begin by examining the magnetization dynamics of the modelled SHO device as a function of I_1 for a single current pulse with a pulse width (τ) of 3 ns and pulse rise and fall times of 1 ns. In Fig. 4.5 a-d, the magnetization components temporal evolution is depicted for I_1 = 5.0, 5.5, 6.0, and 6.5 mA with I_0 = 0 mA. Their corresponding trajectories are plotted in Fig. 4.6 a-d. SOT excitation of a ferromagnetic resonance mode causes magnetization oscillations at I_1 = 5.0 and 5.5 mA. They exhibit small angle precession, in which the cone angle increases with an increase of I_1[12]–[14]. For I_1 = 6.0 and 6.5 mA, the oscillations correspond to the in-plane and out-of-plane auto-oscillation modes, respectively.

The intiation and relaxation of the magnetization oscillation are compared with Mx component and shown by the colored circles in Fig. 4.7a, the initiation and relaxation times vary depending on the strength of I_1. This indicates the manipulation of effective damping by the SOT. These excitations can be converted into self-sustaining auto-oscillations by gradually increasing the precession amplitude with τ until it saturates at the limit cycle of auto-oscillation[15].The equilibrium energy, which regulates the auto-oscillation's limit cycle, is influenced by the device geometry, mode of excitation, and direction of the effective field. The auto-oscillation orbit for the single magnetic domain model is circular in the out-of-plane direction and shaped like a clamshell in the in-plane direction [13]. The single cycle trajectories for each oscillation mode are depicted in Fig. 4.7b.

In Fig. 4.8, the FFT amplitude spectrum for the I_1 range (5 mA to 6 mA) is displayed. For $I_1 < 5.5$ mA, the FFT amplitude increases linearly with increasing I_1, but the oscillation frequency, which can be seen from the peak position of the FFT amplitude in the spectra, is constant at 9.0 GHz. For 5.5 mA $< I_1 <$ 6.0 mA, the frequency undergoes a red shift due to the large angle motion of magnetization components, as can be seen in Fig. 4.7b (I_1 = 6.0 mA), which reduces the effective demagnetization field in the NiFe layer. The magnetization component transverse to H_{ext} undergoes oscillations at twice the oscillation frequency in order to maintain a constant magnitude and thus reduces the frequency. The constant frequency 9.0 GHz corresponds to the ferromagnetic resonance frequency (FMR). The ferromagnetic resonance frequency depends on the H_{eff} and the sample dimension.



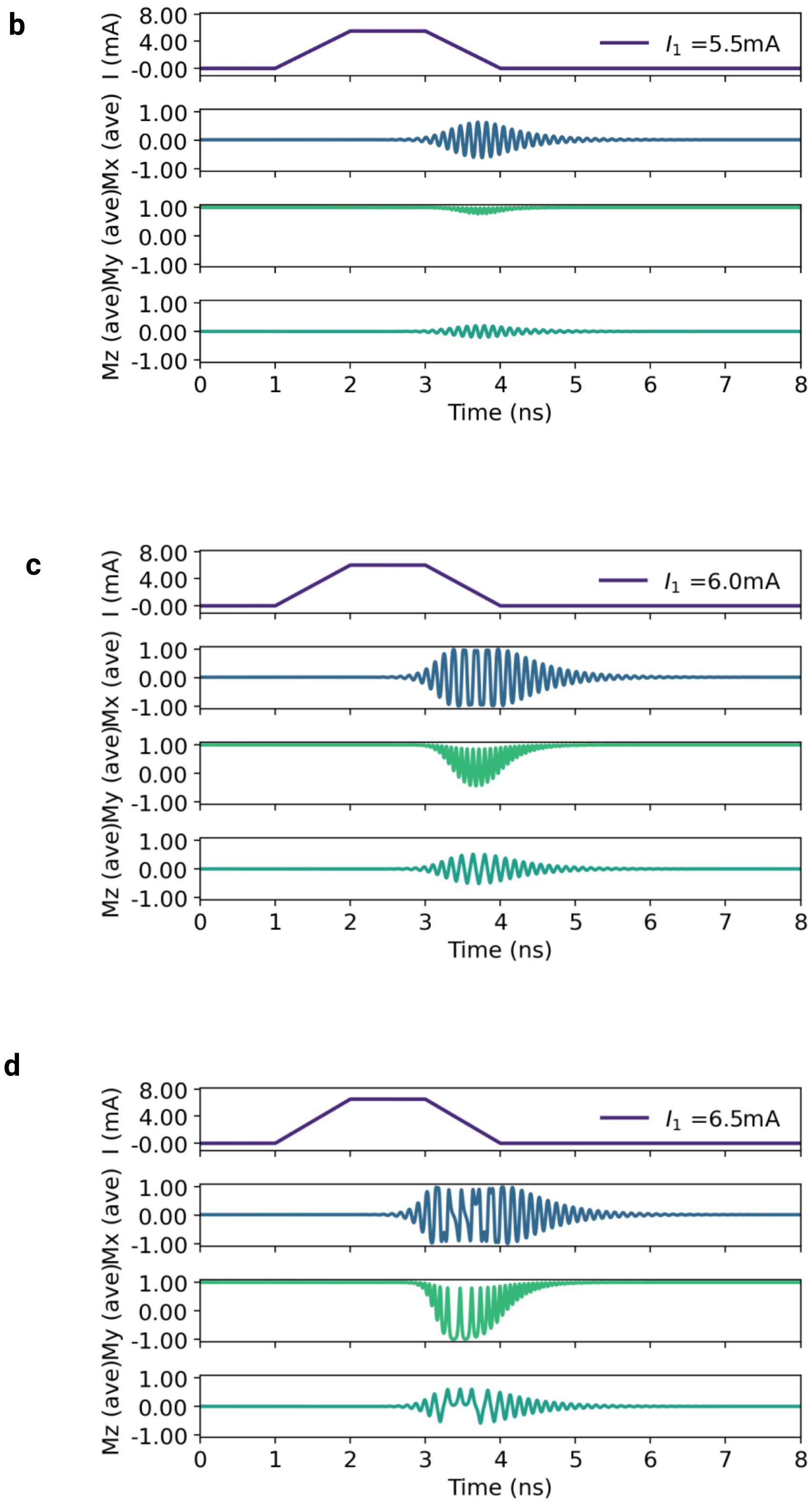


Fig. 4.5 a-d Magnetization oscillation plots for pulsed current excitation for I_1= 5.0 to 6.5 mA.

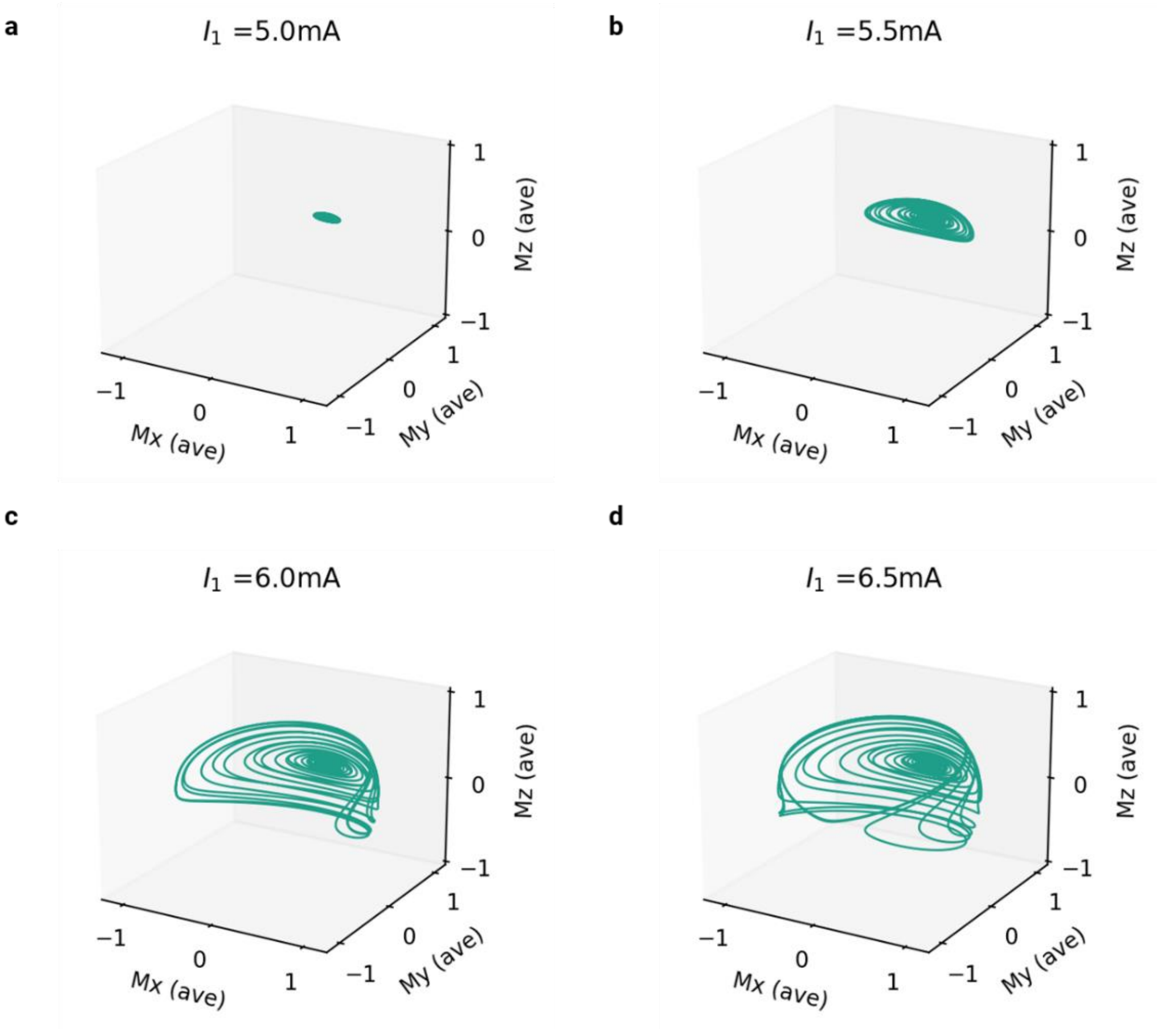


Fig. 4.6 a-d Oscillation trajectories for pulsed current excitation for I_1= 5.0 to 6.5 mA.

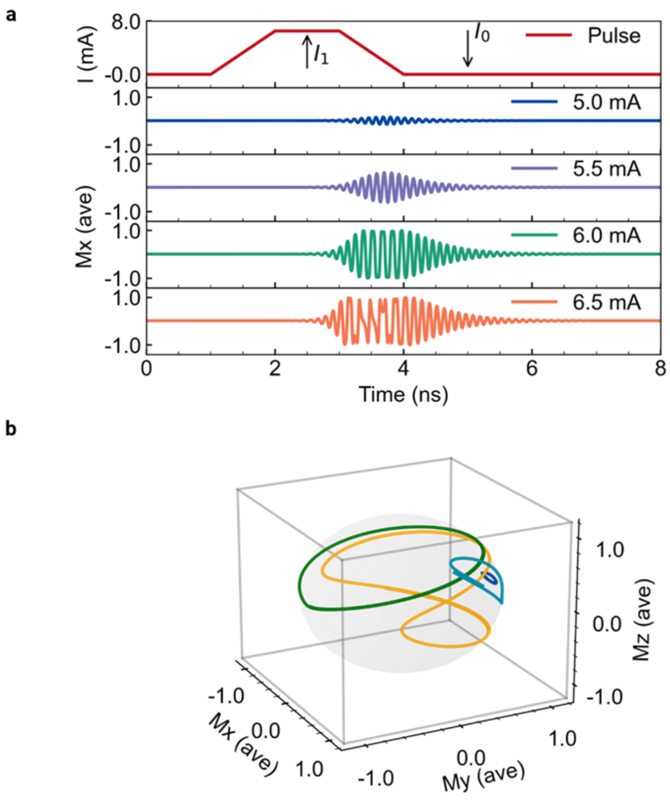


Fig. 4.7 a Pulsed input current and magnetization dynamics of Mx component for I_1 = 5.0 - 6.5 mA, the initiation and relaxation time scales are indicated by colored circles. b. Transition of small angle precession to large angle oscillation trajectories as a function of I_1 [1].

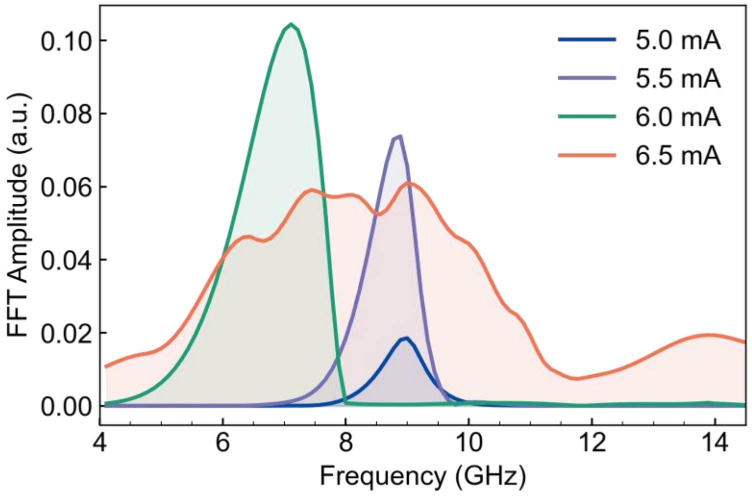


Fig. 4.8 Fast Fourier amplitude spectrum (FFT) of Mx for I_1 = 5.0 - 6.5 mA [1].

To verify this, we simulated the FMR for modelled SHO is shown in Fig. 4.9a. The applied magnetic field (H_{ext}) is oriented along Y axis and the oscillating magnetic field (H_{rf}) at a fixed microwave frequency is applied along X axis with a strength of 1 mT. The H_{ext} is sweeped for fixed frequency. The obtained resonant field (H_o) is plotted in Fig 4.9b as a function of the applied frequency. The data are fitted with the Kittel equation $f = \mu_0\gamma/2\pi\sqrt{H_o(H_o + M_{eff})}$ yielding an effective magnetization, $\mu_0 M_{eff} = 1.0$ T with gyromagnetic ratio $\gamma/2\pi = 27.6$ GHz/T.

As I_1 is increased further, the frequency decreases, reaching 7.1 GHz at $I_1 = 6.0$ mA. This frequency shift is attributed to the complex coupling of oscillatory amplitude and phase as predicted by the nonlinear auto-oscillator theory for STOs [16]. For $I_1 = 6.5$ mA, the frequency increases to 9.1 GHz and the FFT amplitude reduces due to the out-of-plane oscillation, as can be seen in Fig. 4.10 ($I_1 = 6.5$ mA). As a result, the magnetization dynamics in the SHO can be divided into three regimes, as shown in Fig. 1e: the linear excitation regime for $I_1 < 5.5$ mA, where the frequency is constant with increasing precession amplitudes as a function of I_1, the nonlinear excitation regime for 5.5 mA < I_1 < 6.0 mA, where the FFT amplitude is saturated and the frequency decreases drastically with increasing I_1 and the out-of-plane oscillation regime for $I_1 > 6.5$ mA where the frequency increases and the FFT amplitude decreases. These nonlinear frequency amplitude relationships can be used to classify the inputs.

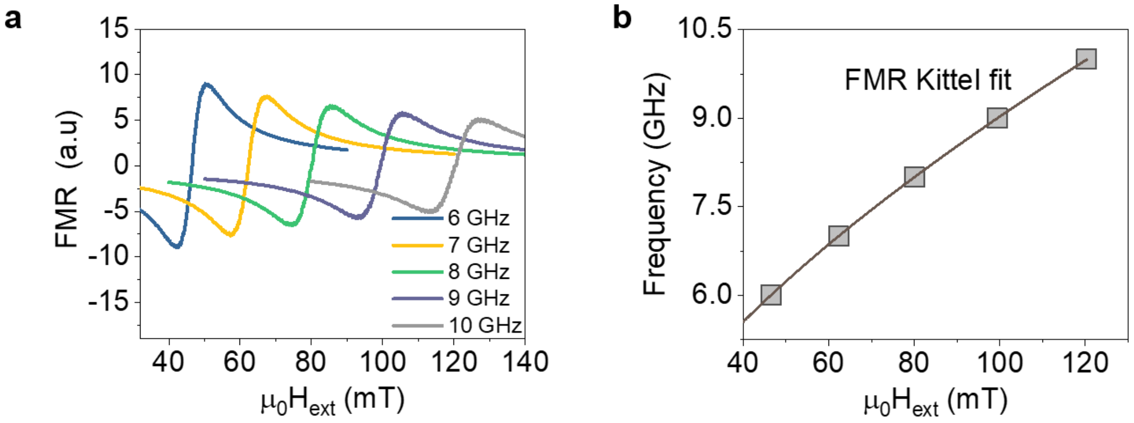


Fig. 4.9 a Simulated ferromagnetic resonance (FMR) for modelled spin Hall oscillator (SHO) in the main text. **b** Fitted resonant field (H_o) as a function of the applied frequency.

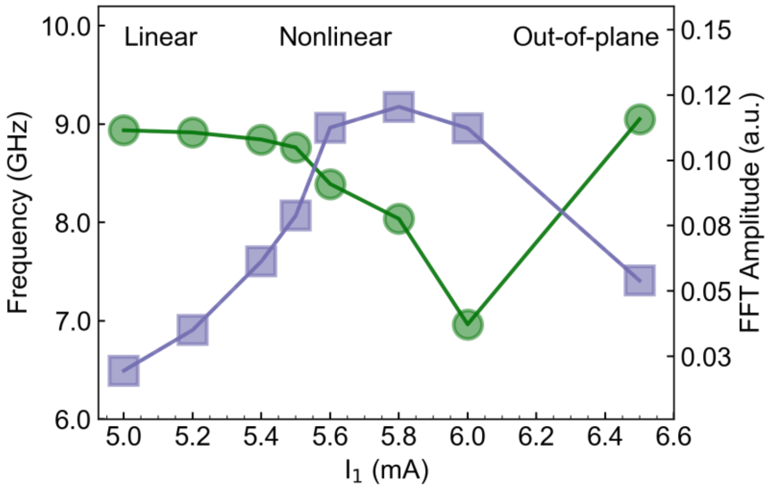


Fig. 4.10 Frequency response and peak amplitude level of FFT spectrum as a function of I_1 [1].

4.3. Binary digit pattern classification

Regular pulse scheme

After investigating magnetization dynamics, we look into the SHO's ability to classify n-binary input data. The pulse stream is represented by the encoded input signal $\mathrm{n-bi(t)}$, which has current values I_0 and I_1 for "1" and "0", respectively. The pulse period (Δt) and width (τ), respectively, 4 ns and 3 ns. The pulse width τ, includes a rise time of 1 ns and a fall time of 1 ns. Figure 4.11 represents $4-\mathrm{bi(t)}$ input pulse patterns, Mx responses, and FFT amplitude spectra (frequency) with input current values of $I_1 = 3.5$ mA and $I_0 = 0$ mA, which lie in the linear excitation regime. For the input pattern 1111 in Fig. 2a, the Mx response in Fig. 4.11b shows the magnetization oscillations with varying amplitude for each "1" input, and the corresponding FFT spectrum is shown in Fig. 4.11c. For the 1001 pattern in Fig. 4.11d, the magnetization relaxes to its initial state in the time between the two "1" inputs, as shown in Fig. 4.11e. The corresponding amplitudes in the FFT spectrum (Figs. 4.11c and 4.11f) for the input patterns of 1111 and 1001 allow one to clearly see the difference in magnetization dynamics.

Classification using Filtered output

We apply the filter neuron concept, which was inspired by recurrent neural networks' use of it for feature extraction and handling time-varying outputs, to filter a specific feature in the output data [17], [18]. The FFT amplitude value at the linear excitation regime frequency, 9.0 GHz, is fixed as the filter characteristic in order to separate the input patterns in the output spectrum. The filtered amplitude values for the input patterns 1111 and 1001 at 9.0 GHz are 0.0042 and 0.00002, respectively, as shown with the help of guidelines in Figs. 4.11c and 4.11f, respectively. Note that this filtering strategy differs from the standard bit slicing techniques used in the computing paradigm. The bit slicing method maps n input elements to n output values and then performs additional computations for weight optimization[19]. But in this case, n input elements are mapped to a single output value using the quantization technique. This method is well suited for the reduction of output data and does not call for weight optimization for the classification task of input patterns, which can lower the computation costs [20], [21].

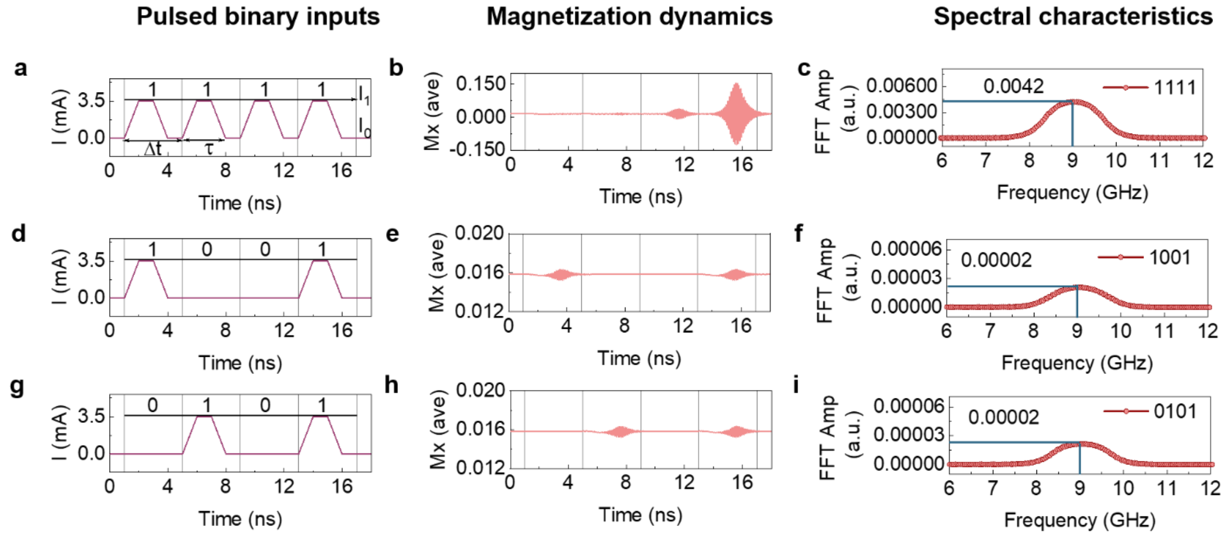


Fig. 4.11 a. Regular pulse scheme for 4-binary digit input pattern 1111 with pulse period (Δt) of 4 ns and pulse width (τ) of 3 ns. **b.** Magnetization dynamics of Mx component and **c.** corresponding spectral characteristics. Guide line in FFT shows the amplitude value at 9.0 GHz. Similar input pattern, magnetization dynamics and FFT in **d - f** for 1001 pulse pattern, and in **g – i** for 0101 pulse pattern [1].

Challenges in classification

The relaxation of magnetization precession during the interval between two consecutive pulses poses a challenge to the classification task. For the 0101 nput pattern in Fig. 4.11g, Fig. 4.11h displays the Mx response. Since the magnetization has relaxed to its initial state prior to the second "1" pulse arrival, the individual "1" pulses exhibit the same oscillating amplitude. As can be seen in Figs. 4.11f and 4.11i, the resulting FFT spectra have the same amplitude as the 1001 pattern. In this case, it is not possible to distinguish between the SHO's output and any of the possible $4 - \mathrm{bi(t)}$ input patterns. Input pulse parameters I_0, I_1 and τ can be varied to affect the dynamics of the magnetization, but in the linear excitation regime, patterns like 1000 and 0001 still produce the same FFT spectra. This is caused by the same magnetization dynamics, but in a different time frame, for each "1" pulse. We refer to this type of input pulse stream as the regular pulse scheme.

Figures 4.12 a and d show the 4-bit binary input pulse patterns with input parameters $I_0 = 0$, $I_1 = 4.0$ mA, $\Delta t = 4$ ns and $\tau = 3$ ns, for the 4-bit patterns 0101 and 1010 respectively. For both pattern 0101 and pattern 1010, the Mx oscillation amplitudes in the Figs. 4.12 **b** and **e** corresponding to the input bit 1 are the same. This prevents the separation of the two patterns as can be seen from the similar value of FFT amplitude in Figs 4.12**c** and **f**. Figures **g** and **j** show the 4-bit binary input pulse with input parameters with I_1 in the nonlinear regime $I_0 = 0$, $I_1 = 6.0$ mA, $\Delta t = 4$ ns and $\tau = 3$ ns, for the 4-bit patterns 0101 and 1010 respectively. For both pattern 0101 and pattern 1010, the Mx response is auto-oscillations as shown in the Figs. 4.12 **h** and **k,** where bit 1 pulses oscillate at the same amplitude level. This again prevents the separation of the two patterns as seen from the similar value of FFT amplitude at the filtering frequency of 9.0 GHz as shown the Figs 4.12**i** and **l**. Figure 4.13 **a**(**d**) shows the 4-bit binary input pulse with input parameters $I_0 = 0$, $I_1 = 3.5$ mA, $\Delta t = 4$ ns and $\tau = 1.5$ ns (3.6ns), for the 4-bit pattern 1111. For $\tau = 1.5$ ns (Fig. 4.13**b**), due to the relaxation of the excited small angle precession, Mx amplitudes corresponding to each of input bit 1 pulses are the same. However, for $\tau = 3.6$ ns (Fig. 4.13**e**), the next bit 1 pulse arrives before the relaxation of the previously excited Mx precession, leading to progressively increasing amplitudes of oscillation.

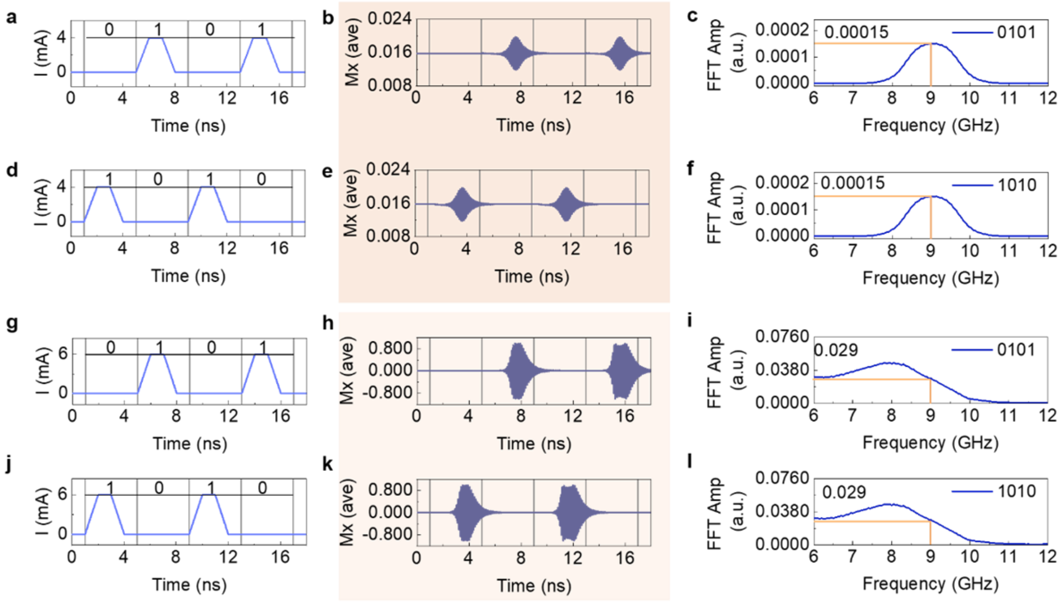


Fig. 4.12 a – l Investigation of magnetization dynamics and 4-bit digit pattern separation on regular pulse scheme with $I_0 = 0$, $I_1 = 4.0$ & 6.0 mA, $\Delta t = 4$ ns, $\tau = 3$ ns [1].

Figures 4.13**c** and 4.13**f** give the FFT amplitude spectra for 4-bit pattern 1111 for $\tau = 1.5$ ns & 3.6 ns respectively. Figure 4.13**g** (**j**) shows the 4-bit binary input pulse with input parameters $I_0 = 0$, $I_1 = 3.5$ mA, $\Delta t = 4$ ns and $\tau = 1.5$ ns (3.6ns), for the 4-bit pattern 0101. For both $\tau = 1.5$ ns (Fig. 4.13**h**) and $\tau = 3.6$ ns (Fig. 4.13**k**), the Mx amplitudes corresponding to both the bit 1 pulses are the same. Figures 4.13**i** and 4.13**l** give the FFT amplitude spectra for 4-bit pattern 1111 for $\tau = 1.5$ ns & 3.6 ns respectively. Similar to the previous pulse scheme, this prevents the separation of any cyclic permutations of 4-bit patterns for both $\tau = 1.5$ ns and $\tau = 3.6$ ns.

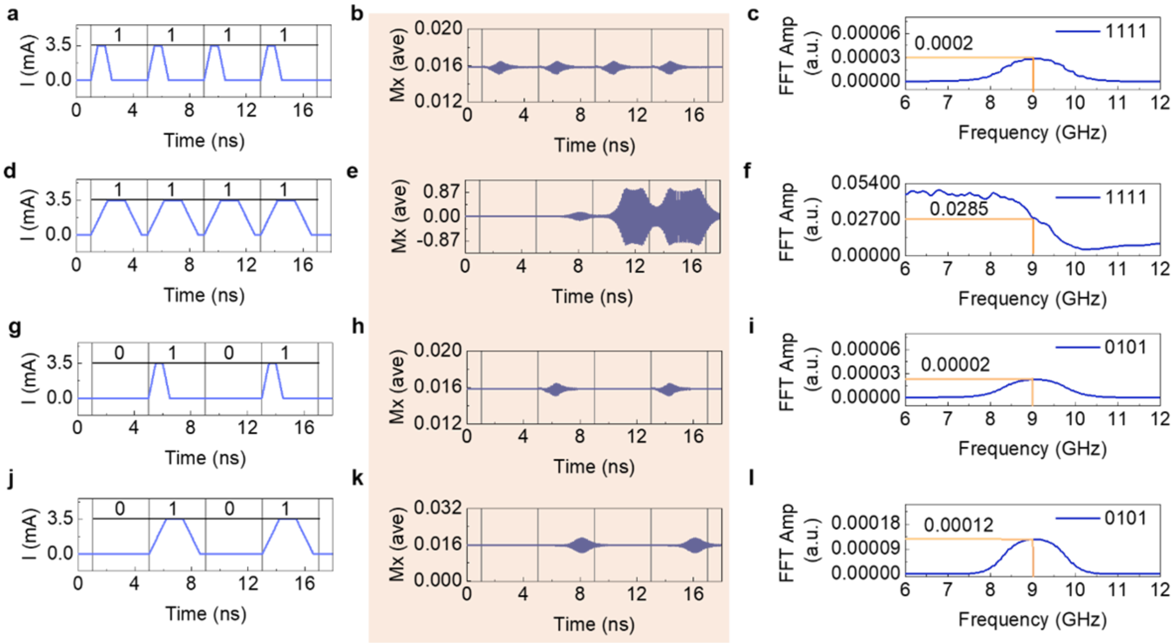


Fig. 4.13 a – l Investigation of magnetization dynamics and 4-bit digit pattern separation on regular pulse scheme with $I_0 = 0$ mA, $I_1 = 3.5$ mA, $\Delta t = 4$ ns, $\tau = 1.5$ ns & 3.6 ns [1].

4.4. Modified pulse scheme

To tackle the challenge faced in the regular pulse scheme, we resort to modifying the input driven magnetization dynamics rather than the internal structure of the device [22]. Figure 4.14a depicts the pulse input of an excitatory pulse I_e, with a pulse width (t_1) of 7 ns. The Mx oscillation, as shown in Fig. 4.14b reaches an amplitude of 0.18 and relaxes to the ground state within 2 ns when $I_e = 3.5$ mA and $I_0 = 0$ mA. This can be clearly seen from the upper envelope plot of Mx shown after the pulse is off at 8 ns in Fig. 4.14c. However, by introducing an offset value for I_0, the relaxation time can be extended. Figure 4.14d shows an excitatory pulse with an offset value for I_0 ($I_e = 3.5$ mA, $I_0 = 1.2$ mA). The oscillation amplitude increases to 0.38 due to the increasing precessional amplitude with the offset current, as shown in Fig. 4.14e, and the relaxation period is extended to 8 ns, as shown in Fig. 4.14f. This allows us to modify the magnetization dynamics in the SHO during the inputs for different $4 - \mathrm{bi(t)}$ patterns, as will be discussed below.

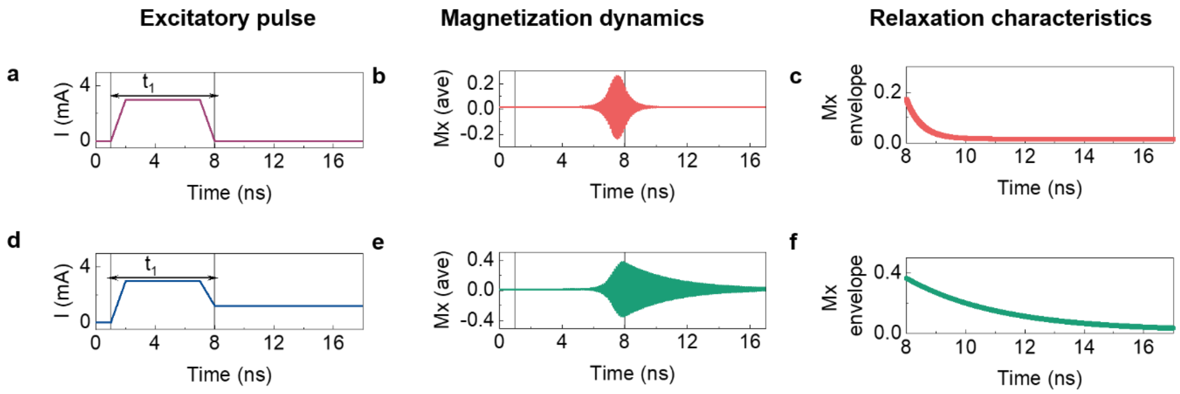


Fig. 4.14. a Excitatory pulse ($I_e = 3.0$ mA) without offset current ($I_0 = 0$ mA). **b** Magnetization dynamics of Mx component and **c** relaxation characteristics plot from the upper envelope of the Mx time domain data after the pulse is off. **d** Excitatory pulse (I_e) with an offset current value ($I_0 = 1.2$ mA). **e**. Magnetization dynamics of Mx component and **f.** relaxation characteristics plot of the Mx time domain data [1].

In order to extend the dynamics of magnetization relaxation for the duration of the input pulse pattern, a modified pulse scheme (I_{mod}) that includes the excitatory pulse (I_e) and a pulse gap (δ) prior to the introduction of the $\mathrm{n - bi(t)}$ is used. Hence, the SHO responds to a combination of two input signals, I_e and bi(t) given by,

$$I_{mod} = \begin{cases} I_e & ; 0 < t < t_1 \\ n - bi(t) & ; t > t_1 + \delta \end{cases}. \tag{4.6}$$

A modified pulse scheme with $I_e = 3.0$ mA, $t_1 = 7$ ns, $\delta = 5$ ns, $I_0 = 1.2$ mA, $I_1 = 2.4$ mA, $\delta = 4$ ns, $\tau = 3$ ns is used for illustration. For simplicity, we denote these input patterns with the above-mentioned parameters as IP_1. Figure 4.15a shows the input pattern of 1010, the Mx response in Fig. 4.15b, and the FFT spectrum in Fig. 4.15c. Similarly, Fig. 4.15d shows the input pattern for 0101, the Mx response in Fig. 4.15e, and the FFT spectrum in Fig. 4.15f. As can be seen from the Mx responses in Figs. 4.15b and 4.15e, each "1" pulse exhibits a different oscillating amplitude as the magnetization dynamics are influenced by both the prior I_e and the corresponding pulses. Since I_e's influence gradually diminishes over time, each output has a unique dynamic, and the degree of influence of previous inputs varies as a function of time. As anticipated, the variation in relaxation dynamics has a significant impact on oscillation amplitude.

Figures 4.15c and 4.15f show the variation in the FFT amplitude and frequency as well as the filtered (9.0 GHz) amplitude values 0.037 and 0.017 for the patterns 1010 and 0101, respectively. Because of the input pattern-specific nonlinear magnetization dynamics, I_{mod} scheme can easily classify from the filtered amplitudes these two input patterns, whereas the previous regular pulse scheme could not. Similarly, the 1000 and 0001 patterns can also be distinguished using the filtered amplitude due to the variations in the relaxation rates. For the I_{mod} scheme, the FFTs were obtained from the Mx in the range of input pulse patterns $n - bi(t)$ and are displayed in Fig. 4.16.

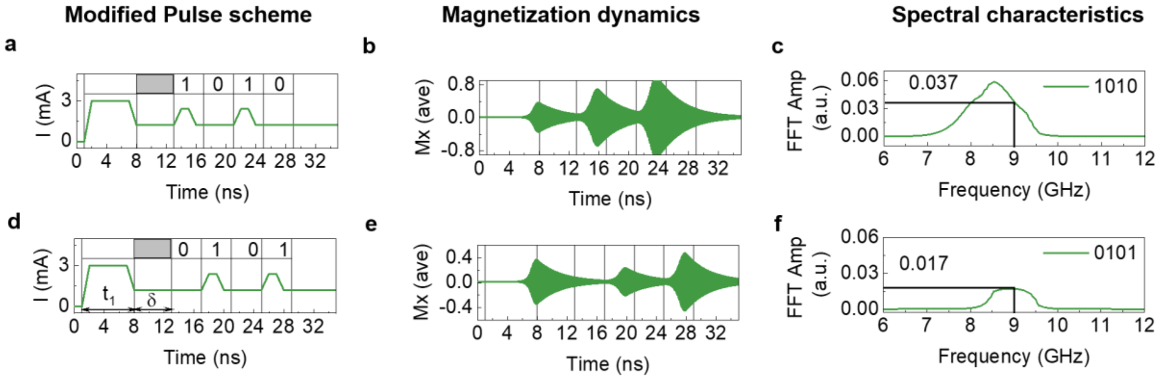


Fig. 4.15 a Modified pulse scheme with an excitatory pulse $I_e = 3.0$ mA of pulse width $t_1 = 7$ ns, for 4-binary digit pattern 1010 with pulse period (Δt) of 4 ns and pulse width (τ) of 3 ns. The pulse gap of δ = 5 ns between I_e and 4-binary digit patterns is shown with the grey box. **b.** Magnetization dynamics of Mx component and **c.** corresponding spectral characteristic FFT plot. **d – f.** Similar plots of input pattern, magnetization dynamics and FFT for 0101 pulse pattern. Guide line in FFT plots show the amplitude value at 9.0 GHz [1].

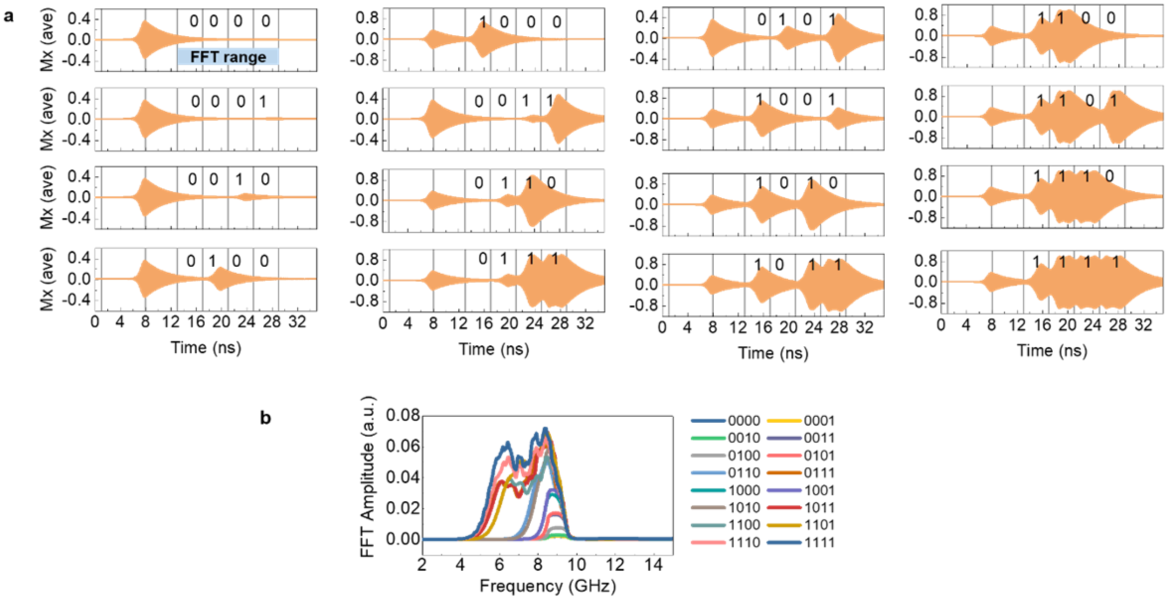


Fig. 4.16 a Magnetization dynamics of 16 combinations of 4-bit digit pattern separation on modified pulse scheme. **b** FFT amplitude spectra corresponding to the 16 different 4-bit input patterns [1].

Figure 4.17 displays the filtered FFT amplitude for the 16, $4 - \mathrm{bi(t)}$ input patterns using IP_1. Due to the distinction in the filtered amplitude values, a $4 - \mathrm{bi(t)}$ pattern can be quantized and represented as a 1-dimensional analog output. This filtered FFT amplitude encapsulates the most significant features extracted from the input patterns by the SHO layer. These features are not explicitly defined but are rather emergent from the dynamic behavior of the SHO in response to the inputs.

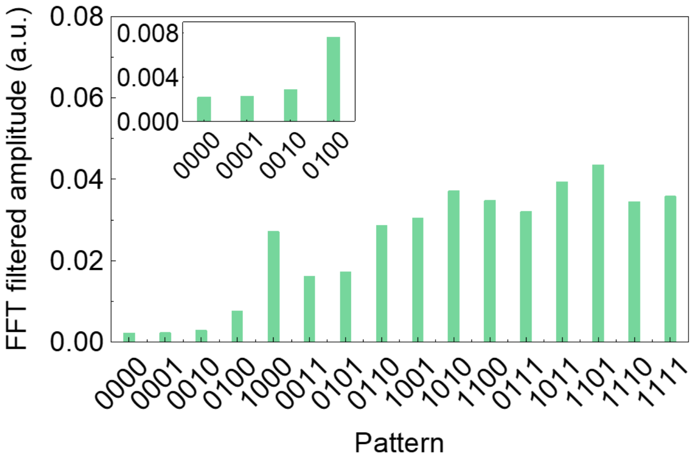
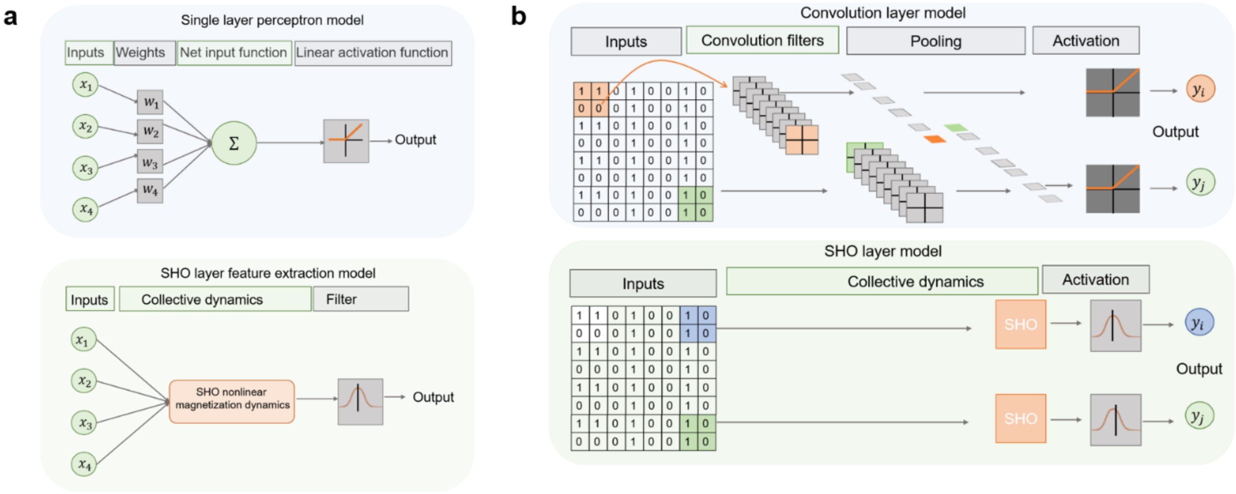


Fig. 4.17. Filtered amplitude at frequency of 9.0 GHz for all 16 4-binary digit inputs for the modified pulse scheme with input parameters [1].

4.5. Comparison of SHO with Perceptron and convolution neural network

The perceptron is a fundamental building block in classical machine learning. It takes an n-dimensional vector as input, applies a predefined function using trained weights, and produces an output. In contrast, the SHO doesn't rely on explicit weight training. Instead, it utilizes its inherent magnetization dynamics to transform input patterns into separable classes. This dynamic response effectively adjusts the "weights" and the activation function during operation is illustrated in the Fig.4.18a. One can view the SHO as a perceptron with built-in weights and activations. These weights and activations are not static but adapt in real-time as the magnetization dynamics evolve. Consequently, the SHO generates outputs that are linearly separable, akin to what a perceptron achieves with trained weights. While it might be tempting to draw an analogy between the SHO and a convolution layer, there are crucial differences. In a traditional CNN, each filter operates on the entire input space. This results in multiple convolution operations for a single set of binary inputs. The SHO, in contrast, directly maps specific inputs to specific outputs. It doesn't apply multiple filters to one input; instead, it has dynamic responses to individual input patterns. It doesn't engage in complex convolution operations like traditional CNNs. The comparison of SHO with CNN is illustrated in the Fig. 4.18b.



1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0

1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0
1	1	0	1	0	0	1	0
0	0	0	1	0	0	1	0

Figure 4.18 a Single layer perceptron model and working principle along with the comparison of SHO feature extraction method. **b**. comparison of SHO with the operating principle of convolution layer in the convolution neural network.

4.6. Effect of temperature in the classification of inputs

In the dynamics of Spin Hall Oscillators (SHO), temperature plays a significant role, as evidenced by experimental findings by Andrew Smith et al [23]. Their study on a nanowire-based SHO unveiled intriguing insights into the impact of temperature on SHO behavior. Notably, they observed a 7% variation in saturation magnetization and a frequency shift of approximately Δf ≈ -0.20 GHz when temperature effects were considered. Expanding upon this, we sought to examine the frequency response of the SHO while accounting for temperature, specifically at 300 Kelvin (K). Interestingly, we observed that while there was no significant shift in the central frequency, the spectra exhibited a noticeable widening, and the amplitude increased notably at 300 K. Figure 4.19 presents a snapshot of the magnetization dynamics at 300 K. Notably, the pulse amplitude of I_e was deliberately set lower than I_1 to induce the auto-oscillation mode driven by thermal effects.

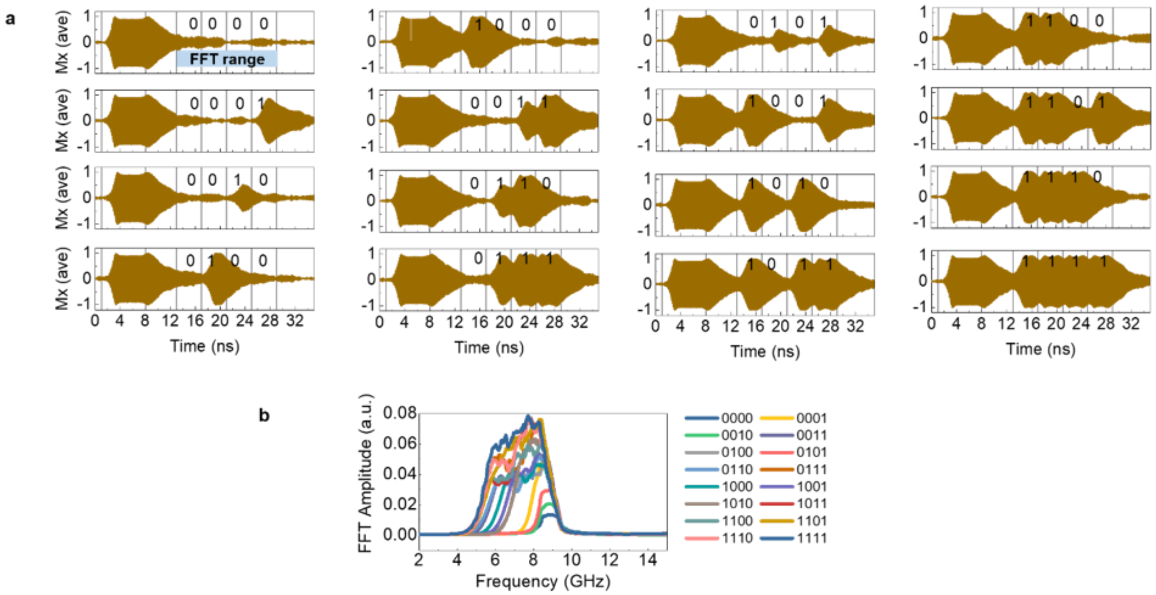


Fig. 4.19 Classification of inputs in the presence of thermal effects. **a** Magnetization dynamics corresponding to the 16 different 4-bit input patterns for the modified pulse scheme with input parameters $I_e = 3.0$ mA, $\delta = 5$ ns, I0 = 1.2 mA, $I_1 = 2.4$ mA, $\Delta t = 4$ ns, $\tau = 3.0$ ns at 300 K. **b** The FFT amplitude spectra corresponding to the 16 different 4-bit input patterns [1].

4.7. The MNIST handwritten digit image recognition

Finally, the SHO device with the modified pulse scheme is evaluated for the recognition of handwritten digits using the Modified National Institute of Standards and Technology (MNIST) handwritten database [24]. The database has 60,000 training images and 10,000 test images for the digits 0 to 9. Each image is 28 × 28 pixels in size and is displayed in grayscale, with pixel intensities ranging from 0 to 255. Figure 4.20a depicts the hypothetical model network, which contains an input layer, the SHO layer, and a classifier layer. The workflow of the network is assumed to have the following layer operations: the images are preprocessed in the input layer so that the original grayscale format is binarized with a threshold (pixel intensity > 125 = 1), where 1 stands for a white pixel and 0 for a black pixel. Following this, the images are divided into 4 pixel segments that move along rows and are then converted into input current pulses, creating a total of 196, $4 - \mathrm{bi(t)}$ input patterns. Each of the $4 - \mathrm{bi(t)}$ inputs is encoded as a pulse stream and fed to the SHO layer using the modified pulse scheme with $\mathrm{IP_1}$ parameters. As was discussed in the previous section, the filtered FFT amplitude is collected as the output of the SHO layer for $4 - \mathrm{bi(t)}$ pattern at 9.0 GHz. For the results shown here, custom functions were created in Matlab and Python programming languages to replace 4 pixel values with corresponding FFT amplitudes. The classifier layer has 10 nodes that are all fully connected and are used to categorize the 10 handwritten digits from the maximum entry.

The MNIST handwritten digit classification accuracy is evaluated as the ratio of the total number of correctly classified digits to the total number of digits. First, using simple linear regression the weight matrix was calculated using the training data and tested on the test set of 10,000 images. The accuracy obtained by this one-step calculation is 83.1%. The predicted vs. true digit confusion matrix, shown in Fig. 4.20b as a color map, displays the classification success for the $\mathrm{IP_1}$ modified pulse scheme. The digit 5 is the least successful at being classified, which lowers the success rate as a whole. In addition, the SHO network model was evaluated in a supervised learning process carried out in Python using the Tensorflow machine learning module [25]. We fixed the softmax activation function for the classifier layer and categorical cross entropy as loss function. The classification accuracy achieved with supervised learning with 20 epochs and 32 batch sizes was 86.6%. This indicates the overall classification accuracy can be improved with classifier layer training. However, our aim is to reduce the previous layer computations for feature extraction.

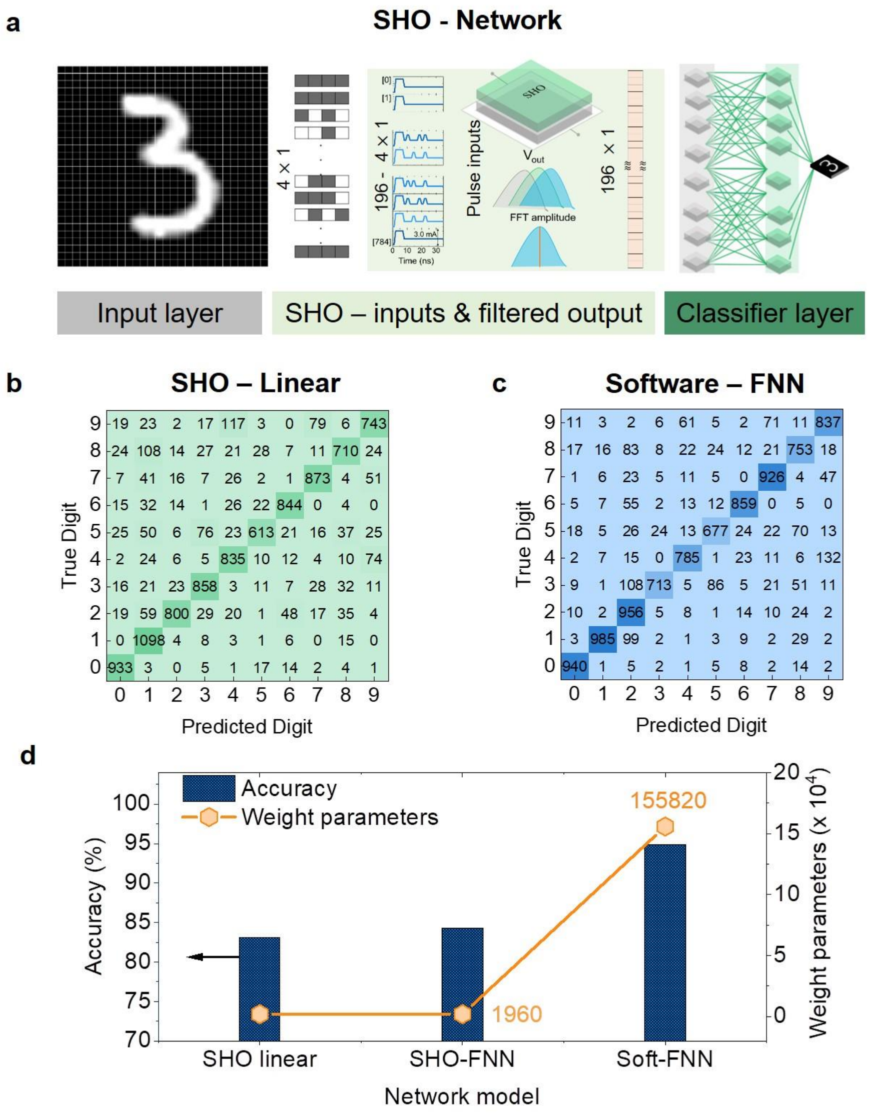


Fig. 4.20. a. Fully connected network model for classification of MNIST handwritten digit datasets with input layer, SHO layer and a classifier layer with 10 neurons for 10 digits. **b**. Color map shows the confusion matrix classification results for 10000 test images classified by SHO for a linear regression activation (SHO linear). **c**. Software implemented feed forward neural network model's, (software-FNN) classification results color map. **d.** Comparison of network model implemented with SHO and software as processing unit for filter operations along with classification accuracies (bar graph) and weight parameters (scatter-line plots) [1].

To illustrate the reduced computations, we performed network training comparing with a software FNN. The software-FNN model consisted of 784 input neurons for each input pixel, a fully connected middle layer of 196 neurons for feature extraction, and ten neurons (the classifier layer) for classification, respectively. We used the rectified linear activation function (ReLU) for the feature extraction layers and the softmax activation function for the classifier layer. It is clear that the software-FNN requires 196 neurons and associated computations to generate 196 features for the classifier layer, similar to the SHO layer outputs. The confusion matrix of the software-FNN, shown in Fig. 4.20c, achieved an accuracy of 93.1%. When compared to the SHO network, the software-FNN improves accuracy by 7.5%. However, the computational cost of calculating the weight matrix is high. Figure 4.20d shows the accuracy and weight parameters required for the classification for the linear SHO (single linear weight computation), SHO-FNN (supervised learning), and software-FNN, respectively. It can be seen that with the use of the SHO layer, the weight computations required for a software network can be significantly reduced from 155820 weight parameters to 1960 parameters. The ability of the SHO layer to directly infer distinct outputs for $4 - \mathrm{bi(t)}$ patterns reduces the computation required. Although there is a trade-off between recognition accuracy and inference computations for the MNIST handwritten digit classification due to various factors like binarazation of images, choice of classifier layer training methods, and loss functions, the SHO network is still favorable since it can achieve 83.1% with linear regression. Moreover, a sequence of 4 binary digits can be easily classified without any requirement of weight storage or computations, which can be beneficial for applications that need faster inference with reduced computations. Nevertheless, there is a huge opportunity to engage in the co-design of SHO as a dedicated feature mapping layer alongside already existing hardware and algorithms.

4.8. Summary

In summary, our comparison between the SHO and a perceptron layer highlights the SHO's capacity to streamline computations through nonlinear input transformations and built-in weight adjustments, emphasizing its significance in data-driven tasks. This feature extraction process is central to the success of the SHO-based classification system. It leverages the SHO's inherent oscillatory behavior and its sensitivity to input patterns to distill the most relevant information needed for accurate digit classification and image recognition.

References

[1] J. R. Mohan *et al.*, “Classification tasks using input driven nonlinear magnetization dynamics in spin Hall oscillator,” *Sci. Rep.*, vol. 13, no. 1, p. 7909, May 2023, doi: 10.1038/s41598-023-34849-

[2] U. Shashank *et al.*, “Enhanced Spin Hall Effect in S-Implanted Pt,” *Adv. Quantum Technol.*, vol. 4, no. 1, p. 2000112, Dec. 2020, doi: 10.1002/qute.202000112.

[3] J. E. Hirsch, “Spin Hall Effect,” *Phys. Rev. Lett.*, vol. 83, no. 9, pp. 1834–1837, Aug. 1999, doi: 10.1103/physrevlett.83.1834.

[4] J. Sinova, S. O. Valenzuela, J. Wunderlich, C. H. Back, and T. Jungwirth, “Spin Hall effects,” *Rev. Mod. Phys.*, vol. 87, no. 4, pp. 1213–1260, Oct. 2015, doi: 10.1103/RevModPhys.87.1213.

[5] J. Ryu, S. Lee, K.-J. Lee, and B.-G. Park, “Current-Induced Spin–Orbit Torques for Spintronic Applications,” *Adv. Mater.*, vol. 32, no. 35, p. 1907148, Sep. 2020, doi: 10.1002/adma.201907148.

[6] T. Nan *et al.*, “Comparison of spin-orbit torques and spin pumping across NiFe/Pt and NiFe/Cu/Pt interfaces,” *Phys. Rev. B*, vol. 91, no. 21, p. 214416, Jun. 2015, doi: 10.1103/PhysRevB.91.214416.

[7] V. E. Demidov, S. Urazhdin, E. R. J. Edwards, M. D. Stiles, R. D. McMichael, and S. O. Demokritov, “Control of Magnetic Fluctuations by Spin Current,” *Phys. Rev. Lett.*, vol. 107, no. 10, Sep. 2011, doi: 10.1103/physrevlett.107.107204.

[8] V. E. Demidov *et al.*, “Magnetic nano-oscillator driven by pure spin current,” *Nat. Mater.*, vol. 11, no. 12, pp. 1028–1031, Dec. 2012, doi: 10.1038/nmat3459.

[9] V. E. Demidov *et al.*, “Magnetization oscillations and waves driven by pure spin currents,” *Phys. Rep.*, vol. 673, pp. 1–31, Feb. 2017, doi: 10.1016/j.physrep.2017.01.001.

[10] S. I. Kiselev *et al.*, “Microwave oscillations of a nanomagnet driven by a spin-polarized current,” *Nature*, vol. 425, no. 6956, pp. 380–383, Sep. 2003, doi: 10.1038/nature01967.

[11] R. H. Liu, W. L. Lim, and S. Urazhdin, “Spectral Characteristics of the Microwave Emission by the Spin Hall Nano-Oscillator,” *Phys. Rev. Lett.*, vol. 110, no. 14, Apr. 2013, doi: 10.1103/physrevlett.110.147601.

[12] D. V Berkov and N. L. Gorn, “Magnetization precession due to a spin-polarized current in a thin nanoelement: Numerical simulation study,” *Phys. Rev. B*, vol. 72, no. 9, Sep. 2005, doi: 10.1103/physrevb.72.094401.

[13] J. Xiao, A. Zangwill, and M. D. Stiles, “Macrospin models of spin transfer dynamics,” *Phys. Rev. B*, vol. 72, no. 1, Jul. 2005, doi: 10.1103/physrevb.72.014446.

[14] A. N. Slavini and P. Kabos, “Approximate theory of microwave generation in a current-driven magnetic nanocontact magnetized in an arbitrary direction,” *IEEE Trans. Magn.*, vol. 41, no. 4, pp. 1264–1273, Apr. 2005, doi: 10.1109/TMAG.2005.845915.

[15] J.-V. Kim, “Spin-Torque Oscillators,” in *Solid State Physics*, Elsevier, 2012, pp. 217–294. doi: 10.1016/b978-0-12-397028-2.00004-7.

[16] A. Slavin and V. Tiberkevich, “Nonlinear Auto-Oscillator Theory of Microwave Generation by Spin-Polarized Current,” *IEEE Trans. Magn.*, vol. 45, no. 4, pp. 1875–1918, Apr. 2009, doi: 10.1109/TMAG.2008.2009935.

[17] G. Holzmann and H. Hauser, “Echo state networks with filter neurons and a delay&sum readout,” *Neural Networks*, vol. 23, no. 2, pp. 244–256, 2010, doi: 10.1016/j.neunet.2009.07.004.

[18] F. Wyffels, B. Schrauwen, D. Verstraeten, and D. Stroobandt, “Band-pass Reservoir Computing,” in *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, IEEE, Jun. 2008, pp. 3204–3209. doi: 10.1109/IJCNN.2008.4634252.

[19] A. Sebastian, M. Le Gallo, R. Khaddam-Aljameh, and E. Eleftheriou, “Memory devices and applications for in-memory computing,” *Nat. Nanotechnol.*, vol. 15, no. 7, pp. 529–544, Mar. 2020, doi: 10.1038/s41565-020-0655-z.

[20] I. Hubara, M. Courbariaux, D. Soudry, R. El-Yaniv, and Y. Bengio, “Quantized Neural Networks: Training Neural Networks with Low Precision Weights and Activations,” *J. Mach. Learn. Res.*, vol. 18, no. 187, pp. 1–30, 2018, [Online]. Available: http://jmlr.org/papers/v18/16-456.html

[21] P. Pham, J. A. Abraham, and J. Chung, “Training Multi-Bit Quantized and Binarized Networks with a Learnable Symmetric Quantizer,” *IEEE Access*, vol. 9, pp. 47194–47203, 2021, doi: 10.1109/ACCESS.2021.3067889.

[22] N. Akashi *et al.*, “Input-driven bifurcations and information processing capacity in spintronics reservoirs,” *Phys. Rev. Res.*, vol. 2, no. 4, Dec. 2020, doi: 10.1103/physrevresearch.2.043303.

[23] A. Smith *et al.*, “Dimensional crossover in spin Hall oscillators,” *Phys. Rev. B*, vol. 102, no. 5, p. 054422, Aug. 2020, doi: 10.1103/PhysRevB.102.054422.

[24] Y. LeCun, C. Cortes, and C. J. C. Burges, “The MNIST Database of Handwritten Digits,” *MNIST Database Handwrit. Digit.*, [Online]. Available: http://yann.lecun.com/exdb/mnist/.

[25] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo *et al.*, “TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems.” 2015. [Online]. Available: https://www.tensorflow.org/

Chapter 5

Spin Hall Oscillator for Reservoir Computing

In this chapter, we investigate the use of a single spin Hall oscillator for reservoir computing approach with the analysis of information processing capabilities and metrics.

5.1. Spin Hall oscillator for Reservoir computing

Reservoir computing (RC) is a computational framework that leverages a fixed, randomly connected dynamical system, known as the reservoir, for processing input data [1], [2]. The reservoir state update equation describes the reservoir state at discrete time T and depends on both the past reservoir state and the present input data as follows,

$$X(T+1) = f(\boldsymbol{W}_{in} u(T) + \boldsymbol{W}_{res} X(T)). \tag{5.1}$$

Here, f represents the activation function, and the matrices $\boldsymbol{W}_{in}$ and $\boldsymbol{W}_{res}$ contain fixed input connection weights and reservoir weights (random internal state connections), respectively. As is common in RC, only the output weights are computed to obtain the output $Y(T)$ which is the weighted sum of reservoir states ($\boldsymbol{W}_{out}$), defined as,

$$y(T) = \boldsymbol{W}_{out} X(T). \tag{5.2}$$

A fundamental feature of RC is that the reservoir's internal dynamics remain fixed during training. Instead, a linear readout layer is trained to map the high-dimensional state space of the reservoir to the desired output. Typically, in physical RC implementations with a single dynamical system, a time-multiplexed input method is employed to diversify the reservoir output. This involves applying a mask element (signal) of fixed length to each input data point, and the dynamics are stored for the duration of the mask as virtual nodes. This approach transforms the input into a higher-dimensional output through time-multiplexing. However, embedded systems and IoT devices often have limited processing capacity and signal processing units, in addition to constraints on available training and testing data. Therefore, it is crucial to assess the applicability of RC in real-world scenarios with limited memory and processing capabilities.

Spintronics oscillator-based reservoir computing (STO-RC) has garnered substantial interest due to its potential for efficient, low-power, and high-performance computing. Spin torque oscillators, including spin transfer torque (STT) and spin orbit torque (SOT) oscillators, have demonstrated remarkable capabilities compared to traditional echo state networks (ESNs) [3]–[6]. In such implementations, the input signal is encoded using magnetic pulses or current pulses, and the phase or frequency of the oscillator's output signal serves as the high-dimensional reservoir for processing the input data. The performance of a reservoir is assessed using various benchmark tasks, such as memory capacity (MC), nonlinear parity check (PC), and nonlinear auto-regressive moving average (NARMA) tasks.

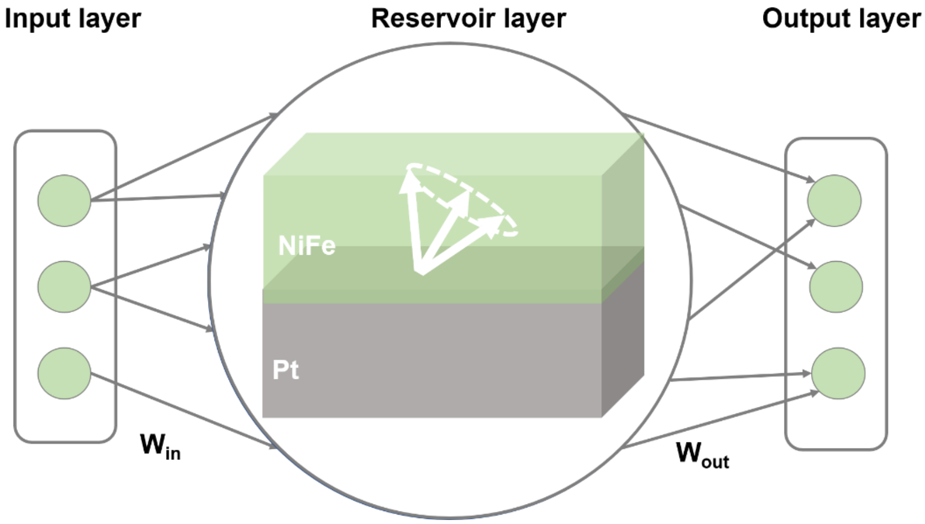


Fig. 5.1 Reservoir computing model with a single spin Hall oscillator.

5.1.1. Input driven dynamics

Figure 5.1 illustrates the schematic model of the reservoir computing scheme, comprising an input layer, the reservoir, and the output layer. In the input layer, a binary input data stream with n elements, denoted as $u_{in} = \{u_1, u_2, \ldots, u_n\}$, is coded as input current pulses. Each input data point $u_i(T)$ is fed into the Spin Hall Oscillator (SHO) for a fixed pulse duration T, encompassing pulse width time t_{pw}, pulse rise time t_r, pulse fall time t_f, and relaxation time t_{rx}, as visually depicted in Fig. 5.2.

The reservoir component comprises a single SHO oscillator, as previously modelled the chapter 4. A charge current is applied along the X direction, with a constant magnetic field along the Y direction with a strength of $\mu_0 \mathrm{H}_{ext} = 100$ mT. Other simulation parameters remain consistent with those listed in Chapter 4, Table 1. The output of the SHO reservoir consists of time-dependent oscillating amplitudes extracted from the envelope signal $S(t)$ of the Mx magnetization component, as shown in Fig. 5.2. This assumption is based on a voltage detection technique relying on the anisotropic magnetoresistance (AMR) property of the ferromagnet. To evaluate reservoir computational tasks, the continuous signal $S(t)$ is discretized into N virtual nodes with a constant time separation between the nodes, effectively representing the reservoir state X. This relationship maps an input binary data u_i to a reservoir state vector X_i with N output nodes.

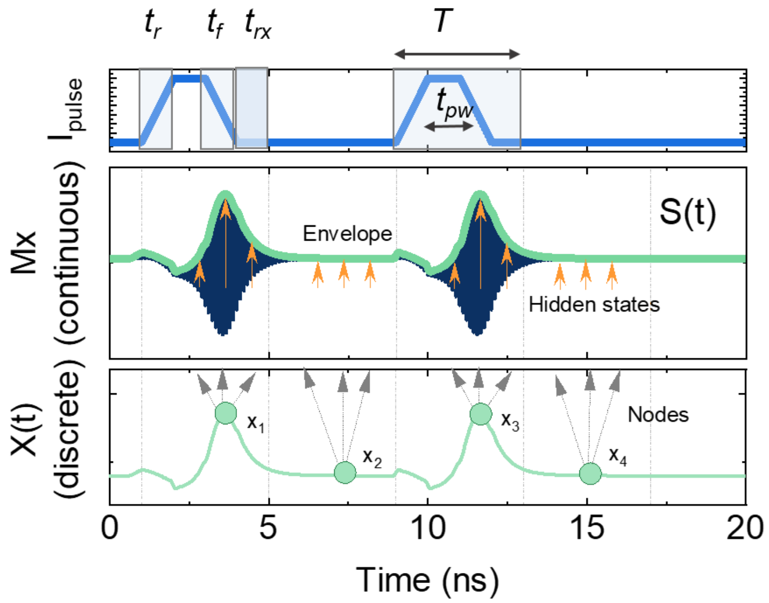


Fig. 5.2 Reservoir binary inputs (top) with pulse parameters, magnetization dynamics Mx with upper envelope S(t) (middle), and reservoir nodes taken from S(t) to represent the reservoir states.

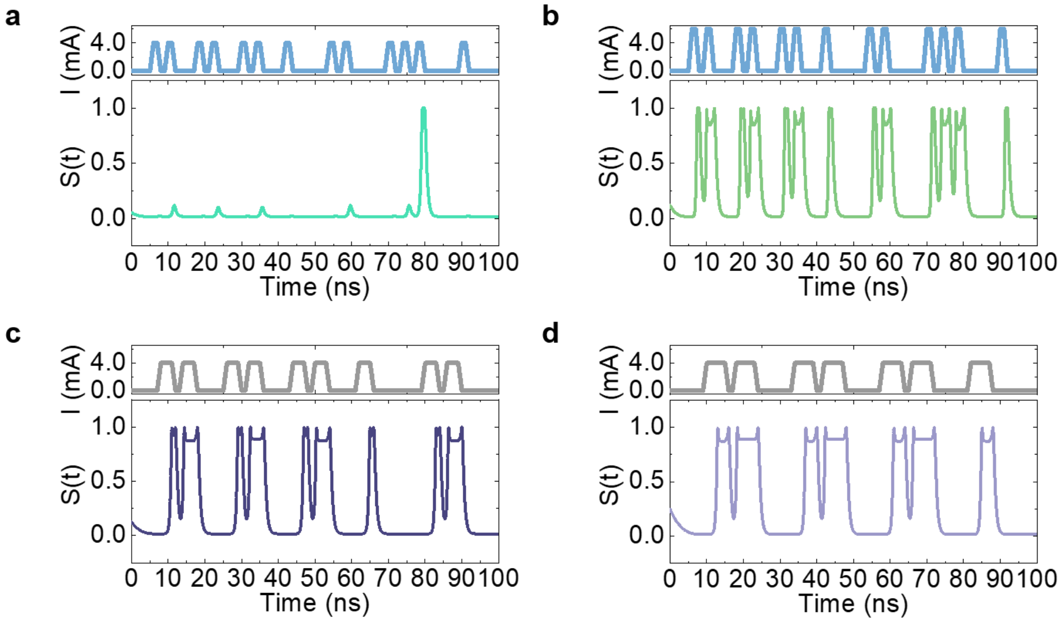


Fig. 5.3. Multiple current pulses with (a) I_1 = 4.0 mA; t_{pw} = 1 ns, (b) I = 5.0 mA; t_{pw} = 1 ns, (c) I = 4.0 mA; t_{pw} = 3 ns, (d) I = 4.0 mA; t_{pw} = 5 ns given as input to the spin Hall oscillator and the corresponding upper envelopes $S(t)$ induced by them.

The input current pulses and the corresponding output Mx oscillation's $S(t)$ are shown for different sets of I and t_{pw} in Figs. 5.3a-d. We find that for a particular value of t_{pw}, as I increase, the value of the corresponding $S(t)$ also increases for the same input bit as shown in Figs. 5.3a and be) and 3(f). At lower values of I and t_{pw} even when multiple finite pulses are consecutively given as input, the output S(t) remains in the transient state with increasing amplitude with each input pulse. As I or t_{pw} is increased, limit cycle oscillations are observed after multiple consecutive input pulses. This results in a combination of transient and limit cycle oscillations in the output $S(t)$. When I or t_{pw} is sufficiently high, all oscillations are in the limit cycle. Having seen the variation in the output dynamics of the SHO with change in amplitude and pulse width of the multiple input current pulses, we proceed to utilize the output S(t) of the SHO for performing various reservoir computing benchmark tasks.

5.2. Reservoir computing Metrics

5.2.1. Memory Capacity

MC is a crucial metric that quantifies how much information about past inputs is retained in the present reservoir state [8] – [10]. It assesses the reservoir's ability to maintain information over time, a vital attribute for various computing tasks. MC is calculated by measuring the correlation between the current reservoir state N at time t and past inputs u_{n-k} at time $(t-k)$. The goal is to reconstruct a sequence of input data (the test signal) from a trained input sequence and the reservoir states using a linear estimator. We proceed to investigate the magnetization dynamics when a stream of 1270 bits are fed as input current pulses to the SHO. Bit 0 is defined as the input current pulse with $I=0$. Bit 1 is the input current pulse having a finite amplitude I. For the MC task, a delay d is defined such that the k^{th} input data's reservoir states N_k are related to the $(k-d)^{th}$ input data as illustrated in Fig.5.4. This relationship can be expressed as,

$$S_{k,i} = S\left[(k-1)T + i\frac{T}{N}\right]. \tag{5.3}$$

Here, i denotes the reservoir state nodes index, ranging from 0 to N and k denotes the input index, ranging from 0 to u_n. The input and output states relationship can be modelled using the reservoir weight w_d satisfying the following condition,

$$u_{k-d} = \sum_{i=1}^{N+1} S_{k,i}\, w_{d,i}, \tag{5.4}$$

where u_{k-d} is the $(k-d)^{th}$ input data, and $N+1$ term represents an added bias term, fixed at unity. The weight $w_{d,i}$ can be determined using the Moore-Penrose pseudo-inverse method, introducing a regularization factor λ.

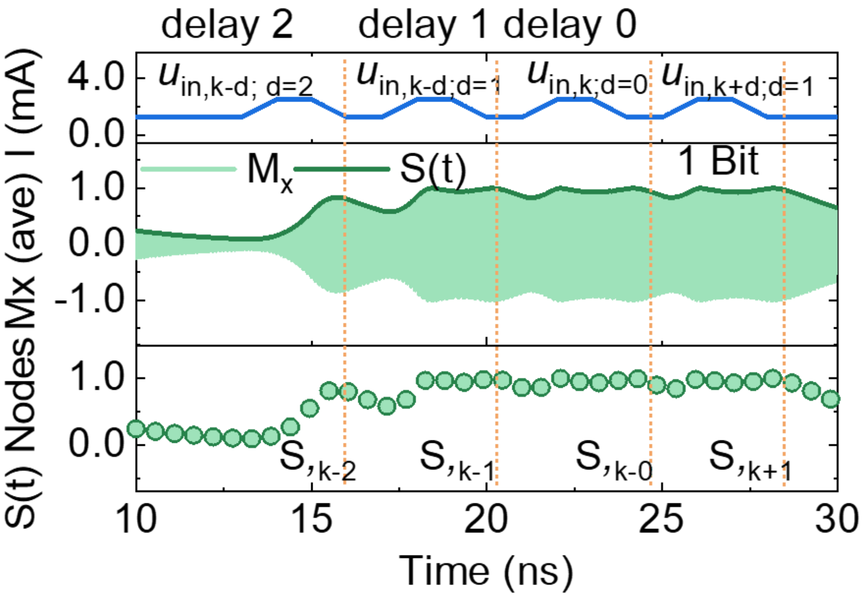


Fig. 5.4. Delayed inputs u_{k-d} represents the $(k-d)^{th}$ previous input and the current reservoir state S_k.

For a finite value of d in Eqn. 5.3, the input bit where $S_{k,i}$ is related to the input bit u_{k-d}, which is d steps in the past. Equation 5.4 can be written in matrix form as given below,

$$WS = B. \tag{5.5}$$

Solving for W using the Moore-Penrose pseudo inverse method [11], we get,

$$W = (S^T S + \lambda I)^{-1} S^T B, \tag{5.5}$$

where λ is the regularization constant (fixed at 1×10^{-5}). After optimization of W using the training set of input bits B and output envelope S, the weight matrix W and the testing output envelope S' are used to predict the testing set of input bits denoted by B',

$$B' = WS'. \tag{5.6}$$

The quality of prediction is quantified by the correlation coefficient $C(d)$ defined as,

$$C(d) = \frac{\text{cov}\left[\left(B'_{rep}, B'\right)\right]^2}{\text{var}\left(B'_{rep}\right), \text{var}(B')}. \tag{5.7}$$

with the functions cov andvar indicating the covariance and variance of the associated quantities. Memory capacity MC is defined as the sum of $C^2(d)$ over all range of delays d, given by,

$$MC = \sum_{d=1}^{d_{max}} [C^2(d)]. \tag{5.8}$$

Figure 5.5a shows the target and predicted data for the first few testing input bits. We see that, as the delay d increases, the difference between the target and predicted data also increases. This indicates that, as we move further back in the past, the SHO output retains lesser information regarding the input. We calculate MC for different values of t_{pw} and I. The MC values obtained for the various combinations of t_{pw} and I are depicted as a color map in Fig. 5.5b. As shown in Fig. 5.5b, when t_{pw} is fixed and I is increased, we observe a difference in the magnetization dynamics that contributes to the MC. For lower values of I, all output oscillations remain in the transient state and hence corresponding MC values are also low. As I is increased, limit cycle oscillations too start to contribute for inputs of continuous input bit 1 pulses. At sufficiently high I, all output oscillations are limit cycle oscillations and hence the maximum possible value of MC is obtained. A similar trend is seen for a fixed I and increase in t_{pw}. MC is found to saturate in the range of 4.5 to 5.0. Any further increase in I or t_{pw} does not increase MC. The reason for the saturation of MC in this range is due to the dynamics approaching the limit cycle of oscillations for all pulses. The input parameters at which limit cycle oscillations are reached are determined by the inherent time scales of the oscillations and their relaxation for the SHO, which is of the order of few ns.

The highest value of MC = 5.0 is found for I = 5.5 mA and t_{pw} = 4 ns. For higher values of I or t_{pw}, the MC is found to be lower. We can infer that when more out of plane limit cycle oscillations contribute to the output of the SHO reservoir, there is a decrease in the memory of the reservoir. This decrease in MC can be attributed to the magnetization precessing about an axis different from the in plane easy axis of the ferromagnet. Higher values of MC are seen when the oscillation dynamics includes transient state oscillations and in plane limit cycle oscillations.

5.2.2. Parity Check

Parity check (PC) is a crucial metric in Reservoir Computing (RC) that demands both fading memory and nonlinearity. It determines the target for each time step by assessing the parity of the sum of consecutive inputs up to a certain past delay. In the context of PC within RC, the target function alters from $u(k-d)$, thereby modifying Eqn. 5.4 as shown below:

$$\sum_{i=1}^{N+1} w_{d,i} S_{k,i} = [\sum_{j-d}^{d-2} u(k-j)](\text{mod } 2) \tag{5.9}$$

All other calculations remain same as that for MC. We compare the predicted output B'_{rep} with the theoretical output B', in terms of the Normalized Mean Square Error (NMSE), given by,

$$NMSE = \frac{\sum_i \left(B'_{rep} - B'\right)^2}{\sum_i \left(B'_i\right)^2}. \tag{5.10}$$

The color map in Fig. 5.6 gives the NMSE of the three-bit parity task for various combinations of t_{pw} and I. We observe similar trends as observed for MC. When fixing t_{pw} and increasing I, and vice-versa, the NMSE is found to decrease. From Fig. 5.5(b) and Fig.5.6, we see that the least NMSE in three-bit PC is found in the same region which corresponds to highest MC. Thus, the high MC possessed by the reservoir is found to improve its performance in the three-bit parity task.

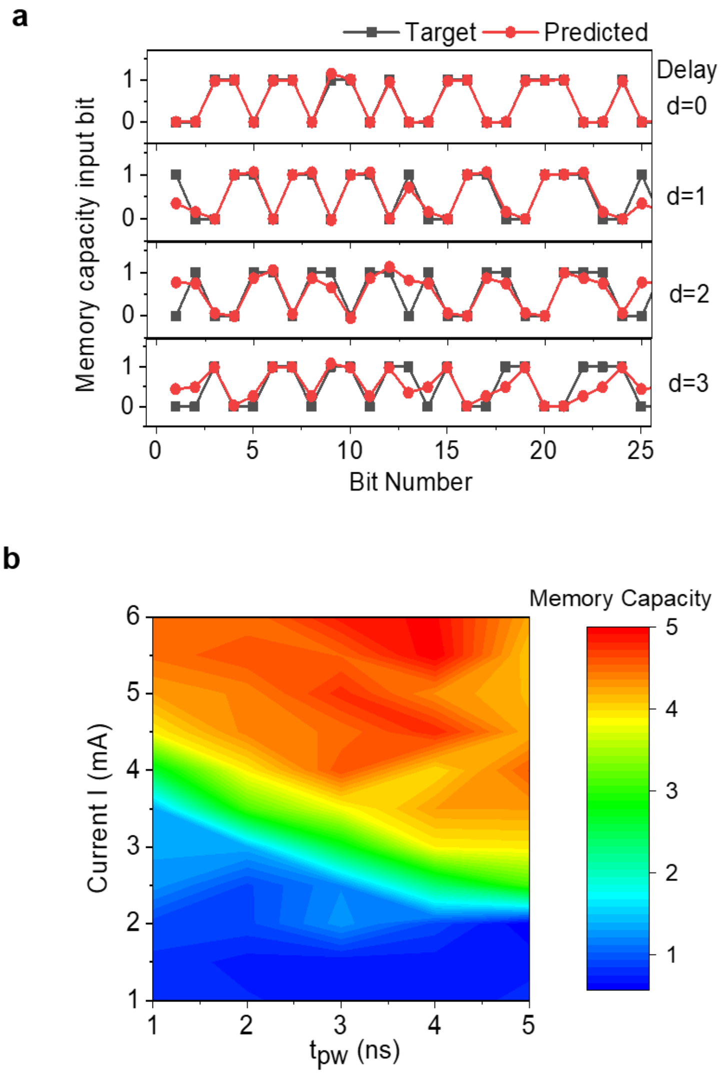


Fig. 5.5 a Target and predicted inputs of the memory capacity task, for the first 25 bits out of the 420 bits of training data (N=10, regularisation constant λ=1e-5, I = 5.5 mA, t_{pw}=1 ns). **b**. Color map showing the memory capacity as a function of input amplitude I and pulse width t_{pw} [7].

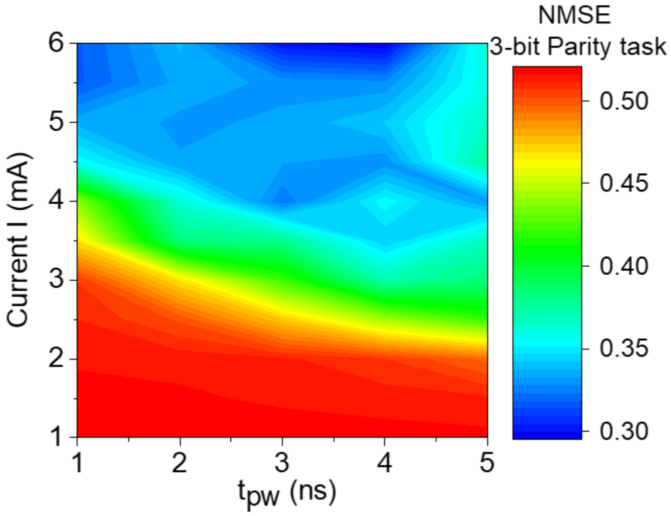


Fig. 5.6 Color map showing the NMSE of the three-bit parity task [7].

5.3. Prediction capability - Nonlinear Auto Regressive Moving Average Task

The Nonlinear Auto Regressive Moving Average task (NARMA) is a significant benchmark in the realm of time-series prediction and dynamic system modelling [12]. In this study, we tackled the NARMA2 task, indicating a task order of 2. In simpler terms, NARMA2 entails predicting a time-series by considering not only the current input but also the previous two time steps. This task poses a considerable challenge due to its inherent nonlinearities, memory requirements, and complex dependencies on past inputs. Additionally, it necessitates modelling intricate temporal dependencies where accurate predictions rely on capturing how past inputs influence present and future outputs.

In the NARMA2 task, we assume a nonlinear dynamical system generating an output, y_k, based on a second-order nonlinear function as expressed as,

$$y_k = 0.4y_{k-1} + 0.4y_{k-1}y_{k-2} + 0.6u_k^3 + 0.1 \tag{5.11}$$

where u_k is the corresponding input in the range [0, 0.5]. Thus, the output at any particular time depends on the corresponding input as well as the two previous outputs. For a reservoir to successfully replicate the second order nonlinear dynamical output given above in Eqn. 5.11, it needs to have sufficient memory to retain information at least up to the two previous outputs. The same 1270 bits used in the MC and PC tasks are fed as input to the SHO. We train our SHO reservoir to map the above function in in Eqn. 5.10, and evaluate the accuracy of the predicted output with respect to the theoretical output P_k, again in terms of the NMSE, given by,

$$NMSE = \frac{\sum_i (y_i - P_i)^2}{\sum_i y_i^2}. \tag{5.11}$$

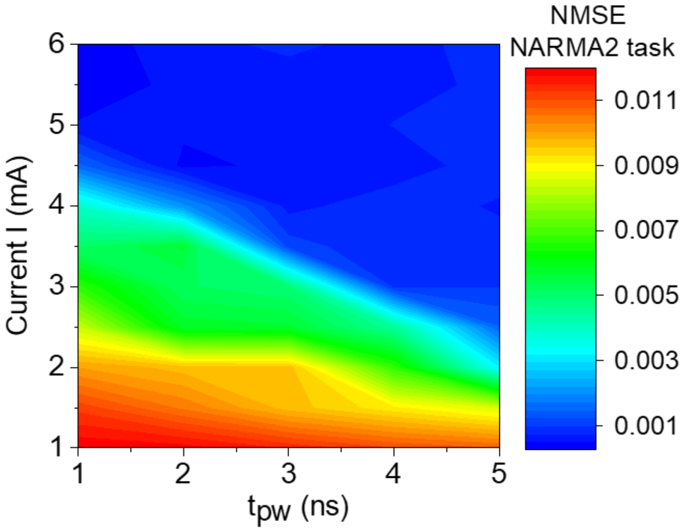


Fig. 5.7 Color map showing the NMSE of the NARMA2 task [7].

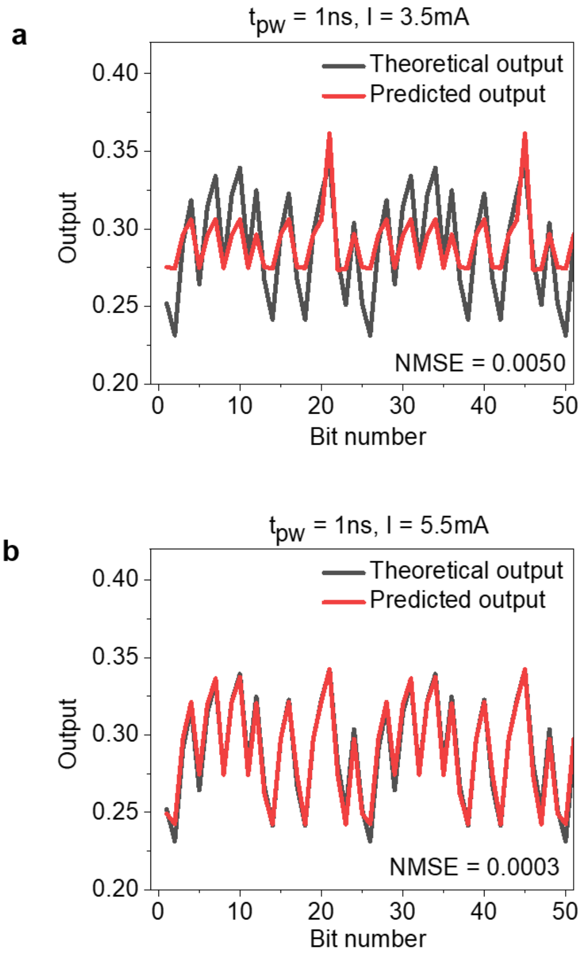


Fig. 5.8 a. Theoretical and predicted outputs of the NARMA2 task for I = 3.5 mA, t_{pw} = 1 ns and **b** I = 5.5 mA, t_{pw} = 1 ns [7].

With increasing I, there is better agreement between theoretical and predicted outputs seen by the decrease in NMSE value. NMSE for the NARMA2 task as a function of I and t_{pw} are shown in the Fig. 5.7. The theoretical and predicted outputs for the NARMA2 task, with $t_{pw} = 1$ ns, I = 3.5 mA and t_{pw} = 1 ns, I = 5.5 mA are shown in Fig. 5.8a and Fig. 5.8b, respectively. For a fixed value of I, as t_{pw} increases, the NMSE decreases. When t_{pw} is fixed, and I increased, again a decrease in NMSE is seen. Correlating the different color maps from Figs. 5.5b, 5.6 and 5.7, we can infer that the decrease in NMSE of the NARMA2 task is due to the increase in the memory capacity of the SHO reservoir. As discussed previously, this increased MC corresponds to magnetization dynamics which includes both transient and in plane limit cycle oscillations, but not to out of plane limit cycle oscillations which decreases MC value. Thus, we are able to confirm that by enhancing its memory capacity, our SHO reservoir is able to improve its performance in the time series prediction task.

5.4. Summary

This numerical investigation shed light into the impact of a single Spin Hall Oscillator's (SHO) magnetization dynamics on its information processing capacity when employed within a reservoir computing framework. By actively controlling input parameters such as input current pulse amplitude and pulse width, the study revealed a substantial improvement in memory capacity (MC). The MC eventually reached saturation within the range of 4.5 to 5.0, a phenomenon linked to the transition of magnetic oscillations from transient dynamics to in-plane limit cycle oscillations. Notably, oscillations associated with out-of-plane limit cycles led to a reduction in MC with increased current pulse amplitude or pulse width. Furthermore, practical Reservoir Computing tasks, including three-bit parity checks and Nonlinear Auto Regressive Moving Average (NARMA2) tasks, were evaluated. The results underscored that the input parameters corresponding to higher MC also yielded lower errors in these tasks, affirming the critical relationship between MC and task performance.

References

[1] K. Nakajima, “Physical reservoir computing—an introductory perspective,” *Jpn. J. Appl. Phys.*, vol. 59, no. 6, p. 060501, Jun. 2020, doi: 10.35848/1347-4065/ab8d4f.

[2] K. Nakajima and I. Fischer, *Reservoir Computing: Theory, Physical Implementations, and Applications*. in Natural Computing Series. Singapore: Springer Singapore, 2021. doi: 10.1007/978-981-13-1687-6.

[3] D. Marković *et al.*, “Reservoir computing with the frequency, phase, and amplitude of spin-torque nano-oscillators,” *Appl. Phys. Lett.*, vol. 114, no. 1, pp. 1–5, 2019, doi: 10.1063/1.5079305.

[4] S. Tsunegi *et al.*, “Physical reservoir computing based on spin torque oscillator with forced synchronization,” *Appl. Phys. Lett.*, vol. 114, no. 16, p. 164101, Apr. 2019, doi: 10.1063/1.5081797.

[5] T. Furuta *et al.*, “Macromagnetic Simulation for Reservoir Computing Utilizing Spin Dynamics in Magnetic Tunnel Junctions,” *Phys. Rev. Appl.*, vol. 10, no. 3, p. 1, 2018, doi: 10.1103/PhysRevApplied.10.034063.

[6] N. Akashi *et al.*, “Input-driven bifurcations and information processing capacity in spintronics reservoirs,” *Phys. Rev. Res.*, vol. 2, no. 4, p. 043303, Dec. 2020, doi: 10.1103/PhysRevResearch.2.043303.

[7] A. J. Mathew *et al.*, “Evaluation of Memory Capacity and Time Series Prediction Using a Spin Hall Oscillator as Reservoir,” *IEEE Trans. Magn.*, vol. 59, no. 6, pp. 1–5, Jun. 2023, doi: 10.1109/TMAG.2023.3266517.

[8] H. Jaeger and H. Haas, “Harnessing Nonlinearity: Predicting Chaotic Systems and Saving Energy in Wireless Communication,” *Science (80-.).*, vol. 304, no. 5667, pp. 78–80, Apr. 2004, doi: 10.1126/science.1091277.

[9] K. Fujii and K. Nakajima, “Harnessing disordered-ensemble quantum dynamics for machine learning,” *Phys. Rev. Appl.*, vol. 8, no. 2, 2017, doi: 10.1103/PhysRevApplied.8.024030.

[10] D. Verstraeten, B. Schrauwen, M. D’Haene, and D. Stroobandt, “An experimental unification of reservoir computing methods,” *Neural Networks*, vol. 20, no. 3, pp. 391–403, 2007, doi: 10.1016/j.neunet.2007.04.003.

[11] R. Penrose, “A generalized inverse for matrices,” *Math. Proc. Cambridge Philos. Soc.*, vol. 51, no. 3, pp. 406–413, 1955, doi: 10.1017/S0305004100030401.

[12] H. Jaeger, “The" echo state" approach to analysing and training recurrent neural networks-with an erratum note’,” *Bonn, Ger. Ger. Natl. Res. Cent. Inf. Technol. GMD Tech. Rep.*, vol. 148, 2001.

Chapter 6

Enhancing information processing capability of SHOs – magnetic dipolar coupling approach

In this chapter, we explore the dipolar coupling of spin Hall oscillators for enhancing the information processing for reservoir computing. We investigate the effect of inter-distance between the SHOs in array structure.

6.1. Dipolar coupled spin Hall oscillator arrays

Spin Hall oscillators (SHOs) are inherently nonlinear oscillators, and their nonlinearity can be further amplified by coupling them together. Coupled spintronics oscillators find a wide range of applications, including boosting oscillator output power, logic devices, and neuromorphic computing. The coupling between SHOs can be established through various methods such as electrical coupling, spinwave coupling, and magnetic dipolar coupling [1]–[4]. Magnetic dipolar coupling naturally occurs when an array of Spin Torque Oscillators (STOs) is closely integrated. In this setup, individual STOs can interact through synchronized oscillations, either through phase or frequency coupling. One significant advantage of dipolar coupling is that it's an intrinsic property dependent on the distance between two oscillators, eliminating the need for additional external sources to facilitate synchronization.

Furthermore, the dynamics of magnetization with synchronization, including phase-locking, can be controlled and adjusted using external parameters like applied magnetic fields and electrical currents. Recent experiments in coupled spintronic oscillator design have demonstrated their high efficiency compared to conventional echo-state networks (ESN) in reservoir computing tasks [5]. This showcases their potential and practicality, particularly in edge computing scenarios.

6.1.1. Simulation model

In order to understand the nature of various oscillatory dynamics of the dipolar coupled SHOs (dSHO) with the influence of the separation distance, i.e., gap between each oscillator (g), and the number of oscillators, we utilized mumax3 simulation framework [6]. Mumax3 offers the advantage of recording the magnetization dynamics of individual oscillators alongside the coupled dynamics. A model dSHO array of $N \times N$ SHO $(N = 2)$ is depicted in Fig. 6.1. The square dimension of each oscillator is 40×40 nm^2

and a thickness of 1 nm. We selected this size to ensure coherent and uniform magnetization oscillation across the structure. The magnetic layer is NiFe, consistent with previous sections, and the simulation parameters are detailed in Table 1 of Chapter 4. The Landau-Lifshitz-Gilbert equation is solved including the spin torque term modified for SOT conditions. Inputs are treated as charge current densities j_c with step pulse of 1 ns and the applied field is oriented at $\phi = 45°$ with a strength of $\mu_0 H_{ext} = 10$ mT. It's important to note that this applied field is lower than in previous simulations and analysis. For NiFe, a strong magnetic field would align all the SHOs in the same direction, causing them to oscillate uniformly without interactions. Therefore, a smaller field was chosen for the subsequent simulations. For the computational tasks, we utilized the collective magnetization dynamics incorporating all SHO's output.

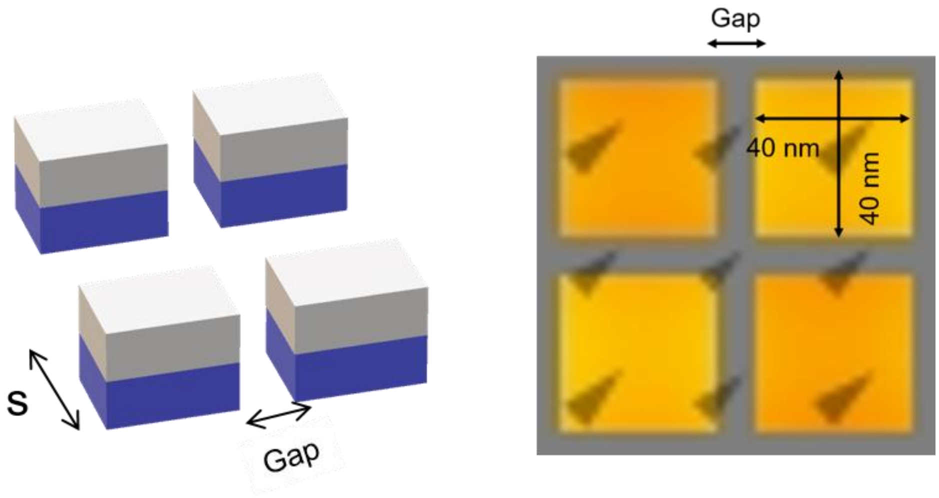


Fig. 6.1 Simulation model of dipolar coupled SHO array.

6.2. Effect of oscillator array size and impact of memory capacity

Figures 6.2a and 6.2b visually represent the magnetization states of the SHOs during pulse excitation, providing a clear illustration of the nonlinear interactions among these oscillators. These interactions become prominent when the oscillation trajectories reach larger angles, allowing them to influence each other. In our pursuit of evaluating memory capacity (MC), we follow the same procedure outlined in Chapter 5, using 1270 bits and conducting the same training and testing processes. Figure 6.3 presents a color map depicting MC as a function of j_c (current density) for various array structures indicated by the number of SHOs. Different regions with varying MC values are labeled as "a," "b," "c," and "d." We'll focus into these regions and their oscillatory characteristics further in Fig. 6.4.

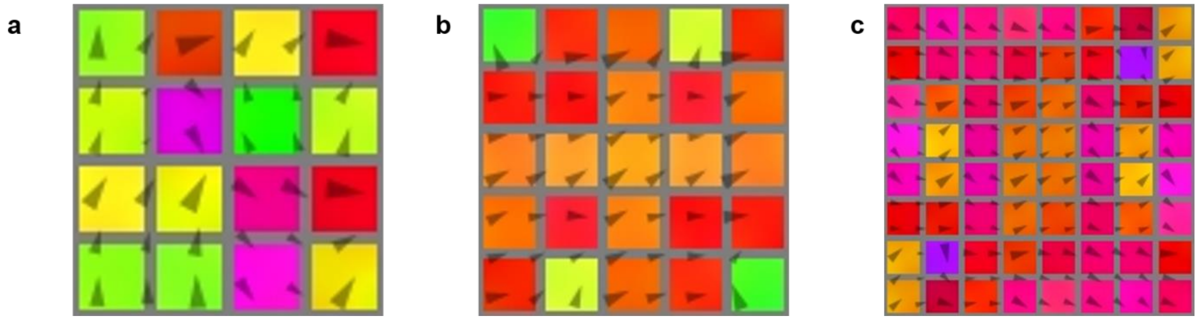


Fig. 6.2 Nonlinear dipolar interaction among SHOs during pulse excitation for **a** 4 × 4, **b** 5 × 5, **c** 8 × 8.

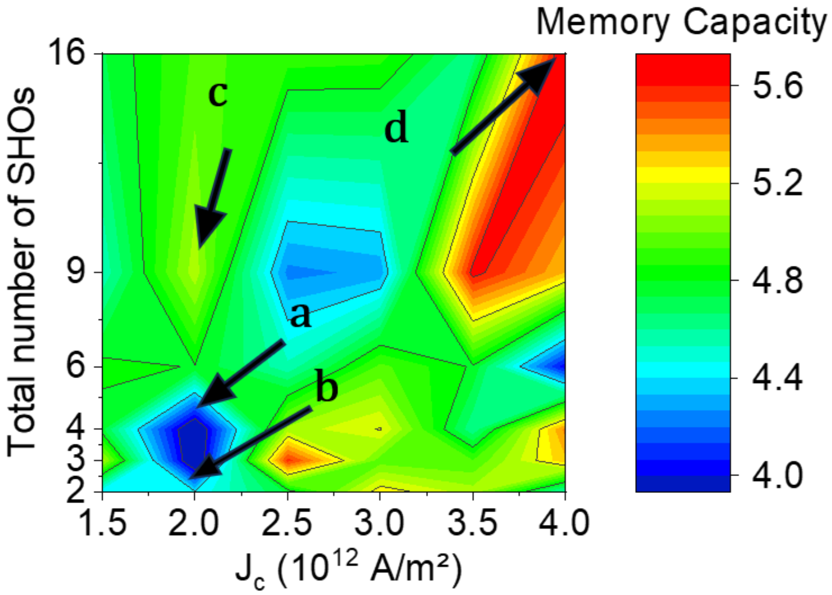


Fig. 6.3 Color map of memory capacity for various array structures with varying number of SHOs. The regions with different MC are marked with letters a, b, c, d from least to higher MC values

Now, focusing on Fig 6.3, the region labeled "a" represents the lowest MC, measuring at 3.83. This corresponds to a 2×2 array with $j_c = 2.0 \times 10^{12}$ A/m². Region "b" consists of a 2×1 structure with a slightly higher MC of 4.41 under the same j_c conditions. To gain insight into this behavior, we examine the magnetization dynamics, as shown in Figs. 6.4a and 6.4b for 2×2 and 2×1 array, respectively. Given that we derive the reservoir states from the m_x components of the inputs, we can identify the reason for the reduced MC in the case of the 2×2 structure. Analyzing the dynamics during three consecutive bit-1 pulses between 85 ns to 90 ns in Fig 6.4a, we notice that each bit-1 exhibits inconsistent dynamics. This suggests an unstable limit cycle oscillation that doesn't repeat its dynamics. In contrast, the 2×1 configuration displayed in Fig 6.4b exhibits repetitive dynamics (Echo state) for each bit-1 within the same time frame of 85 to 90 ns. To gain deeper insights, we observe the oscillation trajectories illustrated in Fig. 6.5a and 6.5b. The 2×2 array in Fig 6.5a demonstrates non-repetitive dynamics, whereas the 2×1 array, after the initial transition, exhibits a steady-state trajectory, as depicted in Fig 6.5b. Similarly, exploring regions "c" (3×3 structure) and "d" (4×4 structure) reveals higher MC values of 5.11 and 5.73, respectively. As we observe the magnetization dynamics shown in Figs 6.4c and 6.4d for 3×3 and 4×4 structures ($j_c = 4.0 \times 10^{12}$ A/m²), respectively, we can establish a correlation between increased MC and the magnetization dynamics.

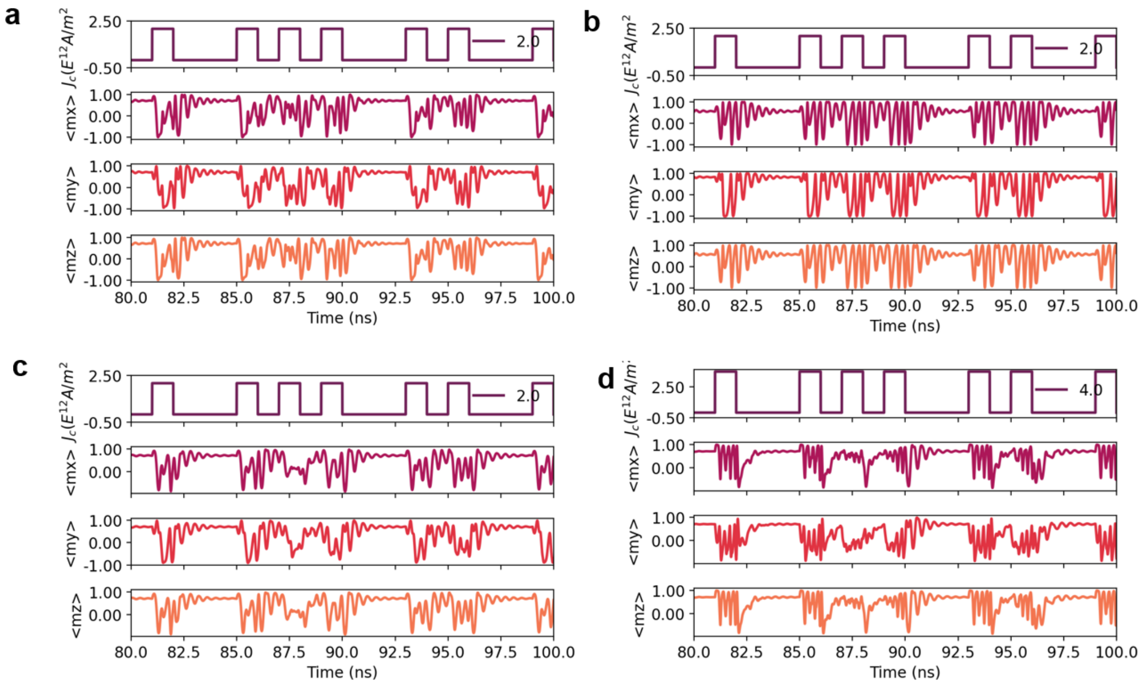


Fig. 6.4 a-d Magnetization dynamics of selected range of time frame for the regions a, b, c, d marked in the Fig. 6.3.

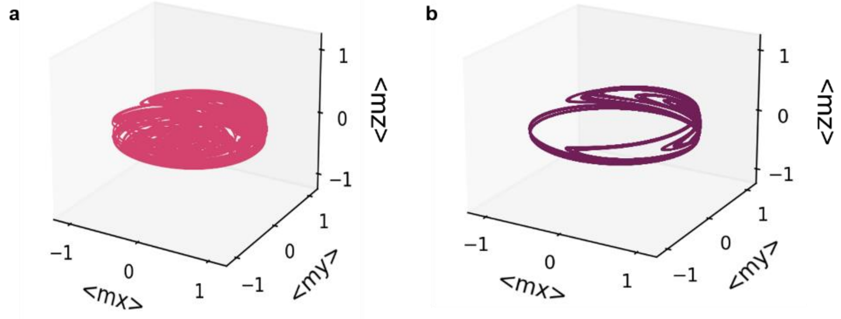


Fig. 6.5 a Oscillation trajectory of 2 × 2 array structure for continuous pulses. **b** Oscillation trajectory of 2 × 1 array structure for continuous pulses.

6.2.1. Enhancement of Memory capacity

To explore deeper into this correlation, we conducted an analysis of the correlation coefficient plots, denoted as $C^2\ (d)$, as a function of delay, represented by 'd'. In Figure 6.6a, you can observe these $C^2(d)$ plots for the four regions, which correspond to reservoir nodes equal to 10, mirroring the scenario of a single SHO as discussed in Chapter 5. In region "a" (represented by purple circles), we notice random fluctuations in $C^2\ (d)$ values across different delays. This pattern aligns with our earlier observations of time-dependent magnetization dynamics, which displayed non-repetitive behavior. These findings suggest that the effects of previous inputs persist within the current reservoir states, contributing to the heightened interaction among SHOs. However, it's worth noting that when we convert magnetization dynamics into reservoir states, we suspect that not all intricate and complex dynamics are fully captured due to the limited number of nodes employed in the calculation.

In typical experimental settings, the determination of virtual nodes is guided by factors such as the relaxation times of the dynamical systems or the limitations of the measurement instrument's sampling rate. As the number of nodes increases, so does the dimensionality of the output when transforming input data. Nevertheless, considering our focus on practical applications for memory-constrained devices, and the self-imposed constraints introduced in Chapter 1, we have deliberately set limitations on both training data and virtual nodes. In this context, we doubled the number of nodes from 10 to 20 and recalculated the memory capacity (MC). As expected, the nonlinearities inherent in the SHO array's oscillation dynamics become incorporated into the reservoir nodes, resulting in an enhancement of MC. Figure 6.6b presents the $C^2\ (d)$ values for the same four regions, now with 20 nodes. It's evident that the $C^2\ (d)$ values for all regions exceed 0.5, indicating a significant influence of previous inputs up to a delay of 10. This suggests that the increase in the number of nodes provides a more comprehensive representation of the system dynamics, capturing a more extended history of input information. An array of 4 × 4 SHOs provided a maximum of MC = 8.2 at a $J_c = 4.0 \times 10^{12}$ A/m² which is the region "d"

6.3. Effect of inter-distance and memory capacity

Our primary objective revolves around enhancing memory capacity (MC) while working within the constraints of limited number of Spin Hall Oscillators (SHOs). To achieve this, we embarked on a comprehensive exploration, focusing on the simplest 2×1 array structure, which had previously exhibited the lowest MC. Within this framework, we manipulated the inter-distance (Gap) between SHOs, varying it from 10 to a maximum of 80 nm. This allowed us to get insights into the dynamics and MC, utilizing 20 nodes as reservoir states. As in our previous investigations, we classified MC into four distinct regions, namely, “a,” “b,” “c,” and “d,” representing the hierarchy from the lowest to the highest MC values. Figure 6.7 vividly illustrates these regions on a color map, revealing the complex phenomena of MC. Notably, we uncovered that the maximum MC, reaching an impressive 8.2, is confined to a narrow range of input current density (J_c) when Gap equals 10 nm. Conversely, the minimum MC of 6.1 occurs within the same J_c range but with a Gap of 80 nm.

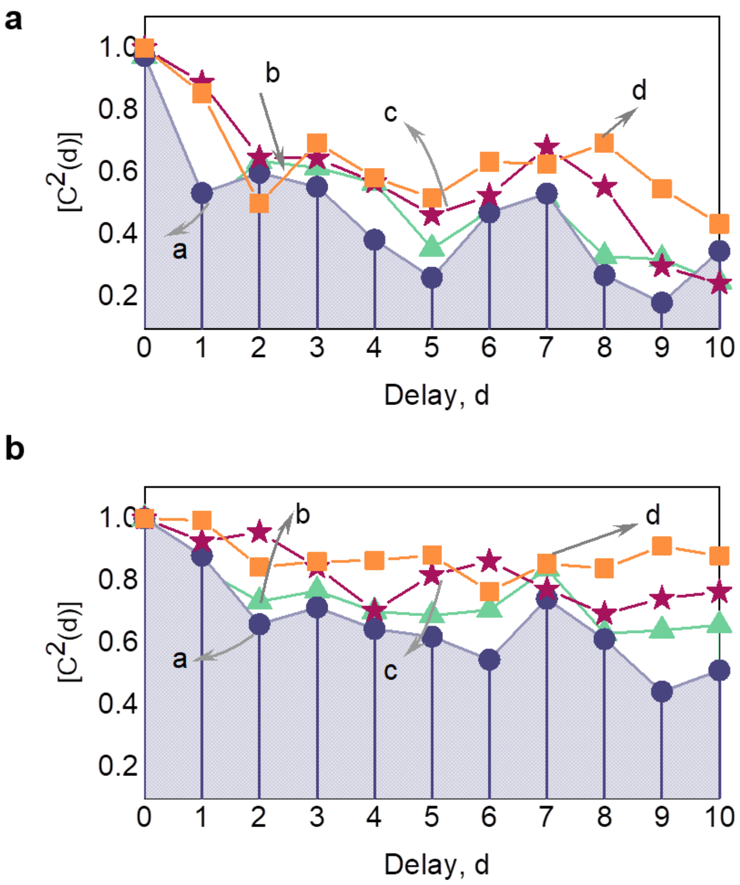


Fig. 6.6 a Square of correlation coefficients $C^2(d)$ for delays up to 10 for the regions “a”, “b”, “c”, “d” regions in the Fig. 6.3 calculated with 10 reservoir nodes **b** calculated with 20 nodes.

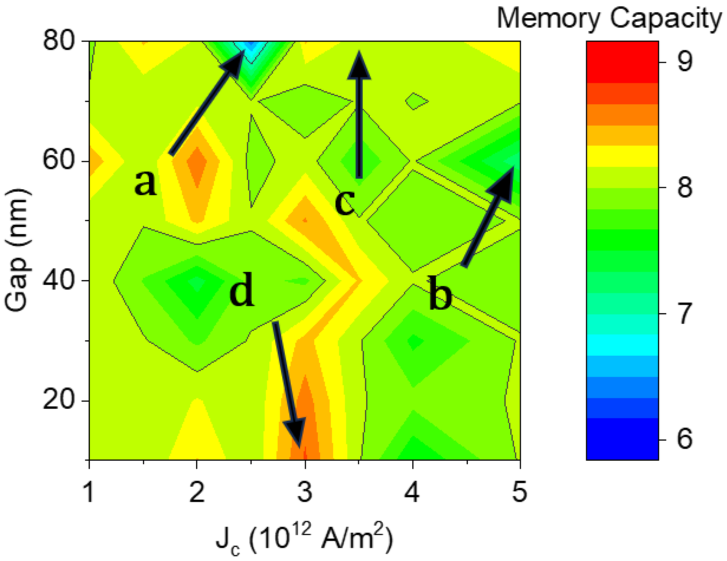


Fig. 6.7 Color map of memory capacity for various inter-distance (Gap) for a 2 × 1 array structure. The regions with different MC are marked with letters a, b, c, d from least to higher MC values.

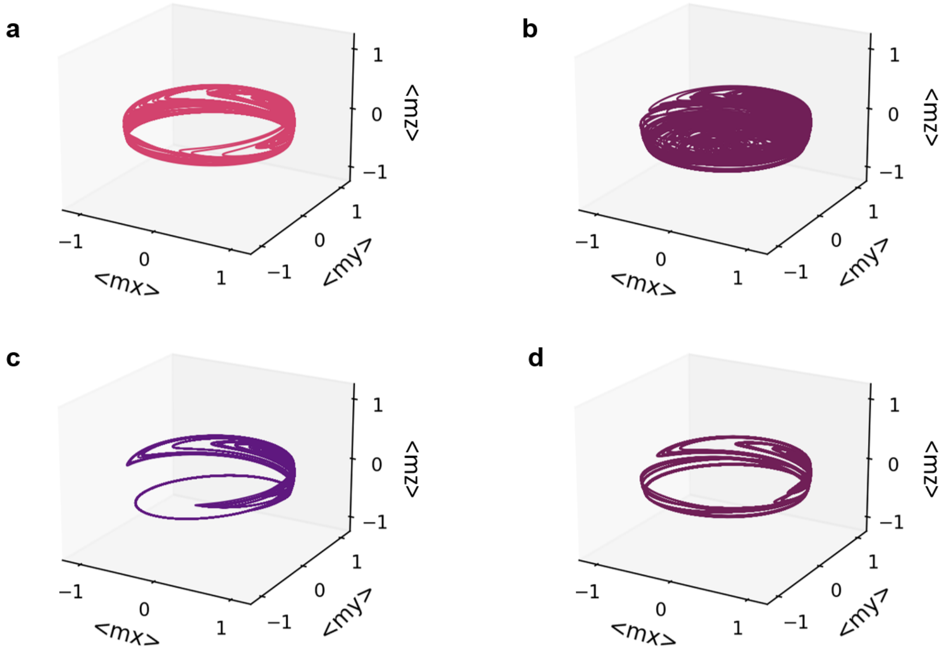


Fig. 6.8 a-d Oscillation trajectory of 2 × 1 array structure for continuous pulses for the regions marked in Fig 6.7.

To deepen our understanding, we turned our attention to the magnetization dynamics, offering valuable insights into the MC variations. Figures 6.8a to 6.8b showcase the oscillation trajectories of the four regions. Notably, when we closely examine regions “a” and “d” alongside their respective counterparts in Figures 6.7a and 6.7d, we observe that although the oscillations are steady-state, the MC still varies. However, in the case of Figure 6.7a, we can discern that steady-state oscillations aren't consistently available for all input pulses. Despite the modest 2-unit difference in MC, the complexity of the collective SHO behavior propels it beyond the MC of a single oscillator (MC = 5, as previously observed in Chapter 5). From this observation, we can conclude that SHOs present themselves as strong candidates for hardware implementations of reservoir computing, even with minimal resources. Remarkably, the dynamics of SHOs can be adeptly controlled through input parameters and intrinsic dipolar coupling, achieved through the selective design of device structures.

6.4. Summary

In summary, our exploration has illuminated the information processing capabilities of SHOs, with two SHOs significantly increasing the complexity of magnetization dynamics. These findings align with the broader context of reservoir computing, shedding light on the potential of SHOs as efficient and adaptable hardware components for advanced computing tasks.

References

[1] A. D. Kovac, M. Koall, G. Pipa, and H. Toutounji, “Persistent memory in single node delay-coupled reservoir computing,” *PLoS One*, vol. 11, no. 10, pp. 1–15, 2016, doi: 10.1371/journal.pone.0165170.

[2] A. A. Velichko, D. V. Ryabokon, S. D. Khanin, A. V. Sidorenko, and A. G. Rikkiev, “Reservoir Computing using High Order Synchronization of Coupled Oscillators,” *arXiv*, no. Figure 1, pp. 1–8, 2020.

[3] F. Abreu Araujo and J. Grollier, “Controlling the synchronization properties of two dipolarly coupled vortex based spin-torque nano-oscillators by the intermediate of a third one,” *J. Appl. Phys.*, vol. 120, no. 10, 2016, doi: 10.1063/1.4962014.

[4] T. Taniguchi, “Synchronization of Spin Torque Oscillators through Spin Hall Magnetoresistance,” *IEEE Trans. Magn.*, vol. 53, no. 11, pp. 1–7, 2017, doi: 10.1109/TMAG.2017.2704588.

[5] N. Akashi *et al.*, “A Coupled Spintronics Neuromorphic Approach for High-Performance Reservoir Computing,” *Adv. Intell. Syst.*, vol. 4, no. 10, p. 2200123, 2022, doi: 10.1002/aisy.202200123.

[6] A. Vansteenkiste, J. Leliaert, M. Dvornik, M. Helsen, F. Garcia-Sanchez, and B. Van Waeyenberge, “The design and verification of Mumax3,” *AIP Adv.*, vol. 4, no. 10, p. 107133, 2014, doi: 10.1063/1.4899186.

Summary and Conclusion

Our study commenced with an in-depth comparison between SHOs and a perceptron layer, elucidating the profound advantages of SHOs in streamlining computations. The SHO's ability to perform nonlinear input transformations and adapt its weights internally emerged as a focal point. The SHO's capacity to carry out nonlinear input transformations and internal modification of its weights as a single self-computing unit that are central to the success of SHO-based classification systems. This innate ability proved crucial for data-driven tasks and provided feature extraction capabilities, which are essential to the effectiveness of SHO-based classification systems. It was demonstrated that a single spin Hall oscillator can classify sixteen distinct binary 4-bit patterns (0000 to 1111) in real time. By adjusting an input digit pulse pattern, it was possible to control and tune the intrinsic magnetization oscillations of the spin Hall oscillator. A linear training network and simplified output layer computations were used to test the sample's performance using the standard MNIST handwritten digit data set classification, which resulted in an accuracy of 83 %.

In a reservoir computing framework, a significant portion of our research was devoted to examining how the magnetization dynamics of a single SHO affects its information processing capability. We discovered a significant improvement in memory capacity (MC) through meticulous numerical analysis and active control of input parameters. The MC eventually reached a saturation point between 4 and for a single SHO. The change in magnetic oscillations from transient dynamics to in-plane limit cycle oscillations was closely related to this phenomenon. Intriguingly, the presence of out-of-plane limit cycle oscillations resulted in a decrease in MC with increased current pulse amplitude or pulse width. Nonlinear Auto Regressive Moving Average (NARMA2) tasks and three-bit parity checks were two practical Reservoir Computing tasks that underwent scrutiny. The findings highlighted the critical connection between MC and task performance, with input parameters corresponding to higher MC producing fewer errors in these tasks.

Even with a simple setup of only two SHOs, our investigation advanced to include the exceptional information processing abilities of SHOs. We highlighted the emergence of complex magnetization dynamics and their association to the information processing capabilities. This finding highlighted SHOs' flexibility and potential as effective hardware elements for high-end computing tasks in memory constraint devices. SHOs demonstrated their skill in tasks that go beyond traditional computing paradigms as magnetization dynamics became more complex.

In summary, this research not only highlights the capabilities of SHOs but also paves the way for their integration into mainstream computing, promising transformative advancements in the field of information processing and machine learning.

ACKNOWLEDGEMENTS

I would like to begin by expressing my deepest gratitude to my supervisor, Prof. Yasuhiro Fukuma, of the Kyushu Institute of Technology. Prof. Fukuma's unwavering support, guidance, and mentorship during my doctoral journey have been invaluable. Under his supervision, I have grown into a more proficient and independent researcher. I am particularly thankful for the freedom he granted me in conducting research within the laboratory and his insightful critique of my findings. Our collaborative efforts with researchers from diverse laboratories and universities have broadened my perspective on scientific research, fostering in me a sense of curiosity and responsibility.

I extend my heartfelt appreciation to the distinguished professors who kindly agreed to review my PhD thesis: Prof. Edmund Soji Otabe and Prof. Masonori Takabayashi from Kyushu Institute of Technology, and Prof. Om Prakash Sinha from Amity University, India. Their expertise and critical feedback have played a crucial role in shaping the quality of my work.

Special thanks go to Prof. Rajdeep Singh Rawat of Nanyang Technological University, Singapore, for granting me access to the micromagnetic simulation facility. His support was instrumental in the completion of my thesis, and I am grateful for his valuable insights, despite his busy schedule.

I am indebted to several inspiring individuals who provided guidance and motivation throughout my research journey: Prof. S. Annapoorni (Delhi University, New Delhi), Prof. Rohit Medwal (IIT Kanpur), and Prof. Surbhi Gupta (Motilal Nehru National Institute of Technology, Allahabad). Their wisdom and encouragement were instrumental in my research progress.

I would like to acknowledge Prof. Hironori Asada from Yamaguchi University for granting me access to the sample fabrication facility and for his guidance in research. Prof. Akiyoshi Baba and Mr. H. Sato from the Center for Microelectronics System at KIT deserve special mention for their continuous support in facilitating access to the cleanroom facilities for device fabrication.

My heartfelt appreciation extends to my fellow laboratory members, with whom I had the privilege to collaborate during my PhD journey. I am grateful for their camaraderie, especially to Angshuman Deka, Shashank Tyagi, Utkarsh Shashnk, Shibata Taiga, Arun Jacob Mathew, Garima Vashisth, Kazuma Nishimuara, Ruoyan Feng, Chisato Yamanka, Yosuke Hasunaka, and all other members, both within and outside the laboratory.

Beyond the academic realm, I owe a debt of gratitude to my family—my parents and brother—for their unwavering support and encouragement throughout my PhD. Their belief in me was my driving force, and I could not have pursued an advanced educational degree without them. I would also like to express

my sincere thanks to Dr. Osamu Nawata for treating me like family and providing accommodation in Iizuka. To my friends, both new and old, who have been a source of constant support, especially Ms. Kobayashi-san, Ms. Yamabe-san, and Dr. Prakash, I extend my appreciation. The friendships I forged with people from different corners of the world during my PhD have enriched my understanding of diverse cultures and expanded my worldview.

Lastly, I extend my heartfelt gratitude to the wonderful community of Iizuka. Your support and warmth have been integral to my journey, and I could not have completed my PhD without the collective encouragement of all those mentioned here and many more. Thank you all for being a part of this significant chapter in my life.

MOHAN JOHN REX

Kyushu Institute of Technology, Japan

Sep 2023