University of Zurich[UZH] | ETH zürich

Institute of Neuroinformatics

Master Thesis

Towards a Neural Implementation of the Kalman Filter

Institute of Neuroinformatics
University of Zurich (UZH)
Swiss Federal Institute of Technology (ETH) Zurich

November 2021

Supervised by:
Prof. Dr. Christoph von der Malsburg
Prof. Dr. Benjamin Grewe

Author:
Raffael Theiler

Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

Title of work (in block letters):

Authored by (in block letters):
For papers written by groups the names of all authors are required.

Name(s):	**First name(s):**

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the 'Citation etiquette' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

Place, date	**Signature(s)**

For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.

Contents

Preface

Here, I would like to thank every person that I had the pleasure to work with and everyone who had supported me during the completion of my Master's thesis. First of all, I would like to thank my supervisor Prof. Christoph von der Malsburg for his supervision, patience and support throughout this project. I can't thank him enough for the thought-provoking, motivating and fruitful discussions and the sharing of his insights, which led me to learn a lot about Neuroscience and finally motivated me to pursue an academic career. I would also like to thank Prof. Benjamin Grewe for his willingness to be my institute supervisor. I was greatly inspired by our joint research project, which led me to dive into the fundamentals of neural computation. Additionally, I would like to greatly thank my friends Jonas and Eric for their academic companionship during my Master studies in Zurich. Thanks to them, I have always enjoyed working on challenging projects. I would also like to thank those who unconditionally and generously supported me on an emotional level during this Master. Jenny and Cheryne, I will always be grateful for all the support and patience. A special thanks to Kaan with whom I started this journey and who was always there and to Kevin for reading through all of my work! Finally, I would like to thank my family for always standing behind me and for their practical and emotional support.

Abstract

An essential component of decision-making is our capability to predict. It is the consistency between our model-based predictions and subsequent sensory input that gives us the necessary confidence to take decisive actions. As a consequence of our inherently noisy and uncertain world, predictive models in living systems and technical applications are probabilistic. A well-known algorithm in that domain is the linear Kalman filter. The Kalman filter represents its state as Gaussian distributed random variable. For biological systems, recent work presents a Bayesian decoder that recovers equivalent latent variables from neurons with Poisson-like variability in living systems [1], [2]. These virtual latent variables are derived and have no underlying neural representation. In this thesis, we follow the alternative hypothesis that the brain directly represents latent random variables with neural activity. We select macrocolumns [3] as candidate representation for their discrete, nonparametric activity distributions. Kalman's gain-based update rules do not apply to nonparametric random variables. Instead, we introduce a set of differential equations for the dynamic evolution of the neural activity on macrocolumns. We evaluate our filter on the task of linear motion estimation where the dynamical system governs input from a sensor and neural coincidence detectors. The filter successfully recovers the true state based on noisy measurements and demonstrates Kalman-like filtering. Additionally, we provide an agent-based simulator for interactive and quantitative experiments.

Acronyms

EDE Evolutionary Differential Equation. 14, 23, 25–27, 29, 30, 32, 35, 36, 38–41, 56–58

KF Kalman Filter. 2–5, 7–10, 15, 22, 24–26, 28, 30, 36–40, 42–44, 46–48, 57

NKF Neural Kalman Filter. 5, 15, 24, 25, 27, 36–40, 42–44, 46–55, 57

PPC Probabilistic Population Coding. 4, 8, 9, 11–13, 27, 56, 57

SBC Sampling-based Coding. 11–13, 23

Chapter 1

Introduction

Consistency between sensory input and model-based predictions of our brain is the basis for the confidence with which we rely on our senses. A confident brain reacts with surprise to mismatch between prediction and sensory input. A primary objective of the brain is to minimize surprise by constantly optimizing the internal state based on new observations.

A system must cope with noisy and inaccurate sensors in addition to captured randomness of the world. Therefore, both, biological and human-made systems are forced to develop strategies to deal with the noise and uncertainty in their inputs. A naive strategy is to improve the sensor quality, but that does not protect against input from inherently random processes. A more robust approach is a framework that encodes randomness.

In 1960, with his paper "a new approach to linear filtering and prediction problems", Kalman introduced such a framework that processes and interprets inputs as random variables for the technical domain [4]. As an equivalent to surprise, he computes a gain that expresses the trust in incoming sensor data. Due to its success in virtually every technical or quantitative field [5], the Kalman filter is still used in its original or in one of its many improved forms today. Historically, the Kalman filter (KF) was not meant to be analyzed from a neuroscientific perspective. The main objectives were accurate tracking of points in space-time and computational tractability.

Mammals were already using tracking intuitively long before it was formally written down by Kalman. We observe that our brain is able to estimate and predict the own bodily state as well as the state of the environment with great success. Even for complex tasks, such as hunting or navigating in the dark, humans and animals are able to rely on their understanding of their environment. Moreover, coherence between sensory input and model-based prediction is one of the fundamental effects that makes learning possible. Especially in hierarchical structures, the identification and classification of sensory mismatch is key to providing a reliable learning signal to subsequent units [6]. From a wider perspective, an agent can also take direct action to reduce uncertainty. This idea is worked out in the active inference framework [7]. In this thesis, however, we refrain from actively changing the environment.

It is observable that a detachment of the model state from the real world has

serious consequences. If an agent does not manage to keep coherence between the sensory input and the model, severe effects result. To humans, prolonged dishomeostasis reflects a chronic state of surprise and results in sickness and stress [6]. Machines may lose track of their current state and position in the environment and catastrophically fail.

1.1 Motivation

One would expect that the fields of estimation theory and neuroscience would harmonize well with respect to the goal of understanding state estimation and tracking. In the first field, there is a strong desire to implement exceptional tracking for technical applications, the second field demonstrates that there are neural models that work in practice. At the task level, the same goal is pursued: extracting and representing meaningful information to understand the environment for taking appropriate action. Sadly, the fields never took full advantage of each other, despite the efforts of pioneers like Marr [8]. With this thesis, we continue the discussion by combining knowledge from estimation theory, neuroscience, as well as evolutionary biology.
Observation of mammals hints at the fact that neural models are to some extent capable of estimating probabilistic quantities, such as the size of the event space, the outcome space and the assigned probability density. Hence we ask if the brain employs algorithms for universally applicable computational concepts above the individual task that we can use for our purpose. We believe that the correct way to deal with perceived randomness is to mirror it conceptually in the model. We observe that uncertainty is naturally embodied in the most fundamental building block of our brain, the neuron [1] and ask ourselves if probabilistic concepts also apply to structures formed by neurons. We identify two significant concepts of which the first is normalization. We assume that the brain implements normalization, aside from other canonical computations, as a modular unit [9]. Hence we assume that probabilistic representations over sets of neural units can be found at different sites.
The second concept is to express a belief that is independent of other variables. This computation is called marginalization. This type of probabilistic inference is performed by the nervous system in seemingly unrelated tasks [1].

1.1.1 Contribution

In this thesis, we aim to use universally applicable computational concepts of neural circuits, such as normalization and marginalization, to isolate a neural system that performs state estimation for the task of recovering the true velocity and position of linear motion. Our hypothesis is as follows: *the brain is able to perform Kalman-like filtering of sensory inputs.*
For a linear internal model with Gaussian noise, the optimal least squares estimator that combines sensory inputs and predictions, is realized by the KF (the estimator is equivalent to the maximum-likelihood estimator). Several authors have proposed models that demonstrate neural circuits that perform KF-like state estimation [1], [2], [10].

The first and second model approximate neural activity to be Poisson distributed. The authors apply Probabilistic Population Coding (PPC) and propose different decoders to arrive at a latent state representation. One may perform certain operations, such as sensory integration and linear shifts on this latent space representation. Using these operations, it is possible to derive weights and update rules for neural activity, such that the neural activity represents a KF. This approach has two shortcomings. Firstly, there is no neural representation of the estimated (latent) state. In a hierarchical neural circuit, this implies that circuits that depend on the Kalman-filtered state are required to lead with an additional decoding step if the aim is to further process the signal. Secondly, the model wants to reconstruct Gaussian random variables, which is an unlikely model assumption for general neural circuits.
The third model is based on Zhangs's line attractor model. The authors present a set of weight-dependent differential equations that update activities on a discrete set of neurons, such that the distribution reflects Kalman-optimal mixing of its activity with the sensory input. The disadvantage of this model is that the weights are expressed as a function of the true velocity which must be known to the algorithm.
The shortcomings that we found motivated us to choose a different algorithmic approach on state estimation. We propose an algorithm that performs a nonparametric processing of sensory input while all latent variables are represented as neural units. We show that for the task of first order linear motion estimation, our algorithm is able to achieve a near optimal state representation.

Chapter 2

Related Work

In this chapter we introduce the building blocks of the Neural Kalman Filter (NKF) and show its ties to previous work on hypercircles [11], a concept from computational biology that studies competitive macromolecules. In this work, we link the dynamic evolution of macromolecules to the evolution of neural populations. We aim to include all relevant concepts that firstly motivated the work on the NKF and secondly are used to evaluate the NKF algorithm. On the technical side, we introduce the KF and explain how the filter is derived from basic operations on normally distributed random variables. We introduce three different neural coding strategies, probabilistic population codes, sampling-based codes and finally macrocolums and explain why the last coding is most probable for our operation.

2.1 The Kalman Filter

> Here, we introduce the concepts of the KF as a member of the family of Bayesian filters in one dimension. For a multivariate KF derivation, we recommend the famous online resource (link)[1].

The KF is a framework to estimate and track the state of a system. The KF integrates sequential measurements to update its internal estimate of the observed world. The resulting estimate is usually closer to the true state of the world, than the naive strategy of fully trusting the latest sensor input. Because of its simplicity, the KF is quickly and efficiently implemented on weak hardware and is therefore a choice in many practical applications such as target tracking, navigation, signal processing, etc [12]. In chapter 4.2, we show a simple application of the KF that was used to benchmark the NKF in this line of work.

All distributions of the KF are by design Gaussian. In the simplest case, the state is encoded by a one-dimensional Gaussian distribution. The update process of the filter from time step $k-1$ to k happens in two steps. In the *prediction* step, the state x is estimated with data up to time step $k-1$. The process

[1] https://missingueverymoment.wordpress.com/2019/12/02/derivation-of-kalman-filter/

model f_x is a mathematical simulation of the system, but it does not consider sensor information. The prediction step makes use of f_x to predict the next time step.

$$\hat{x}_k = x_{k-1} + f_{x_{k-1}} \tag{2.1}$$

As both f_x and x are Gaussian random variables and the update step computes as the sum of the two, that sum is again a Gaussian-distributed variable $\mathcal{N}(\hat{\mu}_x, \hat{\sigma}_x^2)$, with:

$$\hat{\mu}_x = \mu_x + \mu_{f_x} \tag{2.2}$$
$$\hat{\sigma}_x^2 = \sigma_x^2 + \sigma_{f_x}^2 \tag{2.3}$$

Since time has passed and we predict without new data, it makes sense that the variance is increasing in this step. In the *update* step, we apply the Bayes theorem to find the posterior distribution on x. More precisely, that is the multiplication of the probability of the measurement given the current state (likelihood) $\mathcal{L}$ and the current state prediction $\hat{x}$ (prior):

$$p(x|z) = \frac{\overbrace{p(z|x)}^{\mathcal{L}} p(x)}{p(z)} \tag{2.4}$$

Therefore, including the sensor z, the new state is $x|z \propto \mathcal{L}\hat{x}$. Again, $\mathcal{L}$ and $\hat{x}$ are Gaussian random variables and the product of two Gaussians is a Gaussian $\mathcal{N}(\mu_x, \sigma_x^2)$:

$$\mu_x = \frac{\hat{\sigma}^2 \mu_z + \sigma_z^2 \hat{\mu}}{\hat{\sigma}^2 + \sigma_z^2} \tag{2.5}$$
$$\sigma_x^2 = \frac{\hat{\sigma}^2 \sigma_z^2}{\hat{\sigma}^2 + \sigma_z^2} \tag{2.6}$$

In a few steps, one can show that the new μ_x is a mixture of the state estimation $\hat{\mu}$ and the sensor input μ_z. This leads us to the final and integral derivation of the Kalman gain. We can rewrite equation 2.5 to:

$$\mu_x = \underbrace{\frac{\hat{\sigma}^2}{\hat{\sigma}^2 + \sigma_z^2}}_{W_1} \mu_z + \underbrace{\frac{\sigma_z^2}{\hat{\sigma}^2 + \sigma_z^2}}_{W_2} \hat{\mu} \tag{2.7}$$

By definition, the Kalman gain is $K = W_1 = \frac{\hat{\sigma}^2}{\hat{\sigma}^2 + \sigma_z^2}$ the weighting of μ_z. With some algebra, we write:

$$\mu_x = K\mu_z + (1 - K)\hat{\mu} \tag{2.8}$$
$$= \hat{\mu} + K(\mu_z - \hat{\mu}) \tag{2.9}$$

Which shows that weighted by the Kalman gain, we return a new state estimate that is in between μ_z and $\hat{\mu}$. For completeness, we can rewrite equation 2.6 in terms of K:

$$\sigma_x^2 = \frac{\hat{\sigma}^2\sigma_z^2}{\hat{\sigma}^2 + \sigma_z^2} = K\sigma_z^2 = (1-K)\hat{\sigma}^2 \tag{2.10}$$

Further intuition on the KF can be gained from [13] where we took inspiration from.

2.1.1 Technical Extensions to the Kalman Filter

Several extensions to the KF have been developed to address the shortcomings of linear filtering in the KF. The extended Kalman filter (EKF) [14] allows arbitrary, non-linear functions for the process model. It approximates the non-linear filtering with local linearization, The unscented Kalman filter (UKF) [15] is an extended EKF that not only linearly approximates along the mean but chooses a variety of sigma points to do so. Its accuracy increases with the number of sigma points. Particle filters were introduced for computational tractability. Internally, particle filters apply a Monte Carlo method. A set of particles approximate the state distribution of non-linear and non-Gaussian systems, hereby freeing the filter of the strict, but misleading view that the world may be encoded with Normal random variables.
In engineering, artificial neural networks are defined differently than in neuroscience. On the highest level, artificial neural networks are black-box estimators. An artificial neural network is a function $f_{\mathbf{\Theta}}$ with internal weights, $\mathbf{\Theta}$, that maps from a sample space to a label space $f_{\mathbf{\Theta}} : \mathcal{X} \to \mathcal{Y}$. In the supervised case, an artificial neural network improves its weights $\mathbf{\Theta}$, and therefore learns representations, by computing gradients with respect to $\mathbf{\Theta}$ of a loss function. Computationally, this is a global, synchronized update rule. It is very unlikely that the brain implements such a learning rule. Therefore, we categorize all KF that profit from the advances in deep learning as technical extensions.
An implementation of deep learning for KF addresses one of the challenges, the requirement of valid process equations. Process and measurement noise definitions need to be given a priori. Such parameters are a good fit to be replaced or enhanced with deep learning function approximations. An improved version of the KF implements this idea to refine the prediction $\hat{x}(k|k-1)$ and filtering $\hat{x}(k|k)$ states based on incoming data [16].

2.1.2 Neural Implementations of the Kalman Filter

While the original implementation of the KF was technically motivated, its success in applied domains led to the question whether a similar mechanism exists in the brain and whether an implementation with known elements of neural computation is possible. Neuroscientists were encouraged to ask if the operational firing patterns of neural circuits are to be interpreted as an encoded KF-like state estimation process.

In section 2.1, we show that the KF is the optimal choice for a linear model and Gaussian noise. The presence of an exact neural implementation KF thus implies that the brain would be capable of performing optimal linear estimation under these constraints. From that follows that the brain would need a Gaussian understanding of the involved random variables.
Even tough the Gaussian model is tractable and sufficient for many processes it is highly unlikely that the brain represents random variables exclusively as normally distributed. But if we assume that the brain performs near optimal inference with respect to the same sensory input, then a more fundamental question is how biological neural networks perform this computation.

Method 1

In [1] and [17], the authors show that Gaussian latent variables may be encoded using a Bayes encoder / decoder schema. That means that signals from a neural population may be transformed using a Bayesian encoder to a virtual, latent space where well-known probability distributions (e.g. Gaussians) describe the signal. The neurons are arranged as PPC and assumed to be Poisson distributed. The neurons are parametrized with gain g and rate r. They then proceed to define operations in latent space such as:

1. *multi-sensory integration*: The framework proposes Bayes optimal cue (neural response) combination if more than one cue provides information about a stimulus (external input). If the latent distributions are assumed to be independent and Gaussian, then the posterior variance μ_3 and mean σ_3^2 are computed non-linearly as:

$$\mu_3 = \frac{\sigma_2^2}{\sigma_1^2 + \sigma_2^2}\mu_1 + \frac{\sigma_1^2}{\sigma_1^2 + \sigma_2^2}\mu_2 \tag{2.11}$$

$$\frac{1}{\sigma_3^2} = \frac{1}{\sigma_1^2} + \frac{1}{\sigma_2^2} \tag{2.12}$$

   The authors show that the result of the Bayesian decoding, to arrive at the neural activity from equations 2.11 and 2.12, it is equivalent to computing the sum of the rates $\mathbf{r}_3 = \mathbf{r}_1 + \mathbf{r}_2$ and gains $g_3 = g_1 + g_2$ directly.

2. *linear coordinate transformations*: For linear coordinate transformations, the updated μ_A and σ_A^2 are computed linearly:

$$\mu_A = \mu_R + \mu_E \tag{2.13}$$

$$\sigma_A^2 = \sigma_R^2 + \sigma_E^2 \tag{2.14}$$

   This computation can also be performed directly from gains and rate vectors. It, however, is a very different computation compared to multi-sensory integration. The computation requires a quadratic nonlinearity and divisive normalization for the gain

$$g^A = \frac{g^R g^E}{g^R + g^E} \tag{2.15}$$

as well as to directly compute the updated rates with tuning parameters c and w as well as the previous rates r^R and r^E:

$$r_k^A = \frac{\sum_{ij} w_{ij}^k r_i^R r_j^E}{\sum_l c_l^R r_l^R + c_l^E r_l^E} \tag{2.16}$$

Without divisive normalization, a fixed linear decoder is not able to decode the activity which makes it difficult for downstream networks to make optimal use of the signal. It is out of scope of this work to show how eq. 2.16 is implemented neurally. The authors present a method where a two-dimensional intermediate layer for the neural activity of r_{ij} is incorporated.

3. *non-linear coordinate transforms*: While linear transformations are optimal, non-linear transformations are not and a certain information loss is reported by the authors. In the case when the change is small and the signals are unimodal with small variance, the information loss is small. In that situation, the method is locally linear and Gaussian and therefore near optimal and robust.

The introduced operations on PPC may be combined to mimic the functionality of a KF once the variables $r(t)$ and $s(t)$ vary over time. If we interpret those quantities as hidden Markov model, then $s(t)$ is the tracked hidden state and $\mathbf{r}$ the emitted observed sequence. For that task, the authors convert $\mathbf{r}$ to a deterministic firing rate $\mathbf{v}$ by counting spikes over a certain interval (10ms). They find for a one dimensional, linear, noisy movement $\frac{\mathrm{d}s}{\mathrm{d}t} = -\gamma s + \eta(t)$, the deterministic firing rate updates as

$$\frac{\mathrm{d}\mathbf{v}}{\mathrm{d}t} = \gamma \mathbf{W}\mathbf{v} - \sigma_\eta^2 (\mathbf{a}\mathbf{v})\, \mathbf{v} + \mathbf{M}\boldsymbol{\rho}^{in} + \mathbf{c}^{\times} f_c\left(\mathbf{v}, \boldsymbol{\rho}^{in}\right) \tag{2.17}$$

with an arbitrary functional $\mathbf{c}^{\times} f_c\left(\mathbf{v}, \boldsymbol{\rho}^{in}\right)$ on the spiking input $\boldsymbol{\rho}_i^{in} = \sum_j \delta\left(t - t_i^j\right)$. The terms $\gamma\mathbf{W}\mathbf{v}$ increase and $-\sigma_\eta^2 (\mathbf{a}\mathbf{v})\, \mathbf{v}$ decrease the neural activity which corresponds to a decrease / increase of the variance as we would expect from a KF. Further details, including appropriate derivation, should be taken from the appendix of the paper.

Method 2

In [10], the authors use a line attractor model to describe the average activity evolution of a set of neurons. In that discrete time framework, the membrane potential of each neuron u is updated as follows:

$$\mathbf{u}(t+1) = w\mathbf{J}\mathbf{f}[\mathbf{u}(t)] + \mathbf{I}(t+1) \tag{2.18}$$

where $\mathbf{J}$ represents neural interconnections, $\mathbf{f}$ is the activation rule that maps the membrane potential onto the firing rate and $\mathbf{I}$ is the input to the network. A key limitation to the network is that the velocity sensitivity $\gamma(t)$ is directly encoded in the $\mathbf{J}$ matrix and not learned:

$$\mathbf{J} = \mathbf{J}_{sym} + \gamma\,(t)\,\mathbf{J}_{asym} \tag{2.19}$$

A speed detector would therefore implement k different networks governed by equation 2.18 and perform some kind of model selection (not included in the paper). Another limitation is that divisive inhibition is implemented in a biologically implausible form in the activation function $\mathbf{f}$:

$$\mathbf{f}\,[\mathbf{u}] = \frac{[\mathbf{u}]_+}{S + \mu \sum_i [u_i]_+} \tag{2.20}$$

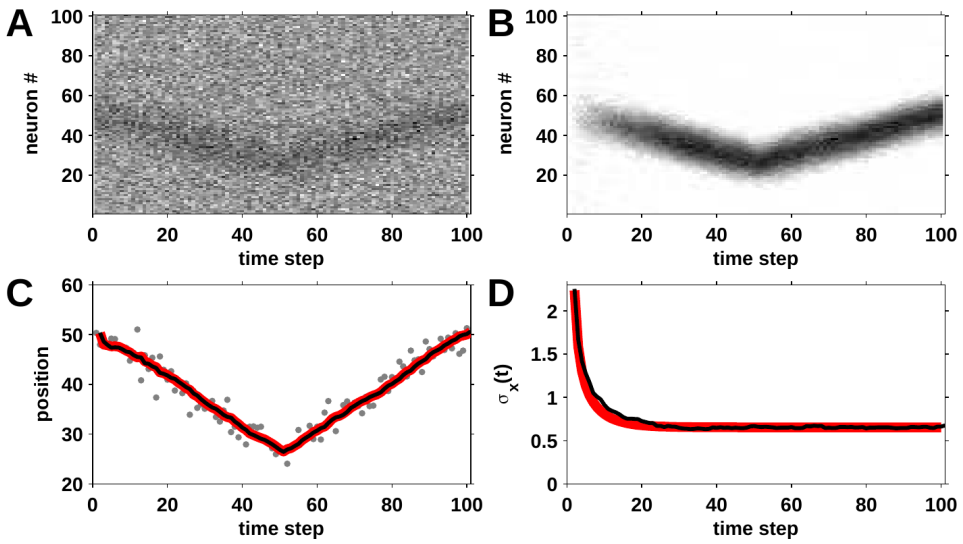


Figure 2.1: (B) The response of eq. 2.18 to a noisy input (A) where the network is aligned with the true velocity ($\gamma(t) = v(t)$). (D) It is notable that the network follows the Kalman optimal variance $\sigma_x(t) \approx \sigma_{kf}(t)$. Figure adapted from [10].

The main contribution of [10] is that the authors, for an ansatz $\mathbf{u}(t) = \alpha(t)\mathbf{U}(\hat{x}(t))$ ($\mathbf{u}$ is a by α scaled version of the fixed point membrane potential $\mathbf{U}$), show that for an optimal w and $\alpha(t)$, the update of $\mathbf{u}(t+1)$ is approximately equal to equation 2.9 of the KF. Figure 2.1 shows the Kalman-optimal evolution of the mean and variance in subplot C and D.

2.2 Probabilistic Neural Representations

Probability theory is based on a set of axioms [18] that formalize and extend straight-forward assumptions, such as that the sum of all probabilities equals to one. Such concepts seem logical and easy to apply for us humans. We

however do not know how these concepts are applied on a neural level. Our discussion on how the brain could perform probabilistic computations is closely related to how we understand the probabilistic structure of the world. We think that the brain must have found a way on the neural level to naturally adapt to statistical properties of the surrounding nature [19]. However, this thought leaves out all assumptions on how the brain remembers and organizes random variables practically, such that it can use them [20].
Decision making is usually learned, therefore, such an organization needs to accumulate past experience in some way, such that, before and after learning, the response to the same stimulus is expected to return the updated uncertainty over a set of possible outcomes [21]. On the neural level, these processes are reflected in the respective firing patterns. Because we can measure the spikes of the respective neurons, the immediate question on the decoding algorithm arises. Different lines of research reveal certain codings that the brain may employ to map from the abstract concept of uncertainty to the firing rates of the underlying neural circuit. For our purpose, we discuss two popular mappings: PPC and Sampling-based Coding (SBC). Figure 2.2 compares the two mappings.
In the PPC framework, neural activity is assumed to encode latent parameters of a probability distribution describing a random variable [17]. The Bayesian framework is used as encoder and decoder in between neural and latent representation. There are many cases where predictions in latent space yield the correct neural representation, however it is still unclear if the brain truly implements a Bayesian decoder.
In SBC, the neural activity directly represents the random variable. No assumptions can be made on the form of the distribution formed by samples of neural activity. The combined activity of the involved neurons at each time step are samples of the joint distribution. Because one needs a certain number of samples to estimate the form of the distribution, time is required to build a reliable estimate of the represented distribution [21]. This is problematic and implausible in cases when fast reactions are required by the situation.
Our line of work does not intend to give a final answer to neural coding, we instead aim to present an alternative view on PPC where the Bayesian decoder and encoder is replaced by a neural mechanism presented in chapter 3. Our goal is to show that fundamental effects such as divisive normalization, filtering and predictive estimation can be achieved within a configuration where two probabilistic neural populations interact with each other. This is fundamentally different from PPC because the latent variables are directly represented in the neural chain, whereas in PPC there is no additional neural structure involved.

2.2.1 Probabilistic Population Codes

In traditional neuroscience, neural activity was found to directly encode values such as orientation sensitivity in the primary visual area (V1) [22]. Contrary to this simple interpretation and inspired by the Bayesian framework, several groups ask how neural activity could collectively encode random variables.
Recent psychophysical experiments show that parameter encoding via Bayes

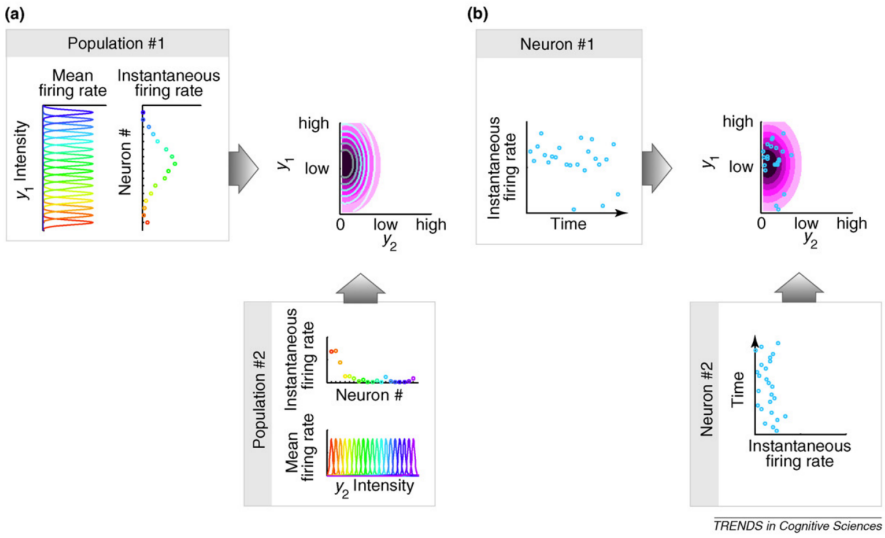


Figure 2.2: Comparison of SBC with PPC. In PPC, a population of neurons is tuned to the same environmental variable with different tuning curves (populations 1 and 2 on the left). In SBC, single neurons, rather than populations correspond to each variable (neuron 1 and 2 on the right). Figure adapted from [21].

theorem is very likely and that humans perform near-optimal Bayesian inference in a wide variety of tasks [17] [23]. For a stimulus function s and the neural population response $\mathbf{r} = \{r_i, \ldots, r_N\}$ the probability $p(s|\mathbf{r})$ is given by the Bayes theorem:

$$p(s|\mathbf{r}) \propto p(\mathbf{r}|s)p(s) \tag{2.21}$$

The broad exponential family of distributions is employed for the neural variability $p(\mathbf{r}|s)$ in PPC. The posterior $p(s|\mathbf{r})$ converges to a Gaussian with increasing number of neurons. We can think of the population as an encoder for the Gaussian's mean and variance. In this framework, the authors show that the framework allows for multisensory integration, linear and non-linear coordinate transforms [17] [1].

2.2.2 Sampling-based Coding

Sampling-based coding dates back to the early experiments on the cat's striate cortex where Hubel and Wiesel were able to map neurons in orientation sensitive columns [22]. Compared to PPC 2.2.1, this approach presents a more direct understanding of the world. Sampling-based coding argues that neural activity patterns are direct samples from a probability distribution [21] [24][25] [26] [27]. In SBC each neuron represents an individual variable x_i from a high-dimensional feature space. Variance of x_i is expressed in the variability of the assigned neurons firing rate. Correlation of two variables x_i, x_j is the equivalent co-variability in firing rates of the two neurons [21]. The direct probabilistic

interpretation of the firing rate has several implications. The spontaneous activity that is often interpreted as stochastic noise may be linked to a generative, top-down process that can be understood as the expectation, attention, or predictive pathway [28].

2.2.3 Neural Macrocolumns

A biolgically plausible algorithm that integrates sensor information must be quick. Humans, for example move their eyes three to four times per second, limiting the effective temporal averaging to a maximum of 250 –350 ms [29]. The cortical macrocolumn is a hierarchical neural model that is based on a homogeneous population of McCulloch-Pitts neurons (with activation function $\mathcal{H}$) and is supported by neuroanatomical and neurophysiological facts [3]. The model is designed to achieve fast reaction time through hierarchical neural collaboration, which is crucial for decision-making and control problems. The model overcomes issues that are linked to SBC from section 2.2.2 such as the time dependence on sample accumulation which is generally slow for neurons and is on the level of macrocolumns linked to PPC representation.
The model is structured hierarchically. Each macrocolumn consists of k minicolumns on the lowest level. Each minicolumn encodes e.g. a visual frequency or orientation. For minicolumns, locally defined update rules lead to well defined global behavior based on a collective firing rate. The following set of difference equations describes the i'th neuron of a minicolumn ($i = 1, \ldots, m$) that belongs to macrocolumn α:

$$\mathcal{H}(x) := \begin{cases} 0, & \text{if } x \leq 0 \\ 1, & \text{if } x > 0 \end{cases} \tag{2.22}$$

$$n_i^\alpha(t+1) = \mathcal{H}\left(\sum_{j=1}^{m} T_{ij} n_j(t) + \sum_{j=1}^{N} R_{ij}^\alpha n_j^E(t) - \Theta(t)\right) \underbrace{\mathcal{H}(1 - n_i^\alpha(t))}_{\text{refraction}} \tag{2.23}$$

$$\Theta(t) = \nu \max_{\alpha=1,\ldots,k} \{p_\alpha(t)\} + \Theta_0 + \Theta_{\text{no}} \tag{2.24}$$

where Θ is an activation threshold and T_{ij} is the minicolumn-internal connectivity. The excitatory neurons of each macrocolumn receive input from an external layer with equal neuron type (figure 2.3). R_{ij} are the afferents to these external neurons n_j^E. The ratio of external (R_{ij}^α) to internal neural connectivity (T_{ij}^α) is assumed to be significantly smaller than one:

$$\frac{\sum_j R_{ij}^\alpha}{\sum_j T_{ij}^\alpha} \ll 1 \tag{2.25}$$

That means the internal input is T usually significantly larger than the external input. Contrary to intuition, due to an internal symmetry breaking process sensitivity to external signals is very high, even for weak coupling. The neighborhood relationship of macrocolumn input neurons is not specified and can be arranged in arbitrary manner. The authors worked with two-dimensional fields, but in this line of work, we will experiment on one-dimensional chains.

Trough statistical considerations (chapter 2 of [30]), km minicolumn difference equations can be replaced by expressions for k macrocolumns.
The authors show that activity-dependent Hebbian plasticity of the input connections $\Delta R_{ij}^{\alpha}(t)$ induces a self-organizing process that allows the individual minicolumns to shape their receptive fields to patterns such that they cover the subspace of extracted features equidistantly. In our line of work, we assume that this process is complete at the beginning of an experiment, which means the optimal organization is given to the agent.
According to [3], the discrete time update rule (2.23) may be replaced by a set of continuous-time differential equations for the activity of a minicolumn α which is now the real valued scalar p_{α}. Neural macrocolumns are on average assumed to be only influenced by their own minicolumns, equal in size and influence.

$$\frac{\mathrm{d}}{\mathrm{d}t}p_{\alpha} = F_{\alpha}(\mathbf{p}) = f\left(p_{\alpha}, h(\mathbf{p})\right), \forall \alpha \in \{1, \ldots, k\} \tag{2.26}$$

where one needs to find the task-specific functions $f : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$ that represent the dynamics and the function $h : \mathbb{R}^k \to \mathbb{R}$ that represents the inhibitory input to a minicolumn. The inhibitory function h should in this context fulfill:

$$\forall y \in \mathbb{R} : f(0, y) = 0. \tag{2.27}$$

One has to select a task-specific function for f that does not violate constraint 2.27. The author derives task-specific dynamics for a two-dimensional pattern memorization task with binary input (black-white). He states that we can choose any function that gives the desired $0 - 1$ bifurcation behavior. Inspired by that statement, we think of an F^{α} such that the system evolves according to the Evolutionary Differential Equation (EDE). In [3], the author demonstrates the derivation of the neural minicolumn activity for his task with a specific choice of f that is a polynominal. Without details, it is in line with our work that the author selects:

$$\frac{\mathrm{d}}{\mathrm{d}t}p_{\alpha} = a p_{\alpha} \left(p_{\alpha} - \underbrace{\nu \max_{\beta=1,\ldots,k} \{p_{\beta}\}}_{h(\mathbf{p})} - p_{\alpha}^2 \right) + \kappa E(R_{ij}^{\alpha}) \tag{2.28}$$

where $a, b > 0$, and $\nu \in (0, 1)$. $E(R_{ij}^{\alpha})$ is the afferent input. p_{α}^2 limits growth and h is an aggregation function over the entire $\mathbf{p}$ vector.

2.3 Divisive Normalization

Divisive normalization describes the simple computation of normalization by division through the total activity $Z = \sum_j x_j$ on a known event space.

$$p(x_i) = \frac{x_i}{Z} = \frac{x_i}{\Sigma_j x_j} \tag{2.29}$$

Even though this computation is very simple on paper, it is practically difficult to compute because of two reasons. The event space needs to be discovered

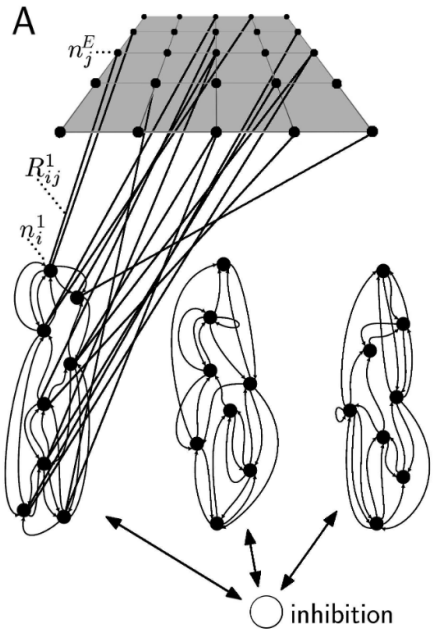


Figure 2.3: $k = 3$ neural minicolums represent a macrocolumn that is connected to $N = 25$ external input neurons [3]. For each minicolumn, we define the afferent connections R and the random internal connections T. Figure adapted from [30].

completely and the event space is usually very large. Therefore the sum is computationally intractable in many practical cases. It is likely that the brain adapts strategies to simplify and hence approximate the summation in Z.

2.4 Entropy

A variety of methods exist to compare the distance between probability measures. Here, we are interested in scalar metrics between probability measures (*simple metrics*). In a broader context, metrics between random variables also exist (*compound metrics*). In applications where one system mimics another, a metric's scalar number is often used to answer qualitative questions about the approximation [31]. Difficulty arises when the two measurable spaces are fundamentally different. In our case, we search for a comparison between a smooth probability density function of the KF that is Gaussian and the discrete, nonparametric density of the NKF. A simple solution for that comparison called the "direct" approach is introduced in [32]. One discretizes both spaces into a finite number of points and then applies measures for discrete distributions. Such a measure is the discrete entropy, that can be used to characterize the behavior of neural systems [33].

$$H(X) = -\sum_{i=1}^{n} \mathrm{P}(x_i) \log_b \mathrm{P}(x_i) \tag{2.30}$$

The concept of entropy as a measure of disorder was introduced in classical thermodynamics but the idea quickly found applications in a wide range of scientific domains. Psychologists and neurophysiologists began to apply the concepts of information theory almost immediately after their introduction. A measure with great significance is the relative entropy (Kullback-Leibler (KL) divergence) which is a distribution-wise asymmetric operation. The KL divergence computes as:

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{2.31}$$

The KL divergence measures the expected number of extra bits required to code samples from $p(x)$ when using a code based on $q(x)$, rather than using a code based on $p(x)$. This is exactly 0 if $p(x) = q(x)$. Strictly speaking, the KL divergence is not a metric, because it is not symmetric and does not satisfy the triangle equality. It however measures the "distance" between the two distributions. We use D_{KL} to compare different distributions with each other throughout simulations.

2.5 Marginalization in neural Circuits

Marginalization is defined as the probalistic computation:

$$p\left(\mathbf{s}'\right) = \int p\left(\mathbf{s}', \mathbf{s}\right) \mathrm{d}\mathbf{s} = \int p\left(\mathbf{s}' | \mathbf{s}\right) p\left(\mathbf{s}\right) \mathrm{d}\mathbf{s} \tag{2.32}$$

where the marginal distribution $p\left(\mathbf{s}'\right)$ is the resulting probability distribution that is defined over a subset of variables from the joint distribution $p\left(\mathbf{s}', \mathbf{s}\right)$. The goal is to find the distribution over this subset without referencing the variables that are 'marginalized out'. In the discrete case, $p\left(\mathbf{s}'\right)$ is found by summing up the probabilities of the joint distribution. As we see in chapter 2.1, the interpretation of the Kalman filter as Markov model states that previous sensor information is marginalized out and compressed to the posterior distribution. Marginalizing effects were found in many neural computations related to Kalman filter-like prediction. For example in models for motor control [34], visual object tracking [35] and function approximation, navigation and visual search [36]. The variety of marginalizing effects that appear throughout the brain suggest that the brain employs general concepts that have the marginalizing effects independent of the application.
In our forward model 3.2 and velocity estimation 3.2 we use convolution $*$ and by reversal of inputs, cross-correlation to compute sums $Y = X_a + X_b$ (conv) and differences $Y = X_a - X_b$ (cc) of discrete random variables:

$$\text{conv}(x[n], h[n]) = x[n] * h[n] = \sum_{k=-\infty}^{\infty} h[k]x[n-k] \tag{2.33}$$

$$\text{cc}(x[n], h[n]) = \sum_{k=-\infty}^{\infty} h[k]x[n+k] \tag{2.34}$$

[37](pp 67) states a formula for a change in variable in a joint probability density for random vectors $\mathbf{X}$ and $\mathbf{Y}$ and their Jacobian determinant$|\mathbf{J}|$.

$$\begin{aligned} f_{\mathbf{Y}}(y_1, y_2, \ldots, y_m) = f_{\mathbf{X}}(h_1(x_1, x_2, \ldots, x_m), h_2(x_1, x_2, \ldots, x_m), \\ \ldots, h_m(x_1, x_2, \ldots, x_m)) \, |\mathbf{J}| \end{aligned} \tag{2.35}$$

We apply eq 2.35 to derive the convolution operation from random variable addition. For that, we define the addition function g_i:

$$Y_1 = g_1(X_1, X_2) = X_1 + X_2 \tag{2.36}$$

$$Y_2 = g_2(X_1, X_2) = X_2 \tag{2.37}$$

where g_i is continuously differentiable and $(g_1, g_2, \ldots, g_m)$ has an inverse h_i which is then:

$$X_1 = h_1(Y_1, Y_2) = X_1 - X_2 \tag{2.38}$$

$$X_2 = h_2(Y_1, Y_2) = X_2 \tag{2.39}$$

By marginalization (eq. 2.32), change in variable (eq 2.35), and X_1, X_2 independence assumption holds:

$$f_{Y_1} = \int_{-\infty}^{\infty} f_{\mathbf{Y}}(Y_1, Y_2) \, \mathrm{d}y_2 = \int_{-\infty}^{\infty} f_{\mathbf{X}}(h_1(y_1, y_2), h_2(y_1, y_2)) \, |J| \, \mathrm{d}y_2 \tag{2.40}$$

$$= \int_{-\infty}^{\infty} f_{\mathbf{X}}(y_1 - y_2, y_2) \, |J| \, \mathrm{d}y_2 \tag{2.41}$$

$$= \int_{-\infty}^{\infty} f_{X_1}(y_1 - y_2) f_{X_2}(y_2) \, |J| \, \mathrm{d}y_2 \tag{2.42}$$

$$= \int_{-\infty}^{\infty} f_{X_1}(y_1 - y_2) f_{X_2}(y_2) \, \mathrm{d}y_2 \tag{2.43}$$

$$= f_{X_1} * f_{X_2} \tag{2.44}$$

because one can show that the Jacobian determinant$|\mathbf{J}|$ is exactly 1 [38]. From eq 2.44 follows that a neural circuit that computes additions or subtractions of discrete random variables performs marginalization. This holds for the forward model and velocity model in this thesis.

2.6 The Reichardt Detector

One can interpret one-dimensional motion as taking steps in a two-dimensional spatiotemporal grid. The Reichardt detector (RD) is an opponent system of two symmetrically arranged subunits that is sensitive to a certain stimulus on the spatiotemporal grid [39]. A RD detects correlation-type motion in between two grid nodes (fig. 2.4). This concept of motion is best understood by thinking of billboard displays. Those displays are an arrangement of many individual light sources. Moving text or images on such a billboard are illusive to the brain because they are nothing but a sequence of on-off configurations of the light sources. A movement is perceived if the billboard controller switches in between those configurations fast enough. One estimates velocity by recognizing letters on a different segment of the billboard at a later time, hence the designation as correlation-type motion detector

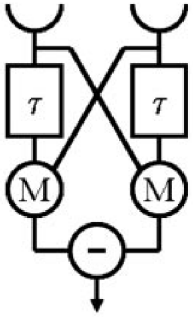


Figure 2.4: The Reichardt detector connects two sites in adjacent images. One signal is delayed (by the τ gate). The undelayed and the delayed signal are multiplied and subtracted from a symetrically arranged unit. It follows that the detector response is positive if the movement is in one direction and negative for the other. Figure adapted from [40].

Due to the aperture problem, RDs are unable to measure the true direction or speed of an object without additional information. [41]. We know that living systems try to solve the aperture problem by not relying on first order motion alone. They include high level features such as contrast modulation, flicker modulation, orientation modulation, direction modulation and depth modulation. Such mechanisms are categorized as second-order motion mechanisms [42]. However, the inclusion of those effects for a more precise motion estimation is out of the scope of this thesis.

2.7 The Hypercycle

Eigen [11] introduces hypercycles as a class of self-replicative chemical reaction networks on the molecular level. A hypercycle describes a reaction graph on self-replicating macromolecules A macromolecule is a large molecule, such as a protein. In the simplest hypercycle, each molecule is linked to itself and its successor for which it catalyses the creation (fig. 2.5 a) Hypercycles are circular

because the last molecule in the chain catalyses the first one. The hypercycle describes a self-replicating process. Macromolecules in hypercycles show important competitive properties for this thesis. If no external forces are applied (read sensory sitmulation), once a macromolecule is ahead of its competition, no competitor is able to take over its dominant position. This is a so called one-for-ever selection. In the limit, one competing macromolecule will dominate the others. They therefore exhibit winner-takes-it-all characteristics. The concept of hypercycles was implemented in different lines of work: As stochastic formulations of Eigen's problem [43], as cellular automata and [44] as partial differental equations [45] . This line of work pursues the last methodology.
Hypercycles are about competitive molecules forming larger replicative units. One can envision different molecular organizations that emerge. The original paper [11] introduces three modes. *Coexistence*, where subunits are mutually present, but due to a lack of interaction do not favour the evolution of functional features. *Compartmentation*, where units are separated in different living spaces and lastly *functionally linked*, when units are cooperating partners via coupling mechanism. Our main interest lies on the functionally linked configuration. We configure links such that net growth regulation leads to a process that is comparable to self-normalization. Figure 2.5 shows different hypercycle configurations.

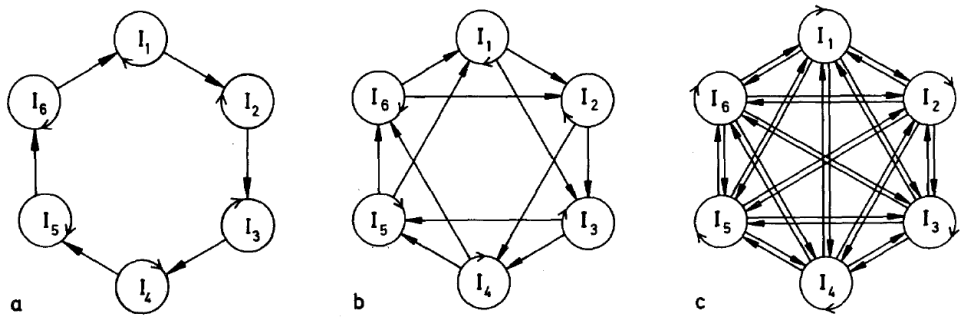


Figure 2.5: Hypercycle configurations of different degrees. a) degree $p = 2$, $n = 6$ b) degree $p = 3$, $n = 6$ c) degree $p = n = 6$ Figure adapted from[11]

Functionally linked hypercycles, expressed as partial differential equations, are characterized by their growth function. The original paper states different growth behaviors for unlimited growth as well as for growth under constraints of constant organization. More formally, the growth is expressed as a dynamic system of n ordinary, first-order and autonomous (time independent) differential equations:

$$\frac{\mathrm{d}x_i}{\mathrm{d}t} = \dot{x}_i = \Lambda_i \left(x_1, \ldots, x_n, k_1, \ldots, k_m; B\right) \tag{2.45}$$

Where x_i is the i'th concentration, k are constant parameters (rate constants, equilibrium constants, etc.) and $B = \mathbf{x}_0$ are the initial conditions. The function Λ_i consists of three terms:

$$\Lambda_i = A_i - \Delta_i - \phi_i \tag{2.46}$$

Those terms are the positive contributions, or amplifications A_i, the negative rates, or decomposition Δ_i and the external constraints applied to the system, ϕ_i. $\Gamma_i = A_i - \Delta_i$ is the net growth function. The author separates solutions to the growth function in three different categories, linear, exponential and hyperbolic growth. In case of $\phi_i = 0$, the system exhibits unlimited growth. Of course, in the limited real world, no population grows unconstrained due to resource availability. It is our main interest to coordinate individual population growth such that the total concentration $c = \sum_i x_i$ can be controlled. Therefore, selecting the appropriate ϕ_i that regulates $\dot{c}$ is key for our objective. In the original paper, a net excess production $\dot{c} = 0$ is referred to as constant organization. Constant organization is of particular importance for our work, because if neurons would express constant organization for the total firing rate of a population, then, the firing rate of the population could be interpreted as a probability measure. Furthermore we want to study convergence to constant organization of imbalanced systems. Because of the stochastic nature of neural input, an iterative normalization process must be robust to strong perturbation. That means, the concentration aka. the neural activity must be normalized again in reasonable time to the controlled c. For that, we study the fixed points $\bar{\mathbf{x}}$ and the attractor dynamics.

A main difference of our work is that the author of [11] states an natural interest in the final outcome of selection, rather than the dynamic process. In this line of work however, we manipulate the selective process of hypercycles by giving catalytic help, proportional to a sensory input, such that the competitive units represent an enviroment variable that is observed. Our main interest is therefore the intermediate state of the system, before it ultimately converges to the winner-takes-it-all configuration.

2.8 Tracking in Mammals

In chapter 1, we argued that we are able to observe predictive behavior in mammals. On a high level, it is well observed within humans, monkeys and other animals that they perform probabilistic inference [46]. However, it is complicated to dissect the tightly integrated behavior to its specifics. For example, when thinking of a hunting predator tracking its prey, the hunter needs to select a target from the herd, preferably one that is weak, then approach, hunt and finally catch the prey. The stages of a successful hunt are an interplay of inference on the behavior of the prey, the mechanical release of energy to follow the animal, and the final, precise attack that slays the prey. It is clear that the final success, which is the catch, is the manifestation of successful tracking over the duration of the hunt. The fact that the predator will use predictive control to jump on the prey is also exploited by the prey. For example, rabbits escape the fox on a zig-zag path that is parametrized by the predators speed, relative distance as well as visual-motor delay. It is obvious that the predator needs to maintain a predictive model of the movements of the prey, otherwise, the prey will outmaneuver its attacks and the predator would loose.

Chapter 3

Methods and Algorithms

In chapter 2 we introduced various building blocks for a neural implementation of the KF. In this section, we gradually connect the building blocks to an algorithm that performs KF-like signal filtering and explains its neurophysiological functioning. For illustrative reasons and simplicity, we develop the filter around the task of one-dimensional motion and velocity estimation. At the end of this section, it should be clear how we layout the computational simulation with the individual components, how we configure the synaptic wiring in between the neural units and how we achieve KF-like velocity estimation in a time-discrete simulation.

3.1 Competitive Neural Signal Filtering

In section 2.1 we showed how the KF is a Bayesian filter that operates on continuous Gaussian random variables. The brain, on the other hand, consists of a finite amount of neurons, unaware of explicit concepts of probabilistic computation. We want to make a strict distinction here between man-made probabilistic models that explain neural activity with some precision and the mechanisms that the neural system itself develops over the time period of its growth. In this picture, it is unlikely that neural computation exactly represents random variables of a certain distributional family.

3.1.1 Structure-giving Distribution Properties

For a KF's Gaussian random variables, but also for other continuous distributions, the parametrization Θ is sufficient to span the probability density function on the space of events. Additionally, most continuous distributions come with certain properties of smoothness and differentiability that limit the shape or structure of the distribution. We argued that neurons could represent, under a normalizing effect, discrete probability distributions. However, without further specification, due to synaptic connectivity and other factors such as neural variability or noise, such neural distributions could take on arbitrary shape. Function-class-specific prior information that is given by the continuous density functions is lost. For Gaussians, such properties are the function's symmetry,

its moments or the principle of maximum entropy. Exemplarily, without any structure, a likely configuration for a velocity estimation could be that for three close velocity values $a > b > c$ whose neural activity representation translates to $p(a) \approx p(c)$ and $p(a) \gg p(b) \ll p(c)$ which does not represent what we would intuitively expect from velocity estimation but could result from random effects such as the lack of stimulation of neurons in region b.
We therefore search for neuronal mechanisms that are structure-giving to the discrete probability distribution represented by the neurons. In this line of work, we assume that structure-giving effects are a consequence of neural connectivity. Contrary to that, other effects such as neurotransmitters influencing the entire neural populations may exist but are not considered here.

3.1.2 Macrocolumns as time-critical Neural Representation

The structure-giving mechanism that we design acts on a population of neurons. For mammals, section 2.8 explains how velocity estimation for tracking and hunting is a time-critical process. From that follows that the neural population of the estimation process must function in a way that guarantees an expected input-to-estimation latency below a certain threshold. For that reason, SBC is not an optimal representation, as it depends on a sampling process that is limited by the relatively low neural firing rates. In other words, SBC, implies an upper bound on the variance estimation by the low amounts of samples available for the distribution estimation. To decrease the latency, we argue that a more plausible representation could be a population that reacts as an ensemble to a certain stimulus. The relevant signal is then the collective firing rate within the sub-population. This idea was introduced in neural macrocolumns (section 2.2.3). By scaling up the population size, as a trade-off to recording temporally, a more precise estimation can be expressed in a shorter time window.
Our neural algorithm employs a large number of equally sized competing neural populations on the lowest level. Equivalent to a hypercycle's macromolecules, those minicolumns are the smallest units that we consider (sec. 2.7). We implement a homogeneous set of competitive update rules for the collective firing rates of the minicolumns. Individual minicolumns can, for example, represent a visual frequency or orientation. The input to the minicolumns is the intensity measured by the sensor of the corresponding region. Globally, the neural macrocolumn formed by the minicolumns evolves such that the competing minicolumns that are stimulated most by the sensor, gradually dominate the unstimulated minicolumns over time. If interpreted as hypercycle, the input corresponds to the catalyst of this process.

3.1.3 The Evolutionary Differential Equation

The evolutionary differential equation (EDE) was first introduced in the field of chemical reaction systems. Section 2.7 on hypercycles introduces the historical background and the relation to chemical processes. In this thesis, we propose to use the evolutionary differential equation as an entropy-decreasing and structure-generating mechanism. The EDE is a macrocolumn-wide homo-

geneous update rule for the evolution of a minicolumn's activity a_x that includes its competitors a'_x.

$$\dot{a}_x = h(a_x, f_x) - a_x \sum_{x'} h(a_{x'}, f_{x'}) \tag{3.1}$$

a_x represents the current average activity over neurons of a minicolumn with index x and f_x is the corresponding input to that minicolumn. h can be a complex function that combines input with the current activity. The most simple choice for h is $h := a_x f_x$:

$$\dot{a}_x = a_x f_x - a_x \underbrace{\sum_{x'} a_{x'} f_{x'}}_{\bar{f}} \tag{3.2}$$

$$= \underbrace{a_x f_x}_{\Gamma} - \underbrace{a_x \bar{f}}_{\phi} \tag{3.3}$$

$$= a_x (f_x - \bar{f}) \tag{3.4}$$

The neural activity cannot grow unconstrained in biological systems. In Eigen's terms (sec. 2.7), $a_x \bar{f}$ corresponds to the growth constraint ϕ that limits the total activity and the net growth function Γ is h, or $f_x a_x$. One can comprehensively interpret eq. 3.2 once the sum of activities is normalized $\sum_i a_i \approx 1$. In that case, $\bar{f}$ is the weighted mean (by a_x) over all inputs f_x. Eq. 3.4 points out that the activity a_x of a minicolumn grows exactly when the current input f_x is above the weighted mean $\bar{f}$.

Self-Normalization

For a probabilistic interpretation of a neural population, the first task is to show that the growth of a_x is limited to ensure that absolute firing rates of a neural minicolumn are comparable in relation to each other after the input is mixed in during the update. This relates to the KF, where we estimate the probability of the next state (x_{t+1}) from $p(x_{t+1}|x_t, y)$ (that depends on input and previous state) in the prediction step. If the macrocolumn's neural activity self-normalizes, then we are able to directly interpret the activity as representative probability from the macrocolumns of the NKF. For that, it is necessary to show that according to the evolutionary differential equation the overall macrocolumn activity converges to a normalized state:

$$\dot{a}_x = a_x \left(f_x - \sum_{x'} a_{x'} f_{x'} \right) \tag{3.5}$$

$$\frac{\mathrm{d}}{\mathrm{d}t} \sum_x a_x(t) = \sum_x \dot{a}_x = \sum_x a_x \left(f_x - \sum_{x'} a_{x'} f_{x'} \right) \tag{3.6}$$

$$= \sum_x a_x f_x - \sum_x a_x \sum_{x'} a_{x'} f_{x'} \tag{3.7}$$

$$= \sum_x a_x f_x \left(1 - \sum_x a_x \right) \tag{3.8}$$

From the last result, it is easy to see that the rate of change of $\sum_x a_x$ is zero in two cases: either the input f_x is zero or the $\sum_x a_x$ is exactly one. One can check that this normalizing effect applied to a family of differential equations of the form

$$\dot{a}_x = h(a_x, f_x) - a_x \sum_{x'} h(a_{x'}, f_{x'}) \tag{3.9}$$

Where the equivalent derivative of the whole activity with respect to time is:

$$\sum_x \dot{a}_x = \sum_x h(a_x, f_x) - \sum_x a_x \sum_{x'} h(a_{x'}, f_{x'}) = \sum_x h(a_x, f_x) \left(1 - \sum_x a_x \right) \tag{3.10}$$

Therefore, the system is self-normalizing for any choice of $h(a_x, f_x)$. We use this finding to test different versions of $h(f_x, a_x)$ to best fit the NKF to its technical role model, the KF.

Exponential Growth

In our proposed method, minicolumn activity is governed by the EDE. Here we study the evolution of minicolumn activity governed by equation 3.2 analytically. For that we examine the EDE with $h(a_x, f_x) = a_x f_x$:

$$\dot{a}_x = a_x f_x - a_x \sum_{x'} a_{x'} f_{x'}$$

The first summand on the right side of the equation grows exponentially because it is proportional to the quantity a itself. Previously, we showed that the EDE is self-normalizing. We combine these two hints to guess an analytical solution. We express the activity $a_x(t)$ as an exponential function $a_x(0)exp(f_x t)$ with the associated normalization constant $Z = \sum_{x'} a_{x'}(0)exp(f_{x'} t)$:

$$a_x(t) = \frac{a_x(0)exp(f_x t)}{Z} \tag{3.11}$$

We compute its derivative $\frac{\mathrm{d}a_x(t)}{\mathrm{d}t}$, to show that $a_x(t)$ is a special solution to eq. 3.2. By applying the quotient rule and reordering of terms, we arrive at:

$$\frac{\mathrm{d}a_x(t)}{\mathrm{d}t} = \underbrace{\frac{a_x(0)exp(f_x t)}{Z}}_{a_x(t)} \frac{[f_x Z - \sum_{x'} a_{x'}(0)exp(f_{x'}t)f_{x'}]}{Z} \tag{3.12}$$

We substitute the full expression (eq. 3.11) for $a_x(t)$ twice and arrive again at the EDE, which concludes the proof that eq. 3.11 is a special solution of eq. 3.2 :

$$\dot{a}_x = \frac{\mathrm{d}a_x(t)}{\mathrm{d}t} = a_x(t) \left[f_x - \sum_{x'} f_{x'} \underbrace{\frac{a_{x'}(0)exp(f_{x'}t)}{Z}}_{a_{x'}(t)} \right] \tag{3.13}$$

$$= a_x(t) \left[f_x - \sum_{x'} f_{x'} a_{x'}(t) \right] \tag{3.14}$$

$$= a_x f_x - a_x \sum_{x'} a_{x'} f_{x'} \tag{3.15}$$

The special solution is only applicable to inputs a_x that are normalized already. The explicit solutionnormalizes the activity in one step for any configuration of a_x where $\sum_x a_x \neq 1$. Eq. 3.11 explicitly shows divisive normalization (Z) and exponential growth of the activity depending on the input f_x.

Mapping on to Probability Simplex

Normalization effectively reduces the system's degree of freedom by attracting the neural activity towards the $N-1$ dimensional probability simplex. This introduces a linear dependency between the elements of $a_{x_i} \in \mathbf{a}$. For linear growth in Γ, the only fix-point is described by the one-hot vector which is the corner of the strongest component a_{x_i}, $\bar{\mathbf{x}} \in \mathbb{1}^N(i)$. From that result follows that for the same input f_x one always ends up in the same configuration and therefore the system is robust with respect to initial conditions $a_x(0)$ and step size [11].
An empirical test shows that the EDE converges towards the same fixpoint for every initial configuration in two dimensions (fig 3.2), given the same input.

3.2 Neural Velocity Estimation

The EDE as sole mechanism is not able to perform KF like velocity estimation, but serves as a denoising and entropy decreasing mechanism. This section explains how the EDE interplays with a Reichhart detector and a linear motion model to achieve one-dimensional neural velocity estimation.

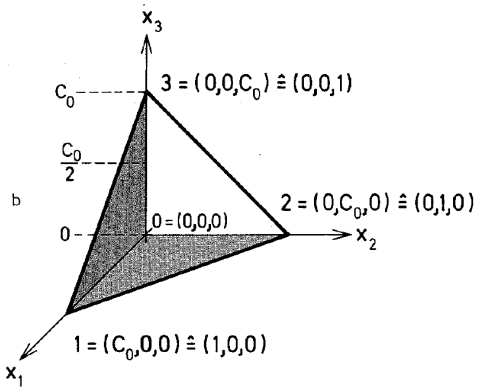


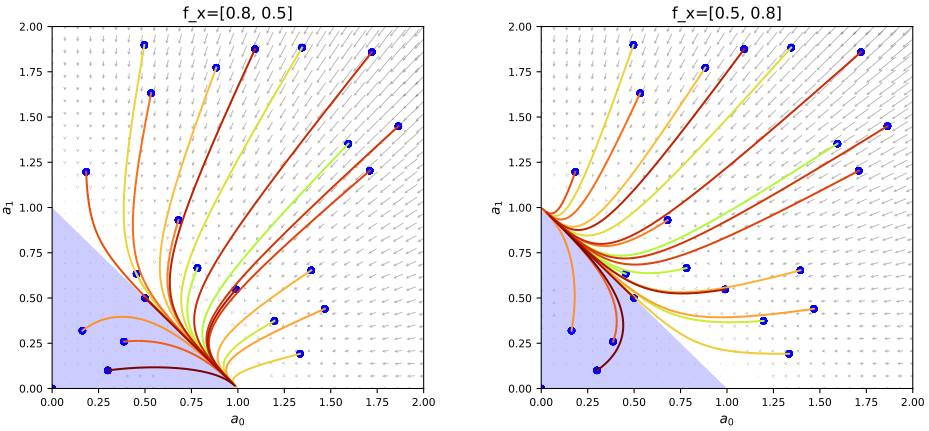
Figure 3.1: The probability simplex is a $N-1$ dimensional plane embedded in the N dimensional neural activity space. Figure adapted from [11].

Neural Representation

Our algorithm consists of two macrocolumns with N minicolumns that are topologically arranged in chains. The macrocolumns $X = [x_0, \dots x_N]$ and $V = [v_0, \dots, v_N]$ represent the finite sets of locations $\mathcal{X}$ and velocities $\mathcal{V}$ that our algorithm covers. The macrocolumns X and V are a discrete representation of the velocity and position that a tracked object could have. Each minicolumn $x_i \in \mathcal{X}$ or $v_i \in \mathcal{V}$ belongs to one neural chain exclusively. The intervals are evenly spaced over N minicolumns such that: $v_i - v_{i+1} = v_j - v_{j+1} \forall i, j$. Each minicolumn has an assigned activity $a_{\{x,v\}}(t) \in \mathbb{R}_{>0}$ that represents the subpopulation's average spike rate at time point t. It is any minicolumn's change of activity $\dot{a} = \Lambda(f, a)$ as a function of input $f \in \mathbb{R}^N_{>0}$, which is governed by the EDE (eq. 3.2). The one-dimensional motion estimation has a memory requirement of $3N$ for the tuple (X^{t-1}, X^t, V). We store the state of the two chains together with the coincidence fibers that effectively transmit the state X^{t-1}. In that configuration, we arrange the macrocolumn in a way that resembles PPC. The PPC represents the two random variables: velocity and position. Each minicolumns's momentaneous firing rate represents a probability measure assigned to a certain position x_i or velocity v_i. Therefore, for minicolumn x_i at location i, $p(x^\star = x_i) \propto a_{x_i}$

Input

The goal is to estimate the true position $x^\star$ and velocity $v^\star$ of a moving object. The tracked object's position is always on an interval $x^\star \in (x_-, x_+)$ and has a velocity in range $v^\star \in (v_-, v_+)$. The algorithm receives information about the tracked object from recordings of a noisy sensor $f_x \in \mathbb{R}^N_{>0}$ that records on N sites. The sensor abstracts away tasks from computer vision such as pattern matching and object segmentation. Its signal can be interpreted as intensity or activity due to pattern stimulation at a specific site.
We pass the raw sensor signal f_x directly to the NKF. This input f_x modulates the activity a_x of the neural chain X that is the first layer. The neural chain



(a) $f_x = [0.8, 0.5]$ (b) $f_x = [0.5, 0.8]$

Figure 3.2: The probability simplex is a line for two-dimensional neural activity spaces. We show the evolution for several initial configurations. The background shows a vector field that corresponds to $\dot{a}(x_1, x_2)$. The location of the fixpoint depends on the input f_x. We show two configurations. In a), the input is $f_x = [0.8, 0.5]$, hence the attraction towards the position $(1, 0)$. In b), we reverse the input to $f_x = [0.5, 0.8]$ which mirrors the fixpoint to $(0, 1)$. In both cases, the system arrives at a winner-takes-it-all configuration.

covers the sensor's range entirely and neurons are placed equidistantly along the sensor axis. For an activity detector in between (a, b) that means the leftmost minicolumn is assigned to a and the rightmost to b. Because sensor input is blurry, the total stimulus F may be distributed over multiple abstract neurons. We directly project the input activity onto the neural chain. That means our system is compatible with pixel-based sensors such as RGB(D) cameras, pixel-based laser sensor or neuromorphic image sensors. At each time step t, we capture a new frame of sensor data as the input to the algorithm. This method is a first-order motion detector.

Memory

The memory of the system is the tracked object's estimated position and velocity represented as a discrete random variable on one-dimensional macrocolumns. The algorithm remembers the positional distribution on the first macrocolumn X and the velocity distribution on the second V across epochs. Analogous to the KF, we assume that our algorithm is a Markovian process. That means for all minicolumn activities $a_{\{x,v\}}$ the current activity depends solely on the previous epoch's activity.

$$a^t | a^{t-1}, a^{t-2} \ldots, a^0 = a^t | a^{t-1} \tag{3.16}$$

Secondly, a short term memory is realized in the coincidence fibers of Reichardt

detectors that connect sites pairwise on the X macrocolumn. This delayed signal is used for the velocity estimation based on the updated a_x activity.

Update of a_x and a_v

An update cycle (*epoch*) is separated into two steps: a generative forward model g_x for the predictive step and a backward process that refines macrocolumn activities. In the forward model, we use the current macrocolumn activity (memory) $(\mathbf{a}_x^t, \mathbf{a}_v^t)$ to predict the next input, whereas the backward model formulates update rules for the activity. These update rules regulate the fusion of input and current prediction, as well as the entropy decrease in the activity distributions a_x and a_v.
Mixing occurs naturally within the neural filter when the EDE governs the update of the neural activity. Evolution of a neural distribution according tho EDE happens in $EDE(h)$ filter blocks. ($EDE(h)$ evolves the activity $a_{\{x,v\}}$ for a time step Δt using the EDE). The mixing outcome depends on two factors, the input-activity function f and the evolution strength which follows from the number of iterations for the numerical solver of the EDE filter S_x, S_v or the time-constants β_x and β_v and the step size Δt

The forward model g_x

The algorithm uses the previous activity a_x^t and a_v^t on the macrocolumns to predict the activity of the next time step $\tilde{a}_x^{t+1}$. With this prediction, the algorithm tries to be least surprised about the environment in the next time step once a new sensory input arrives. For the forward model, a neural implementation of equation 4.4 is necessary to update our estimation in the position of the tracked object.
We assume that the brain has this general understanding of motion and the neural circuit is already laid out. We restrict ourselves to estimating constant translational velocity and position in a straight line. From that follows that the prediction of a new position is equivalent to a shift of the X activity distribution A_x [1]. We compute the linear shift as addition of the two random variables $A_x^{t+1} = A_x^t + A_v^t$.
For Gaussian distributions, we show in section 4.2 that this operation is equal to the addition of mean and variances. For arbitrary discrete activation distributions, activity is defined as convolution. The activity of the generative model g_x therefore computes as follows:

$$g_x(z) \leftarrow \tilde{a}_x^{t+1}(z) = \sum_{k=-\infty}^{\infty} a_x^t(k) a_v^t(z-k) \tag{3.17}$$

where we denote the activity of the i'th x or v minicolumn as $a_{\{x,v\}}(i)$. We show a neural implementation of eq. 3.17 in figure 3.3.

Velocity estimation

For each minicolumn $x \in X$ we employ a set of N excitatory fibers $e(x, x')$, one for each velocity $v \in \mathcal{V}$. They propagate the initial activity of the minicolumn and can be seen as short-term memory. We expect the signal to take a certain amount of time ($S_x * \Delta t$) to arrive at the target population of the same macrocolumn. Within that time, the X macrocolumn is updated by EDE evolution. In this delay-based coincidence detector, simultaneous activity in the target x' and source neuron x (shifted in time) affects a coincidence-sensitive synapse. This set of coincidence sensitive synapses is wired as the input to the V macrocolumn (f_v). We show an implementation of this system in figure 3.4 where the signal to the V macrocolumn is transported on bidirectional neurites. We envision bidirectional connectivity, because the exact same connection pairs are required for the generative model g_x. Coincidence detection is a heavily studied subject in auditory systems. Mathematically, it is an operation akin to cross correlation [47]. For discrete random variables, the cross correlation computes the subtraction $A_v^{t+1} = A_x^t - A_x^g$. In our algorithm, we compute the cross correlation (CC) with respect to the previous time step $CC(a_x^t, a_x^g)$ to estimate the displacement Δx (using the same notation as in eq. 3.17):

$$f_v(\Delta x) \leftarrow \sum_{k=-\infty}^{\infty} a_x^g(k) a_x^t(k + \Delta x) \tag{3.18}$$

We then continue by transforming the displacement d_x to the discrete space of velocities $\mathcal{V}$. This is possible because we know the time constant Δt ahead of evaluation. We stimulate the V macrocolumn with a second EDE filter that, through competition, refines the belief about the velocity based on a_v^t and this transformed displacement f_v.

The algorithm

Here, we present the whole algorithm (see algorithm 1). For each time step, we compute the updates of $A_x = [a_x(0), \ldots, a_x(N)]$ and $A_v = [a_v(0), \ldots, a_v(N)]$ sequentially. The iteration numbers S_x and S_v are given as hyperparameters. The order of computation of g_x and f_x does not matter. Due to the convolution (eq. 3.17), the entropy of g_x is bigger than the entropy of a_x^t. This is equivalent to the variance-increasing prediction step of the KF.

3.2.1 Neural Interpretation

In this section, we explain the potential neural wiring of a system that estimates position and velocity using the mechanism of section 3.2 and the EDE 3.2. Contrary to the Bayesian interpretation, we use dedicated neural populations (macrocolumns) that directly represent (latent) random variables. We refer to [3] from section 2.2.3 for detailed instructions on the conversion of the i'th neurons activity in cortical minicolumns $n_i(t)$ to the collective activity p which is a positive real scalar that we consider here. In figure 3.5, we sketch a possible neural circuit for velocity estimation. The convolution operations for

Algorithm 1: One-Dimensional Velocity Estimation

Data: $a_v^t, a_x^t, N, T, S_x, S_v, X^\star, V^\star$
Result: a_x^{t+1}, a_v^{t+1}
$t \leftarrow 0$;
for $t \leq T$ **do**
  $g_x(z) \leftarrow \tilde{a}_x^{t+1}(z) = \sum_{k=-\infty}^{\infty} a_x^t(k) a_v^t(z-k)$
  $o_x \leftarrow \text{world}(x_t^\star, v_t^\star, \sigma_s)$
  $f_x = \mathcal{N}(o_x, \sigma_o)$
  for $s \leq S_x$ **do**
    $a_x^g \leftarrow \text{EVO}(h_x(g_x, f_x))$
  end
  $f_v(\Delta x) \leftarrow \sum_{k=-\infty}^{\infty} a_x^g(k) a_x^t(k+\Delta x)$
  for $s \leq S_v$ **do**
    $a_v \leftarrow \text{EVO}(h_v(a_v, f_v))$
  end
  $a_v^{t+1} \leftarrow a_v$, $a_x^{t+1} \leftarrow a_x^g, t \leftarrow t+1$
end

g_x and f_v use the same connections (green) at different times. This double function suggests to invoke bidirectional connections. Whereas axons and dendrites conduct signals in only one direction, there are examples of neural processes in the peripheral nervous system that conduct signals in both directions. These are called "neurites". Not much is known about neurites in the central nervous system, therefore it could be equally plausible that two unidirectional fibers perform the functions of a single neurite.

We want to highlight two non-standard computations of our system: the first non-standard computation is that the input f_x is multiplied with the current activity a_x. In that configuration, we interpret the input as a modulation of the synaptic conductivity of the cell and not an input transmitted via an axon with a given synaptic weight. While multiplicative interactions are not the norm, it remains a frequently observed pattern in V1 [48], [49], V3 [50], MT [51], MST [51], [52], LIP[53], [54], VIP[55], V6a[56], premotor cortex[57], area 5[58], the primary auditory cortex [59] and the inferior colliculus [60] (list adapted from [1]).

The second problem with equation 3.2 is that in reality only signals a_x are transmitted by axons, not the products $f_x a_x$. A neurally acceptable implementation would be combining an excitatory part (Γ) of the form $\dot{a}_x = f_x a_x$ with an inhibitory system that maintains the normalization $\sum_x a_x = 1$. Such a system could be implemented by lateral inhibition over the whole string of neurons. If one solves this simple ODE with Euler iteration, with an additional

normalization step after each Euler iteration, the result one gets is:

$$a'_x = a_x(t + \delta t) \tag{3.19}$$

$$= \frac{a_x + \delta t f_x a_x}{1 + \delta t \sum_{x'} f_{x'} a_{x'}} \tag{3.20}$$

$$\approx (a_x + \delta t f_x a_x)(1 - \delta t \sum_{x'} f_{x'} a_{x'}) \tag{3.21}$$

$$= a_x + \delta t (f_x a_x - a_x \sum_{x'} f_{x'} a_{x'}). \tag{3.22}$$

This is exactly the Euler increment of our EDE. (The approximation corresponds to neglecting the $(\delta t)^2$ term.)

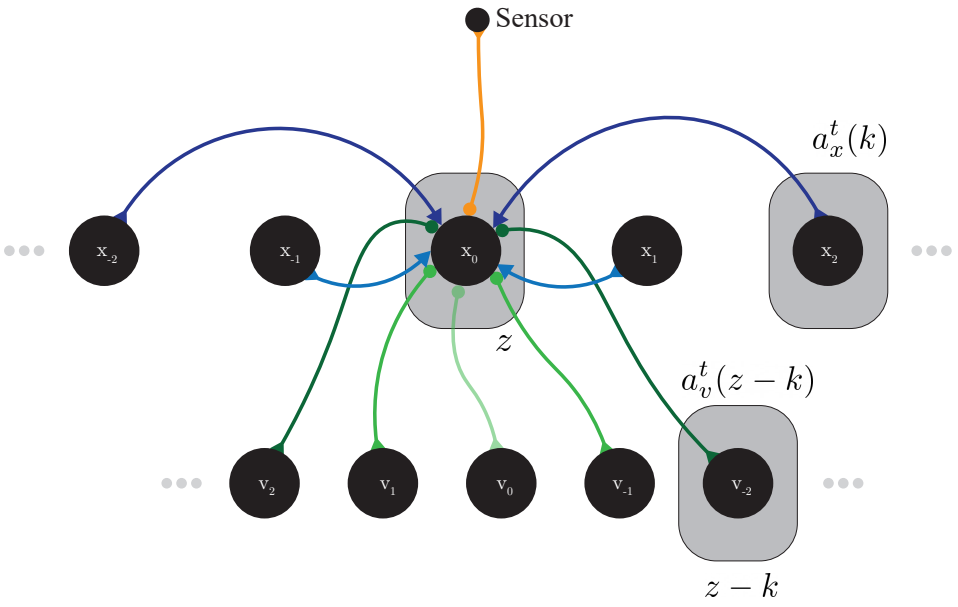


Figure 3.3: A neural implementation of the forward model g_x of algorithm 1. (black) The graphic shows a section on the macrocolumns. The section consists of 5 minicolums of the X and 5 of the V macrocolumn. (blue shades) coincidence fibers for $v \in -2, -1, 0, 1, 2$ that implement the Reichardt detector. (green) the x and v minicolumns are connected via bidirectional neurites that pair with the coincidence fibers. For each X-minicolumn at position z (we show only one operation), the neurites modulate the activity on all afferent coincidence fibers $a_x^t(k)$ proportional to the current activity for that velocity $a_v^t(z-k)$. The minicolumn (z) then accumulates the modulated signals, implementing the convolution of the forward model. (We mirror the V macrocolumn for a better visualization. Synaptic functions: Arrows $\rightarrow$ excitatory, circles $\rightarrow$ modulatory)

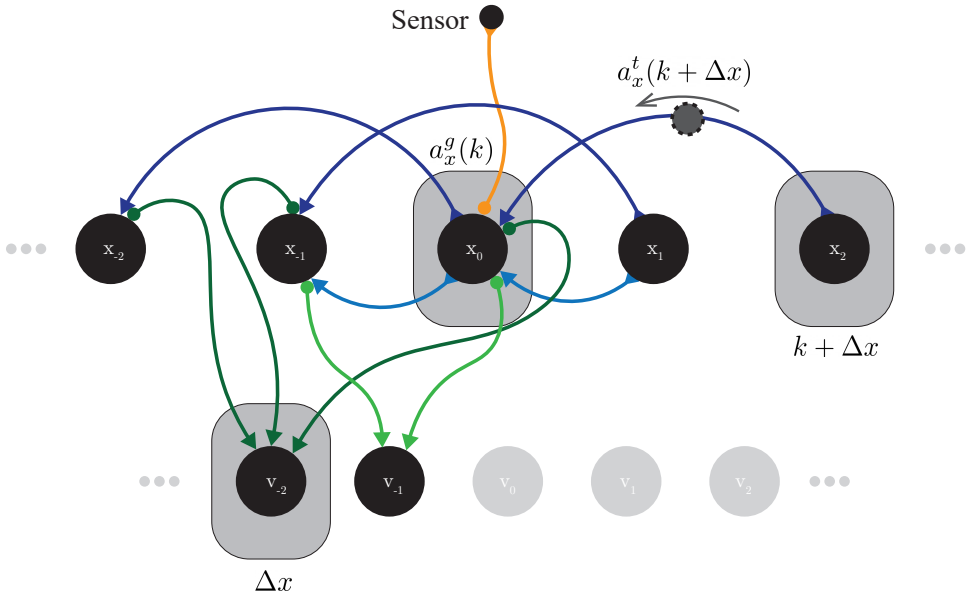


Figure 3.4: A neural implementation of the velocity estimation of algorithm 1. (black) The graphic shows a section on the macrocolumns. The section consists of 5 minicolums of the X and 2 of the V macrocolumn. (blue shades) A selection of coincidence fibers for $v = -1$ (light) and $v = -2$ (dark) that implement the Reichardt detector. (green) The x and v minicolumns are connected via bidirectional neurites that pair with the coincidence fibers. Here the neurites propagate the activity of the respective coincidence fiber from minicolumn $k + \Delta x$ to the V minicolumn. The signal is modulated by the current activity of the X minicolumn $a_x^g(k)$. On the V minicolumn, signals accumulate from all X-minicolumn sites, implementing the cross-correlation of the velocity estimation. (Synaptic functions: Arrows → excitatory, circles → modulatory)

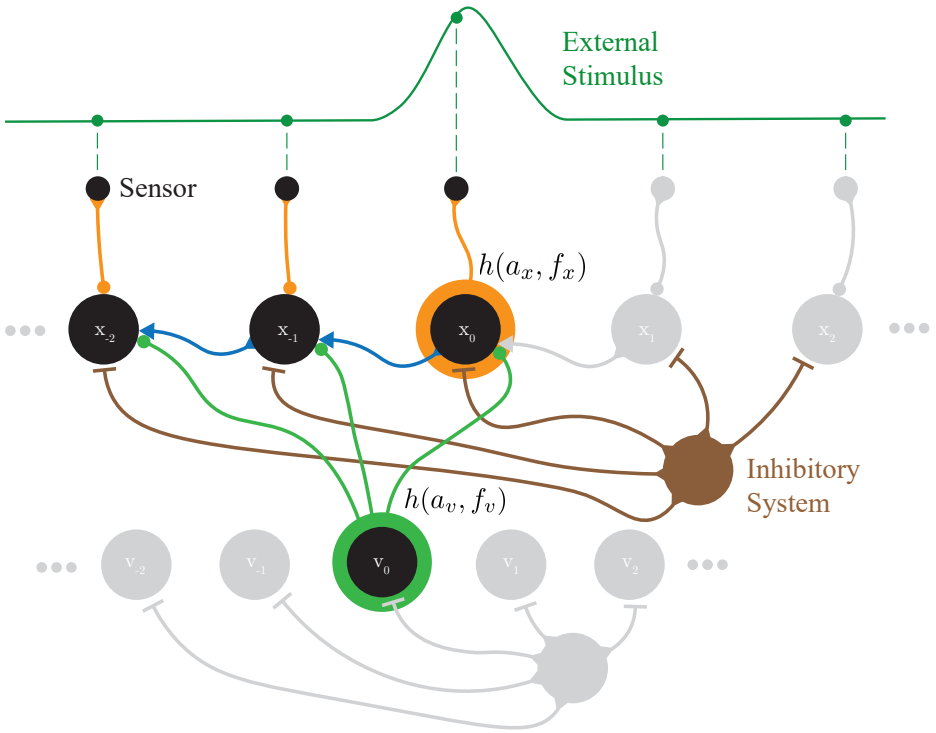


Figure 3.5: A neural implementation of the EVO(h) function of algorithm 1. (black) The graphic shows a section on the macrocolumns. The section consists of 3 minicolums of the X and 1 of the V macrocolumn. (blue shades) A selection of coincidence fibers for $v = -1$ that implement the Reichardt detector. (green) The x and v minicolumns are connected via bidirectional neurites that pair with the coincidence fibers. The neurites propagate the V macrocolumn activity a_v in the forward model g_x and backwards ($X \rightarrow V$), accumulate the delayed signals on the coincidence fibers for one specific velocity $v = v^*$ of all Reichardt detectors sensitive to v^* to form the input f_v. The input f_x originates directly from the sensor that is sensitive to the external stimulus. (large green / orange circles) The competitive effect of the EDE affects each of the minicolumns at all times. The complex function h of the EDE is implemented in the synaptic terminals. The inhibitory system is responsible for divisive normalisation. For readability, we do not show the synapses to the inhibitory system. (Synaptic functions: Arrows $\rightarrow$ excitatory, bars $\rightarrow$ inhibitory, circles $\rightarrow$ modulatory)

Chapter 4

The Simulator

In this chapter, we present the simulator that we use to explore the evolutionary EDE (eq. 3.2) introduced in chapter 3. From the simulator's perspective, NKF internals are non-accessible if it is not for the purpose of visualization. For each simulation epoch, the simulator's task is to generate and present all required input, trigger the simulations and process the outputs. Each simulation consists of three components. Those are the NKF simulation, the simulation of the real world, as well as the comparative KF model. One can think of the simulator as the NKF's interface to the world. The simulator creates a one-dimensional motion schedule ahead of time that is then presented to the agent. Here we explain how the motion schedule is computed. For that, we introduce the associated noise models for the sensor and the world.

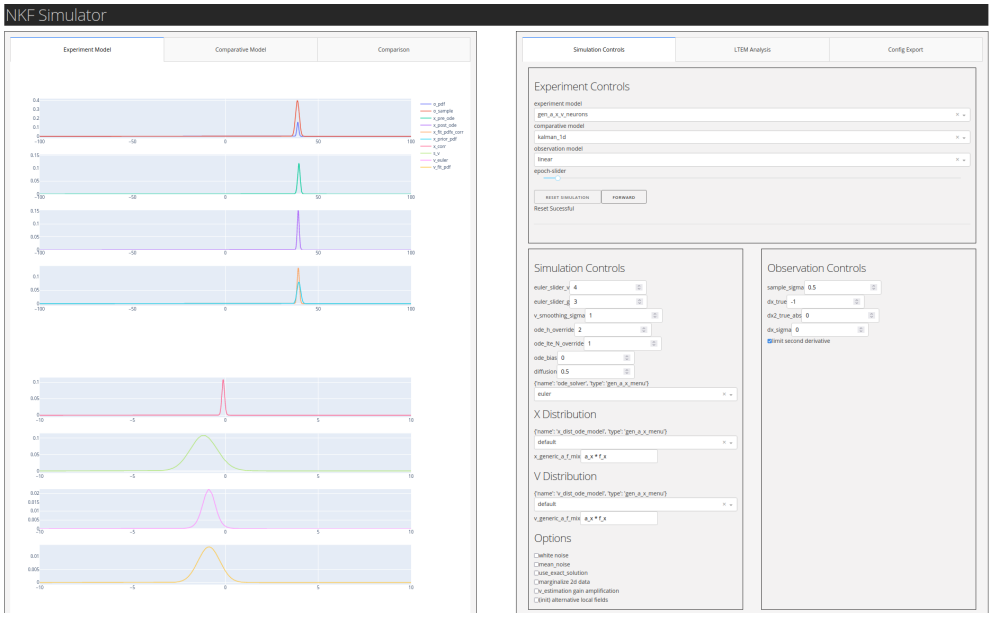


Figure 4.1: The simulator web interface.

4.1 The World Model

In the world model $f_x = \text{world}(x^\star, v^\star, \sigma_s)$, we simulate the environment of this agent-based simulation. That is the true position of the tracked object $x^\star$ for a pre-set velocity profile $v^\star(t)$ and the noisy sensor of which the output f_x is given as input to the estimation process. If $\epsilon = 0$, then the world model is a simple one dimensional linear motion for a given velocity:

$$x(t+1) = v(t)\Delta t + x(t) + \epsilon \tag{4.1}$$

$$\epsilon \sim \mathcal{N}(0, \sigma_w) \tag{4.2}$$

For $\epsilon > 0$, $x(t+1)$ is a Gaussian random walk. For every new $x(t+1)$, we apply the sensor model $f_x = \mathcal{N}(x(t+1), \sigma_s)$ with sensor variance σ_s.

4.2 Applied Kalman Filtering

We employ a multivariate ($d = 2$) KF for one-dimensional velocity estimation. That is necessary because an univariate KF lags behind the true state. The reason for that is because it knows no concept of motion [13] and therefore is unable to predict the shift of x that is caused by the current velocity. The state vector of the multivariate KF consists, equivalent to the NKF, of the position and the velocity. We use the framework **filterpy**[13] and basic kinematic equations as prerequisites:

$$x_k = x_{k-1} + \Delta t v_{k-1} \tag{4.3}$$

$$v_k = v_{k-1} \tag{4.4}$$

Filterpy is a python package that implements the KF. It provides methods to compute the update equations for gain and state that we derived by hand in 2.9. **filterpy** needs to be initialized with all parameters relevant to Kalman filtering ($\mathbf{H}, \mathbf{F}, \mathbf{Q}, \mathbf{R}, \mathbf{P}, \mathbf{x}_0$).
In our case, we set the observation model (measurement function) $\mathbf{H}$ to $\mathbf{H} := [1, 0]$. By that, we state that we are only and directly observing the position. The velocity is indirectly estimated. We use equation 4.4 to define the process model $\mathbf{F}$ as:

$$\mathbf{F} := \begin{bmatrix} 1 & t \\ 0 & 1 \end{bmatrix} \tag{4.5}$$

The sensor information that we provide to the KF is scalar, unlike for the NKF, The reason for that is that the KF assumes the sensor to be a Gaussian random variable where μ_s equals the sample. We set the sensor's scalar measurement uncertaincy $\mathbf{R}$ to 5 and the process covariance $\mathbf{Q}$, which is also scalar in this application, to $\sigma^2_{f_x} = 0.01$. In practical applications, the initialization of $\mathbf{Q}$ and $\mathbf{R}$ requires domain knowledge and iterative fine-tuning for a good approximation because sensors are usually non-linear and non Gaussian [13]. Additionally

we initialize the state vector to $\mathbf{x}_0 = [0, 0]$ and the state covariance matrix $\mathbf{P}$ as:

$$\mathbf{P} := \begin{bmatrix} 500 & 500 \\ 500 & 500 \end{bmatrix} \tag{4.6}$$

Which is approximately equivalent to uniform initialization, which is the strategy that we choose for the NKF.

4.3 Applied Neural Filtering

Analogous to the KF, we describe here the details and extensions needed to apply the theoretical model of the NKF to the practical task of one-dimensional velocity estimation. Additionally, we introduce experimental additions to the neural filter algorithm.

Choice of $h(f_x, a_x)$

In section 3.1.3 we showed that the EDE normalizes the macrocolumn activity for any function $h(a_x, f_x)$. By default, we suggest the simple function $h(a_x, f_x) = a_x f_x$ in the simulator, but we provide a *generic* mode where arbitrary h may be implemented by the operator. In table 4.1, we show an overview on functions that we experimented with as a reference.

Var.	$h(f_x, a_x)$	Description
1	$a_x f_x$	Direct excitation of a_x by the input f_x
2	$h(a_x, f_x) + \alpha D(A_x^N)$	A diffusion process D acts on the neighborhood of a_x, A_x^N.
3	$a_x^y f_x^z$	The exponents y and z control the strength of the a_x / f_x contribution relative to each other.
4	$\beta h(a_x, f_x)$	β is a time constant that controls the overall rate of change. A change of β is equivalent to an increase of the number of iterations S.
5	$h(f_x, a_x) + b$	Biased version of h. The bias b counteracts neural starvation. See 5.1 for details.
6	$\alpha a_x + (1 - \alpha) f_x$	Convex combination of a_x and f_x
7	$\alpha a_x + \beta f_x$	Linear combination of a_x and f_x

Table 4.1: Different variants of $h(f_x, a_x)$ which we implement in the simulator.

Hyperparameters

Similar to the KF's sensor and process model, the NKF comes with hyperparameters that need tuning. The neural filter is sensitive to the number of iterations chosen for the numerical solver. This corresponds to the timing between sensor updates and the progress of the evolutionary process. This is true for both macrocolumns representing location X and velocity V. Hence, the

iteration counts S_x and S_v dictate the decrease of entropy of the distributions that evolve on the macrocolumns. The operator defines the hyperparameters $S_{\{x,v\}}$ before an evaluation. Additionally, the simulator allows adjusting $S_{\{x,v\}}$ mid-simulation for experimentation. A more realistic way to change the timing is by adding a time-constant to h, $h' = \beta h(a_x, f_x)$.

$$\dot{a}_x = \beta h(a_x, f_x) - a_x \sum_{x'} (\beta h(a_{x'}, f_{x'})) \tag{4.7}$$

$$\dot{a}_x = \beta \left(h(a_x, f_x) - a_x \sum_{x'} h(a_{x'}, f_{x'}) \right) \tag{4.8}$$

For Euler iterations, the addition of β is equivalent of changing the step size Δt for $\dot{a_0} \approx \dot{a_1}$:

$$\begin{aligned} a_2 &= a_1 + \Delta t \beta \dot{a_1} && (4.9)\\ &= (a_0 + \Delta t \beta \dot{a_0}) + \Delta t \beta \dot{a_1} && (4.10)\\ &\approx a_0 + 2\Delta t \beta \dot{a_0} && (4.11) \end{aligned}$$

Thus, for a time constant $\beta = 0.5$, we compress two steps into one.

Bias

Dictated by the EDE, the activity for off-stimulus minicolumns decays exponentially. In simulation time, this effect moves off-stimulus minicolumn activity close to zero quickly. We call this effect *starvation*. As a consequence, a very long warm-up period is required to move the starved minicolumns back to a reactive zone. In starved regions, the system is therefore unable to react dynamically to observed changes. A bias, that is a very small fraction of the peak activity, prevents minicolumns from starvation. We therefore introduce the bias b to h or $h' = \beta h(a_x, f_x)$. That way, the bias does not influence $\sum_x a_x \approx 1$ and the activity can still be interpreted as a discrete probability measure:

$$\dot{a}_x = h(a_x, f_x) + b - a_x \sum_{x'} (h(a_{x'}, f_{x'}) + b) \tag{4.12}$$

Diffusion

The KF controls mean and variance of the Gaussian state vector $\mathbf{x}$. We show in experiments that the NKF estimates similar means, but the variance evolves differently. Due to the competitive evolution with winner-takes-it-all characteristics, the weighted variance of the unimodal discrete distributions of the NKF is constantly decreasing. The limit would be the Dirac distribution. A strategy to control the variance is to add a diffusion term.
In thermodynamic systems, heat flows from hotter to adjacent colder sections. The change in temperature is an effect that proportionally depends on the neighbor's temperature. In the limit, temperature equalizes and is uniformly

the same. We want a similar effect to increase the variance of the activity in the NKF if an uninformative stimulus is applied.
A possible implementation of the diffusion process D is an approximation to the second derivative of the activity $\frac{\partial^2 a_x(t)}{\partial x^2}$ along a macrocolumn using the finite difference method:

$$D(A_x^N) = \frac{a_x(t+1) - a_x(t)}{\Delta t} = \frac{a_{x+1}(t) - 2a_x(t) + a_{x-1}(t)}{(\Delta x)^2} \tag{4.13}$$

D is added to h with a coupling factor α, $h' = h(a_x, f_x) + \alpha D$:

$$\dot{a}_x = h(a_x, f_x) + \alpha D - a_x \sum_{x'} (h(a_{x'}, f_{x'}) + \alpha D) \tag{4.14}$$

4.3.1 Internal Sampling Process

The macrocolumn activity is a consequence of continuous stimulation. We are therefore able to identify two variables that have an influence on peak-building. The process is time-sensitive and input-sensitive. This is different to the KF which is only sensitive to input, but not to time in between epochs.
With algorithm 1, the NKF is stimulated with the same activity f_x for the whole time evolution in between two samples. This leads to a *burn-in effect* where the activity becomes bumpy exactly where the random sample o_x lands. This artificial, sample-based bias is then amplified by EDE induced concentration of activity. If a bumpy probability measure forms on a macrocolumn, then it is further amplified even with uniform input f_x. The peaky result does rarely represent the real distribution
An internal sampling process significantly reduces this prejudice towards individual samples and smoothes the probability measure. Algorithm 2 shows the modifications with respect to algorithm 1. Because we assume that the internal sampling process is independent in time, it effectively applies the central limit theorem, hence the overall stimulation per epoch approximates a normal distribution with mean o_x. The variance σ_o is a hyperparameter. This internal noise significantly improves its robustness against external noise. Additionally, having a wider stimulation allows to increase the hyperparameter (β) for a stronger concentration effect.

Simulator Initialization

The activity on the X and V macrocolumns are initialized uniformly ($p(x_i) = a_x = 1/N$).

4.4 Technical Implementation

We provide two implementations of the NKF. The first one is a **numpy**-based simulator with an optional graphical frontend. One can also run quantitative simulations in terminal mode. The performance of sequentially updating minicolumns on CPUs significantly drops for two- and three-dimensional

Algorithm 2: Extended 1D Velocity Estimation: An internal sampling process stimulates the X chain with K samples drawn from a normal distribtion with mean o_x.

Data: $a_v^t, a_x^t, N, T, S_x, S_v, X^\star, V^\star$
Result: a_x^{t+1}, a_v^{t+1}
$t \leftarrow 0$;
for $t \leq T$ **do**
  $g_x(z) \leftarrow \tilde{a}_x^{t+1}(z) = \sum_{k=-\infty}^{\infty} a_x^t(k) a_v^t(z-k)$
  $o_x \leftarrow \text{world}(x_t^\star, v_t^\star, \sigma_s)$
  for $k \leq K$ **do**
    $f_x \sim \mathcal{N}(o_x, \sigma_o)$
    for $s \leq S_x$ **do**
      $a_x^g \leftarrow \text{EVO}(h_x(g_x, f_x))$
    end
    $f_v(\Delta x) \leftarrow \sum_{k=-\infty}^{\infty} a_x^g(k) a_x^t(k+\Delta x)$
    for $s \leq S_v$ **do**
      $a_v \leftarrow \text{EVO}(h_v(a_v, f_v))$
    end
  end
  $a_v^{t+1} \leftarrow a_v$, $a_x^{t+1} \leftarrow a_x^g, t \leftarrow t+1$
end

macrocolumns. For that case, we provide a high-performance implementation in **triton**[61]. Triton is an API to the graphics card. The framework allows to run Python programs on Cuda-enabled GPUs. The Triton implementation accelerates computation by updating minicolumns in parallel on GPU's tensor cores.

4.4.1 Numerical Solutions, Convergance and Stability

We showed in section 3.2.1, that explicit Euler iteration, combined with constant renormalization by an inhibitory network, is a neurally acceptable implementation of the EDE. When solving differential equations by the Euler method, instability arises if the step size Δt is chosen too large. We safeguarded against that by measuring the local truncation error of the Euler solutions and reduced Δt if necessary. With the additive diffusion process, where a_x depends on the local neighborhood, we experienced oscillatory behavior. Switching to a Runge-Kutta method ($rtol = 1 \times 10^{-9}$ and $atol = 1 \times 10^{-9}$) with adaptive step size produced precise results.

Chapter 5

Experiments

In this chapter, we present the one-dimensional motion experiments that we conducted. All experiments are based on the simulator framework that we provide with this thesis. First, we study how well the NKF mimics the traditional KF. Later we test its robustness to noisy input patterns, as well as the effect of bias and diffusion to the stability of the filter. All experiments are recorded and available with the report. The theoretical introduction to this filter is in chapter 3. Implementation details are given in chapter 4. We use a NKF $h(a_x, f_x)$ variant with bias b, time constant β and diffusion $\{D, \alpha\}$:

$$h(a_x, f_x) = \beta\,(a_x f_x) + \alpha D(A_x^N) + b \tag{5.1}$$

5.1 General Observations

Here, we summarize effects that are present regardless of the experiment-specific filter configurations:

1. *Winner-takes-it-all characteristics*: In his work on hypercycles 2.7, Eigen states, that the final configuration of a system of differential equation as we express it with eq. 3.1 and 5.1 is a Dirac distribution. It is never our intent to arrive at that configuration, hence the filter needs to be carefully configured to balance activity between competing macrocolumns. Over-contraction should be avoided because for unbiased estimators, the Cramer-Rao bound expresses a lower bound on the variance. By that bound, we have to limit the system's contractive power to avoid biased estimates.

2. *Lack of Interpolation on instantaneous change*: The KF shifts its mean in direction of a new sensory input based on the Kalman gain. With that mechanism, the filter quickly reacts to instantaneous change. The NKF is unable to interpolate in between points. Hence, to overcome instantaneous change, a new peak-building process is initiated. For small changes, the activity around the previous mean already strongly favours neighboring minicolumns in the competitive process, so a new peak forms quickly,

which is not the case for larger, abrupt shifts in position or velocity. The bias b and diffusion process D in eq. 5.1 keep off-center minicolumns at a certain level of activity so they can react quickly to abrupt changes.

3. *Starvation of neural activity*: We showed in section 3.1.3 that a_x grows exponentially. On the other hand, if off-center neurons receive no stimulation for a certain period of time, the activity decays exponentially to 0. This would be acceptable if the minicolumns would exhibit a hysteresis loop which would allow them to change to active state quickly, but that is not the case. Due to the exponential characteristic, it takes a very long time to reactivate neurons that are close to 0. The bias b prevents that, but increases the variance of the system.

5.1.1 Comparison of the NKF to the KF

The states of the NKF and the KF are not directly comparable. We interpret the activity on a macrocolumn of the NKF as nonparametric density. On the other hand, the state of the KF is a multivariate, continuous Gaussian. We read out the mean of the activity distribution with a simple center-of-mass estimate [2] to compare both:

$$\hat{x} = \frac{\sum_i a_i x_i}{\sum_i a_i} \tag{5.2}$$

We compute the weighted sample variance s^2 from the weighted mean $\hat{x}$

$$s^2 = \frac{\sum_i a_i (x_i - \hat{x})^2}{\sum_i a_i} \tag{5.3}$$

Where a_i is the activity of a minicolumn and x_i corresponds to the assigned location or velocity. We take this network estimate $(\hat{x}, s^2)$ to compare its mean and variance to the KF.

5.2 Sharp Turn Recovery

A winner-takes-it-all process cannot easily react to changing input. An abrupt change means that most of the activity that accumulated on a certain minicolumn must be passed onto a different one which is tuned for the new stimulus. In our experiments we study the filter response to such abrupt changes. We are interested in how much time redistribution takes and how that influences the variance. We initiate a movement in one direction and after N_e epochs, we abruptly stop and change the sign of the velocity. The NKF is sensitive to the iteration strength parameters β_v and β_x for the EVO filters. Larger time constants β increase the convergence rate, but result in reduced sensitivity to the abrupt change. We compare two filters in figure 5.1 ($\beta = 1.2$) and figure 5.2 ($\beta = 1$). We show that the Kalman filter is also sensitive to configuration. By setting a high process noise $\mathbf{Q}$, the KF is more sensitive to abrupt changes and reacts immediately (5.1), but the variance in estimated position and velocity

increases and the estimator begins to overfit to the data points, once noise is added. Both effects also apply to the NKF.

5.3 Noise

In the previous section, we studied the filter's capability to react to unforeseen change. For that, the world was assumed to be noise-free. We expect our sensor to be able to deal with a certain amount of noise. In section 4.1 we introduced a world model with configurable Gaussian noise for the sensor σ_s and the process σ_w. In this section, we present the results for sensor and process noise of different magnitudes.

5.3.1 Sensor Noise

In figure 5.3 ($\sigma_w = 5$) and 5.4 ($\sigma_w = 10$) we show typical responses of the NKF to sensor noise σ_s. We found that the NKF is relatively robust to medium sensor noise $\sigma_s = 5$ and $\hat{x}_{KF} \approx \hat{x}_{NKF}$. For larger noise $\sigma_s = 10$ we observe two outcomes. If we tune the sensor to be aggressive to estimate a precise velocity, (larger K, $\beta_v > \beta_x > 1$), then we observe a small bias in $\hat{x}$ estimation, but large variance, because the filter is overly sensitive to new input. The velocity estimation $\hat{v}$ becomes especially unreliable. If we decrease competitive concentration on the velocity, then we increase the variance in the V distribution. This effect propagates to the X distribution via generative model g_x. Additionally, due to the predominance of the low-sensitivity velocity estimation, we observe the addition of a systematic bias to the position estimate (figure 5.5).

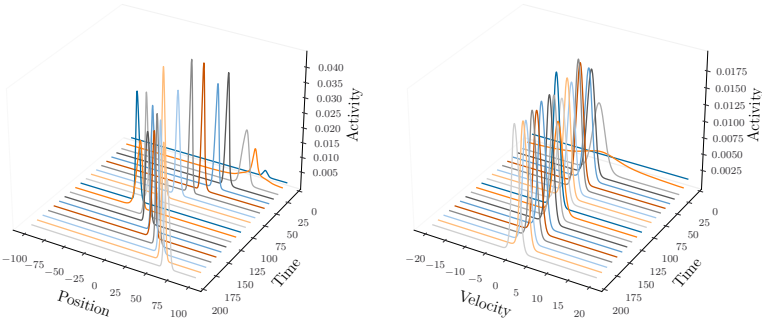
5.3.2 Process Noise

The addition of moderate process noise ($\sigma_w = 2$) distorts the path of the tracked object significantly away from the previously shown v-like trajectory. In comparison to sensor noise, this noise affects the true trajectory of the tracked object. We expect to estimate a non-continuous velocity estimation due to the implicit encoding of velocity via probabilistic addition and subtraction in the NKF. We expect the same results from the KF as the linear model of velocity is encoded in the process model $\mathbf{F}$. In figure 5.6, we show what we consider a successful response of the NKF to process noise σ_w. In figure 5.7, we show the most prominent failure that we observed, loss of tracking. In all observed cases, the filter was able to recover from that failure mode over time.

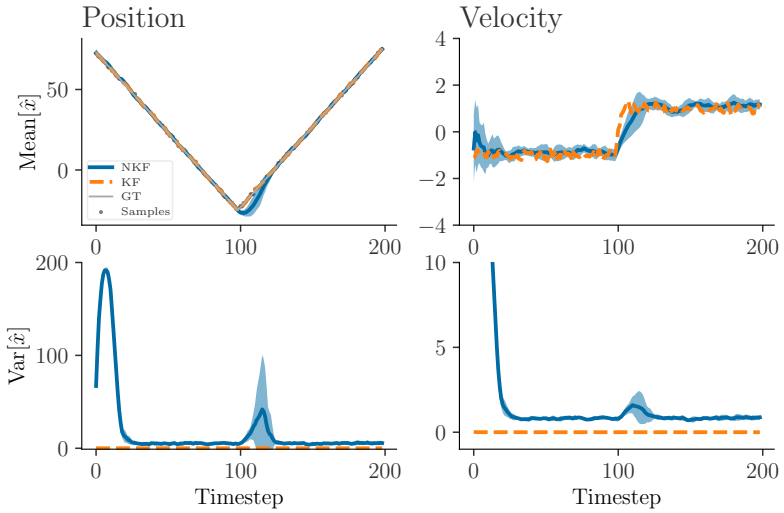
5.4 Bias, Diffusion

We argued in sections 4.3, 4.3 and 5.1 why regulatory effects such as bias b and diffusion D are required. Here, we want to experimentally show how the filter reacts to abrupt change without such regulation. In figure 5.8, we set $b = 0$ and $\alpha = 0$. We notice in the neural activity snapshots (subplot a), that all activity on the macrocolumn quickly collapses to a single, all-dominating minicolumn for both, position estimation and velocity. The filter precisely estimates the

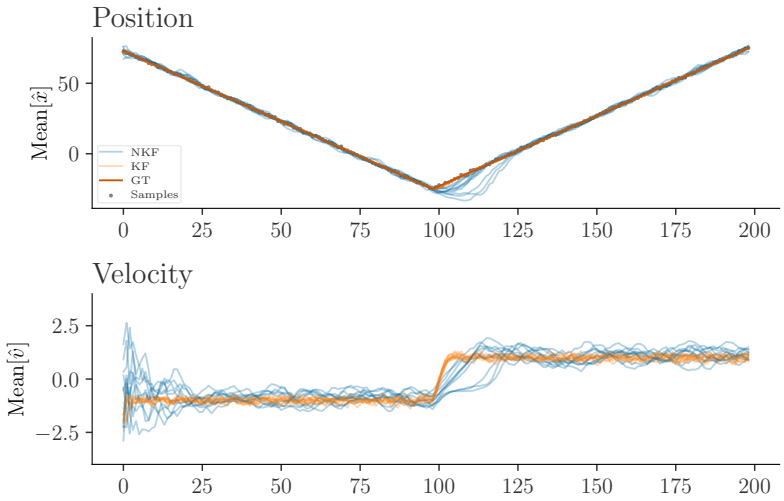
correct quantities $\hat{v} \approxeq v^{\star}$ and $\hat{x} \approxeq x^{\star}$. However, once we flip the sign of the velocity, the filter is unable to adapt, because off-peak minicolumn activity is ≈ 0. By adding a very small bias ($b = 1 \times 10^{-7}$, figure 5.9), we show that the filter still overshoots, but it reacts significantly faster to the change. It is in the interest of the operator to balance the filter such that the bias (which increases the variance of the estimator) is small, but large enough such that the filter's minicolumns operate in a reactive zone of the exponential curve.
Adding diffusion to the process is an inductive bias. We state that if the sensory input does not stimulate a region strongly anymore, then we transfer activity to neighboring units, so that the probability of states in the immediate neighborhood increases. Additionally to a bias b, diffusion increases the off-center neural activity further. We observe in figure 5.10 that in case of a peak-shift the variance is much better controlled, even with moderate diffusion ($\alpha = 0.2$). From the detailed neural activity (subplot a), we see that the old peak quickly diffuses to the right, which accelerates the formation of the new peak.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

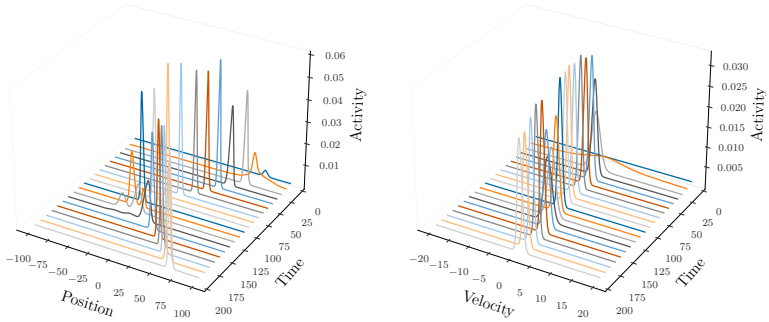


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

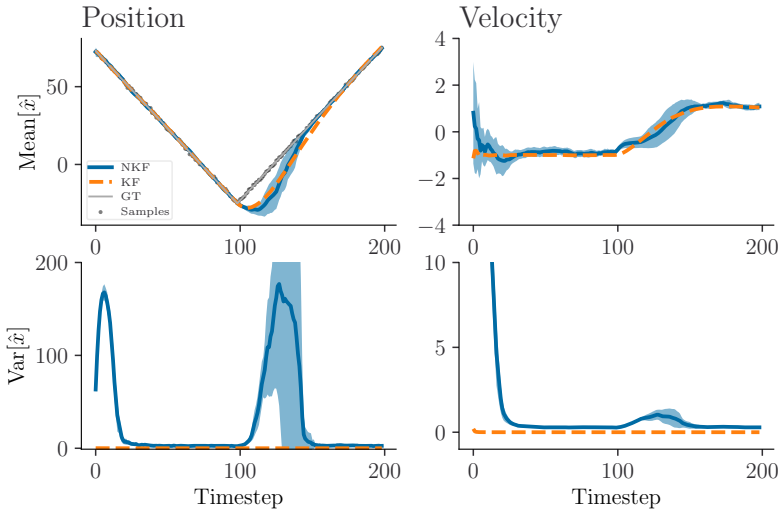


(c) Traces for $\hat{x}$ and $\hat{v}$ for all trials.

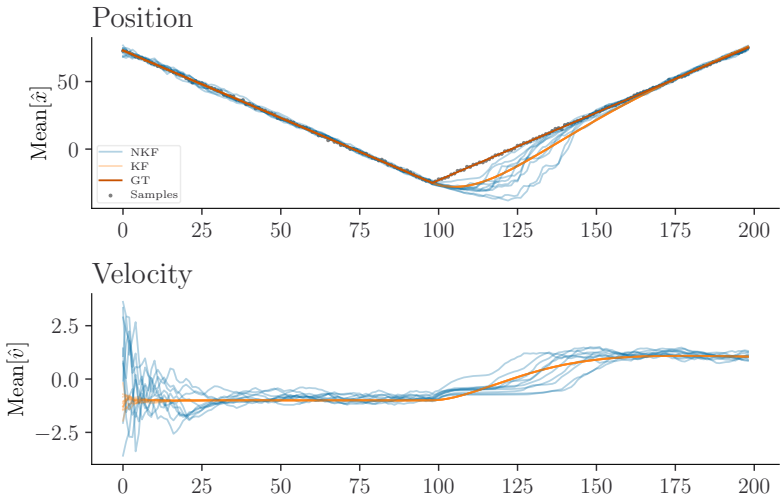
Figure 5.1: Sharp turn recovery of a NKF and reference KF with high sensitivity setting ($\beta_x = 1.2$ and $\beta_v = 1.2$). The NKF quickly concentrates its activity to the new velocity $v^\star = 1$. Due to peak-building, we observe minimal overshoot at the bottom.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

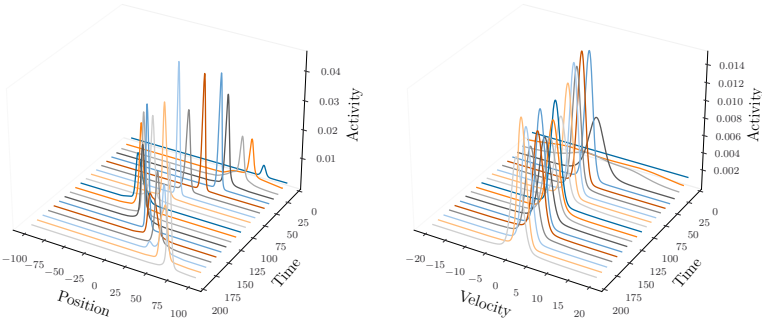


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

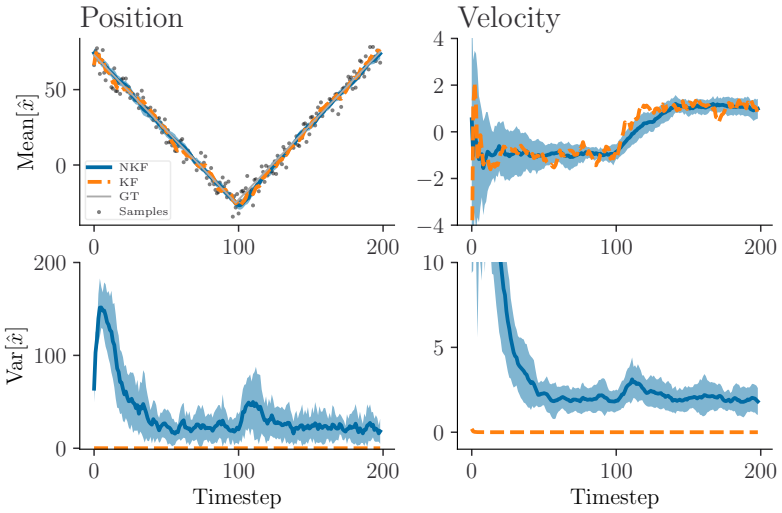


(c) Traces for $\hat{x}$ and $\hat{v}$ for all trials.

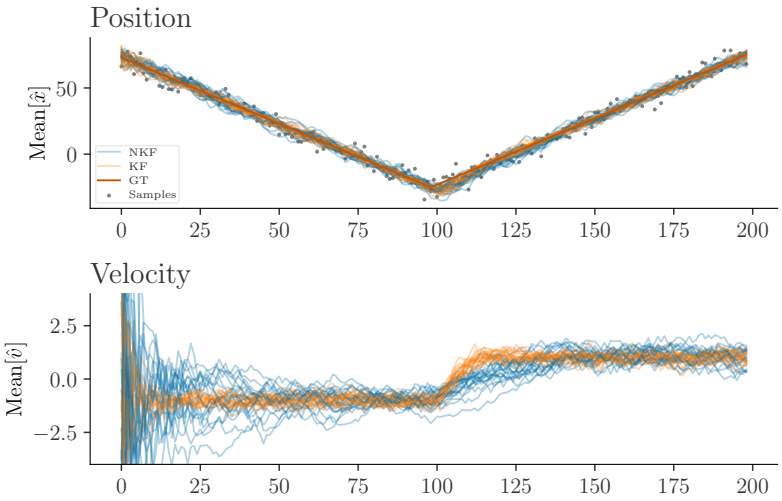
Figure 5.2: Sharp turn recovery of a NKF and reference KF with low sensitivity setting ($\beta_x = 1$ and $\beta_v = 1$). The NKF is less aggressively tuned. Therefore, once the stimulus f_x moves in the opposite direction, the activity at the old site takes longer to fade. Bias b and diffusion D push the activity towards a uniform configuration, hence, without strong concentration, the variance increases dramatically.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

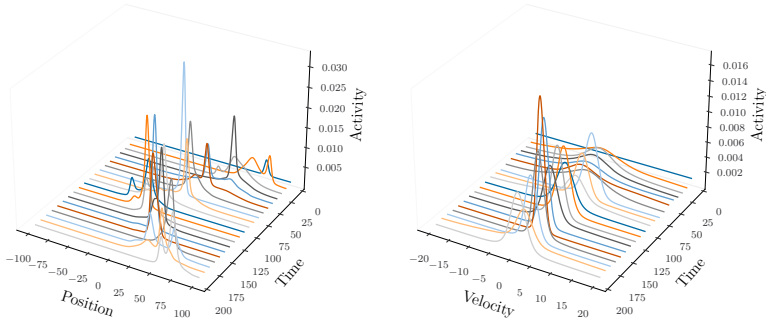


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

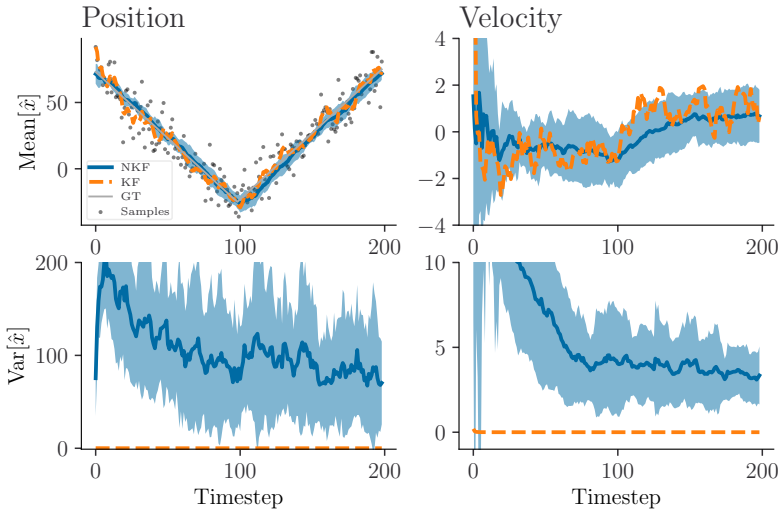


(c) Traces for $\hat{x}$ and $\hat{v}$ for all trials.

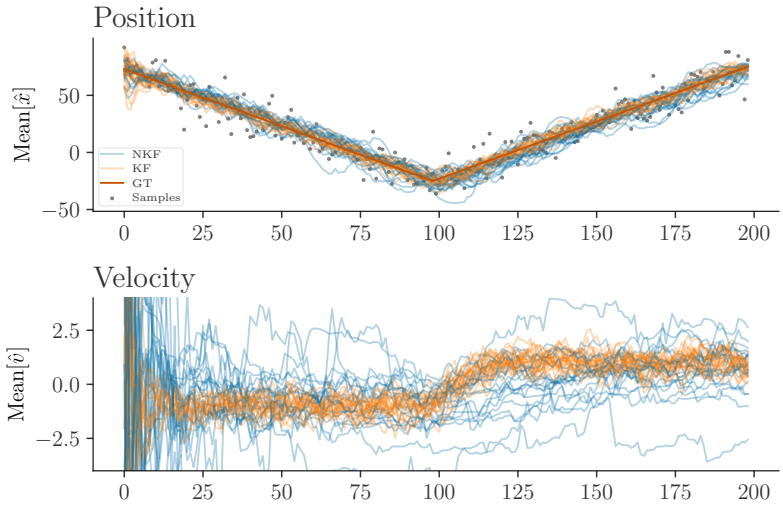
Figure 5.3: Here we show the response of the NKF to medium sensor noise $\sigma_s = 5$. The filter correctly estimates position and velocity. The NKF shows a warm-up period to find the correct velocity from its unifom initialization, it however reacts competitvely compared to the KF after we change the true velocity to $v^\star = 1$.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

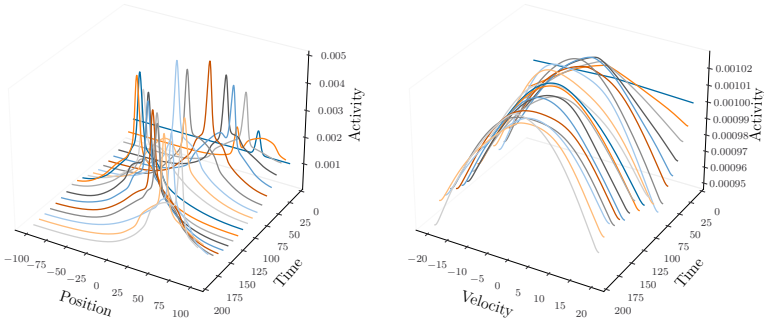


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

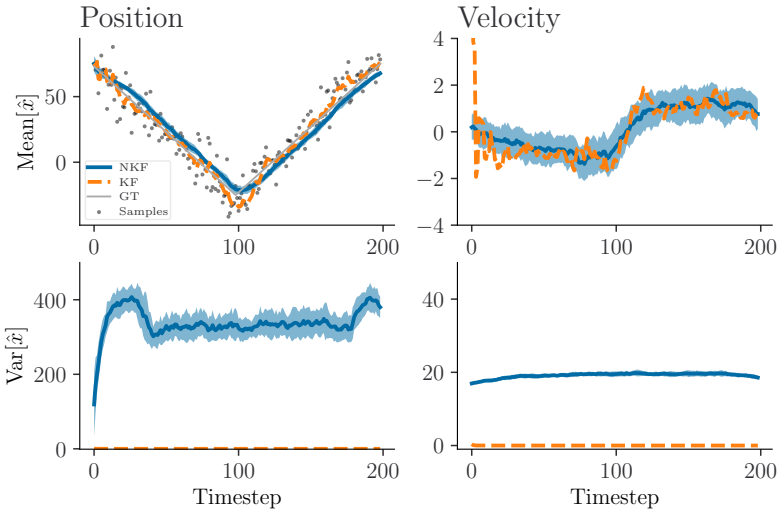


(c) Traces for $\hat{x}$ and $\hat{v}$ for all trials.

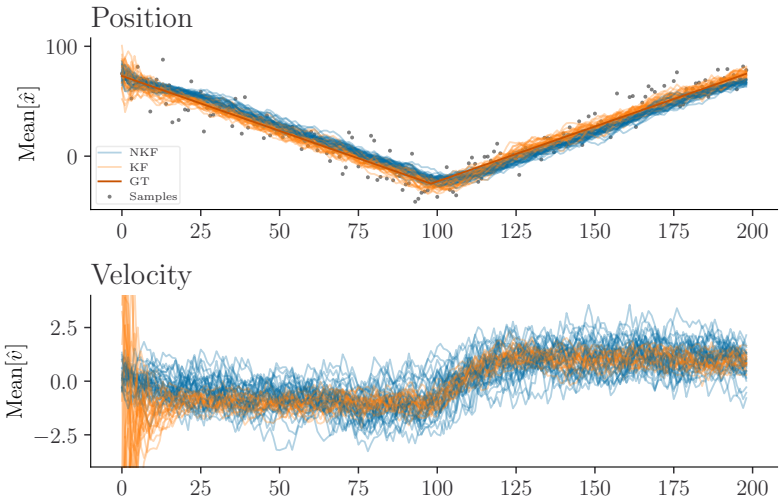
Figure 5.4: For agressive sensor noise $\sigma_s = 10$, the NKF fails to recover the true velocity frequently. Hence in c) we observe in $\hat{x}$ that for extreme mispredictions of $\hat{v}$, the filter fails to keep its estimate close to the ground truth and diverges.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

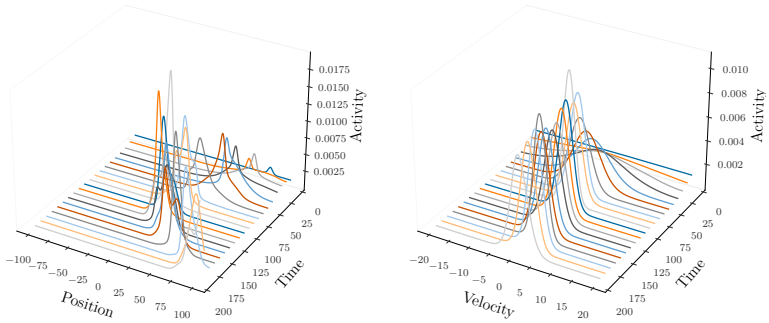


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

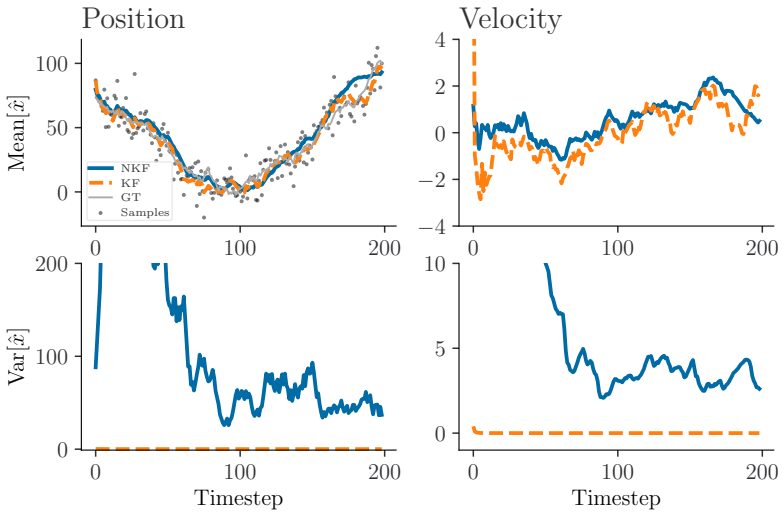


(c) Traces for $\hat{x}$ and $\hat{v}$ for all trials.

Figure 5.5: An alternative NKF configuration for large sensor noise $\sigma_s = 10$ with decreased competitive concentration in the V macrocolumn. In a) we observe large variance in the V activity. However, because the weighted mean is stable, the position estimate $\hat{x}$ shows much less variance, but a systematic bias.

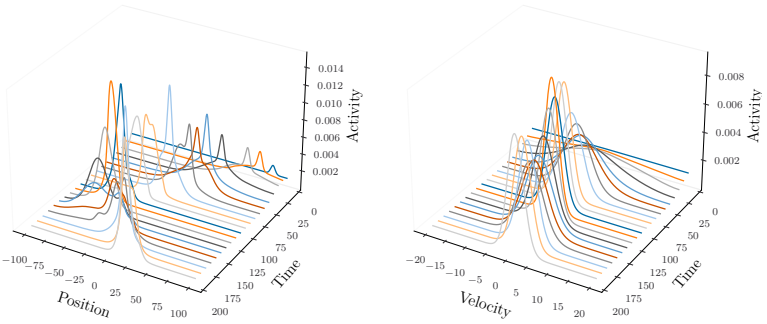


(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

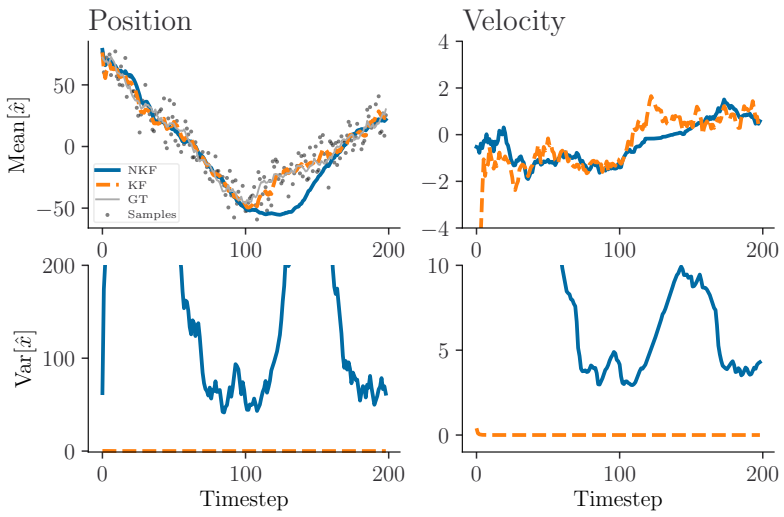


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

Figure 5.6: Successful tracking of the position. Sensor noise $\sigma_s = 10$ and process noise $\sigma_w = 2$.

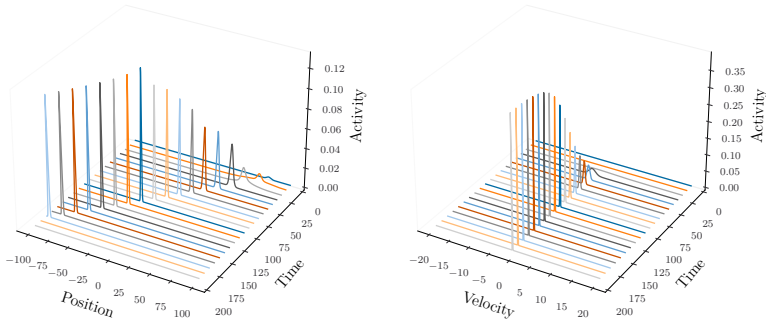


(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

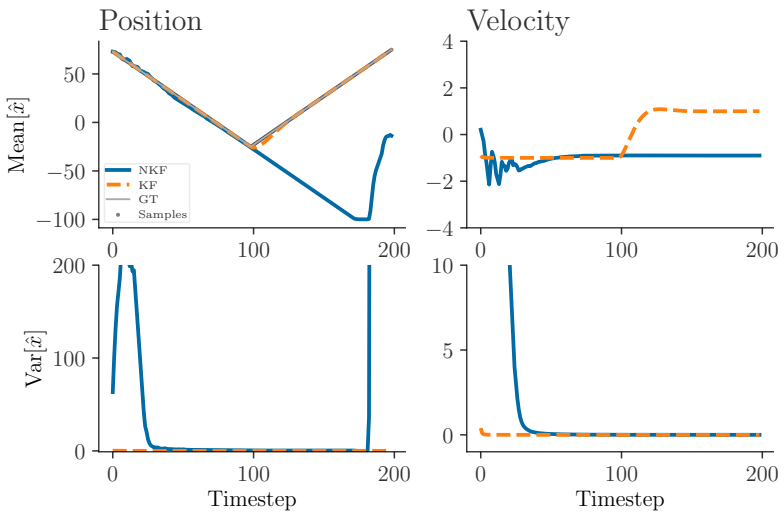


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

Figure 5.7: Typical failure during tracking of the position. The NKF manages to reliably recover from this class of failures. Sensor noise $\sigma_s = 10$ and process noise $\sigma_w = 2$.

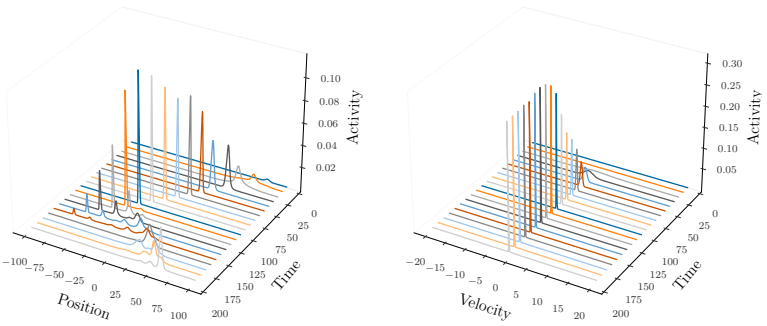


(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

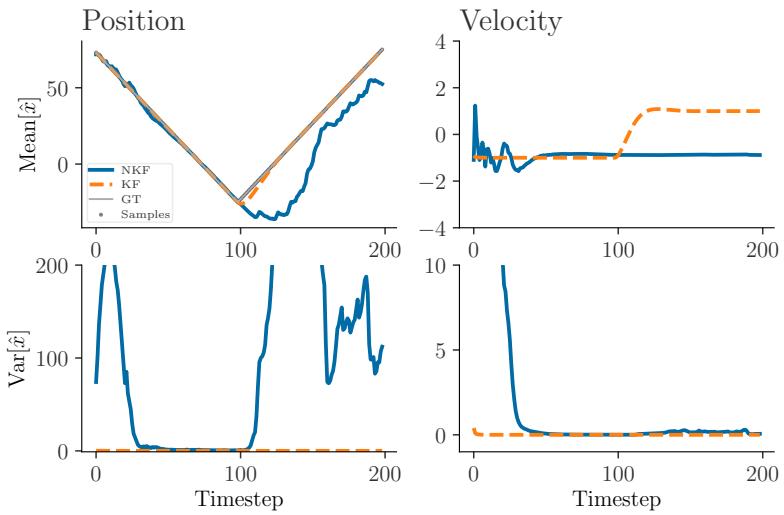


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

Figure 5.8: With no bias $b = 0$, diffusion $\alpha = 0$ and noise, the macrocolumn that is most sensitive to the stimulus quickly accumulates all of the limited activity available in the self normalizing process. However, this configuration burns in due to off stimulus activity approaching zero. As a consequence, the filter is unable to quickly react to changes.

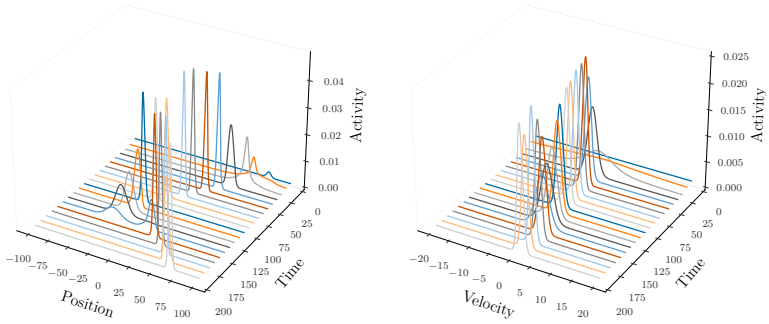


(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.

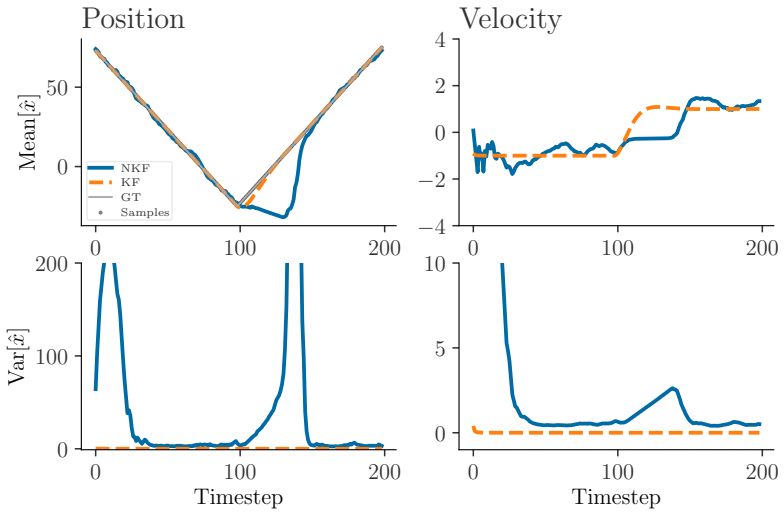


(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

Figure 5.9: The macrocolumn that is most sensitive to the stimulus only accumulates activity equivalent to $1 - Nb$. Therefore, the unstimulated minicolumns operate in a more reactive zone of exponential growth, which renders the macrocolumn more reactive to change. In a) we observe that once the stimulus changes direction, several peaks appear before the correct one dominates because the system has no prejudice towards neighboring stimuli.



(a) Snapshots of the neural acivity of the X and V macrocolumns taken every 10^{th} epoch for a single trial.



(b) Averages for weighted means and weighted variances of the NKF for $n = 30$ trials (GT: ground truth).

Figure 5.10: Diffusion adds two effects to the macrocolumn activity distribution. The strength α controls the width of a peak over the neighborhood of the minicolumn that is most sensitive to the stimulus. As a consequence, once the stimulus changes direction, the filter is more selective to minicolumns in the neighborhood of the previous activity. Hence, we observe a much cleaner formation of the new peak and with that, less variance build-up on both X and V macrocolumns.

Chapter 6

Discussion

In this thesis, we present a neural filtering algorithm that applies the evolutionary differential equation (EDE), a competitive update scheme for macromolecules from computational biology, to compute the dynamic evolution of neural populations (minicolumns). We topologically arrange minicolumns in chains (macrocolums) that cover the full range of each environmental random variable that we want to estimate. We define a PPC on each macrocolumn by tuning the activity of minicolumns to a certain range of the environmental variable. In that configuration, all random variables of our neural filtering algorithm are represented with neural firing. This method has the advantage that downstream circuits can directly form synaptic connections to the representation of a random variable. Our strategy differs from related approaches [1], [2], where the authors apply a Bayesian decoding scheme to transform the PPC to a virtual latent space, in which they only mathematically represent the environmental variable as a continuous Gaussian random variable.

We study the EDE to understand the dynamical evolution of a macrocolum. In a theoretical analysis, we find that for any sensory input, the attractor dynamics of the EDE moves the activity of a macrocolumn towards a normalized state. We find that this inhibitory effect is akin to divisive normalization for certain configurations. This finding allows for a direct interpretation of the average neural activity at each site as a probability measure. Divisive normalization has been observed in various brain areas such as the primary visual cortex [62]–[64], the extrastriate visual cortex [65]–[71], the superior colliculus [72] and the antenna lobe of the Drosophila [73] but the question about neural implementation remains open [1]. We show that divisive normalization in EDE-governed dynamic evolution can be implemented with different mechanisms: either by subtraction of accumulated modulated activity or with a separate (lateral) inhibitory system. We hope that this will open up additional paths for solving the open question about the circuit implementation.

We find an particular analytical solution to the EDE that shows that individual minicolumns react exponentially to a stimulus. Based on this finding, we discovered an optimal bias adjustment such that the minicolumn activity is sensitive to abrupt changes. In biological systems, this bias adjustment could be realized by neuronal noise. We further study the structure of unimodal nonparametric

densities that evolve on macrocolumns and compare them to the statistics of the KF. We find ways to control properties of a nonparametric density such as the variance and kurtosis by tuning the hyperparameter of the EDE.
We study the addition and subtraction of nonparametric random variables, which are both convolution operations, to implement a linear motion model within the NKF. For that, we show that the marginalization step performed by the KF corresponds to the convolution of the velocity and position signal. We present a neural circuit, equivalent to our proposed algorithm, that combines all operations for the task of linear motion estimation.
Finally, for the performance evaluation of our algorithm, we developed an interactive, agent-based simulator that provides methods to compare the nonparametric distributions to the KF. We find that the filtering is Kalman-like for typical trajectories with low to medium noise. In the evaluation, we found that the competitive peak-building of our algorithm must be reverted if there are jumps in the signal because we have no higher-order concepts to shift the center of mass in our activity distributions. This is a disadvantage if the stimulus acts unpredictably (e.g. flips the sign of direction). To make the algorithm more robust to moderate to large noise, we add an internal sampling process, that stimulates the macrocolumns around the stimulus. It remains unclear how this mechanism is realized neurally.

6.1 Future Work

Our model significantly improves our understanding of the conceptual elements required for nonparametric filtering of sensory information, but some questions remain open. Here we give a brief overview of how we would continue the work on the NKF.
With our proposed algorithm, we restrict ourselves to one-dimensional neural chains because multivariate distributions scale exponentially with the number of dimensions in PPC representations. Our approach requires significantly fewer units and connections, but there is a cost in terms of performance, since we are unable to represent correlations between position and velocity [2]. A future approach that considers covariances would require us to maintain $2N^2$ minicolumns. This would drastically increase simulation time. We lay out the groundwork to tackle this challenge by providing a GPU implementation of the EDE that accelerates the process by a factor of 70.
Currently, we provide all connectivity to the algorithm and assume that synapses have been learned previously. For a theory of the development of synaptic connections between two neural fields see [74]. Using that as guide, it would be interesting to find out how the connections we assume for our algorithm emerge.
The Kalman filter computes the gain to interpolate between hidden state and measurement to update mean and variance. Our algorithm lacks a mechanism to neurally compute an equivalent interpolation. Hence, different mechanisms are required to optimally estimate weighted mean and variance. From plots, it is clear that our algorithm has sub-optimal variance compared to the KF. The study [10] presents a system with optimal variance, though implemented

in non-neural fashion. This approach works well if the recurrent connections of the network are a function of the optimal velocity. Two open questions remain. Firstly, if it is possible to express the recurrent connections as a function of a macrocolumn representing velocity instead and secondly, how to connect the neurons of the macrocolumn to the recurrent connections in practice.
Interestingly, [2] suggests that neurophysiological evidence shows that neural activity decay follows a power law during prolonged presentation of the same stimulus. This is what we observe for EDE-governed evolution in off-stimulus regions. Further investigation is required to compare the two lines of work.

Bibliography

[1] J. M. Beck, P. E. Latham, and A. Pouget, "Marginalization in Neural Circuits with Divisive Normalization," en, *Journal of Neuroscience*, vol. 31, no. 43, pp. 15 310–15 319, Oct. 2011, ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.1706-11.2011. [Online]. Available: https://www.jneurosci.org/lookup/doi/10.1523/JNEUROSCI.1706-11.2011 (visited on 05/05/2021).

[2] S. Denève, J.-R. Duhamel, and A. Pouget, "Optimal Sensorimotor Integration in Recurrent Cortical Networks: A Neural Implementation of Kalman Filters," en, *Journal of Neuroscience*, vol. 27, no. 21, pp. 5744–5756, May 2007, ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.3985-06.2007. [Online]. Available: https://www.jneurosci.org/content/27/21/5744 (visited on 11/02/2021).

[3] J. Lücke and C. v. d. Malsburg, "Rapid Processing and Unsupervised Learning in a Model of the Cortical Macrocolumn," *Neural Computation*, vol. 16, no. 3, pp. 501–533, Mar. 2004, ISSN: 0899-7667. DOI: 10.1162/089976604772744893. [Online]. Available: https://doi.org/10.1162/089976604772744893 (visited on 11/14/2021).

[4] R. Kálmán, "A new approach to linear filtering and prediction problems" transaction of the asme˜journal of basic," 1960. DOI: 10.1115/1.3662552.

[5] J. Humpherys, P. Redd, and J. West, "A Fresh Look at the Kalman Filter," en, *SIAM Review*, vol. 54, no. 4, pp. 801–823, Jan. 2012, ISSN: 0036-1445, 1095-7200. DOI: 10.1137/100799666. [Online]. Available: http://epubs.siam.org/doi/10.1137/100799666 (visited on 06/22/2021).

[6] K. E. Stephan, Z. M. Manjaly, C. D. Mathys, *et al.*, "Allostatic Self-efficacy: A Metacognitive Theory of Dyshomeostasis-Induced Fatigue and Depression," eng, *Frontiers in Human Neuroscience*, vol. 10, p. 550, 2016, ISSN: 1662-5161. DOI: 10.3389/fnhum.2016.00550.

[7] K. Friston, T. FitzGerald, F. Rigoli, P. Schwartenbeck, J. O'Doherty, and G. Pezzulo, "Active inference and learning," en, *Neuroscience & Biobehavioral Reviews*, vol. 68, pp. 862–879, Sep. 2016, ISSN: 0149-7634. DOI: 10.1016/j.neubiorev.2016.06.022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0149763416301336 (visited on 11/13/2021).

[8] N. V. K. Medathati, H. Neumann, G. Masson, and P. Kornprobst, "Bio-Inspired Computer Vision: Towards a Synergistic Approach of Artificial and Biological Vision," *Computer Vision and Image Understanding*, vol. 150, Apr. 2016. DOI: 10.1016/j.cviu.2016.04.009.

[9] M. Carandini and D. J. Heeger, "Normalization as a canonical neural computation," en, *Nature Reviews Neuroscience*, vol. 13, no. 1, pp. 51–62, Jan. 2012, ISSN: 1471-0048. DOI: 10.1038/nrn3136. [Online]. Available: https://www.nature.com/articles/nrn3136 (visited on 04/30/2021).

[10] R. Wilson and L. Finkel, "A Neural Implementation of the Kalman Filter," en, p. 9, 2009. [Online]. Available: https://papers.neurips.cc/paper/2009/file/6d0f846348a856321729a2f36734d1a7-Paper.pdf.

[11] M. Eigen and P. Schuster, *The Hypercycle: A Principle of Natural Self-Organization*, en. Berlin Heidelberg: Springer-Verlag, 1979, ISBN: 978-3-540-09293-3. DOI: 10.1007/978-3-642-67247-7;https://web.archive.org/web/20210913100832/https://www.springer.com/gp/book/9783540092933. [Online]. Available: https://www.springer.com/gp/book/9783540092933 (visited on 09/13/2021).

[12] F. Auger, M. Hilairet, J. M. Guerrero, E. Monmasson, T. Orlowska-Kowalska, and S. Katsura, "Industrial applications of the kalman filter: A review," *IEEE Transactions on Industrial Electronics*, vol. 60, no. 12, pp. 5458–5471, 2013. DOI: 10.1109/TIE.2012.2236994.

[13] R. Labbe, *Kalman and Bayesian Filters in Python*, Oct. 2021. [Online]. Available: https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python/blob/ee586f5d84664383d079aa0bcb480fa78c8b0d2d/06-Multivariate-Kalman-Filters.ipynb (visited on 10/15/2021).

[14] M. Hoshiya and E. Saito, "Structural identification by extended kalman filter," *Journal of engineering mechanics*, vol. 110, no. 12, pp. 1757–1770, 1984.

[15] E. A. Wan, R. Van Der Merwe, and S. Haykin, "The unscented kalman filter," *Kalman filtering and neural networks*, vol. 5, no. 2007, pp. 221–280, 2001.

[16] Y.-t. Bai, X.-y. Wang, X.-b. Jin, Z.-y. Zhao, and B.-h. Zhang, "A Neuron-Based Kalman Filter with Nonlinear Autoregressive Model," en, *Sensors*, vol. 20, no. 1, p. 299, Jan. 2020, ISSN: 1424-8220. DOI: 10.3390/s20010299. [Online]. Available: https://www.mdpi.com/1424-8220/20/1/299 (visited on 05/05/2021).

[17] W. J. Ma, J. M. Beck, P. E. Latham, and A. Pouget, "Bayesian inference with probabilistic population codes," en, *Nature Neuroscience*, vol. 9, no. 11, pp. 1432–1438, Nov. 2006, ISSN: 1097-6256, 1546-1726. DOI: 10.1038/nn1790. [Online]. Available: http://www.nature.com/articles/nn1790 (visited on 05/05/2021).

[18] "Foundations of the theory of probability : Kolmogorov, a. n : Free download, borrow, and streaming : Internet archive." (), [Online]. Available: https://archive.org/details/foundationsofthe00kolm (visited on 08/03/2021).

[19] H. Barlow, "Possible Principles Underlying the Transformations of Sensory Messages," *Sensory Communication*, vol. 1, Jan. 1961, ISSN: 9780262518420. DOI: 10.7551/mitpress/9780262518420.003.0013.

[20] J. Thrommershäuser, L. T. Maloney, and M. S. Landy, "Decision Making, Movement Planning, and Statistical Decision Theory," *Trends in cognitive sciences*, vol. 12, no. 8, pp. 291–297, Aug. 2008, ISSN: 1364-6613. DOI: 10.1016/j.tics.2008.04.010. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2678412/ (visited on 10/03/2021).

[21] J. Fiser, P. Berkes, G. Orbán, and M. Lengyel, "Statistically optimal perception and learning: From behavior to neural representations," en, *Trends in Cognitive Sciences*, vol. 14, no. 3, pp. 119–130, Mar. 2010, ISSN: 1364-6613. DOI: 10.1016/j.tics.2010.01.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1364661310000045 (visited on 10/03/2021).

[22] D. H. Hubel and T. N. Wiesel, "Receptive fields of single neurones in the cat's striate cortex," *The Journal of Physiology*, vol. 148, no. 3, pp. 574–591, Oct. 1959, ISSN: 0022-3751. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC1363130/ (visited on 10/04/2021).

[23] *Probabilistic Population Codes for Bayesian Decision Making — Elsevier Enhanced Reader*, en. DOI: 10.1016/j.neuron.2008.09.021. [Online]. Available: https://reader.elsevier.com/reader/sd/pii/S0896627308008039?token=012D7C9A18C149EA061172409918751071F09ABC63A079C362BE80F86DDAD71A21A4D31C4E2BC34FCDA24C00F390D64D&originRegion=eu-west-1&originCreation=20211003095047 (visited on 10/03/2021).

[24] P. Berkes, G. Orban, M. Lengyel, and J. Fiser, "Spontaneous Cortical Activity Reveals Hallmarks of an Optimal Internal Model of the Environment," en, *Science*, vol. 331, no. 6013, pp. 83–87, Jan. 2011, ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1195870. [Online]. Available: https://www.sciencemag.org/lookup/doi/10.1126/science.1195870 (visited on 10/04/2021).

[25] G. Orbán, P. Berkes, J. Fiser, and M. Lengyel, "Neural Variability and Sampling-Based Probabilistic Representations in the Visual Cortex," en, *Neuron*, vol. 92, no. 2, pp. 530–543, Oct. 2016, ISSN: 0896-6273. DOI: 10.1016/j.neuron.2016.09.038. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0896627316306390 (visited on 10/03/2021).

[26] P. O. Hoyer and A. Hyvärinen, "Interpreting Neural Response Variability as Monte Carlo Sampling of the Posterior," en, p. 8,

[27] T. S. Lee and D. Mumford, "Hierarchical Bayesian inference in the visual cortex," eng, *Journal of the Optical Society of America. A, Optics, Image Science, and Vision*, vol. 20, no. 7, pp. 1434–1448, Jul. 2003, ISSN: 1084-7529. DOI: 10.1364/josaa.20.001434.

[28] C. D. Gilbert and M. Sigman, "Brain States: Top-Down Influences in Sensory Processing," en, *Neuron*, vol. 54, no. 5, pp. 677–696, Jun. 2007, ISSN: 0896-6273. DOI: 10.1016/j.neuron.2007.05.019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0896627307003765 (visited on 10/04/2021).

[29] D. H. Ballard, M. M. Hayhoe, G. Salgian, and H. Shinoda, "Spatio-temporal organization of behavior," eng, *Spatial Vision*, vol. 13, no. 2-3, pp. 321–333, 2000, ISSN: 0169-1015. DOI: 10.1163/156856800741144.

[30] J. Lücke, "Information Processing and Learning in Networks of Cortical Columns," Ph.D. dissertation, Bochum, 2004. [Online]. Available: https://vfs.fias.science/d/3cfce0fe5a/files/?p=/LueMal2004.pdf.

[31] A. L. Gibbs and F. E. Su, "On choosing and bounding probability metrics," *arXiv:math/0209021*, Sep. 2002. [Online]. Available: http://arxiv.org/abs/math/0209021 (visited on 11/01/2021).

[32] L. Paninski, "Estimation of Entropy and Mutual Information," *Neural Computation*, vol. 15, no. 6, pp. 1191–1253, Jun. 2003, ISSN: 0899-7667. DOI: 10.1162/089976603321780272.

[33] D. S. Andres, D. Cerquetti, M. Merello, and R. Stoop, "Neuronal Entropy Depends on the Level of Alertness in the Parkinsonian Globus Pallidus in vivo," *Frontiers in Neurology*, vol. 5, p. 96, 2014, ISSN: 1664-2295. DOI: 10.3389/fneur.2014.00096. [Online]. Available: https://www.frontiersin.org/article/10.3389/fneur.2014.00096 (visited on 11/01/2021).

[34] D. Wolpert, Z. Ghahramani, and M. Jordan, "An Internal Model for Sensorimotor Integration," *Science (New York, N.Y.)*, vol. 269, pp. 1880–2, Oct. 1995. DOI: 10.1126/science.7569931.

[35] K. Kording, U. Beierholm, W. Ma, S. Quartz, J. Tenenbaum, and L. Shams, "Causal Inference in Multisensory Perception," *PloS one*, vol. 2, e943, Sep. 2007. DOI: 10.1371/journal.pone.0000943.

[36] A. Blaisdell, K. Sawa, K. Leising, and M. Waldmann, "Causal Reasoning in Rats," *Science (New York, N.Y.)*, vol. 311, pp. 1020–2, Mar. 2006. DOI: 10.1126/science.1121872.

[37] T. Koski, "Lecture notes for sf2940 Probability Theory Edition: 2017," en, *University: KTH Royal Institute of Technology*, p. 346, 2017.

[38] M. (https://stats.stackexchange.com/users/247889/mossmyr), *Why is the sum of two random variables a convolution?* Cross Validated. eprint: https://stats.stackexchange.com/q/427357. [Online]. Available: https://stats.stackexchange.com/q/427357.

[39] J. P. H. van Santen and G. Sperling, "Elaborated Reichardt detectors," en, *Journal of the Optical Society of America A*, vol. 2, no. 2, p. 300, Feb. 1985, ISSN: 1084-7529, 1520-8532. DOI: 10.1364/JOSAA.2.000300. [Online]. Available: https://www.osapublishing.org/abstract.cfm?URI=josaa-2-2-300 (visited on 11/01/2021).

[40] J. Haag, W. Denk, and A. Borst, "Fly motion vision is based on Reichardt detectors regardless of the signal-to-noise ratio," en, *Proceedings of the National Academy of Sciences*, vol. 101, no. 46, pp. 16 333–16 338, Nov. 2004, ISSN: 0027-8424, 1091-6490. DOI: 10.1073/pnas.0407368101. [Online]. Available: https://www.pnas.org/content/101/46/16333 (visited on 11/01/2021).

[41] D. C. Bradley and M. S. Goyal, "Velocity computation in the primate visual system," en, *Nature Reviews Neuroscience*, vol. 9, no. 9, pp. 686–695, Sep. 2008, ISSN: 1471-0048. DOI: 10.1038/nrn2472. [Online]. Available: https://www.nature.com/articles/nrn2472 (visited on 08/06/2021).

[42] S. Nishida, T. Kawabe, M. Sawayama, and T. Fukiage, "Motion Perception: From Detection to Interpretation," en, *Annual Review of Vision Science*, vol. 4, no. 1, pp. 501–523, Sep. 2018, ISSN: 2374-4642, 2374-4650. DOI: 10.1146/annurev-vision-091517-034328. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev-vision-091517-034328 (visited on 08/06/2021).

[43] F. Musso, "A Stochastic Version of the Eigen Model," *Bulletin of mathematical biology*, vol. 73, pp. 151–80, Mar. 2010. DOI: 10.1007/s11538-010-9525-4.

[44] M. Boerlijst and P. Hogeweg, "Spiral wave structure in pre-biotic evolution: Hypercycles stable against parasites," en, *Physica D: Nonlinear Phenomena*, vol. 48, no. 1, pp. 17–28, Feb. 1991, ISSN: 01672789. DOI: 10.1016/0167-2789(91)90049-F. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/016727899190049F (visited on 10/04/2021).

[45] M. Boerlijst and P. Hogeweg, "Spatial gradients enhance persistence of hypercycles," *Physica D: Nonlinear Phenomena*, vol. 88, no. 1, pp. 29–39, 1995. DOI: 10.1016/0167-2789(95)00178-7.

[46] A. Pouget, J. M. Beck, W. J. Ma, and P. E. Latham, "Probabilistic brains: Knowns and unknowns," en, *Nature Neuroscience*, vol. 16, no. 9, pp. 1170–1178, Sep. 2013, ISSN: 1097-6256, 1546-1726. DOI: 10.1038/nn.3495. [Online]. Available: http://www.nature.com/articles/nn.3495 (visited on 10/04/2021).

[47] P. X. Joris, P. H. Smith, and T. C. T. Yin, "Coincidence Detection in the Auditory System: 50 Years after Jeffress," English, *Neuron*, vol. 21, no. 6, pp. 1235–1238, Dec. 1998, ISSN: 0896-6273. DOI: 10.1016/S0896-6273(00)80643-1. [Online]. Available: https://www.cell.com/neuron/abstract/S0896-6273(00)80643-1 (visited on 10/29/2021).

[48] Y. Trotter, S. Celebrini, B. Stricanne, S. Thorpe, and M. Imbert, "Neural processing of stereopsis as a function of viewing distance in primate visual cortical area V1," eng, *Journal of Neurophysiology*, vol. 76, no. 5, pp. 2872–2885, Nov. 1996, ISSN: 0022-3077. DOI: 10.1152/jn.1996.76.5.2872.

[49] Y. Trotter and S. Celebrini, "Gaze direction controls response gain in primary visual-cortex neurons," en, *Nature*, vol. 398, no. 6724, pp. 239–242, Mar. 1999, ISSN: 1476-4687. DOI: 10.1038/18444. [Online]. Available: https://www.nature.com/articles/18444 (visited on 11/15/2021).

[50] C. Galletti and P. P. Battaglini, "Gaze-dependent visual neurons in area V3A of monkey prestriate cortex," eng, *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, vol. 9, no. 4, pp. 1112–1125, Apr. 1989, ISSN: 0270-6474.

[51] F. Bremmer, U. J. Ilg, A. Thiele, C. Distler, and K.-P. Hoffmann, "Eye Position Effects in Monkey Cortex. I. Visual and Pursuit-Related Activity in Extrastriate Areas MT and MST," *Journal of Neurophysiology*, vol. 77, no. 2, pp. 944–961, Feb. 1997, ISSN: 0022-3077. DOI: 10.1152/jn.1997.77.2.944. [Online]. Available: https://journals.physiology.org/doi/full/10.1152/jn.1997.77.2.944 (visited on 11/15/2021).

[52] S. Ben Hamed, W. Page, C. Duffy, and A. Pouget, "MSTd neuronal basis functions for the population encoding of heading direction," eng, *Journal of Neurophysiology*, vol. 90, no. 2, pp. 549–558, Aug. 2003, ISSN: 0022-3077. DOI: 10.1152/jn.00639.2002.

[53] R. A. Andersen, G. K. Essick, and R. M. Siegel, "Encoding of Spatial Location by Posterior Parietal Neurons," *Science*, vol. 230, no. 4724, pp. 456–458, Oct. 1985. DOI: 10.1126/science.4048942. [Online]. Available: https://www.science.org/doi/10.1126/science.4048942 (visited on 11/15/2021).

[54] B. Stricanne, R. A. Andersen, and P. Mazzoni, "Eye-centered, head-centered, and intermediate coding of remembered sound locations in area LIP," eng, *Journal of Neurophysiology*, vol. 76, no. 3, pp. 2071–2076, Sep. 1996, ISSN: 0022-3077. DOI: 10.1152/jn.1996.76.3.2071.

[55] M. Avillac, S. Denève, E. Olivier, A. Pouget, and J.-R. Duhamel, "Reference frames for representing visual and tactile locations in parietal cortex," en, *Nature Neuroscience*, vol. 8, no. 7, pp. 941–949, Jul. 2005, ISSN: 1546-1726. DOI: 10.1038/nn1480. [Online]. Available: https://www.nature.com/articles/nn1480 (visited on 11/15/2021).

[56] A. Battaglia-Mayer, S. Ferraina, A. Genovesio, *et al.*, "Eye-hand coordination during reaching. II. An analysis of the relationships between visuomanual signals in parietal cortex and parieto-frontal association projections," eng, *Cerebral Cortex (New York, N.Y.: 1991)*, vol. 11, no. 6, pp. 528–544, Jun. 2001, ISSN: 1047-3211. DOI: 10.1093/cercor/11.6.528.

[57] D. Boussaoud, T. M. Barth, and S. P. Wise, "Effects of gaze on apparent visual responses of frontal cortex neurons," eng, *Experimental Brain Research*, vol. 93, no. 3, pp. 423–434, 1993, ISSN: 0014-4819. DOI: 10.1007/BF00229358.

[58] C. A. Buneo, M. R. Jarvis, A. P. Batista, and R. A. Andersen, "Direct visuomotor transformations for reaching," eng, *Nature*, vol. 416, no. 6881, pp. 632–636, Apr. 2002, ISSN: 0028-0836. DOI: 10.1038/416632a.

[59] U. Werner-Reiss, K. A. Kelly, A. S. Trause, A. M. Underhill, and J. M. Groh, "Eye Position Affects Activity in Primary Auditory Cortex of Primates," English, *Current Biology*, vol. 13, no. 7, pp. 554–562, Apr. 2003, ISSN: 0960-9822. DOI: 10.1016/S0960-9822(03)00168-4. [Online]. Available: https://www.cell.com/current-biology/abstract/S0960-9822(03)00168-4 (visited on 11/15/2021).

[60] J. M. Groh, A. S. Trause, A. M. Underhill, K. R. Clark, and S. Inati, "Eye position influences auditory responses in primate inferior colliculus," eng, *Neuron*, vol. 29, no. 2, pp. 509–518, Feb. 2001, ISSN: 0896-6273. DOI: 10.1016/s0896-6273(01)00222-7.

[61] F. Saudel and J. Salwan, "Triton: A dynamic symbolic execution framework," in *Symposium sur la sécurité des technologies de l'information et des communications*, ser. SSTIC, Rennes, France, Jun. 2015, pp. 31–54.

[62] M. Carandini, D. J. Heeger, and J. A. Movshon, "Linearity and Normalization in Simple Cells of the Macaque Primary Visual Cortex," en, *Journal of Neuroscience*, vol. 17, no. 21, pp. 8621–8644, Nov. 1997, ISSN: 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.17-21-08621.1997. [Online]. Available: https://www.jneurosci.org/content/17/21/8621 (visited on 11/15/2021).

[63] D. J. Heeger, "Normalization of cell responses in cat striate cortex.," *Visual neuroscience*, vol. 9, no. 2, pp. 181–197, 1992. DOI: 10.1017/S0952523800009640.

[64] D. J. Tolhurst and D. J. Heeger, "Comparison of contrast-normalization and threshold models of the responses of simple cells in cat striate cortex," eng, *Visual Neuroscience*, vol. 14, no. 2, pp. 293–309, Apr. 1997, ISSN: 0952-5238. DOI: 10.1017/s0952523800011433.

[65] H. W. Heuer and K. H. Britten, "Contrast Dependence of Response Normalization in Area MT of the Rhesus Macaque," *Journal of Neurophysiology*, vol. 88, no. 6, pp. 3398–3408, Dec. 2002, ISSN: 0022-3077. DOI: 10.1152/jn.00255.2002. [Online]. Available: https://journals.physiology.org/doi/full/10.1152/jn.00255.2002 (visited on 11/15/2021).

[66] E. K. Miller, P. M. Gochin, and C. G. Gross, "Suppression of visual responses of neurons in inferior temporal cortex of the awake macaque by addition of a second stimulus," eng, *Brain Research*, vol. 616, no. 1-2, pp. 25–29, Jul. 1993, ISSN: 0006-8993. DOI: 10.1016/0006-8993(93)90187-r.

[67] M. Missal, R. Vogels, and G. A. Orban, "Responses of macaque inferior temporal neurons to overlapping shapes," eng, *Cerebral Cortex (New York, N.Y.: 1991)*, vol. 7, no. 8, pp. 758–767, Dec. 1997, ISSN: 1047-3211. DOI: 10.1093/cercor/7.8.758.

[68] G. H. Recanzone, R. H. Wurtz, and U. Schwarz, "Responses of MT and MST Neurons to One and Two Moving Objects in the Receptive Field," *Journal of Neurophysiology*, vol. 78, no. 6, pp. 2904–2915, Dec. 1997, ISSN: 0022-3077. DOI: 10.1152/jn.1997.78.6.2904. [Online]. Available: https://journals.physiology.org/doi/full/10.1152/jn.1997.78.6.2904 (visited on 11/15/2021).

[69] E. T. Rolls and M. J. Tovee, "The responses of single neurons in the temporal visual cortical areas of the macaque when more than one stimulus is present in the receptive field," eng, *Experimental Brain Research*, vol. 103, no. 3, pp. 409–420, 1995, ISSN: 0014-4819. DOI: 10.1007/BF00241500.

[70] S. Treue, K. Hol, and H.-J. Rauber, "Seeing multiple directions of motion—physiology and psychophysics," en, *Nature Neuroscience*, vol. 3, no. 3, pp. 270–276, Mar. 2000, ISSN: 1546-1726. DOI: 10.1038/72985. [Online]. Available: https://www.nature.com/articles/nn0300_270 (visited on 11/15/2021).

[71] D. Zoccolan, D. D. Cox, and J. J. DiCarlo, "Multiple object response normalization in monkey inferotemporal cortex," eng, *The Journal of Neuroscience: The Official Journal of the Society for Neuroscience*, vol. 25, no. 36, pp. 8150–8164, Sep. 2005, ISSN: 1529-2401. DOI: 10.1523/JNEUROSCI.2058-05.2005.

[72] M. A. Basso and R. H. Wurtz, "Modulation of neuronal activity by target uncertainty," en, *Nature*, vol. 389, no. 6646, pp. 66–69, Sep. 1997, ISSN: 1476-4687. DOI: 10.1038/37975. [Online]. Available: https://www.nature.com/articles/37975 (visited on 11/15/2021).

[73] S. R. Olsen, V. Bhandawat, and R. I. Wilson, "Divisive normalization in olfactory population codes," eng, *Neuron*, vol. 66, no. 2, pp. 287–299, Apr. 2010, ISSN: 1097-4199. DOI: 10.1016/j.neuron.2010.04.009.

[74] A. F. Haussler, "DEVELOPMENT OF RETINOTOPIC PROJECTIONS: AN ANALYTIC TREATMENT," en, *J. Theoret. Neurobiol. 2*, pp. 47–73, 1989.