EPFL

Thèse n° 9121

# Timing is Everything

Présentée le 19 avril 2022

Faculté informatique et communications
Laboratoire de communications audiovisuelles
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

**Karen ADAM**

Acceptée sur proposition du jury

Dr O. Lévêque, président du jury
Prof. M. Vetterli, Dr A. J. Scholefield, directeurs de thèse
Prof. A. Lazar, rapporteur
Prof. P. L. Dragotti, rapporteur
Prof. G. Indiveri, rapporteur

École polytechnique fédérale de Lausanne

2022

# Acknowledgements

While the degree resulting from this thesis is attributed to one person, the work itself was far from being a one-(wo)man show.

First, I would like to thank my two advisors, Martin and Adam. Looking around, I see that scientific enthusiasm is not very easily coupled to moral support, and I have been very lucky to benefit from both.
Martin, every meeting brought me more inspiration and motivation. Thank you for giving me the freedom to pursue my own path, and for always making me feel like what I worked on was of utmost importance. Thank you, also, for realizing that PhD students are more than paper-making machines, and for asking about the collateral damages, but also about the collateral pleasures.
Adam, you spent countless hours reading and commenting my writing and teaching me the ropes. Thank you for being patient when I entangled these ropes and was too stubborn to accept any advice. Thank you, also, for being so reliable and thorough. You smoothened out many bumps and potholes during the first years of my PhD.

To Pier Luigi Dragotti, Giacomo Indiveri, Aurel Lazar, and Olivier Lévêque, who made up the jury for my private defense, thank you for carefully reading and commenting the thesis, and for following up with an exciting discussion.
I am also grateful to Luciano Sbaiz and Feng Yang, for hosting me during my (virtual) time at Google, and for keeping the discussion going after that; and to Peter Maurer, Xiaofei Yu and Liang Jiang for the collaboration that turned into the second part of this thesis and for patiently explaining things about which I otherwise never would have learnt.

As for my collegues in LCAV, Nicoletta, thank you for making all the paperwork go smoothly, always with a smile and a readiness to chat. Paolo, thank you for the cynical discussions and for complimenting my tartiflette. Kiki, you contributed to this thesis through many ways, from your excitement to discuss science at 7AM, to your super-comprehensive feedback on my writing, to providing support during turbo coffee breaks whenever I needed it. Michalina, thank you for listening to my incomprehensible formulations and making clean statements or questions out of them, which you then often respectively proved or answered. I learned a lot from you.
Adrien, thank you for being cheerful and talkative, and for the occasional boat ride. Adrian,

thank you for sharing an office, despite my numerous interruptions in a day (let's pretend you had a choice). Eric, thank you for the life discussions during runs and breaks. Dalia, Golnoosh, Julien, Luc, Matthieu, Mihailo, Miranda, Sepand, thank you for the discussions, lunches, coffee breaks, runs, reading groups, and lab outings.

To my friends in Lausanne, thank you Sandra and Bharath for playing devil's advocate and keeping me humble, Mariam and Bogdan for the very random late night talks, Jakab, for running along (sorry about your knees again), Kiki (again) and Sven, for all the ideas and pots you left behind, Elsa, Virginia, Ahmad, for all the corny, corny jokes, and Fayez, for trying so many terrible versions of lebanese zucchini omelets. And thank you to many others, including Elene, George, Martin, Matteo, Merlin, Paritosh, Roman, Sahand, Sebastien, Sena, for sharing lunches, dinners, coffees, hikes, and political and scientific discussions.

To Bernd, thank you for going through the motions of the rollercoasters, and patiently waiting for me on the ski slopes, hiking trails, and water, with encouragement always at the ready. Thank you for the constant reminder: "Es ist nie zu spät für eine glückliche Kindheit".

To my parents, thank you for giving me every opportunity, and for teaching me resilience in the face of hardships. Elie and Romy, thank you for helping me set out on this journey. Marc and Yara, thank you for making me feel like you're always within reach, and like I'm always welcome. You all provided invaluable support, even when distance was unsurmountable. This thesis is dedicated to you.

*Lausanne, April 2022* Karen Adam

# Abstract

In this thesis, timing is everything. In the first part, we mean this literally, as we tackle systems that encode information using timing alone. In the second part, we adopt the standard, metaphoric interpretation of this saying and show the importance of choosing the right time to sample a system, when efficiency is a key consideration.

Time encoding machines, or, alternately, integrate-and-fire neurons, encode their inputs using *spikes*, the timings of which depend on the input and therefore hold information about it. These devices can be made more power efficient than their clocked counterparts and have thus been studied in the fields of signal processing, event-based vision, computational neuroscience and machine learning. However, their timing-based spiking output has so far often been considered a nuisance that one must make do with, rather than a potential advantage. In this thesis, we show that this timing-based output equips spiking devices with capabilities that are out of reach for classical encoding and processing systems.

We first discover the benefits of *time* encoding on multi-channel encoding and recovery of a signal: with time encoding, clock alignment is easy to solve, although it poses problems in the classical sampling scenario. Then, we study the time encoding of low-dimensional signals and see that the asynchrony of spikes allows for a lower sample complexity in comparison with synchronous sampling. Thanks to this same asynchrony, time encoding of video results in an entanglement between spatial sampling density and temporal resolution—a relationship which is not present in frame-based video. Finally, we show that the all-or-none nature of the spikes allows training spiking neural networks in a layer-by-layer fashion—a feat that is impossible with clocked, artificial neural networks, due to the credit assignment problem.

The second part of this thesis shows that choosing the right timing of samples can be crucial to ensure efficiency when performing nanoscale magnetic sensing. We are given a stochastic process, where each sample at time $t$ follows a Bernoulli distribution, and which is characterized by oscillation frequencies that we are interested in recovering. We search for an optimal approach to sample this process, such that the variance of the frequencies' estimates is minimized, given constraints on the measurement time. The models we assume stem from the field of nanoscale magnetic sensing, where the number of parameters to be estimated varies with the number of spins one is trying to sense. We present an adaptive approach to choosing samples in both the single-spin and two-spin cases and compare the

adaptive algorithm's performance to classical approaches to sampling.

In both parts of the thesis, we move away from classical amplitude sampling and consider cases where timing takes the forefront and amplitude information is merely binary, to show that timing can carry information and that it can control the amount of information gain.

**Keywords:** time encoding, sampling, event video, spiking neural networks, nanoscale magnetic sensing, adaptive sampling

# Résumé

Dans cette thèse, nous étudions l'importance du "timing". Dans la première partie, nous nous intéressons à des systèmes qui utilisent un codage temporel pour représenter l'information. Dans la deuxième partie, nous montrons l'importance de choisir le bon moment pour échantillonner un processus, lorsque l'efficacité de l'échantillonage est une considération clé.

Les machines à codage temporel, ainsi que les neurones à intégration-et-tir, codent leurs entrées à l'aide d'impulsions dont le *moment* dépend de l'entrée et qui contiennent donc des informations sur cette entrée-là. Ces appareils peuvent être rendus plus économes en énergie que leurs homologues synchronisés et ont donc été étudiés dans les domaines du traitement du signal, de la vision événementielle, des neurosciences computationnelles et de l'apprentissage machine. Cependant, leur sortie basée sur la temporalité des impulsions a jusqu'à présent souvent été considérée comme un inconvénient qu'il faut apprendre à gérer, plutôt que comme un avantage. Dans cette thèse, nous montrons que cette sortie basée sur le temps confère aux systèmes à codage temporel des capacités qui sont hors de portée des systèmes de codage et de traitement classiques.

Nous découvrons d'abord les avantages du codage *temporel* sur le codage et la récupération multicanaux d'un signal : avec le codage temporel, l'alignement des horloges de différents échantilloneurs est facile à résoudre, alors que cet alignement est difficile à réaliser dans le scénario classique d'échantillonnage. Ensuite, nous étudions le codage temporel de signaux de petite dimension et voyons que l'asynchronie des impulsions permet une complexité d'échantillonnage plus efficace par rapport à l'échantillonnage synchrone, pour atteindre une fidélité de reconstruction semblable. Grâce à cette même asynchronie, le codage temporel de la vidéo entraîne un enchevêtrement entre la densité d'échantillonnage spatiale et la résolution temporelle - une relation qui n'est pas présente dans la vidéo basée sur les images. Enfin, nous montrons que la nature tout ou rien des impulsions permet d'entraîner les réseaux de neurones à impulsion, couche par couche, ce qui est impossible avec les réseaux de neurones artificiels cadencés, en raison du problème d'attribution des crédits.

La deuxième partie de cette thèse montre que choisir le bon moment pour échantilloner un processus peut être crucial pour garantir l'efficacité de la détection magnétique à l'échelle nanométrique. Nous considérons un processus stochastique, où chaque échantillon au temps $t$ suit une distribution de Bernoulli, et qui est caractérisé par des fréquences d'oscillation

que nous voudrions récupérer. Nous cherchons une approche optimale pour échantillonner ce processus, de telle sorte que la variance des estimations des fréquences soit minimisée, étant donné les contraintes sur le temps de mesure. Les modèles que nous supposons proviennent du domaine de la détection magnétique à l'échelle nanométrique, où le nombre de paramètres à estimer varie avec le nombre de spins que l'on essaie de détecter. Nous présentons une approche adaptative pour le choix des échantillons dans les cas à un et deux spins et nous comparons les performances de l'algorithme adaptatif aux approches classiques d'échantillonnage.

Dans les deux parties de la thèse, nous nous éloignons de l'échantillonnage d'amplitude classique et considérons des cas où le moment d'èchantillonage ou d'impulsion est au premier plan et où l'information d'amplitude est simplement binaire, afin de montrer que le temps peut contenir de l'information et qu'il peut contrôler la quantité de gain d'information.

**Mots clés :** codage temporal, échantillonnage, vidéo événementielle, réseaux de neurones à impulsions, détection magnétique à l'échelle nanométrique, échantillonnage adaptatif

# Contents

# Acronyms

| | | |
|---|---|---|
| ANN | Artificial Neural Network | Section 6.1 |
| C-TEM | Crossing Time Encoding Machine | Definition 2.1 |
| FRI | Finite Rate of Innovation | Definition 2.5 |
| FS | Fourier Series | Section 2.4 |
| LIF | Leaky-Integrate and Fire | Figure 1.3 |
| NMR | Nuclear Magnetic Resonance | Section 7.1.1 |
| nNMR | Nanoscale Nuclear Magnetic Resonance | Section 7.1.1 |
| NV | Nitrogen Vacancy | Section 7.1.1 |
| POCS | Projection onto Convex Sets | Definition 2.4 |
| RIP | Restricted Isometry Property | Section 2.5 |
| SNN | Spiking Neural Network | Section 1.3.2 |
| SVP | Singular Value Projection | Section 2.5 |
| TEM | Integrate-and-Fire Time Encoding Machine | Definition 2.2 |

# Information from Timing Part I

# 1 Introduction

## 1.1 Breaking with Tradition

The classical approach to sampling a signal consists of recording this signal's amplitude at uniformly spaced times. The result is a set of (time, amplitude) pairs which are the *discretization* of the signal. This scheme is popular because it is simple, intuitive and well understood.

In fact, if given a continuous signal with varying amplitudes, the most intuitive way to track its trajectory is to record the signal's value at certain times. Research abounds for this type of sampling scheme, the most canonical result being most often attributed to Shannon. He showed that a bandlimited signal can be *perfectly* reconstructed from its amplitude samples if the samples have a small enough time separation between them (Shannon, 1949). This result was published some seventy years ago, and since then, the beaten path to uniform sampling has branched out into many extensions and detours (Unser, 2000).

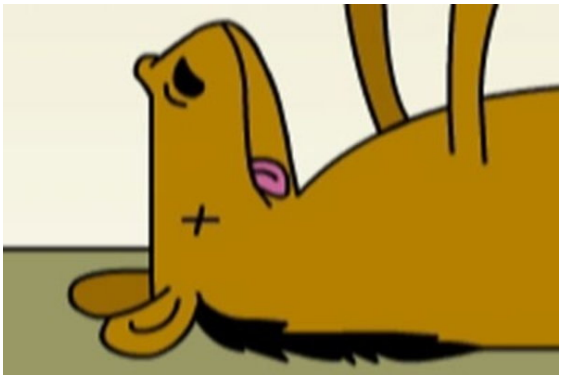


Figure 1.1 – The current state of the theory behind uniform amplitude sampling. Confused readers can refer to the footnote[1].

[1] See https://dictionary.cambridge.org/dictionary/english/flog-a-dead-horse.

It is therefore understandable that this clocked, uniform approach be not only a main component of sampling but also the chosen route for many engineered systems. However, while this approach works well for many applications, it is not necessarily the most resource-efficient approach.

To provide intuitive examples, it is useful if a subway predictably arrives at a station every five minutes, following a clocked scheme, such that people can plan their time. On the other hand, it is less useful for people to say "hello" every five minutes (or any interval for that matter), it is better if they say it when they meet someone. In our greeting example, the second operational scheme can be labeled as an event-based scheme: an action (which we call an "event" and which, in this case, corresponds to saying "hello") is *triggered* only when a specific criterion is met (in this case, the trigger is the encounter).

This part of the thesis concerns itself with using such an event-based operational scheme to encode signals.

Actually, such sampling or encoding schemes are very frequently used in real life. For example, one can measure rain using a tipping bucket rain gauge: water is accumulated in a bucket, and when the bucket is full, it tips over, triggering an electric signal which is used to mark the time at which the bucket tipped over. These *trigger times* give information about the amount of rainfall.

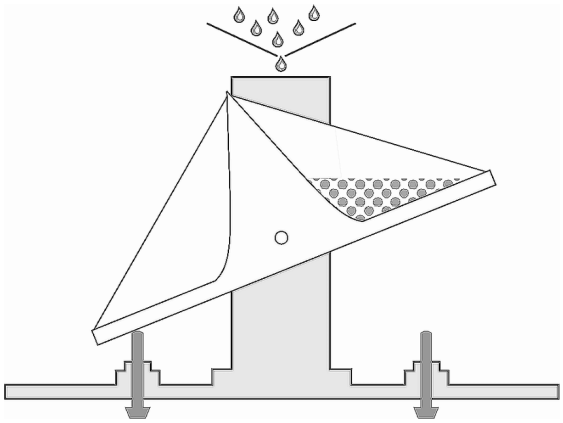


Figure 1.2 – A tipping bucket rain gauge records information about rainfall by waiting for rain to accumulate in one of the cusps until it tips and triggers the recording of the tipping time—An example of *time* encoding.

While I personally find tipping bucket rain gauges fascinating, they are not exactly hiding behind every bush. Neurons in the human brain, however, are ubiquitous and constitute the prime example for event-based systems. The accumulate-and-trigger mechanism of the rain gauge is actually closely related to part of the mechanism behind neuronal communication, as we will explore in the next section.

## 1.2 Time Encoding in the Wild

Neurons are cells in the brain that receive information about the world from receptors, process this information and transmit the result to other neurons for further processing, eventually controlling bodily functions and movement.

In a (very small) nutshell, neurons communicate with eachother by emitting (or *firing*) *action potentials* (or *spikes*)(Gerstner et al., 2014). Action potentials are sudden and short changes in a neuron's membrane potential, i.e. its voltage. They generally have a fixed shape and amplitude, and can be chemically transmitted to other neurons. According to simplified models of neurons, these spikes are generated when the spatial and temporal summation (or integration) of the electric current input to a neuron exceeds a threshold. In reality, the mechanism is much more complex, and includes many chemically and electrically triggered ion channels. Attempting to explain it here would lead us astray. A good reference is Kandel et al. (2000).

This simplified "integrate-and-fire" neuron model behaves similarly to the tipping bucket: a neuron integrates its electrical input current (essentially collecting the current in a bucket) and when the integral reaches a threshold (meaning the bucket is full), a *spike* is emitted (and the bucket tips over). Thus, the input to the neuron is *only* encoded in the timings of the spikes: the neuron's output, which is transmitted to other neurons, is a stream of spikes with fixed amplitude.

This output follows the *all-or-none* principle: the output either exhibits a spike of a fixed, large amplitude, which is strong enough to be conveyed to other neurons (the "all" level), or only fluctuates around the "resting potential", unnoticed by other neurons (the "zero" or "none" level).

It is often said that the brain evolved to use spikes because it did not have copper wires, but leaky axons, as wiring between neurons. Information needs to be transmitted between neurons, and spikes are a robust way to do that when wiring has high attenuation.

**In depth: Neuronal communication**

Neurons are comprised of dendrites, cell bodies, and axons. *Dendrites* are the extensions of the neurons that receive input from other neurons. *Axons* are the extensions that provide information to other cells. This information transfer happens through a synapse: when an action potential or spike reaches the axon terminal of the *presynaptic neuron*, *neurotransmitters* are released from *synaptic vesicles* located at the axon terminals. These neurotransmitters are released into the *synaptic cleft* where they bind to receptors on the dendrite of the *postsynaptic neuron*.

Neuronal axons are long and subject any signal traveling along them to attenuation.

Using streams of spikes as a signal remedies this, as fixed-amplitude spikes can easily be regenerated (by simply comparing the signal to a threshold and generating an action potential if needed). This mechanism is further facilitated by *Myelin sheaths* which are wrapped around neuronal axons and partially insulate them, such that the regeneration of the signal occurs at sites where there is a gap in this insulation, these sites are called *nodes of Ranvier*.

**Classical samplers versus neurons**

**Example 1.1.** We provide an example of signal encoding using classical amplitude sampling and using spiking integrate-and-fire neurons with a leak (Burkitt, 2006), in Fig. 1.3. Notice how, in the former case, the amplitude samples are always regularly spaced whereas, in the case of the simulated neuron, the rate at which spikes are emitted varies and increases as the signal's amplitude increases.

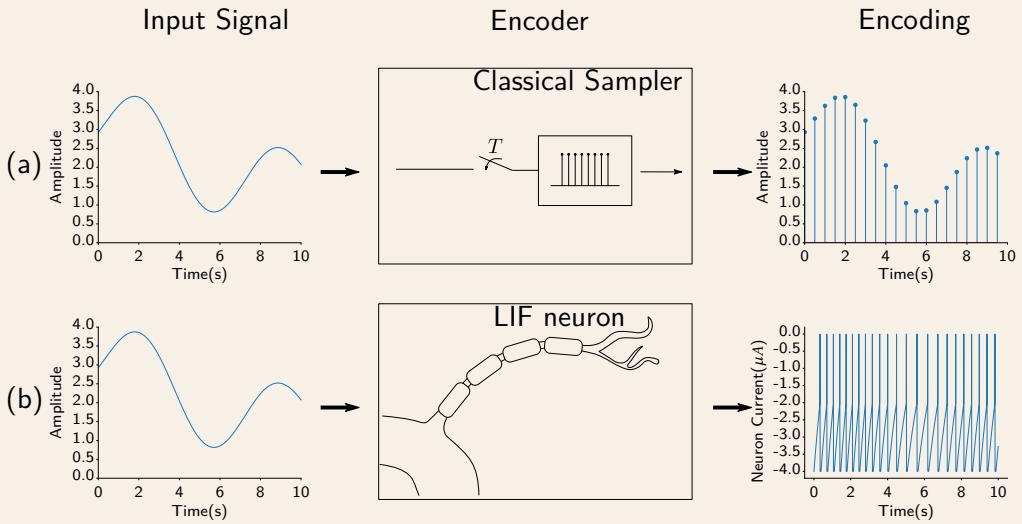



Figure 1.3 – Encoding of the same signal using **(a)** a classical uniform sampler, and **(b)** a spiking leaky-integrate and fire (LIF) neuron. In **(a)**, the signal is sampled every $T$ seconds. The output is then a series of equally spaced amplitude measurements. In **(b)**, we assume that the signal is injected as a current into a spiking leaky-integrate-and-fire neuron following the model described in Burkitt (2006) and implemented using a spiking neural network simulator called Brian (Goodman and Brette, 2009). The recorded output is the outgoing current which exhibits a series of action potentials, or spikes. Notice how the output spike streams are denser when the signal is stronger and sparser when the signal is weaker.

While it is clear why the brain "has to" use spikes from an implementation perspective, we will show in this thesis that the all-or-none and asynchronous nature of spikes also provides advantages in terms of efficiency in computation or richness of information—advantages

that do not exist in traditional, clock-based systems. As a consequence, our arguments help show that *individual* spike times are important in neuronal encoding, rather than accepting neuronal spike *rates* as the encoding paradigm in the brain.

## 1.3 Time Encoding in Captivity

Inspired by the efficacy of spikes in biology, devices that perform *time* encoding have been developed both to better understand the brain and for practical applications.

Such devices are nowadays mainly considered because they consume very little power. In fact, spikes are only triggered by specified, desired criteria; in the integrate-and-fire case, this is the crossing of a threshold. Moreover, spikes are all-or-none, which overcomes the need for power-hungry precise amplitude quantization, and only requires precise temporal quantization, at a time when clocks that run in the GHz range are readily available.

Two applications of spiking devices are event-based cameras and spiking neural networks.

### 1.3.1 Event-Based Cameras

The current approach to recording video—multiple frames taken in close succession to each other— evolved as it did because techniques to take pictures were established first, and video was developed as an extension thereof.

While there is no doubt that this approach works very well, there is no guarantee that it is the most efficient one. Assume, for example, that you are taking a video of a bird against a blue sky background. To record this video, every pixel records the intensity of the light it receives for *every* frame. It is easy to see that pixels waste energy by repeatedly recording the intensity of the sky, a static background.

This example illustrates why frame-based video is suboptimal from a sample-complexity perspective, i.e. many samples are needed to encode little information. Other inconveniences also stem from high power requirements and limited dynamic range.

Event-based video can help counter these issues. Event-based cameras have pixels, each of which emits an event (the equivalent of a spike) whenever its input exhibits a change that is "large enough" (Delbrück et al., 2010; Gallego et al., 2019). The output of an event-based camera is a stream of events associated with each pixel, where the streams are different across pixels, and the timings of the events depend on the input to the pixel which emits it, as depicted in Fig. 1.4. In this encoding paradigm, frames are obsolete, and information is only recorded when the sensors detect sought-after criteria in the input.

While the pixels of such a camera follow a *differentiate-and-fire* approach rather than an integrate-and-fire one, many of the theoretical results established for the latter approach can

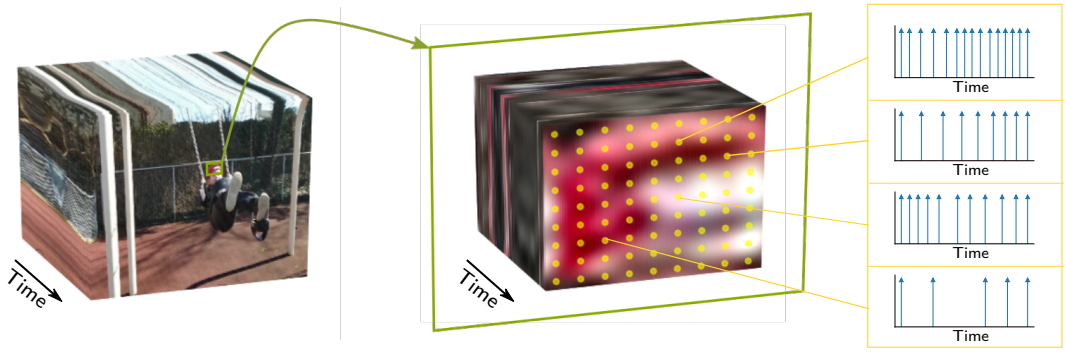



Figure 1.4 – Video setup: we assume that we have an array of time encoding devices, each of which is observing a scene at a particular location. The input to the device at a given location is a time varying signal and the device will output a stream of spikes, the timing of which is dependent on the input. On the left, we show the projection of the scene which is being observed. In the middle, we show a patch of this scene, which is interpolated under bandlimited periodic assumptions, with an overlay of event-based pixels shown in yellow. To its right, we zoom in to view the spiking output of some of the pixels. The video used is taken from the Need for Speed dataset (Galoogahi et al., 2017).

also be used for the former, as is further explained in Section 2.1.

In Chapter 5, we see how the asynchrony of the emitted events allows more efficient encoding of a scene compared to the frame-based video scenario.

### 1.3.2 Spiking Neural Networks

Artificial neural networks are undeniably successful at solving complex tasks, such as classifying images, navigating new terrain and processing natural language (LeCun et al., 2015; Schmidhuber, 2015).

The simplest neural networks process their inputs by recursively feeding them through weight matrices, then layers of nonlinearities, as illustrated in Fig. 1.5. In order for the network to learn useful tasks, the weight matrices are optimized, or learned, by striving to mimic examples that the network is given.

While these networks take inspiration from neuronal networks in biology, major discrepancies exist between the inspiration and the implementation, and one might wonder: If choices set by evolution were followed more closely, could neural networks be improved?

Neural networks traditionally apply nonlinearities in an instantaneous fashion: at time $t$, an artificial neuron, or *node*, receives an input $x(t)$ and applies a nonlinearity $f$ to it. The output $f(x(t))$ depends only on the *instantaneous* input at time $t$, disregarding the input's history[2].

[2]Of course, this does not prevent one from *enforcing* dependence on input history by introducing recurrence

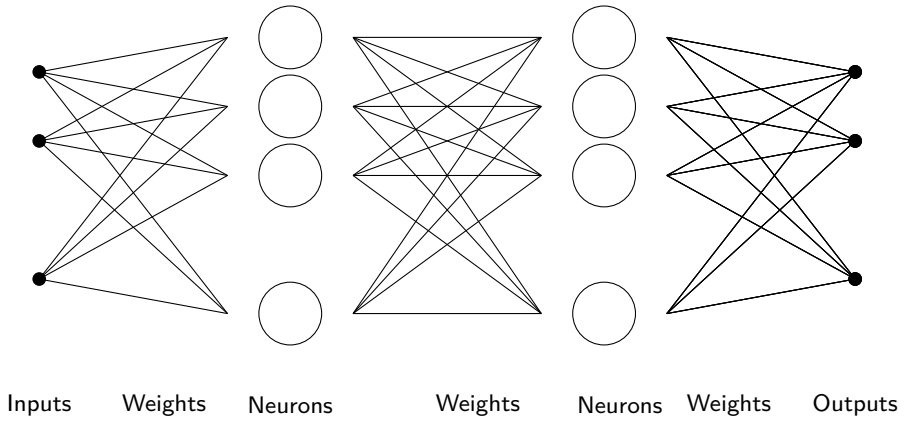



Figure 1.5 – A two-layer neural network, which transforms its input in a layer-by-layer fashion: at each layer, the input is passed through a weight matrix and fed into a series of artifical neurons or nodes, that apply a nonlinearity to their input.

When these traditional nodes are replaced by models which are more biologically sound—spiking neurons—the playing field suddenly changes. First, the output becomes dependent on the history of the input to the individual neuron. Second, the information at the output is encoded using times of spikes rather than amplitudes.

The reason behind this change lies, of course, in the behavior of spiking neurons which result in considerable consequences. In fact, not only are Spiking Neural Networks (SNNs) more power-efficient (Seo et al., 2011; Davies et al., 2018), they can also do analog processing until the very end, thus avoiding quantization errors which occur when digitizing inputs (Maass, 1997; Ghosh-Dastidar and Adeli, 2009). Furthermore, more resources can easily be used to encode more complex parts of a signal. In fact, in contrast to uniform sampling, it is easy to emit more frequent spikes, and SNNs can thus provide more flexible information transfer rates.

However, training spiking neural networks remains an open question with many challenges and opportunities (Neftci et al., 2019; Comsa et al., 2020; Indiveri and Horiuchi, 2011; Indiveri and Liu, 2015).

In Chapter 6, we see how spiking neural networks can learn layer by layer—a feat that is hard to achieve in artificial neural networks.

## 1.4 Our Contributions

Efforts have recently been invested to better understand spike-based systems, whether in the realm of event-based vision (Gallego et al., 2019), in the realm of spiking neural networks (Tavanaei et al., 2019) or in the theoretical realm of decoding bandlimited or

in the architecture as done in recurrent neural networks (Graves and Schmidhuber, 2005).

finite-rate-of-innovation signals from their time encoding (Lazar and Tóth, 2003; Gontier and Vetterli, 2014; Lazar, 2005; Lazar and Pnevmatikakis, 2009; Lazar, 2010; Alexandru and Dragotti, 2019; Hilton et al., 2021b,a; Rudresh et al., 2020; Kamath et al., 2021; Florescu and Coca, 2015; Martínez-Nuevo et al., 2016; Lai et al., 2017; Feichtinger et al., 2012; Thao and Rzepka, 2020; Lazar et al., 2008; Saxena and Dahleh, 2014). However, comparisons between event-based sensing and standard sampling have mostly considered the timing-based output of event-based sensing to be more of a pesky necessity that requires some work-around rather than a blessing in disguise. Actually, it is precisely this asynchrony and all-or-none nature of the spikes that can allow for better resolution and more flexibility when using event-based sampling.

We will focus on the latter claim and support it by establishing reconstruction guarantees for spike-based sampling and comparing these guarantees to the equivalent uniform sampling scenario.

We start by providing a summary of the notation used throughout this part of the thesis. We then establish the background knowledge needed for our results in Chapter 2, where we explain the time encoding paradigm as a formalization of a spiking or event-based sampler, and cover essential results on signal reconstruction from time encoding, and on the reconstruction of "sparse signals" from more general sampling schemes. In the following chapters, we proceed to study time encoding under different setups of multi-channel integrate-and-fire systems: single-signal time encoding and recovery, mixed multi-signal time encoding and recovery, video time encoding and recovery, and training of two-layer spiking neural networks.

We advise readers who are unfamiliar with time encoding to peruse at least Section 2.1 of the background chapter. Other sections can be perused as needed; recommended reading is indicated at the beginning of each chapter.

### 1.4.1 Chapter 3: Spike *Timing* allows Clock Alignment

We see how the timing-based nature of the output of spiking devices gives an advantage over classical uniform sampling in the case of multi-channel encoding of a signal.

More specifically, we show that, if a $2\Omega$-bandlimited signal can be perfectly reconstructed after sampling using one time encoding machine or integrate-and-fire neuron, then a $2N_c\Omega$-bandlimited signal can be perfectly reconstructed after sampling using $N_c$ time encoding machines, if these machines have integrator shifts that ensure that their outputs are different. This result doesn't require the knowledge of said shifts.

The equivalent scenario in uniform sampling is more difficult to solve: the constraints on the input that result from uniform sampling from *different* channels with unknown clock shifts cannot easily be assigned to a common time frame and complex nonlinear techniques are required.

With spiking devices, information lies in the timing of the spikes at the output which helps overcome the difficulty of aligning clocks.

### 1.4.2 Chapter 4: Spike *Asynchrony* allows Group Work

We extend the results of Chapter 3, and see how the asynchrony of spikes across different spiking devices allows gathered information to be less redundant and encoding to be more efficient.

We examine the problem of time encoding mixed signals using multiple channels, resembling a feedforward step of a spiking neural network. We show that, if the underlying dimensionality $N_s$ of the input is lower than the number $N_c$ of channels, under certain conditions on the input and mixing, the number of spikes needed for reconstruction depends on the number of degrees of freedom in the low-dimensional space, rather than that in the high-dimensional space, if the different channels work together.

More specifically, we show that if a certain channel or neuron spikes too little, it can be compensated for by having another neuron spike more often, but only up to a certain extent. This ability to work in groups is non-existant in synchronous uniform sampling, where samples are taken at the same time and provide linearly *dependent* constraints.

### 1.4.3 Chapter 5: Spike *Asynchrony* entangles Temporal and Spatial Resolution in Video

We build on the results of Chapter 4 to investigate the problem of time encoding bandlimited video and see that using event sensors or time encoding machines results in streams of events that are asynchronous across sensors, resulting in an entanglement between spatial and temporal resolution.

In the frame based case, spatial and temporal resolution are uniquely and respectively defined by the pixel gridding and frame rate, but in the case of event-based vision, we will see that the temporal resolution is also affected by the pixel gridding. As a result, both temporal and spatial resolution in event-based vision can be increased by increasing the number of pixels.

### 1.4.4 Chapter 6: *All-or-none* Spikes help train SNNs

We start to bridge the gap between time encoding and spiking neural networks by formulating the training problem of SNNs as a constraint satisfaction problem. This formulation allows us to understand the power of spikes compared to traditional ANN nonlinearities that are graded (i.e. have varying amplitudes) and synchronous.

We understand how to recover, i.e. how to learn, the weights of a two-layer spiking neural

network by bypassing the backpropagation algorithm and using constraints that are applied to each layer at a time, constraints that are obtained from a teacher network with the same architecture.

A key ingredient at play is the all-or-none and asynchronous nature of the spikes within a spiking neural network, which will allow, in the noiseless case, perfect recovery of hidden activations, and thus of the weights of the network.

Without further ado, we proceed to laying the groundwork for this part of the thesis, and leave the introduction for the second part for Chapter 7.

# Notation

| Chapter 2 | |
|---|---|
| $t$ | Time |
| $\omega$ | Frequency |
| $\Omega$ | Bandwidth |
| $\delta(t)$ | Dirac delta |
| $y(t)$ | Time-varying signal input to TEM |
| $\kappa$ | Integrator constant of a TEM |
| $\theta$ | Threshold of a TEM |
| $\beta$ | Bias of a TEM |
| $z(t)$ | Output of the integrator of a TEM |
| $\tau_\ell$ | $\ell^{\text{th}}$ spike time of a TEM |
| **Chapter 3** | |
| $N_c$ | Number of TEMs (where $c$ stands for channels) encoding a signal |
| $y_i(t)$ | Time-varying signal input to TEM$_i$ |
| $\Delta z_i$ | Integrator shift between TEMs $i$ and $i+1$ (modulo the number of total TEMs) |
| $z_i(t)$ | Output of the integrator of TEM$_i$ |
| $\tau_{i,\ell}$ | $\ell^{\text{th}}$ Spike time of TEM$_i$ of a series of TEMs |
| $\tilde{\tau}_\ell$ | Combined and ordered spike times of a series of TEMs |
| **Chapter 4** | |
| $\mathbf{y}(t)$ | Vector of time-varying signals input to TEM(s) |
| $\mathbf{x}(t)$ | Vector of the $N_s$ underlying signals input to a series of TEMs, before mixing |
| $\mathbf{C}(\mathbf{y})$ | Coefficients defining $\mathbf{y}(t)$, as in (A4) |
| $\mathbf{C}(\mathbf{x})$ | Coefficients defining $\mathbf{x}(t)$, as in (A5) |
| $N_s$ | Number of underlying signals that can represent $\mathbf{y}(t)$ |
| $x_j(t)$ | $j^{\text{th}}$ entry in $\mathbf{x}(t)$ |
| $\mathbf{A}$ | Mixing matrix in the context of time encoding |
| **Chapter 5** | |
| $y(d^{(1)}, d^{(2)}, t)$ | Time-and-space-varying signal |

| | |
|---|---|
| $D^{(1)}, D^{(2)}, T$ | Periods in the first spatial, the second spatial and the temporal dimensions, respectively |
| Chapter 6 | |
| $n^{(k)}$ | Number of TEMs in the $k^{\text{th}}$ layer of a feedforward SNN |
| $\text{TEM}_i^{(k)}$ | $i^{\text{th}}$ TEM of the $k^{\text{th}}$ layer of a feedforward SNN |
| $\tau_{i,\ell}^{(k)}$ | $\ell^{\text{th}}$ spike time of $\text{TEM}_i{}^{(k)}$ of the $k^{\text{th}}$ layer of TEMs in a SNN |
| $\mathbf{W}^{(k)}$ | Weight matrix of layer $i$ in a feedforward SNN |
| $x_j^{(k)}(t)$ | $j^{\text{th}}$ input to mixing weight matrix $\mathbf{W}^{(k)}$ of the $k^{\text{th}}$ layer of a SNN |

# 2 Background

This chapter provides the technical background needed to tackle the next chapters in Part I. We strongly encourage the reader to at least read Section 2.1 on the basics of time encoding before proceeding to the next chapters. The other sections can be perused as needed.

## 2.1 Time Encoding: Basics

Time encoding differs from traditional sampling, not only in the way it gathers information about a signal, but also in the way the information is represented and thus reconstructed.

In fact, in traditional sampling, a signal is filtered, sampled, and then encoded using (time, amplitude) pairs. In time encoding, as the name suggests, a signal is encoded using (signal-dependent) times only; these are the times at which a filtered version of the signal matches a known test signal.

We start with a general definition of a time encoding machine and later consider specific models, which include the integrate-and-fire time encoding machine—the model this thesis mainly focuses on.

**Crossing TEM**

**Definition 2.1.** A *Crossing Time Encoding Machine* (C-TEM) with test functions $\{\phi_\ell(t)\}_{\ell\in\mathbb{Z}}$, linear filter $h(t)$, and time-varying input $y(t)$ outputs the sequence of times $\tau_\ell$ such that:

1. $\phi_\ell(t)$ may be recovered from the set $\{\tau_i, i \le \ell - 1\}$,
2. $\big(h * y\big)(\tau_\ell) = \phi_\ell(\tau_\ell)$, and
3. $\big(h * y\big)(t) \neq \phi_\ell(t) \forall\, t \in (\tau_{\ell-1}, \tau_\ell)$.

This definition is adapted from the definition of a C-TEM from Gontier and Vetterli (2014). For those familiar with the latter definition, ours additionally includes a filter before the comparison between input and test function and allows for non-continuous test functions $\{\phi_\ell\}$. These changes allow our definition to also encompass integrate-and-fire time encoding machines.

**Sampling using a crossing time encoding machine**

**Example 2.1.**

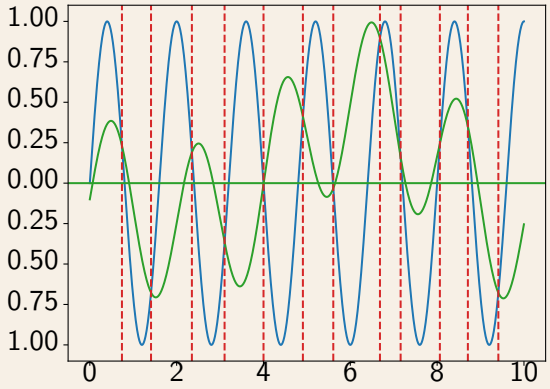



Figure 2.1 – Sampling of a signal, shown in green, using a crossing time encoding machine that has a single test function $\phi_\ell(t) = \sin\left(\frac{5}{4}\pi t\right), \forall \ell$ shown in blue. The recorded sequence $\tau_\ell$ marks the intersections of the input signal with the test function, here shown in dashed red lines.

**Integrate-and-fire TEM**

**Definition 2.2.** An *integrate-and-fire time encoding machine* (TEM) with parameters $\kappa$, $\theta$, and $\beta$ takes an input signal $y(t)$, adds to it a bias $\beta$, and integrates the result, scaled by $1/\kappa$ ($\kappa$ being the integrator constant), until a threshold $\theta$ is reached. Once this threshold is reached, a time $\tau_\ell$ is recorded, the value of the integrator resets to $-\theta$ and the mechanism restarts. We say that the machine spikes at the integrator reset and call the corresponding recorded time $\tau_\ell$ a *spike time*. We assume that we have access to the spike times $\tau_\ell$, either as a list, or in the form of a stream of unit-amplitude Dirac deltas emitted at the spike times.

In this thesis, the results we present concern integrate-and-fire time encoding machines, and, unless stated otherwise, it is the model we assume whenever we use the term “time encoding machine". However, many of the results we obtain can be extended to other models of

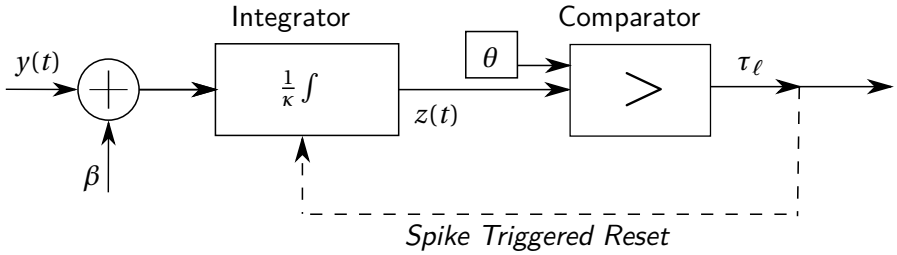



Figure 2.2 – Circuit of a time encoding machine with input $y(t)$ and parameters $\kappa$, $\theta$ and $\beta$, where $\kappa$ is the integrator constant, $\theta$ is a threshold above which a spike is triggered and $\beta$ is a positive bias added to the signal.

crossing time encoding machines, such as those presented in the Further Reading box below.

Figure 2.2 depicts the circuit of a TEM and Fig. 2.3 provides an example of how an input generates its output.

The first similar definition of a TEM appeared in Lazar and Tóth (2003), with some differences in how the information is output[1]. One can also define a TEM to have its integrator reset to zero, rather than to $-\theta$ when the threshold is reached, or define it to have the integrator constant $\kappa$ and threshold $\theta$ combined into one parameter $\theta' = \kappa\theta$. However, we prefer to keep the two parameters separate as each has a different origin in hardware: the integrator constant $\kappa$ arises from the integrator circuit and is therefore hard to change, whereas the threshold is a parameter of the comparator and is easier to manipulate.

Note how, with time encoding, our definition of a sample has changed. In traditional sampling, a sample denoted a (time, amplitude) pair, whereas here, a sample denotes a spike time. We use the terminology "spike time", to keep the analogy with integrate-and-fire neurons in the brain which produce responses by emitting action potentials. These action potentials have a fixed shape and amplitude, so the relevant information in a neuron's output lies in the *timing* of these action potentials, or spikes.

**Time encoding outputs**

**Definition 2.3.** A TEM outputs a stream of spikes or Dirac deltas, the timing of which holds the information. We call this type of output a *raw output*.

This stream can be transformed into a list of *times* of these spikes or Dirac deltas. In practice, this transformation requires time quantization, although we mostly ignore quantization errors in this thesis. We call this type of output a *listed output*.

[1] According to Lazar and Tóth (2003, 2004), a TEM outputs a piecewise constant signal, which switches between two levels at the spike times $\tau_\ell$, i.e. every time the threshold is reached. Therefore, the $\tau_\ell$'s could be recognized by observing the time at which the output of the TEM changes levels.

Further note that time encoding (whether crossing or integrate-and-fire) not only provides spike times, but also information about the sampling process (i.e. the test functions $\phi_\ell$) through these spike times, which eventually also provides one with a form of amplitude information. Unlike in traditional sampling, however, this amplitude information can be hidden without being lost, all the while preserving properties of signal reconstruction from traditional sampling, and simultaneously offering some benefits thanks to an encoding which is, in practice, purely time-based.

As a result, and as we will see in Section 2.2, such TEMs can encode bandlimited signals and recover them perfectly, provided a Nyquist-like condition is satisfied.

**Processing steps of a TEM**

**Example 2.2.**

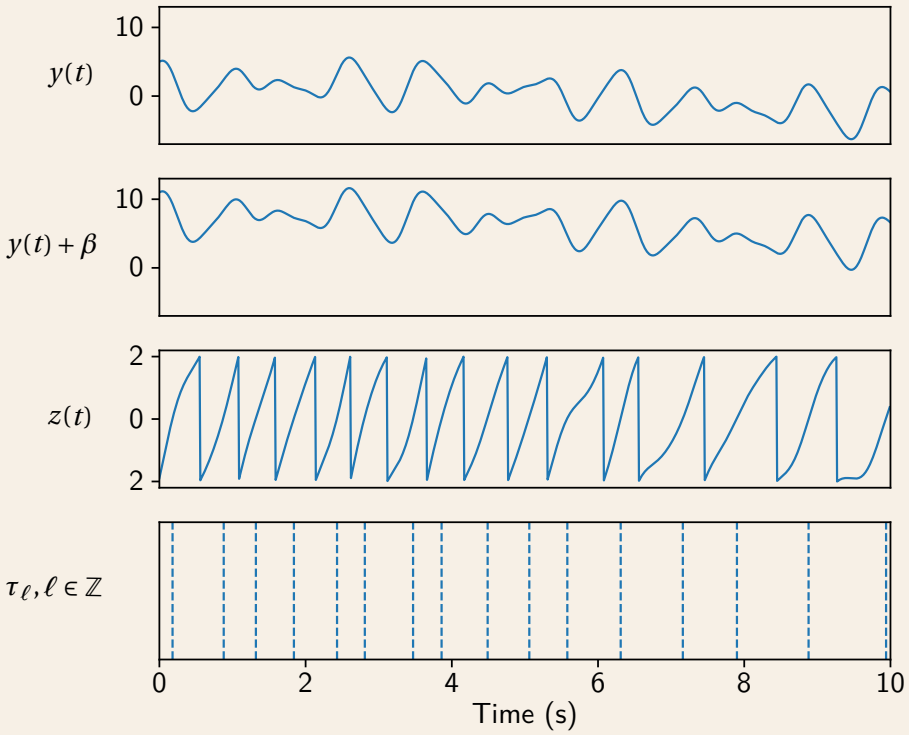



Figure 2.3 – Processing of a signal $y(t)$ as it goes through the different stages of a time encoding machine. From top to bottom, we have: the input signal $y(t)$; the result of the bias addition where $\beta$ is the bias; the result of the integration and reset; and the spike stream output where $\tau_\ell$ denotes the $\ell^{\text{th}}$ spike and $\ell \in \mathbb{Z}$. Note that, in practice and in our simulations, $\ell$ takes a finite number of values as there is a finite number of spikes.

**Further reading**

In this section, we presented only two definitions of TEMs, one of them is very general (the crossing TEM in Definition 2.1) and one of them is very specific (the integrate-and-fire TEM in Definition 2.2). However, many more models of TEMs exist.

In fact, the crossing TEM model we presented in Definition 2.1 is flexible and can allow for various filters $h(t)$ and test functions $\{\phi_\ell\}$. Some more specific models can, for example, mimick event based sensors which emit events when they see a *change* in their inputs (Gallego et al., 2019; Delbrück et al., 2010). In this case, the filter $h(t)$ would resemble a derivative operator and the $\{\phi_\ell\}$'s would simply be a constant threshold value.

On the other hand, time encoding machines (TEMs) encode inputs using *times* that are dependent on the input itself and are thus reminiscent of real neurons (a property we will focus more on in Chapter 6) or biological sensors such as photoreceptors . The integrate-and-fire TEM already follows a simple neuron model (Burkitt, 2006) but models for leaky integrate-and-fire TEMs with refractory periods and for Hodgkin-Huxley neurons have also been studied (Lazar, 2005, 2010; Thao et al., 2022).

## 2.2 Time Encoding: Reconstruction Guarantees and Algorithms

### 2.2.1 Unique Time Encoding of Bandlimited Signals

We consider a TEM as in Definition 2.2. Results on the reconstruction of a signal $y(t)$ from its time encoding were first obtained under the following assumptions:

(A1) The input signal $y(t)$ is $2\Omega$-bandlimited, i.e. the Fourier transform $Y(\omega)$ of $y(t)$ is zero for $|\omega| > \Omega$..

(A2) The input signal $y(t)$ is in $\mathscr{L}^2(\mathbb{R})$, i.e. $\int_{-\infty}^{\infty} \left|y(u)\right|^2 du < \infty$[2].

(A3) The input signal $y(t)$ is bounded by a constant $c$, i.e. $|y(t)| < c, \forall\, t \in \mathbb{R}$.

It was shown by Lazar and Tóth that such a signal $y(t)$ can be perfectly reconstructed from the samples obtained from a TEM with parameters $\kappa$, $\theta$, and $\beta$, if $\beta > c$ and

$$\Omega < \frac{\pi\left(\beta - c\right)}{2\kappa\theta}. \tag{2.1}$$

[2]Note that a signal $y(t)$ is in $L^2(a,b)$ if $\|y(t)\|_2 = \left(\int_a^b |y(u)|^2\, du\right)^{1/2} < \infty$.

The reconstruction scheme and the proof of convergence are based on two key elements:

1. The time encoding scheme is tightly related to the scheme of sampling averages, allowing the results developped for the reconstruction from averages to be used for time encoding and reconstruction (Feichtinger and Gröchenig, 1994; Aldroubi and Feichtinger, 2002), and
2. When performing time encoding, the maximal delay between two consecutive spike times is dictated by the parameters of the machine:

$$\tau_{\ell+1} - \tau_\ell < \frac{2\kappa\theta}{\beta - c}. \tag{2.2}$$

### 2.2.2 Iterative Reconstruction of Bandlimited Signals

To recover the input signal $y(t)$, the reconstruction algorithm uses the spike times $\tau_\ell$ to compute integrals of the original signal (Lazar and Tóth, 2004). In fact, the set of spike times recorded by our TEM provides linear constraints on the input signal:

$$\int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\,du = 2\kappa\theta - \beta(\tau_{\ell+1} - \tau_\ell), \tag{2.3}$$

where $\tau_\ell$ and $\tau_{\ell+1}$ are any two consecutive trigger times. Now, let $\mathscr{R}$ be an operator such that:

$$\mathscr{R}(x(t)) = \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du \quad \mathrm{sinc}_\Omega(t - s_\ell), \tag{2.4}$$

where $s_\ell = (\tau_\ell + \tau_{\ell+1})/2$ and $\mathrm{sinc}_\Omega(t) = \sin(\Omega t)/(\pi t)$.

Given this $\mathscr{R}$, one can estimate $y(t)$ iteratively by setting

$$\hat{y}^{(0)}(t) = \mathscr{R}\big(y(t)\big), \tag{2.5}$$

$$\hat{y}^{(k+1)}(t) = \hat{y}^{(k)}(t) + \mathscr{R}\Big(y(t) - \hat{y}^{(k)}(t)\Big). \tag{2.6}$$

To prove that the algorithm described in (2.4)-(2.6) converges to the right solution, one can use induction to prove that the $k^{th}$ estimate $\hat{y}^{(k)}(t)$ is a partial sum of a Neumann series: $\hat{y}^{(k)}(t) = \mathscr{R}\sum_{n=0}^{k}(\mathscr{I} - \mathscr{R})^n\, y(t)$—where $\mathscr{I}$ is the identity opertor—which converges to $\lim_{k\to\infty}\hat{y}^{(k)}(t) = \mathscr{R}\mathscr{R}^{-1}y(t) = y(t)$, if

$$\|\mathscr{I} - \mathscr{R}\| < 1. \tag{2.7}$$

Here, $\|.\|$ denotes the operator norm.

To prove convergence of the algorithm, it remains to be proven that (2.7) holds. To do so, Lazar and Tóth use the following two lemmas (Feichtinger and Gröchenig, 1994).

**Bernstein's inequality**

**Lemma 2.1.** If $y = y(t)$ is a function defined on $\mathbb{R}$ bandlimited to $[-\Omega, \Omega]$ then $dy/du$ is also bandlimited and

$$\left\| \frac{dy}{du} \right\|^2 \leq \Omega \|y\|^2.$$

**Wirtinger's inequality**

**Lemma 2.2.** If $y, dy/dt \in L^2(a,b)$ and either $y(a) = 0$ or $y(b) = 0$, then

$$\int_a^b |y(u)|^2 \, du \leq \frac{4}{\pi^2}(b-a)^2 \int_a^b \left| \frac{dy}{du} \right|^2 du.$$

Using these two lemmas, they deduce a bound for $\|\mathscr{I} - \mathscr{R}\|$.

**Iterative reconstruction operator is Bounded (Lazar and Tóth, 2004)**

**Lemma 2.3.**

$$\|\mathscr{I} - \mathscr{R}\| \leq \frac{\Omega}{\pi} \left( \sup (\tau_{\ell+1} - \tau_\ell) \right). \tag{2.8}$$

As a result, one can deduce a sufficient condition for perfect reconstruction of a bandlimited signal from its time encoding:

**Perfect reconstruction from time encoding (Lazar and Tóth, 2004)**

**Theorem 2.1.** Assume $y(t)$ is a $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$ that is bounded such that $|y(t)| \leq c$. If $y(t)$ is passed through a TEM with parameters $\kappa$, $\theta$ and $\beta$, such that $\beta > c$, and

$$\Omega < \frac{\pi(\beta - c)}{2\kappa\theta},$$

then $y(t)$ is uniquely determined by the spike times of the TEM and can be recovered by applying the algorithm from (2.4)-(2.6).

Notice that this result imposes a Nyquist-like constraint on the bandwidth: The bound in (2.1) requires a bandwidth which is inversely proportional to the separation between spike times. Reconstruction of the original signal is then very similar to the reconstruction of a bandlimited signal sampled with irregularly spaced amplitude samples (Feichtinger and Gröchenig, 1994).

### 2.2.3 Closed-form Reconstruction of Bandlimited Signals

Lazar and Tóth (2004) also obtain a closed-form matrix formulation for the above recursive algorithm. First, let $\mathscr{G}$ be the operator defined as

$$\mathscr{G}(\mathbf{b}) = \sum_{\ell \in \mathbb{Z}} b_\ell \operatorname{sinc}_\Omega(t - s_\ell),$$

where $s_\ell = (\tau_\ell + \tau_{\ell+1})/2$ and $\operatorname{sinc}_\Omega(t) = \sin(\Omega t)/(\pi t)$ as before. In addition, define $\mathbf{b}$ to be a column vector with

$$\mathbf{b} = [b_\ell]_{\ell \in \mathbb{Z}} = \left[ \int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\, du \right]_{\ell \in \mathbb{Z}}$$

and $\mathbf{H}$ to be a matrix

$$\mathbf{H} = [H_{\ell k}]_{\ell, k \in \mathbb{Z}} = \left[ \int_{\tau_\ell}^{\tau_{\ell+1}} g(u - s_k)\, du \right]_{\ell, k \in \mathbb{Z}}.$$

Then, under the conditions of Theorem 2.1, one can recover $y(t)$ by setting $y(t) = \mathscr{G}\left(\mathbf{H}^+ \mathbf{q}\right)$ where $\mathbf{H}^+$ is the pseudoinverse of $\mathbf{H}$. We refer the reader to Lazar and Tóth (2004) for a proof.

We have covered the main results established in Lazar and Tóth (2004) and now wish to reformulate the iterative reconstruction algorithm from the perspective of projections onto convex sets. We will use this perspective in Chapters 3 and 4 to design recovery algorithms when multiple time encoding machines are used.

## 2.3 Time Encoding: Recovery using Projections onto Convex Sets

We wish to reach a more intuitive interpretation of the recursive algorithm presented above, and to adapt it to new, potentially more complex scenarios. To do so, we will slightly modify the algorithm to fit a projection onto convex sets approach.

**Projection onto convex sets**

**Definition 2.4.** The *Projection Onto Convex Sets (POCS)* method searches for a point in the intersection of $N \in \mathbb{N}$ convex sets $\mathscr{C}_1, \mathscr{C}_2, \cdots, \mathscr{C}_N$ which are subsets of a Hilbert space $\mathscr{H}$. It obtains a solution $\hat{y}$, by alternately projecting on each of the convex sets $\mathscr{C}_1, \mathscr{C}_2, \cdots, \mathscr{C}_N$, using firmly nonexpansive projection operators $\mathscr{P}_1, \mathscr{P}_2, \cdots, \mathscr{P}_N$.

The POCS algorithm is known to converge to a fixed point which lies in the intersection of the sets at hand $\bigcap_{i=1}^N \mathscr{C}_i$ (Bauschke and Borwein, 1996). Thus, if the intersection of the sets

consists of a single element, then the algorithm converges to the *correct* solution.

The algorithm presented by Lazar and Tóth (Lazar and Tóth, 2004) resembles a POCS algorithm. In particular, notice that $\mathscr{R}$, defined in (2.4), can be rewritten as

$$\mathscr{R}(x(t)) = \mathscr{B}(x(t)) * \mathrm{sinc}_\Omega(t), \tag{2.9}$$

where

$$\mathscr{B}(x(t)) = \sum_{\ell\in\mathbb{Z}} \left(\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du\right) \quad \delta(t-s_\ell), \tag{2.10}$$

where $\delta(t)$ is the Dirac delta. Recursively applying $\mathscr{R}$, as in (2.5) and (2.6), therefore alternately projects onto two convex sets, the set of bandlimited functions (by convolving with a sinc) and the set of functions which match the measurements $\{\tau_\ell, \ell\in\mathbb{Z}\}$.

In fact, $\mathscr{B}$ adds Diracs in the center of each inter-spike interval[3], where the Diracs are weighted in such a way that the input and output have the same integrals between $\tau_\ell$ and $\tau_{\ell+1}$.

However, the range of operator $\mathscr{B}$ does not lie in a Hilbert space because Dirac deltas have infinite energy, so the algorithm does not meet all the technical requirements for a properly converging POCS algorithm (Thao and Rzepka, 2019). To remedy this, we assume that our input signals are in $L^2(\mathbb{R})$, and define operator $\tilde{\mathscr{B}}$ as follows:

$$\tilde{\mathscr{B}}(x(t)) = \sum_{k\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du \quad \frac{1}{\tau_{\ell+1}-\tau_\ell}\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t), \tag{2.11}$$

where $\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t)$ is a function which takes value one when $t\in[\tau_\ell,\tau_{\ell+1})$ and zero elsewhere. $\tilde{\mathscr{B}}$ and $\mathscr{B}$ produce signals that have the same integrals over intervals $[\tau_\ell,\tau_{\ell+1})$, but the result obtained from applying $\tilde{\mathscr{B}}$ is in $L^2(\mathbb{R})$, which is a Hilbert space.

Now, we define operator $\tilde{\mathscr{R}}$ as follows:

$$\tilde{\mathscr{R}}(x(t)) = \tilde{\mathscr{B}}(x(t)) * \mathrm{sinc}_\Omega(t). \tag{2.12}$$

Defining

$$\hat{y}^{(0)}(t) = \tilde{\mathscr{R}}\big(y(t)\big), \quad \hat{y}^{(k+1)}(t) = \hat{y}^{(k)}(t) + \tilde{\mathscr{R}}(y-\hat{y}^{(k)}(t)), \tag{2.13}$$

we can show that $\hat{y}^{(k)}(t)$ is $2\Omega$-bandlimited at every iteration $k$. Therefore,

$$\begin{aligned}\hat{y}^{(k+1)}(t) &= \hat{y}^{(k)}(t) * \mathrm{sinc}_\Omega(t) + \tilde{\mathscr{B}}\Big(y(t)-\hat{y}^{(k)}(t)\Big) * \mathrm{sinc}_\Omega(t),\\ &= \Big(\hat{y}^{(k)}(t) + \tilde{\mathscr{B}}\Big(y(t)-\hat{y}^{(k)}(t)\Big)\Big) * \mathrm{sinc}_\Omega(t).\end{aligned}$$

.

[3]The inter-spike intervals are the intervals $[\tau_\ell,\tau_{\ell+1}]$ between any two consecutive spikes $\tau_\ell$ and $\tau_{\ell+1}$.

Notice the similarity between the iterations of this algorithm and those of the algorithm presented in Section 2.2.2. Earlier, at each iteration, Diracs were placed between consecutive spikes to make the signal consistent with the spike times and the result was then low-pass filtered. Here, indicator functions are placed between consecutive spikes before the low-pass filter is applied.

To formalise the POCS perspective, we can divide the computation of $\hat{y}^{(k+1)}(t)$ into two steps:

$$\hat{y}^{(k+1)}(t) = \mathscr{P}_{\Omega}\left(\mathscr{P}_{\mathrm{TEM}}\left(\hat{y}^{(k)}(t)\right)\right), \tag{2.14}$$

where

$$\mathscr{P}_{\mathrm{TEM}}\left(x(t)\right) = x(t) + \tilde{\mathscr{B}}\left(y(t) - x(t)\right), \tag{2.15}$$

and

$$\mathscr{P}_{\Omega}\left(x(t)\right) = x(t) * \mathrm{sinc}_{\Omega}(t). \tag{2.16}$$

Letting $\mathscr{C}_{\Omega}$ be the space of $2\Omega$-bandlimited functions which are also in $L^2(\mathbb{R})$, we have the following two lemmas.

**Bandlimited projection is firmly nonexpansive**

**Lemma 2.4.** $\mathscr{P}_{\Omega}$ is a firmly nonexpansive projection operator onto $\mathscr{C}_{\Omega}$.

*Proof.* See Appendix 2.A. □

**Bandlimited functions set is convex**

**Lemma 2.5.** $\mathscr{C}_{\Omega}$ is convex.

*Proof.* See Appendix 2.A. □

As for $\mathscr{P}_{\mathrm{TEM}}$, we can substitute (2.11) into (2.15), yielding

$$\mathscr{P}_{\mathrm{TEM}}\left(x(t)\right) = x(t) + \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left[y(u) - x(u)\right] du \;\; \frac{\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t)}{\tau_{\ell+1} - \tau_\ell}. \tag{2.17}$$

We thus see that the operator depends on the spike times $\tau_\ell$ emitted by a TEM with input $y(t)$. $\mathscr{P}_{\mathrm{TEM}}$ uses operator $\tilde{\mathscr{B}}$ to produce a function which is consistent with the measurements $\{\tau_\ell, \ell \in \mathbb{Z}\}$. We call $\mathscr{C}_{\mathrm{TEM}}$ the space of such functions $\hat{y}(t) \in L^2(\mathbb{R})$ satisfying $\int_{\tau_\ell}^{\tau_{\ell+1}} \hat{y}(u)\,du = \int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\,du, \quad \forall \ell \in \mathbb{Z}$. These functions could generate the spike times $\tau_\ell$ when passed through the TEM.

**Time encoding projection is firmly nonexpansive**

**Lemma 2.6.** $\mathscr{P}_{\mathrm{TEM}}$ is a firmly nonexpansive projection operator onto $\mathscr{C}_{\mathrm{TEM}}$.

*Proof.* See Appendix 2.A. □

**Time encoding set is convex**

**Lemma 2.7.** $\mathscr{C}_{\mathrm{TEM}}$ is convex.

*Proof.* See Appendix 2.A. □

Since both $\mathscr{P}_{\mathrm{TEM}}$ and $\mathscr{P}_{\Omega}$ are projection operators onto $\mathscr{C}_{\mathrm{TEM}}$ and $\mathscr{C}_{\Omega}$ respectively, the entire iterative reconstruction algorithm then consists of alternately projecting onto two sets, each being convex.

We summarize our results in the following theorem.

**Perfect reconstruction from time encoding**

**Theorem 2.2.** Assume $y(t)$ is a $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$ that is bounded such that $|y(t)| \leq c$. If $y(t)$ is passed through a TEM with parameters $\kappa$, $\theta$ and $\beta$, such that $\beta > c$, and

$$\Omega < \frac{\pi(\beta - c)}{2\kappa\theta},$$

then $\lim_{k \hookrightarrow \infty} \hat{y}^{(k)}(t) = y(t)$ if $\hat{y}^{(k)}(t)$ is defined as in (2.13).

**Further reading**

Results in the last two sections are heavily based on the reconstruction from averages algorithm provided in Feichtinger and Gröchenig (1994); Sun and Zhou (2002b,a), but the POCS formulation provides intuition on the process and allows for a different approach to proving convergence.
Following the initial findings in Lazar and Tóth (2003, 2004) on bandlimited signal reconstruction, studies were also conducted to evaluate the performance of different reconstruction approaches.

Some results are a consequence of posing the problem as uniform sampling in the

amplitude domain (Florescu and Coca, 2015; Martínez-Nuevo et al., 2016; Lai et al., 2017).
Other work focuses on developing iterative algorithms that are implementable in hardware. For example, in Thao and Rzepka (2020), the authors develop a multiplierless iteration algorithm that can be implemented in hardware with bit shifts and additions/subtractions only.
Further research has also been done to explore algorithms that can perform decoding in real time, for example using a stitching algorithm (Lazar et al., 2008) or using a Kalman filter approach (Saxena and Dahleh, 2014).

Results such as those presented here were also extended to a wider range of signals, such as signals in shift-invariant subspaces (Gontier and Vetterli, 2014) and signals with finite rate of innovation as we will discover in Section 2.4.2. A more comprehensive literature review on reconstructing FRI signals from their time encoding is provided in the Further Reading box of that section.

## 2.4 Sampling at the Rate of Innovation

Historically speaking, recovery guarantees under different sampling schemes were most often first established for signals with a finite bandwidth or that lie in a shift-invariant subspace (Shannon, 1949; Unser, 2000). However, such signal classes can be restrictive in applications where certain models of sparsity are more appropriate, for example assuming that signals have a finite rate of innovation.

### 2.4.1 Finite Rate of Innovation Signals: Uniqueness Guarantees and Recovery Algorithms

**Finite-rate-of-inovation Signals**

**Definition 2.5.** We consider a class of signals $y(t)$ that can be written:

$$y(t) = \sum_{k\in\mathbb{Z}} \sum_{r=1}^{R} c_{kr}\phi_r(t - t_k), \tag{2.18}$$

where a signal is defined by $R$ known functions $\{\phi_r(t)\}_{r=1}^{R}$, and a set of corresponding arbitrary shifts $t_k$ and amplitudes $c_{kr}$. We define the rate of innovation of $y(t)$,

$$\rho = \lim_{u\to\infty} \frac{1}{u} C_y\left(-\frac{u}{2}, \frac{u}{2}\right) \tag{2.19}$$

where $C_y(t_a, t_b)$ counts the number of degrees of freedom of $y(t)$ over $[t_a, t_b]$. The signal $y(t)$ is said to have finite rate of innovation if $\rho$ is finite.

In this thesis we will focus on the canonical example of a finite-rate-of-innovation signal, the $T$-periodic stream of Diracs:

$$y(t) = \sum_{k=1}^{K} c_k \sum_{n \in \mathbb{Z}} \delta(t - t_k - nT). \tag{2.20}$$

Here the $c_k$'s denote the amplitudes of the Diracs and the $t_k$'s denote the timings of the Diracs. Therefore, recovering such a signal from its samples implies recovering the amplitudes $c_k$ and timings $t_k$ that are apriori unknown. In total, one wishes to recover $2K$ degrees of freedom.

From Vetterli et al. (2002), we know that such a signal can be written:

$$y(t) = \sum_{n \in \mathbb{Z}} \frac{1}{T} \left( \sum_{k=1}^{K} c_k e^{-\mathbf{j}(2\pi n t_k / T)} \right) e^{\mathbf{j}(2\pi n t / T)}. \tag{2.21}$$

This form makes the Fourier series (FS) representation of $y(t)$ evident. In fact, the FS coefficients $Y[m]$ take the form

$$Y[m] = \frac{1}{T} \sum_{k=1}^{K} c_k e^{-\mathbf{j}(2\pi m t_k / T)}. \tag{2.22}$$

Given the nonlinear relationship between the $Y[m]$'s and the $t_k$'s, recovering the $t_k$'s from a signal's FS coefficients is nontrivial. To perform this recovery, one can search for an annihilating filter of length $K+1$ for $Y[m]$.

**Annihilating filter**

**Definition 2.6.** Consider a filter $A[m]$, $m = 0, \cdots, K$, with $z$-transform:

$$A(z) = \sum_{m=0}^{K} A[m] z^{-m} \tag{2.23}$$

This filter is called an annihilating filter for $y(t)$ if $A[m] \neq 0$ and the result of the convolution of the two FS coefficients $A[m]$ and $Y[m]$ is zero:

$$A[m] * Y[m] = 0. \tag{2.24}$$

If $y(t)$ follows the model in (2.20), then the $z$-transform, $A(z)$, of $A[m]$ will have zeros at $u_k = e^{-\mathbf{j}(2\pi t_k/T)}$, where the $t_k$'s are the timings of the Diracs of $y(t)$.

There are now three questions to be solved. First, how can one determine a signal's FS coefficients? Second, how can one determine the corresponding annihilating filter? Third, how can one then recover the Dirac timings and amplitudes using the annihilating filter?

### How to determine a signal's FS coefficients?

The FS coefficients of a $T$-periodic signal can be obtained from its time domain samples under certain conditions. This is the case for example if the input $y(t)$ is passed through a filter that satisfies the alias cancellation property before sampling.

**Alias cancellation property**

**Definition 2.7.** A filter $h(t)$ satisfies the *alias cancellation* property, if its Fourier transform $H(\omega)$ satisfies, for $\omega_0 = 2\pi/T$ and $m \in \mathbb{Z}$,

$$H(m\omega_0) = \begin{cases} h_m \neq 0 & \text{if } m = \{-K, \cdots, K\}, \\ 0 & \text{otherwise.} \end{cases} \tag{2.25}$$

If the input signal $y(t)$, as defined in (2.20) is passed through such a filter, following which time domain samples of $y(t)$ are taken, only the FS coefficients $-K$ to $K$ influence the samples and these coefficients can be recovered by solving a linear system (provided that there are enough samples, i.e. at least $2K+1$ samples that are different from one another).

Notice that we enforce $H(\omega)$ to have $2K+1$ consecutive FS components because our input signal has $2K$ degrees of freedom, and one needs at least $2K+1$ consecutive FS coefficients of $y(t)$ to recover it using our method. More FS coefficients can also be used. However, to obtain these FS coefficients, more time-domain samples of $y(t)$ will be needed.

### How to find an annihilating filter?

An annihilating filter for $y(t)$ has to satisfy (2.24). Assuming we have recovered FS coefficients $Y[-K], \cdots, Y[K]$ of the input as described in the previous section, finding the annihilating filter $A[m]$ for $Y[m]$ amounts to fixing $A[0]$ (to 1 for example) and solving a linear set of

equations:

$$\begin{bmatrix} Y[0] & Y[-1] & \cdots & Y[-K+1] \\ Y[1] & Y[0] & \cdots & Y[-K+2] \\ & & \ddots & \\ Y[K-1] & Y[K-2] & \cdots & Y[0] \end{bmatrix} \cdot \begin{bmatrix} A[1] \\ A[2] \\ \vdots \\ A[K] \end{bmatrix} = - \begin{bmatrix} Y[1] \\ Y[2] \\ \vdots \\ Y[K] \end{bmatrix}. \tag{2.26}$$

Both the matrix that pre-multiplies the coefficients $A[m]$ and the result of the multiplication are known if one knows the FS coefficients. Therefore, $A[m]$ can be obtained exactly by pseudo-inversion.

### How to recover the Dirac timings and amplitudes using an annihilating filter?

We mentioned in Definition 2.6 that the $z$-transform, $A(z)$, of $A[m]$ will have zeros at $u_k = e^{-\mathbf{j}(2\pi t_k/T)}$, where the $t_k$'s are the timings of the Diracs of $y(t)$. Therefore, if one has access to the FS coefficients $A[m]$, one can describe its $z$-transform $A(z)$ as a polynomial with coefficients $A[m]$ as done in (2.23) and can therefore factor it into its roots. Once the roots $u_k$ are obtained, the timings $t_k$ can easily be recovered.

Then, knowing the $t_k$'s, the $c_k$'s can be recovered by solving this linear system:

$$\begin{bmatrix} Y[0] \\ Y[1] \\ \vdots \\ Y[K] \end{bmatrix} = \frac{1}{T} \begin{bmatrix} 1 & 1 & 1 & \\ e^{-\mathbf{j}(2\pi t_1/T)} & e^{-\mathbf{j}(2\pi t_2/T)} & \dots & e^{-\mathbf{j}(2\pi t_K/T)} \\ & & \ddots & \\ e^{-\mathbf{j}(2K\pi t_1/T)} & e^{-\mathbf{j}(2K\pi t_2/T)} & \dots & e^{-\mathbf{j}(2K\pi t_K/T)} \end{bmatrix} \cdot \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_K \end{bmatrix}, \tag{2.27}$$

where only the $c_k$'s are unknown.

### Putting it all together

In sum, the recovery of a $T$-periodic signal $y(t)$ with $2K$ degrees of freedom as in (2.20) can be achieved by filtering $y(t)$ using a filter that satisfies the alias cancellation property such that all but FS coefficients $Y[-K], \cdots, Y[K]$ are cancelled out, then obtaining $2K+1$ time-domain samples of $y(t)$ and recovering the FS coefficients $Y[-K], \cdots, Y[K]$. Once these are obtained, one can find an annihilating filter $A[m], m = 0, \cdots, K$ for $Y[m]$, $m = -K, \cdots, K$ and recover the Dirac times by factoring the $z$-transform $A(z)$ into roots (a nonlinear process) and the Dirac amplitudes $c_k$ by solving a linear system.

**Further reading**

The literature on sampling signals with finite rate of innovation is much more wide-reaching than what we have covered here and has applications in, for example, the compression of ECG signals (Baechler et al., 2013) and the recovery of star positions in the sky (Pan et al., 2017).
We refer readers interested in the theory behind FRI sampling to the first paper on this topic (Vetterli et al., 2002) and to a subsequent article that deals with wider classes of filters (Dragotti et al., 2007).
The algorithms were later extended to deal with noisy scenarios by applying Cadzow denoising (Cadzow, 1988) and with multi-dimensional signals such as sampling curves with finite rate of innovation or sampling FRI signals that lie on a sphere (Pan et al., 2013, 2017).

### 2.4.2 Recovering FRI Signals from their Time Encoding

Until now, we assumed that uniform (time, amplitude) samples of the filtered input signal are taken to recover the $2K$ degrees of freedom underlying the input $y(t)$ as defined in (2.20). Here, we are interested in recovering this $y(t)$ from its time encoding. To do so, we see that our input $y(t)$ has $2K$ degrees of freedom we would like to recover, and we pass it through a filter $h(t)$ that satisifies the alias cancellation property as defined in Definition 2.7, thus preserving $2K+1$ FS coefficients of $y(t)$, $Y[-K],\cdots,Y[K]$. Then, the filtered signal $y(t)*h(t)$ is passed through a TEM.

As previously explained, the timings at the output of the TEM provide linear constraints on the input signal. We therefore combine the knowledge of the integrals (2.3) and the FS expression for $y(t)$ to obtain a linear system in terms of the FS coefficients $Y[k]$. If we know $L$ spike times $\tau_\ell$, $\ell=0,\cdots,L-1$, then

$$\begin{bmatrix} b_0 \\ b_1 \\ \vdots \\ b_{L-2} \end{bmatrix} = \begin{bmatrix} \int_{\tau_0}^{\tau_1} y(u)\,du \\ \int_{\tau_1}^{\tau_2} y(u)\,du \\ \vdots \\ \int_{\tau_{L-2}}^{\tau_{L-1}} y(u)\,du \end{bmatrix} = \mathbf{G}_{IF}. \begin{bmatrix} Y[-K]\times H[-K]/-\mathbf{j}2K\pi \\ Y[-K+1]\times H[-K+1]/-\mathbf{j}2(K-1)\pi \\ \vdots \\ Y[0]\times H[0] \\ \vdots \\ Y[K-1]\times H[K-1]/\mathbf{j}2(K-1)\pi \\ Y[K]\times H[K]/\mathbf{j}2K\pi \end{bmatrix}, \tag{2.28}$$

where

$$\mathbf{G}_{IF}^{T} = \begin{bmatrix} e^{-\mathrm{j}2K\pi\tau_1} - e^{-\mathrm{j}2K\pi\tau_0} & e^{-\mathrm{j}2K\pi\tau_2} - e^{-\mathrm{j}2K\pi\tau_1} & \cdots & e^{-\mathrm{j}2K\pi\tau_{L-1}} - e^{-\mathrm{j}2K\pi\tau_{L-2}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{-\mathrm{j}2\pi\tau_1} - e^{-\mathrm{j}2\pi\tau_0} & e^{-\mathrm{j}2\pi\tau_2} - e^{-\mathrm{j}2\pi\tau_1} & & e^{-\mathrm{j}2\pi\tau_{L-1}} - e^{-\mathrm{j}2\pi\tau_{L-2}} \\ \tau_1 - \tau_0 & \tau_2 - \tau_1 & \cdots & \tau_{L+1} - \tau_L \\ e^{\mathrm{j}2\pi\tau_1} - e^{\mathrm{j}2\pi\tau_0} & e^{\mathrm{j}2\pi\tau_2} - e^{\mathrm{j}2\pi\tau_1} & \cdots & e^{\mathrm{j}2\pi\tau_{L-1}} - e^{\mathrm{j}2\pi\tau_{L-2}} \\ \vdots & \vdots & \ddots & \vdots \\ e^{\mathrm{j}2K\pi\tau_1} - e^{\mathrm{j}2K\pi\tau_0} & e^{\mathrm{j}2K\pi\tau_2} - e^{\mathrm{j}2K\pi\tau_1} & \cdots & e^{\mathrm{j}2K\pi\tau_{L-1}} - e^{\mathrm{j}2K\pi\tau_{L-2}} \end{bmatrix}. \tag{2.29}$$

Notice that we expanded the *transpose* of $\mathbf{G}_{IF}$ because of space limitations.

According to (Kamath et al., 2021), an FRI signal as in (2.20) can be recovered from its time encoding given enough spikes at the output of the TEM:

**Uniqueness of a FRI signal from its time encoding (Kamath et al., 2021)**

**Lemma 2.8.** The matrix $\mathbf{G}_{IF}$ defined in (2.29) has full column-rank whenever $L \geq 2K+2$.

This means if a TEM emits $2K+2$ spike times for an input $y(t)$, then this time encoding of $y(t)$ uniquely represents the FS coefficients of its input because the matrix $\mathbf{G}_{IF}$ is full rank. One can then proceed, as in the previous section, with finding the annihilating filter for these FS coefficients and finally the Dirac timings and weights.

**Further reading**

We have elaborated one approach to recovering FRI signals from their time encodings—an approach which uses a filter that satisfies an alias cancellation property and which is performed offline. Further work has been done to tackle scenarios where streams of Diracs are filtered using exponential and polynomial splines and where the reconstruction is done in a sequential fashion (Alexandru and Dragotti, 2019). Extensions were also done to understand how to reconstruct FRI signals from their time encoding after filtering with a hyperbolic secant kernel or an alpha-synaptic function, that is, filters that have more biological motivation (Hilton et al., 2021b,a).

## 2.5 Low Rank Matrix Factorization

### 2.5.1 Problem Formulation

In the problem of low-rank factorization, one wishes to recover a matrix $\mathbf{M} \in \mathbb{R}^{N_1 \times N_2}$ with rank $r << \min(N_1, N_2)$ that satisfies a set of available observations $\mathbf{y} \in \mathbb{R}^n$ about $\mathbf{M}$. In other words, one assumes that the solution $\mathbf{M}$ can be written as a product of two matrices $\mathbf{L}\mathbf{R}^T$, $\mathbf{L} \in \mathbb{R}^{N_1 \times r}, \mathbf{R} \in \mathbb{R}^{N_2 \times r}$.

Notice that, even in situations where $\mathbf{M}$ is uniquely defined by the rank and measurements provided, the matrices $\mathbf{L}$ and $\mathbf{R}$ are never unique beyond invertible transformations, i.e. if $\mathbf{M}$ can be decomposed into $\mathbf{M} = \mathbf{L}\mathbf{R}^T$, then it can also be decomposed into $\mathbf{M} = \tilde{\mathbf{L}}\tilde{\mathbf{R}}^T$ where $\tilde{\mathbf{L}} = \mathbf{L}\mathbf{H}$ and $\tilde{\mathbf{R}} = \mathbf{R}\left(\mathbf{H}^T\right)^{-1}$ where $\mathbf{H}$ is an invertible matrix.

In more precise terms, one would ideally like to recover $\mathbf{M}$ by minimizing its rank, while enforcing the constraints from $\mathbf{y} \in \mathbb{R}^n$ obtained from applying operator a linear operator $\mathscr{S}$ to $\mathbf{M}$ and measuring the result in $b$:

$$\begin{aligned} &\text{minimize} && \text{rank}(\mathbf{M}) \\ &\text{subject to} && \mathscr{S}(\mathbf{M}) = b. \end{aligned} \tag{2.30}$$

This optimization problem is non-convex and NP-hard to solve. It has been shown that the nuclear norm is the tightest convex lower bound of the rank function (Recht et al., 2010). Therefore, the above optimization problem is often relaxed to the following:

$$\begin{aligned} &\text{minimize} && ||\mathbf{M}||_* \\ &\text{subject to} && \mathscr{S}(\mathbf{M}) = b, \end{aligned} \tag{2.31}$$

where $||\mathbf{M}||_*$ is the nuclear norm of $\mathbf{M}$, i.e. the sum of the singular values of $\mathbf{M}$. This problem can be reformulated as a semi-definite problem and solved using interior-point methods (Fazel, 2003). However, this can become intractable as the matrix dimensions become high.

### 2.5.2 The Singular Value Projection Algorithm

Different algorithms were developed to more efficiently solve the problem in (2.31). We focus on the Singular Value Projection (SVP) algorithm which was developed to recover matrix $\mathbf{M}$ from measurements $\mathbf{y} = \mathscr{S}(\mathbf{M})$ (Jain et al., 2010).

The Singular Value Projection (SVP) algorithm alternately applies the low-rank constraint and the measurement constraint on the matrix of interest $\mathbf{M}$ and is detailed in Algorithm 1, where we let $\mathbf{M}^{(k)}$ be the estimate at iteration $k$ of the target matrix to reconstruct and $\mathbf{P}^{(k)}$ be a proxy matrix to perform the iterations.

The SVP algorithm is based on projected gradient descent, and has been proven to converge

**Algorithm 1** Singular Value Projection

**Input:** $\mathscr{S}$, $b$, tolerance $\epsilon$, $\eta_k$ for $k = 0, 1, 2...$

$\mathbf{M}^{(0)} = 0$ and $k = 0$
**repeat**
  $\mathbf{P}^{k+1} \leftarrow \mathbf{M}^{(k)} - \eta_k \mathscr{S}^T(\mathscr{S}(\mathbf{M}^{(k)}) - b)$
  Compute top $r$ singular vectors of $\mathbf{P}^{(k+1)} : U_r, \Sigma_r, V_r$
  $\mathbf{M}^{(k+1)} \leftarrow U_r \Sigma_r V_r^T$
  $k \leftarrow k + 1$
**until** $\|\mathscr{S}(\mathbf{M}^{(k+1)} - b)\|_2^2 \leq \epsilon$

to the correct solution in case where the operator $\mathscr{S}$ satisfies the restricted isometry property.

**Restricted isometry property**

**Definition 2.8.** An operator $\mathscr{S}$ satisfies the *restricted isometry property* (RIP) if $\exists$ an isometry constant $\delta_r \in (0,1)$ s.t. $\forall \mathbf{M} \in \mathbb{R}^{N_1 \times N_2}$ of rank at most $r$,

$$(1 - \delta_r)\|\mathbf{M}\|_2^2 \leq \|\mathscr{S}(\mathbf{M})\|_2^2 \leq (1 + \delta_r)\|\mathbf{M}\|_2^2. \quad (2.32)$$

Then, the operator $\mathscr{S}$ is said to satisfy the $r$-restricted isometry property with restricted isometry constant $\delta_r$.

The Singular Value Projection algorithm will be useful when we tackle time encoding of signals with an unknown low dimensional structure in Chapter 4.

With this, we have now covered the main technical foundations needed to tackle the results in this part of the thesis and can now proceed to explore new results.

## 2.A Appendix: Proofs for POCS

**Bandlimited projection is firmly nonexpansive**

**Lemma 2.4.** $\mathscr{P}_\Omega$ is a firmly nonexpansive projection operator onto $\mathscr{C}_\Omega$.

*Proof of Lemma 2.4.* First, we show that $\mathscr{P}_\Omega$ is idempotent:

$$\mathscr{P}_\Omega(\mathscr{P}_\Omega(x(t))) = x(t) * \mathrm{sinc}_\Omega(t) * \mathrm{sinc}_\Omega(t) = x(t) * \mathrm{sinc}_\Omega(t) = \mathscr{P}_\Omega(x(t)),$$

since a $\mathrm{sinc}_\Omega$ convolved with itself is still a $\mathrm{sinc}_\Omega$ (given that its Fourier transform has value one for frequencies $\omega$ such that $|\omega| < \Omega$ and zero otherwise.).

We now show that $\mathscr{P}_\Omega$ has as range the space $\mathscr{C}_\Omega$ of $2\Omega$-bandlimited functions in $L^2(\mathbb{R})$.

First, let $x(t)$ be an arbitrary $L^2(\mathbb{R})$ function, its Fourier transform $X(\omega)$ is then also $L^2(\mathbb{R})$, according to Parseval's theorem. The result of the projection will have Fourier transform equal to $Y(\omega), \forall\,|\omega| < \Omega$, and zero otherwise. Therefore $\mathscr{P}_\Omega(x(t))$ is $2\Omega$-bandlimited and $\in L^2(\mathbb{R})$, and is therefore in $\mathscr{C}_\Omega$.

Now, let $x(t)$ be a $2\Omega$-bandlimited function $\in L^2(\mathbb{R})$, then its Fourier transform $X(\omega)$ is such that $X(\omega) = 0, \forall|\omega| > \Omega$. Convolving $x(t)$ with $\mathrm{sinc}_\Omega(t)$ in the time domain only multiplies $X(\omega)$ by 1 in the region where it is nonzero. Therefore $x(t) * \mathrm{sinc}_\Omega(t) = x(t) \in \mathscr{C}_\Omega$.

Finally, $\mathscr{P}_\Omega$ is firmly non-expansive, since it is an orthogonal projection operator Vetterli et al. (2014). □

**Bandlimited functions set is convex**

**Lemma 2.5.** $\mathscr{C}_\Omega$ is convex.

*Proof of Lemma 2.5.* Let $x_1(t)$ and $x_2(t)$ be in $\mathscr{C}_\Omega$. Then let $x_3(t)$ be a convex combination of $x_1(t)$ and $x_2(t)$, i.e. $x_3(t) = \lambda x_1(t) + (1-\lambda)x_2(t)$, where $\lambda \in [0,1]$. Then, let $X_1(\omega)$, $X_2(\omega)$ and $X_3(\omega)$ be the respective Fourier transforms of $x_1(t)$, $x_2(t)$ and $x_3(t)$. By linearity of the Fourier transform, we find that $X_3(\omega) = \lambda X_1(\omega) + (1-\lambda)X_2(\omega)$. Therefore, since $x_1(t)$ and $x_2(t)$ are in $\mathscr{C}_\Omega$ and $X_1(\omega) = X_2(\omega) = 0\ \forall|\omega| > \Omega$, $X_3(\omega) = 0\ \forall|\omega| > \Omega$. Therefore, $x_3(t)$ is also $2\Omega$-bandlimited. $L^2(\mathbb{R})$ is also a convex set (as it is a linear space), therefore $x_3(t)$ is also in $L^2(\mathbb{R})$, as $x_1(t)$ and $x_2(t) \in L^2(\mathbb{R})$. Therefore $x_3(t) \in \mathscr{C}_\Omega$, thus showing that $\mathscr{C}_\Omega$ is convex. □

**Time encoding projection is firmly nonexpansive**

**Lemma 2.6.** $\mathscr{P}_{\mathrm{TEM}}$ is a firmly nonexpansive projection operator onto $\mathscr{C}_{\mathrm{TEM}}$.

*Proof of Lemma 2.6.* First we show that $\mathscr{P}_{\mathrm{TEM}}$ is idempotent. Note that $\int_{\tau_\ell}^{\tau_{\ell+1}} \mathscr{P}_{\mathrm{TEM}}(x(u)) = \int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\,du,\ \forall \ell \in \mathbb{Z}$, where $y(t)$ is the input to the TEM. Therefore,

$$\begin{aligned}\mathscr{P}_{\mathrm{TEM}}(\mathscr{P}_{\mathrm{TEM}}(x(t))) &= \mathscr{P}_{\mathrm{TEM}}(x(t)) + \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left[y(u) - \mathscr{P}_{\mathrm{TEM}}(x(u))\right] du\ \frac{\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t)}{\tau_{\ell+1}-\tau_\ell} \\ &= \mathscr{P}_{\mathrm{TEM}}(x(t)).\end{aligned}$$

Now we show that the range of $\mathscr{P}_{\mathrm{TEM}}$ is indeed the space of functions $f(t)$ with $\int_{\tau_\ell}^{\tau_{\ell+1}} f(u)\,du = \int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\,du$.

First let $x(t)$ be a function in $L^2(\mathbb{R})$. It is easy to show that $\mathscr{P}_{\text{TEM}}(x(t))$ will have $\int_{\tau_\ell}^{\tau_{\ell+1}} \mathscr{P}_{\text{TEM}}(x(u))\,du = \int_{\tau_\ell}^{\tau_{\ell+1}} y(u)\,du$. One can also show that $\mathscr{P}_{\text{TEM}}(x(t))$ will be in $L^2(\mathbb{R})$, by using Lemma 2.13.

Now, let $x(t)$ be in $\mathscr{C}_{\text{TEM}}$, then

$$\begin{aligned}\mathscr{P}_{\text{TEM}}(x(t)) &= x(t) + \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left[y(u) - x(u)\right] du \quad \frac{\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t)}{\tau_{\ell+1}-\tau_\ell} \\ &= x(t) + \sum_{\ell\in\mathbb{Z}} 0 \times \frac{\mathbb{1}_{[\tau_\ell,\tau_{\ell+1})}(t)}{\tau_{\ell+1}-\tau_\ell} \\ &= x(t).\end{aligned}$$

Therefore, $\mathscr{P}_{\text{TEM}}$ has range $\mathscr{C}_{\text{TEM}}$.

It remains to be shown that $\mathscr{P}_{\text{TEM}}$ is firmly nonexpansive. To do so, it is sufficient to show that $\mathscr{P}_{\text{TEM}}$ can be written

$$\mathscr{P}_{\text{TEM}} = \frac{1}{2}\mathscr{I} + \frac{1}{2}\mathscr{N},$$

where $\mathscr{I}$ is the identity operator and $\mathscr{N}$ is an nonexpansive operator. Indeed, $\mathscr{P}_{\text{TEM}}$ can be written as such if we set

$$\mathscr{N}(x(t)) = \mathscr{I}(x(t)) + 2\tilde{\mathscr{B}}\left(y(t) - x(t)\right).$$

We want to show that $\mathscr{N}$ is nonexpansive, therefore it is sufficient to show that, for any $x_1(t)$ and $x_2(t)$ in $L^2(\mathbb{R})$,

$$||\mathscr{N}(x_1) - \mathscr{N}(x_2)|| \le ||x_1 - x_2||.$$

We will start with the left hand side of the equation:

$$\begin{aligned}||\mathscr{N}(x_1) - \mathscr{N}(x_2)|| &= \Big|\Big|\mathscr{I}(x_1(t)) + 2\tilde{\mathscr{B}}\left(y(t) - x_1(t)\right) \\ &\quad - \mathscr{I}(x_2(t)) - 2\tilde{\mathscr{B}}\left(y(t) - x_2(t)\right)\Big|\Big| \\ &= \Big|\Big|\mathscr{I}(x_1) - 2\tilde{\mathscr{B}}(x_1) - \mathscr{I}(x_2)(t) + 2\tilde{\mathscr{B}}(x_2)(t)\Big|\Big| \\ &= \Big|\Big|\left(\mathscr{I} - 2\tilde{\mathscr{B}}\right)(x_1 - x_2)\Big|\Big| \\ &\le \Big|\Big|\mathscr{I} - 2\tilde{\mathscr{B}}\Big|\Big|\,||x_1 - x_2|| \\ &\le ||x_1 - x_2||,\end{aligned}$$

where we use the fact that $\|\mathscr{I} - 2\tilde{\mathscr{B}}\| = 1$, as we show in Lemma 2.14, below. □

**Time encoding set is convex**

**Lemma 2.7.** $\mathscr{C}_{\text{TEM}}$ is convex.

*Proof of Lemma 2.7.* Let $x_1(t)$ and $x_2(t)$ be in $\mathscr{C}_{\text{TEM}}$. Then let $x_3(t)$ be any convex combi-

nation of $x_1(t)$ and $x_2(t)$, i.e. $x_3(t) = \lambda x_1(t) + (1-\lambda)x_2(t)$, where $\lambda \in [0,1]$. Then,

$$\begin{aligned}\int_{\tau_\ell}^{\tau_{\ell+1}} x_3(t)\,dt &= \int_{\tau_\ell}^{\tau_{\ell+1}} \lambda x_1(t) + (1-\lambda)x_2(t)\,dt \\ &= \lambda \int_{\tau_\ell}^{\tau_{\ell+1}} x_1(t)\,dt + (1-\lambda)\int_{\tau_\ell}^{\tau_{\ell+1}} x_2(t)\,dt \\ &= \lambda \int_{\tau_\ell}^{\tau_{\ell+1}} y(t)\,dt + (1-\lambda)\int_{\tau_\ell}^{\tau_{\ell+1}} y(t)\,dt \\ &= \int_{\tau_\ell}^{\tau_{\ell+1}} y(t)\,dt.\end{aligned}$$

The first equality holds because of the definition of $x_3(t)$, and the third equality holds because $x_1(t)$ and $x_2(t)$ are in $\mathscr{C}_{\text{TEM}}$. The result shows that $x_3(t)$ is also consistent with the spike times $\{\tau_\ell, \ell \in \mathbb{Z}\}$. On the other hand, $L^2(\mathbb{R})$ is a linear space (and therefore a convex set), so $x_3(t) \in L^2(\mathbb{R})$ as well. Therefore, $x_3(t) \in \mathscr{C}_{\text{TEM}}$, thus proving that $\mathscr{C}_{\text{TEM}}$ is a convex set. □

### $\tilde{\mathscr{B}}$ projects onto a subspace of $L^2(\mathbb{R})$

**Lemma 2.13.** If $x(t)$ is in $L^2(\mathbb{R})$, then $\tilde{\mathscr{B}}(x(t))$ is also in $L^2(\mathbb{R})$.

*Proof.* Let $x(t) \in L^2(\mathbb{R})$, so $\int_{-\infty}^{\infty} |x(t)|^2 = d < \infty$ for some $d \in \mathbb{R}$.

$$\begin{aligned}\int_{-\infty}^{\infty} \left|\tilde{\mathscr{B}}(x(t))\right|^2 dt &= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left|\tilde{\mathscr{B}}(x(t))\right|^2 dt \\ &= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left|\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}\right|^2 dt \\ &= \sum_{\ell\in\mathbb{Z}} (\tau_{\ell+1}-\tau_\ell)\left|\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}\right|^2 \\ &= \sum_{\ell\in\mathbb{Z}} \frac{\left(\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du\right)^2}{\tau_{\ell+1}-\tau_\ell} \\ &\overset{(a)}{\leq} \sum_{\ell\in\mathbb{Z}} (\tau_{\ell+1}-\tau_\ell)\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} (x(u))^2\,du}{\tau_{\ell+1}-\tau_\ell} \\ &= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} (x(u))^2\,du \\ &= \int_{-\infty}^{\infty} (x(u))^2\,du = d < \infty.\end{aligned}$$

Here, inequality (a) arises from the Cauchy-Schwarz inequality. □

**Operator norm of $\mathscr{I}-2\tilde{\mathscr{B}}$ is one**

**Lemma 2.14.** The operator norm of operator $\mathscr{I}-2\tilde{\mathscr{B}}$, $\left|\left|\mathscr{I}-2\tilde{\mathscr{B}}\right|\right|$ is equal 1.

*Proof.* To find the operator norm $\left|\left|\mathscr{I}-2\tilde{\mathscr{B}}\right|\right| = \sup_x \frac{\|(\mathscr{I}-2\tilde{\mathscr{B}})(x)\|}{\|x\|}$, let us compute, for any $x(t)$,

$$\begin{aligned}
\left|\left|\left(\mathscr{I}-2\tilde{\mathscr{B}}\right)(x(t))\right|\right|^2 &= \int_{-\infty}^{\infty} \left|x(t)-2\tilde{\mathscr{B}}\,(x(t))\right|^2 dt \\
&= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left(x(t)-2\tilde{\mathscr{B}}\,(x(t))\right)^2 dt \\
&= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} \left(x(t)-2\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}\right)^2 dt \\
&= \sum_{\ell\in\mathbb{Z}} \left(\int_{\tau_\ell}^{\tau_{\ell+1}} (x(t))^2 + 4\left(\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}\right)^2 - 4\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}x(t)\,dt\right) \\
&= \sum_{\ell\in\mathbb{Z}} \left(4\,(\tau_{\ell+1}-\tau_\ell)\left(\frac{\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du}{\tau_{\ell+1}-\tau_\ell}\right)^2 - 4\frac{\left(\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du\right)^2}{\tau_{\ell+1}-\tau_\ell} + \int_{\tau_\ell}^{\tau_{\ell+1}} (x(t))^2\,dt\right)
\end{aligned} \tag{2.33}$$

$$\begin{aligned}
&= \sum_{\ell\in\mathbb{Z}} \left(4\frac{\left(\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du\right)^2}{\tau_{\ell+1}-\tau_\ell} - 4\frac{\left(\int_{\tau_\ell}^{\tau_{\ell+1}} x(u)\,du\right)^2}{\tau_{\ell+1}-\tau_\ell} + \int_{\tau_\ell}^{\tau_{\ell+1}} (x(t))^2\,dt\right) \\
&= \sum_{\ell\in\mathbb{Z}} \int_{\tau_\ell}^{\tau_{\ell+1}} (x(t))^2\,dt \\
&= \int_{-\infty}^{\infty} (x(t))^2\,dt = \|x(t)\|^2.
\end{aligned}$$

Therefore, $\|(\mathscr{I}-\tilde{\mathscr{B}})\,(x(t))\| = \|\,(x(t))\|$ for any $x(t)$. Thus, $\|\mathscr{I}-\tilde{\mathscr{B}}\| = 1$. □

# 3 Time Encoding a Single Signal using Multiple Channels

***Recommended Reading:*** Sections 2.1, 2.2, 2.3.
*The content in this chapter is adapted from Adam et al. (2019, 2020b).*

## 3.1 Introduction

We begin our journey of exploring the wonders of spikes and time encoding by focusing on the ease of clock-alignment and, therefore, multi-channel sampling using time encoding, and show that time encoding machines, hold an advantage over classical uniform sampling in this scenario.

In the classical uniform case, when sampling a signal using multiple samplers, one requires a shift between the clocks of the samplers to capture more information about the input than in the single-sampler case. However, if this shift is unknown, *recovering* the input from the multichannel samples becomes a more complex procedure (Asl et al., 2010).

On the other hand, when one samples a signal using multiple TEMs, one requires a shift between the *integrators* to capture more information about a signal, as we will see later. In this case, even if this shift is unknown, reconstruction guarantees are established for signals with a bandwidth that is $N_c$ times larger than in the single channel case and the reconstruction algorithm can be as simple as the one in the single channel case.

In fact, we will show that reconstructing a $2\Omega$-bandlimited signal from its $N_c$-channel time encoding is possible, whenever the bandwidth satisfies

$$\Omega < \frac{N_c \pi (\beta - c)}{\kappa \theta}, \tag{3.1}$$

where $\theta$, $\kappa$ and $\beta$ are the machine parameters, whatever the (potentially unknown) integrator shifts between the machines

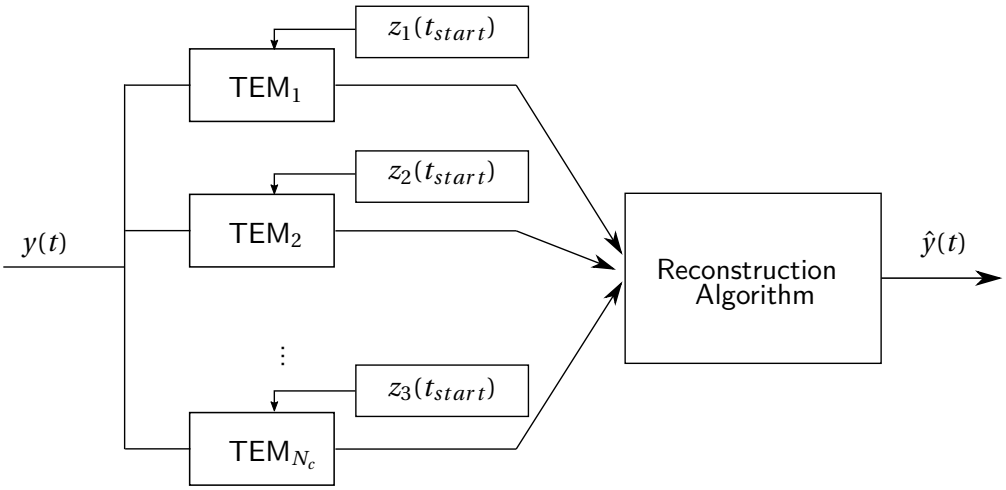



Figure 3.1 – Multi-channel time encoding and decoding pipeline. In Practice, the TEMs are initialized with some initial value of their integrators $z_i(t_{start})$ and the integrator shift between two machines TEM$_i$ and TEM$_{i+1}$ is $\Delta z_i = z_{i+1}(t_{start}) - z_i(t_{start}) \mod 2\theta$, for $1 < i < N_c$ (the shift between machines TEM$_{N_c}$ and TEM$_1$ is $\Delta z_{Nc} = z_1(t_{start}) - z_{N_c}(t_{start}) \mod 2\theta$). The output streams of the different machines can be combined into one before being fed into a single decoding machine because of the perfect ordering of the spikes provided in (3.4).

Practically, clock synchronization, which poses an issue in multi-channel amplitude sampling, can be entirely bypassed here because TEMs encode information in the timing of the spikes that are output. These outputs can then be summed (in hardware) into one final spike train, to find the relative positioning of spikes across channels, as we will see in Fig. 3.4.

These results allow more flexibility in device designs. To encode signals that are more information-rich, rather than design TEMs that have higher spiking rate (thus introducing more margin for noise), one can use multiple TEMs, allowing each TEM to have a lower spiking rate.

We will first show that an $N_c$-channel TEM, as depicted in Fig. 3.1 uniquely encodes a signal with a bandwidth $N_c$ times larger than in the single channel case. We then provide an iterative and a one-shot reconstruction algorithm which perform perfect reconstruction of the input and we finally provide simulations to evaluate the performance of our algorithms with varying number of machines, varying integrator shifts and varying noise levels.

## 3.2 Background

As we previously mentioned, time encoding can mimic sensory information processing in neuroscience. Therefore, an intuitive extension to the time encoding machine introduced in 2.1 is a system consisting of multiple time encoding machines: human sensory systems are

comprised of many neurons that encode inputs using spikes, which are later used in higher order processes in the brain.

Moreover, neurons in sensory systems are modeled with receptive fields. This means that certain neurons are sensitive to certain shapes of inputs, and the spiking output of these neurons is essentially driven by filtered versions of the original input signal (Hubel and Wiesel, 1962). Therefore, different neurons spike at different times and therefore encode different sets of information.

Inspired by such experimental findings, Lazar and Pnevmatikakis (2008) defined a setup with $N_c$ linearly independent filters and $N_c$ leaky integrate-and-fire time encoding machines. A 1-dimensional signal $y(t)$ is then fed into filter $i$ before being input to $\text{TEM}_i$, for $i = 1 \cdots N_c$, then reconstructed from the samples. Within this setup, the authors were able to quantify the improvement one obtains from the multi-channel encoding and decoding setup.

In the present chapter, our approach to time encoding is different: we assume that we are dealing with multiple *similar* neurons encoding the same input, i.e. they all respond to the same kinds of stimuli.

Therefore, we assume that our signal is not prefiltered before being input to each machine or neuron, but that machines output different spike times because of different initial configurations of the time encoding machines or neurons. In Chapter 4, we extend the results of this chapter to time encoding and decoding of vectors of inputs where the connection between the inputs is more complex. This extension mimics the way neurons automatically form receptive fields: these fields arise naturally because of the structure of the connection between input and neuron.

The setup presented here also draws a parallel with the multi-channel sampling setup in the classical sampling scenario, where sampling devices have unknown shifts in their clocks. Here, our time encoding machines will have unknown shifts in their *integrators*. The former problem seems to be quite difficult to solve, whereas the latter seems no harder to solve than the single-channel variant.

## 3.3 $N_c$-Channel TEM Definition

First, we refer the reader once more to the definition of a TEM in Definition 2.2 and to its circuit in Fig. 2.2 and define integrator shifts between TEMs with the same parameters.

**Integrator shifts**

**Definition 3.1.** $N_c$ TEMs with parameters $\kappa$, $\theta$ and $\beta$ have *integrator shifts* $\Delta z_1, \Delta z_2, \cdots, \Delta z_{N_c}$ if, for the same input $y(t)$, and for any time $t$, the outputs of the

integrators $z_1(t), z_1(t), \cdots, z_{N_c}(t)$ satisfy

$$z_{i+1}(t) = (z_i(t) + \Delta z_i) \mod 2\theta, \quad i = 1 \cdots N_c - 1 \tag{3.2}$$

$$z_1(t) = (z_{N_c}(t) + \Delta z_{N_c}) \mod 2\theta. \tag{3.3}$$

Here, the $\Delta z_i$'s naturally satisfy [a] $\left(\sum_i \Delta z_i\right) \mod 2\theta = 0$.

[a] This arises from recursively expanding the expression of $z_1(t)$ in (3.3) and noting that $z_1(t) = z_1(t) + \sum_i \Delta z_i \mod 2\theta$.

Consider two time encoding machines $\text{TEM}_1$ and $\text{TEM}_2$, with the same parameters $\kappa$, $\theta$, and $\beta$. At first sight, it seems that $\text{TEM}_1$ and $\text{TEM}_2$ will output the same encoding of an input signal $y(t)$. However, consider a scenario where the integrator of $\text{TEM}_1$ is always $\Delta z \neq 0$ ahead of the integrator of $\text{TEM}_2$ (modulo $2\theta$). Then, the threshold is reached at different times in the two machines. Therefore, the recorded spike times of $y(t)$ are different and the overall encoding of the signal is different.

Intuition tells us that we can probably gain more information about $y(t)$ by considering the outputs of both machines.

More generally, our multi-channel time encoding setup assumes that a bandlimited signal is passed through an $N_c$-channel time encoding machine, as defined next.

### $N_c$-channel time encoding machine

**Definition 3.2.** An *$N_c$-channel time encoding machine* consists of $N_c$ single-channel TEMs $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$, $\text{TEM}_{N_c}$, with parameters $\kappa$, $\theta$ and $\beta$ and integrator shifts $\Delta z_1, \cdots, \Delta z_{N_c}$.

When these shifts are all nonzero, the machines will spike in this order $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$, $\text{TEM}_{N_c}$, $\text{TEM}_1$, i.e.

$$\tau_{1,\ell} < \tau_{2,\ell} < \cdots < \tau_{N_c,\ell} < \tau_{1,\ell+1} \qquad \forall \ell \in \mathbb{Z}, \tag{3.4}$$

where $\{\tau_{i,\ell}, \ell \in \mathbb{Z}\}$ is the set of spike times emitted by $\text{TEM}_i$.

Equation (3.4) forces a strict order of the spike times on the $N_c$ machines, which naturally arises from nonzero shifts between the integrators of the machines.

**Two-channel time encoding**

**Example 3.1.**

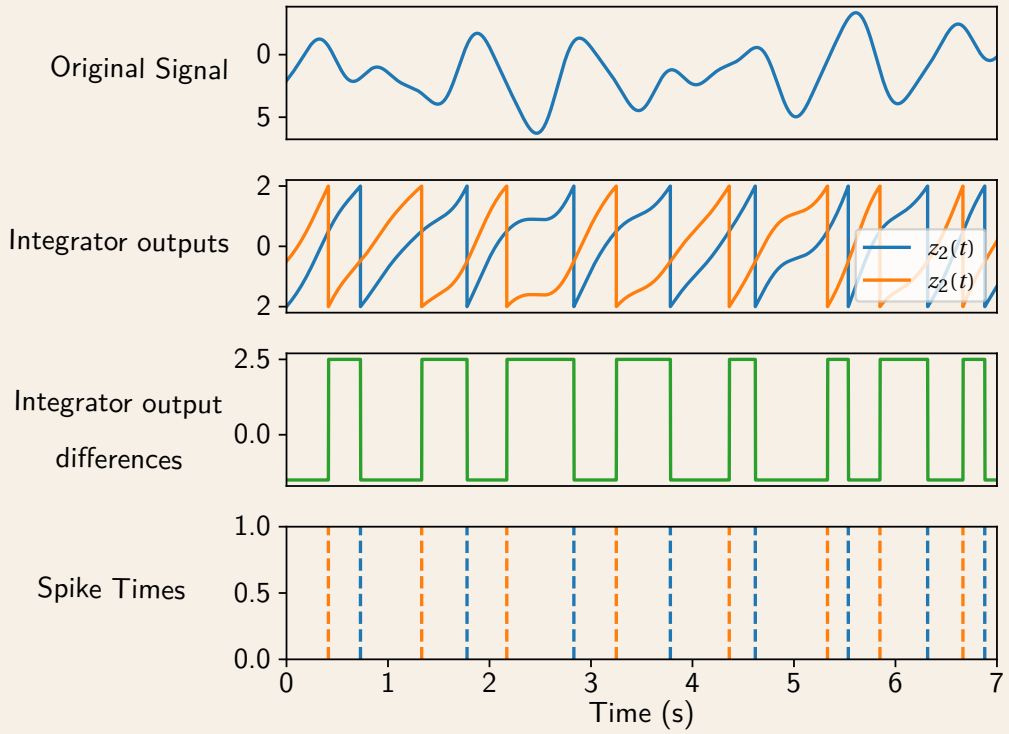



Figure 3.2 – Output of the integrators of two TEMs with nonzero shifts. We assume both TEMs have a threshold $\theta = 2$ and that $\text{TEM}_2$ is leading $\text{TEM}_1$ by $\Delta z_1 = 0.75$. This means $z_2(t) = z_1(t) + \Delta z_1 \mod 2\theta$, $\forall t$. We plot, from top to bottom: The original signal input to the machines, the output $z_i(t)$ of the integrator of each machine $\text{TEM}_i$, the difference between the integrators of the two machines, and the output spikes of each machine. Note how the spike times are interleaved, i.e. there is always one spike of $\text{TEM}_1$ between any two spikes of $\text{TEM}_2$ and vice versa.

Figure 3.2 shows an example of 2-channel time encoding. We pass an input signal through the two TEMs (with nonzero integrator shifts) and record the output of each integrator. Notice how the integrator values are always shifted by the same amount (modulo $2\theta$). Therefore, as the spike times are generated at the integrator reset, the TEMs are guaranteed to spike at different times so that $\tau_{i,k} \neq \tau_{j,\ell} \quad \forall k, \ell \in \mathbb{Z}, \quad \forall i \neq j$, where $i, j \in [1, \cdots N_c]$. Moreover, the spike times are interleaved, satisfying (3.4).

We have yet to explain where the integrator shifts come from, in practice. In short, they come from different initial conditions of the integrators as we explain in the “In depth" box below.

**In depth: How integrator shifts occur**

So far, we have assumed that input signals have infinite support, and that each TEM samples its inputs for infinite time. In practice, however, a TEM would start recording a signal at a certain time $t_{\text{start}}$ and stop recording at $t_{\text{end}}$. In these scenarios, integrator shifts can be well defined and implemented.
Indeed, these integrator shifts will result from different initial conditions on the integrators of the TEMs at $t_{\text{start}}$. For example, assume $\text{TEM}_1$ and $\text{TEM}_2$ start integrating at the same time $t_{\text{start}}$ with initial values $z_1(t_{\text{start}})$ and $z_2(t_{\text{start}})$, respectively. Then $\text{TEM}_2$ will always lead $\text{TEM}_1$ (modulo $2\theta$)by $\Delta z_1 = z_2(t_{\text{start}}) - z_1(t_{\text{start}})$.
More practically, an integrator can be represented in circuitry by an operational amplifier coupled with a resistor and capacitor, as seen in Fig. 3.3. This capacitor can be charged with a certain voltage before the input is fed into the circuit. This initial charge of the capacitor can practically implement the initial value of the integrator. Therefore, having different initial charges on the capacitors of each machine would lead to nonzero integrator shifts.

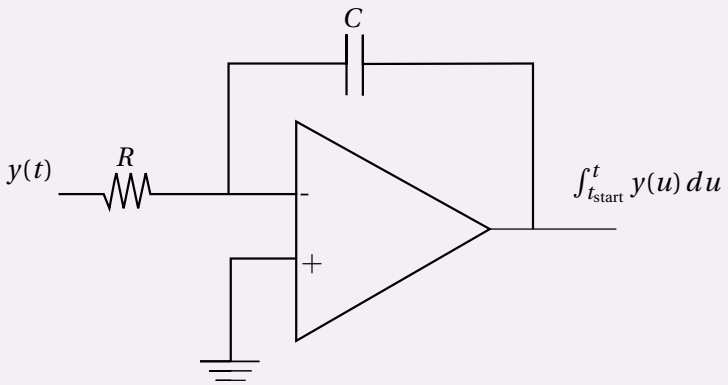



Figure 3.3 – The circuit of a simple integrator comprises of an operational amplifier, a resistor $R$ and a capacitor $C$ in the shown configuration. The circuit does not provide a perfect integrator as we require in our model but it serves a good approximation of it and allows the implementation of our setup in hardware. An analysis of time encoding using leaky integrators such as this one is presented in Lazar and Pnevmatikakis (2008).

However, we recall that our setup assumes a perfect integrator and infinite time support. The initial conditions formulation in this section serves as a more intuitive explanation of how integrator shifts arise, and as a practical explanation of where these shifts come from, in hardware.

## 3.4 Uniqueness of $N_c$-Channel Reconstruction using POCS

Our findings are summarized into the following theorem.

**Unique definition of a bandlimited signal from its multi-channel time encoding**

**Theorem 3.1.** Assume $y(t)$ is a $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$ that is bounded such that $|y(t)| \leq c$ and that has a well-defined integral $\int_{-\infty}^{t} y(u)\,du =: Y(t) < \infty$. If $y(t)$ is passed through $N_c$ TEMs with parameters $\kappa$, $\theta$ and $\beta$, such that $\beta > c$, the shifts $\Delta z_i$, $i = 1 \cdots N_c$ between the TEMs are nonzero and

$$\Omega < \frac{N_c \pi (\beta - c)}{2\kappa\theta}, \tag{3.5}$$

then $y(t)$ is uniquely determined by the spike times of the TEMs.

*Proof.* The proof is included in Appendix 3.B. □

The above theorem states that using $N_c$ time encoding machines can uniquely encode a signal $y(t)$ with a bandwidth which is $N_c$ times larger than in the single channel case, no matter how the shifts between the machines are configured, as long as they are all nonzero.

## 3.5 Convergence of $N_c$-Channel Reconstruction using POCS

We have found sufficient conditions for a signal to be uniquely defined by its $N_c$-channel time encoding. We now develop algorithms that perform perfect reconstruction from the spike times of $N_c$-channel TEMs.

First, we use the projection onto convex sets (POCS) formulation to devise a reconstruction algorithm for multi-channel time encoding.

The POCS method, as defined in Definition 2.4 can guarantee convergence onto a fixed point by alternately projecting onto convex sets, as explained in Definition 2.4. The averaged projection method works similarly.

**The averaged projections method**

**Definition 3.3.** The *averaged projections method* assumes that we have $N$ convex sets $\mathscr{C}_1, \cdots, \mathscr{C}_N$ with corresponding projection operators $\mathscr{P}_1, \cdots, \mathscr{P}_N$ and that we compute an estimate of $y$ at iteration $k+1$ by taking

$$\hat{y}^{(k+1)} = \frac{1}{N} \sum_{i=1}^{N} \mathscr{P}_i \left( \hat{y}^{(k)} \right). \tag{3.6}$$

This algorithm can be reduced into an alternating projection algorithm and therefore also

converges to a fixed point in the intersection of the sets $\mathscr{C}_i$.

To apply the POCS approach, we first need to identify the convex sets and the corresponding projection operators in the reconstruction problem.

We recall that $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$, $\text{TEM}_{N_c}$ are our $N_c$ time encoding machines, and $\{\tau_{i,\ell}\ell \in \mathbb{Z}\}$ are the spike times emitted by machine $i$, $i = 1 \cdots N_c$, when the input is $y(t)$—a $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$ such that $|y(t)| < c$, for some $c \in \mathbb{R}$.

Then, we follow a similar treatment to the one in Section 2.3 and let $\tilde{\mathscr{R}}_i$ be the reconstruction operator associated with $\text{TEM}_i$, such that

$$\tilde{\mathscr{R}}_i\left(x(t)\right) = \sum_{k\in\mathbb{Z}} \left( \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} x(u)\, du \right) \frac{\mathbb{1}_{[\tau_{i,\ell},\tau_{i,\ell+1})}(t)}{\tau_{i,\ell+1} - \tau_{i,\ell}} * \text{sinc}_\Omega\left(t\right), \tag{3.7}$$

where $\text{sinc}_\Omega(t) = \sin\left(\Omega t\right)/\left(\pi t\right)$.

Note that each of these reconstruction operators $\tilde{\mathscr{R}}_i$ is the same as the operator defined in (2.12) for each of the individual machines $\text{TEM}_i$ and their respective spike times $\tau_{i,\ell}$, and therefore also consists of two projections onto convex sets $\mathscr{C}_{\text{TEM}_i}$ and $\mathscr{C}_\Omega$, where $\mathscr{C}_{\text{TEM}_i}$ is the set of signals that satisfy the measurements of $\text{TEM}_i$ and $\mathscr{C}_\Omega$ is the set of $2\Omega$-bandlimited signals. Each of these sets are convex by Lemma 2.5 and Lemma 2.7.

Then, we define a new reconstruction operator

$$\tilde{\mathscr{R}}_{1\cdots N_c} = \frac{1}{N_c} \sum_{i=1}^{N_c}, \tilde{\mathscr{R}}_i \tag{3.8}$$

and recursively estimate $y(t)$ by setting

$$\hat{y}^{(0)}(t) = \tilde{\mathscr{R}}_{1\cdots N_c}\left(y(t)\right), \qquad \hat{y}^{(k+1)}(t) = \hat{y}^{(k)}(t) + \tilde{\mathscr{R}}_{1\cdots N_c}\left(y(t) - \hat{y}^{(k)}(t)\right). \tag{3.9}$$

This recursive algorithm defined in (3.9) is equivalent to taking alternating projections on the sets $\mathscr{C}_{\text{TEM}_i}$ and $\mathscr{C}_\Omega$.

The POCS algorithm is guaranteed to converge to the intersection of the convex sets onto which projections are performed. Given that this intersection is unique, as Theorem 3.1 states, the POCS algorithm we detailed here converges to the correct solution. Note that this algorithm does not require knowing the shifts $\Delta z_i$ between the integrators of the machines, it only requires knowing the parameters $\kappa$, $\theta$ and $\beta$ of the machines.

The strength of this algorithm lies in its simplicity. We have $N_c$ TEMs with integrators that are shifted with respect to each other by some shifts $\Delta z_i$, and if the set of $\Delta z_i$'s changes, the spike outputs of the machines change. However, this algorithm does not require knowledge of

the shifts, it only operates on the spike times generated by the machine. Moreover, explicit labeling of spike times according to the machine they come from is not necessary. TEMs are shifted with respect to each other by $\Delta z_i$, so the order of spiking of the machines is fixed: we will always have spikes coming from $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$,$\text{TEM}_{N_c}$, $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$. Therefore, the algorithm operates on a model as depicted in Fig. 3.1, and is still able to disentangle spike streams.

**In depth: Clock alignment in time encoding is easy...**

As explained in Definition 2.3, the output of a time encoding machine $\text{TEM}_i$ can take two forms. It is either a stream of spikes or Dirac deltas that are emitted at the times $\tau_{i,\ell}$, or it is simply a (in practice, quantized) list of spike times $\tau_{i,\ell}$. This list can only be obtained from the spike stream, and if the spike times of two streams are registered separately, issues with clock alignment will occur. In other words, one will know how far apart the spikes of the first output stream occur with respect to each other, but one will not know how far apart the spikes of one output stream are with respect to those of the other output stream.

To solve this, one easy solution is to combine the two output streams (as they are being output) into one and record the spike times into lists using this single stream as a reference.

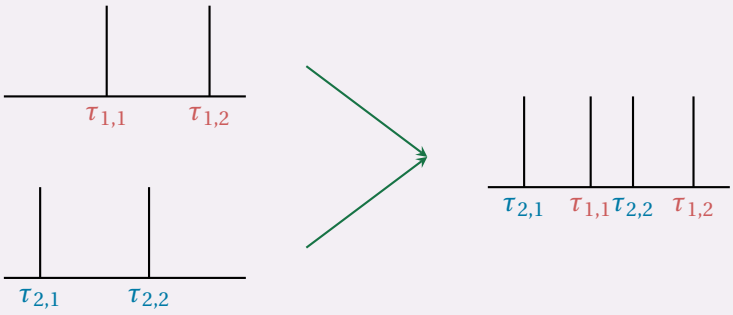



Figure 3.4 – If two streams of spikes or Diracs are added together, and the spikes occur at *different* times, the distances between spike times of different machines can be obtained by examining the sum of the two streams.

Further note that in our scenario, TEMs are assumed to have the same parameters. As a consequence, in the noiseless case, spikes always occur in the same order, as explained in (3.4), making the spike time recovery task easier. If this same order is not guaranteed (because of different parameters or noise), the summed spike stream can be interpreted together with the original output streams to recover the TEM-specific spike times.

## 3.6 Closed Form Solution

We have described an iterative algorithm to reconstruct an input signal from the output of an $N_c$-channel TEM. However, adopting a POCS algorithm in practice might be quite slow, and the performance is dependent on the number of iterations or the stopping criteria. A more in-depth analysis is presented in Thao and Rzepka (2020). Therefore, we further propose an equivalent closed form solution to the problem.

First, let $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ denote the set of combined and ordered spike times from all machines $\text{TEM}_1$, $\cdots$, $\text{TEM}_{N_c}$. Now define the operator $\tilde{\mathscr{G}}$ on a vector $\mathbf{v}$:

$$\tilde{\mathscr{G}}(\mathbf{v}) = \sum_{\ell \in \mathbb{Z}} v_\ell \tilde{g}_{[\tilde{\tau}_\ell, \tilde{\tau}_{\ell+N_c})}(t), \tag{3.10}$$

where $\tilde{g}_{[\tilde{\tau}_\ell, \tilde{\tau}_{\ell+N_c})}(t) = \mathbb{1}_{[\tilde{\tau}_\ell, \tilde{\tau}_{\ell+N_c})}(t) * \text{sinc}_\Omega(t)$. Also define

$$\tilde{\mathbf{q}} = \left[\int_{\tilde{\tau}_\ell}^{\tilde{\tau}_{\ell+N_c}} y(u)\,du\right]_{\ell \in \mathbb{Z}}, \qquad \tilde{\mathbf{H}} = \left[\tilde{H}_{\ell,j}\right]_{\ell,j\in\mathbb{Z}} = \left[\int_{\tilde{\tau}_\ell}^{\tilde{\tau}_{\ell+N_c}} \tilde{g}_{[\tilde{\tau}_j, \tilde{\tau}_{j+N_c})}(u)\,du\right]_{\ell,j\in\mathbb{Z}}. \tag{3.11}$$

Then, one can show by induction that $\hat{y}^{(k)}$, as defined in (3.8) - (3.9), can be expressed as

$$\hat{y}^{(k)}(t) = \tilde{\mathscr{G}}\left(\sum_{m=0}^{k} \left(\mathbf{I} - \tilde{\mathbf{H}}\right)^m \tilde{\mathbf{q}}\right), \tag{3.12}$$

where $\tilde{\mathscr{G}}$, $\tilde{\mathbf{H}}$ and $\tilde{\mathbf{q}}$ are known. Now we note that $\lim_{k\to\infty} \sum_{m=0}^{k} \left(\mathbf{I} - \tilde{\mathbf{H}}\right)^m = \tilde{\mathbf{H}}^+$, where $\tilde{\mathbf{H}}^+$ denotes the pseudo-inverse of $\tilde{\mathbf{H}}$. Therefore, one can reconstruct $y(t)$ in closed-form under the same conditions as the ones posed in Theorem 3.1 by setting

$$\hat{y}(t) = \tilde{\mathscr{G}}\left(\tilde{\mathbf{H}}^+ \tilde{\mathbf{q}}\right). \tag{3.13}$$

We summarize the previous two results in a Corollary.

**Perfect recovery of bandlimited signals from their multichannel time encoding**

**Corollary 3.2.** Assume $y(t)$ is a $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$ that is bounded such that $|y(t)| \le c$ and that has a well-defined integral $\int_{-\infty}^{t} y(u)\,du =: Y(t) < \infty$. If $y(t)$ is passed through $N_c$ TEMs with parameters $\kappa$, $\theta$ and $\beta$, such that $\beta > c$, the shifts $\Delta z_i$, $i = 1 \cdots N_c$, between the TEMs are nonzero and

$$\Omega < \frac{N_c \pi (\beta - c)}{2\kappa\theta}, \tag{3.14}$$

then $y(t)$ can be recovered either as $y(t) = \lim_{k\to\infty} \hat{y}^{(k)}(t)$ where $\hat{y}^{(k)}(t)$ is as defined in the recursive algorithm in (3.9) or as $y(t) = \hat{y}(t)$ as defined in the closed-form algorithm in (3.13).

.

**Reconstructing a bandlimited signal from its two-Channel time encoding**

**Example 3.2.** We show, in Fig. 3.5, a reconstruction example demonstrating that the algorithm we suggested for the $N_c$-channel case can reconstruct a wider range of signals than is possible in the single channel case.

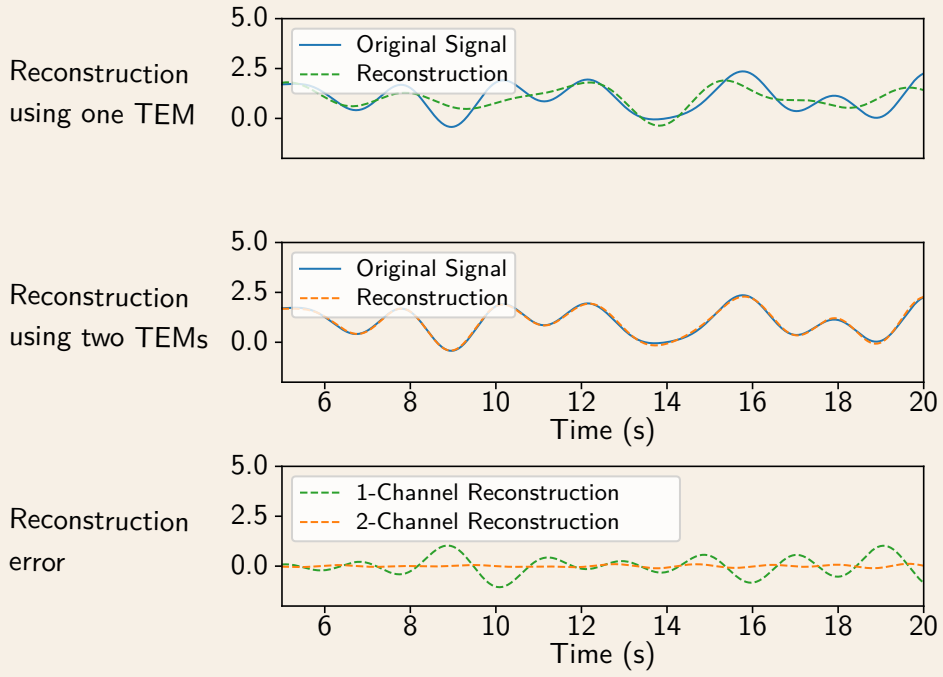



Figure 3.5 – (Top) Reconstruction of a signal from its time encoding, using one channel. (Middle) Reconstruction of the same signal from its time encoding using two channels with integrators shifted by an unknown value. (Bottom) Reconstruction error when using outputs of 1-channel TEM and 2-channel TEM.

## 3.7 Simulations

### 3.7.1 Simulation Setup

To validate our theory, we evaluate the reconstruction algorithm's success while varying four main variables in different combinations: the bandwidth of the inputs $\Omega$, the number of machines $N_c$, the shifts $\Delta z_i$ and the variance of the noise added on top of the spike times.

The parameters of the TEMs are always kept constant, taking $\kappa = \theta = 1$ and $\beta = \max_t |y(t)| + 1$ where $y(t)$ is the input signal. In fact, when evaluating the algorithm performance, we can keep these parameters constant without loss of generality as long as we vary $\Omega$ as we explain

in Section 3.A.2 of the appendix.

The implementation details can be found in Section 3.A.1 of the appendix and the figures are reproducible using code available online (Adam, 2019).

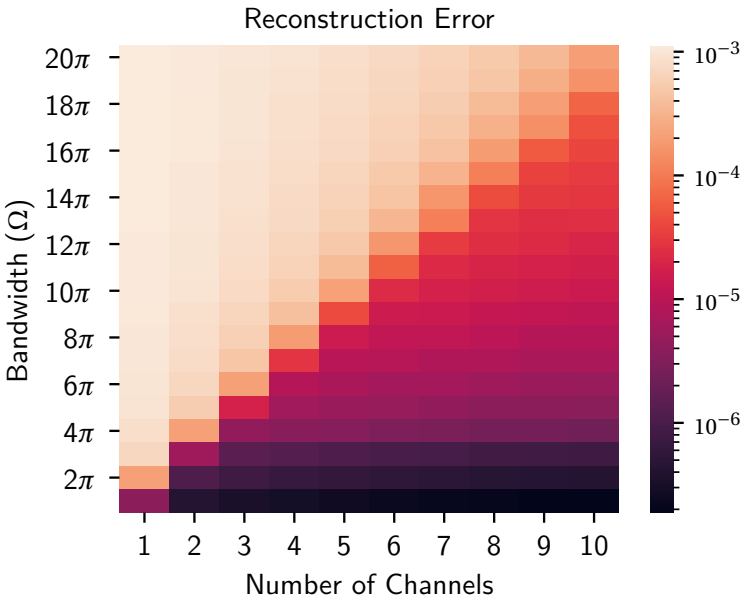



Figure 3.6 – Error of time encoding reconstruction when $N_c = 1 \cdots 10$ channels with equally-shifted integrators encode a signal as its bandwidth varies. The mean-squared error is averaged over one hundred trials and plotted as a function of the bandwidth and the number of samples.

### 3.7.2 Experimental Validation of Theorem 3.1

In Fig. 3.6, we randomly generate one hundred $2\Omega$-bandlimited signals, for each value of $\Omega = \pi, 2\pi, \cdots, 20\pi$. We provide the reconstruction error when using $N_c = 1 \cdots 10$ channels with the same parameters $\kappa$, $\theta$ and $\beta$ to sample (simulated using discrete time encoding) and reconstruct the signals. The channels are constructed with equally spaced shifts, i.e. for an $N_c$-channel TEM, the integrator shifts are $\Delta z_i = 2\theta / N_c$, $\forall i = 1 \cdots N_c$.

For every number of channels $N_c$ used to perform the signal sampling and reconstruction, we have a different constraint on the bandwidth which ensures that this $N_c$-channel TEM can reconstruct its input signal as given in Theorem 3.1. Looking at Fig. 3.6, for each number of channels $N_c$, we can see a degradation of the reconstruction as the bandwidth increases beyond the constraint placed in Theorem 3.1. Notice how this degradation happens for higher $\Omega$ as the number of channels $N_c$ increases. The separation between "good" and "bad" performance seems to change linearly with the number of channels $N_c$ as we would expect from Theorem 3.1.

To show that this $N_c$-fold improvement on the bound for the bandwidth is independent of the value of the shift, we evaluate the reconstruction error when signals are sampled using 2-channel TEMs with different values for the shift (simulated using discrete time encoding). In Fig. 3.7, we again simulate one hundred $2\Omega$-bandlimited signals where $\Omega$ now varies between $\pi/4$ and $15\pi$, and plot the averaged reconstruction error for 2-channel decoding, as well as the averaged reconstruction error for single-channel decoding. For both the single-channel and the 2-channel case, the reconstruction error is low for low values of $\Omega$ and becomes much higher as $\Omega$ surpasses the bound provided in Theorem 3.1 and plotted using the dashed vertical lines in Fig. 3.7. Notice how the reconstruction is successful for wider ranges of the bandwidth in the 2-channel case, compared to the single-channel case, and how different values of the integrator shifts between the two channels do not affect this region of success.

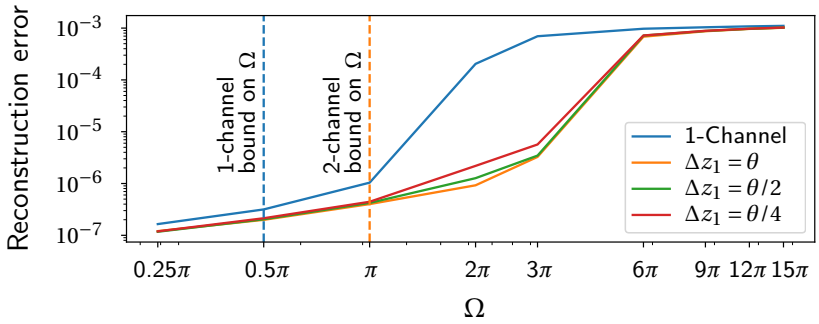



Figure 3.7 – Error of time encoding reconstruction, when using a single channel (blue), and when using 2 channels with different spacing configurations (orange, green, red). The mean-squared error is averaged over two hundred trials and plotted as a function of the bandwidth. The shift takes value $\Delta z_1$.

### 3.7.3 Problem Ill-Conditioning for Small Shifts

We have shown that, in theory, the condition we placed in Theorem 3.1 is sufficient for the reconstruction algorithm to converge no matter the shifts between the integrators of different machines. Moreover, Fig. 3.7 verified this result for a few values of the integrator shifts. Intuitively, however, the problem should become more ill-posed as the shifts approach zero.

To investigate this, we evaluate the performance of two-channel time encoding (using continuous time encoding) and decoding as the shifts between the channels approach zero. We randomly generate one hundred $2\Omega$-bandlimited signals, where $\Omega$ varies between $\pi/4$ and $8\pi$. These signals are then encoded and decoded using two-channel TEMs with fixed parameters $\kappa$, $\theta$ and $\beta$ and with varying shift $\Delta z_1$. We then estimate the reconstruction success by computing the reconstruction error.

Figure 3.8 is essentially a two dimensional version of the plot in Fig. 3.7 which investigates smaller shifts. As one of the integrator shifts approaches zero, the outputs of the two channels

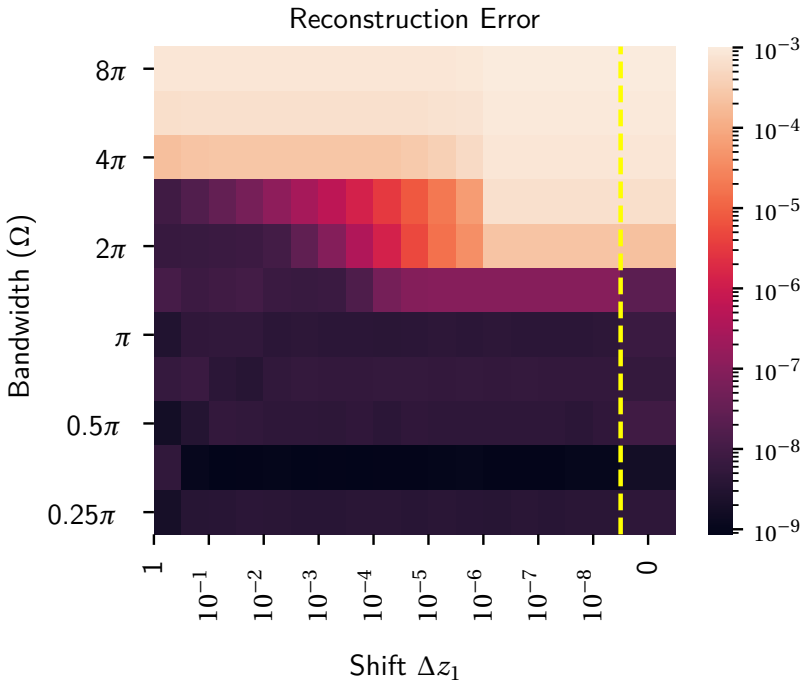



Figure 3.8 – Reconstruction error plotted as a function of bandwidth and integrator shift. A hundred $2\Omega$-bandlimited signals, where $\Omega$ varies between $\pi/4$ and $8\pi$ are generated and subsequently sampled and reconstructed using two-channel time encoding and decoding. The TEM used has fixed parameters $\kappa = 1$, $\theta = 1$ and $\beta = \max_t |y(t)| + 1$ but variable integrator shifts. Here, we plot one of the two shifts $\Delta z_1$, the value of the second shift can be obtained by solving $\Delta z_2 = 2\theta - \Delta z_1$. The mean-squared error is averaged over the hundred randomly generated signals and plotted as a function of bandwidth and shift. Although we have shown that the value of the integrator shifts should have no effect on the reconstructible bandwidth Theorem 3.1 very small shifts perform less well than shifts that are within the same order of magnitude as the threshold $\theta$. The rightmost column, separated by the dashed yellow line, shows the reconstruction error when a shift of zero is used. In other words, this is the reconstruction error when using single-channel time encoding and decoding.

of the TEM start to resemble each other more and more, so our two-channel encoding starts to resemble single-channel encoding. Therefore, we also include, in Fig. 3.8 the reconstruction error of a single-channel TEM, to compare it to the result obtained with two-channel encoding with very small shift.

Note that the system seems to perform reasonably well in the noiseless case, when the condition in Theorem 3.1 is satisfied, for shifts that are not too small (greater than $10^{-4}$).

Therefore, if the shifts are randomly assigned, they will, with very high probability fall in a regime where the algorithm provides good performance.

As for the physical implementation of the shifts, as previously mentioned, the hardware

implementation of time encoding machines uses a capacitor which can hold some initial charge. One can make sure that different machines have different initial charges on their capacitors by first feeding the TEMs with different signals to randomly initialize the values of the capacitor before beginning to encode the signal of interest.

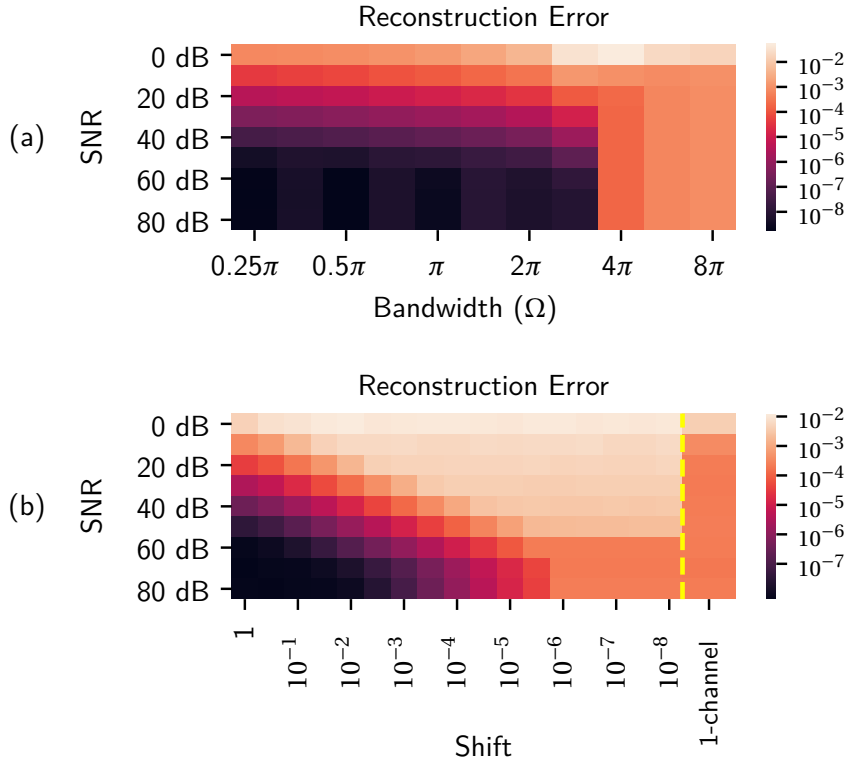



Figure 3.9 – (a) Reconstruction error of two-channel time encoding with equal spacing in the shifts, with gaussian noise added to the spike times, when the SNR varies between 80 dB and 0 dB and the bandwidth varies between $0.25\pi$ and $8\pi$. (b) To the left of the yellow line, reconstruction error of two-channel time encoding of $2\Omega$-bandlimited signals with $\Omega = 2\pi$, with gaussian noise added to the spike times, when the SNR varies between 80 dB and 0 dB and the integrator shifts vary between 1 (equal spacing) and $10^{-8}$. To the right of the yellow line, reconstruction error of single-channel time encoding of signals with the same bandwidth.

### 3.7.4 Algorithm Performance in Noisy Settings

We now provide basic analyses to understand the system's performance in the case of noise. We study the effect of noise on reconstruction when varying two other parameters: the bandwidth (Fig. 3.9.a) and the shift (Fig. 3.9.b). In both scenarios, we assume that we have a two-channel TEM with parameters $\kappa$, $\theta$ and $\beta$ fixed, such that the TEM is guaranteed to be able to reconstruct $2\Omega$-bandlimited signals with $\Omega = \pi$.

In Fig. 3.9.a, we assume that the two machines are equally spaced ($\Delta z_1 = \Delta z_2$) and that

Gaussian noise is added to the spike times. We then vary the SNR between 80 dB and 0 dB and the bandwidth between $0.25\pi$ and $8\pi$. Notice how, with high SNR, i.e. with low noise, the machines can reconstruct signals with bandwidths that go up to $2\pi$. As the SNR becomes lower, the machines become less apt at reconstructing signals with high bandwidths.

In Fig. 3.9.b, we assume that the input signals to the machines are $2\Omega$-bandlimited where $\Omega = 2\pi$. We then vary the SNR between 80 dB and 0 dB and the integrator shifts of the machines between $1$ (equal spacing) and $10^{-8}$. Notice how the reconstruction of the input becomes worse as one of the shifts between the machines approaches zero and as the SNR decreases. We also provide the reconstruction error for the single-channel time encoding as a comparison.

## 3.8 Conclusion

We have studied multi-channel time encoding of $2\Omega$-bandlimited signals, we have proposed an algorithm for reconstructing an input signal from its samples, and have provided sufficient conditions on $\Omega$ for the algorithm to converge to the correct solution. We have shown that if one TEM can perfectly encode a $2\Omega$-bandlimited signal, then $N_c$ TEMs with the same parameters and with shifts in their integrators can perfectly encode a $2N_c\Omega$-bandlimited signal. This reconstruction algorithm is based on a projection onto convex sets method.

The improvement in the condition on the bandwidth that we have found is independent of the value of the shifts between the machine integrators, as long as these shifts are nonzero. We have also shown that the knowledge of the relative shifts between the machines is not necessary for reconstruction to be possible. This is not the case in similar setups of multi-channel encoding in the classical sampling scenario where an unknown shift makes the inverse problem more difficult to solve.

Our setting has focused on reconstructing signals using TEMs with the same parameters $\kappa$, $\theta$ and $\beta$, where $\beta > 0$. As a more general rule, a $2\Omega$-bandlimited signal can still be reconstructed, if $\Omega$ is proportional to the rate of linearly independent constraints that arise from the spike times generated by the machines. In the next chapters, we will provide conditions that guarantee that a signal is unique and recoverable from its time encoding by examining the number of linearly independent constraints arising from spike times.

## 3.A Appendix: Simulation Details

### 3.A.1 Implementation

1. *Signal generation:* We wish to generate signals that are $2\Omega$-bandlimited, and sample them over a finite time window $[t_{\text{start}}, t_{\text{end}}]$. We assume the signals to be a linear combination of sincs centered at uniform time points between $t_{\text{start}}$ and $t_{\text{end}}$ with a

separation of $\pi/\Omega$. The amplitudes of these sincs are randomly generated assuming a uniform distribution over $[0,1]$, and the signals are finally all normalized to have unit $L^2$ norm.

2. *Signal sampling:* Time encoding is performed using either of two techniques. In the first "discrete time encoding" approach, we sample the signal discretely using small steps and approximate the integral of the signal with a cumulative sum. In the second "continuous time encoding" technique, signals are assumed to be generated according to our signal generation procedure described above, and we search for each spike time using binary search, by evaluating the signal's integral at different time points. On one hand, the first technique is more versatile to different signal types. On the other hand, the second technique allows more spike time precision without requiring extra space requirements which arise from heavily oversampling the input signal.

3. *Signal reconstruction:* Signal reconstruction is performed using the closed form solution provided in (3.13). The reconstruction can also be done using the iterative POCS algorithm, but obtaining the reconstruction then becomes more time consuming and the reconstruction's performance depends on the chosen number of iterations or the stopping criterion.

4. *Performance evaluation:* To evaluate the performance of our reconstruction, we compute the difference between our (discretized) reconstruction and the original (discretized) signal. We then compute the power of this difference for the middle $90\%$ of the signal (assuming the start and end generally have a less precise reconstruction because of our finite support sampling and reconstruction setup). We call this the mean-squared reconstruction error. As all signals are normalized to have unit norm, the mean-squared reconstruction error of different signals are comparable.

### 3.A.2 Parameter Variation Design

We would like to understand why it is enough to fix the threshold $\theta$, integrator constant $\kappa$ and $\beta = c+1$ and vary $\Omega$ for a fair view on performace.

Assume we have a $2\Omega$-bandlimited signal $y_1(t)$ that is sampled using parameters $\kappa$, $\theta$ and bias $\beta$ and generates spike times $\{\tau_\ell(y_1), \ell \in \mathbb{Z}\}$. Then, let $y_2(t)$ be a $2p\Omega$-bandlimited signal such that $y_2(t) = y_1(pt)$. Now assume $y_2(t)$ is sampled using parameters $\kappa$, $\theta/p$ and $\beta$ and generates the spike times $\{\tau_\ell(y_2), \ell \in \mathbb{Z}\}$, then $\tau_{\ell+1}(y_2) - \tau_\ell(y_2) = (\tau_{\ell+1}(y_1) - \tau_\ell(y_1))/p$. In essence, the information content of the two signals is the same, and the increase in bandwidth of one can be compensated for by a decrease in the threshold $\theta$ and vice versa. One can also perform similar analyses for the other parameters $\kappa$ and $\beta$. Therefore we decide to fix the first three of the four parameters $\kappa$, $\theta$, $\beta$ and $\Omega$ and only vary the last one.

## 3.B Appendix: Proof of Theorem 3.1

Assume the input signal, $y(t)$, is a $c$-bounded, $2\Omega$-bandlimited signal in $L^2(\mathbb{R})$, which has its integral well defined: $\Psi(t) := \int_{-\infty}^{t} y(u)\,du$ s.t. $|\Psi(t)| < \infty$. We wish to find an estimate of the input signal $y(t)$, which we denote $\hat{y}(t)$, using the output spike times of $N_c$ time encoding machines. Let us show that if $\hat{y}(t)$ satisfies the constraints provided by the spike times of the TEMs, then $\hat{y}(t) = y(t)$.

We start by noting that such an $\hat{y}(t)$ would then satisfy, $\forall i = 1 \cdots N_c$,

$$\int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} \hat{y}(u)\,du = \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} y(u)\,du, \quad \forall k \in \mathbb{Z}. \tag{3.15}$$

Then we use two lemmas that concern the integrals $\Psi(t) = \int_{-\infty}^{t} y(u)\,du$ and $\hat{\Psi}(t) = \int_{-\infty}^{t} \hat{y}(u)\,du$.

**Integrals are bandlimited**

**Lemma 3.1.** The integrals $\Psi(t)$ and $\hat{\Psi}(t)$ are both $2\Omega$-bandlimited.

*Proof.* The original signals $y(t)$ and $\hat{y}(t)$ are both $2\Omega$-bandlimited. Taking the integrals of these signals corresponds to a division by $\mathbf{j}\omega$ in the frequency domain, where $\omega$ denotes the frequency, so the frequency content of $\Psi(t)$ and $\hat{\Psi}(t)$ remains concentrated in $[-\Omega, \Omega]$. □

**Integrals match at spike times**

**Lemma 3.2.** $\Psi(\tau_{i,\ell}) = \hat{\Psi}(\tau_{i,\ell}), \quad \forall\ \ell \in \mathbb{Z}, \quad \forall i = 1 \cdots N_c.$

*Proof.*

$$\begin{aligned}
\Psi(\tau_{i,\ell}) &\overset{(a)}{=} \int_{-\infty}^{\tau_{i,\ell}} y(u)\,du \\
&\overset{(b)}{=} \sum_{k=-\infty}^{\ell-1} \left(\Psi(\tau_{i,k+1}) - \Psi(\tau_{i,k})\right) \\
&\overset{(c)}{=} \sum_{k=-\infty}^{\ell-1} \left(\hat{\Psi}(\tau_{i,k+1}) - \hat{\Psi}(\tau_{i,k})\right) \\
&\overset{(d)}{=} \int_{-\infty}^{\tau_{i,\ell}} \hat{y}(u)\,du \\
&\overset{(e)}{=} \hat{\Psi}(\tau_{i,\ell}), \quad \forall \ell \in \mathbb{Z},
\end{aligned}$$

where equalities $(a)$ and $(e)$ follow from the definitions of $\Psi(t)$ and $\hat{\Psi}(t)$, respectively, $(b)$ and $(d)$ follow from $y(t)$ and $\hat{y}(t)$ having well-defined integrals, and $(c)$ follows from the fact that $\Psi(\tau_{i,\ell+1}) - \Psi(\tau_{i,\ell}) = \hat{\Psi}(\tau_{i,\ell+1}) - \hat{\Psi}(\tau_{i,\ell}), \forall \ell \in \mathbb{Z}$. So $\Psi(t)$ and $\hat{\Psi}(t)$ match at all $\tau_{i,\ell},\ \ell \in \mathbb{Z},\ i = 1 \cdots N_c$. □

Therefore, $\Psi(t)$ and $\hat{\Psi}(t)$ are two $2\Omega$-bandlimited functions which coincide at time points $\tau_{i,\ell},\ \forall \ell \in \mathbb{Z},\ \forall i = 1 \cdots N_c$. In other words, if both $\Psi(t)$ and $\hat{\Psi}(t)$ are sampled at the $\tau_{i,\ell}$'s, their samples would have the same values.

Let us combine and order all spike times from the machines into one set $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$. To show that these samples are sufficient to ensure that $\Psi(t)$ and $\hat{\Psi}(t)$ match, we use a result from Jaffard (1991), where it was proven that a sampling sequence $\{\tau_\ell, \ell \in \mathbb{Z}\}$ generates a frame for the space of $2\Omega$-bandlimited functions if and only if $\{\tau_\ell, \ell \in \mathbb{Z}\}$ is relatively separated and

$$\liminf_{r \to \infty} \frac{n(r)}{r} > \frac{\Omega}{\pi}, \tag{3.16}$$

where $n(r)$ is the number of samples in an interval of length $r$.

This finding provides sufficient conditions for irregular (time, amplitude) samples to completely characterize a bandlimited signal: the sample set has to be relatively separated, and the average sampling rate needs to be higher than the Nyquist rate.

**Relatively separated sets**

**Definition 3.4.** A set $\{\tau_\ell, \ell \in \mathbb{Z}\}$ is called *relatively separated* if it can be divided into a finite number of subsets so that $|\tau_n - \tau_m| \geq \gamma > 0$ for a fixed $\gamma > 0$, $n \neq m$, and $\tau_n$, $\tau_m$ in the same subset.

Note that the set being relatively separated is only required for the reformulation in terms of a problem about frames.

It is a technical condition which is naturally satisfied in our scenario. In fact, it ensures a minimum separation between sample times.

**Combined spike times are relatively separated**

**Lemma 3.3.** Assume we have an $N_c$-channel TEM with parameters $\kappa$, $\theta$ and $\beta$, with shifts $\Delta z_i \neq 0, i = 1 \cdots N_c$, and input $y(t)$ such that $|y(t)| \leq c < \beta$. Let $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ be the spike times generated by this $N_c$-channel TEM. In other words, $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ is the combined and ordered set of spike times generated by all channels of $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$, $\text{TEM}_{N_c}$. Then, the spike times $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ are relatively separated.

*Proof.* Assume, without loss of generality that the channels $\text{TEM}_1$, $\text{TEM}_2$, $\cdots$, $\text{TEM}_{N_c}$ are ordered by spike time:

$$\tau_{1,\ell} < \tau_{2,\ell} < \cdots < \tau_{N_c,\ell} < \tau_{1,\ell+1} \qquad \forall \ell \in \mathbb{Z},$$

If we denote, as in Definition 3.2, $\Delta z_i$, $i = 1 \cdots N_c$ to be the shifts between two consecutively spiking machines, then a pair of consecutive spike times $\tilde{\tau}_\ell$ and $\tilde{\tau}_{\ell+1}$ will satisfy

$$\int_{\tilde{\tau}_\ell}^{\tilde{\tau}_{\ell+1}} y(u)\, du = 2\kappa\Delta z_i - \beta(\tilde{\tau}_{\ell+1} - \tilde{\tau}_\ell), \tag{3.17}$$

for some $\Delta z_i$ that depends on the provenance of $\tilde{\tau}_\ell$ and $\tilde{\tau}_{\ell+1}$, which is determined by $\ell$ since different machines always spike in order (see Definition 3.2).

Now recall that $|y(t)| \le c$, which, when substituted into (3.17), yields

$$\begin{aligned} c(\tilde{\tau}_{\ell+1} - \tilde{\tau}_\ell) &\ge 2\kappa\Delta z_i - \beta(\tilde{\tau}_{\ell+1} - \tilde{\tau}_\ell), \\ \tilde{\tau}_{\ell+1} - \tilde{\tau}_\ell &\ge \frac{2\kappa\Delta z_i}{\beta + c}, \end{aligned}$$

for some $i \in \{1, \cdots, N_c\}$ which depends on $\ell$. Then,

$$\tilde{\tau}_{\ell+1} - \tilde{\tau}_\ell \ge \frac{2\kappa \min_i(\Delta z_i)}{\beta + c}.$$

Now denote $\gamma = 2\kappa \min_i(\Delta z_i)/(\beta + c)$. Note that $\gamma$ is nonzero because all $\Delta z_i$'s are assumed to be nonzero. Therefore, our sampling set $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ is relatively separated. □

On the other hand, to help us prove that the Nyquist-like condition in (3.16) is satisfied, the following lemma provides us with a lower bound on the average sampling rate of our spike times $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$.

**Minimal multi-channel sampling rate**

**Lemma 3.4.** The sampling set $\{\tilde{\tau}_\ell, \ell \in \mathbb{Z}\}$ has an average sampling rate which is at least $N_c(\beta - c)/(2\kappa\theta)$.

*Proof.* Spike times have a maximal separation between them defined by (2.2). According to this bound, every machine $\text{TEM}_i$ produces a sampling set $\{\tau_{i,\ell}, \ell \in \mathbb{Z}\}$ where two spike times have a separation of at most $2\kappa\theta/(\beta - c)$. Therefore, the sampling rate $n(r)/r$ is at least $(\beta - c)/2\kappa\theta$, for any $r \in \mathbb{R}$ . Therefore, the average sampling rate of a machine $\liminf_{r\to\infty} n(r)/r$ is at least $(\beta - c)/2\kappa\theta$. Since all machines fire at distinct time points (because the shifts between them are nonzero), together, they have an average sampling rate which is at least $N_c(\beta - c)/2\kappa\theta$. □

It follows that the samples emitted by the TEMs are sufficient to determine uniqueness for a $2\Omega$-bandlimited signal, provided that $\Omega$ satisfies (3.5).

Hence, a signal $\Psi(t)$ which is $2\Omega$-bandlimited, with $\Omega$ satisfying (3.5), is uniquely defined by the samples provided by a $N_c$-channel TEM with parameters $\kappa$, $\theta$ and $\beta$ and input $y(t)$, such that $|y(t)| \leq c < \beta$, $y(t) \in L^2(\mathbb{R})$ and has a well defined integral, if the shifts between the machines are nonzero. Therefore, $\Psi(t)$ and its estimate $\hat{\Psi}(t)$ match exactly, and as $y(t)$ and $\hat{y}(t)$ are their respective derivatives, they are also completely characterized by the samples and match exactly. Therefore, the input to the $N_c$-channel TEM is uniquely defined by iits spike time output under the conditions of Theorem 3.1.

# 4 Time Encoding Multiple Signals with Low Dimensional Structure

***Recommended Reading:*** Sections 2.1, 2.2, 2.3, 2.5.
*The content in this chapter is adapted from Adam et al. (2020a, 2022).*

## 4.1 Introduction

One would expect that, when sampling many signals that have a low-dimensional structure, that is, they can be written as linear combinations of *few* signals, the number of samples needed to ensure recovery would depend on the complexity of these *few* underlying signals, and not on the "apparent" complexity of the input.

However, if one goes about sampling these many signals at the same uniformly spaced times, the above statement falls through.

In fact, it only holds if signals are sampled at *different* times.

Therefore, as time encoding paves a natural way to asynchronous sampling, we here study multi-channel time encoding of multiple signals with low-dimensional representations.

More specifically, we assume that $N_c \geq 1$ time encoding machines receive, as input, signals that can each be written as a linear combination of $N_s$ signals where $N_s << N_c$. We find that, if each input can be described using a finite number of degrees of freedom that are unknown apriori, a Nyquist-like criterion applies on the number of spikes needed for reconstruction, requiring as many linearly independent constraints as degrees of freedom *in the lower dimensional representation*.

In fact, we will see that the minimum number of spike pairs required will be $N_s K$, with some conditions on the provenance of the spikes, where $K$ is the number of degrees of freedom per signal, and where $N_s$ is potentially much smaller than $N_c$. Note that, when uniformly and synchronously sampling $N_c$ signals, a setup where all samplers record their information at the

same time, one would need $N_c K$ measurements to recover the input.

We will see that this setup not only allows more efficient reconstruction, but also a more collaborative one. Time encoding machines or neurons that spike too little can be compensated for by having other machines that spike more frequently. However, every machine's contribution is limited to a maximum amount.

The results of this chapter follow the same spirit as those in Chapter 3, and indicate that time encoding or event-based sensing allows compensating for low spiking rates by increasing the number of sensors to ensure signal recovery. This, in turn, means that (1) sensors do not require higher spiking rates to encode more complex information, and (2) the effect of noise can be limited because sensors can have a lower spiking rate, and can thus integrate information over more time.

First, we present the sampling setup for mixed multi-channel time encoding. We then provide a bound for reconstruction of two classes of bandlimited signals assuming the mapping to the low-dimensional representation is known. This bound will depend on the number of degrees of freedom in the low-dimensional space. Then, as the time encoding problem can be cast as a problem of rank-one sensing (Pacholska et al., 2020), this allows us to derive two reconstruction algorithms, one of them based on Projections onto Convex Sets as in Section 2.3 and the other based on the pseudo-inversion of a linear system. Later, in Section 4.4, we treat the case where the mapping to the lower dimensional space is not known but the dimensionality is known. We consider an algorithm based on Singular Value Projection as described in Section 2.5 to perform more efficient reconstruction and conclude with simulation results in Section 4.4.2.

## 4.2 Problem Setup

In this chapter, we consider many time-varying signals $y_i(t), i = 1 \cdots N_c$, that are correlated with each other. These signals are encoded using time encoding machines and we assume that each signal follows a parametric model which we know.

The goal is to recover the unknown inputs $y_i(t)$ from their time encoding.

Correlated signals arise in applications where principle component analysis yields little loss in information, such as, among others, meteorological data, biomarkers in human patients, regional economic data, audio, and video. The latter example will be given particular attention in Chapter 5.

In our setup, we let $\mathbf{y}(t)$ denote the vector signal composed of $y_i(t)$'s and let $\mathbf{y}(t)$ be such that

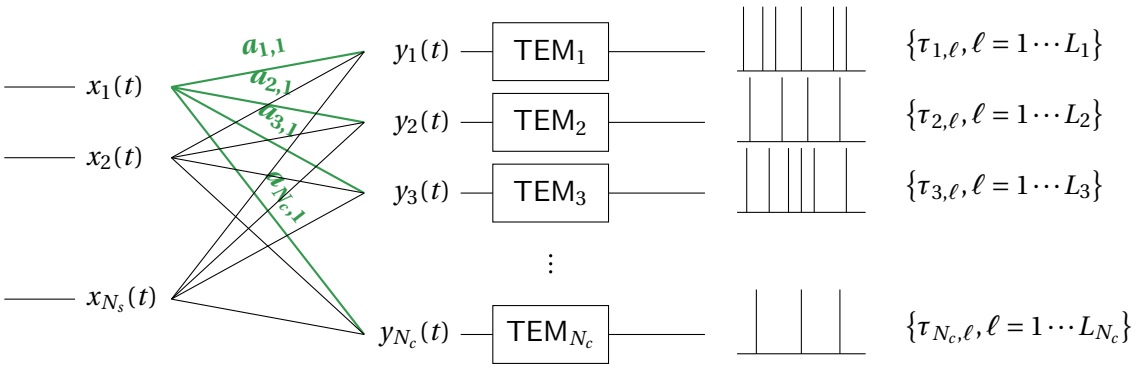



Figure 4.1 – Sampling setup: $N_s$ input signals $x_j(t)$, $j = 1 \cdots N_s$ are mixed using a matrix $\mathbf{A}$ and produce signals $y_i(t)$, $i = 1 \cdots N_c$. Each $y_i(t)$ is then sampled using a time encoding machine TEM$_i$ which produces spike times $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}$.

(A4) each $y_i(t)$ has a finite parametric representation:

$$y_i(t) = \sum_{k=1}^{K} c_{i,k}(\mathbf{y}) f_k(t), \tag{4.1}$$

where the $c_{i,k}(\mathbf{y})$ are fixed coefficients that are unknown apriori and form a matrix $\mathbf{C}(\mathbf{y})$, and the $f_k(t)$'s, $k = 1...K$ are known functions,

(A5) each $y_i(t)$ can be written as a linear combination of $x_j(t)$'s, $j = 1 \cdots N_s$ where $N_s < N_c$:

$$\mathbf{y}(t) = \mathbf{A}\mathbf{x}(t), \tag{4.2}$$

for $x_j(t)$'s characterized by a matrix of coefficients $\mathbf{C}(\mathbf{x})$ (where the $x_j(t)$'s and $\mathbf{C}(\mathbf{x})$ are unknown apriori) and a mixing matrix $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$, such thats $\mathbf{C}(\mathbf{y}) = \mathbf{A}\mathbf{C}(\mathbf{x})$, and

(A6) each $y_i(t)$ is sampled using a time encoding machine TEM$_i$ with parameters $\kappa_i$, $\theta_i$ and $\beta_i$ which are known and can vary between machines. The outputs of the machines are denoted $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}$.

The sampling setup we described is depicted in Fig. 4.1.

Our results will hold if the integrals of the functions $f_k(t)$ are linearly independent functions from a linear space of functions $\mathscr{F}$, such that every non-zero element of $\mathscr{F}$ has a set of zeros with Lebesgue measure equal to zero. We will consider two options for the functions $f_k(t)$:

(A7.a) $f_k(t)$ is a sinc function

$$f_k(t) = \mathrm{sinc}_\Omega(t - t_k) = \frac{\sin(\Omega(t - t_k))}{\pi(t - t_k)}, \tag{4.3}$$

for $\Omega$ and $t_k$ known, so that the $y_i(t)$'s are a finite sum of sincs, or

(A7.b) $f_k(t)$ is a complex exponential function,

$$f_k(t) = \exp\left(\mathbf{j}\frac{2\pi}{T}kt\right), \tag{4.4}$$

so that the $y_i(t)$'s are bandlimited periodic functions.

For both functions in (A7.a) and (A7.b), we consider the reconstruction conditions with $\mathbf{A}$ satisfying either of the following two assumptions.

(A8.a) The linear map from the low dimensional space $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ is known.

(A8.b) The linear map from the low dimensional space $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ is unknown but the dimension of the low dimensional space $N_s$ is known.

We first consider the case where $\mathbf{A}$ is known and provide conditions for perfect reconstruction in Section 4.3.1 and a reconstruction algorithm in Sections 4.3.2 and 4.3.3.

Later, we will consider the case where $\mathbf{A}$ is unknown and provide a reconstruction algorithm based on singular value projection for low-rank matrix recovery in Section 4.4. We then follow with simulations to show results and with example applications for time encoding time-varying scenes.

Later, In Chapter 5, we provide applications for the known mixing matrix scenario, where we deal with time encoding of video.

## 4.3 Known Low-Rank Factorization: Time Encoding and Reconstruction

### 4.3.1 Conditions for Perfect Reconstruction

We can establish the following sufficient conditions to ensure that a series of inputs $y_i(t)$ drawn at random are reconstructible from their time encoding using machines $\text{TEM}_i$.

**Perfect reconstruction from mixed time encoding**

**Theorem 4.1.** Let $N_c$ signals $y_i(t), i = 1 \cdots N_c$ satisfy assumptions (A4), (A5) and (A6), and their functions $f_k(t)$ satisfy either of (A7.a) or (A7.b) with the corresponding coefficients $c_{j,k}(\mathbf{x})$ being drawn from a Lipschitz continous probability distribution. Now assume $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ as defined in (A5) is known and has every $N_s$ rows linearly independent. Then the inputs $y_i(t), i = 1 \cdots N_c$ are exactly determined by

the spike times $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}, i = 1 \cdots N_c$, with probability one, if:

$$\sum_{i=1}^{N_c} \min(L_i - 1, K) > N_s K. \tag{4.5}$$

An intuitive explanation of this result follows in Section 4.3.4.

We can prove the above theorem by writing it as a problem of rank one measurements, also called bi-linear measurements in (Pacholska et al., 2020). The full proof is provided in Appendix 4.A.

### 4.3.2 One-Shot Reconstruction Algorithm

The spike time outputs of the machines $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}$ provide constraints on the integral of the input signals:

$$\int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} y_i(u)\, du = 2\kappa_i\theta_i - \beta_i(\tau_{i,\ell+1} - \tau_{i,\ell}) =: \dot{b}_{i,\ell}. \tag{4.6}$$

These measurements can be rewritten to fit the rank one measurements formulation (Pacholska et al., 2020). Letting $\mathbf{C}(\mathbf{x})$ denote the matrix of coefficients $c_{j,k}(\mathbf{x})$ for the underlying signals $x_j(t)$, we can reconstruct $\mathbf{C}(\mathbf{x})$ (and therefore $\mathbf{y}(t)$) by solving

$$\dot{b}_{i,\ell} = \mathrm{vec}\left(\mathbf{a}_i \left[\dot{\mathbf{F}}_{i,\ell}\right]^T\right) \mathrm{vec}\left(\mathbf{C}(\mathbf{x})\right), \tag{4.7}$$

where $\dot{b}_{i,\ell}$ is known, vec() denotes the vectorization operation,

$$\left[\dot{\mathbf{F}}_{i,\ell}\right]_k = \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} f_k(u)\, du,$$

and $\tau_{i,1}$ denotes the first spike time of $\mathrm{TEM}_i$.

Under the conditions of Theorem 4.1, the linear system in (4.7) is full rank and $\mathbf{C}(\mathbf{x})$ can be recovered perfectly. Once the matrix $\mathbf{C}(\mathbf{x})$ has been recovered, one can recover the coefficients $c_{i,k}(\mathbf{y})$ of the $y_i(t)$'s by setting $\mathbf{C}(\mathbf{y}) = \mathbf{A}\mathbf{C}(\mathbf{x})$ and can therefore recover the original sampled signals.

Note that, given this formulation, there are different approaches to solving for $\mathbf{C}(\mathbf{x})$. One can either invert the system in (4.7) by using a pseudo-inverse, or use iterative approaches to minimize the mean-squared error, such as gradient descent or projections onto convex sets (POCS) (Bauschke and Borwein, 1996; Feichtinger and Gröchenig, 1994; Thao and Rzepka, 2020) as we did in Adam et al. (2020a). All of these methods converge to the same result,

given that a pseudo-inverse approach minimizes the mean-squared error by definition, that the gradient descent approach is given a convex loss function and that the POCS approach uses convex sets which have a unique intersection. For the experiments in this chapter, we use the pseudo-inverse approach, mostly for its speed of execution. Nonetheless, we explain how to go about a POCS approach, next.

### 4.3.3 POCS Reconstruction Algorithm

As in Section 2.3, we use a projection onto convex sets algorithm to reconstruct signals with a low dimensional representation from the spike times they generate when fed through TEMs.

We again refer the reader to our definition of the POCS method in Section 2.3 (Definition 2.4).

The POCS algorithm is known to converge to a fixed point which lies in the intersection of the sets at hand $\bigcap_{n=1}^{N} \mathscr{C}_n$ (Bauschke and Borwein, 1996; Thao and Rzepka, 2020). Thus, if the intersection of the sets consists of a single element, then the algorithm converges to the *correct* solution.

Theorem 4.1 states that, if (4.5) is satisfied, the solution $\mathbf{x}(t)$, and thus $\mathbf{y}(t) = \mathbf{A}\mathbf{x}(t)$, is unique. We will therefore set up a POCS algorithm to first recover $\mathbf{y}(t)$ and then $\mathbf{x}(t)$.

To recover $\mathbf{y}(t)$, we define three convex sets: the set $\mathscr{C}_\Omega$ of collections of $N_c$ signals formed using a sum of $K$ functions $f_k(t)$ which are either sincs as in (A7.a) or exponentials as in (A7.b) depending on the starting assumption, the set $\mathscr{C}_{\mathbf{spikes}}$ of collections of signals that satisfy the constraints that are set by the spike times of each machine $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}$ and the set $\mathscr{C}_{\mathbf{A}}$ of collections of signals $\mathbf{y}(t)$ which can be written $\mathbf{y}(t) = \mathbf{A}\mathbf{x}(t)$.

**The intersection $\mathscr{C}_\Omega \cap \mathscr{C}_{\mathbf{spikes}} \cap \mathscr{C}_{\mathbf{A}}$ is the set of solutions**

**Lemma 4.1.** The intersection $\mathscr{C}_\Omega \cap \mathscr{C}_{\mathbf{spikes}} \cap \mathscr{C}_{\mathbf{A}}$ is the set of solutions $\mathbf{y}(t)$, given spike times $\tau_{i,\ell}$ and mixing matrix $\mathbf{A}$.

*Proof.* It is easy to see that a solution $\hat{\mathbf{y}}(\mathbf{t})$ lies in $\mathscr{C}_\Omega \cap \mathscr{C}_{\mathbf{spikes}} \cap \mathscr{C}_{\mathbf{A}}$. Now assume that $\hat{\mathbf{y}}(t) \in \mathscr{C}_\Omega \cap \mathscr{C}_{\mathbf{spikes}} \cap \mathscr{C}_{\mathbf{A}}$. Then $\exists \hat{\mathbf{x}}(t)$ formed using a sum of $K$ functions $f_k(t)$ defined either as in (A7.a) or as in (A7.b), depending on the starting assumptions, such that $\hat{\mathbf{y}}(t) = \mathbf{A}\hat{\mathbf{x}}(t)$ and $\hat{\mathbf{y}}(t)$ produces the obtained spike times. Therefore $\hat{\mathbf{y}}(t)$ is a solution to the input of the machines. □

Each of these sets is convex, therefore, we define operators $\mathscr{P}_\Omega$, $\mathscr{P}_{\mathbf{spikes}}$, and $\mathscr{P}_{\mathbf{A}}$ that project orthogonally onto $\mathscr{C}_\Omega$, $\mathscr{C}_{\mathbf{spikes}}$ and $\mathscr{C}_{\mathbf{A}}$, respectively. Then, we alternately apply these projection operators to an initial estimate, and, since the intersection is unique, we converge

to the correct solution.

If we assume (A7.a) holds, to project onto $\mathscr{C}_\Omega$, we convolve the input with a sinc of bandwidth $\Omega$, sample the obtained signal at values $t_k$ and use the values as amplitudes of the sincs located at $t_k$:

$$\mathscr{P}_{\Omega,i}\left(\hat{\mathbf{y}}(t)\right) = \sum_{k=1}^{K} \hat{y}_{\Omega,i}(t_k)\mathrm{sinc}_\Omega(t-t_k), \tag{4.8}$$

where $\hat{y}_{\Omega,i}(t) = \hat{y}_i(t) * \mathrm{sinc}_\Omega(t)$. Otherwise, if we assume (A7.b) holds, the projection is performed by simply finding the closest periodic bandlimited function to each $\hat{y}_{\Omega,i}(t_k)$, using inner products with the complex exponentials of interest. To project onto $\mathscr{C}_{\mathbf{spikes}}$, we define $\mathscr{P}_{\mathbf{spikes}}$ to act on each row of $\hat{\mathbf{y}}(t)$ individually:

$$\mathscr{P}_{\mathbf{spikes},i}\left(\hat{\mathbf{y}}(t)\right) = \hat{y}_i(t) + \sum_{\ell=1}^{L} q_{i,\ell}\frac{\mathbb{1}_{[\tau_{i,\ell},\tau_{i,\ell+1})}(t)}{\tau_{i,\ell+1}-\tau_{i,\ell}} \tag{4.9}$$

where $q_{i,\ell} = \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} \left(\hat{y}_i(u) - y_i(u)\right) du$, and $\mathbb{1}_{[a,b)}(t)$ is the indicator function over $[a,b)$.

In words, for each $\hat{y}_i(t)$, $\mathscr{P}_{\mathbf{spikes}}$ adds rectangles over the intervals $\left[\tau_{i,\ell},\tau_{i,\ell+1}\right]$ with appropriate weights to satisfy the constraints set by $\left\{\tau_{i,\ell}, \ell = 1\cdots L_i\right\}$.

Now, to project $\hat{\mathbf{y}}(t)$ onto the set $\mathscr{C}_{\mathbf{A}}$, we let

$$\mathscr{P}_{\mathbf{A}}\left(\hat{\mathbf{y}}(t)\right) = \mathbf{A}\left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\hat{\mathbf{y}}(t). \tag{4.10}$$

Thus, our reconstruction algorithm runs iteratively and computes new estimates $\hat{\mathbf{y}}^{(k)}(t)$ of the originally sampled signals:

$$\hat{\mathbf{y}}^{(0)}(t) = 0, \qquad \hat{\mathbf{y}}^{(k+1)}(t) = \mathscr{P}_\Omega\left(\mathscr{P}_{\mathbf{A}}\left(\mathscr{P}_{\mathbf{spikes}}\left(\hat{\mathbf{y}}^{(k)}(t)\right)\right)\right). \tag{4.11}$$

In the end, we set $\hat{\mathbf{x}}^{(k)}(t) = \left(\mathbf{A}^T\mathbf{A}\right)^{-1}\mathbf{A}^T\hat{\mathbf{y}}^{(k)}(t)$.

Given that this is a POCS algorithm and that our theorem states uniqueness, $\hat{\mathbf{y}}^{(k)}(t)$, and therefore $\hat{\mathbf{x}}^{(k)}(t)$, will converge to the correct solution as $k \to \infty$.

### 4.3.4 Interpretation

The result in Theorem 4.1 establishes a Nyquist-like criterion for recovery. It specifies how to count the number of linearly independent constraints in the multi-channel TEM setup and requires as many of these constraints as there are degrees of freedom to recover the sampled signals. Given that each *pair* of spike times corresponds to one linear constraint on the input, the results can be summarized by a few key points:

1. When sampling a collection of signals with a known linear mapping to or from a lower

dimensional representation, what matters is the number of degrees of freedom in the low dimensional space, rather than the number of degrees of freedom in the high dimensional space. More practically, to ensure perfect reconstruction, we need the number of linearly independent constraints to be at least the number of degrees of freedom in the low dimensional space $N_s K$. In the case where $N_s << N_c$, we can see how this can be a major improvement in spiking rate.

2. When multiple correlated signals are sampled using different time encoding machines, a lower spiking rate of one machine can be compensated for by higher spiking rates from others. This can be seen by observing the summation in (4.5) and noting that the *total* spiking rate of the machines matters more than the individual spiking rates.

3. One machine can only compensate for another machine's low spiking rate up to a certain degree. This can be seen by the $\min$ term in (4.5) which implies that every machine has a maximal "useful" spiking rate depending on the signal and that going above this rate does not add further information.

This has a series of implications. First, signals that have lower dimensional representations can be sampled at lower rates overall, increasing sampling efficiency. Second, if TEMs have limited capacity in terms of spiking rates (for example they have a refractory period), this can be compensated for by adding more TEMs. This would still ensure reconstruction of the input since the reconstruction condition in (4.5) is only linked to the number of degrees of freedom in the low dimensional space. Third, we will see in Section 5.3 how the results help us solve time encoding of time-varying spatial signals which have certain structure in space.

Note that these results provide a stark improvement to sampling high-dimensional but low-complexity signals using regular clock-based sampling. In fact, Theorem 4.1 holds because of one key element: different dimensions of the signal are sampled at *different* times with continuous probability distributions. Regular-based sampling does not have this property; and indeed, it only takes a short mental exercice to see that the recovery of $\mathbf{y}(t)$ takes $N_c K$ samples if the $y_i(t)$'s are all sampled at the same sampling times.

To be fair, one *could* ensure that different $y_i(t)$'s are sampled at different times (minus the continuous probability condition), but this condition is much more elegantly ensured in the time encoding scenario. Moreover, using different clocks in the classical sampling setup poses difficulties because it is hard to align them. Clock alignment is not an issue in time encoding: the outputs are trains of spikes and finding delays between TEMs is solved by adding spike trains and comparing spike times on one time axis, as explained in the "in depth" box of Section 3.5.

#### Possible extensions

The results in this section assumed a fixed signal model and relied on two key elements: the structure of the matrix $\mathbf{A}$ and the timing of the spikes which is asynchronous across machines.

In this case, the asynchrony of the spikes across TEMs occurs because the matrix $\mathbf{A}$ has different rows. However, this asynchrony can also be ensured by TEMs having different parameters, as discussed in Chapter 3.

In such as scenario, repeated (or linearly dependent) entries in $\mathbf{A}$ are allowed and our assumptions can be relaxed, so that $\mathbf{A}$ does not have to have every $N_s$ rows linearly independent.

We choose not to further explore this possibility, here, but rather to provide a basic framework to understand time encoding of mixed signals.

## 4.4 Unknown Low-Rank Factorization

### 4.4.1 Problem Formulation and Algorithm

We revisit the setup exposed in Section 4.2. So far, we have assumed that we are given the time encodings of a collection of signals $y_i(t)$ with a low dimensional structure which we can reach by a *known* linear transformation $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ and that we are asked to reconstruct the inputs $y_i(t)$. While this is a useful model in itself, we are also interested in studying the case where the linear transform $\mathbf{A}$ is unknown.

Once again, we assume we have the time encodings of a collection of signals $y_i(t)$ which satisfy assumptions (A4), (A5) and (A6). Furthermore, we assume the functions $f_k(t)$ of $y_i(t)$ satisfy either of (A7.a) or (A7.b) and that the linear transformation $\mathbf{A}$ is unknown as in (A8.b).

We wish to recover the signals $y_i(t), i = 1 \cdots N_c$, from their time encoding, with as few samples as possible.

To do so, we aim to reconstruct the coefficients of the parametric representation of $\mathbf{y}(t)$, $c_{i,k}(\mathbf{y})$ as defined in (A4). These coefficients are placed in the matrix $\mathbf{C}(\mathbf{y})$, with row $i$ containing the coefficients of signal $y_i(t)$. We note once more that $\mathbf{C}(\mathbf{y})$ can be written:

$$\mathbf{C}(\mathbf{y}) = \mathbf{A}\mathbf{C}(\mathbf{x})$$

where $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$, $\mathbf{C}(\mathbf{x}) \in \mathbb{R}^{N_s \times K}$, $N_s \leq N_c$ and $N_s$ is known.

In words, $\mathbf{C}(\mathbf{y})$ is a matrix which has a low rank matrix decomposition with a known rank.

The matrix $\mathbf{C}(\mathbf{y})$ is probed using a sensing operator which we will call $\mathscr{S}$. The sensing operator

performs the measurements in (4.7), i.e.

$$\mathscr{S}_n\left(\mathbf{C}(\mathbf{y})\right) = \dot{b}_n, \tag{4.12}$$

where we index a pair $(i,\ell)$ by $n$ so that $\dot{b}_n = \dot{b}_{i,\ell}$ is as defined in (4.6)

Given this measurement setup, we can adopt the Singular Value Projection approach to recover the matrix $\mathbf{C}(\mathbf{y})$ from few measurements (Jain et al., 2010).

The Singular Value Projection (SVP) algorithm alternately applies the low-rank constraint and the measurement constraint on the matrix of interest $\mathbf{C}(\mathbf{y})$. In Algorithm 1 we let $\hat{\mathbf{C}}(\mathbf{y})^{(m)}$ be the estimate at iteration $m$ of the target matrix to reconstruct (in our case this is $\mathbf{C}(\mathbf{y})$) and $\mathbf{P}^m$ be a proxy matrix to perform the iterations.

**Algorithm 2** Singular Value Projection for Mixed Time Encoding

**Input:** $\mathscr{S}$, $\dot{\mathbf{b}}$, rank $N_s$, tolerance $\epsilon$, $\eta_m$ for $m = 0,1,2...$

1: $\hat{\mathbf{C}}(\mathbf{y})^{(0)} = 0$ and $m = 0$
2: **repeat**
3: $\quad \mathbf{P}^{m+1} \leftarrow \hat{\mathbf{C}}(\mathbf{y})^{(m)} - \eta_m \mathscr{S}^T(\mathscr{S}(\hat{\mathbf{C}}(\mathbf{y})^{(m)}) - \dot{\mathbf{b}})$
4: $\quad$ Compute top $N_s$ singular vectors of $\mathbf{P}^{(m+1)} : U_{N_s}, \Sigma_{N_s}, V_{N_s}$
5: $\quad \hat{\mathbf{C}}(\mathbf{y})^{(m+1)} \leftarrow U_{N_s}\Sigma_{N_s}V_{N_s}^T$
6: $\quad m \leftarrow m+1$
7: **until** $\|\mathscr{S}(\hat{\mathbf{C}}(\mathbf{y})^{(m+1)} - \dot{\mathbf{b}})\|_2^2 \leq \epsilon$

The SVP algorithm is based on projected gradient descent. Reconstruction guarantees for this algorithm were initially established in cases where the sensing operator satisfies the Restricted Isometry Property (Jain et al., 2010; Candès and Recht, 2009; Recht et al., 2010). This property does not hold in our case, given that our measurement operators $\mathscr{S}_n$ have rank one. The rank one scenario has been treated in Zhong et al. (2015) where Gaussianity assumptions are made. Again, these assumptions do not hold for our case and we leave the theoretical analysis of convergence for future work. We do, however, illustrate the utility of our approach with simulations in the next section.

### 4.4.2 Simulations

We provide simulation results to evaluate the reconstruction performance in different regimes. We consider the scenario where we time encode and reconstruct twenty signals that are composed of 25 $\mathrm{sinc}$ functions at known locations and that can be written as linear combinations of two such signals.

We evaluate the reconstruction performance that varies as the number of spikes of all machines increase uniformly. We do this in the following cases:

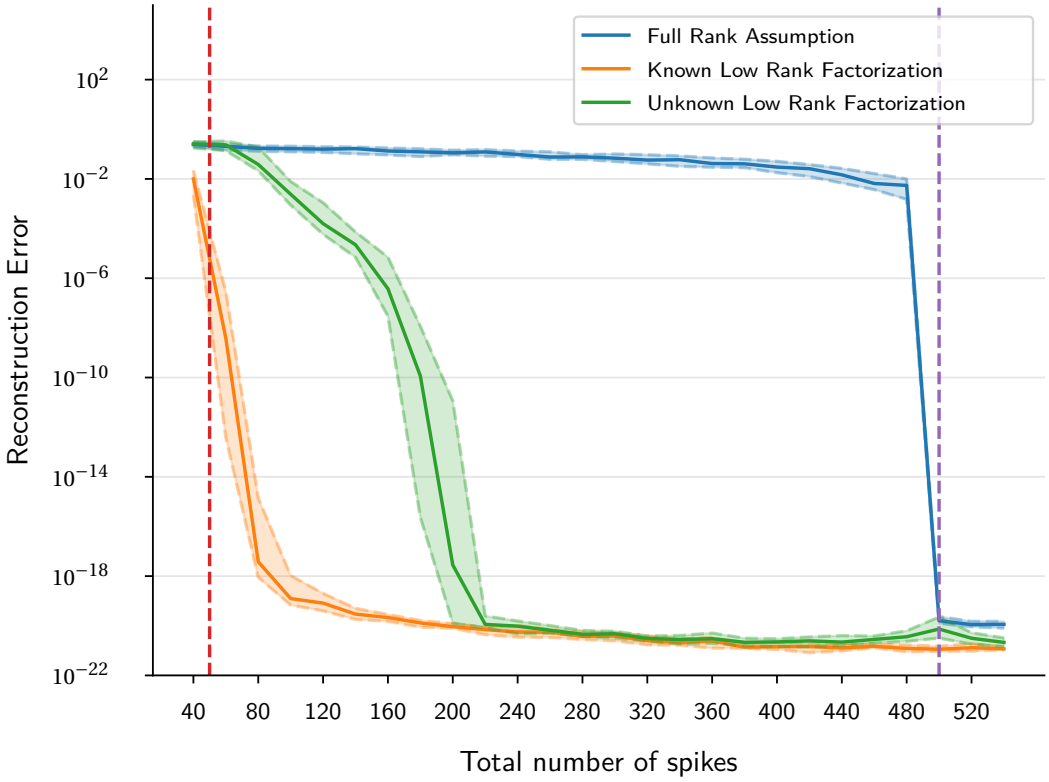



Figure 4.2 – Reconstruction error of one out of $N_c = 20$ signals that have rank $N_s = 2$, as the number of emitted spikes increases. The red dashed line marks the perfect reconstruction condition assuming the transform to the low dimensional space is known, and the purple dashed line marks the perfect reconstruction condition assuming there is no lower dimensional representation of the signals. We show the median and quartiles of the reconstruction error for 25 random trials, when assuming the signals have no low dimensional structure, when assuming they have a low dimensional structure with a known linear mapping, and when they have a low dimensional structure with an unknown linear mapping.

(S1) when assuming the signals have no underlying low dimensional structure,

(S2) when assuming the signals have an underlying low dimensional representation which we can reach through a known linear transform $\mathbf{A}$, and

(S3) when the signals have an underlying low dimensional representation with an unknown mapping $\mathbf{A}$.

For each of these cases, we draw the entries of $\mathbf{C}(\mathbf{x})$ and $\mathbf{A}$ uniformly at random and time encode and reconstruct all twenty signals, computing the obtained normalized mean-squared error for the first signal among the twenty, assuming a random mapping $\mathbf{A}$ to low dimensional space. Then we plot the median and quartiles of the mean-squared error on a log plot to compare performance. Results are included in Fig. 4.2.

Note that, if we assume no underlying low dimensional structure (S1), the signals can be reconstructed assuming there are $N_c \times K$ linearly independent constraints. In this case, since the number of spikes of all machines increase uniformly, we will need $N_c \times K = 500$ spikes. As for the scenario (S2), according to Theorem 4.1, the signals can be reconstructed assuming that there are $N_s \times K$ linearly independent constraints. As before, this means we would need $N_s \times K = 50$ spikes.

We draw each of these conditions in Fig. 4.2 to see if the performance is consistent with our expectations.

Assuming we know the transformation $\mathbf{A}$ to a low-dimensional space (S2) greatly improves reconstruction compared to when we assume that there is no low rank structure for the input (S1): the error decays much earlier in the first case than it does in the second case.

Assuming such a transformation exists but that we do not know it (S3), also offers benefit. While the reconstruction algorithm can be quite unstable in regimes where the number of spikes is not sufficient, it can yield a very good reconstruction for a higher number of spikes, where the scenario (S1) fails entirely.

## 4.5 Conclusion

We have shown how time encoding can be used to encode and reconstruct multiple signals that have lower-dimensional representations.

We have reformulated our problem as a rank-one matrix measurement problem and have shown that signals that have a known lower dimensional representation require fewer spikes for perfect reconstruction than if this lower dimensional representation did not exist. This is stark contrast to the same scenario where the signals are sampled using synchronous, uniform sampling. In the case of time encoding, it is the "underlying" complexity of the signal which dictates the spiking rate required for recovery, whereas in synchronous uniform sampling, the "apparent" complexity dictates the necessary sampling rate.

We have also examined the case where the signals of interest have low rank but we do not know the transformation to the low rank space. We applied low rank factorization algorithms and found significant experimental improvements compared to the case where no low rank structure is assumed.

For readers who wonder how difficult it is to achieve the same results with uniform sampling, we refer to the next chapter on time encoding video. There, we stress on the need for *asynchronous* sampling approaches. We contextualize our results to show that frame-based video is suboptimal in terms of sampling efficiency and that a paradigm shift is needed in the way we perform sampling. Time encoding provides a tool to achieve this paradigm shift.

## 4.A Appendix: Known Low Dimensional Mapping - Elaboration and Proof of Theorem 4.1

To prove Theorem 4.1, we will use results about rank-one matrix measurements (Pacholska et al., 2020). The work in Pacholska et al. (2020) assumes that one is attempting to reconstruct a matrix $\mathbf{C}$ using measurements of the form:

$$b_n = \mathbf{g}_n^T \mathbf{C} \mathbf{h}_n, \tag{4.13}$$

and rewrites the measurements as

$$b_n = \mathrm{vec}(\mathbf{g}_n \mathbf{h}_n^T)^T \mathrm{vec}(\mathbf{C}), \tag{4.14}$$

where vec() is the vectorization operator.

Note that we adopted a change of notation with respect to Pacholska et al. (2020) to avoid confusion.

The results of Pacholska et al. (2020) then hold under two further assumptions.

(A9) $\mathbf{h}_n$ can be parametrized by one variable $t \in \mathbb{R}$. More precisely, we assume the $k$-th entry of $h_n$ has the form $[h_n]_k = h_k(t_n)$ where $h_k : \mathscr{I} \to \mathbb{R}, k = 0, \cdots, K-1$ are linearly independent functions from a linear space of fucntions $\mathscr{F}$, $\mathscr{I} \in \mathbb{R}$ is an interval or the whole real line and $t_n \in \mathscr{I}, n = 0, ..N-1$ are sampling times. Moreover, it is assumed that the sampling times $(t_0, ...t_{N-1})$ follow a continuous probability distribution on $\mathscr{I}^N$ and that for every non-zero element $h \in \mathscr{F}$, the set of zeros of $h$ has Lebesgue measure $(\lambda)$ equal to zero: $\lambda(\{t | f(t) = 0\}) = 0$.

(A10) The vectors $\mathbf{g}_n$ are taken from a set $\mathscr{A}$, where every $J$ elements of $\mathscr{A}$ are linearly independent.

As a result, a uniqueness condition can be obtained.

**Matrix recovery from bilinear measurements (Pacholska et al., 2020)**

**Theorem 4.2.** Consider the set of $KJ$ vectors of the form $\mathrm{vec}(\mathbf{g}_n \mathbf{h}_n^T)$. It is a basis in $\mathbb{R}^{KJ}$ if and only if no more than $K$ vectors $\mathbf{g}_n$ are equal.

We are able to rewrite our problem as a rank-one measurement problem and use Theorem 4.2 in combination with the following lemmas to prove Theorem 4.1.

**Spike times follow a continuous probability distribution**

**Lemma 4.2.** Under the assumptions of Theorem 4.1, the spike times $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}, i = 1 \cdots N_s$ follow a continuous probability distribution.

*Proof.* This closely follows the proof in Pacholska et al. (2020). □

**Zero set of $f_k$'s is measure zero**

**Lemma 4.3.** Let the $f_k(t)$'s be the functions as defined in (A7.a) or (A7.b) and define $F_k(t) = \int_{t_0}^{t} f_k(u)\,du$. Then the $F_k$'s are linearly independent functions from a linear space of functions $\mathscr{F}$ which is the space of bandlimited functions. Moreover, every non-zero element of $\mathscr{F}$ has a set of zeros with Lebesgue measure equal to zero.

*Proof.* This follows by construction of the $f_k$'s which are linearly independent, leading to their integrals being linearly independent. The second part of the lemma follows from both sums of complex exponentials and sinc functions having a countable number of zeros. The sum of complex exponentials has a countable number of zeros because it can be rewritten as a polynomial, which has zeros described by the fundamental theorem of algebra, and the sum of sincs can also be written as a polynomial of a function of time, divided by the time, which also yields countable zeros (Vetterli et al., 2014). Given that the set of zeros of each of these functions is countable, they have Lebesgue measure zero (Tao, 2011). □

Before proving Theorem 4.1, we prove the following lemma.

**Perfect reconstruction from mixed time encoding with known initial conditions**

**Lemma 4.4.** Let $N_c$ signals $y_i(t), i = 1 \cdots N_c$, satisfy assumptions (A4), (A5) and (A6), and their function $f_k(t)$ satisfy either of (A7.a) or (A7.b) with the corresponding coefficients $c_{j,k}(\mathbf{x})$ being drawn from a Lipschitz continous probability distribution. Now assume $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ as defined in (A5) is known and has every $N_s$ rows linearly independent. Then the inputs $y_i(t), i = 1 \cdots N_c$ are exactly determined by the spike times $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}, i = 1 \cdots N_c$, if:

$$\sum_{i=1}^{N} c \min(L_i, K) > N_s K, \tag{4.15}$$

if the time encoding machines start sampling at $t_0$ with a known integrator value $z_i(t_0)$.

The integrator value $z_i(t_0)$ indicates the value of the integral of $\text{TEM}_i$ at time $t_0$, before the time encoding begins.

*Proof of Lemma 4.4.* We will assume that we operate under the assumptions set out in Lemma 4.1.

We start by showing that different constraints imposed by the time encoding machines can be written as in (4.13). In fact, two consecutive spike times $\tau_{i,\ell}$ and $\tau_{i,\ell+1}$ from a machine $\text{TEM}_i$ impose a constraint on the integral of the concerned signal:

$$\int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} y_i(u)\,du = 2\kappa_i\theta_i - \beta_i(\tau_{i,\ell+1} - \tau_{i,\ell}) = \dot{b}_{i,\ell}. \tag{4.16}$$

We define $\Psi_i(t) = \int_{t_0}^{t} y_i(u)\,du$ to be the integral of the signal $y_i(t)$ between $t_0$ and any later time $t$. Given (4.16), and that we know the initial integrator value $z_i(t_0)$, we can compute $\Psi_i(\tau_{i,\ell})$ for any spike time $\tau_{i,\ell}$:

$$\Psi_i(\tau_{i,\ell}) = \int_{t_0}^{\tau_{i,\ell}} y_i(u)\,du = 2\ell\kappa_i\theta_i - \beta_i(\tau_{i,\ell} - t_0) + z_i(t_0). \tag{4.17}$$

We define this quantity to be $b_{i,\ell} := \Psi_i(\tau_{i,\ell})$ and denote the function $F_k(t) = \int_{t_0}^{t} f_k(u)\,du$. We then rewrite the right-hand side of (4.17) in terms of the parametrization of $y_i(u)$:

$$\begin{aligned} b_{i,\ell} &= \int_{t_0}^{\tau_{i,\ell}} \sum_{k=1}^{K} c_{i,k}(\mathbf{y}) f_k(u)\,du \\ &= \sum_{k=1}^{K} c_{i,k}(\mathbf{y}) \int_{t_0}^{\tau_{i,\ell}} f_k(u)\,du \\ &= \sum_{k=1}^{K} c_{i,k}(\mathbf{y}) F_k(\tau_{i,\ell}) \\ &= \left[\mathbf{F}(\tau_{i,\ell})\right] \left[\mathbf{C}^{(\mathbf{y})}\right]_i^T \end{aligned} \tag{4.18}$$

Where we defined $\left[\mathbf{F}(\tau_{i,\ell})\right]$ to be the vector of integrals $F_k(\tau_{i,\ell})$ for $k = 1...K$. We also defined $\mathbf{C}^{(\mathbf{y})}$ to be the matrix of coefficients $c_{i,k}(\mathbf{y})$ as defined in (A4) and $\left[\mathbf{C}^{(\mathbf{y})}\right]_i$ is the $i^{th}$ row containing the coefficients for signal $y_i(t)$.

We further rewrite $\mathbf{C}^{(\mathbf{y})} = \mathbf{A}\mathbf{C}^{(\mathbf{x})}$ (from (A5)) and obtain $\left[\mathbf{C}^{(\mathbf{y})}\right]_i = [\mathbf{A}]_i \mathbf{C}^{(\mathbf{x})}$. We thus obtain:

$$\begin{aligned} b_{i,\ell} &= \left[\mathbf{F}(\tau_{i,\ell})\right] ([\mathbf{A}]_i \mathbf{C}^{(\mathbf{x})})^T \\ b_{i,\ell} &= \left[\mathbf{F}(\tau_{i,\ell})\right] \mathbf{C}^{(\mathbf{x})T} [\mathbf{A}]_i^T \\ b_{i,\ell} &= [\mathbf{A}]_i \mathbf{C}^{(\mathbf{x})} \left[\mathbf{F}(\tau_{i,\ell})\right]^T \end{aligned} \tag{4.19}$$

We can thus reindex the above equations: we let every $n$ correspond to a pair $(i,\ell)$ (where we sometimes write $n(i,\ell)$ to emphasize the correspondance) and let $b_n = b_{i,\ell}$, $\mathbf{g}_n = [\mathbf{A}]_i$ and $\mathbf{h}_n = \left[\mathbf{F}(\tau_{i,\ell})\right]$.

We can now see that the vectors $\mathbf{h}_n$ can be parametrized by one variable $t \in \mathbb{R}$ using a set of functions $h_k$ which satisfy assumption (A9), as stated by Lemma 4.3. Moreover, according to Lemma 4.2, the spike times follow a continuous probability distribution, as required in assumption (A9).

We can also see that the vectors $\mathbf{g}_n$ just defined satisfy assumption (A10) by construction since this is a condition in Lemma 4.4.

Then, under the conditions of Lemma 4.4, one can extract $N_s K$ constraints that satisfy the constraints of Theorem 4.2, thus ensuring perfect reconstruction of the matrix $\mathbf{C}(\mathbf{x})$. □

We now restate Theorem 4.1 before proving it.

**Perfect reconstruction from mixed time encoding**

**Theorem 4.1.** Let $N_c$ signals $y_i(t), i = 1 \cdots N_c$ satisfy assumptions (A4), (A5) and (A6), and their functions $f_k(t)$ satisfy either of (A7.a) or (A7.b) with the corresponding coefficients $c_{j,k}(\mathbf{x})$ being drawn from a Lipschitz continous probability distribution. Now assume $\mathbf{A} \in \mathbb{R}^{N_c \times N_s}$ as defined in (A5) is known and has every $N_s$ rows linearly independent. Then the inputs $y_i(t), i = 1 \cdots N_c$ are exactly determined by the spike times $\{\tau_{i,\ell}, \ell = 1 \cdots L_i\}, i = 1 \cdots N_c$, with probability one, if:

$$\sum_{i=1}^{N_c} \min(L_i - 1, K) > N_s K. \tag{4.20}$$

*Proof of Theorem 4.1.* Using similar notation used for the Proof of Lemma 4.4, we note that the value $b_{i,\ell}$ is not known, when the initial integrator values $z_i(t_0)$ are not known, instead, we know the value of

$$\tilde{b}_{i,\ell} = b_{i,\ell} + z_i(t_0) = [\mathbf{A}]_i \, \mathbf{C}(\mathbf{x}) \left[\mathbf{F}(\tau_{i,\ell})\right]^T + z_i(t_0). \tag{4.21}$$

Continuing in the same logic as before, we let $\tilde{b}_n = \tilde{b}_{i,\ell}$, $\mathbf{g}_n = [\mathbf{A}]_i$ and $\mathbf{h}_n = \left[\mathbf{F}(\tau_{i,\ell})\right]$.

We then obtain

$$\tilde{b}_n = \mathrm{vec}(\mathbf{g}_n \mathbf{h}_n^T)^T \mathrm{vec}(\mathbf{C}(\mathbf{x})) + z_{i_n}(t_0). \tag{4.22}$$

To keep things in matrix form, we first denote $\mathbf{R}$ to be a matrix with rows $\mathbf{r_n} = \mathrm{vec}(\mathbf{g_n}\mathbf{h_n}^T)^T$ and then denote $\tilde{\mathbf{R}}$ to be a matrix with row vectors $\tilde{\mathbf{r}}_\mathbf{n} = [\mathbf{r_n}, e_{i_n}]$ where $e_{i_n}$ is a length-$N_c$ row vector

with value one in the column corresponding to the machine that generated measurement $n$ and zeros otherwise. We also denote the column vector $\tilde{\mathbf{C}}(\mathbf{x}) = [\mathbf{vec}(\mathbf{C}(\mathbf{x}))^T, z_1(t_0), z_2(t_0), \cdots, z_{N_c}(t_0)]^T$.

The measurement therefore satisfies

$$\tilde{b}_n = \tilde{\mathbf{r}}_n \tilde{\mathbf{C}}(\mathbf{x}). \tag{4.23}$$

For the system to be invertible we need $N_s \times (K+1)$ of the vectors $\tilde{\mathbf{r}}_n$ to be linearly independent.

If we satisfy the condition set by Theorem 4.1, we also satisfy the condition set by Lemma 4.4 in (4.15). This means that there are $N_s K$ rows $\mathbf{r}_n$ of $\mathbf{R}$ that are linearly independent. Now let us consider the extension $\tilde{\mathbf{R}}$, the corresponding $N_s K$ rows from $\tilde{\mathbf{R}}$ will still be linearly independent (otherwise we reach a contradiction).

According to the assumptions of the corollary, $\tilde{\mathbf{R}}$ has, in addition to the $N_s K$ rows already mentioned, one extra row $\tilde{\rho}_i$ coming from each $\text{TEM}_i$.

We would therefore like to check if there exist $N_s K + N_c$ coefficients $p_{n(i,\ell)}$ such that

$$\begin{aligned} \sum_{i=1}^{N_c} \sum_{\ell=1}^{L_i} p_{n(i,\ell)} \tilde{r}_{n(i,\ell)} &= 0 \\ \sum_{i=1}^{N_c} \sum_{\ell=1}^{L_i} p_{n(i,\ell)} \begin{bmatrix} r_{n(i,\ell)} & e_i \end{bmatrix} &= 0 \end{aligned} \tag{4.24}$$

Because the $e_i$'s are orthogonal, the above constraint can be translated to:

$$\begin{aligned} \sum_{\ell=1}^{L_i} p_{n(i,\ell)} \tilde{r}_{n(i,\ell)} &= 0, \quad \text{and} \\ \sum_{\ell=1}^{L_i} p_{n(i,\ell)} &= 0, \quad \forall i = 1, \cdots, N_c \end{aligned} \tag{4.25}$$

Because $L_i \leq K+1$ for $\text{TEM}_i$, and the first $L_i - 1$ rows (other than $\tilde{\rho}_i$) from $\text{TEM}_i$ have full rank equal to $L_i - 1$ (according to Lemma 4.4), there is at most one solution for the $p_{n(i,\ell)}$'s. This solution will depend on the $r_{n(i,\ell)}$'s which follow a continuous probability distribution (given that the $\tau_{i,\ell}$'s also follow a continuous probability distribution). Therefore the $p_{n(i,\ell)}$'s almost surely don't satisfy the second condition of (4.25), and the measurement vectors of $\tilde{\mathbf{R}}$ are almost surely linearly independent, making the system uniquely invertible with probability one. □

# 5 How Asynchronous Events Encode Video

***Recommended Reading:*** Sections 2.1, 2.2, 2.3, 5.
*The content in this chapter is adapted from Adam et al. (2022, 2021).*

## 5.1 Introduction

Event-cameras forego the frame-based, clock-driven limitation that standard cameras abide by. To record video, they use a set of pixels, each of which emits an event whenever its input exhibits sufficient change (Liu and Delbruck, 2010; Gallego et al., 2019; Brandli et al., 2014). Event-cameras have recently been gaining traction among researchers (Kim et al., 2016; Mueggler et al., 2017; Rebecq et al., 2018; Gehrig et al., 2020; Cordone et al., 2021; Duwek et al., 2021; Zheng et al., 2021; Zhao et al., 2021) by virtue of their power efficiency, higher dynamic range and reduced motion blur compared to standard cameras.

Simulated output of an event camera can be used to reconstruct videos with a frame rate of over $5000$ frames per second while running in real-time (Rebecq et al., 2019). Event-based sensors also have an impressively low data consumption. For example, the dynamic and active pixel vision sensor (DAVIS) has $240 \times 180$ sensors and operates at 5-14mW of power depending on the activity of the sensors (Brandli et al., 2014).

Power reduction also extends beyond the sensing devices themselves, as the output of these sensors can be processed using event-based processors. These are neuromorphic chips that contain units which act like neurons and output spikes, and have reduced power consumption compared to standard processors (Merolla et al., 2014; Akopyan et al., 2015; Davies et al., 2018). Studies have shown for example that live-streamed event data can be used to recognize hand gestures using 200mW of power if neuromorphic hardware is used (Amir et al., 2017).

While the technology behind event-based sensing and neuromorphic hardware gains in popularity, theoretical insight and guarantees are advancing at a slower pace (Liu et al., 2019). One key difficulty with understanding event-based vision is the fact that different sensors

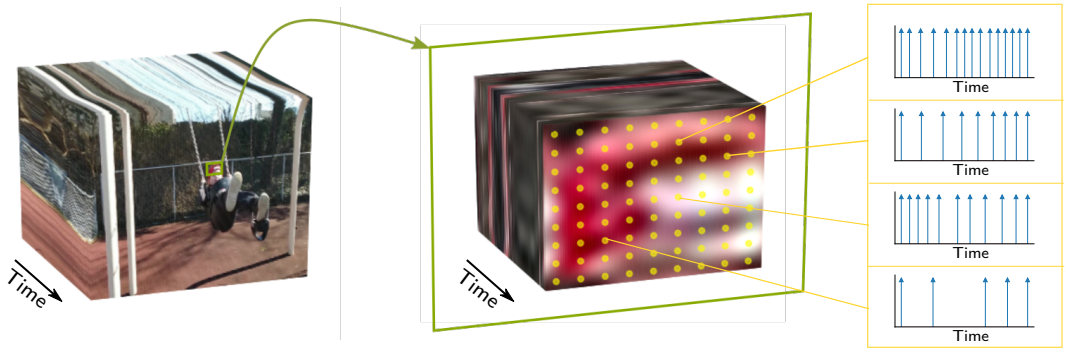



Figure 5.1 – Vision setup: we assume that we have an array of spiking devices, such as TEMs, each of which is observing a scene at a particular location. The input to the sensor at this location is a time varying signal and the sensor will output a stream of spikes, the timing of which is dependent on the input. On the left, we show the projection of the scene which is being observed. In the middle, we show a patch of this scene, which is interpolated under bandlimited periodic assumptions, with an overlay of event-based sensors shown in yellow. To its right, we zoom in to view the spiking output of some of the sensors. The video used is taken from the Need for Speed dataset (Galoogahi et al., 2017).

emit events at different times, making it difficult to build frames and thus difficult to perform a reconstruction.

In this chapter, we will show that the asynchrony of events across pixels actually consitutes an advantage of event-based vision over frame-based vision and allows an encoding of information that is more sample-efficient.

We use time encoding machines (TEMs) as a model for event sensors to understand how these sensors encode their data, and we build on results from Chapter 4 to understand time encoding of video. We will see that using event sensors or TEMs results in streams of events that are asynchronous across sensors, thus allowing for richer information content than in the frame-based approach. To do so, we assume our scene can be modeled by a periodic bandlimited function in three dimensions (two spatial and one temporal dimension) and will see how the setup results in an entanglement between spatial and temporal resolution. In the frame based case, spatial and temporal resolution are uniquely and respectively defined by the pixel gridding and frame rate, but in the case of event-based vision, we will see that the temporal resolution is also affected by the pixel gridding. As a result, both temporal and spatial resolution in event-based vision can be increased by increasing the number of pixels.

## 5.2 Background

We use a time encoding machine (TEM), as defined in Definition 2.2 as a model for an event-based sensor (Lazar and Tóth, 2003), where the TEM follows an integrate and-fire mechanism and encodes its input using *times* that are dependent on the input itself, depicted

in Fig. 2.2.

We adopt the integrate-and-fire model for the event-based sensor because it is well understood and results on sampling and reconstruction are widely available. However, the results can be extended to different types of time encoding machines which filter the signal, compare it to an output and emit events when the comparison yields a match (Gontier and Vetterli, 2014). A differentiate-and-fire model, for example, can be written similarly, as well as more complicated event-based sensing models.

In this chapter, we tackle the problem of time encoding video, achieved by replacing pixels in a standard frame-based camera by event-based sensors or, in this case, time encoding machines, as depicted in Fig. 5.1. Video time encoding machines have been examined before, and results on perfect reconstruction have already been established (Lazar and Pnevmatikakis, 2011; Lazar and Zhou, 2014). However, previous work relied on applying linearly independent filters to a bandlimited video before it is processed by the TEMs. Here, we propose a filter-less approach where the scene is encoded without preprocessing. Moreover, we clarify a dependency between spatial sampling density and temporal resolution that was not apparent before.

To do so, we build on Theorem 4.1 from Chapter 4, on multi-channel time encoding of mixed low-dimensional signals. When $N_c$ signals observed by $N_c$ TEMs have a lower dimensional representation and can be written as a linear combination of $N_s \leq N_c$ signals, as depicted in Fig. 4.1, the low dimensionality of the input can be used to reduce the number of spike pairs needed to ensure reconstruction of the input.

This results shows that a signal with low dimensional representation can be reconstructed from its time encoding if the number of *pairs* of spikes scales with the number of parameters in the *low dimensional* space rather than the high dimensional space. In total, one requires $N_s K \leq N_c K$ linearly independent constraints where $N_s$ is the number of signals in the underlying low dimensional representation and $N_c$ is the number of signals seen by the TEMs.

We will show that we can formulate the problem of time encoding video to fit the framework presented in Assumptions (A4)-(A6), allowing us to use Theorem 4.1.

## 5.3 Time Encoding Video: Theory

### 5.3.1 Video Model

To tackle the problem of time encoding video, we model video as a continuous signal $y(d^{(1)}, d^{(2)}, t)$ which varies in three dimensions: two spatial dimensions $d^{(1)}$ and $d^{(2)}$ and one temporal dimension along $t$. Such a signal is then sampled using a collection of time encoding machines.

For the remainder of this chapter, we assume that $y(d^{(1)}, d^{(2)}, t)$ is periodic bandlimited in all dimensions. Such a signal can be described by a finite number of parameters and can thus potentially be fully characterized by a finite number of "measurements". In more mathematical terms, we assume that the input signal to an event-based camera or to a time encoding camera can be written:

$$y(d^{(1)}, d^{(2)}, t) = \sum_{k_0=-K_0}^{K_0} \sum_{k_1=-K_1}^{K_1} \sum_{k_2=-K_2}^{K_2} c_{k_0,k_1,k_2}(y) \exp\left(\mathbf{j}2\pi\left(\frac{tk_0}{T} + \frac{d^{(1)}k_1}{D^{(1)}} + \frac{d^{(2)}k_2}{D^{(2)}}\right)\right), \tag{5.1}$$

where the $c_{k_0,k_1,k_2}(y)$'s denote the 3D Fourier series coefficients of $y(d^{(1)}, d^{(2)}, t)$. Note that we assume that $y(d^{(1)}, d^{(2)}, t)$ has $(2K_0+1) \times (2K_1+1) \times (2K_2+1)$ of these coefficients with periods $T$, $D^{(1)}$ and $D^{(2)}$ in the time dimension and in the first and second space dimensions, respectively.

While the periodic bandlimited model choice may seem restrictive at first sight, note that frame-based video has limited frame rate and finite pixel separation, thus inherently assuming that the input is "smooth enough" between temporal and spatial samples. Moreover, frame-based video records data over finite amounts of time and limited space, and assuming periodicity in the input is a natural way to deal with finite sampling windows.

We have described the signal model, we now focus on the measurement approach. We assume that the scene is recorded using integrate-and-fire time encoding machines. Each TEM$_i$ observes a specific direction $\mathbf{d}_i = (d_i^{(1)}, d_i^{(2)})$ in $2D$ space, where $d_i^{(1)}, d_i^{(2)} \in \mathbb{R}$, and fires corresponding spikes $\tau_{i,\ell}$.

Then, the input $y_i(t)$ observed by TEM$_i$ is

$$y_i(t) = y(d_i^{(1)}, d_i^{(2)}, t). \tag{5.2}$$

For example, the pixels can be made to lie on a uniform grid, as illustrated in Fig. 5.1.

### 5.3.2 Uniqueness of Video Time Encoding

Assuming the input is periodic bandlimited allows our problem to fit within the framework we presented in Section 5.2, where low-dimensional signals are mixed and time-encoded.

**Mixed time encoding assumptions are respected**

**Lemma 5.1.** The signals $y_i(t)$, as defined in (5.2), satisfy assumptions (A4)-(A6), with $N_s = (2K_1+1)(2K_2+1)$.

*Proof.* We first define proxy signals $x^{(k_1,k_2)}(t)$, with $k_1 \in \{-K_1, \cdots, K_1\}$ and $k_2 \in \{-K_2, \cdots, K_2\}$:

$$x^{(k_1,k_2)}(t) = \sum_{k_0=-K_0}^{K_0} c_{k_0,k_1,k_2}(y) \exp\left(\mathbf{j}2\pi\left(\frac{tk_0}{T}\right)\right). \tag{5.3}$$

With this definition, we can write the input to each $\text{TEM}_i$ as

$$y_i(t) = \sum_{k_1=-K_1}^{K_1} \sum_{k_2=-K_2}^{K_2} x^{(k_1,k_2)}(t) \exp\left(\mathbf{j}2\pi\left(\frac{d_i^{(1)}k_1}{D^{(1)}} + \frac{d_i^{(2)}k_2}{D^{(2)}}\right)\right). \tag{5.4}$$

We further define an index $j$ that has a unique correspondence to $k_1$ and $k_2$: $j := (k_1 + K_1 + 1) \times (2K_2 + 1) + (k_2 + K_2 + 1)$. The index $j$ takes on values $1, \cdots, N_s$ where $N_s = (2K_1+1)(2K_2+1)$, and we write $k_1(j)$ and $k_2(j)$ the unique indices $k_1$ and $k_2$ that define $j$, so that

$$x_j(t) = x^{(k_1(j),k_2(j))}(t). \tag{5.5}$$

We further define a matrix $\mathbf{A}$ with entries

$$a_{i,j} = \exp\left(\mathbf{j}2\pi\left(\frac{d_i^{(1)}k_1(j)}{D^{(1)}} + \frac{d_i^{(2)}k_2(j)}{D^{(2)}}\right)\right). \tag{5.6}$$

Given these definitions, we see that $y_i(t)$ can be written:

$$y_i(t) = \sum_{j=1}^{(2K_1+1)\times(2K_2+1)} a_{i,j} x_j(t). \tag{5.7}$$

As a result, and given that the inputs $y_i(t)$ are periodic bandlimited functions that are input to TEMs, assumptions (A4)-(A6) are satisfied, with the entries of mixing matrix $\mathbf{A}$ defined by the known directions $\mathbf{d}_i$ the TEMs or pixels are observing, as described in (5.6). □

A similar result to Theorem 4.1 can thus be established:

**Perfect reconstruction of video time encoding**

**Corollary 5.1.** Assume a signal $y(d^{(1)}, d^{(2)}, t)$ is defined as in (5.1) with the coefficients $c_{k_0,k_1,k_2}(y)$ being drawn from a Lipschitz continuous probability distribution. Further assume $y(d^{(1)}, d^{(2)}, t)$ is sampled using $N_c \geq N_s = (2K_1+1)(2K_2+1)$ TEMs observing directions $\mathbf{d}_i$ such that the resulting matrix $\mathbf{A}$ with entries defined in (5.6) has every

$N_s$ rows linearly independent. Then, if each TEM$_i$ fires $L_i$ spikes and

$$\sum_{i=1}^{N_c} \min(L_i - 1, K) > N_s K, \tag{5.8}$$

where we defined $K = (2K_0 + 1)$, the scene $y(d^{(1)}, d^{(2)}, t)$ can be perfectly reconstructed from its spike times.

Before we dive into the assumptions for this theorem, namely the condition on the mixing matrix $\mathbf{A}$, let us understand the implications. First recall that the reconstruction of any input signal to a TEM can be achieved because *pairs* of consecutive spike times provide linear constraints on the input as explained in (2.3). According to the result in Corollary 5.1, the number $N_c$ of TEMs used to encode an input signal $y(d^{(1)}, d^{(2)}, t)$ does not affect the number of pairs of spike times needed to ensure a unique characterization of the scene, as long as the matrix $\mathbf{A}$ has every $N_s$ rows linearly independent. It is rather the complexity $N_s K = \prod_{n=0}^{2} (2K_n + 1)$ of the scene itself, determined by the number of Fourier series coefficients, which dictates the required number of spike time pairs.

We now tackle the requirement that the mixing matrix $\mathbf{A}$ have every $N_s = (2K_1 + 1)(2K_2 + 1)$ rows linearly independent. In Adam et al. (2022), we showed that a $(2K_1 + 1) \times (2K_2 + 1)$ grid of equally spaced pixels results in a matrix $\mathbf{A}$ that is full rank and therefore fulfills the requirement. However, this property cannot be extended to uniform grids with more pixels than in the sufficient gridding case. Fortunately, the requirement on matrix $\mathbf{A}$ having every $N_s$ rows linearly independent is merely a *sufficient* requirement for perfect reconstruction rather than a necessary one. We will show in the upcoming simulations that one can still achieve perfect reconstruction under condition (5.8) even if this requirement on the mixing matrix is not strictly obeyed.

### 5.3.3 Reconstruction Algorithm

As previously mentioned, each pair of spike times emitted by each TEM$_i$ provides a linear constraint on the input to this TEM$_i$. We define a measurement vector $\{\dot{b}_{i,\ell}\}_{\ell=1\cdots L_i}$ associated with each TEM$_i$ where

$$\dot{b}_{i,\ell} := \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} y_i(u)\, du, \tag{5.9}$$

where $\dot{b}_{i,\ell}$ is known and as described in (2.3). We can show that our problem can be rewritten as a rank-one sensing problem (Pacholska et al., 2020) and, with some vectorization operations, the right hand side of (5.9) can be rewritten to obtain

$$\dot{b}_{i,\ell} = \mathrm{vec}\left(\mathbf{a}_i \left[\dot{\mathbf{F}}_\ell^{(i)}\right]^T\right) \mathrm{vec}\left(\mathbf{C}(\mathbf{x})\right), \tag{5.10}$$

where $\mathbf{a}_i$ denotes a row of matrix $\mathbf{A}$, vec() denotes the vectorization operation and

$$\left[\dot{\mathbf{F}}_\ell^{(i)}\right]_k = \int_{\tau_{i,\ell}}^{\tau_{i,\ell+1}} \exp\left(\mathbf{j}\frac{2\pi(k-K_0-1)}{T}u\right) du.$$

Therefore, as recovering the video is equivalent to recovering the coefficients $\mathbf{C}(\mathbf{x})$, and as $\mathbf{A}$ and $\left[\dot{\mathbf{F}}_\ell^{(i)}\right]$ are known, one can recover the video by simply inverting the linear system in (5.10). The system is full rank and uniquely invertible if the conditions of Corollary 5.1 are met.

### 5.3.4 Interpretation of Results

The observation in Corollary 5.1 provides intuition on how the parameters of the input signal are recovered. Essentially, our result states that if there are more pixels than needed for full spatial resolution (i.e. $N_c \geq (2K_1+1)(2K_2+1)$), recovery occurs when there are sufficient spike pairs coming from *all* TEMs or pixels, provided that these spike pairs yield non-redundant information. In fact, our statement is supported by two components of the condition in (5.8): (1) the min term ensures that only "useful" information is counted from each TEM (given that each TEM has a limited information capacity determined by the complexity of its input) and (2) the summation ensures that the information gathered from the different machines is used collaboratively.

An interesting effect ensues: a machine that spikes too rarely can be compensated for by having other machines spike more often or simply by having *more* machines, and one can thus always improve signal reconstruction by increasing the number of TEMs or pixels used, because the emitted spike times will almost surely be asynchronous.

In some sense, spatial sampling density now has not only an effect on spatial resolution, but on temporal resolution as well, an effect that does not exist in classical frame-based video. This effect occurs because TEMs emit spikes at different times, thus collecting information about their inputs at different times and providing linearly independent constraints on the input. In the frame-based scenario, on the other hand, each frame is captured at *the same time* and increasing the pixel grid size cannot improve temporal resolution.

Therefore, if we have TEM-like receptors or sensors that have a limited spiking rate, spatial and temporal resolution can be regained by adding more sensors that observe new directions. In the frame-based scenario, spatial and temporal resolution are independendent of eachother: the former is defined by the pixel grid used in each frame and the latter is defined by the frame rate used to capture the video. On the other hand, employing event-based vision creates an entanglement between spatial and temporal resolution and the latter can also be improved by increasing the size of the pixel grid, as we will show in the upcoming simulations.

## 5.4 Time Encoding Video: Experiments

In the previous section, we discussed the relationship between spatial sampling density and temporal resolution: in event-based vision, the former influences the latter, whereas in frame-based video, the two quantities are independent.

We would like to show this effect through simulations. As mentioned, using a grid of uniformly spaced pixels and increasing the number of pixels beyond the necessary number $(2K_1+1)(2K_2+1)$ violates the condition on $\mathbf{A}$ in Corollary 5.1. However, we will show that under the same assumptions, the predicitions of the corollary can still be realized.

We time encode a patch of a video recorded with a standard frame-based camera from the Need for Speed dataset (Galoogahi et al., 2017). The patch is originally 9 pixels high, 9 pixels wide and 9 frames long and we therefore assume that it is periodic bandlimited with $9\times9\times9$ Fourier series coefficients (where we assume $K_0=K_1=K_2=4$). This assumption allows us to have a continuous model for the video and to perform time encoding of a smooth and continuous input signal.

We sample the smoothly varying patch using different numbers of time encoding machines with different spiking rates and evaluate the result under the different assumptions. Different spiking rates can be achieved by manipulating the threshold of the TEMs: the lower the threshold, the higher the number of spikes emitted over a certain time. Note that emitting more spikes requires more power, so the choice of the threshold always entails a tradeoff between power consumption and reconstruction error.

We place TEMs, for example, at the yellow dots in Fig. 5.1 in a 9×9 grid of time encoding machines. We will show in our experiments that this is the minimum number of TEMs required to achieve perfect reconstruction.

We will also show how we can use more TEMs in the spatial dimensions to obtain better resolution in the time dimension. This will not necessarily be the case the other way around: more sampling in time does not always provide improved spatial frequency resolution. To achieve this, we run two experiments.

**Higher spatial density increases temporal resolution**

**Example 5.1.** In the first experiment, depicted in the left part of Fig. 5.2, we evaluate the reconstruction performance when the grid of TEMs has more or fewer TEMs. When there are at least as many TEMs as necessary (i.e. a $9\times9$ or $9\times15$ grid of TEMs), we see that the reconstruction error indeed decreases sharply when the condition for Corollary 5.1 is achieved, as indicated by the vertical orange line. When there are fewer TEMs than necessary (a $9\times5$ grid of TEMs), the spatial density is not sufficient and perfect reconstruction can never be reached as the system will always

be underdetermined due to the too few number of sensors.

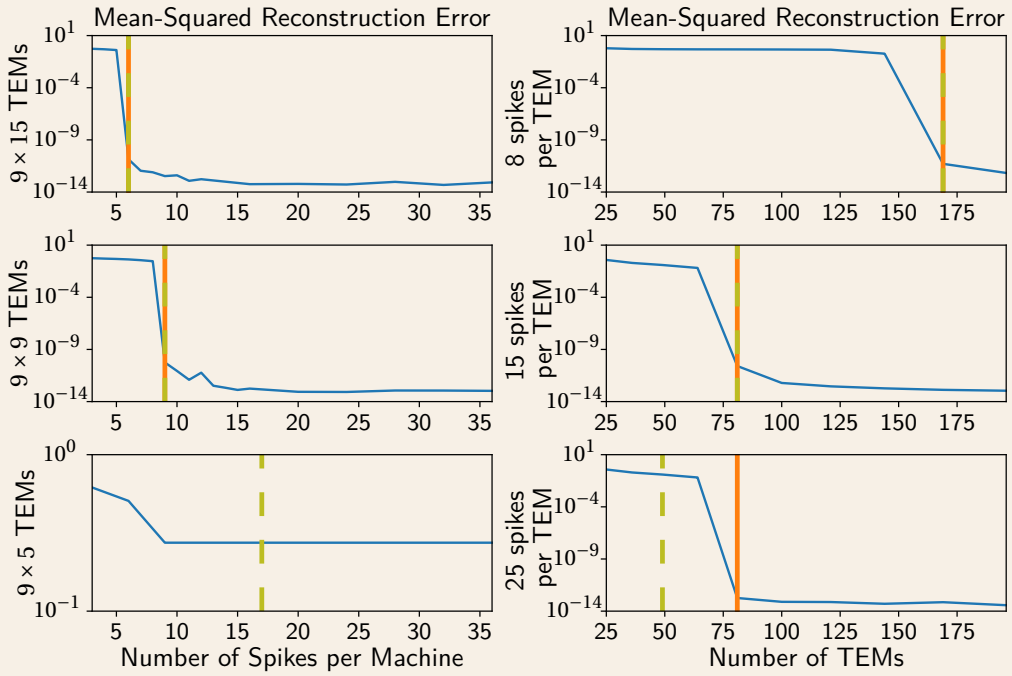



Figure 5.2 – Mean-squared reconstruction error with varying number of pixel TEMs and varying number of spike pairs per TEM. The original video has $9 \times 9 \times 9$ Fourier series coefficients that we wish to recover. On the left-hand side, we study the evolution of the error as the number of spikes increases, assuming we time encode the video using grids of uniformly spaced TEMs with sizes that are decreasing from top to bottom, and where the second row assumes the minimal number of TEMs needed to reconstruct the video. On the right-hand side, we study the evolution of the error as the number of TEMs increases for numbers of spikes emitted per machine which are increasing from top to bottom, and where the second row assumes the maximal useful spiking rate per TEM. For each plot, the vertical orange line marks the point starting from which the condition for Corollary 5.1 is satisfied. The dashed green line marks the point starting from which we have more constraints than unknowns, without accounting for linear independence, i.e. counting the number of obtained spike pairs rather than estimated the quantity on the left hand side in (5.8). Note that $N$ spike pairs corresponds to $N+1$ spikes.

In the second experiment, depicted in the right part of Fig. 5.2, we evaluate the reconstruction performance given fixed spiking rates of the TEMs. When the spiking rate is at most the maximal rate per TEM (i.e. 9 spikes per TEM allow each machine to perfectly resolve its input), the reconstruction error decreases with the increase of number of TEMs used, when the condition for Corollary 5.1 is achieved, as indicated

by the vertical orange line. When the spiking rate increases beyond the useful rate (in this case, we have 15 spikes per machine), the higher spiking rate only provides redundant information in the noiseless case and the profile of the reconstruction error resembles that in the case of a spiking rate of 9.

With these experiments, we see how increased spatial density can increase overall reconstruction (including temporal resolution), even if each TEM has a limited spiking rate.

## 5.5 Conclusion

We have shown how to use time encoding to understand event-based video and have consequently demonstrated that event-based vision has an advantage over frame-based vision when it comes to sample complexity.

This advantage arises because event-based cameras emit streams of events from their pixels. As these events are asynchronous across pixels, they provide information about the input that is almost surely linearly independent. This uncovers a relationship between spatial sampling density and temporal resolution in event-based vision. As sensors emit events at different times, increasing the number of sensors used in event-based video increases both spatial and temporal resolution, without requiring a higher firing rate per sensor.

We have seen how spikes or events can be a sample-efficient way of encoding video and we know that they can be implemented in a power-efficient manner. While spikes are more difficult to treat compared to uniform samples, their asynchronous and all-or-none nature provide avenues for improvement over clocked systems, as will become clearer when we investigate spiking neural networks in Chapter 6.

# 6 A Time Encoding Approach to Training Spiking Neural Networks

***Recommended Reading:*** Sections 2.1, 2.2, 2.3.
*The content in this chapter is adapted from Adam (2022).*

## 6.1 Introduction

We have seen the benefits of the asynchrony of spike times that naturally occurs in spiking integrate-and-fire devices. This chapter additionally shows how the all-or-none nature of spikes allows training of spiking neural networks in a layer-by-layer fashion, by enforcing constraints.

Spiking Neural Networks (SNNs) transform their input using nonlinearities that follow simple models of spiking neurons as depicted in Fig. 6.1. SNNs are gaining in popularity for a number of reasons. They can provide insight on information processing in the brain and can guide experimental studies to validate or refute this insight. They can inspire new learning algorithms for artificial neural networks (ANNs), as SNNs are often made to rely on local operations, thus reducing computational complexity and power load. The hardware needed to implement them can also be very power efficient thanks to the all-or-none nature of SNNs' spiking output.

Despite these motivations, advances in SNNs are quite far from those achieved in the realm of ANNs. While neuromorphic hardware is available (Davies et al., 2018; Akopyan et al., 2015; Indiveri et al., 2011) and numerous learning algorithms have been developed (Bohte et al., 2002; Neftci et al., 2019; Comsa et al., 2020), implemented on these chips and tested on various tasks (Cordone et al., 2021; Davies et al., 2021), the tasks that SNNs are trained on are still much simpler than those tackled by ANNs (Comsa et al., 2020; Wunderlich and Pehle, 2021; Ma et al., 2021), because training SNNs on more complex data seems prohibitively difficult.

The difficulty in training SNNs lies in the discontinuity of the function applied by spiking

neurons. These neurons fire spikes when their input reaches a threshold—a behavior which results in a discontinuity, posing problems when derivatives are required for backpropagation. Current approaches to training SNNs avoid this discontinuity in different ways, whether by using spike rates instead of times (Boerlin et al., 2013), using surrogate gradients (Neftci et al., 2019), or using spike times as a continuous variable with respect to which differentiation is done when performing backpropagation (Bohte et al., 2002; Comsa et al., 2020; Wunderlich and Pehle, 2021).

An under-explored approach, however, comes from the theory behind time encoding. This approach can provide a different perspective on learning SNNs which not only avoids the discontinuity mentioned above, but bypasses the backpropagation algorithm altogether.

TEMs can be used to encode and decode weighted sums of bandlimited input, following an architecture that is reminiscent of a layer of a feedforward network, as we showed in Chapter 4. TEMs can also encode and reconstruct spike streams that are passed through various filters (Alexandru and Dragotti, 2019; Rudresh et al., 2020; Kamath et al., 2021), such as the alpha synaptic function which is often considered in simpler models of neurons (Hilton et al., 2021a). These filters are needed both for biological similarity but also for reconstruction to be possible. Many of these results are possible because the recovery from time encoding can be formulated as a problem of linear constraint satisfaction (Thao and Rzepka, 2020).

In this chapter, we bridge the gap between TEMs and SNNs by formulating the training problem of SNNs as a constraint satisfaction problem. This formulation allows us to understand the power of spikes compared to traditional ANN nonlinearities that are graded (i.e. have varying amplitudes) and synchronous.

We assume that we want an SNN to learn to associate certain inputs with corresponding output spike streams, a task which we further assume to be perfectly learnable, in the noiseless case, for the particular network used. For a single-layer SNN, when using the TEM formulation, we will see that the SNN's weights can be learnt by solving for one set of linear equations. We can then build on this result to train two-layer SNNs using the TEM formulation. In the latter scenario, a key ingredient at play is the all-or-none and asynchronous nature of the spikes within an SNN, which will allow layers to be trained one after the other.

We evaluate our method by examining the trained weights of SNNs and their distance to a set of known ground-truth weights. This is different from standard approaches to evaluating neural networks, where emphasis is placed on the task rather than the weights. However, we focus on the weights becausue it is actually possible for weights to be recovered exactly if an SNN is trained as we propose.

The work in this chapter is a first step towards training SNNs using the TEM paradigm, allowing one to bypass traditional backpropagation algorithms, and providing a new tool to tackle this problem.

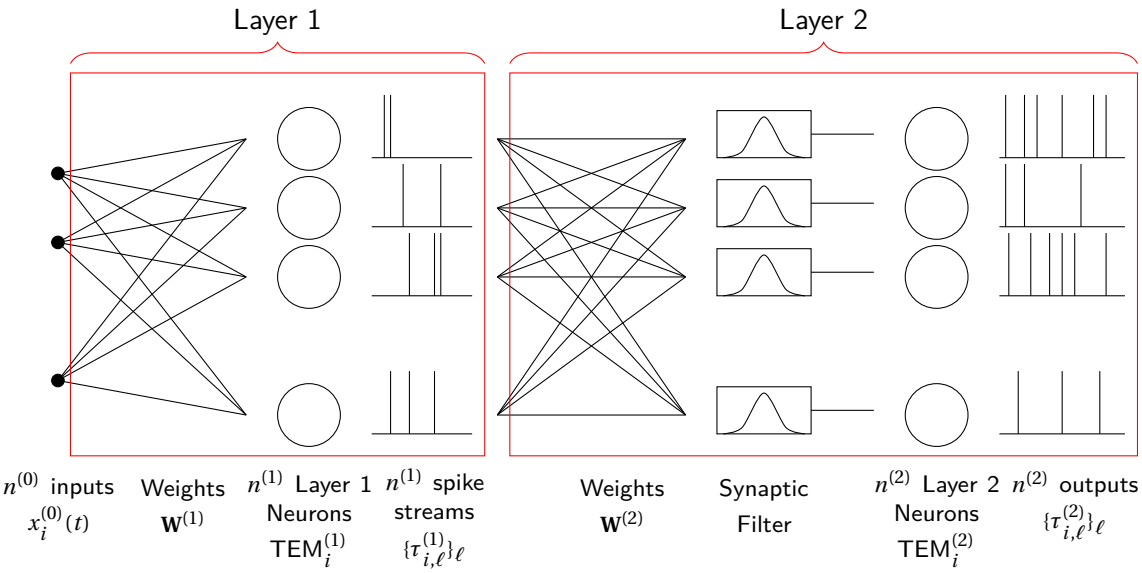



Figure 6.1 – A two-layer spiking neural network where each of the individual neurons (marked by circles) is a TEM as defined in 2.2. When learning such an SNN, we assume that we know the synaptic filters as well as the parameters of the TEMs, that we are given a set of examples of input-output pairs and that we wish to learn the weights of the network $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$.

## 6.2 Training SNNs: Background

There is ongoing debate about the modality the brain uses to encode information: is it spike times or spike rates? Of course, this debate also extends to SNNs, where the question targets the quantity used to perform learning.

Evidence has been found for both spike rate and spike time coding in the brain, and the preference in coding scheme depends on the area in question. It is argued, for example, that some tasks are performed too quickly for them to depend on the computation of spike rates and rather rely on quantities such as the time to first spike (VanRullen et al., 2005).

Following these two schools of thought when it comes to brain activity, we also see two trends for learning rules of SNNs.

In the spike rate paradigm, a neuron is often assumed to emit spikes according to a Poisson process with a varying and information-carrying rate (which is its input) and operations are done by calculating and using the rate of the neuron over moving windows as done e.g. in Boerlin et al. (2013). This follows a clock-based approach, where updates are made after uniform time steps. It allows one to use classical approaches to training deep neural networks, such as error-backpropagation, which uses the gradient of the loss function with respect to every weight or parameter to update said parameter[1].

[1]Note that the use of backpropagation might not be desired. Often, when training SNNs one wishes to achieve some locality in the computations and the backpropagation algorithm is far from local.

In the spike timing paradigm, updates are made in (almost) continuous time and spikes are sent to receiving neurons as they are emitted[2]. Because spikes create sharp transitions in the outputs of nodes, discontinuities arise in the gradient of the loss function.

On one hand, one can tackle this by finely discretizing all neuron currents, and training an SNN by reformulating it as a recurrent neural network and using surrogate gradients (Neftci et al., 2019).

On the other hand, there exist different spike-timing based training approaches to SNNs, which bypass this discontinuity altogether. For example, single-layered SNNs can be learned using *spike timing dependent plasticity* (Pfister and Gerstner, 2006). For learning rules for deeper SNNs, we point to the SpikeProp algorithm which does not consider currents as a *discontinuous* function of weights, but rather considers spike times as a *continuous* function of these weights and performs backpropagation on this basis (Bohte et al., 2002). A similar line of work was also explored by Comsa et al. (2020) and Wunderlich and Pehle (2021), often prioritizing quantities such as the *time to first spike*, i.e. the timing of the first spike of an output neuron.

We will next see how results from time encoding can inspire learning algorithms of a different kind.

## 6.3 Learning a Single-Layer SNN

### 6.3.1 Input Processing through a Single-Layer SNN

We consider the training of an SNN from a constraint satisfaction perspective.

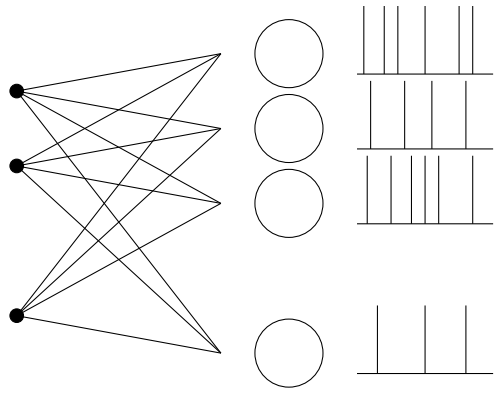
To begin tackling this problem, we first consider the case of a single-layered SNN, i.e. we focus on the first layer in Fig. 6.1. The SNN has nonlinearities which are modeled by integrate-and-fire neurons, i.e. TEMs as defined in Definition 2.2 and with streams of Diracs as an output, and are assumed to have a known set of parameters $\kappa$, $\theta$ and $\beta$.

We would like to recover an unknown weight matrix $\mathbf{W}^{(1)}$ and to do so, we are given a set of examples. Each of these examples is composed of an input $\mathbf{x}^{(0)}(t) = \left[x_1^{(0)}(t), \cdots, x_{n^{(0)}}^{(0)}(t)\right]^T$ which is a collection of $n^{(0)}$ signals, and of the corresponding target output spike streams $\{t_{i,\ell}^{(1)}\}_\ell$ for each of the $n^{(1)}$ neurons $\text{TEM}_i^{(1)}$.

Given the known input and output of each example, we would like to find the weights $\mathbf{W}^{(1)}$ such that the SNN generates the correct output for each input of the examples.

Notice the relationship between the problem we are solving here and the one we solved in

[2]We say that time is almost continuous, because many simulators for spiking neural networks actually use a clock-based approach (for which our computers are well suited) with a very small discretization step.



| | $n^{(0)}$ inputs $x_i^{(0)}(t)$ | weights $\mathbf{W}^{(1)}$ | $n^{(1)}$ neurons $\text{TEM}_i^{(1)}$ | $n^{(1)}$ outputs $\{\tau_{i,\ell}^{(1)}\}_\ell$ |
|---|---|---|---|---|
| **Chapter 4** | unknown | known | known | known |
| **Chapter 6** | known | unknown | known | known |

Figure 6.2 – Structure of a single-layer feedforward spiking neural network, where $x_i^{(0)}$ are the inputs, $\mathbf{W}^{(1)}$ is a weight matrix that mixes the input, $\text{TEM}_i{}^{(1)}$ are the spiking neurons that apply the nonlinearities, and the streams of Diracs at locations $\{\tau_{i,\ell}\}_{\ell=1}^{Li}$ are the outputs.

Chapter 4. The processing steps that an input vector to the network goes through are the same. However, in Chapter 4, we looked to recover the input itself from its time encoding, whereas, here, we look to recover the weights from the time encoding, as depicted in Fig. 6.2

### 6.3.2 Recovery of Weights through the Translation from Spikes to Linear Constraints

The recovery of the weights $\mathbf{W}^{(1)}$ can be performed in a row by row fashion. In fact, the timing of the spikes $\{\tau_{i,\ell}^{(1)}\}_\ell$ of $\text{TEM}_i^{(1)}$ provide linear constraints on the input $\sum_j w_{i,j} x_j^{(0)}(t)$ to $\text{TEM}_i^{(1)}$ and therefore on the weight matrix row $w_{i,:}$

$$b_{i,\ell} = \int_{\tau_{i,\ell}^{(1)}}^{\tau_{i,\ell+1}^{(1)}} \sum_j w_{i,j} x_j^{(0)}(t) = \sum_j w_{i,j} \int_{\tau_{i,\ell}^{(1)}}^{\tau_{i,\ell+1}^{(1)}} x_j^{(0)}(t), \tag{6.1}$$

where $b_{i,\ell} = 2\kappa\theta - \beta(\tau_{i,\ell}^{(1)} - \tau_{i,\ell+1}^{(1)})$ as mentioned in (2.3).

Given that the inputs $\mathbf{x}^{(0)}(t)$ are known, their integrals needed for (6.1) are also known. It is also possible to further show that the spike times will almost surely provide constraints that are linearly independent, if one assumes, for instance, that the $\mathbf{x}^{(0)}(t)$ are periodic bandlimited signals and have their coefficients drawn from a Lipschitz-continuous probability distribution, along similar lines as those in the proof of Lemma 4.2.

Notice that, in (6.1), we assume we have one example with corresponding input and output spike stream, but this can be extended to include multiple examples, by concatenating the constraints from each example on $\mathbf{W}^{(1)}$ into a single measurement matrix and vector.

### 6.3.3 Simulations

We assume that we would like to train an SNN that has a known number of inputs and outputs, using a varying number of examples.

To do so, we create a *Simulator SNN* with weights $\mathbf{W}^{(1)}$ drawn uniformly at random and process a set of input signals through the Simulator SNN to obtain our *ground truth* output spike streams. The input signals are here assumed to be periodic bandlimited, with Fourier series coefficients drawn uniformly at random.

Then, we learn the weights $\hat{\mathbf{W}}^{(1)}$ of a *Learned SNN* that match these examples, by finding the least squares solution to (6.1), which can either be done in one shot or through gradient descent.

Note that the Learned SNN has the exact same architecture as the Simulator SNN, and only the weights $\hat{\mathbf{W}}^{(1)}$ are being learned, so that we are essentially simulating a teacher-student setup where teacher and student are single-layer feedforward SNNs with the same number of inputs and outputs.

To evaluate the success of our learning, we look at the mean squared error of our estimate of the weights $\hat{\mathbf{W}}^{(1)}$, as the number of examples, the duration of exposure, and the noise level changes. This is not the standard evaluation metric when training neural networks, but it is a pertinent measure here as the weights should be recoverable exactly.

**Learning a single-layer SNN with varying number of examples and exposure**

**Example 6.1.** We assume our network has $n^{(0)} = 20$ inputs and $n^{(1)} = 5$ outputs. We study the reconstruction error on the weights after training using a varying number of examples and varying exposure duration per example, in Fig. 6.3. By exposure duration, we mean the time over which the Simulator SNN is exposed to the input and allowed to spike. The higher the exposure duration, the more spikes (and constraints) are generated.

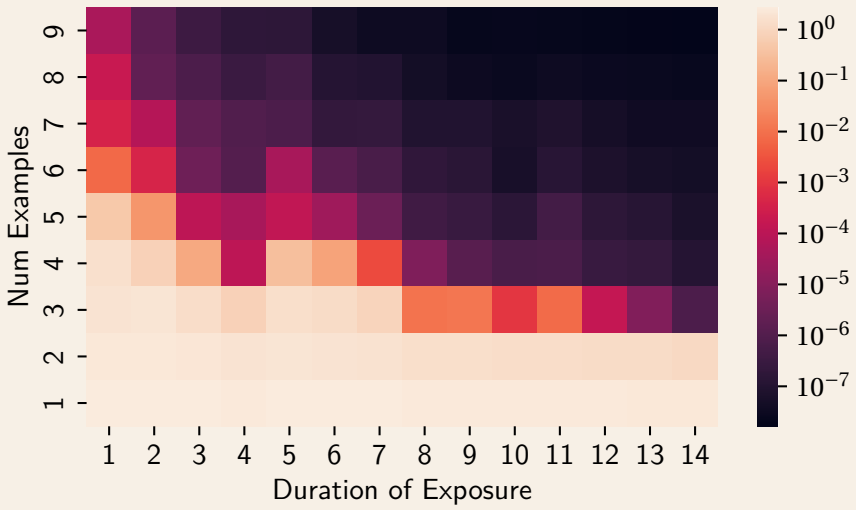



Figure 6.3 – Heatmap of the reconstruction error of the weight matrix $\hat{\mathbf{W}}^{(1)}$, using the $L^2$ norm, as a function of the number of examples and the exposure time used to train the network.

**Learning a single-layer SNN with varying noise levels**

**Example 6.2.**

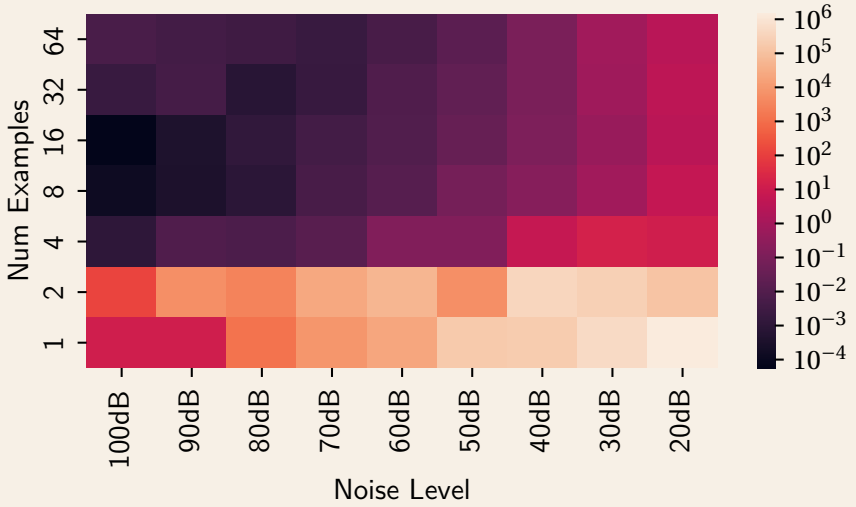



Figure 6.4 – Heatmap of the reconstruction error of the weight matrix $\hat{\mathbf{W}}^{(1)}$, using the $L^2$ norm, as a function of the number of examples and the SNR of the spike times used to learn the network.

In Fig. 6.4, we provide simulations where we introduce uniform noise on the spike

times, varying the signal-to-noise ratio (SNR) of the spike times, and evaluate the reconstruction error on the weights after learning. Notice how increasing the number of examples can partly compensate for higher noise levels.

## 6.4 Learning a Two-Layer SNN

### 6.4.1 Input Processing through a Two-Layer SNN

We have seen that we can train a single-layer SNN in a one-shot inversion of linear equations (6.1), and now consider the case where we have two layers of neurons or TEMs as depicted by Fig. 6.1.

For a certain input $\mathbf{x}^{(0)}(t)$ to the network, this input is passed through a weight matrix $\mathbf{W}^{(1)}$, through a layer of nonlinearities $\left\{\mathrm{TEM}_i^{(1)}\right\}_{i=1}^{n^{(1)}}$, each of which outputs a stream of spikes (i.e. Dirac deltas) at times $\{\tau_{i,\ell}^{(1)}\}_\ell$, which are passed through a second weight matrix $\mathbf{W}^{(2)}$, through a filter (which can be likened to synaptic filters in real neurons), and then through the second layer of nonlinearities $\left\{\mathrm{TEM}_i^{(2)}\right\}_{i=1}^{n^{(2)}}$, each of which outputs a stream of spikes at times $\{\tau_{i,\ell}^{(2)}\}_\ell$.

As before, we assume that we would like to train the network, i.e. find the weight matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$. To do so, we have a set of examples, each of which is composed of an input $\mathbf{x}^{(0)}(t)$—a collection of $n^{(0)}$ signals $x_i^{(0)}(t)$—and of the corresponding target output spike streams $\{\tau_{i,\ell}^{(2)}\}_\ell$ for each of the $n^{(2)}$ output neurons $\{\mathrm{TEM}_i^{(2)}\}_i$. We further assume that the filters that are used between the first and second layer of TEMs are known and satisfy the alias cancellation property as defined in Section 2.4.

Following a similar approach to the one in Section 6.3, we use the examples to constrain the weight matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$. Of course, in the two-layer case, this does not result in *linear* constraints on $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$. Therefore, we propose an approach which performs a layer by layer recovery of the weight matrices, starting with $\mathbf{W}^{(2)}$.

### 6.4.2 Recovery of Weights Leveraging the All-or-None and Asynchronous Properties of Spikes

Learning both weight matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ would be straightforward if the output of the hidden layer $\{\mathrm{TEM}_i^{(1)}\}_i$ were known. As this is not the case, we look to recover it. The trick we use lies in the all-or-none nature of the spikes output by the $\{\mathrm{TEM}_i^{(1)}\}_i$. These outputs are streams of Diracs, as described in (2.20), but with $c_k = 1$, given that a TEM encodes its input using spike times only.

Therefore, the inputs to the second layer $\{\mathrm{TEM}_i^{(2)}\}_i$ will also be streams of Diracs, the *times* of which are dictated by the outputs of $\{\mathrm{TEM}_i^{(1)}\}_i$ and the *weights* of which are dictated by $\mathbf{W}^{(2)}$.

**In depth: All-or-none property of spikes allows weight recovery**

The figure below examines the all-or-none property more closely and shows how the weighted inputs to TEMs can be deconstructed into a sum of many inputs with unit-amplitude Diracs, multiplied by a weight matrix with recoverable weights.
As such, the times of the original signals can be recovered by finding the times of the Diracs in the mixed signals and assigning the Diracs by clustering them according to weight. Then, the weights will simply be the weights of the Diracs observed in the mixed signals, given that the Diracs in the original signals always have amplitude one—as the "all-or-none" property dictates.

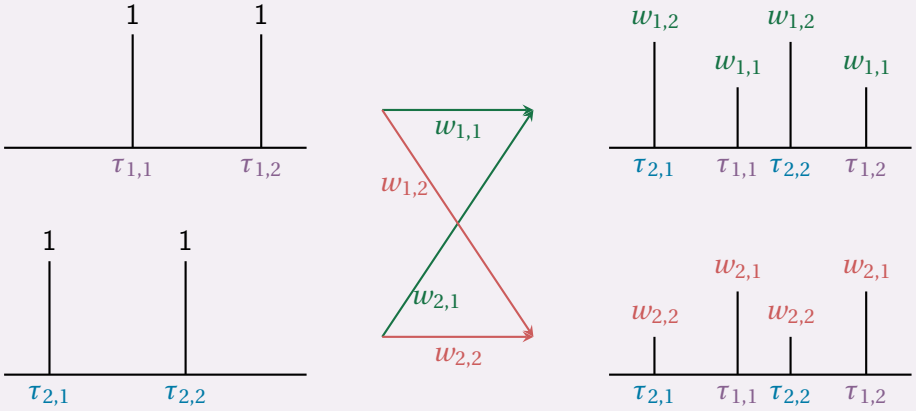



Figure 6.5 – Two streams of unit-amplitude and asynchronous Diracs can be mixed and the different Diracs and weights remain resolvable.

Consequently, if the $\{\mathrm{TEM}_i^{(2)}\}_i$ spikes often enough (and we will clarify this shortly), the Fourier series coefficients of the inputs to the $\{\mathrm{TEM}_i^{(2)}\}_i$ can be recovered, and a joint annihilating filter can be found for all of these inputs. It is then easy to recover the locations of the Diracs, then assign them to different TEMs of $\{\mathrm{TEM}_i^{(1)}\}_i$ while recovering $\mathbf{W}^{(2)}$ (up to a permutation) simply by using a k-means algorithm. Note that this is *only possible* because of the all-or-none and asynchronous nature of the spikes. This is not possible with classical nonlinearities that are graded and simultaneous. Once the outputs of $\{\mathrm{TEM}_i^{(1)}\}_i$ are known, it is easy to recover the weights $\mathbf{W}^{(1)}$ as we did in Section 6.3.

Let us now specify how many spikes at the output of $\{\mathrm{TEM}_i^{(2)}\}_i$ are needed in order for the output of $\{\mathrm{TEM}_i^{(1)}\}_i$ and for $\mathbf{W}^{(2)}$ to be recoverable. Following the result in (Kamath et al., 2021) and reformulated in 2.4.2, we need every TEM at the output to generate $(2L+2)n^{(1)}$ spikes if each of the TEMs in the hidden layer fires $L$ spikes. We will discuss the implications

of this in the conclusion.

### 6.4.3 Simulations

To provide a proof of concept, we design simulations where we again work with a Simulator SNN and a Learned SNN. The Simulator SNN has its weight matrices $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ drawn uniformly at random and is used to obtain the spike stream outputs for the inputs of (in this case) two examples. Then, the Learned SNN "learns" the appropriate weight matrices $\hat{\mathbf{W}}^{(1)}$ and $\hat{\mathbf{W}}^{(2)}$ to reproduce the examples using our described approach.

**Learning a two-layer SNN with varying exposure duration**

**Example 6.3.**

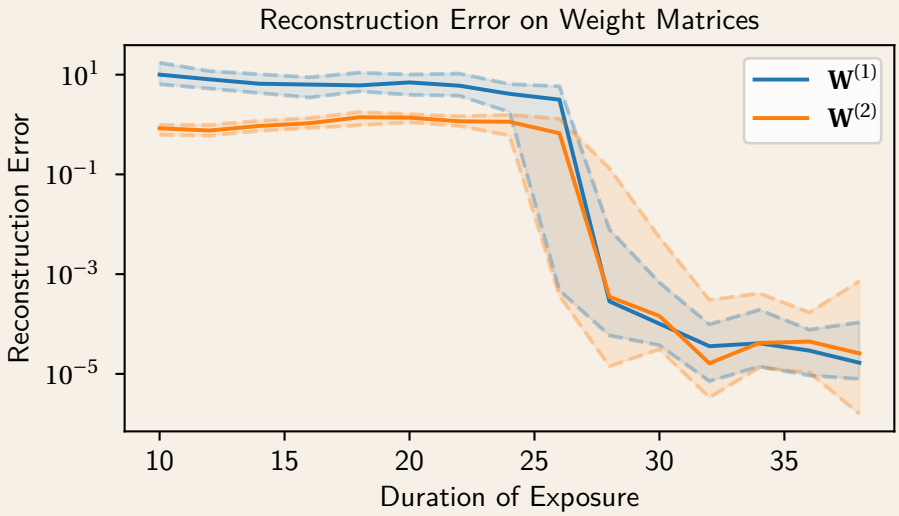



Figure 6.6 – Reconstruction Error, using the $L^2$ norm, of $\mathbf{W}^{(1)}$ and $\mathbf{W}^{(2)}$ of a two-layer SNN, as the duration of exposure increases. We plot the median and the first and second quartiles of the error for 50 randomly generated networks, each trained with two examples.

Both Simulator and Learned SNNs are assumed to have $n^{(0)} = n^{(1)} = 2$ input and hidden nodes and $n^{(2)} = 4$ output nodes. We show the reconstruction error of the weight matrices as the duration of exposure increases. As before, the longer the duration of the exposure, the more spikes the SNN can generate at the output.

## 6.5 Conclusion

We have developed a method that learns the weights of SNNs in a layer-by-layer approach, leveraging the all-or-none and asynchronous nature of the spikes. In fact, these properties of spikes allow the recovery of weight matrices up to permutations, whereas a similar approach applied to ANNs, for example, would never allow recovery beyond Hermitian transformations, making credit assignment a difficult problem. Here, it is easy to assign contributions of nodes to the desired output, allowing the weights to be learned in a layer-by-layer fashion

It is clear, however, that questions remain to be solved for this approach to be actionable in reality. These questions motivate different research directions, each of which is worthy of attention for its own sake. First, our approach currently only deals with the case where the desired task can be perfectly learned by the given network, in the noiseless case. Second, we currently assume that we know what output spike stream to associate with input examples, which is not applicable in every situation. Such a setup can be conceived when training denoising autoencoders, for example, but it should be clarified how one can translate generic loss functions to corresponding spike streams that minimize them.

Moreover, while our approach can technically be extended to more layers, we currently require that the number of spikes at the output of any layer scale as the *total* number of spikes in the previous layer, *multiplied* by the number of nodes in the given layer. As a result, the number of spikes required scales badly with the number of layers. We hope that this requirement can be optimized by leveraging the fact that the input to all neurons within a layer have spikes at the same time (even if these spikes have different weights).

Finally, note that our results assume that the output spike stream of $\{\mathrm{TEM}_i^{(1)}\}_i$ is periodic, which we can enforce in our simulations but is, in reality, more difficult to ensure.

Nonetheless, the results in this chapter provide a stepping stone to train SNNs in a layer-by-layer way, with no concern about the difficulty of credit assignment or about the cost of backpropagation.

# Conclusion and Future Work

To conclude this part of the thesis, we would like to emphasize, once more, the importance of bringing spikes out of their niches.

Spikes provide a power-efficient way to encode information and can provide better sample efficiency in comparison to clocked and synchronous sampling schemes. This is due to the *timing*-based nature of the output, the *asynchrony* of spikes within a device's output stream and therefore across devices, and the *all-or-none* property of spikes.

Many avenues still exist for the exploration of spike-based sampling and processing; we here highlight a few of them that would constitute a natural continuation of the work in this thesis.

### Further theoretical results on mixed time encoding

It would prove useful to better understand when one can recover a low rank input with an unknown factorization from its multi-channel time encoding, and how to perform the recovery. This would not only complete our analysis of the mixed time encoding scenario we presented in Chapter 4, but would also allow lower sample-complexity approaches to recording video with event-based sensors, much as the field of compressed sensing has helped bolster the field of computational photography.

Moreover, it would help improve the sample-complexity needed to train spiking neural networks. In fact, results in low rank matrix factorization of time encoded signals would help solve the joint annihilating filter problem with fewer samples, thus reducing the number of spikes needed to recover every layer of the network.

### Further development of video time encoding to increase sample efficiency

In Chapter 5, we studied how to time encode and recover video which can be represented as a periodic bandlimited signal. As a means to improve the sample efficiency of the system, we propose two extensions.

On one hand, we suggest to better understand time encoding of video with *different* models: it is clear that video can be much sparser than a 3D periodic bandlimited signal, thus requiring a lower spiking rate to ensure recovery.

On the other hand, we suggest to design approaches to adapt the spiking rate of video time encoding *in real time*, in order to capture the varying complexity of the scene. Event-based vision already provides a form of adaptation, as events are triggered by pre-defined criteria, but one can further refine this adaptation by controlling the parameters of the pixels, such as the biases and firing thresholds, as the need arises.

### More accessible implementations of reconstruction and learning algorithms

Algorithms that perform recovery or learning from time encoding are, still, merely sufficient to provide proofs of concept of input recovery. In other words, they are successful, but they are slow and often assume offline data processing. The target is clear: faster online algorithms that can be implemented with low power. For the last action point, it is necessary to also develop hardware that implements these algorithms and that is well suited for the task. Clearly, our classical clocked computers were not designed to process event-based data.

# Timing from Information Part II

# 7 Introduction and Background

## 7.1 Introduction

We once again wander off the path of classical amplitude sampling, where one records the amplitude of a signal at uniformly spaced times, to, this time, show how more tailored approaches can lead to better sampling efficiency.

To start with an example, imagine a friend of yours is going through a rough patch; it is not too serious but you would like to monitor their state over a few weeks. However, this friend's mood fluctuates during the day (e.g. due to hunger or sleepiness) and they conveniently happen to only give (somewhat random) indications about their mood with "good" or "bad" labels. You care about this friend, but you are quite busy so your time is limited. Thus, you would like to maximize the information you get about this friend's state while spending the least amount of time possible checking on them.

This part of the thesis studies how to best sample such a friend's state, except that the friend is not a friend, but a stochastic process with more abstract but (luckily) well defined dynamics, a process that occurs in the field of nanoscale magnetic sensing. The overarching goal is to obtain a good estimate about parameters of the process that are apriori unknown. Furthermore, it is useful to achieve good estimates in the least amount of time possible, so we investigate the problem of choosing sampling times such that they maximize information gain about a parameter, while minimizing total measurement time.

In this context, the title of this thesis—Timing is everything—focuses on the necessity to perform this sampling at the right point in time, to ensure more efficient sampling processes.

### 7.1.1 Nanoscale Magnetic Sensing

We are interested in characterizing a process that occurs in the field of nanoscale Nuclear Magnetic Resonance, through efficient sampling.

**In depth: NMR spectroscopy**

Nuclear Magnetic Resonance (NMR) spectroscopy allows the identification and description of molecules, by searching for magnetic fields around atomic nuclei. It uses a strong magnetic field to align nuclei, then perturbs this strong magnetic field using weak oscillations of different frequencies. The strength of the response to these oscillations varies and can be recorded to form a spectrogram. Frequencies at which these responses are strong are called resonant frequencies and help characterize the sampled molecule.

NMR spectroscopy is the go-to approach for identifying organic compounds. However, it has limited spatial resolution and cannot detect weak magnetic fields, causing single spins to only be detected under extreme conditions of temperature and vaccuum (Rugar et al., 2004).

Nanoscale nuclear magnetic resonance (nNMR) is a new approach that arose in the last fifteen years, and that can measure fewer spins with higher spatial resolution, allowing for higher precision in molecular structure recovery while operating at room temperature (Maze et al., 2008; Balasubramanian et al., 2008).

**In depth: Nanoscale NMR**

Nanoscale Nuclear Magnetic Resonance (nNMR) uses a strong magnetic field to align nuclei of interest, as in NMR spectroscopy, but the rest of the process is different. Nanoscale NMR uses nitrogen-vacancy (NV) centers in diamonds: these are point defects with one unpaired electron which can be coupled to a nucleus of interest. The electron can be excited using a laser, thus producing a fluorescent response which can be read out. This fluorescent response yields information about the electron's own state but also about the state of the spin(s) to which it is coupled.

In the case of nNMR, one obtains samples from a stochastic process which has well-defined dynamics, although parameters of interest, mainly the "resonant" frequencies, are unknown. The samples occur in the form of Bernoulli trials, as probing the electron state gives one out of two results with varying probabability. Thus, in nNMR, one can no longer recover the frequencies of interest by looking for a resonant response but requires a more statistical approach to perform the recovery.

Such a new acquisition technique requires new sampling strategies. With NMR, the technology

imposed a Fourier-spectrum approach to signal and structure estimation. In the nNMR case, one no longer obtains spectra, but rather the result of Bernoulli trials as the fluorescence of the diamond NV center is recorded. As these Bernoulli trials have expectations that vary with time, an adaptive approach is better suited to sample the system at the correct times.

### 7.1.2 Our Contribution

We begin this part of the thesis by covering essential topics in statistical inference and signal processing in Section 7.2. For the remainder of this thesis, the language used will keep a signal processing flavor, abstracting away the physical phenomenon at hand.

#### Chapter 8: Single Frequency Recovery

We assume that we are given a random process, with a sample $X(t)$ at time $t$ following a Bernoulli trial with time-varying mean. This mean varies as an amplitude-shifted and time-decaying cosine with an unknown frequency of interest.

We formulate the problem of the recovery of this frequency from samples, and explain how to go about this recovery assuming arbitrary sampling times. We then present the task at hand—maximizing recovery accuracy, while minimizing measurement time. We quantify the Fisher information obtained from sampling at any time and develop an adaptive sampling approach that maximizes the gained Fisher information while minimizing the total measurement time. Finally, we compare our approach to other non-adaptive approaches in simulations.

#### Chapter 9: Multi Frequency Recovery

As before, we are given a random process, with each sample following a Bernoulli trial with time-varying mean. This time, the mean varies as an amplitude-shifted and time-decaying *product* of multiple cosines of different frequencies, where we are interested in recovering these apriori unknown frequencies.

We define the Fisher information about the unknowns in different forms and consider the case of recovering the frequencies of two cosines which multiply, resulting in a slow and fast oscillations that manifest in the process. We define an estimate of *added* information gained from sampling at any time $t$, and develop a tailored adaptive approach to choosing sample times. Finally we again compare the performance of our adaptive sampling scheme to that of a uniform and an exponential scheme.

## 7.2 Background

### 7.2.1 Statistical View on Parameter Information from Samples

Let's start by understanding how to quantify the information one obtains about an unknown parameter, by sampling a random variable or process.

First we denote $\mathbb{P}[\mathscr{A}]$ to be the probability that an event $\mathscr{A}$ occurs. Now let $X$ be a discrete random variable, for which we define the probability mass function.

**Probability mass function**

**Definition 7.1.**
The *Probability Mass Function* (PMF) $p(x)$ of a discrete random variable $X$ specifies the probability that $X$ take the value $x$:

$$p(x) := \mathbb{P}[X = x]. \tag{7.1}$$

Further assume that $X$ has a probability mass function $p(x;\theta)$ which is parametrized by $\theta$, an unknown parameter which we would like to estimate. We only observe $X$ and our estimate of $\theta$ depends on our observation of $X$. The Fisher information $\mathscr{I}_\theta$ with respect to the parameter $\theta$ is then defined as the variance of the partial derivative of the natural logarithm of the PMF $p(x;\theta)$, with respect to the parameter $\theta$, as detailed below.

**Fisher information**

**Definition 7.2.** The *Fisher information* that a random variable $X$ holds about an unknown scalar parameter $\theta$ is defined to be the variance of the score of the PMF $p(x;\theta)$:

$$\mathscr{I}_\theta = \mathbb{E}\left[\left(\frac{\partial}{\partial\theta}\log p(x;\theta)\right)^2 \middle| \theta\right]. \tag{7.2}$$

If $\log p(x;\theta)$ is twice differentiable, the Fisher information can also be written

$$\mathscr{I}_\theta = -\mathbb{E}\left[\frac{\partial^2}{\partial\theta^2}\log p(x;\theta) \middle| \theta\right]. \tag{7.3}$$

The above definition of the Fisher information applies when one wishes to estimate a scalar parameter $\theta$. In the case where one wishes to estimate a vector of $N$ parameters $\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_N]^T$, the Fisher information is an $N \times N$ matrix with the entry in the $i^{\text{th}}$ row

and $j^{\text{th}}$ column defined as:

$$[\mathscr{I}_{\boldsymbol{\theta}}]_{i,j} := \mathbb{E}\left[\left(\frac{\partial}{\partial\theta_i}\log p(x;\theta)\right)\left(\frac{\partial}{\partial\theta_j}\log p(x;\theta)\right)\middle|\,\theta\right], \tag{7.4}$$

or, if $p(X;\theta)$ is twice differentiable,

$$[\mathscr{I}_{\boldsymbol{\theta}}]_{i,j} = -\mathbb{E}\left[\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log p(x;\boldsymbol{\theta})\middle|\,\boldsymbol{\theta}\right]. \tag{7.5}$$

Further note that the Fisher information obtained from a collection of samples is simply the sum of the Fisher information obtained from each sample.

The Fisher information gives an idea about the *maximal* amount of information one can gain about a parameter. Its inverse provides a lower bound on the variance of any unbiased estimate one makes of this parameter.

**Unbiased estimator**

**Definition 7.3.** An unbiased estimator $\hat{\theta}$ of a parameter $\theta$ is an estimator which has its expectation equal to the true, unknown parameter $\theta$:

$$\mathbb{E}\left[\hat{\theta}\right] = \theta. \tag{7.6}$$

In cruder, less precise english than the original result, we can now present the Cramér-Rao bound.

**The Cramér-Rao bound (Cramér, 2016; Ibragimov and Has' Minskii, 2013)**

**Theorem 7.1.** Let $X$ be a random variable with a probability mass function $p(x;\theta)$ which is parametrized by $\theta$, an unknown parameter. Any *unbiased* estimate $\hat{\theta}$ of $\theta$ based on an observation of $X$ will have a variance which is lower bounded by the inverse of the Fisher information:

$$\mathrm{Var}(\hat{\theta}) \geq \mathscr{I}_{\theta}^{-1}. \tag{7.7}$$

The result also generalizes to PMFs with parameter vectors $\boldsymbol{\theta} = [\theta_1, \theta_2, \cdots, \theta_N]^T$, in which case

$$\mathrm{Cov}_{\boldsymbol{\theta}}(\hat{\boldsymbol{\theta}}) \geq \mathscr{I}_{\boldsymbol{\theta}}^{-1}. \tag{7.8}$$

If an unbiased estimator meets this lower bound with equality, the estimator is called the minimum-variance unbiased estimator (MVUE). Such an MVUE does not always exist, but it is well defined for certain distributions. For example, when searching for the mean of a

Gaussian distribution, the MVUE is simply the average of the sample values.

### 7.2.2 Signal-Processing View on Parameter Uniqueness from Samples

We further require, for this part of the thesis, some basics on sampling a time-varying signal. Below, we present the sampling theorem, a canonical result in the field of signal processing.

**Sampling theorem (Shannon, 1949)**

**Theorem 7.2.** If a function $y(t)$ is bandlimited such that its Fourier transform $Y(f)$ is zero for frequencies $|f| > B\,Hz$, this function is completely determined by its samples if these samples are spaced $1/(2B)$ seconds apart.

#### Recovering the frequency of a single cosine

In the next sections, we will more specifically need to understand how to sample and recover signals that relate to the following signal $y(t)$

$$y(t) = \cos(2\pi f t), \tag{7.9}$$

where the oscillation frequency $f \in [A, B)$ is unknown and where we assume that we can take noiseless samples.

Such a signal can actually be recovered from a single (noiseless) sample, provided the timing of the sample occurs at an appropriate time.

**Nyquist sampling time**

**Definition 7.4.** Assume we wish to recover a frequency $f \in [A, B)$ from a signal defined as in (7.9). Then we call $T_{\text{Nyquist}} = 1/(2(B-A))$ the *Nyquist sampling time*.

**Frequency recovery from one sample**

**Lemma 7.1.** If a signal of the form presented in (7.9) has an unknown frequency lying in $[A, B)$, and one noiseless sample is taken at a time $t < T_{\text{Nyquist}} = 1/(2(B-A))$, the frequency of the cosine can be recovered perfectly. If the same cosine is sampled once at a time $t > 1/(2(B-A))$, the frequency of the cosine cannot be recovered uniquely.

*Proof.* Assume a signal $y(t) = \cos(2\pi f t)$ is sampled at time $t_0$ has value $c_0$. The frequency

$f$ is know to lie in $[A, B)$ and satisfies the following:

$$\begin{aligned} \cos(2\pi f t_0) &= c_0 \\ f &= \frac{1}{2\pi t_0}\left(\pm \arccos(c_0) + 2k\pi\right) \\ f &= \frac{\pm \arccos(c_0)}{2\pi t_0} + \frac{k}{t_0}. \end{aligned} \tag{7.10}$$

If the sample time is assumed to be smaller than the Nyquist sampling time $T_{\text{Nyquist}} = 1/(2(B-A))$, the different solutions for $f$ will $k/t_0 \geq 2(B-A)$ apart. Given that $f \in [A, B)$, there is only one solution for $f$ which satisfies the constraints. Otherwise, if the sample time $t_0$ is larger than $T_{\text{Nyquist}}$, there can be at least two solutions for $f$. In the case where $t_0 \leq 1/(B-A)$, these solutions are $\hat{f} = \arccos(c_0)/(2\pi t_0) + k/t_0$ and $\hat{f} = -\arccos(c_0)/(2\pi t_0) + (k+1)/t_0$ for $k = \lceil A t_0 - \arccos(c_0)/(2\pi) \rceil$. Note that these solutions are sometimes equal if $\arccos(c_0)$ is an integer multiple of $\pi$. In cases where $t_0 > 1/(B-A)$, multiple solutions arise because $k$ can take different values. □

We provide two examples to illustrate the above results.

**Sub-Nyquist sampling time**

**Example 7.1.**

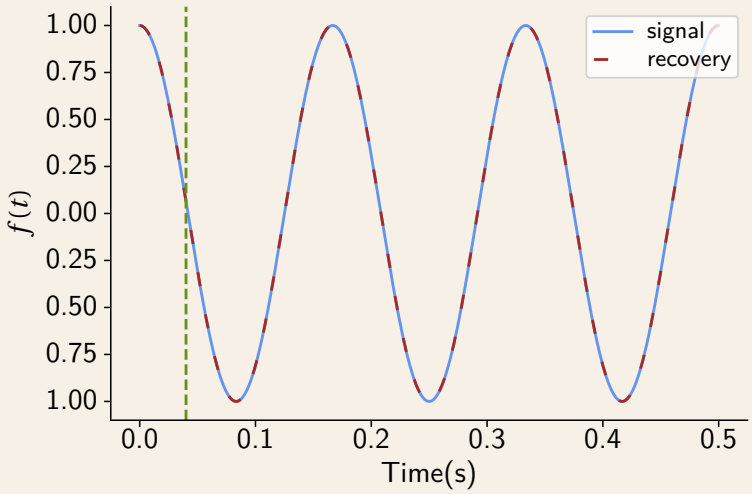



Figure 7.1 – The original cosine is plotted in blue and is assumed to have a frequency $\in [0\text{Hz}, 10\text{Hz})$, i.e. the Nyquist spacing is $T_{\text{Nyquist}} = 0.05$. It is sampled at $t_0 = 0.04$, indicated by the dashed green vertical line. There is only one possible cosine frequency which fits this measurement and we show the corresponding signal in dashed red.

We consider the case of a cosine with unknown frequency and which is sampled once, at a time which is *smaller* than the Nyquist rate. In this case, the cosine frequency can be recovered perfectly.

**Above-Nyquist sampling time**

**Example 7.2.** We consider the case of a cosine with unknown frequency $f \in [A\text{Hz}, B\text{Hz})$ and which is sampled once, at a time which is *larger* than the Nyquist rate. In this scenario, the cosine frequency cannot be uniquely recovered.

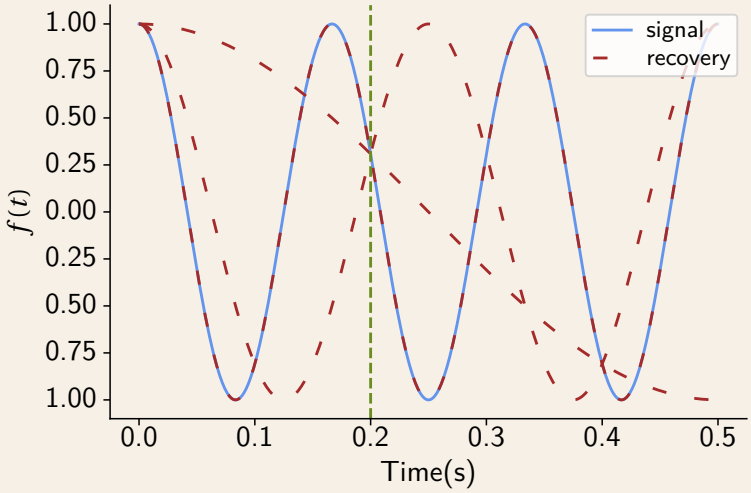



Figure 7.2 – The original cosine is plotted in blue and is assumed to have a frequency $\in [0\text{Hz}, 10\text{Hz})$, i.e. $T_{\text{Nyquist}} = 0.05$. It is sampled at $t_0 = 0.2 > T_{\text{Nyquist}}$, indicated by the dashed green vertical line. There are three possible cosine frequencies which fit this measurement and we show the corresponding signals in dashed red.

### 7.2.3 Sampling Schemes

Finally, we also define two sampling schemes: The uniform sampling scheme and the exponential sampling scheme.

### Uniform sampling scheme

**Definition 7.5.** A *uniform sampling scheme* with *sampling step* $\Delta_t$ samples a signal $y(t)$ once at each time $t_n$ which are uniformly spaced from one another:

$$t_n = n\Delta_t, \tag{7.11}$$

where $n$ indexes the measurements.

The uniform sampling scheme is a very classical and well-understood approach to sampling. The advantage of using it include the simplicity of the scheme as well the rich theoretical result which guarantee reconstruction from sampling under certain conditions (such as the sampling theorem in Theorem 7.2).

### Exponential sampling scheme

**Definition 7.6.** An *exponential sampling scheme* with parameters $\tau_0$, $K$, $M_K$ and $F$ samples a signal $y(t)$ at times $t_k$ such that:

$$t_k = \tau_0 2^k, \tag{7.12}$$

and where $y(t)$ is sampled $M(K,k)$ times at each $t_k$, where

$$M(K,k) = M_K + F(K-k). \tag{7.13}$$

The exponential sampling scheme has many tunable parameters, but allows one to reach samples at higher times more quickly, allowing for increased recovery precision in some scenarios. For example, in the case of a signal $y(t) = at$ with an unknown $a$ and with additive noise $e(t) \sim \mathcal{N}(0, \sigma^2)$ of variance that is independent of the sampling time, sampling at a later point in time allows for a more precise recovery of $a$.

# 8 Nanoscale Magnetic Sensing: Single Frequency Estimation

## 8.1 Introduction

Consider the case where the spin of an electron in a Nitrogen Vacancy center in a diamond is coupled to a single nuclear spin, leading to the Bernoulli distribution of the electron's fluorescene to be dictated by a single unknown variable: the frequency behind its oscillation. We would like to be able to recover this variable as accurately as possible, while requiring the least amount of sampling time possible.

Standard approaches to recovering such a frequency include simple sampling schemes such as the well-understood uniform sampling scheme or the exponential sampling scheme which allows for better frequency resolution (Waldherr et al., 2012).

However, as we will see, the problem is well defined, placing the development of a more tailored approach within reach. The process we deal with showcases oscillations and an eventual decay, which helps clearly define regions where samples are more "informative".

In this chapter, we start with a definition of the sampled process. Then, we expose the goals and constraints we deal with, and clarify how we perform the frequency recovery given samples at arbitrary times. We then present our adaptive approach to sampling which is based on the Fisher information which, in this case, varies over time. We conclude the chapter with simulations to compare the performance of our adaptive sampling scheme to that of uniform and exponential sampling.

## 8.2 Problem Formulation

### 8.2.1 Sampled Process

We assume we can sample a single spin process $X(t)$, as defined below.

**Single spin process**

**Definition 8.1.** A random process $X(t)$ is a *single spin process* if it can take values 0 or 1 and which has a time-varying probability mass function (PMF) $p_{X(t)}(x)$, and where we denote $p(t) = p_{X(t)}(1)$ with $p(t)$ taking the following form:

$$p(t) = \mathbb{P}[X(t) = 1] = \frac{1}{2}\left(1 + \cos(2\pi f t)\exp\left(-\frac{t}{T_2}\right)\right), \tag{8.1}$$

where we call $f$ the oscillation frequency and $T_2$ the evolution time.

Note that the probability of taking value 1 is a continuous function of time, but the random variable $X(t)$ remains a *discrete* random variable although it can be *sampled* continuously in time.

**Time-varying Bernoulli probability**

**Example 8.1.**

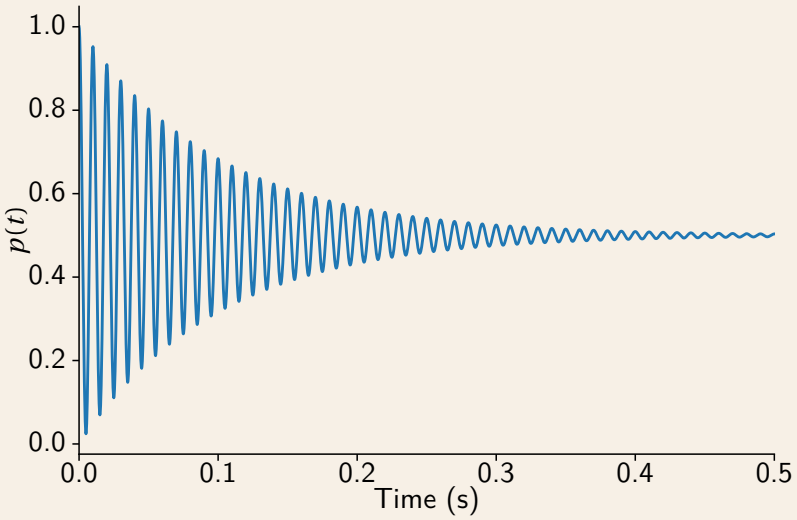



Figure 8.1 – The probability of a process $X(t)$ taking value 1 at time $t$, as described in (8.1). The signal has an oscillation frequency of $f = 100\text{Hz}$ and a time constant of $T_2 = 0.1\text{s}$.

Given that the process which we measure takes value 1 with time-varying probability, all samples are not created equal. For example, if a sample is taken at a time $t_0$ when $p(t_0) = 0.5$, the variance of $X(t_0)$ takes its maximal possible value, at 0.25. On the other hand, if a sample is taken at time $t$ when $p(t) = 0.9$, the variance is lower, and is equal 0.09.

Intuitively, and after observing the graph in Fig. 8.1, one would ideally like to sample $X(t)$ at the peaks or troughs of the oscillations, a bit later in time to gain more precision (the sample at time $t=0$ being deterministically equal to 1), but not so late that the effect of the decay kicks in. We would like to formally characterize this ideal sampling time.

### 8.2.2 Goal and Constraints

Let us formalize the task we want to achieve by writing it as an optimization problem.

For the rest of this chapter, we are interested in recovering $f$ from samples of $X(t)$ and wish to determine the sample times that allow the "best" recovery of $f$. More specifically, we denote $t_n$ the times at which $X(t)$ is sampled, and $x_n$ the outcomes of the sampling, where each $x_n$ takes as value either 0 or 1. Moreover, we assume that, to take any such sample of $X(t_n)$ the process needs to restart from time $t=0$ because of the physical properties of spins which, once projected onto a state, can no longer evolve afterwards.

Our goal is to minimize the mean-squared error of our estimate of $f$ while observing a limited total sampling time. More specifically, we would like to choose the $t_n$'s such that

1. the expected mean-squared error $\mathbb{E}\left[\left(\hat{f}-f\right)^2\right]$ is minimized, and
2. the overall sampling time $\sum_n t_n$ is limited.

We further assume that the frequency of interest can be written $f = f_{\text{carrier}} - \delta_f$ where $f_{\text{carrier}}$ is known and is in the KHz-MHz range and $\delta_f$ is unknown but assumed to be uniformly distributed over $(0\text{Hz}, 10\text{Hz}]$. Moreover, the evolution time is known and is in the range of tens of milliseconds to seconds.

## 8.3 Frequency Recovery from Samples

First let us specify how to estimate the parameter $f$ of such a process $X(t)$, when $X(t)$ is sampled at arbitrary times $t_n$ and yields the measurements $x_n$, for $n=1,\cdots,N$.

We follow a Bayesian approach, and seek to maximize the likelihood (or, equivalently, the log likelihood) of the samples' occurrence given the parameter $f$ which we are searching for. In more mathematical terms, our estimate $\hat{f}$ of $f$ is the solution to

$$\begin{aligned} \hat{f} = &\operatorname*{argmax}_f \log \mathbb{P}\left[X(t_n)=x_n, \forall n=1,\cdots,N; f\right] \\ &\text{subj. to} \quad f_{\text{carrier}} - 10 \le f < f_{\text{carrier}}. \end{aligned} \tag{8.2}$$

In the above, the log likelihood of the samples' occurrence is equal

$$\begin{aligned}\log \mathbb{P}\big[X(t_n) = x_n, \forall n = 1, \cdots, N; f\big] &= \log \prod_{n=1}^{N} \mathbb{P}\big[X(t_n) = x_n; f\big] \\ &= \log \prod_{n=1}^{N} p(t_n; f)^{x_n} \times (1 - p(t_n; f))^{1-x_n} \\ &= \sum_{n=1}^{N} \big(x_n \log p(t_n; f) + (1 - x_n) \log(1 - p(t_n; f))\big) \end{aligned} \tag{8.3}$$

Such an estimate of $f$ is called the maximum likelihood estimate (MLE). The MLE is often an intuitive choice for a solution, and has the benefit of often being the minimum variance unbiased estimator, which meets the Cramér Rao lower bound. For example, this is the case of a random variable $Y$ which takes values 0 or 1 with an unknown mean value: the MLE of the mean value will provide the most accurate (i.e. minimum-variance), unbiased estimate.

## 8.4 Adaptive Sampling Strategy

The recovery approach we exposed above is agnostic to how one chooses the sampling times $t_n$.

Clearly, observing Fig. 8.1, all samples do not yield the same amount of information. Here, we present an adaptive scheme to sample the process at hand in the most "efficient" way, such that the requirements in Section 8.2.2 are satisfied. In more mathematical terms, our goal is to solve for the sample times $t_n$ in the below optimization problem:

$$\begin{aligned} \underset{[t_n]_{n=1}^N}{\operatorname{argmin}} \quad & \mathbb{E}\big(\hat{f}\big([t_n]_{n=1}^N\big) - f\big)^2\big] \\ \text{subj. to} \quad & \sum_{n=1}^{N} t_n \leq T, \end{aligned} \tag{8.4}$$

where $\hat{f}([t_n]_{n=1}^N)$ denotes the maximum likelihood estimate for $f$ given the sample times $t_n$.

### 8.4.1 Fisher information as a measure of sample utility

As it is difficult to derive the variance of the estimator analytically, we solve a simpler problem. Rather than minimizing the variance of the estimator, we aim to minimize the lower bound on the variance, i.e. the Cramér-Rao bound. As mentioned in Theorem 7.1, the Cramér-Rao bound is the inverse of the Fisher information with respect to the unknown $f$. Therefore, rather than minimizing the Cramér-Rao lower bound, we maximize the Fisher information

obtained from the samples and our optimization problem becomes the following:

$$\begin{aligned} &\underset{[t_n]_{n=1}^N}{\operatorname{argmax}}\, \mathcal{I}_f([t_n]_{n=1}^N) \\ &\text{subj. to} \quad \sum_n t_n \leq T. \end{aligned} \tag{8.5}$$

where $\mathcal{I}_f([t_n]_{n=1}^N)$ is the total Fisher information obtained about $f$ from $X(t)$ when sampling at times $t_n$. It is equal to the sum of the Fisher informations obtained from sampling at each time $t_n$:

$$\mathcal{I}_f([t_n]_{n=1}^N) = \sum_{n=1}^N \mathcal{I}_f(t_n), \tag{8.6}$$

where $\mathcal{I}_f(t)$ is the Fisher information gained from sampling at time $t$:

$$\mathcal{I}_f(t) = 4\pi^2 \frac{t^2 \sin^2(2\pi f t)}{\exp(2t/T_2) - \cos^2(2\pi f t)}. \tag{8.7}$$

The expression for the Fisher information may not seem inviting, but it essentially comprises of oscillations with frequencies that depend on $f$, a quadratic increase and an exponential decay factor as illustrated in the example below.

**Time-varying Fisher information**

**Example 8.2.**

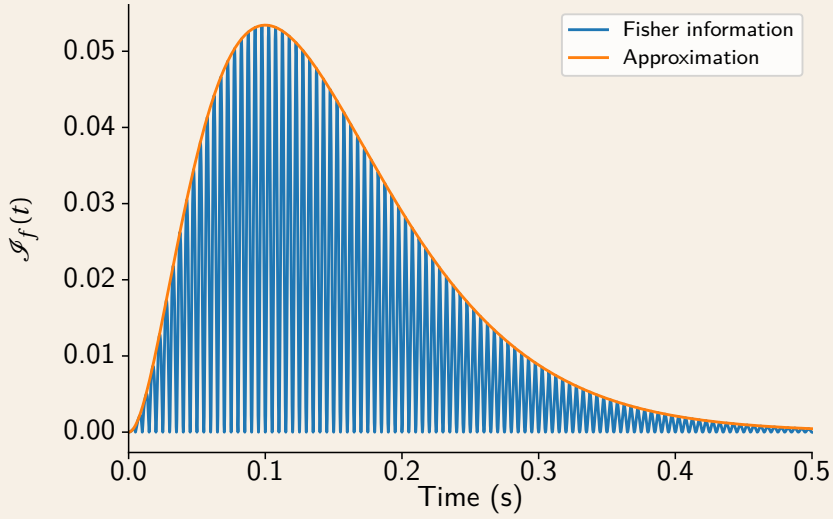



Figure 8.2 – (In blue) The Fisher information of the process in Fig. 8.1. (In orange) The estimated envelope as defined in (8.8). The signal has an oscillation frequency of $f = 100\text{Hz}$ and a time constant of $T_2 = 0.1\text{s}$.

We provide an example of the time-varying Fisher information. We plot, in Fig. 8.2, the Fisher information of a process $X(t)$ with frequency frequency of $f = 100\text{Hz}$ and a time constant of $T_2 = 0.1\text{s}$. Notice how the Fisher information has fine oscillations and a coarser and smoother increase followed by a decrease, caused by the $t^2$ term and the decay with evolution time $T_2$, respectively.

### 8.4.2 Maximizing the Fisher Information

Clearly, the time-varying Fisher information of the process $X(t)$ is not convex with respect to time, as we also see in Fig. 8.2, rendering it difficult to maximize. However, for high enough frequency, this Fisher information highly resembles an oscillation contained in an envelope $g(t)$ of the form:

$$g(t) = t^2 \exp(-2t/T_2). \tag{8.8}$$

To show this, we overlay the above defined envelope with the plot of the time-varying Fisher information in Fig. 8.2.

Thus, as the optimization problem in (8.5) is difficult to solve in closed form, we look for an approximate closed-form solution by applying the following:

1. We replace $\mathscr{I}_f(t)$ by its envelope $g(t)$, and would now like to solve

$$\begin{aligned} \underset{[t_n]_n}{\text{argmax}} \quad & \sum_n t_n^2 \exp(-2t_n/T_2) \\ \text{subj. to} \quad & \sum_n t_n \leq T. \end{aligned} \tag{8.9}$$

2. We can also show by contradiction that the above can be maximized by taking $n_0$ samples at an optimal sample time $t_{\text{opt}}$ for $n_0 = \lfloor T/t_{\text{opt}} \rfloor$ and one sample at $t_{\text{res}} = T - n_0 t_{\text{opt}}$. We can therefore relax the problem to find $t_{\text{opt}}$ such that:

$$\begin{aligned} t_{\text{opt}} &= \underset{t}{\text{argmax}} \quad \frac{g(t)}{t} \\ &= \underset{t}{\text{argmax}} \quad t\exp(-2t/T_2) \end{aligned} \tag{8.10}$$

   The above optimization problem also has the benefit of resembling a measure of Fisher information *per* unit of sample time.

The simplified problem in (8.10) can be solved by setting $t_{\text{opt}} = T_2/2$. Given that this only solves the simplified problem, and not our target in (8.5) where the objective function exhibits fast oscillations, we take a sample as close to $t_{\text{opt}}$ as possible, such that this sample lies at the peak of an oscillation of the Fisher information.

At this point, it has probably become clear to the reader that it is actually impossible to maximize the true Fisher information, as it depends on $f$, a quantity that is apriori unknown, and which we would like to estimate. Therefore, we proceed in an adaptive manner, where we iteratively search for an estimate of $f$, refine our estimate of the Fisher information, choose the next sample time such that the *estimated* Fisher information is maximized, and repeat.

### 8.4.3 Further Considerations about Aliasing

Given that the ideal sampling point we choose is close to $t_{\text{opt}} = T_2/2$, this can become problematic as the evolution time becomes large. In fact, we saw, in Lemma 7.1, that, if a single sample time is chosen such that it is greater than the Nyquist sampling time $T_{\text{Nyquist}}$, which is here equal to 0.05, the frequency of interest cannot be uniquely recovered.

For this reason, whenever the evolution time $T_2$ is large, we also sample closer to $T_{\text{Nyquist}}$ to ensure a better recovery of the frequency, as we explain in more detail in Section 8.5.3.

### 8.4.4 Our Adaptive Sampling Scheme in a Nutshell

We summarize our adaptive sampling algorithm for further clarity.

**Adaptive sampling scheme for single spin processes**

**Definition 8.2.** The *adaptive sampling scheme* is allotted a total permitted measurement time, assumes an initial estimate for $\hat{f}$ and iterates through the following steps.

1. It assumes that a certain fraction of the permitted measurement time can be used to sample the process $X(t)$,
2. it chooses the sample times $t_n$ such that they maximize the gained Fisher information given this fraction of the measurement time, according to the latest estimate of $\hat{f}$,
3. it samples the process and obtains a new estimate $\hat{f}$ from all the samples it has obtained so far, and
4. it repeats these steps until the total permitted measurement time is elapsed.

Note that, if aliasing can occur as described in Section 8.4.3, step 2 is modified such that sample times that are close to the Nyquist sampling time are also chosen.

Notice that the above definition leaves some design choices undetermined. For example, it is unclear how much measurement time should be allotted at each iteration and how many

samples should be taken near the Nyquist sampling time, should that be needed. We address these practical questions in Section 8.5.3.

## 8.5 Simulations

We wish to evaluate the performance of our method. To this end, we consider different values for the carrier frequency $f_{\text{carrier}}$ and evolution time $T_2$ and estimate the performance of our approach assuming different total sampling times.

More practically, we simulate the process with time-varying probability described in (8.1) for carrier frequencies equal to $10^4$Hz, $10^5$Hz and $10^6$Hz and for evolution times $T_2$ equal to 1s, 100ms, and 10ms.

Then, we simulate three sampling approaches on such a process: our adaptive approach, a uniform approach and an exponential approach (all further explained later). The recovery of the frequency once the samples are taken is always performed by iteratively solving for the maximum likelihood for $f$, as described in (8.2). Our code is implemented in C++ and uses the NLOpt to perform iterative optimization (Johnson).

Each sampling approach is tested on 500 processes drawn at random, i.e. the frequency shift $\delta_f$ is drawn uniformly from $[0\text{Hz}, 10\text{Hz}]$. Our plots in Fig. 8.3 present, per sampling approach, the median squared error of the frequency estimate, as well as the first and fourth quartiles, plotted against varying total measurement times, on log-log plots. The simulations implement a sweep over total measurement times. However, as one cannot guarantee that the total measurement times be used precisely, especially in the case of adaptive sampling, we use the median of the obtained measurement time, when generating these plots.

Before we discuss the results, we quickly detail our implementations of the three sampling approaches we use.

### 8.5.1 Uniform Sampling

We adopt a uniform sampling approach as defined in Definition 7.5, with a sampling step of $\Delta_t = 1\mu s$. Every sample time is used only once, and there is no cap on the allowed timing of the samples.

Such an approach allows one to use theoretical results on uniform sampling, permitting, for example, a good understanding of time-frequency resolution tradeoffs.

### 8.5.2 Exponential Sampling

We adopt an exponential sampling approach as defined in Definition 7.6 and originally presented in Waldherr et al. (2012).

In Waldherr et al. (2012), the values of the parameters $M_K$ and $F$ are empirically set to be 36 and 8 respectively. The value of $\tau_0$ is also fixed to $20\mu s$, while $K$ varies within the experiments to evaluate the improvement of the error with increased sampling time.

In our scenario, we fix $\tau_0 = 1\mu s$ to match our highest frequency ($10^6$Hz) and we always choose $K$ such that our last sampling time $t_K$ is as close to the evolution time as possible. That means that, for a fixed evolution time, $K$ is also fixed. We do this, because we notice that the amount of gained information decreases when we surpass the evolution time, as is illustrated in Fig. 8.2.

Then, we find $M_K$ and $F$ by setting a fixed ratio $M_K/F = 4$ between the two and we solve for the appropriate values of $M_K$ and $F$ assuming a maximal sampling time which we would like to use.

### 8.5.3 Adaptive Sampling

We follow the definition for the adaptive sampling scheme presented in Definition 8.2 and provide, here, further specifications about the algorithm's implementation.

#### Sample choice

Given an estimate $\hat{f}$ of the frequency, the next optimal sample time is taken to be close to $T_2/2$ (the solution to (8.10)), and such that the estimated Fisher information is maximized. The Fisher information roughly oscillates with a frequency of $2\hat{f}$ and the optimal time is chosen such that it lies at the latest peak that occurs before $T_2/2$.

If this optimal sample time is lower than the Nyquist sample time $T_{\text{Nyquist}}$, it is repeated as many times as possible, such that the allowed sampling time is not surpassed.

If the optimal sampling time is higher than the Nyquist sample time $T_{\text{Nyquist}}$, the total sampling is divided into sampling at this optimal time and sampling close to $T_{\text{Nyquist}}$. The samples are taken such that the amount of information gathered at the two locations are roughly the same. More precisely, if a sample at the optimal location yields four times as much information as a sample near $T_{\text{Nyquist}}$, then we sample four times as often near $T_{\text{Nyquist}}$, at a peak of the oscillation in the Fisher information.

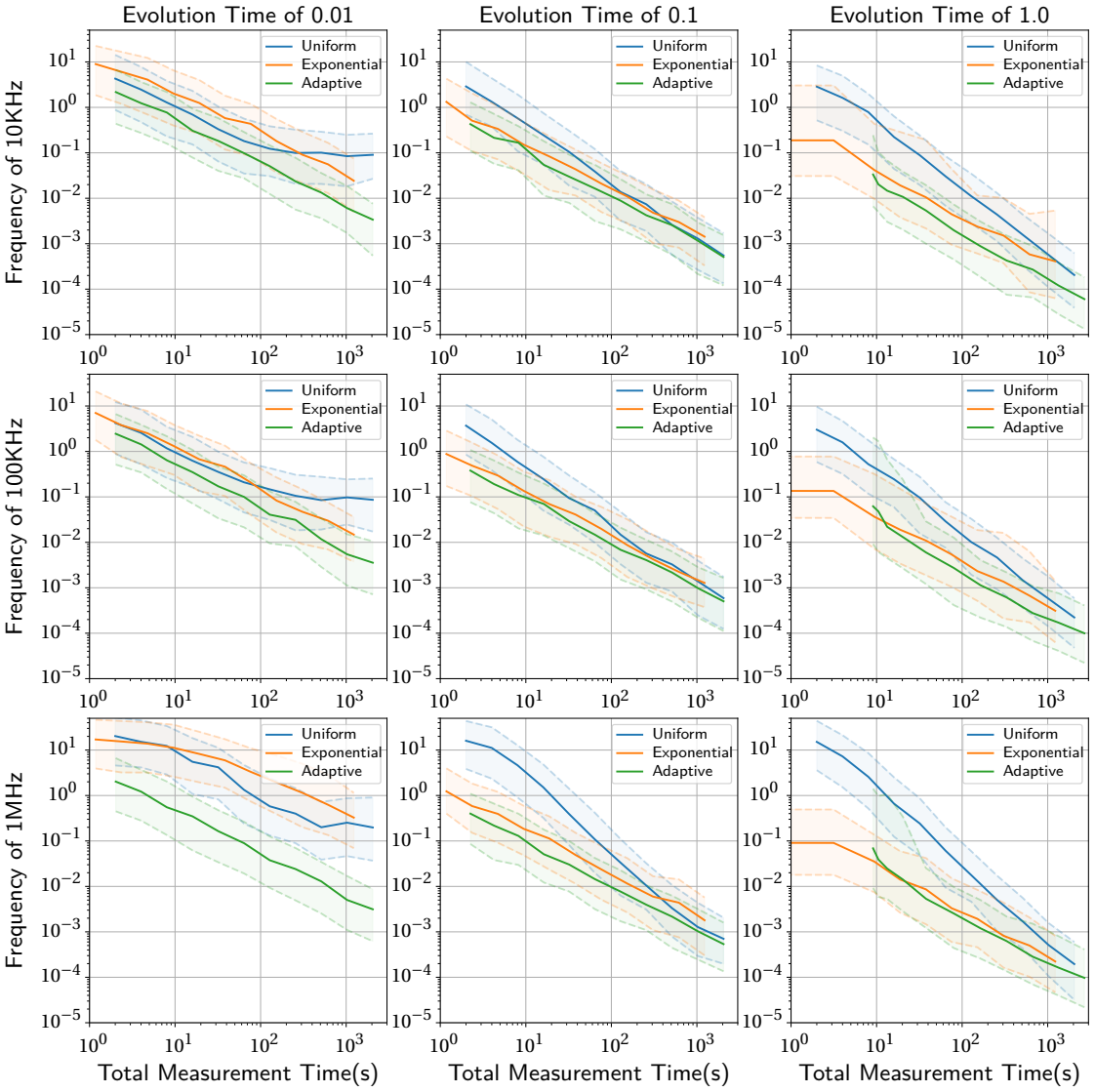



Figure 8.3 – Reconstruction error as a function of total measurement time using the (blue) adaptive sampling scheme, the (green) exponential scheme and the (orange) uniform sampling scheme. We plot the median squared reconstruction error using the solid lines, as well as the first and third quartile using the dashed lines. Different subplots represent results for different pairs of evolution time $T_2$ and carrier frequency $f_{\text{carrier}}$ as specified by the column and row labels, at the top and left-hand side of the plots, respectively.

### Step size

We experimented with schemes that determine the allotted measurement time for each iteration of our adaptive scheme, from the total allowed time, for $L$ iterations. Based on the fact that our estimate $\hat{f}$ becomes closer to the true value with increasing measurement time,

we set the alloted measurement time $T_\ell$ for iteration $\ell$ to be:

$$T_\ell = T_0 \ell^2, \tag{8.11}$$

where $T_0$ is chosen such that the total allowed measurement time is used and $L$ takes as value the minimum of $10$ and the total measurement time divided by twice the evolution time.

### 8.5.4 Results

We provide, in Fig 8.3, the results of the simulations for the three sampling schemes. Notice how, for all approaches, the mean-squared error of the estimate $\hat{f}$ decreases with increased total measurement time, as expected.

Monitoring the green curve across the different plots, i.e. the curve for the adaptive sampling method, we notice that, in some case, it performs better than the other two sampling methods, assuming the same total measurement time. In the other cases, for example in the case of a frequency of $1\mathrm{MHz}$ and an evolution time of $1\mathrm{s}$, the adaptive method is not worse than the uniform and classical methods. However, the adaptive scheme seems more consistent across different operation assumptions than the exponential sampling scheme.

The reader may have further noticed that, when the evolution time is 0.01, the uniform sampling scheme seems to provide decreasing reconstruction error up to a certain measurement time, and this decrease slows down past this point. This occurs because the uniform samples of the process are taken at increasingly large times, eventually surpassing the relatively small evolution time. Hence, this effect does not occur for larger evolution times, where we did not simulate total measurement times that are large enough to exhibit this behavior. It also does not occur for the other two techniques, by design. In the adaptive scheme, we do not sample beyond the evolution time because this would contradict our maximization of Fisher information per unit of time. In the exponential scheme, we explicitly avoid sampling beyond the evolution time as explained in Section 8.5.2.

## 8.6 Conclusion

We have presented the problem of recovering a frequency from a process with time-varying probability which oscillates with this frequency and decays over time. We then presented an adaptive scheme that chooses the best sampling times for this process such that the mean-squared error of the estimate is minimized, while observing a maximal total sampling time. This adaptive scheme is based on maximizing the time-varying Fisher information about the parameter of interest provided by samples at different times and provides, according to our simulations, a more robust approach to efficiently sampling such processes.

The process we considered assumed that a single frequency governs the oscillation in the

time-varying probability. In the next chapter, we will consider the case where we would like to recover multiple frequencies which characterize a process's dynamics.

# 9 NV Sensing: Multi Spin Estimation

## 9.1 Introduction

We consider the case where the electron spin of an NV center in a diamond is coupled to multiple nuclear spins. In this case, the electron's fluorescence follows a Bernoulli distribution which varies over time and is dictated by *multiple* unknown variable: the frequencies of oscillations that multiply. In fact, the mean of the Bernoulli distribution varies as an amplitude-shifted, time-decaying *product* of cosines of different frequencies, leading to varying amounts of information being provided by samples at different times.

Once again, the aim is to recover these frequencies as accurately as possible, while operating under time constraints.

In this chapter, we start with a definition of the sampled multi-spin process, and show how it can be interpreted using either sums or products of cosines. Following this definition, we explain how to recover the frequencies from samples taken at arbitrary times using maximum likelihood estimation. Then, we derive the Fisher information obtained about each of the parameters as a function of time. This allows us to study the case of a two-spin system and devise a sampling approach that takes into account the effect of the frequencies on the Fisher information of the process. Finally, we compare our approach with the more standard approaches of uniform and exponential sampling.

## 9.2 Problem Formulation

In this chapter, we assume that we are dealing with a multi-spin process where a random variable $X(t)$ can be sampled at any time $t$.

**Multi-spin process**

**Definition 9.1.** A random process $X(t)$ is a *multi-spin process* if it can take values 0 or 1 with a time-varying probability mass function (PMF) $p_{X(t)}(x)$. We further denote $p(t) = p_{X(t)}(1)$ where $p(t)$ takes the following form:

$$p(t) = \mathbb{P}[x(t) = 1] = \frac{1}{2^{N_f}}\left(1 + \left(\prod_{\ell=1}^{N_f} \cos(2\pi(f_\ell)t)\right)\exp\left(-\frac{t}{T_2}\right)\right), \tag{9.1}$$

where $N_f$ is the number of oscillation frequencies $f_\ell$ to estimate, and $T_2$ is the evolution time.

Naturally, the product in the above expression can also be expanded, and the result is an amplitude-shifted, time-decaying *sum* of $2^{N_f}$ cosines:

$$p(t) = \frac{1}{2^{N_f}}\left(1 + \left(\frac{1}{2^{N_f}}\sum_{m=1}^{2^{N_f}} \cos\left(2\pi\phi_m t\right)\right)\exp\left(-\frac{t}{T_2}\right)\right) \tag{9.2}$$

$$\tag{9.3}$$

where we let $\boldsymbol{\phi} = \left[\phi_1, \cdots, \phi_{N_\phi}\right]^T$ be the summed frequencies with $N_\phi = 2^{N_f}$ and

$$\phi_m = \sum_{\ell=1}^{N_f} (-1)^{(m \mod 2^\ell)} f_\ell. \tag{9.4}$$

Note that there are symmetries in the frequencies $\phi_m$, leading to a lower effective number of contributing frequencies of $2^{N_f - 1}$.

As in Chapter 8, our task is to recover the frequencies $f_\ell$ or $\phi_m$, while assuming, as before, that the evolution time $T_2$ is in the range of tens of milliseconds to seconds and that each of the frequencies $f_\ell$ is drawn from a uniform distribution over $\left[f_{\text{carrier}} - 10, f_{\text{carrier}}\right)$.

## 9.3 Frequency Recovery from Samples

First, we explain how to recover the frequencies $f_\ell$ or $\phi_m$ from samples at arbitrary times $t_n$. As before, we wish to maximize the likelihood of the samples' occurrence with respect to the parameters $f_\ell$ or $\phi_m$. Thus, we can define $\boldsymbol{f}$ to be the vector of frequencies $f_\ell$ and solve for:

$$\begin{aligned} \hat{\boldsymbol{f}} &= \underset{\boldsymbol{f}}{\operatorname{argmax}} \log \mathbb{P}\left[X(t_n) = x_n, \forall n = 1, \cdots, N; \boldsymbol{f}\right] \\ &\text{subj. to} \quad f_{\text{carrier}} - 10 \le f_\ell < f_{\text{carrier}}, \quad \forall \ell = 1, \cdots, N_f. \end{aligned} \tag{9.5}$$

Alternately, defining $\boldsymbol{\phi}$ to be the vector of frequencies $\phi_m$, we can solve for:

$$\begin{aligned}\hat{\boldsymbol{\phi}} = &\underset{\boldsymbol{\phi}}{\operatorname{argmax}} \log \mathbb{P}\big[X(t_n) = x_n, \forall n = 1, \cdots, N; \boldsymbol{\phi}\big] \\ &\text{subj. to} \quad f_{\text{carrier}} - 10 \leq f_\ell(\phi) < f_{\text{carrier}}, \quad \forall \ell = 1, \cdots, N_f,\end{aligned} \tag{9.6}$$

where $f_\ell(\phi)$ is the recovered frequency $f_\ell$ from the frequencies $\phi_m$, by simple inversion of (9.4).

## 9.4 Fisher Information about Frequencies

We would like to find the optimal time to sample the process, such that the variance of the estimates defined above is minimized while operating under time constraints. Our optimization problem can therefore be written as

$$\begin{aligned}\underset{[t_n]_{n=1}^N}{\operatorname{argmin}} \quad &\sum_{\ell=1}^{N_f} \mathbb{E}\big(\hat{f}_\ell\big([t_n]_{n=1}^N\big) - f_\ell\big)^2] \\ &\text{subj. to} \quad \sum_{n=1}^{N} t_n \leq T,\end{aligned} \tag{9.7}$$

if one would like to focus on the recovery of the frequencies $f_\ell$. Otherwise, one can also write an optimization problem in terms of the frequencies $\phi_m$:

$$\begin{aligned}\underset{[t_n]_{n=1}^N}{\operatorname{argmin}} \quad &\sum_{m=1}^{2^{N_f-1}} \mathbb{E}\big(\hat{\phi}_m\big([t_n]_{n=1}^N\big) - \phi_m\big)^2] \\ &\text{subj. to} \quad \sum_{n=1}^{N} t_n \leq T.\end{aligned} \tag{9.8}$$

Note that we wish to solve for $2^{N_f-1}$ frequencies rather than $2^{N_f}$ frequencies given the symmetries we previously mentioned.

As was the case in Chapter 8, deriving and thus minimzing the variance of our estimator is difficult so we will, for the remainder of the chapter, instead aim to minimize the *lower bound* on the variance, as dictated by the Cramér-Rao bound in Theorem 7.1:

$$\begin{aligned}\underset{[t_n]_{n=1}^N}{\operatorname{argmin}} \quad &\sum_{m=1}^{2^{N_f-1}} \frac{1}{\mathscr{I}_{\phi_m}([t_n]_{n=1}^N)} \\ &\text{subj. to} \quad \sum_{n=1}^{N} t_n \leq T.\end{aligned} \tag{9.9}$$

To solve such an optimization problem, one needs the Fisher informations about the frequencies $f_\ell$ or $\phi_m$ obtained from sampling at time $t$. In the case where we are interested in the

frequencies $f_\ell$,

$$\mathscr{I}_{f_\ell}(t) = \frac{1}{p(t)(1-p(t))}\left(\frac{\partial p(t)}{\partial f_\ell}\right)^2 \tag{9.10}$$

where the derivative of the time-varying probability with respect to frequency $f_\ell$ is

$$\frac{\partial p(t)}{\partial f_\ell} = -\frac{2\pi t}{2^{N_f}}\sin(2\pi f_\ell t)\left(\prod_{\substack{m=1\\ m\neq\ell}}^{N_f}\cos(2\pi f_m t)\right)\exp\left(-\frac{t}{T_2}\right). \tag{9.11}$$

Similarly, in the case where we are interested in the frequencies $\phi_m$,

$$\mathscr{I}_{\phi_m}(t) = \frac{1}{p(t)(1-p(t))}\left(\frac{\partial p(t)}{\partial \phi_m}\right)^2, \tag{9.12}$$

where the derivative of the time-varying probability with respect to frequency $\phi_m$ is:

$$\frac{\partial p(t)}{\partial \phi_m} = -\frac{2\pi t}{(2^{N_f})(2^{N_f-1})}\sin(2\pi\phi_m t)\exp\left(-\frac{t}{T_2}\right). \tag{9.13}$$

While these expressions for the Fisher information may seem non-inviting, we will present approximations thereof in the next section, where we consider the case of a two spin process.

## 9.5 Two Spin Scenario

We consider the case of sampling a two-spin process. Given that the frequencies $\phi_m$ are linear combinations of the frequencies $f_\ell$, and as the number $N_f$ of frequencies is even, the resulting summed oscillations will exhibit a low frequency and a high frequency. In fact, the probability of the sampled process $X(t)$ taking value 1 takes the following form:

$$p(t) = \mathbb{P}[x(t)=1] = \frac{1}{4}\left(1+\left(\frac{1}{2}\cos(2\pi\phi_{\text{fast}}t)+\frac{1}{2}\cos(2\pi\phi_{\text{slow}}t)\right)\exp\left(-\frac{t}{T_2}\right)\right), \tag{9.14}$$

where $\phi_{\text{fast}} = f_1 + f_2$ is the high frequency, leading to a fast oscillation in $p(t)$, and $\phi_{\text{slow}} = |f_1 - f_2|$ is the low frequency, leading to a slow oscillation.

**Time-varying Bernoulli probability for two spins**

**Example 9.1.**
Notice how the two frequencies $f_1$ and $f_2$ result in a slow oscillation which affects the general shape of the signal in Fig. 9.1 and a fast oscillation which is more clearly discerned in the inset. Further notice how the evolution time $T_2$ ensures the timely

decay of the function to a value of $1/2^{N_f} = 1/4$.

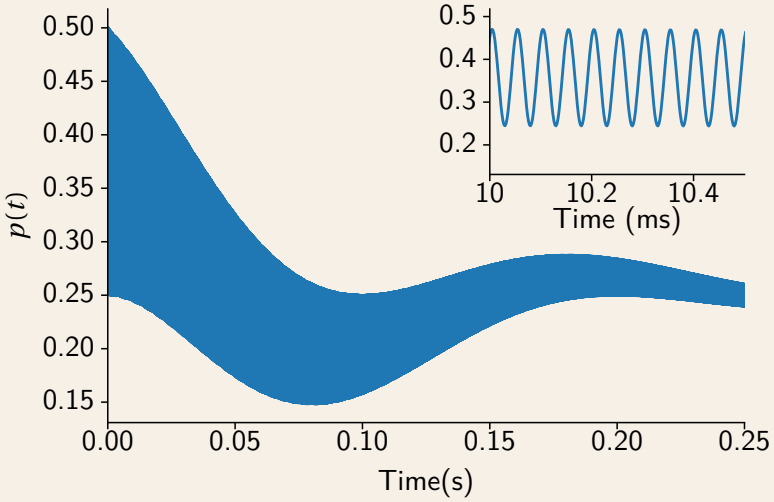



Figure 9.1 – The probability of the process $X(t)$ taking value 1 at time $t$. The signal has an evolution time of $T_2 = 0.1\text{s}$ and frequencies $f_1 = 9.998\text{KHz}$ and $f_2 = 9.993\text{KHz}$.

### 9.5.1 Fisher Information and Approximations

The slow frequency $\phi_{\text{slow}}$ has an effect not only on the probability, but also on the Fisher information about the parameters obtained from any sample at time $t$.

**Fisher information about $f_1$ and $f_2$**

**Example 9.2.**

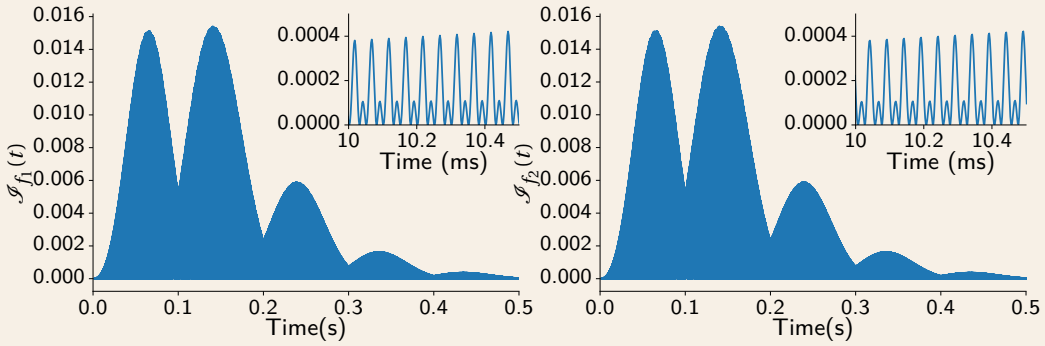



Figure 9.2 – The Fisher information $\mathscr{I}_{f_1}(t)$ and $\mathscr{I}_{f_2}(t)$ about each of the frequencies $f_1$ and $f_2$ obtained from sampling the same process as in Fig. 9.1 at time $t$. The process has an evolution time of $T_2 = 0.1\text{s}$ and frequencies $f_1 = 9.998\text{KHz}$ and $f_2 = 9.993\text{KHz}$.

In Fig. 9.2, we examine the Fisher informations $\mathcal{I}_{f_1}(t)$ and $\mathcal{I}_{f_2}(t)$ about each of the variables $f_1$ and $f_2$ as defined in (9.10) and (9.11).

As is clear from Fig. 9.2, both the low and high frequencies $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$ have an effect on the Fisher information about each of the variables $f_1$ and $f_2$. To attempt to isolate these effects, we now look at the Fisher informations $\mathcal{I}_{\phi_{\text{slow}}}(t)$ and $\mathcal{I}_{\phi_{\text{fast}}}(t)$ about the low frequency $\phi_{\text{slow}}$ and the high frequency $\phi_{\text{fast}}$, respectively, as defined in (9.12) and (9.13).

**Fisher information about $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$**

**Example 9.3.**

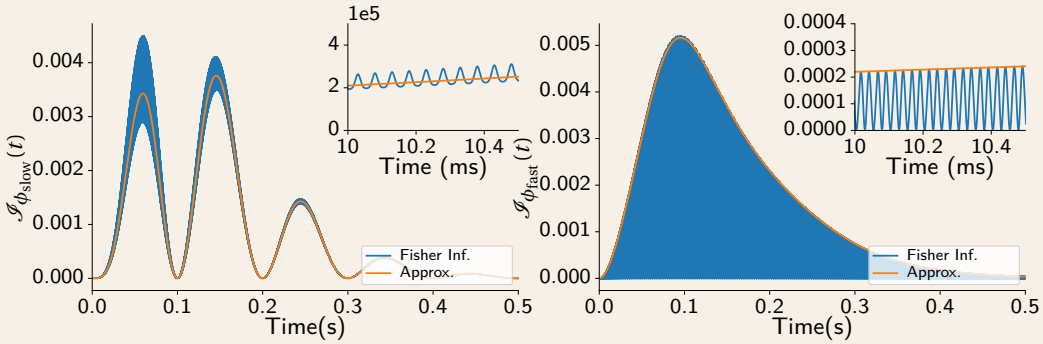



Figure 9.3 – (In blue) The Fisher information $\mathcal{I}_{\phi_{\text{slow}}}(t)$ and $\mathcal{I}_{\phi_{\text{fast}}}(t)$ about each of the frequencies $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$ obtained from sampling the same process as in Fig. 9.1 at time $t$. (In orange), the approximation $g_{\text{slow}}(t)$ (to the left)and $g_{\text{fast}}(t)$ (to the right) to the time-varying Fisher information as defined in (9.16) and (9.17), respectively. The process has an evolution time of $T_2 = 0.1\text{s}$ and frequencies $f_1 = 9.998\text{KHz}$ and $f_2 = 9.993\text{KHz}$, i.e. $\phi_{\text{slow}} = 5\text{Hz}$ and $\phi_{\text{fast}} = 19.991\text{KHz}$.

As we see from Fig.9.3, considering the Fisher information about the low and high frequencies separately allows disentangling the contribution of the two on the Fisher informations. As such, for the continuation of this section, we choose to consider the low and high frequencies $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$ to be our variables of interest.

Similarly to our treatment in Chapter 8, we look for an approximation of the Fisher information that is easier to analyze than the expression in (9.12).

To do so, we first define $\tilde{p}(t)$ to resemble $p(t)$, omitting the fast oscillation:

$$\tilde{p}(t) = \frac{1}{4}\left(1 + \frac{1}{2}\cos\left(2\pi\phi_{\text{slow}}t\right)\exp\left(-t/T_2\right)\right). \tag{9.15}$$

Then we define $g_{\text{slow}}(t)$ to be the approximation of $\mathscr{I}_{\phi_{\text{slow}}}(t)$, where

$$g_{\text{slow}}(t) = \frac{1}{\tilde{p}(t)(1-\tilde{p}(t))} t^2 \exp(-2t/T_2) \sin^2\left(2\pi\phi_{\text{slow}} t\right), \tag{9.16}$$

and $g_{\text{fast}}(t)$ to be the approximation of $\mathscr{I}_{\phi_{\text{fast}}}(t)$, where

$$g_{\text{fast}}(t) = \frac{1}{\tilde{p}(t)(1-\tilde{p}(t))} t^2 \exp(-2t/T_2). \tag{9.17}$$

We plot $g_{\text{slow}}(t)$ and $g_{\text{fast}}(t)$ alongside $\mathscr{I}_{\phi_{\text{slow}}}(t)$ and $\mathscr{I}_{\phi_{\text{fast}}}(t)$ respectively, in Fig. 9.3, and see how these approximations indeed follow the slower fluctuations of the true Fisher information.

### 9.5.2 Our Adaptive Approach

We have defined the Fisher information obtained about $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$ from sampling at time $t$, and we have examined its shape as a function of the low and high frequency of the process.

As in the single spin case, the *unknown* frequencies affect the shape of the Fisher information as a function of time, thus requiring an adaptive approach to sampling, where one alternates between estimating the frequencies and choosing the ideal sampling times depending on the Fisher information. In the two-spin case, $\phi_{\text{slow}}$ additionally affects the general shape of the Fisher information, thus requiring us to slightly change our adaptive sampling scheme.

First, we assume that we wish to place equal weight on the variance of each of the two frequencies $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$ and thus aim to solve the optimization problem in 9.9, where the next sampling point is chosen to minimize the overall variance.

Furthermore, when choosing the next sampling location, we wish to account for information gained in previous sampling steps. More specifically, we note that if a variance is already low because of previous samples, an extra sample reduces this variance less than if this variance were originally high. This occurs because new information gained from samples is *added* to the original Fisher information, but the variance is lower bounded by the *inverse* of this information. Thus, if many samples have already been taken such that they minimize the variance on, say, $\phi_{\text{slow}}$, it would be useful to now take samples that reduce the variance on $\phi_{\text{fast}}$.

Therefore, for any parameter $\phi_m$, we call the variance difference $\Delta\text{Var}_{\phi_m}\left(\mathbf{t}_{\text{prev}}, \mathbf{t}_{\mathbf{n}}\right)$, where

$$\Delta\text{Var}_{\phi_m}\left(\mathbf{t}_{\text{prev}}, \mathbf{t}_{\mathbf{n}}\right) = \mathbb{E}\left[\left(\hat{\phi}_m\left(\mathbf{t}_{\text{prev}}\right) - \phi_m\right)^2\right] - \mathbb{E}\left[\left(\hat{\phi}_m\left(\mathbf{t}_{\text{prev}} + \mathbf{t}_{\mathbf{n}}\right) - \phi_m\right)^2\right] \tag{9.18}$$

where $\hat{\phi}_m(\mathbf{t})$ denotes the estimate of $\phi_m$ using samples at times $\mathbf{t}$ , where $\mathbf{t}_{\text{prev}} + \mathbf{t}_{\mathbf{n}}$ denotes the concatenation of $\mathbf{t}_{\text{prev}}$ and $\mathbf{t}_{\mathbf{n}}$, and where the variances are lower bounded by the respective

Fisher informations:

$$\mathbb{E}\left[\left(\hat{\phi}_m\left(\mathbf{t}_{\text{prev}}+\mathbf{t_n}\right)-\phi_m\right)^2\right] \geq \frac{1}{\mathscr{I}_{\phi_m}(\mathbf{t}_{\text{prev}}+\mathbf{t_n})} = \frac{1}{\mathscr{I}_{\phi_m}(\mathbf{t}_{\text{prev}})+\mathscr{I}_{\phi_m}(\mathbf{t_n})}. \tag{9.19}$$

Finally, we define an estimate of added information $\tilde{\mathscr{I}}(\mathbf{t}_{\text{prev}}, t)$ which is the inverse of the sum of the variance differences:

$$\tilde{\mathscr{I}}_{\phi_m}(\mathbf{t}_{\text{prev}}, \mathbf{t_n}) = \frac{1}{\Delta\text{Var}_{\phi_{\text{slow}}}\left(\mathbf{t}_{\text{prev}}, \mathbf{t_n}\right) + \Delta\text{Var}_{\phi_{\text{fast}}}\left(\mathbf{t}_{\text{prev}}, \mathbf{t_n}\right)} \tag{9.20}$$

We call this value $\tilde{\mathscr{I}}_{\phi_m}(\mathbf{t}_{\text{prev}}, \mathbf{t_n})$ the *added* information obtained about $\phi_m$ from additionally sampling at times $\mathbf{t_n}$ after already having sampled at $\mathbf{t}_{\text{prev}}$. Notice that in the case of a single spin, this added information $\tilde{\mathscr{I}}_f(\mathbf{t}_{\text{prev}}, t)$ is exactly equal to the Fisher information $\mathscr{I}_f(\mathbf{t}_{\text{prev}}, t)$ with respect to the single frequency $f$ at time $t$.

We can now pose a new optimization problem, such that the total added information is maximized under time constraints:

$$\begin{aligned} \underset{[t_n]_{n=1}^N}{\text{argmax}} \quad & \tilde{\mathscr{I}}(\mathbf{t}_{\text{prev}}, [t_n]_{n=1}^N) \\ \text{subj. to} \quad & \sum_{n=1}^{N} t_n \leq T. \end{aligned} \tag{9.21}$$

where $\mathbf{t}_{\text{prev}}$ is a set of known previous sampling times.

As before, rather than solving the above constrained optimization problem, we rather aim to maximize the added information *per* unit of time:

$$t = \underset{u}{\text{argmax}} \quad \tilde{\mathscr{I}}(\mathbf{t}_{\text{prev}}, u)/u. \tag{9.22}$$

To be fair, in the single spin case, solving this simplified optimization problem, in addition to a few extra steps, resulted in solving the original problem in (9.21). This is not the case here, in the multi-spin scenario, but we nonetheless use this simplified problem to devise an adaptive algorithm.

### Adaptive sampling scheme for double spin processes

**Definition 9.2.** The *adaptive sampling scheme* is allotted a total permitted measurement time, assumes initial estimates for $\hat{\phi}_{\text{slow}}$ and $\hat{\phi}_{\text{fast}}$ and iterates through the following steps.

1. It assumes that a certain fraction of the permitted measurement time can be

used to sample the process $X(t)$.

2. It chooses the sample times $t_n$ at at least 2 locations such that they maximize the normalized, *added* Fisher information $\tilde{\mathscr{I}}(\mathbf{t}_{\text{prev}}, \mathbf{t_n})$ given this fraction of the measurement time, according to the latest estimate of $\hat{\phi}_{\text{slow}}$ and $\hat{\phi}_{\text{fast}}$. This is done by:
   (a) maximizing the added fisher info by considering only the slow frequecy, thus replacing $\mathscr{I}_{\phi\text{fast}}(t)$ and $\mathscr{I}_{\phi\text{slow}}(t)$ by their respective approximations $g_{\text{fast}}(t)$ and $g_{\text{slow}}(t)$ to find the general positioning of the optima in the normalized added Fisher information, and
   (b) refining the estimate by now maximizing the true value for $\tilde{\mathscr{I}}(\mathbf{t}_{\text{prev}}, t)$, taking into account the fast frequency's influence on the normalized added Fisher information by considering the true expressions for $\mathscr{I}_{\phi\text{fast}}(t)$ and $\mathscr{I}_{\phi\text{slow}}(t)$.
3. It samples the process and obtains new estimates $\hat{\phi}_{\text{slow}}$ and $\hat{\phi}_{\text{fast}}$ from all the samples it has obtained so far.
4. It repeats these steps until the total permitted measurement time has elapsed.

Note that, if aliasing can occur as described in Section 8.4.3, step 2 is modified such that an addition sample location that is close to the Nyquist sampling time are also chosen.

The step size is chosen as described in Section 8.5.3 and the samples are taken at different times and repeated depending on the normalized, added information at each time.

### 9.5.3 Simulations

We compare our adaptive sampling approach to uniform and exponential sampling under different assumptions for the evolution time $T_2$, and assuming the carrier frequency $f_{\text{carrier}}$ takes a value of $100\text{KHz}$.

More practically, we simulate the process with time-varying probability described in (9.1) and with frequencies drawn uniformly at random from $\left[f_{\text{carrier}} - 10, f_{\text{carrier}}\right)$.

Then, we simulate three sampling approaches on such a process: our adaptive approach, a uniform approach and an exponential approach. Once the samples are obtained, the recovery of the frequencies is always performed by iteratively solving for the maximum likelihood estimate of the frequencies $\phi_{\text{slow}}$ and $\phi_{\text{fast}}$, as described in Section 9.3.

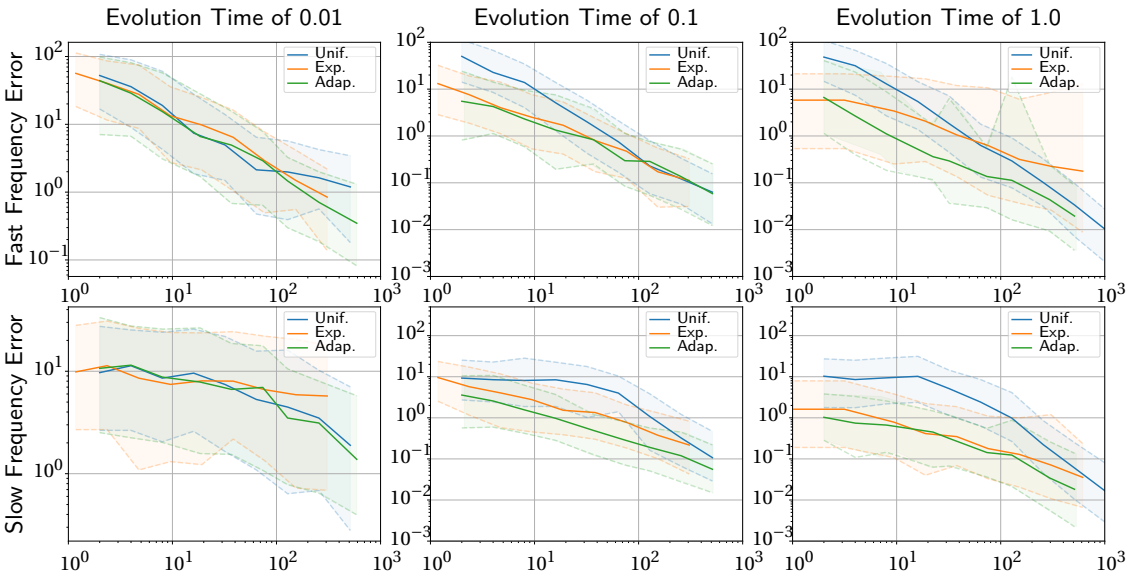



Figure 9.4 – Reconstruction error as a function of total measurement time using the (blue) adaptive sampling scheme, the (green) exponential scheme and the (orange) uniform sampling scheme. We plot the median squared reconstruction error using the solid lines, as well as the first and third quartile using the dashed lines. The top subplots represent the squared reconstruction error of the high frequency $\phi_{\text{fast}}$ and the bottom subplots represent the squared reconstruction error of the low frequency $\phi_{\text{slow}}$, for different values of evolution time $T_2$ as specified by the column labels.

Each sampling scheme is tested on 200 processes drawn at random. Our plots in Fig. 9.4 present, per sampling approach, the median squared error of the frequency estimate, as well as the first and fourth quartiles, plotted against varying total measurement times, on log-log plots.

We notice that, in some cases of evolution time, our adaptive approach surpasses the other two approaches in terms of accuracy. When this is not the case, our approach at least does not perform worse than the others and provides an estimation with consistent performance. Note for example, in the case of an evolution time of 0.01s, the adaptive approach continues to decrease as the total measurement time increases, although we see a slower decrease of the error of the high frequency estimate in the uniform sampling case and a slower decrease of the error of the low frequency in the exponential sampling case.

## 9.6 Conclusion

We have formulated the problem of recovering frequencies from Bernoulli trials obtained from sampling a multi-spin process and have shown how the frequencies of the process influence the change of Fisher information about the parameters of interest with respect to the sample

time.

We specifically considered the two-spin case and developped an adaptive sampling scheme which iteratively estimates the parameters and chooses the ideal sampling times, which maximize an estimate of added information. Then, we compared our approach to classical sampling and exponential sampling in simulations, and found a favorable performance of our approach compared to the others.

# Conclusion

We have presented the time-varying probability distribution that underlies samples of the fluorescence of an electron spin which is coupled to nuclear spins of interest and exhibits frequencies we would like to recover.

We have studied the case where this electron is coupled to one spin and to multiple spins, leading to one or multiple frequencies to recover, respectively. We have shown how to devise an optimal sampling strategy in the case of one or two spins, leveraging the fact that the Fisher information varies with time, depending on the evolution time and on the frequencies of interest. Then, we compared our technique to other sampling schemes in simulation and showed that our adaptive technique compares favorably.

Naturally, the optimization problems we pose, both for the recovery of the frequencies and for the choice of the sample time, become more difficult to solve as the number of frequencies to recover becomes higher. It is thus important to optimize speed and scalability, and further work should be invested into providing fast algorithms and implementations for sampling and recovery.

Another question to address concerns our sampling scheme's performance (in comparison to others) as the number of spins increases. With increasing number of frequencies, parameter uncertainty increases and the computed estimate of the Fisher information becomes less precise, for the same measurement time, given that the estimate of the Fisher information depends on the estimated frequencies. In these situations, it would be interesting to see what the adaptive scheme converges to at the initial phases of estimation, when parameter uncertainty is still high.

In this part of the thesis, we saw that sampling a process near half of its evolution time tends to maximize information per unit time, and that it is important to respect a Nyquist rate dictated by the range of values that the frequencies can take. An adaptive approach currently compares favorably to other approaches to sampling. Regardless of how well this effect scales with the number of unknowns, we hope that lessons can be drawn from the results in this part of the thesis to achieve more efficient sampling.

# Bibliography


K. Adam. Code for Sampling and Reconstruction of Bandlimited Signals with Multi Channel Time Encoding, Nov. 2019. URL https://doi.org/10.5281/zenodo.3558507.

K. Adam. A time encoding approach to training spiking neural networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022.

K. Adam, A. Scholefield, and M. Vetterli. Multi-channel time encoding for improved reconstruction of bandlimited signals. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 7963–7967. IEEE, 2019.

K. Adam, A. Scholefield, and M. Vetterli. Encoding and decoding mixed bandlimited signals using spiking integrate-and-fire neurons. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 9264–9268. IEEE, 2020a.

K. Adam, A. Scholefield, and M. Vetterli. Sampling and reconstruction of bandlimited signals with multi-channel time encoding. *IEEE Transactions on Signal Processing*, 68:1105–1119, 2020b.

K. Adam, A. Scholefield, and M. Vetterli. How asynchronous events encode video. In *2021 Conference Record of the fifty fifth Asilomar Conference on signals, systems and computers*. IEEE, 2021.

K. Adam, A. Scholefield, and M. Vetterli. Asynchrony increases efficiency: Time encoding of videos and low-rank signals. *IEEE Transactions on Signal Processing*, 70:105–116, 2022. doi: 10.1109/TSP.2021.3133709.

F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, et al. Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip. *IEEE transactions on computer-aided design of integrated circuits and systems*, 34(10):1537–1557, 2015.

A. Aldroubi and H. G. Feichtinger. Non-uniform sampling: exact reconstruction from non-uniformly distributed weighted-averages. In *Wavelet Analysis: Twenty Years' Developments*, pages 1–8. World Scientific, 2002.

R. Alexandru and P. L. Dragotti. Reconstructing classes of non-bandlimited signals from time encoded information. *IEEE Transactions on Signal Processing*, 68:747–763, 2019.

A. Amir, B. Taba, D. Berg, T. Melano, J. McKinstry, C. Di Nolfo, T. Nayak, A. Andreopoulos, G. Garreau, M. Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7243–7252, 2017.

H. A. Asl, P. L. Dragotti, and L. Baboulaz. Multichannel sampling of signals with finite rate of innovation. *IEEE Signal Processing Letters*, 17(8):762–765, 2010.

G. Baechler, N. Freris, R. F. Quick, and R. E. Crochiere. Finite rate of innovation based modeling and compression of ecg signals. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 1252–1256. IEEE, 2013.

G. Balasubramanian, I. Chan, R. Kolesov, M. Al-Hmoud, J. Tisler, C. Shin, C. Kim, A. Wojcik, P. R. Hemmer, A. Krueger, et al. Nanoscale imaging magnetometry with diamond spins under ambient conditions. *Nature*, 455(7213):648–651, 2008.

H. H. Bauschke and J. M. Borwein. On projection algorithms for solving convex feasibility problems. *SIAM Review*, 38(3):367–426, 1996.

M. Boerlin, C. K. Machens, and S. Denève. Predictive coding of dynamical variables in balanced spiking networks. *PLoS computational biology*, 9(11):e1003258, 2013.

S. M. Bohte, J. N. Kok, and H. La Poutre. Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing*, 48(1-4):17–37, 2002.

C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck. A 240× 180 130 db 3 $\mu$s latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10): 2333–2341, 2014.

A. N. Burkitt. A review of the integrate-and-fire neuron model: I. homogeneous synaptic input. *Biological Cybernetics*, 95(1):1–19, 2006.

J. A. Cadzow. Signal enhancement-a composite property mapping algorithm. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 36(1):49–62, 1988.

E. J. Candès and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational mathematics*, 9(6):717, 2009.

I. M. Comsa, K. Potempa, L. Versari, T. Fischbacher, A. Gesmundo, and J. Alakuijala. Temporal coding in spiking neural networks with alpha synaptic function. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 8529–8533. IEEE, 2020.

L. Cordone, B. Miramond, and S. Ferrante. Learning from event cameras with sparse spiking convolutional neural networks. *arXiv preprint arXiv:2104.12579*, 2021.

H. Cramér. *Mathematical Methods of Statistics (PMS-9), Volume 9*. Princeton university press, 2016.

M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, et al. Loihi: A neuromorphic manycore processor with on-chip learning. *Ieee Micro*, 38(1):82–99, 2018.

M. Davies, A. Wild, G. Orchard, Y. Sandamirskaya, G. A. F. Guerra, P. Joshi, P. Plank, and S. R. Risbud. Advancing neuromorphic computing with loihi: A survey of results and outlook. *Proceedings of the IEEE*, 109(5):911–934, 2021.

T. Delbrück, B. Linares-Barranco, E. Culurciello, and C. Posch. Activity-driven, event-based vision sensors. In *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, pages 2426–2429. IEEE, 2010.

P. L. Dragotti, M. Vetterli, and T. Blu. Sampling moments and reconstructing signals of finite rate of innovation: Shannon meets strang–fix. *IEEE Transactions on signal processing*, 55 (5):1741–1757, 2007.

H. C. Duwek, A. Shalumov, and E. E. Tsur. Image reconstruction from neuromorphic event cameras using laplacian-prediction and poisson integration with spiking and artificial neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1333–1341, 2021.

S. M. Fazel. Matrix rank minimization with applications. 2003.

H. G. Feichtinger and K. Gröchenig. Theory and practice of irregular sampling. *Wavelets: Mathematics and Applications*, 1994:305–363, 1994.

H. G. Feichtinger, J. C. Príncipe, J. L. Romero, A. S. Alvarado, and G. A. Velasco. Approximate reconstruction of bandlimited functions for the integrate and fire sampler. *Advances in computational mathematics*, 36(1):67–78, 2012.

D. Florescu and D. Coca. A novel reconstruction framework for time-encoded signals with integrate-and-fire neurons. *Neural computation*, 27(9):1872–1898, 2015.

G. Gallego, T. Delbruck, G. M. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

H. K. Galoogahi, A. Fagg, C. Huang, D. Ramanan, and S. Lucey. Need for speed: A benchmark for higher frame rate object tracking. *arXiv preprint arXiv:1703.05884*, 2017.

D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza. Video to events: Recycling video datasets for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3586–3595, 2020.

W. Gerstner, W. M. Kistler, R. Naud, and L. Paninski. *Neuronal dynamics: From single neurons to networks and models of cognition*. Cambridge University Press, 2014.

S. Ghosh-Dastidar and H. Adeli. Spiking neural networks. *International journal of neural systems*, 19(04):295–308, 2009.

D. Gontier and M. Vetterli. Sampling based on timing: Time encoding machines on shift-invariant subspaces. *Applied and Computational Harmonic Analysis*, 36(1):63–78, 2014.

D. F. Goodman and R. Brette. The brian simulator. *Frontiers in Neuroscience*, 3:26, 2009.

A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural networks*, 18(5-6):602–610, 2005.

M. Hilton, R. Alexandru, and P. L. Dragotti. Guaranteed reconstruction from integrate-and-fire neurons with alpha synaptic activation. In *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5474–5478. IEEE, 2021a.

M. Hilton, R. Alexandru, and P. L. Dragotti. Time encoding using the hyperbolic secant kernel. In *2020 28th European Signal Processing Conference (EUSIPCO)*, pages 2304–2308. IEEE, 2021b.

D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of Physiology*, 160(1):106–154, 1962.

I. A. Ibragimov and R. Z. Has' Minskii. *Statistical estimation: asymptotic theory*, volume 16. Springer Science & Business Media, 2013.

G. Indiveri and T. K. Horiuchi. Frontiers in neuromorphic engineering. *Frontiers in neuroscience*, 5:118, 2011.

G. Indiveri and S.-C. Liu. Memory and information processing in neuromorphic systems. *Proceedings of the IEEE*, 103(8):1379–1397, 2015.

G. Indiveri, B. Linares-Barranco, T. J. Hamilton, A. Van Schaik, R. Etienne-Cummings, T. Delbruck, S.-C. Liu, P. Dudek, P. Häfliger, S. Renaud, et al. Neuromorphic silicon neuron circuits. *Frontiers in neuroscience*, 5:73, 2011.

S. Jaffard. A density criterion for frames of complex exponentials. *The Michigan Mathematical Journal*, 38(3):339–348, 1991.

P. Jain, R. Meka, and I. S. Dhillon. Guaranteed rank minimization via singular value projection. In *Advances in Neural Information Processing Systems*, pages 937–945, 2010.

S. G. Johnson. The NLopt nonlinear-optimization package. URL http://github.com/stevengj/nlopt.

A. J. Kamath, S. Rudresh, and C. S. Seelamantula. Time encoding of finite-rate-of-innovation signals. *arXiv preprint arXiv:2107.03344*, 2021.

E. R. Kandel, J. H. Schwartz, T. M. Jessell, S. Siegelbaum, A. J. Hudspeth, and S. Mack. *Principles of neural science*, volume 4. McGraw-hill New York, 2000.

H. Kim, S. Leutenegger, and A. J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European Conference on Computer Vision*, pages 349–364. Springer, 2016.

H.-Y. Lai, P. Martínez-Nuevo, and A. V. Oppenheim. An iterative reconstruction algorithm for amplitude sampling. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4576–4580. IEEE, 2017.

A. A. Lazar. Multichannel time encoding with integrate-and-fire neurons. *Neurocomputing*, 65:401–407, 2005.

A. A. Lazar. Population encoding with hodgkin–huxley neurons. *IEEE Transactions on Information Theory/Professional Technical Group on Information Theory*, 56(2), 2010.

A. A. Lazar and E. A. Pnevmatikakis. Faithful representation of stimuli with a population of integrate-and-fire neurons. *Neural Computation*, 20(11):2715–2744, 2008.

A. A. Lazar and E. A. Pnevmatikakis. Reconstruction of sensory stimuli encoded with integrate-and-fire neurons with random thresholds. *EURASIP Journal on Advances in Signal Processing*, 2009:1–14, 2009.

A. A. Lazar and E. A. Pnevmatikakis. Video time encoding machines. *IEEE Transactions on Neural Networks*, 22(3):461–473, 2011.

A. A. Lazar and L. T. Tóth. Time encoding and perfect recovery of bandlimited signals. In *2003 IEEE International Conference on Acoustics, Speech, and Signal Processing, 2003. Proceedings.(ICASSP'03).*, volume 6, pages VI–709. IEEE, 2003.

A. A. Lazar and L. T. Tóth. Perfect recovery and sensitivity analysis of time encoded bandlimited signals. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 51(10): 2060–2073, 2004.

A. A. Lazar and Y. Zhou. Reconstructing natural visual scenes from spike times. *Proceedings of the IEEE*, 102(10):1500–1519, 2014.

A. A. Lazar, E. K. Simonyi, and L. T. Tóth. An overcomplete stitching algorithm for time decoding machines. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 55(9): 2619–2630, 2008.

Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

S.-C. Liu and T. Delbruck. Neuromorphic sensory systems. *Current opinion in neurobiology*, 20(3):288–295, 2010.

S.-C. Liu, B. Rueckauer, E. Ceolini, A. Huber, and T. Delbruck. Event-driven sensing for efficient perception: Vision and audition algorithms. *IEEE Signal Processing Magazine*, 36 (6):29–37, 2019.

C. Ma, J. Xu, and Q. Yu. Temporal dependent local learning for deep spiking neural networks. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–7. IEEE, 2021.

W. Maass. Networks of spiking neurons: the third generation of neural network models. *Neural Networks*, 10(9):1659–1671, 1997.

P. Martínez-Nuevo, H.-Y. Lai, and A. V. Oppenheim. Amplitude sampling. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 17–22. IEEE, 2016.

J. R. Maze, P. L. Stanwix, J. S. Hodges, S. Hong, J. M. Taylor, P. Cappellaro, L. Jiang, M. G. Dutt, E. Togan, A. Zibrov, et al. Nanoscale magnetic sensing with an individual electronic spin in diamond. *Nature*, 455(7213):644–647, 2008.

P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, et al. A million spiking-neuron integrated circuit with a scalable communication network and interface. *Science*, 345(6197):668–673, 2014.

E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017.

E. O. Neftci, H. Mostafa, and F. Zenke. Surrogate gradient learning in spiking neural networks: Bringing the power of gradient-based optimization to spiking neural networks. *IEEE Signal Processing Magazine*, 36(6):51–63, 2019.

M. Pacholska, K. Adam, A. Scholefield, and M. Vetterli. Matrix recovery from bilinear and quadratic measurements. *arXiv preprint arXiv:2001.04933*, 2020.

H. Pan, T. Blu, and P. L. Dragotti. Sampling curves with finite rate of innovation. *IEEE Transactions on Signal Processing*, 62(2):458–471, 2013.

H. Pan, T. Blu, and M. Vetterli. Towards generalized FRI sampling with an application to source resolution in radioastronomy. *IEEE Transactions on Signal Processing*, 65(4): 821–835, 2017.

J.-P. Pfister and W. Gerstner. Triplets of spikes in a model of spike timing-dependent plasticity. *Journal of Neuroscience*, 26(38):9673–9682, 2006.

H. Rebecq, D. Gehrig, and D. Scaramuzza. Esim: an open event camera simulator. In *Conference on Robot Learning*, pages 969–982. PMLR, 2018.

H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 2019.

B. Recht, M. Fazel, and P. A. Parrilo. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM review*, 52(3):471–501, 2010.

S. Rudresh, A. J. Kamath, and C. S. Seelamantula. A time-based sampling framework for finite-rate-of-innovation signals. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5585–5589. IEEE, 2020.

D. Rugar, R. Budakian, H. Mamin, and B. Chui. Single spin detection by magnetic resonance force microscopy. *Nature*, 430(6997):329–332, 2004.

S. Saxena and M. Dahleh. Analyzing the effect of an integrate and fire encoder and decoder in feedback. In *53rd IEEE Conference on Decision and Control*, pages 3821–3828. IEEE, 2014.

J. Schmidhuber. Deep learning in neural networks: An overview. *Neural networks*, 61:85–117, 2015.

J.-s. Seo, B. Brezzo, Y. Liu, B. D. Parker, S. K. Esser, R. K. Montoye, B. Rajendran, J. A. Tierno, L. Chang, D. S. Modha, et al. A 45nm cmos neuromorphic chip with a scalable architecture for learning in networks of spiking neurons. In *2011 IEEE Custom Integrated Circuits Conference (CICC)*, pages 1–4. IEEE, 2011.

C. E. Shannon. Communication in the presence of noise. *Proceedings of the IRE*, 37(1): 10–21, 1949.

W. Sun and X. Zhou. Reconstruction of band-limited signals from local averages. *IEEE Transactions on Information Theory*, 48(11):2955–2963, 2002a.

W. Sun and X. Zhou. Reconstruction of band-limited functions from local averages. *Constructive Approximation*, 18(2):205–222, 2002b.

T. Tao. *An introduction to measure theory*, volume 126. American Mathematical Society Providence, RI, 2011.

A. Tavanaei, M. Ghodrati, S. R. Kheradpisheh, T. Masquelier, and A. Maida. Deep learning in spiking neural networks. *Neural Networks*, 111:47–63, 2019.

N. T. Thao and D. Rzepka. Pseudo-inversion of time encoding of bandlimited signals, 2019.

N. T. Thao and D. Rzepka. Time encoding of bandlimited signals: reconstruction by pseudo-inversion and time-varying multiplierless fir filtering. *IEEE Transactions on Signal Processing*, 69:341–356, 2020.

N. T. Thao, D. Rzepka, and M. Miśkowicz. Bandlimited signal reconstruction from leaky integrate-and-fire encoding using pocs. *arXiv preprint arXiv:2201.03006*, 2022.

M. Unser. Sampling – 50 years after shannon. *Proceedings of the IEEE*, 88(4):569–587, 2000.

R. VanRullen, R. Guyonneau, and S. J. Thorpe. Spike times make sense. *Trends in neurosciences*, 28(1):1–4, 2005.

M. Vetterli, P. Marziliano, and T. Blu. Sampling signals with finite rate of innovation. *IEEE Transactions on Signal Processing*, 50(6):1417–1428, 2002.

M. Vetterli, J. Kovačević, and V. K. Goyal. *Foundations of signal processing*. Cambridge University Press, 2014.

G. Waldherr, J. Beck, P. Neumann, R. Said, M. Nitsche, M. Markham, D. Twitchen, J. Twamley, F. Jelezko, and J. Wrachtrup. High-dynamic-range magnetometry with a single nuclear spin in diamond. *Nature nanotechnology*, 7(2):105–108, 2012.

T. C. Wunderlich and C. Pehle. Event-based backpropagation can compute exact gradients for spiking neural networks. *Scientific Reports*, 11(1):1–17, 2021.

J. Zhao, R. Xiong, H. Liu, J. Zhang, and H. Tiejun. Spk2imgnet: Learning to reconstruct dynamic scene from continuous spike stream. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11996–12005, 2021.

Y. Zheng, L. Zheng, Z. Yu, B. Shi, Y. Tian, and T. Huang. High-speed image reconstruction through short-term plasticity for spiking cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6358–6367, 2021.

K. Zhong, P. Jain, and I. S. Dhillon. Efficient matrix sensing using rank-1 gaussian measurements. In *International conference on algorithmic learning theory*, pages 3–18. Springer, 2015.

# Curriculum Vitae

Karen Adam
karen.adam@epfl.ch

## Education

**PhD in Computer and Communication Sciences:** Audiovisual Communication Laboratory, Ecole Polytechnique Fédérale de Lausanne (EPFL), *Sep 2017-Apr 2022*

**Bachelor of Engineering** in Computer and Communications Engineering, Minors in Maths and Biomedical Engineering:, American University of Beirut, *Sep 2013-Jun 2017*

## Research Experience

**Audiovisual Communications Laboratory, EPFL** Time encoding and decoding: from single-input, single-output systems to spiking neural networks, Supervisors: Martin Vetterli and Adam Scholefield, *Feb 2018 - Now*

**Audiovisual Communications Laboratory, EPFL** Sampling and reconstructing spin states using nitrogen vacancy sensing: developing signal processing tools for optimal estimation, Collaborator: Peter Maurer, *Jan 2019-Now*

**Laboratory for Computational Neuroscience, EPFL** Designing autoencoders that satisfy biological constraints, Supervisor: Aditya Gilra, *Sep 2017 - Jan 2018*

**American University of Beirut** Seizure onset and propagation analysis, Supervisor: Fadi Karameh, *Sep 2016 - Jun 2017*

**Neuroscience Statistics Research Lab, Massachusetts Institute of Technology (MIT)** Kalman filter based estimation of EEG content during anaesthesia, Supervisor: Emery N. Brown, *Jun 2016 - Aug 2016*

## Awards

**IC distinguished services award,** EPFL, *Dec 2019*

**IC distinguished services award,** EPFL, *Dec 2020*

**IC teaching assistant award,** EPFL, *Dec 2021*

**EDIC Fellowship,** EPFL, *Sep 2017-Aug 2018*

**Murex for Excellence award,** best collaborative software development project, *Jun 2017*

**Graduated with High Distinction,** American University of Beirut, *Jun 2017*

## Publications

K. Adam, "A time encoding approach to training spiking neural networks," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2022.

K. Adam, A. Scholefield, and M. Vetterli, "How asynchronous events encode video," in 2021 Conference Record of the fifty fifth Asilomar Conference on signals, systems and computers, IEEE, 2021.

K. Adam, A. Scholefield and M. Vetterli, "Asynchrony Increases Efficiency: Time Encoding of Videos and Low-Rank Signals," in IEEE Transactions on Signal Processing, vol. 70, pp. 105-116, 2022, doi: 10.1109/TSP.2021.3133709.

M. Pacholska, K. Adam, A. Scholefield, and M. Vetterli, "Matrix recovery from bilinear and quadratic measurements," arXiv preprint arXiv:2001.04933, 2020.

K. Adam, A. Scholefield, and M. Vetterli, "Encoding and decoding mixed bandlimited signals using spiking integrate-and-fire neurons," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) 2020.

K. Adam, A. Scholefield, and M. Vetterli, "Sampling and reconstruction of bandlimited signals with multi-channel time encoding," IEEE Transactions on Signal Processing, vol. 68, pp. 1105–1119, 2020.

K. Adam, A. Scholefield, and M. Vetterli, "Multi-channel time encoding for improved reconstruction of bandlimited signals," in IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP). IEEE, 2019, pp. 7963–7967.

## Languages Coded

Python, C++, Latex/Tex, Matlab, Java...

## Languages Spoken

English, French, Arabic, German